B.C.A. SCIENCE : SEMESTER-IV

OOSE

NILESH MAGAR
UMAKANT SHIRSHETTI
SHREERAM GHOLAP

NIRALI PRAKASHAN

Text Book Of

OOSE

[OBJECT ORIENTED SOFTWARE ENGINEERING]

For

B.C.A. Science : Semester - IV

As Per New Syllabus

Nilesh Magar
M.Sc. (Computer Science)
Lecturer, Computer Science Department
MIT Group of Institutions
MAEER's Arts, Commerce and Science College
Pune

Umakant Shirshetti
M.E. (Computer Network)
H.O.D., Information Technology Department
STES's Sou. Venutai Chavan Polytechnic
Pune

Shreeram Gholap
M.E. (Computer)
Lecturer, Computer Engineering Department
Navsahyadri Institute of Technology
A/p Naigoan,Tal. Bhor, Dist. Pune

Price ₹ 160.00

NIRALI PRAKASHAN
ADVANCEMENT OF KNOWLEDGE

N3879

OOSE (Object Oriented Software Engineering)

First Edition : December 2017

Published By :
NIRALI PRAKASHAN
Abhyudaya Pragati, 1312, Shivaji Nagar
Off J.M. Road, PUNE – 411005
Tel - (020) 25512336/37/39, Fax - (020) 25511379
Email : niralipune@pragationline.com

Printed By :
STAR COPIERS PVT. LTD.
Kumthekar Road, Sadashiv Peth
Pune - 411 030
Tel - (020) 24479201

➢ DISTRIBUTION CENTRES

PUNE

Nirali Prakashan : 119, Budhwar Peth, Jogeshwari Mandir Lane, Pune 411002, Maharashtra
Tel : (020) 2445 2044, 66022708, Fax : (020) 2445 1538
Email : bookorder@pragationline.com, niralilocal@pragationline.com

Nirali Prakashan : S. No. 28/27, Dhyari, Near Pari Company, Pune 411041
Tel : (020) 24690204 Fax : (020) 24690316
Email : dhyari@pragationline.com, bookorder@pragationline.com

MUMBAI

Nirali Prakashan : 385, S.V.P. Road, Rasdhara Co-op. Hsg. Society Ltd.,
Girgaum, Mumbai 400004, Maharashtra
Tel : (022) 2385 6339 / 2386 9976, Fax : (022) 2386 9976
Email : niralimumbai@pragationline.com

➢ DISTRIBUTION BRANCHES

JALGAON

Nirali Prakashan : 34, V. V. Golani Market, Navi Peth, Jalgaon 425001,
Maharashtra, Tel : (0257) 222 0395, Mob : 94234 91860

KOLHAPUR

Nirali Prakashan : New Mahadvar Road, Kedar Plaza, 1st Floor Opp. IDBI Bank
Kolhapur 416 012, Maharashtra. Mob : 9850046155

NAGPUR

Pratibha Book Distributors : Above Maratha Mandir, Shop No. 3, First Floor,
Rani Jhanshi Square, Sitabuldi, Nagpur 440012, Maharashtra
Tel : (0712) 254 7129

DELHI

Nirali Prakashan : 4593/21, Basement, Aggarwal Lane 15, Ansari Road, Daryaganj
Near Times of India Building, New Delhi 110002 Mob : 08505972553

BENGALURU

Pragati Book House : House No. 1, Sanjeevappa Lane, Avenue Road Cross,
Opp. Rice Church, Bengaluru – 560002.
Tel : (080) 64513344, 64513355,Mob : 9880582331, 9845021552
Email:bharatsavla@yahoo.com

CHENNAI

Pragati Books : 9/1, Montieth Road, Behind Taas Mahal, Egmore,
Chennai 600008 Tamil Nadu, Tel : (044) 6518 3535,
Mob : 94440 01782 / 98450 21552 / 98805 82331,
Email : bharatsavla@yahoo.com

Preface...

We take an opportunity to present this book entitled as **"OOSE"** to the students of **B.C.A. Science, Semester-IV** as per the New Syllabus, June 2017-2018.

