1885 SMI UNIVERSITY

Course Description

This course introduces a number of aspects of software engineering and activities included in this domain that focuses on process maturity and team building as well. It gives a good idea of software development right from its inception till maintenance and compartmentalizes the product attributes to have full command over it.

About Lab Manual

This lab manual is developed to make students work on some of the core software engineering concepts and get insight of the course more practically.

The Software Engineering Lab Manual consists of the theoretical assignments along with their practical implementations so that students can understand the importance of the theory of software engineering.

Goals

To help students to develop skills that will enable them to construct software of high quality – software that is reliable, and that is reasonably easy to understand, modify and maintain.

Outcomes

- Understand why software engineering skills are important.
- Understand process modelling and maturity.
- Identify what it takes to produce and maintain a quality software.

Software engineering is an engineering discipline that's applied to the development of software in a systematic approach (called a software process).

It's the application of theories, methods, and tools to design build a software that meets the specifications efficiently, cost-effectively, and ensuring quality.

It's not only concerned with the technical process of building a software, it also includes activities to manage the project, develop tools, methods and theories that support the software production.

Objectives

1. **Prepare Problem Statement for any one of the following systems:**

S.No.	List of Systems
1.	**E-bidding**
2.	**Leave System**
3.	***Online Exam Registration System***
4.	**Inventory Control System**
5.	**Blood Bank System**
6.	**Library Management System**
7.	**Video Streaming Portal**

Problem Statement:

- **Exam** Registration system is used in the effective dispatch of registration form to all of the students.
- **The** core of the system is to get the online registration form (with details such as name, reg.no etc.) filled by the student whose testament is verified for its genuineness by the Exam Registration System with respect to the already existing information in the database.
- **This** system adopts a comprehensive approach to minimize the manual work and schedule resources, time in a cogent manner.
- **After** the first round of verification done by the system, the information is in turn forwarded to the Exam Controller.
- **These** forms the first and foremost step in the processing of exam application.
- **The** application is then processed manually based on the report given by the system. The system also provides the student the list of exam dates. The controller will be provided with fees details to display the current status of application to the student, which they can view in their online interface.
- **After** all the necessary criteria has been met, the original information is added to the database and the hall ticket is sent to the student.

2. Develop list of Functional and Non-functional Requirements for the system selected in part 1.

Functional Requirements:

- **Login / Logout:** The user should sign before starting using the system and sign out after he finishes exam.
- **Register:** Users should register an account for exam and their data are registered in the database.
- **Send Email:** Lecturers can send email to students to give some announcements regarding the exam.
- **Upload:** Lecturers can add questions and answers to the system.
- **Student Report:** This allows lecturers to view the students who fail or pass the exam.
- **Select:** This feature helps students to choose the subject and then proceed to answer the questions.
- **Help:** This enables users to see the information about the system and also the user manual.

Non-Functional Requirements:

- ***Reliability:*** The system can update its content in real time. Therefore, changes such as addition, deletion or modification can be done immediately. The system will also be able to produce all related output to queries.
- ***Availability:*** The system can operate 24 hours per week and 365 days a year. As long as the user not shut down the desktop. All the information will be kept in the database. Even though, the desktop is shut off information still exist in the database.
- ***Secure and safe:*** In order to avoid security and safety breach occur users need to login with username and password before they access the system. In database there have record the username and password. Only registered users can access the system and use it.

- ***Performance:*** The system performance is very fast. The processed transactions and event response time is quick. So, user can do the transaction any event without feel stress on waiting.
- ***Efficiency:*** System should be efficient enough to meet all kinds of requirements as required by the lecturers and students. The system should not hang or lose its efficiency in any kind of worse conditions. It should provide the correct output in all manners.
- ***User Friendly:*** System should be user friendly, so that any user can use and access the system with easiness.
- ***Software Flexibility:*** System is working easily on the Intranet with the username and password of the user. The system has given the rights to the lecturers and the students to use the system with their username.

