VTU eNotes On

Object Oriented Modeling & Design

(Computer Science)

OBJECT-ORIENTED MODELING AND DESIGN

Introduction

Contents:

Introduction

Object Oriented Approach is innovative and modern approach of designing the system by focusing primarily on Data elements of the application domain. It differs from the functional/traditional approach by providing features like data hiding, encapsulation and better reuse.

Modeling is not a separate phase but it is involved in every phase of software engineering. Modeling is all about making models/prototypes of the system/situations needed to do better analysis, design, coding and testing

Prerequisite

The course is aimed at:
Students who have prior knowledge to the concept of Software Engineering and any one Object oriented language like C++.

Course Relevance

Object Oriented Modeling and Design is thinking about the problem using models organized around the real world concepts. Earlier to this was the Procedural oriented paradigm. Today's applications have grown to be very Complex. In order to handle this inherent complexity OOMD was framed. Object Oriented Paradigm addresses the problem Domain by considering the problem as a set of related interacting objects.

The modeling task then is specifying, for a specific context, those Objects (or the Class the Objects belongs to), their respective set of Properties and Methods, shared by all Objects members of the Class.

Design Patterns is a general reusable solution to a commonly occurring problem in software design. A design pattern is not a finished design that can be transformed

directly into code. It is a description or template for how to solve a problem that can be used in many different situations

Learning Outcomes:

At the end of the course the student would have the:

- The Knowledge of the basic concepts of Object oriented modeling and Design.
- Will be able to use the Object Oriented notations and process that extends from analysis through design to implementations.
- Be able to use all the standard UML notations.
- Capable to model the requirements with use cases and describe the dynamic behavior and structure of the design.
- Easily create a modular design with components and relate the logical design to the physical environment.
- The Student will be able to use the concept of design patterns and apply it where suitable.

Overview of the syllabus

The syllabus is divided into eight units covering up the topics of Modeling and design patterns.

Unit I:

- What is Object Orientation?
- What is OO development?
- OO themes;
- Evidence for usefulness of OO development;
- OO modeling history.
- *Modeling as Design Technique:* Modeling; abstraction; The three models.
- Class Modeling: Object and class concepts; Link and associations concepts; Generalization and inheritance; A sample class model; Navigation of class models; Practical tips.

1.1 Terminologies:

- What is Object Orientation: Organization of software as collection of discrete objects that incorporate both data structures and behavior.
- Identity: data is quantized into discrete, distinguishable entities called Objects.

- Classification: means objects with the same data structures and behavior are grouped into a class
- Inheritance: is sharing of attributes and operations among classes based on a hierarchical relationship.

- Polymorphism : same operation may behave differently for different classes. (eg move operation)

