OBJECT ORIENTED ANALYSIS AND DESIGN
UNIT IV

Object Oriented Design: Designing Classes, methods – access layer object storage and object interoperability –access layer for the ATM banking system. View layer – designing interface objects – prototyping User interface – view layer for the ATM banking system

1. How to Design Classes? Explain in detail.

The most important activity in designing an application is coming up with a set of classes that works together to provide the needed functionality. Underlying the functionality of any application is the quality of its design.

UML – OCL: UML is a graphical language with a set of rules & semantics in English in form of OCL. Object Constraint Language (OCL) is specification language that uses simple logic for specifying properties of a system. Syntax of some common navigational expressions is shown here

- *Item.selector*: The selector is name of an attribute in item. The result is the value of attribute
- *Item.selector [qualifier–value]*: The selector indicates a qualified association that qualifies the item. Result is related object selected by qualifier, eg., array indexing as form of qualification.
- *Set select (Boolean–expression)*: Boolean expression is written in terms of objects within the set

The Process

During the design phase the classes identified in OOA must be revisited with a shift in focus to their implementation. New classes or attributes & methods are to added for implementation purposes & user interfaces.

The process consists of following activities

1. *Apply design axioms* to design classes, their attributes, methods, associations, structures & protocols. It constitutes two separate steps

⇒ *Refine & complete* the static UML class diagram by adding details. This steps consists of Refine attributes

Design methods & protocols by UML activity diagram to represent methods algorithm Refine associations between classes (if required)

Refine class hierarchy & design with inheritance (if required)

⇒ *Iterate* and refine again

Object oriented design is an iterative process. At each iteration, you can improve the design.

Class visibility

The main objective is designing well defined public, private & protected protocols

Public protocols define the functionality & external messages of an object; private protocols define implementation of an object.

Private Protocol (*visibility*) of class includes messages that normally should not be sent from other objects; it is accessible only to operations of that class.

In protected protocol (visibility), subclasses use method in addition to class itself. *Encapsulation* leakage – is a lack of well designed protocol

The problem of *encapsulation* leakage occurs when details about a class‟s internal implementation are disclosed through the interface.

Refining Attributes

The main goal of this activity is to refine existing attributes (identified in analysis) or add attributes that can elevate the system into implementation.

Attributes Types

The three basic types of attributes are

- Single-value attributes.
- Multiplicity or Multi-value attributes.
- Reference to another object, or instance connection.

Attributes represent the state of an object. When the state of the object changes, these changes are reflected in the value of attributes.

Single value attribute has only one value or state. (Eg). Name, address, salary.

Multiplicity or multivalue attribute can have a collection of many values at any time. (Eg) If we want to keep track of the names of people who have called a customer support line for help.

Instance connection attributes are required to provide the mapping needed by an object to fulfill its responsibilities.

(E.g.) A person may have one or more bank accounts.

A person has zero to many instance connections to Account(s). Similarly, an Account can be assigned to one or more person(s) (joint account). So an Account has zero to many instance connection to Person(s).

UML Attribute presentation: The following is the attribute presentation suggested by UML
Visibility name: type–expression = initial–

value where visibility is one of following
+ public visibility (accessibility to all classes)
# protected visibility (accessibility to subclasses & operations of class)
– private visibility (accessibility only to operations of the class)

Type–expression is language–dependent specification of implementation type of an attribute. Initial value is language–dependent expression for initial value of newly created object and is optional.

2. How to Design Methods and Protocols? Explain in detail.

A class can provide several types of methods:

- Constructor: Method that creates instances (objects) of the class
- Destructor: The method that destroys instances
- Conversion Method: The method that converts a value from one unit of measure to another.
- Copy Method: The method that copies the contents of one instance to another instance
- Attribute set: The method that sets the values of one or more attributes
- Attribute get: The method that returns the values of one or more attributes
- I/O methods: The methods that provide or receive data to or from a device
- Domain specific: The method specific to the application.