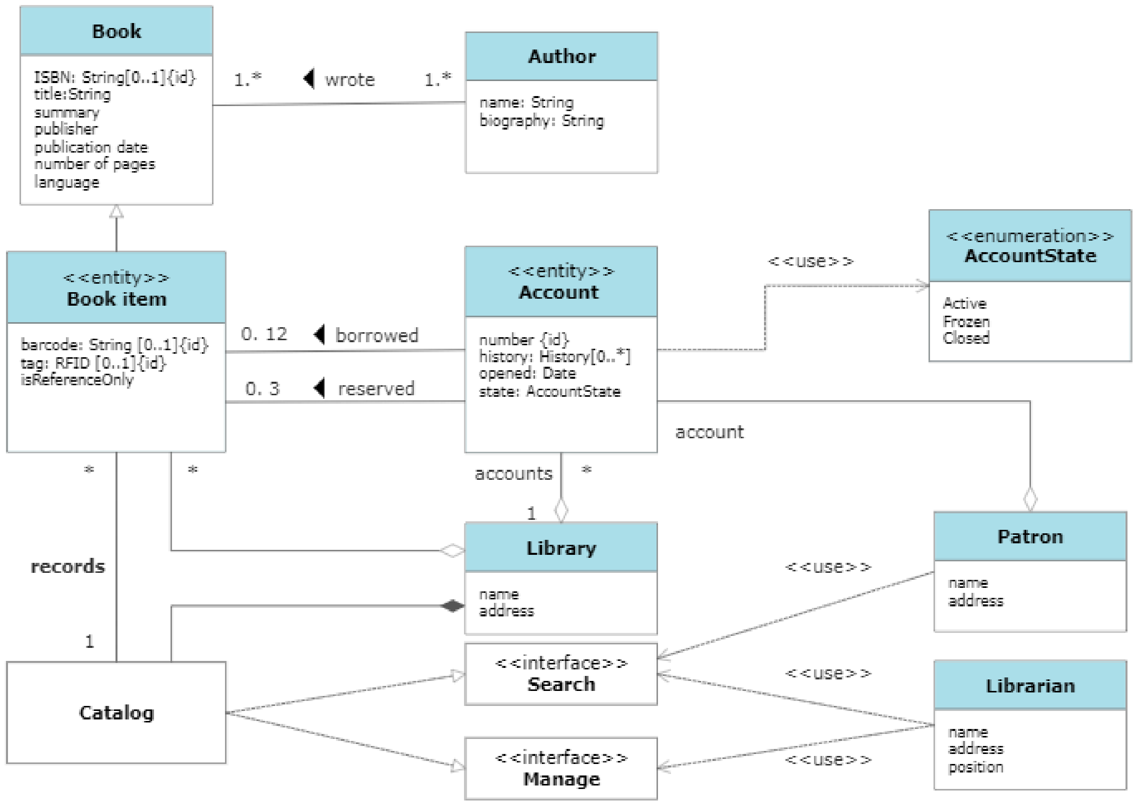
3. List down different types of UMLs and describe each along with their diagrams.

UML is a way of visualizing a software program using a collection of diagrams.

Types of UML Diagrams: These diagrams are organized into two distinct groups.

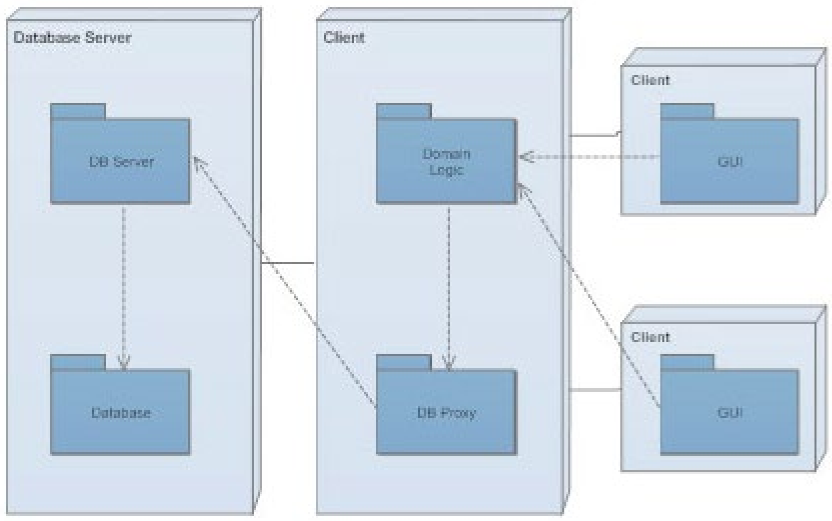
1. Structural UML Diagram:

- ***Class diagram*** are the backbone of almost every object-oriented method, including UML. They describe the static structure of a system.

