OBJECT ORIENTED ANALYSIS AND DESIGN
LABORATORY MANUAL

Subject Code : 15A05509
Regulations : JNTUA – R15
Class : V Semester (CSE)

CHADALAWADA RAMANAMMA ENGINEERING COLLEGE
(AUTONOMOUS)
Chadalawada Nagar, Renigunta Road, Tirupati – 517 506
Department of Computer Science and Engineering
INDEX

<table>
<thead>
<tr>
<th>S. No</th>
<th>Name of the Experiment</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>UML diagrams to be developed are:</td>
</tr>
<tr>
<td>1</td>
<td>Use Case Diagram.</td>
</tr>
<tr>
<td>2</td>
<td>Class Diagram.</td>
</tr>
<tr>
<td>3</td>
<td>Sequence Diagram.</td>
</tr>
<tr>
<td>4</td>
<td>Collaboration Diagram.</td>
</tr>
<tr>
<td>5</td>
<td>State Diagram.</td>
</tr>
<tr>
<td>6</td>
<td>Activity Diagram.</td>
</tr>
<tr>
<td>7</td>
<td>Component Diagram.</td>
</tr>
<tr>
<td>8</td>
<td>Deployment Diagram.</td>
</tr>
<tr>
<td>9</td>
<td>Test Design</td>
</tr>
</tbody>
</table>

Problems that may be considered are

|       |                                      |
| 1     | College Information System           |
| 2     | Hostel Management System             |
| 3     | ATM System                            |
INTRODUCTION:

USECASE DIAGRAM:
A behavioral diagram that shows a set of use cases and actors and their relationships.

USECASE:
It is a description of set of sequence of actions that a system performs that yields an observable result of value to a particular actor.
An use case is used to structure the behavioral things in a model and it is rendered as an ellipse with solid line along with its name.

Actor:
It specifies a coherent set of roles that users of use cases play when interacting with these use cases.

CLASS DIAGRAM:
A structural diagram that shows a set of classes, interfaces, collaborations, and their relationships.

OBJECT DIAGRAM:
A structural diagram that shows a set of objects and their relationships.

SEQUENCE DIAGRAM:
A behavioral diagram that shows an interaction, emphasizing the time ordering of messages.

COLLABORATION DIAGRAM:
A behavioural diagram that shows an interaction, emphasizing the structural organization of the objects that send and receive messages.

STATECHART DIAGRAM:
A behavioural diagram that shows a state machine, emphasizing the event-ordered behavior of an object.

ACTIVITY DIAGRAM:
An activity diagram represents the execution state of a mechanism as a sequence of steps grouped sequentially as parallel control flow branches. It is a variant of state chart diagrams organized according to actions and internal behavior of a method or a usecase.

Activity diagrams are used to model the dynamic aspects of a system.

- Swimlanes
- Forking
- Joining

**COMPONENT DIAGRAM:**

Component diagrams are basically used to model static view of the system. This can be achieved by modeling various physical components like libraries, tables, files etc. which are residing within a node.

Component diagrams are very essential for constructing executable systems. This can be done using concepts of forward and reverse engineering. The graphical representation of a component diagrams basically include collection of vertices and arcs.

**DEPLOYMENT DIAGRAM:**

Deployment is the stage of development that describes the configuration of the running system in a real time environment. For deployment, decisions should be made about configuration parameters, performance, resource allocation, distribution and concurrency. The component developed or reused should be deployed on some set of hardware for execution. Nodes are used to model the topology of the hardware on which the system executes. A node usually represents a processor or a device on which components can be deployed.
EXPERIMENT NO: 1

COLLEGE INFORMATION SYSTEM

AIM:

To develop a college Information System

PROBLEM ANALYSIS AND PROJECT PLANNING

A college Information System (SIS) is a software application for educational establishments to manage COLLEGE data. COLLEGE Information System provide capabilities for entering student test and other assessment scores, building schedules, tracking student attendance, and managing many other COLLEGE-related data needs in a school, college or COLLEGE.

PROBLEM STATEMENT

a. Effective for Administration Purpose
b. Cheap
c. Better Service

UML DIAGRAMS:

The following UML diagrams describe the process involved in the online recruitment system

a. Use case diagram
b. Class diagram
c. Sequence diagram
d. Collaboration diagram
e. Activity diagram
f. Component diagram
g. Data Flow Diagram
h. Deployment Diagram
1. USE CASE DIAGRAM:

A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. It is represented using ellipse. Actor is any external entity that makes use of the system being modeled. It is represented using stick figure.