The book covers theory of Review of Object Oriented Concepts, Introduction to Modeling and UML, Structural Modeling, Structural Diagrams, Behavioral Modeling, Behavioral Diagrams, Architectural Modeling and Case Study.

A special words of thank to Shri. Dineshbhai Furia, Mr. Jignesh Furia for showing full faith in us to write this text book. We also thank to Mr. Amar Salunkhe and Mr. Akbar Shaikh of M/s Nirali Prakashan for their excellent co-operation.

We also thank Ms. Chaitali Takale, Mr. Ravindra Walodare, Mr. Sachin Shinde, Mr. Nilesh Deshmukh, Mr. Ashok Bodke, Mr. Moshin Sayyed and Mr. Nitin Thorat.

Although every care has been taken to check mistakes and misprints, any errors, omission and suggestions from teachers and students for the improvement of this text book shall be most welcome.

Authors

■■■

Syllabus...

1. Review of Object Oriented Concepts (04 L)

1.1 System Concepts - Software Engineering Concepts

1.2 Review of Object Orientation: Classes and Objects, Polymorphism, Inheritance

1.3 Classes and Relationship

1.4 Interfaces

2. Introduction to Modeling and UML (04 L)

2.1 Importance of Modeling

2.2 Principles of Modeling, Object Oriented Modeling

2.3 Overview of UML

2.4 Conceptual Model of UML

2.5 Architecture

3. Structural Modeling (08 L)

3.1 Classes and Advance Classes

3.2 Relationships and Advance Relationships

3.3 Interfaces, Types and Roles

3.4 Packages

3.5 Common Mechanisms

3.6 Diagrams

4. Structural Diagrams (08 L)

4.1 Class Diagrams

4.2 Instances

4.3 Object Diagrams

4.4 Case Study (Minimum Two)

5. Behavioral Modeling (04 L)

5.1 Interactions

5.2 Use Cases

5.3 Events and Signals

5.4 Processes and Threads

5.5 Time and Space

6. Behavioral Diagrams (14 L)

6.1 Use Case Diagrams

6.2 Interaction Diagrams

6.3 Activity Diagrams

6.4 State Machines

6.5 State Chart Diagrams

6.6 Case Study (Minimum Two)

7. Architectural Modeling (08 L)

7.1 Components

7.2 Deployment

7.3 Collaborations

7.4 Pattern and Framework

7.5 Component Diagram

7.6 Deployment Diagram

7.7 Case Study (Minimum Two)

8. Case Study (06 L)

Detail Case Studies for practice including (but not limited) following areas:

ATM / Library Management / Hotel Management / Book Store / Document Editor / College Administration / Railway Reservation / Vending Machine etc.

■■■

Contents...

■■■

CHAPTER 1

Review of Object Oriented Concepts

Contents...

Objectives...

After reading this chapter you will be able,

- To learn Object Oriented Concepts
- To review System and Software Engineering Concepts
- To understand Basic Concepts of Objects, Classes, Interfaces, Relationships, Inheritance etc.

1.1 INTRODUCTION

- The conventional approaches like "waterfall model" may not be very useful due to non-availability of iterations, no provision of reuse and difficulty in incorporating changing requirements. We may also build every software system from the scratch that results into a costly software system.
- An object-oriented paradigm is becoming popular to design, develop and maintain large size, complex and critical software systems.
- Due to its popularity and acceptability in customers, companies/organizations are also releasing the object-oriented versions of their existing software products. Many of the customers expect object-oriented software solutions and request the same for their forthcoming projects.
- The complexity, criticality and size of the software is increasing day by day, and resulting in a situation/condition where the traditional approaches to software development may not be effectively applicable.
- The necessity of developing and maintaining a large size, complex and varied functionalities software system has forced to look for new approaches of software design and development.