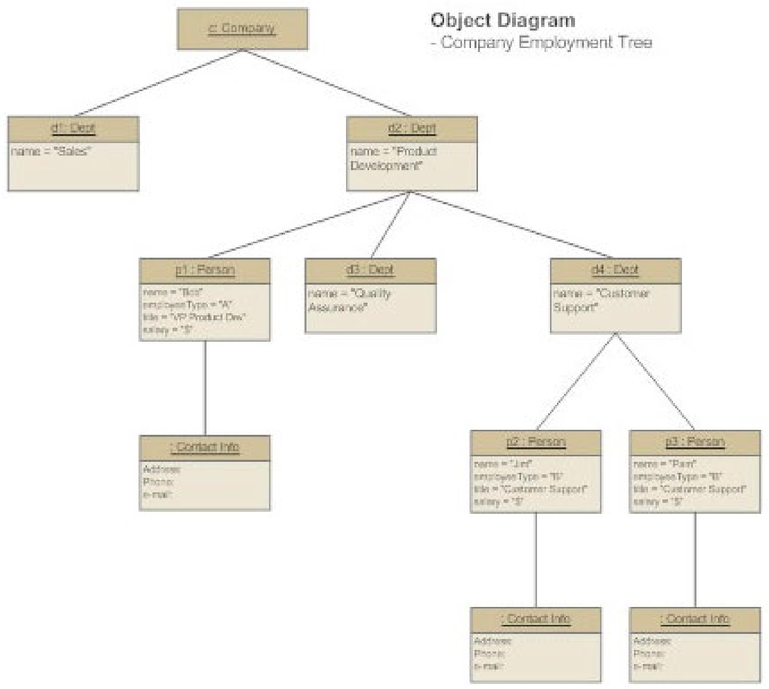


- ***Package diagram*** are a subset of class diagrams, but developers sometimes treat them as a separate technique. Package diagrams organize elements of a system into related groups to minimize dependencies between packages.

