Available at **www.ijsred.com**

RESEARCH ARTICLE OPEN ACCESS

Object Oriented UML Modeling: LMS Case Study

Tarun Kumar[*] Dr. Deepak Chaudhary[#]
[*]Lecturer, Government Polytechnic Alapur, Badaun, UP, India
Mobile: 9808705655, Email: tarunkumar124@gmail.com
[#]Assistant Teacher,ShriRajendarsingh Intermediate college,kushinagar, UP
Emailid- deepak.se17@gmail.com

Abstract

In software, there are several ways to approach a model. The two most common ways are from an algorithmic perspective and from an object-oriented perspective The Object-Oriented Modeling assists the programmer to address the complexity of a problem domain by considering the problem not as a set of functions that can be performed but primarily as a set of related, interacting Objects. This article is based on the approach of Object Oriented modeling through Unified Modeling Language (UML) for Library Management System. The main aim of this article is to provide a flexible and faithful environment for a student and library member, who want to do query related to the book and member status. Some of the main features in this software are the capability of searching through the book using various parameters like author, ISBN number, book title etc. In this article UML Class, Interaction diagram, Activity diagram & Use Case diagram are also designed for the Library management System

Keyword: UML, Library Management System, class diagram

1. INTRODUCTION

Structured programming was the mainstream in the earlier days of software engineering. Programmers begandeveloping standard blocks of code to perform operations like printing, and then copied and pasted that codeinto every application they wrote. While this reduced the development time for new applications, it wasdifficult if a change was needed in that block of code, because the developer had to make the changeeverywhere that code had been copied. Structured programming presented some challenges for whichobject–orientedprogramming was designed to solve.With object–oriented programming, developers create blocks of code, called objects. These objects are thenused by the various applications. Should one of the objects require modification, a developer needs to makethe change only once. Companies are rushing out to adopt this technology and integrate it into their existingapplications. In fact, most applications being developed today are object–oriented. Some languages, such asJava, require an object–oriented structure. But what does it mean?

The object–oriented paradigm is a different way of viewing applications. With the object–oriented approach,you divide an application into many small chunks, or objects, that are fairly independent of one another. Youcan then build the application by piecing all of these objects together. Think of it as building a castle out ofblocks. The first step is to make or buy some basic objects, the different types of blocks. Once you have thesebuilding blocks, you can put them together to make your castle. Once you build or buy some basic objects inthe computer world, you can simply put them together to create new applications[1]

2. FEATURES OF LIBRARY MANAGEMENT SYSTEM.

TheLibrary Management System is intended to replace the manual model of issue the book.The target user of this system is any person who wants to return and issue with the prerequisite that they should have a valid roll number from the college.

2.1 Description and Priority

A user whose identity has been verified will be able to log on to the Library management system and issue the book whatever he wants and whatever is available on the site. His verification data will be added to the database which will be used for further references.

2.2 Stimulus/Response Sequences

Stimulus: User opens the site.
Response: System displays a login screen where the user will be required to enter the username and password. If he is not a member, then he will need to register first.
Stimulus: User clicks on the login button present on the login screen.
Response: If the user is a member, then the system logs him in and if he is not, then the system will ask the user to register first.
Stimulus: User logs on to the site.

Response: The system shows a list of products available on the site.
Stimulus: User enters a particular ISBN or author name in the search box and hits enter.
Response: The system shows the requested item if it is available on the site.

2.3 Functional Requirements

The system logs the user in after verifying their identity.The system creates also creates a new member registration form for the user if he is not a member.The system provides the user with a list of book available on the site.The user can also search for a particular product on the site.The system allows the user to add book to the library.

3. OBJECT-ORIENTED ANALYSIS

In object oriented analysis UML 1.0 divided into two categories[2]
(a). Structural diagram.
(b). Behavioral diagram.
Structural diagram gives the detail of the static aspect of any system. It has five UML diagram namely class diagram, object diagram, use case diagram, deployment diagram and component diagram. Behavioral diagram gives the detail about the dynamic aspect of any system. It has four diagrams namely activity diagram, sequence diagram, collaboration diagram. State chart diagram. Object-oriented analysis looks at the problem domain, with the aim of producing a conceptual model of theinformation that exists in the area being analyzed.Analysis models do not consider any implementationconstraints or how the system is to be built. The identifiedobjects reflect entities and operations that are associatedwith the problem to be solved.