Use private and protected protocols to define the functionality of the object. Remember five rules to avoid bad design:

If it looks messy, then its probably a bad design
If it looks too complex, then its probably a bad design
If it is too big, then its probably a bad design
If people don‟t like it, then its probably a bad
design If it doesn‟t work, then its probably a bad
design

Apply design axioms and corollaries to avoid design pitfalls and use UML operation presentation which is similar to syntax of UML attribute representation

UML operation presentation: The following is the operation presentation suggested by UML
Visibility name: (parameter-list): return-type-expression
where visibility is one of following
+ public visibility (accessibility to all classes)
# protected visibility (accessibility to subclasses & operations of class)
– private visibility (accessibility only to operations of the class)
parameter-list: is a list of parameters, separated by commas, each
specified by name: type-expression = default value
return-type-expression: is a language dependent specification of the implementation of the value returned by the method.
Eg: +getName(): aName +getAccountNumber (account: type): account Number

3. Explain briefly the functions of Access Layer in detail. Object Storage and Object Interoperability

Object Storage and persistence:-

A database management system (*DBMS*) is a set of programs that enables the creation maintenance of a collection of related data. The fundamental purpose is to provide a reliable, persistent data storage facility & mechanisms for efficient, convenient data access & retrieval *Persistence* refers to the ability of some objects to outlive the programs that created them. A program will create a large amount of data throughout its execution. Each item of data will have a different life time. Atkinson et al. describe *six broad categories* of life time of data:

- Transient results to evaluation of expressions
- Variables involved in procedure activation (parameters & variables with a localized scope)
- Global variables & variables that are dynamically allocated
- Data that exist between the executions of a program
- Data that exist between the versions of a program
- Data that outlive a program

The first three categories are *transient data*, data that cease to exist beyond lifetime of creating process. The other three are non-transient or persistent data. A file or a database can provide a longer life for objects – longer than duration of process in which they were created. From a language perspective, this characteristic is called persistence. Essential elements in providing a persistent store are

1. Identification of persistent objects or reach ability (*object ID*).
2. Properties of objects & their interconnections. The store must be able to coherently manage non-pointer & pointer data (i.e., *inter-object references*).
3. Scale of the object store. The object store should provide a conceptually infinite store.
4. *Stability*: The system should be able to recover from unexpected failures and return the system to a

recent self–consistent state. This is similar to reliability requirements of a DBMS.

Data Base Management Systems: -

DBMS is a set of programs that enable the creation & maintenance of a collection of related data. They have number of properties that distinguish them from file–based data management approach

A *fundamental characteristic* of database approach is that DBMS contains not only data but complete definition of data formats it manages. This description is known as schema or meta–data containing a complete definition of data formats, such as data structures, types & constraints

Advantage of database approach is that it will provide a generic storage management mechanism. Another one is program – data independence

Database Views: DBMS provides the database users with a conceptual representation that is independent of low–level details (physical view) of how the data are stored. The database an provide an abstract data model that uses logical concepts such as field, records, tables & their interrelationships. DBMS can provide multiple virtual views of data that are tailored to individual applications. This allows convenience of private data representation with advantage of globally managed information

Database Models: It is a collection of logical constructs used to represent the data structure & data relationships within the database. It is grouped into two categories:
Conceptual model is concerned with what is represented in the database & *Implementation model* is concerned with how it is represented. It can be stated as Hierarchical model, Network model, Relational model (*tuples – Primary key & foreign key*) *Database*
Interface: The interface on a database must include a data definition language (*DDL*), query and data manipulation language (*DML*)

- Database systems adopt *two approaches* for interfaces with system. One is to embed a database language (SQL) & other is to extend the host programming language with database constructs
- *Database Schema & DDL*: DDL is the language used to describe structure of & relationships between objects stored in a database. This structure of information is database schema
- *DML & Query Capabilities*: DML is language that allows users to access & manipulate data organization. SQL is standard DML for relational DBMSs. It is widely used for its query capabilities. The

Query usually specifies
- The domain of the discourse over which to ask the query
- The elements of general interest
- The conditions or constraints that apply