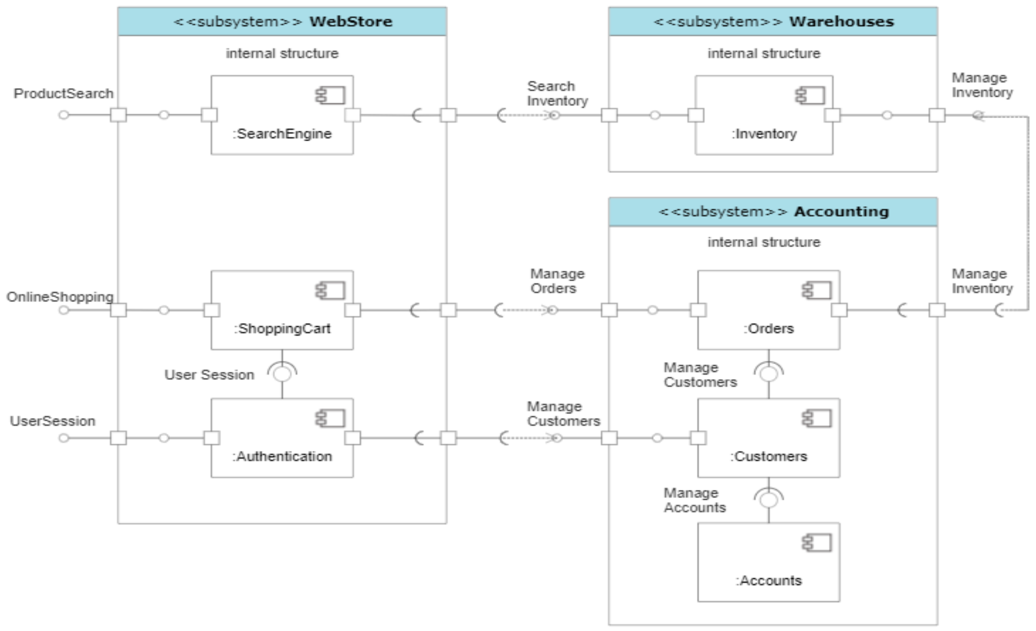
UML Package Diagram - Encapsulation



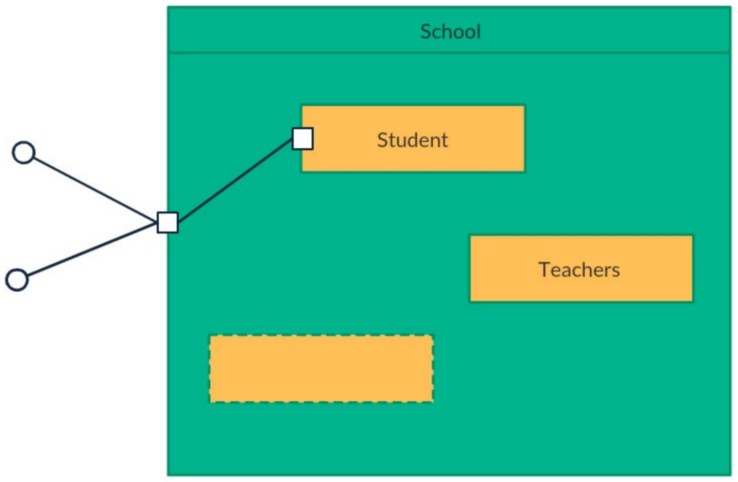
- ***Object diagram*** describe the static structure of a system at a particular time. They can be used to test class diagrams for accuracy.



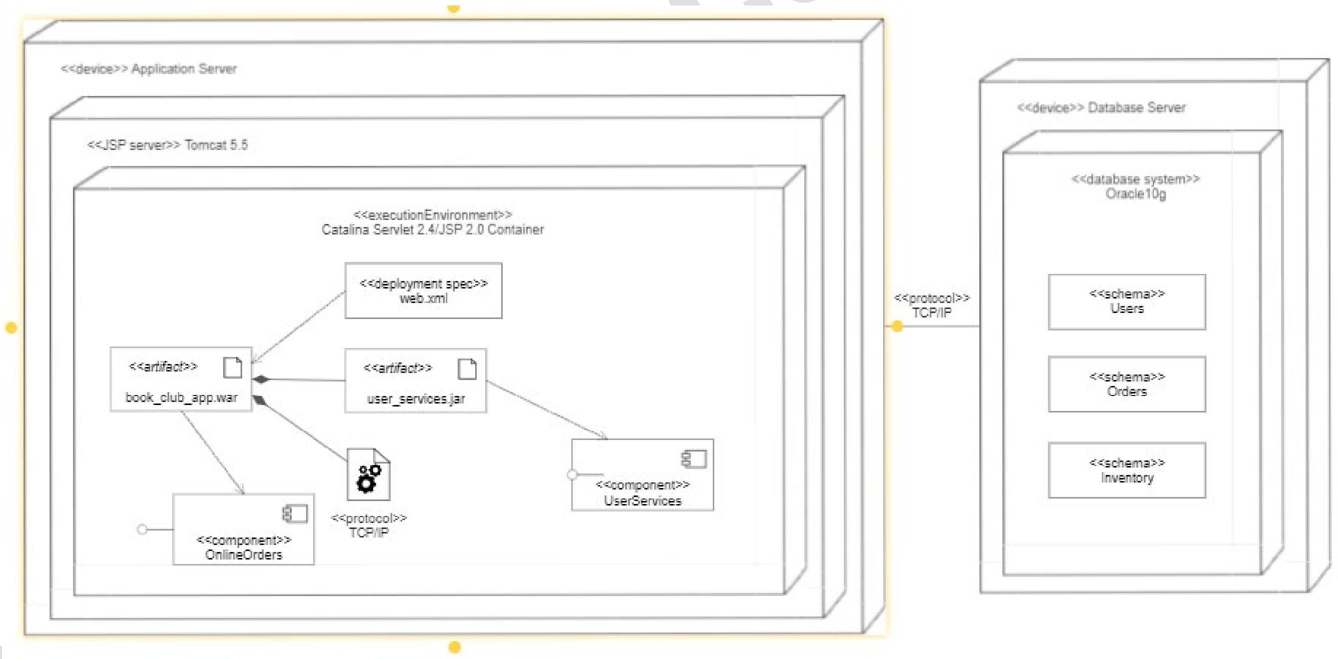
- ***Component diagram*** can help in breaking down the system into smaller components whenever you are dealing with the documentation of quite complex systems.



- ***Composite structure diagram*** Composite structure diagrams show the internal part of a class.