3.1. UML static modeling for Library Management System

This part describes the way that system should look. It analyses the structure and substructure of the modeled system based on objects, attributes, operations and relationships[3].

3.1.1. Use case modeling for Library Management System

Use cases and actors define the scope of the system you are building. Use cases include anything that is withinthe system; actors include anything that is external to the system. We'll discuses someof the fundamental concepts of use case, or system, modeling: use case, actor, association relationship, includes relationship, extends relationship, generalization relationship, flow of events, activity diagram, and Use Case diagram
A Use Case diagramshows you some of the use cases in your system, some of the actors in your system, and the relationships between them. As you know, *a use case is* a high–level piece of functionality that the system will provide. An actor is anyone or anything that interacts with the system being built

3.1.2. Class Diagram

Class diagrams model one aspect of the system, the composition of classes. Given that class definitions are static when the system is in use, class diagrams model a static perspective of the system. A given system may have many class diagrams, portraying various structural aspects of the system. The objective of the class diagram is to portray the elements that are part of a class and the essential relationships that exist between classes.

The class diagram models the classes that comprise a system. The relationships that exist between classes must also be shown in class diagrams. The three possible relationships between classes in class diagrams are association, generalization and dependency and eachis represented by a different type of line and arrow[4].

3.1.3. Object Diagram

An object diagram models a set of objects and their interrelationships during a system snapshot. A system snapshot is the state of the software system at a selected moment in time. The notational elements comprising object diagrams are objects and links. Recall that an object is an instance of a class. In a class diagram, two classes may be related to each other via an association. In an object diagram, instances of two classes, called objects, are related to each other via a link, which is simply an instance of the association relationship[5].

In this object diagram, a particular Patron object of type Student currently has three Resource objects, all of type Book, checked out. The object diagram shows that the student contains a List object which lists the Resources currently checked out by the Patron. Unlike other diagram types, an object diagram may contain multiple instances of a particular class. The representation of multiple objects shows that more than one instance of a class may be allocated at a particular point during execution. This information helps the developers of the system determine cardinality on the relationships between classes[6].

3.2. Dynamic Modeling for Library Management System

This section describes the way that the system should work. It analyses the system behavior, including sequence and collaboration diagrams, activity diagram, and state diagram.

3.2.1 Activity Diagrams

Activity diagram is used here for modeling the dynamic aspects of Library Management System. This diagram is showing the flow of control from activity. As in the diagram it is showing that how the control flows from one activity user login process to user validation and updating of various information in database, checking the status of items in the library at the time of issuing them.

An activity diagram is basically a projection of the elements found in an activity graph, special case of state machine in which all or most states are activity states and in which all or most transitions are triggered by completion of activities in the source state.A UML activity diagram offers rich notation to show a sequence of activities. It may be applied to any purpose (such as visualizing the steps of a computer

Algorithm), but is considered especially useful for visualizing business workflows and processes, or use cases. One of the UP workflows (disciplines) is Business Modeling; its purpose is to understand and communicate "the structure and the dynamics of the organization in which a system is to be deployed" [RUP]. A key artifact of the Business Modeling discipline is the Business Object Model (a superset of the UP Domain Model), which essentially visualizes how a business works, using UML class, sequence, and activity diagrams. Thus, activity diagrams are especially applicable within the Business Modelingdiscipline of the UP