- In the traditional approaches, there is a difficulty to produce reusable and low-cost maintainable software. The popular approach to address such situations can be addressed using object-oriented paradigm.
- Hence, the software industry is shifting towards development of software using object-oriented concepts and practices at each phase of software development, and Object Oriented Software Engineering (OOSE) has emerged as a separate and important discipline.
- OOSE emphasizes visual modeling based on real-world objects and entities. Modeling helps in understanding the problems, developing proper documents and producing well-designed programs.
- Modeling produces well-understood requirements, robust designs, and high-quality and maintainable systems. The models must depict the various views of the system and these models must be verified to check whether they capture the customer's requirements.

1.2 SYSTEM CONCEPTS

- The word System is derived from Greek word Systema, which means an organized relationship between any set of components to achieve some common goal or objective.
- A system is defined as, “an orderly grouping of interdependent components linked together according to a plan to achieve a specific goal.”
- A system must have three basic constraints as given below:
  1. A system must have some **structure and behavior** which is designed to achieve a predefined objective.
  2. **Interconnectivity** and **interdependence** must exist among the system components.
  3. The **objectives of the organization** have a **higher priority** than the objectives of its subsystems.
- Some examples of system are traffic management system, payroll system, online library system, human resources information system and so on.

Properties of a System:

1. **Organization** implies structure and order. It is the arrangement of components that helps to achieve predetermined objectives.
2. **Interaction** is defined by the manner in which the components operate with each other.
3. **Interdependence** means how the components of a system depend on one another.
4. **Integration** is concerned with how a system components are connected together.
5. **Central Objective** may be real or stated. It is not uncommon for an organization to state an objective and operate to achieve another.

Elements of a System:

- The Fig. 1.1 shows the elements of a system:

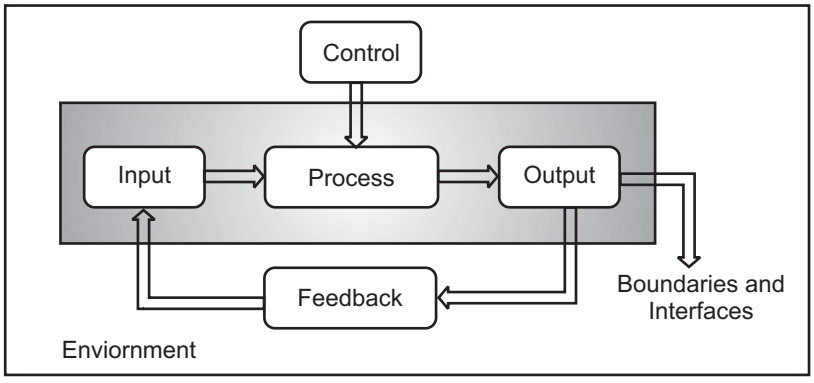

Fig. 1.1: Elements of the System

- **Inputs** are the information that enters into the system for processing. **Output** is the outcome of processing. The **processor (process)** is the element of a system that involves the actual transformation of input into output. Processors are the operational component of a system. The **control** element guides the system and it is the decision-making sub-system that controls the pattern of activities governing input, processing, and output. **Feedback** provides the control in a system. The **environment** is the "supersystem" within which an organization operates. **Boundaries** are the limits that identify its components, processes, and interrelationship when it interfaces with another system.

- Fig. 1.2 shows an example of an system (organization).