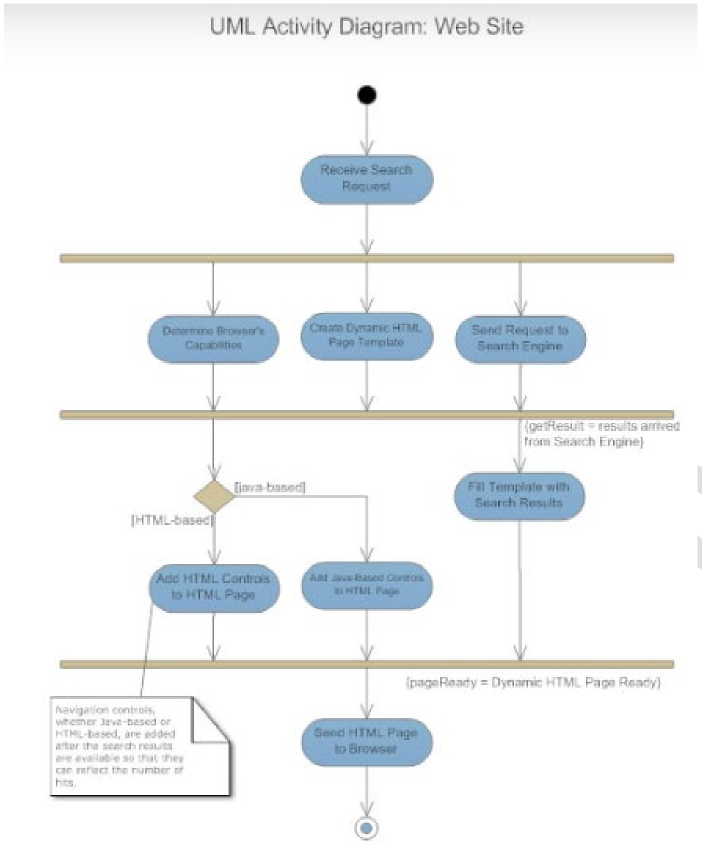


- ***Deployment diagram*** are generally used in visualizing the relationship between the software and the hardware.