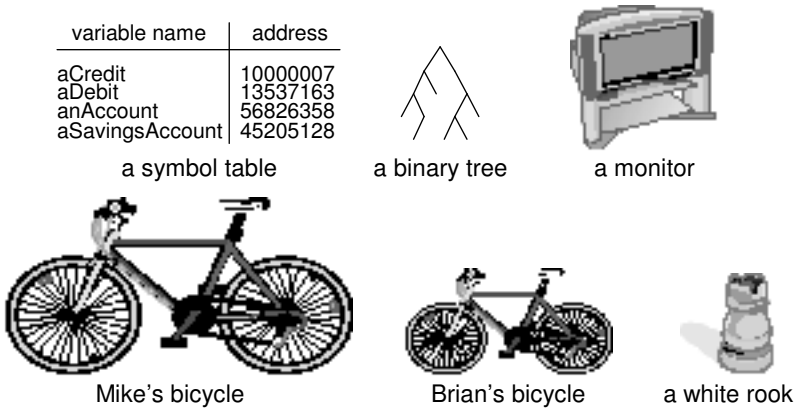

Figure 1: Objects: Objects lie at the heart of object –Oriented Technology

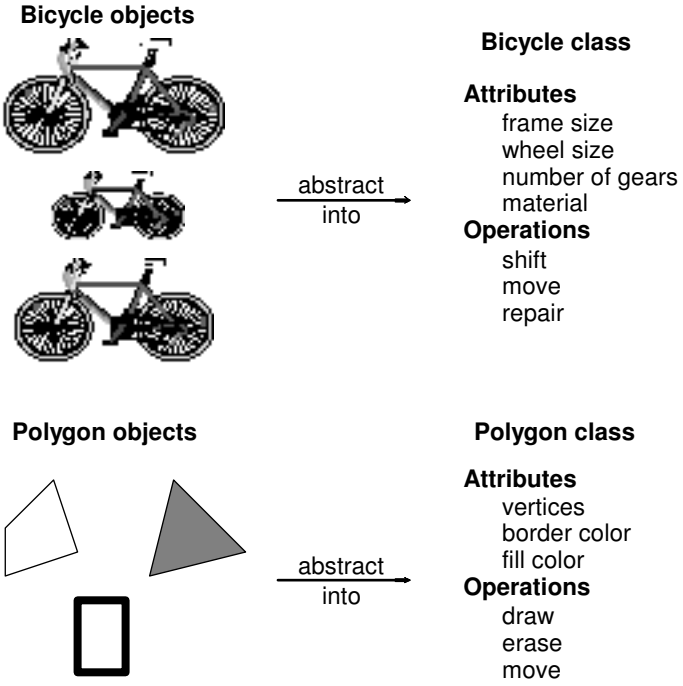

Figure2: Objects and classes. Each class describes a possibly infinite set of individual objects.

1.2. What is Object Oriented Development?

- Development refers to Software Life Cycle: analysis, design and implementation.
- Object oriented Development approach encourages software developers to work and think in terms of the application throughout the software life cycle
- OO Development is a conceptual process independent of programming language until the final stage.
- Greatest Benefit come from helping specifiers, developers and customers express abstract concepts clearly and communicate them to each other.
- It can serve as a medium for specification, analysis, documentation and interfacing as well as for programming

1.2.1 Modeling Concepts, not Implementation

.

- Emphasis here is on modeling and not pertains to any programming language.
- It is a fundamentally a way of thinking in the design perspective and not a programming technique.

1.2.2 Object Oriented Methodology:

The OO methodology has the following stages:

- **System conception:** Software development begins with business analyst or users conceiving an application and formulating tentative requirements.
- **Analysis:** The analysis model is a concise, precise abstraction of what the desired system must do, not how it will be done. The analysis model has two parts: the domain model and the application model.
- **System design.** The development team devises a high level strategy – the system architecture for solving the application problem. The system designer must decide what performance characteristics to optimize, chooses a strategy of attacking the problem and make tentative resource allocation.
- **Class design:** The class designer adds details to the analysis model in accordance with the system design strategy. The focus of the class design is the data structures and algorithms needed to implement each class.
- **Implementation:** Implementers translate the classes and relationships developed during class design into particular programming language, database or hardware. During implementation, it is important to follow good software engineering practice so that traceability to the design is apparent and so that the system remains flexible and extensible.

1.2.3 Three Models:

Three models to describe a system from different viewpoints:

- Class model: Describes the static structure of the objects in a system and their relationships (class diagrams).
- State Model: Describes the aspects of an object that change over time.(State Diagrams.)
- Interaction model: Describes how the objects in a system cooperate to achieve broader results. (Use cases, Sequence diagrams, activity diagrams.)

1.3 Object oriented Themes:

- Abstraction: let's focus on essential aspects of an application while ignoring details.
- Encapsulation :(Information Hiding) separates the external aspects of an object , that are accessible to other objects from internal implementation details.
- Combining data and Behavior.

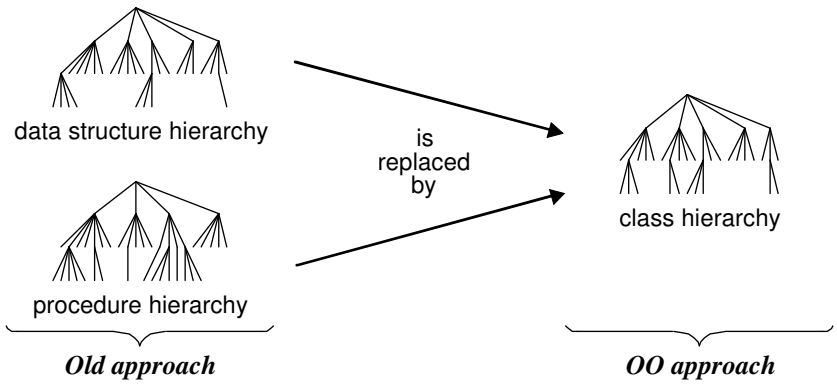

Figure3: OO vs Prior approach. An OO approach has one unified hierarchy for both data and behavior