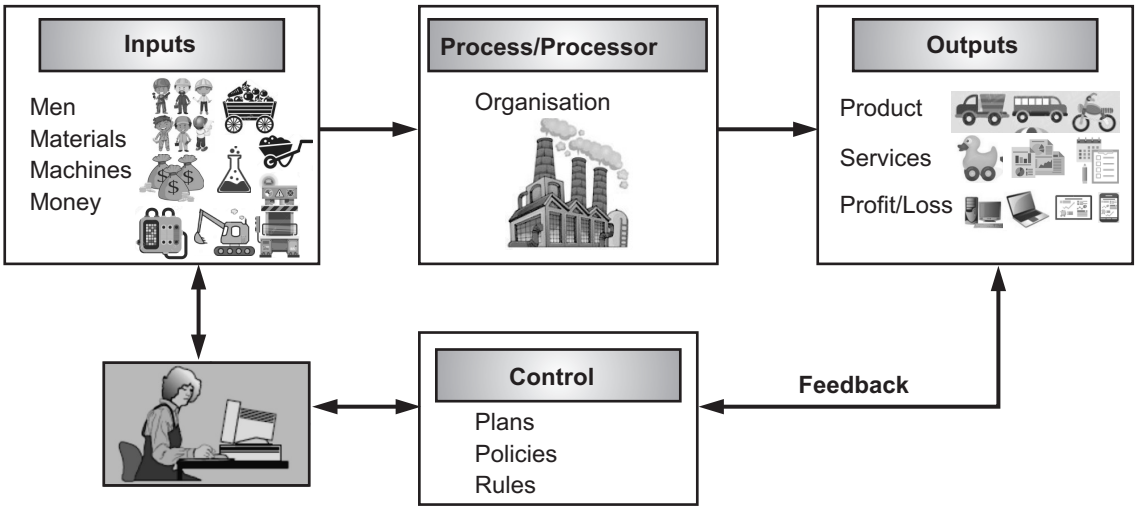

Fig. 1.2: Example of an System (Organization)

1.2.1 Software Engineering Concepts

- The dependency on software is increasing day-by-day and its correct operation is becoming a real challenge for automated modern civilization. In order to handle this challenge, we must have sound engineering principles and processes to produce a good quality software product and such software should also be maintainable.
- The term 'software engineering' is made of following two words:
  - **Software** is more than just a program code. A program is an executable code, which serves some computational purpose. Software is considered to be collection of executable programming code, associated libraries and documentations. Software, when made for a specific requirement is called **software product.**
  - **Engineering** on the other hand, is all about developing products, using well-defined, scientific principles and methods.
- IEEE define software engineering as, "the application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software and the study of these approaches, that is, the application of engineering to software".
- Software engineering is an engineering branch associated with development of software product using well-defined scientific principles, methods and procedures. The outcome of software engineering is an efficient and reliable software product.
- The primary goal of software engineering is to provide the quality of software with low cost. The software engineering discipline helps us to achieve the object and also increases the probability of survival of the software during operation and maintenance.
- The need of software engineering arises because of higher rate of change in user requirements and environment on which the software is working.

- Software engineering can be viewed as a layered technology. Various layers are the **process layer** (allows the development of software on time. It defines an outline for a set of key process areas that must be acclaimed for effective delivery of software engineering technology), the **method layer** (provides technical knowledge for developing software. This layer covers a broad array of tasks that include requirements analysis, design, coding, testing, and maintenance phase of the software development) and the **tools layer** (provides computerized or semi-computerized support for the process and the method layer. Sometimes tools are integrated in such a way that other tools can use information created by one tool. This multi-usage is commonly referred to as Computer-Aided Software Engineering (CASE)).

1.3 REVIEW OF OBJECT ORIENTATION

- In object oriented the term "object" is about the most general word in the English language and can defined as, "a thing presented to or capable of being presented to the senses". In other words, an object is just about anything!
- The word "oriented" is defined as "directed toward", it usually plays the role of casting the term "object oriented" into an adjective. Thus, we have, Object Oriented (OO) "directed toward just about anything you can think of".
- In short, "Object-Oriented" means to organize software as a collection of discrete, real-world objects that incorporate both data structure and behaviour.
- Object orientation is a technique for system modeling. System modeling helps the analyst to understand the functionality of the system.
- In the object-oriented approach, the focus is on capturing the structure and behavior of information systems into small modules that combines both data and process.
- Methodology is a set of methods and principles that signify a systematic way of developing a system using tools, techniques, processes and procedures.
- Object orientation is a technique to model a system. A model is an abstract view of the system. A model is a simplification of reality. Modeling a system using object oriented concepts involves many object interactions with each other.
- Around us lot many objects are there, like people, bus, house, fruits, animals. These objects are some way related to each other. So representing any model involves the objects that we want to represent.
- **For example:** Fig. 1.3 include the real-world objects like Man, Car, Fruit, House etc. which are related with each other. Objects from reality are directly mapped into objects in the model, the semantic gap is minimized.
- A model designed through object oriented, technology has certain features like:
  1. Easy to understand,
  2. Can be related to reality, and
  3. Easy to modify.