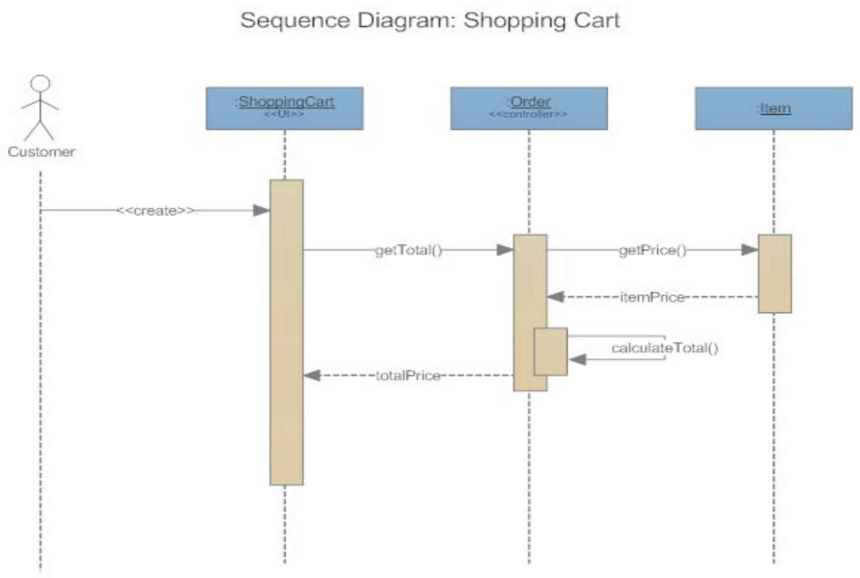


2. Behavioral UML Diagram:

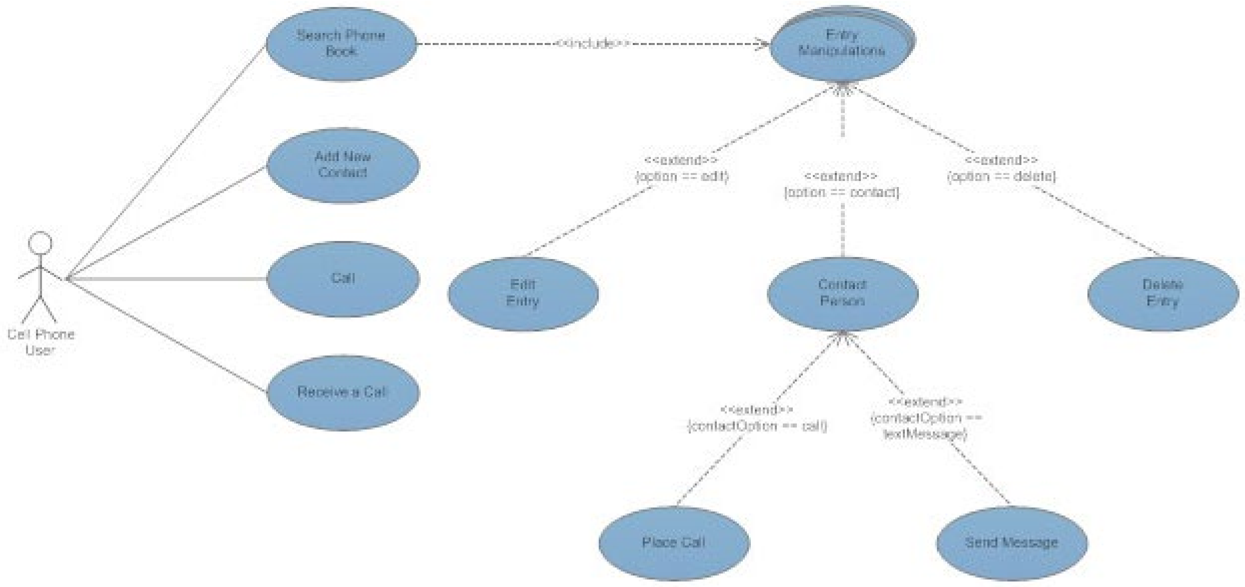
- ***Activity diagram*** illustrate the dynamic nature of a system by modeling the flow of control from activity to activity. An activity represents an operation on some class in the system that results in a change in the state of the system. Typically, activity diagrams are used to model workflow or business processes and internal operation.


UML Activity Diagram: Web Site
Receive Search Request
Determine Browser's Capabilities
Create Dynamic HTML Page Template
Send Request to Search Engine
{getResult = results arrived from Search Engine}
[java-based]
[HTML-based]
Fill Template with Search Results
Add HTML Controls to HTML Page
Add Java-Based Controls to HTML Page
{pageReady = Dynamic HTML Page Ready}
Navigation controls, whether Java-based or HTML-based, are added after the search results are available so that they can reflect the number of hits.
Send HTML Page to Browser

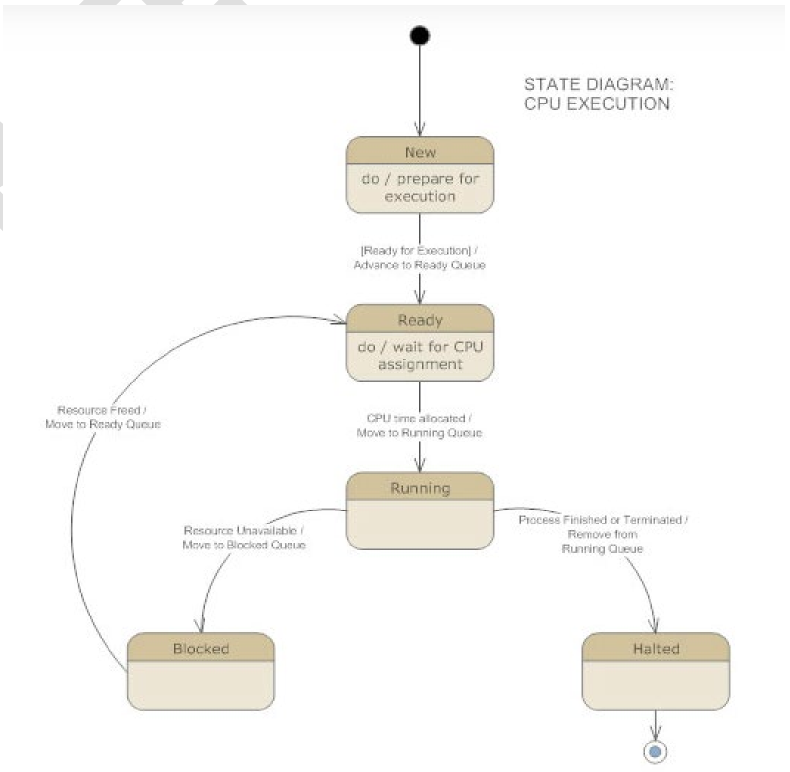
- ***Sequence diagram*** describe interactions among classes in terms of an exchange of messages over time.