- Sharing (reuse)
- Emphasis on the essence of an object:OO technology stresses what an object is, rather than how it is used.
- Synergy: Identity, classification, polymorphism, and inheritance characterize OO languages. Use all together.

1.4 Evidence for usefulness of OOD

- Applications at General Electric Research and Development Center.(1990)
- OO techniques for developing compilers, graphics, user interfaces, databases ,an OO language, etc.
- OO technology is a part of the computer science and software engineering mainstream.
- Important forums: (OOPSLA,ECOOP) Object Oriented Programming systems, Languages and applications. European Conference on OOP.
- IEEE Computer and Communications of the ACM.

1.5 OO Modeling history

- Work at GE Rand D led to OMT in 1991.
- Rumbaugh, Grady Booch on unifying the OMT and Booch Notaions in 1994.
- In 1996 the OMG issued a request for the proposals for a standard OO modeling notation.
- 1997 UML was accepted by OMG as a standard.
- In 2001 OMG released UML 1.Added features and released UML 2.)in 2004. www.omg.org

TEXT BOOKS:

1. **Object-Oriented Modeling and Design with UML–**Michael Blaha, James Rumbaugh , 2nd Edition, Pearson Education, 2005.
2. **Pattern-Oriented Software Architecture A System of Patterns, Volume 1 –** Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal John Wiley and Sons, 2006.

OBJECT-ORIENTED MODELING AND DESIGN

Class Modeling

CONTENTS:

Class Modeling:

Captures the static structure of a system by characterizing the objects in the system, the relationships between the objects, and the attributes and operations for each class

1. Object and Class Concepts:

Object: is a concept, abstraction, or a thing with identity that has meaning for an application Eg. two apples) each have identity and are distinguishable.
Class: Describes a group of objects with the same properties(attributes), behavior (operations), kinds of relationship, and Semantics. (*Eg: Person, company, Process and Window*)

Class Diagram: Provide a Graphical notation for modeling classes and their relationships, thereby describing possible objects as shown in figure 1.

Object Diagram: Shows individual Objects and their relationships.

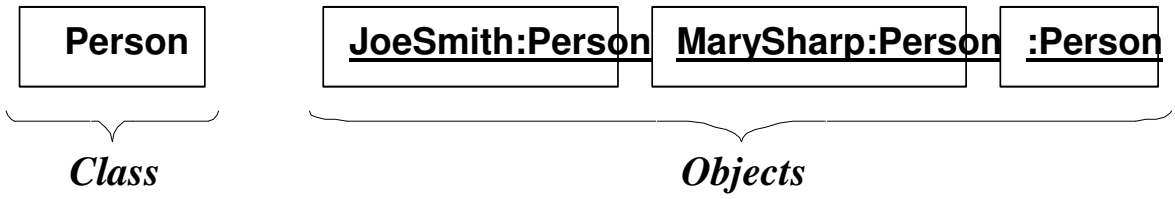

Figure 1: A Class and Objects: Objects and Classes are the focus of Class Modeling

Values: A Value is a piece of data. **Attributes:** named property of class. These elaborate classes as shown in figure 2.

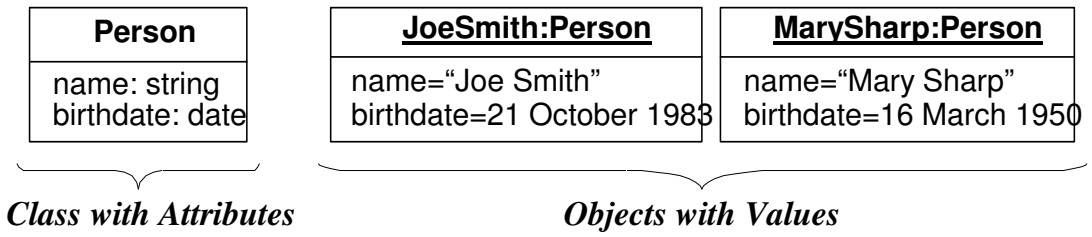

Figure 2: Attributes and Values.