For Administrator:

```
For Administrator:

Admin Login
Add Student
Delete Student
Update Student Records
```

![Diagram](image-url)
DOCUMENTATION OF USE CASE DIAGRAM
The actors in this use case diagram are Admin, Student, Database. The use cases are the activities performed by actors.

a. Admin register login, and store the student records details in database.
b. Student Register from the Student Login process.
c. Then the database is searched for details and verified.
d. Database stores the details and returns acknowledgement

2. CLASS DIAGRAM: A class diagram in the unified modeling language (UML) is a type of static structure diagram that describes the structure of a system by showing the system’s classes, their attributes, and the relationships between the classes. It is represented using a rectangle with three compartments. Top compartment have the class name, middle compartment the attributes and the bottom compartment with operations.
DOCUMENTATION OF CLASS DIAGRAM

This class diagram has three classes Login, Student details and Update details in database.

a. **Students** – is the class name. Its attributes are name, Address, DOB, Gender, College, Subjects, Semester, Year, Degree, Branch. The operations performed in the students class, Store database and Update.

b. **Administration** – is the class name. Its attributes are Login, Password and database. The operations performed are Student Details store in database and send acknowledgement.

c. **Database** – is the class name. The operations performed are storing Search and storing the values.
3. SEQUENCE DIAGRAM:
A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. There are two dimensions.

1. Vertical dimension - represent time.
2. Horizontal dimension - represent different objects.

For Validity:

For Administrator:
For Student:

DOCUMENTATION OF SEQUENCE DIAGRAM:

The sequence diagram describes the sequence of steps to show

a. The Admin login and registering for add Student Details.
   b. The verification done by the interface and sending acknowledgement for registration.
   c. Searching the database with login and displaying it for maintenance.

4. COLLABORATION DIAGRAM:

A collaboration diagram, also called a communication diagram or interaction diagram. A sophisticated modeling tool can easily convert a collaboration diagram into a sequence diagram and the vice. A collaboration diagram resembles a flowchart that portrays the roles, functionality and behavior of individual objects as well as the overall operation of the system in real time.
For Validity:

1: Request for validity()
2: Check for validity()
3: Validate()
4: Successful validate()

For Administrator:

1: Request for validity()
2: Check for validity()
3: Validate()
4: Successful validate()
5: Enter details for adding a student()
6: Add()
7: Successful()
8: Successfully added()
9: Enter credentials for update student record()
10: Update()
11: Successful()
12: Successfully Updated()

Admin/Student

Database

Login
For Student:

**DOCUMENTATION OF COLLABRATION DIAGRAM**

The collaboration diagram is to show how the Student registers and the authorities maintains the details of the registered students in the Information system. Here the sequence is numbered according to the flow of execution.

5. **ACTIVITY DIAGRAM:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of
control. An activity is shown as an rounded box containing the name of the operation.

DOCUMENTATION OF ACTIVITY DIAGRAM

This activity diagram flow of stepwise activities performed in recruitment system.

a. The student details are Add and stored in database.
b. Select the course from the given Course by student.
c. Search Profile and Result with login and if data present in the database.
d. The searched data is displayed if available and then Log Out.
6. COMPONENT DIAGRAM:

The component diagram's main purpose is to show the structural relationships between the components of a system. It is represented by boxed figure. Dependencies are represented by communication association.

**DOCUMENTATION OF COMPONENT DIAGRAM**

The main component in this component diagram is Student Information system. And register, User and Manage, Request details are the components comes under the main component.
7. DEPLOYMENT DIAGRAM:

A deployment diagram in the unified modeling language serves to model the physical deployment of artifacts on deployment targets. Deployment Diagrams show "the allocation of artifacts to nodes according to the Deployments defined between them. It is represented by 3-dimentional box. Dependencies are represented by communication association.