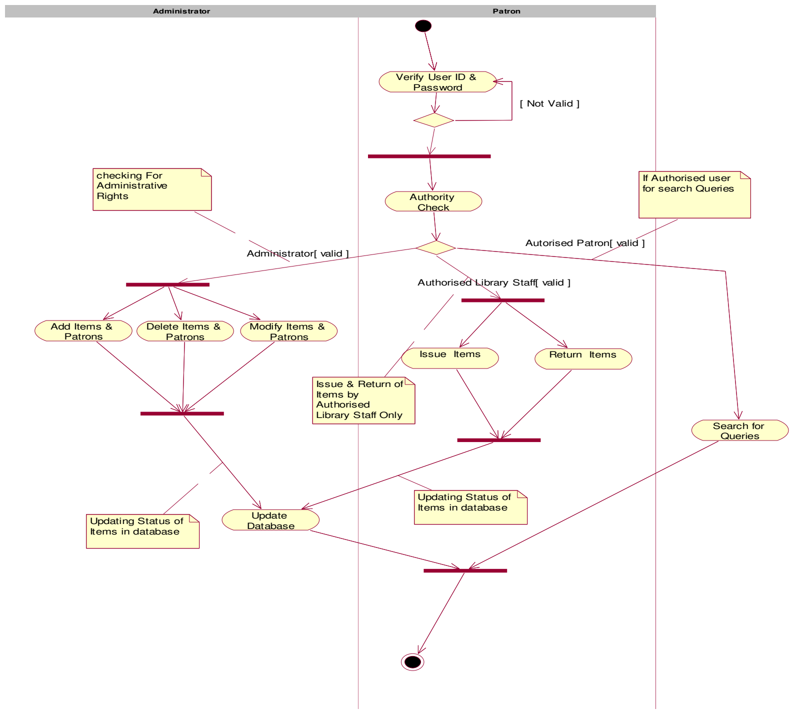

Administrator
Patron
Verify User ID & Password
[Not Valid]
checking For Administrative Rights
Authority Check
If Authorised user for search Queries
Administrator[valid]
Autorised Patron[valid]
Authorised Library Staff[valid]
Add Items & Patrons
Delete Items & Patrons
Modify Items & Patrons
Issue Items
Return Items
Issue & Return of Items by Authorised Library Staff Only
Search for Queries
Updating Status of Items in database
Update Database
Updating Status of Items in database

Fig 1: Activity diagram

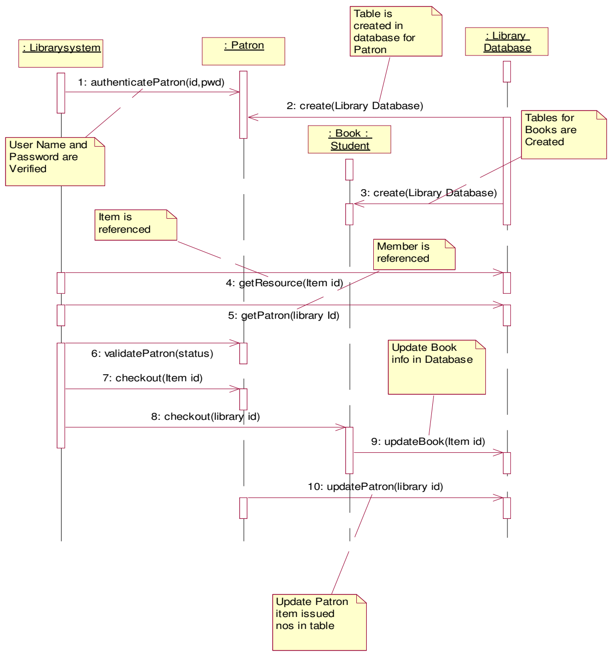
3.2.2. Interaction Diagrams

An Interaction diagramshows you, step by step, one of the flows through a use case: what objects areneeded for the flow, what messages the objects send to each other, what actor initiates the flow, and what order the messages are sent [8].

The two types of Interaction diagrams we'll talk about are Sequence diagrams and Collaboration diagrams. A Sequence diagram is ordered by time.

A sequence diagram, like a collaboration diagram, shows the messages that are passed between objects during the execution of a particular scenario. Unlike a collaboration diagram, a sequence diagram specifies the time ordering of messages. Figure on next page shows a sequence diagram modeling the CheckOutResource use case. Time is represented vertically in the diagram. Thus, the first message generated is getResource(ResourceID) sent from the LibrarySystem object to the LibraryDatabase, and the last message generated is update(Patron) sent from the Patron object to the LibraryDatabase.

The diagram shows that an external actor, the LibraryStaff, assists the Patron by entering the Patron identification number, via the enterPatron() method, and the Resource identification number, via the enterResource() method, into the actual LibrarySystem. The LibrarySystem sends information, again via method invocations, to the Patron object to determine whether the Patron is valid and to store the Resource information in the Patron object. In addition, the LibrarySystem sends messages to the particular Resource object (in this case, a Book) as well as to the Library Database.