Operations: is a function or procedure that may be applied to or by objects in a class as shown in figure 3.

Methods: is a implementation of an operation for a class.

Polymorphism: same operation applied to many different class.

Signature: all methods must have same signature ie *print* should not have *filename* as argument for one method and *filepointer* as argument for other.

Feature: generic word for either an attribute or operation.

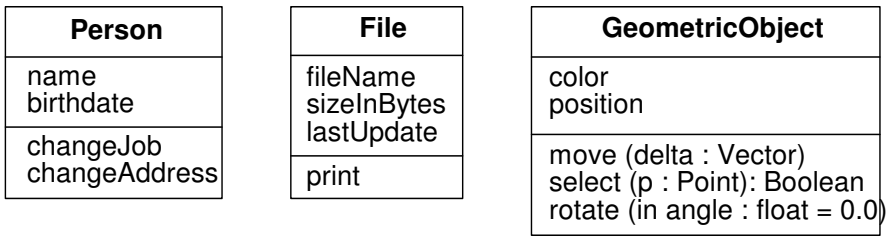

Figure 3 : Operations : function that may be applied to or by objects in a class.

Summary of notations for classes: Figure 4 summarizes the notations for classes.

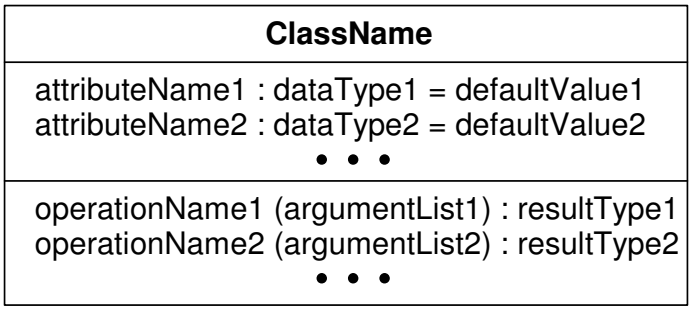

Figure 4: Summary of modeling notations for classes: A Box represents a class and may have as many as three components.

The compartments in the box contain, from top to bottom: class name, list of attributes, and list of operations. Optional details are type, default values, argument list and result type may follow each operation name.

2. Link and Association Concepts:

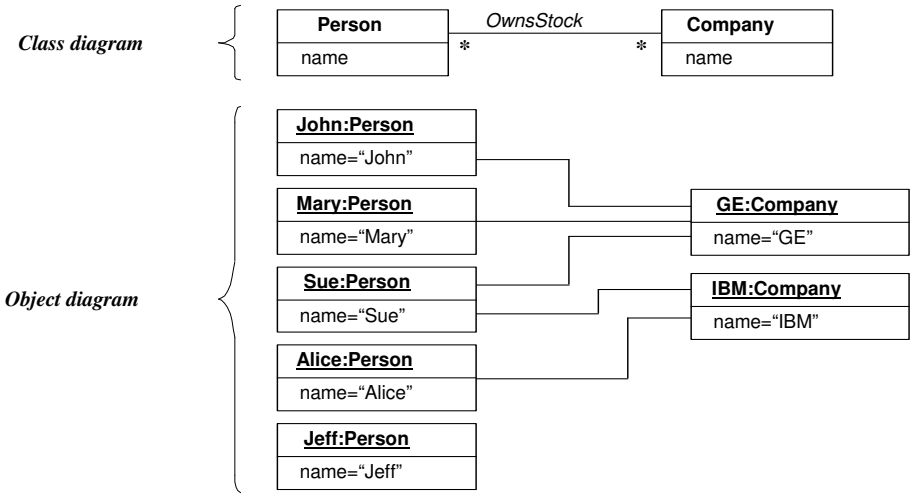

Figure 5: Many to Many Associations

VTU eNotes On Object Oriented Modeling and Design (Computer Science)