DOCUMENTATION OF DEPLOYMENT DIAGRAM

The processor in this deployment diagram is the COLLEGE Information System which is the main part and the Student are the Admin, verify and search which are the some of the main activities performed in the system.
Case Study 1: Library Application

1. **Problem Statement:**

A lightweight set of features for the first version of the library application might look like this:

It is a support system for a library. The library lends books and magazines to borrowers, who are registered in the system, as are the books and magazines. The library handles the purchase of new titles for the library. Popular titles are bought in multiple copies. Old books and magazines are removed when they are out of date or in poor condition. The librarian is an employee of the library who interacts with the customers (borrowers) and whose work is supported by the system. A borrower can reserve a book or magazine that is not currently available in the library, so that when it’s returned or purchased by the library, that borrower is notified. The reservation is cancelled when the borrower checks out the book or magazine or through an explicit canceling procedure. The librarian can easily create, update, and delete information about the titles, borrowers, loans, and reservations in the system.

The system can run on all popular Web browser platforms (Internet Explorer 5.1+, Netscape 4.0+, and so on). The system is easy to extend with new functionality.

2. **Identification of actors and use cases:**

The use cases in the library system are as follows:

- Login
- Search
- Browse
- Make Reservation
- Remove Reservation
- Checkout Item
- Return Item
- Manage Titles
- Manage Items
- Manage Borrowers
- Manage Librarians
- Assume Identity of Borrower
The outline of the basic flow for the use case Checkout Item (which means that a Borrower can check out an Item) is described as follows:

1. The borrower chooses to perform a “Search” for desired titles/
2. The system prompts the borrower to enter Search criteria.
3. The borrower specifies the search criteria and submits the search.
4. The system locates matching titles and displays them to the borrower.
5. The borrower selects a title to check out.
6. The system displays the details of the title, as well as whether or not there is an available item to be checked out.
7. The borrower confirms that he or she wishes to checkout the item.
8. The system checks out the item.
9. Steps 1 to 8 can be repeated as often as desired by the borrower.
10. The borrower completes checkout.
11. The system notifies a librarian that the borrower has concluded the checkout item session and displays instructions for the borrower to collect the contents.
3: Identification of actors and use cases:

Fig: A use-case diagram for library system

Class diagram for library system: Fig:
classes for the library system
classes for the library system.
Sequence diagram for use case Return Item:

1: specify criteria()
2: search()
3: search(criteria)
4: get matching(criteria)
5: select title()
6: get item(title)
7: get item()
8: select item()
9: return()
10: return(item)
11: can return(item)
12: [borrower is allowed to return this item]
13: set status()
Fig: collaboration diagram for use case Return Item

1: specify criteria()
2: search()
3: search(criteria)
4: get matching(criteria)
5: select title()
6: get item(title)
7: get item()
8: specify criteria()
9: return()
10: return(item)
11: can return (item)
12: [borrower is allowed to return this item]
13: set status()
Sequence diagram for use cases:

Checkout Item:

1: specify criteria()
2: search()
3: search(criteria)
4: get matching(criteria)
5: select title()
6: get items(title)
7: get items()
8: select item()
9: checkout()
10: checkout(item)
11: can checkout(item)
12: checkout(item)
13: set status()
1: Specify criteria()
2: Search()
5: select title()
8: select item()
9: checkout()

3: search(criteria)
6: getitem(title)
10: checkout(item)

4: getmatching(criteria)
7: getitems()
Sequence diagram for use case login:

:EmployeeActor : EmployeeLoginUI : LoginWorkflow : User

displayLoginForm()
submitName AndPassword()
validateLogin() -> findbyName()

INVALID
displayErrorMessage()
Fig: collaboration diagram for login
Activity diagram for library application:
STATE MACHINE DIAGRAM FOR THE TITLE CLASS:

STATE DIAGRAM FOR LIBRARY SYSTEM
Component diagram:

Component diagram for library application:

**Figure1:**

**Figure2:**
Deployment diagram for library applications

Deployment usually refers to transferring the project to the required end users along with the project documentation. Deployment diagrams are also essential in each application since it narrates packaged scenario of interaction following is a deployment diagram referring to unified library application scenario

**Figure 1:**

- student/staff
- library management
- library database

**Figure 2:**

- <<processor>> library system
- <<processor>> login server
- <<processor>> check out
- <<processor>> {book server}
Case Study 2: **STUDENT INFORMATION SYSTEM**:

**Problem statement:-**

To find the information regarding students studying in a particular institute which include attendance and marks