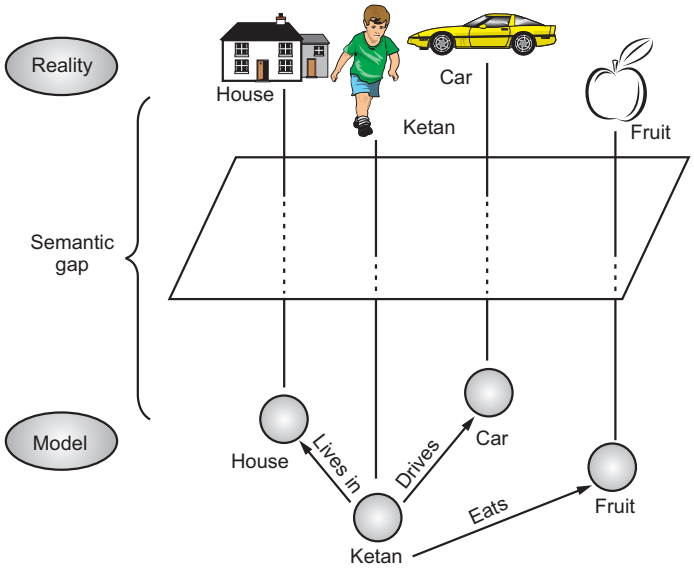

Fig. 1.3: The Real World Objects

Object Orientation Terminology:

- There are many terms available for the object oriented concepts, we have seen and different people use different terms to refer the same thing.
- Table 1.1 lists some common terms used in OO.

Table 1.1: OO Terminology

Sr. No.	Terms	OO Terminology
1.	Object	Objects are separate, distinguish and basic run-time entities.
2.	Classes	A class is a collection of objects of similar type.
3.	Attribute	It is a small piece of information like color, height etc., that describes one characteristic of an object.
4.	Field	It is a named value inside an object.
5.	Operation	It is a piece of code belonging to an object.
6.	Method	It is a synonym for operation.
7.	Instance	An object is an instance of a class.
8.	Message	It is a request sent from one object to another.
9.	Invocation	The carrying out of an operation in response to a message.
10.	Execution	It is a synonym for invocation.
11.	Association	It is a direct or indirect connection between two objects.
12.	Aggregation	It is a strong association implying some kind of part whole hierarchy.
13.	Composition	It is a strong aggregation, where the part is inside exactly one whole - the part may be also be created and destroyed by the whole.
14.	Interface	It is a set of messages understood by an object.
15.	Protocol	It is an agreed way of passing messages over a network.
16.	Behavior	It is a collective term for all of an object's operation.

1.3.1 Objects

- An object is the basic unit of object-orientation approach. An object is a real-world element in an object–oriented environment that may have a physical or a conceptual existence.
- An object is a chunk of structured data in a running software system. Each object has:
  1. **Identity** that distinguishes it from other objects in the system.
  2. **State** that determines the characteristic properties of an object as well as the values of the properties that the object holds.
  3. **Behavior** that represents externally visible activities performed by an object in terms of changes in its state.
- Object can be defined technically as "a combination of data and their associated functions." **OR**
- An object is "an abstraction of something of interest to the program, normally something in the real world".
- An object-oriented model consists of a number of objects; these are clearly delimited parts of the modeled system.
- Any real-world thing can be treated as an object in the OO as shown in Fig. 1.4.