: Librarysystem
: Patron
Table is created in database for Patron
: Library Database
1: authenticatePatron(id,pwd)
2: create(Library Database)
Tables for Books are Created
: Book : Student
User Name and Password are Verified
3: create(Library Database)
Item is referenced
Member is referenced
4: getResource(Item id)
5: getPatron(library Id)
6: validatePatron(status)
Update Book info in Database
7: checkout(Item id)
8: checkout(library id)
9: updateBook(Item id)
10: updatePatron(library id)
Update Patron item issued nos in table

LIBRARY MANAGEMENT SYSTEM SEQUENCE DIAGRAM

Fig 2: Sequence Diagram

A Collaboration diagram shows the same information, but is organized differently. Although a Sequence diagram and a Collaboration diagram show you the same information, there are a couple of differences between these two diagrams. Sequence diagrams can show a focus of control; Collaboration diagrams can show a data flow.

We'll talk about these differences when discussing Interaction diagrams contain a lot of the same detail that is spelled out in the flow of events, but here the information is presented in a way that is more useful to the developers. While the flow of events focuses on what the system needs to do, Sequence and These diagrams focus on the objects that will be created to implement the functionality spelled out in the use cases. Sequence and Collaboration diagrams can show objects, classes, or both.Collaboration diagrams help to define how the system will do it.

4. OBJECT ORIENTED DESIGN

4.1. Component Diagram

Components are the physical replaceable parts of the system that conforms to and provides the realization of a set of interfaces. Components live in the material world of bits and therefore are an important building block in modeling the physical aspect of the system [9].

In the component diagram for Library Management system various components are shown such as Database, tables, application exe, imported packages such as package forSQL and package for input and output, package for applets etc.

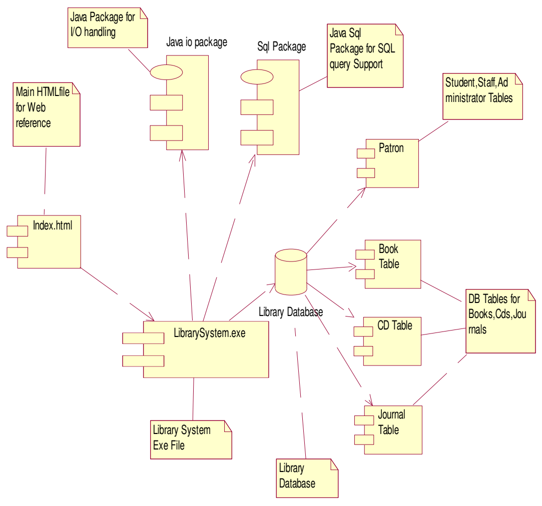

Java Package for I/O handling
Java io package
Sql Package
Java Sql Package for SQL query Support
Student,Staff,Administrator Tables
Main HTMLfile for Web reference
Patron
Index.html
Book Table
Library Database
DB Tables for Books,Cds,Journals
LibrarySystem.exe
CD Table
Journal Table
Library System Exe File
Library Database

Fig 3:Component Diagram

4.2. Deployment Diagram

Deployment diagram shows the configuration of the run time processing nodes and the components thatThe deployment diagram for Library Management System is shown on the next page. In this diagramlive on that. A deployment addresses the static deployment view of the system.Various processing nodes are shown such as Database server which is responsible for providing database query services,and other server is web server which is J2EE server, two type of clients one is remote which sends/receives HTTP request/response, while localclient uses TCP/IP connection for communication with J2EE server. Here J2EE server is working as an intermediary between Database server and various clients.A Deployment diagram shows all of the nodes on the network, the connections between them, and the processes that will runon each one.

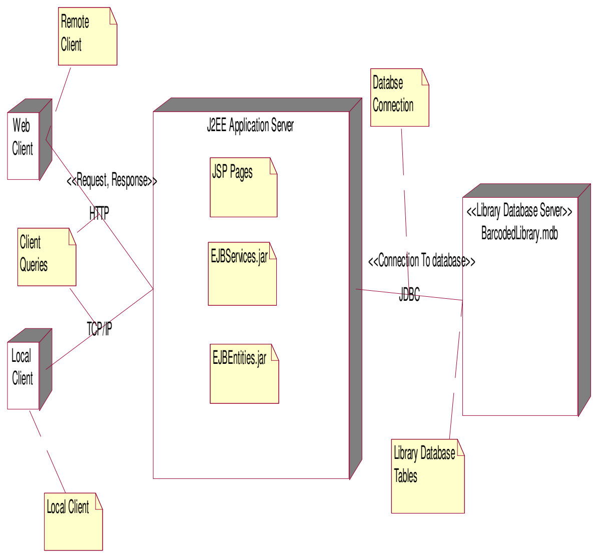


Fig 4: Deployment Diagram

4.3 state chart diagram.