**Use case diagram**
MAINTAIN CURRICULUM

- Maintain curriculum
- Introduce new course
- Maintain in take details
- Maintain syllabus
- Maintain circulam
- Maintain student info
- Maintain staff info

Registrar
Class diagram:

- **University**
  - uni_name
  - uni_code
  - uni_clz
  - admissions()
  - schedule()
  - +1
  - member
  - +*

- **Student**
  - stu_name
  - stu_id
  - stu_add
  - stu_ph
  - attend_class()
  - listen()
  - write_exam()
  - attends

- **Course**
  - cou_name
  - cou_id
  - cou_type
  - attends
  - +1..*

- **Department**
  - dept_name
  - dept_id
  - con_class()
  - con_exam()
  - +1
  - referred

- **Staff**
  - staff_name
  - staff_id
  - staff_add
  - staff_ph
  - take_class()
  - evaluate()
  - teaches
  - +*
  - +1..*

- **University** has **Department** +1
- **Department** has **Course** +1..*
- **Course** has **Staff** +1..*
- **University** has **Student** +1
- **Student** attends **Course** +1..*
- **Department** referred **Staff** +1..*
- **Department** assigned to **Staff** +1
- **Staff** assigned to **Department** +1..*
- **Staff** teaches **Course** +*
- **University** head of **Staff** +1
- **Staff** head of **Department** +1..*

Class diagram:

- **Department**
  - name
  - number
  - Fax num
  - HOD
  - find()
  - Delete()
  - Amend()

- **Scot**
  - +has
  - 1..*

- **law**

- **Student b**
  - Name
  - Address
  - Student number
  - Gender
  - Find()
  - add()
  - Delete()
  - Amend()
  - +has
  - 1..*

- **Module**
  - code
  - Module leader
  - level
  - name
  - Find()
  - select()
  - change()
  - Allocate()
  - +canTake
  - 1..*
Sequence diagram:

Collaboration:
Sequence diagram:

Collaboration:
ACTIVITY DIAGRAMS

student sends a req for grade

confirms that the given req student no is available

check pass?

third class

check%<65

YES

first class

check%>75

NO

second class
get the marks of all the students

calculate percentages

if \% \geq 75
  yes
  A grade
  no
  no

if 60 \leq \% < 75
  yes
  B grade
  no

if \% < 60
  yes
  C grade
  no

stop process
Start

Enter student details

<<Condition Checking>>
ValidStudentId( No )
check student id

Invalid id Cancel Request

<<Condition Checking>>
ValidStudentId( Yes )

Calculate Attendence

Calculate Marks

Display Results

End
STATE TRANSITION DIAGRAM FOR STUDENT INFORMATION

State chart diagram

- **Create**
  - Probational
  - On successful completion of 6 month period
    - Successfully completed 6 months period
    - Resigned
    - Left
  - Permanent

STAT
E TRANSITION DIAGRAM FOR STUDENT INFORMATION

initialization
Do: initialize batch

open
Entry: Register students
Exit: increment student

[ canceled Do: notify
registered students ]

add student/set c=0

[c=n]

cancel

cancel

cancel

close Do: finalize
batch

STATE TRANSITION DIAGRAM FOR STUDENT INFORMATION
Initialization
- Do: initialize batch
  - add student set c=0

Entry: Register
- Open student
- Exit: Increment student

Cancel
- do: notify registered students
  - cancel

Grades
- Do: > 75% 1st class
- < 65% and > 50% 2nd class
- < 50% & pass 3rd class

Close
- Do: finalize batch
- cancel

Student state chart diagram
Case study 3:

Bank ATM SYSTEM:

Problem statement:-

S/w system is to be designed for supporting a computerized ATM banking n/w. All the accounts maintained in the bank and also the transactions effected, including ATM transactions, are to be processed by the computers in the bank.

An ATM accepts a relevant cash card interacts with the user, communicates with the central system to carry out the transaction, dispenses cash, and print receipts.