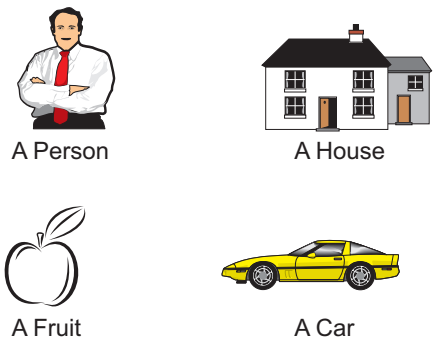

Fig. 1.4: Different Objects

- Operations are the methods that abstract services of objects. An operation is a function that may be applied to or by object in a class.
- In the Fig. 1.5, an object Yash is represented from outside. We are able to see operations, only the properties are hidden in an object oriented system.

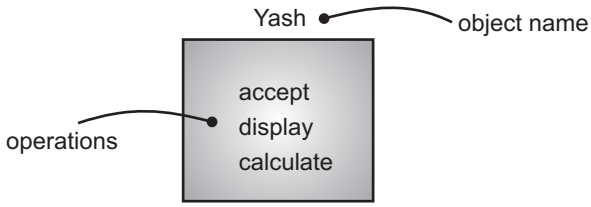

Fig. 1.5: Object Yash and its Operations

- In an object oriented modeling, objects have relations with one another. There are two types of relations namely, static relation and dynamic relation as explained below:
  1. **Static relations** are the relations existing over long period which mean that two objects know existence of each other.
  2. **Dynamic relations** are the relations by which two objects actually communicate with each other.

1.3.2 Classes

- A class represents a template (specification) for several objects and describes how these objects are structured internally.
- A collection of similar types of objects is considered as a class. The objects that have similar behaviour and information structure can be grouped under a class.
- A class serves as a blueprint for its objects. That is, once a class has been defined, any number of objects belonging to that class be created.
- The objects of a class are also known as the instances or the variables of that class and the process of creating objects from a class is known as instantiation.
- A class is defined as, "a user-defined data type, which contains the entire set of a similar data and the functions that the objects possess". **OR**
- Class is defined as, "an abstract data type characterized by a set of properties (attributes and functions) common to its objects".
- Fig. 1.6 shows a real world example of Classes and Objects.
- Fig. 1.7 shows an example of class. The data abstractions (attributes) that describe the class are enclosed by a "wall" of procedural abstractions (called operations, methods, or services) that are capable of manipulating the data in some way.

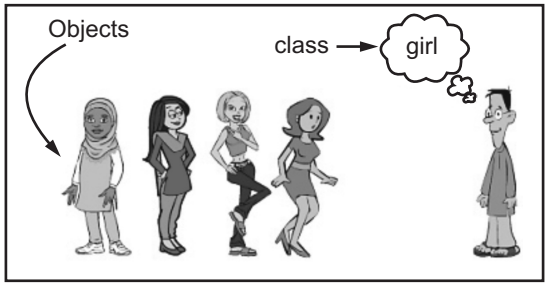

Fig. 1.6: Example of Classes and Objects

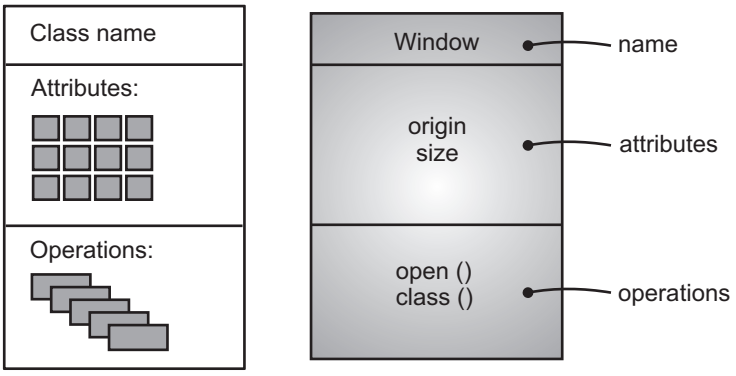

Fig. 1.7: Declaration of Class

Instance:

- An instance of a class is an actual object of that class. In other words, an object is an instance or occurrence of a class.

OOSE (Object Oriented Software Engineering)

Publisher : Nirali Prakashan

Author : Nilesh Magar, Umakant Shirshetti, Shreeram Gholap

Type the URL : http://www.kopykitab.com/product/19657