UML state machine diagrams depict the dynamic behavior of an entity based on its response to events, showing how the entity reacts to various events depending on the current state that it is in.

This state chart diagram is showing the dynamic aspect of Library Management System. This is modeling that how the Library Management system will react to various input events. It shows that how the system behaves to the outside events.

This Library Management System state chart diagram shows the behavior that specifies the sequences of states, an object goes through during itslife time in response to events together it's response to those events [10].

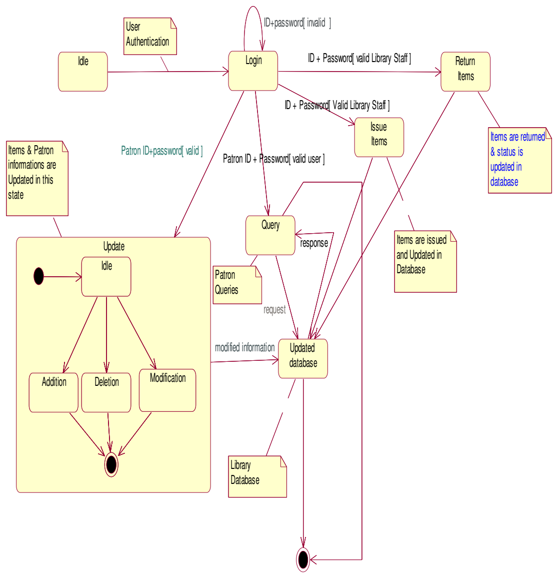

User Authentication
ID+password[invalid]
Idle
Login
ID + Password[valid Library Staff]
Return Items
ID + Password[Valid Library Staff]
Issue Items
Items are returned & status is updated in database
Items & Patron informations are Updated in this state
Patron ID+password[valid]
Patron ID + Password[valid user]
Query
response
Items are issued and Updated in Database
Update
Idle
Patron Queries
request
modified information
Updated database
Addition
Deletion
Modification
Library Database

Fig 5: State Diagram

5. CONCLUSIONS

This article has investigated system requirements, use-case diagrams and the UML model for the Library Management System. Such diagrams were designed to support the system designers and developers, to help customers in observing a software system from various perspectives, and to improve their understanding of cohesion and abstraction in designing a system or software. The requirements produced are sufficiently detailed to form the basis for the development of an information system using both the practical analysis of use cases and the conceptual model for UML. The analysis of businessand user needs, system requirements, and use cases are essential steps for building a system based on Unified Modeling Language. From the above work it is concluded that the UML modeling is a powerful language used to design for the software research problems. In this article complete modeling is done for Library Management System which is efficient & useful for the software developer to convert the above model through Object Oriented language. UML could be adopted for knowledge modeling as well. While UML in its current state has its limitations, it is an extensible language and thus can be used to support the knowledge modeling activity through the profiles mechanism.

6. REFERENCE

[1] BoochG.,Rumbaugh J., JacobsonI. , TheUnified Modelling Language User Guide,AddisonWesley, Reading,1999, vol 2, pp-23-36.
[2] Booch, G., Object-Oriented Analysis and Designwith Applications, Addison-Wesley1994, vol 2, pp-456-466.
[3] Blaha,M., RumbaughJ., Object-OrientedModeling and Design with UML,Upper Saddle River, New Jersey, Prentice Hall,(2005), vol 2, pp 34-45.
[4] Penedo, M., Ploedereder, E., and Thomas, I., Object Management Issues for Software Engineering Environments. SIGPLAN Notices , 1988, vol. 24(2).
[5] Reenskaug, T., and Skaar, A, An Environment for Literate Smalltalk Programming. SIGPLAN Notices, 1989, vol. 24(10).
[6] Erman, L., Lark, J., and Hayes-Roth, ABE: An Environment for Engineering Intelligent Systems. IEEE Transactions on Software Engineering , 2005, vol. 14(12)
[7] Anh T. L., "A UML Model of the ClientTracking System", 18th Australasian Conference onInformation system,5-7 Dec 2007, Toowoomba.
[8] RumbaughJ.,"Object Oriented Modeling and Design", PHI, 2000 vol 3.
[9] ChakiS,Rajamani S. K., and RehofJ., Types as Models: Model Checking Message-Passing Programs, POPL, ACM SIGPLAN Notices,2002, vol. 37, no. 1, pp. 45-57.