The system to be designed and implemented must include appropriate record keeping and security provisions. The system must handle concurrent accesses to the same account.
1: Insert card
2: Obtain Pin
3: Enter Pin
4: Validate Pin
5: Valid Pin
6: Enter deposit amt
7: Deposit Amt
8: Open deposit slot
9: Insert Envelop amt slot
10: Envelop received
COLLABORATION

1: Insert card
2: Obtain Pin
3: Enter Pin
4: Validate Pin
5: Valid Pin
6: Enter deposit amt
7: Deposit Amt
8: Open deposit slot
9: Insert Envelop slot
10: Envelope received

: Bank server

: ATM machine

: Customer
SEQUENCE DIAGRAM

1: Insert Card
3: Enter Pin
5: Request
6: Obtain types of enquiry
7: Type
2: Obtain Pin
4: Send Pin
8: Get a/c no's
10: Bal enquiry
11: Current bal
12: Transaction history
13: Previous trans
14: View a/c details
15: Display(a/c no, bal, names)
Enquiry collaboration:

1: Insert Card
3: Enter Pin
5: Request
6: Obtain types of enquiry
7: Type
9: self
11: Current bal
13: Previous trans
15: Display(a/c no, bal, names)

2: Obtain Pin
4: Send Pin
8: Get a/c no's
10: Bal enquiry
12: Transaction history
14: View a/c details

COLLABORATION
Withdraw sequence diagram:

1: Insert Card
2: Validate Pin
3: Validate
4: Choose Transaction
5: Withdraw
6: Check Balance
7: Sufficient Balance
8: Credit cash
9: Insufficient Balance
10: Low balance

Withdraw sequence diagram:
COLLABORATION

1: Insert Card
4: Choose Transaction

5: Withdraw
8: Credit card balance
10: Low balance

3: Validate
7: Sufficient Balance
9: Insufficient Balance

2: Validate Pin
6: Check Balance

: Customer

: Bank server

: ATM machine
ACTIVITY FLOW

1. Insert card
2. Enter PIN
3. Validate PIN
4. Select transaction
5. Start transaction
6. Transaction withdraw
7. Receive cash
8. Print receipt
9. Close transaction
10. Transaction [success]
11. Validate PIN
12. Start transaction
13. Select transaction
14. Enter PIN
15. Insert card
COMPONENT Diagram:

STATE CHART FOR ATM

idle → ready

press [first-dig>0]

next number → validate

validate → invalid

valid

transactions → cancel

cancel pressed

cancel

cancelled

complete

customer account

ATM Machine

bank manager

withdraw

balance enquiry

checking account

saving account
Component diagram: A sample component files

Deployement diagram:
Case Study 4: cellular phone system

Problem statement:
The cellular network must place the phone call currently, and also schedule the receiving and conference calls.

Usecase diagram:
Class diagram:

- **subscriber**
  - name
  - address
  - make a call()

- **telephone line**
  - telephone no
  - 1..n
  - 1..n

- **operator**
  - chek bill()
  - register new subscriber()

- **individual**
  - ID

- **Organization**
  - business registration no.

- **call**
  - callee number
  - date
  - start time
  - end time
  - call details()

- **call**
  - 0..n
  - 1..n
  - 0..n
  - 1..n

- **make a call()**

- **telephone no**
  - 1..n
  - 0..n
ACTIVITY diagram for elaboration of make a call
Component diagram
Case study 5: Trading System

Problem statement:

The trading system must take care of sales information of the company and must analyze the potential of the trade.

Usecase diagram:
Class diagram:

- **Customer** with properties: number, name, address. Associations: supplies order, request(request()), receive(receive()), 1..n.

- **Order** with properties: number, date, type. Associations: contains orderline, 1..n.

- **Order Line** with properties: number, quty, name. Associations: for product, 1..n.

- **Product** with properties: number, name, colour, unit price. Associations: for orderline, 1..n.

- **Warehouse** with properties: city, phone number. Associations: stored in product, 1..n.
Sequence diagram:

1. place order() 
2. check product 
3. request stock 
4. product available() 
5. deliver product

Collaboration diagram:

1. place order() 
2. check product 
3. request stock 
4. product available() 
5. deliver product
ACTIVITY DIAGRAM FOR TRADING SYSTEM

customer sends an order request

order request, system confirms the receipt of ...

if order is normal?

no

order is special?

yes

confirm the order

dispatch the order

no

yes

dispatch the order

STATE TRANSITION DIAGRAM FOR TRADING SYSTEM

Idle

send order request

select normal or special order

normal quit

confirm event

failures

order confirmation

complete transaction

dispatch order