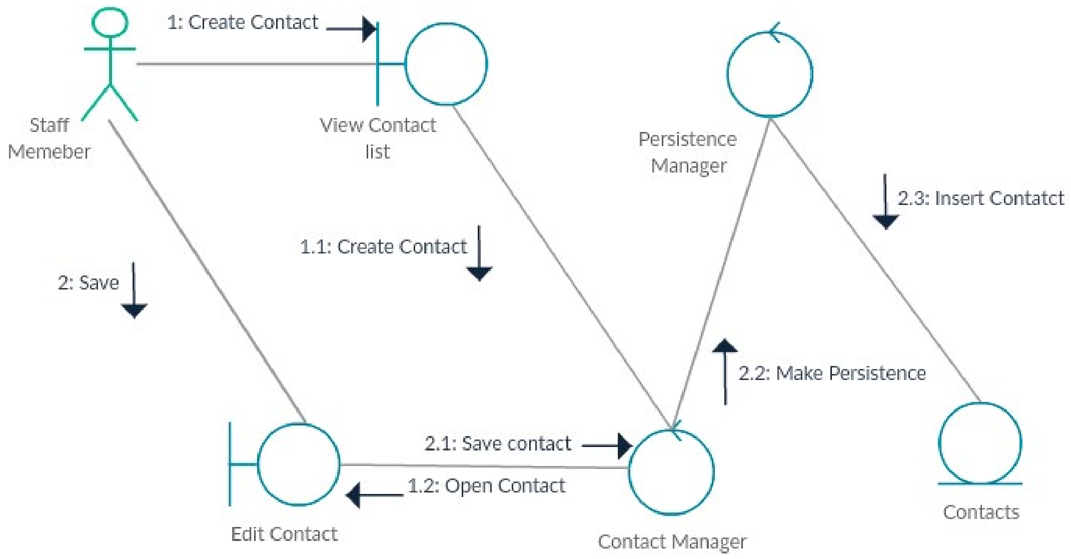
- ***Use case diagram*** model the functionality of a system using actors and use cases.


Search Phone Book
<<include>>
Entry Manipulations
Add New Contact
<<extend>> (option == edit)
<<extend>> [option == contact]
<<extend>> (option == delete)
Call
Edit Entry
Contact Person
Delete Entry
Cell Phone User
Receive a Call
<<extend>> (contactOption == call)
<<extend>> (contactOption == textMessage)
Place Call
Send Message

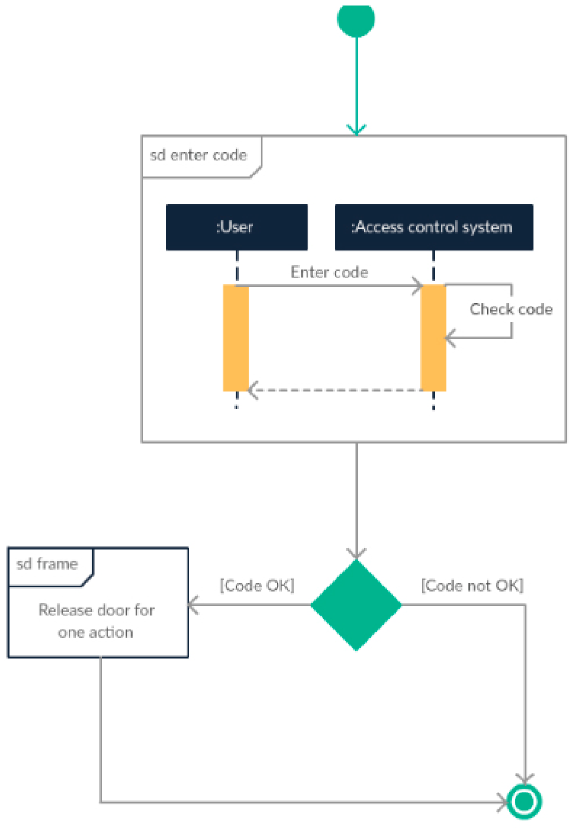
- ***State diagram,*** now known as state machine diagrams and state diagrams describe the dynamic behavior of a system in response to external stimuli.


STATE DIAGRAM: CPU EXECUTION
New
do / prepare for execution
[Ready for Execution] / Advance to Ready Queue
Ready
do / wait for CPU assignment
Resource Freed / Move to Ready Queue
CPU time allocated / Move to Running Queue
Running
Resource Unavailable / Move to Blocked Queue
Process Finished or Terminated / Remove from Running Queue
Blocked
Halted

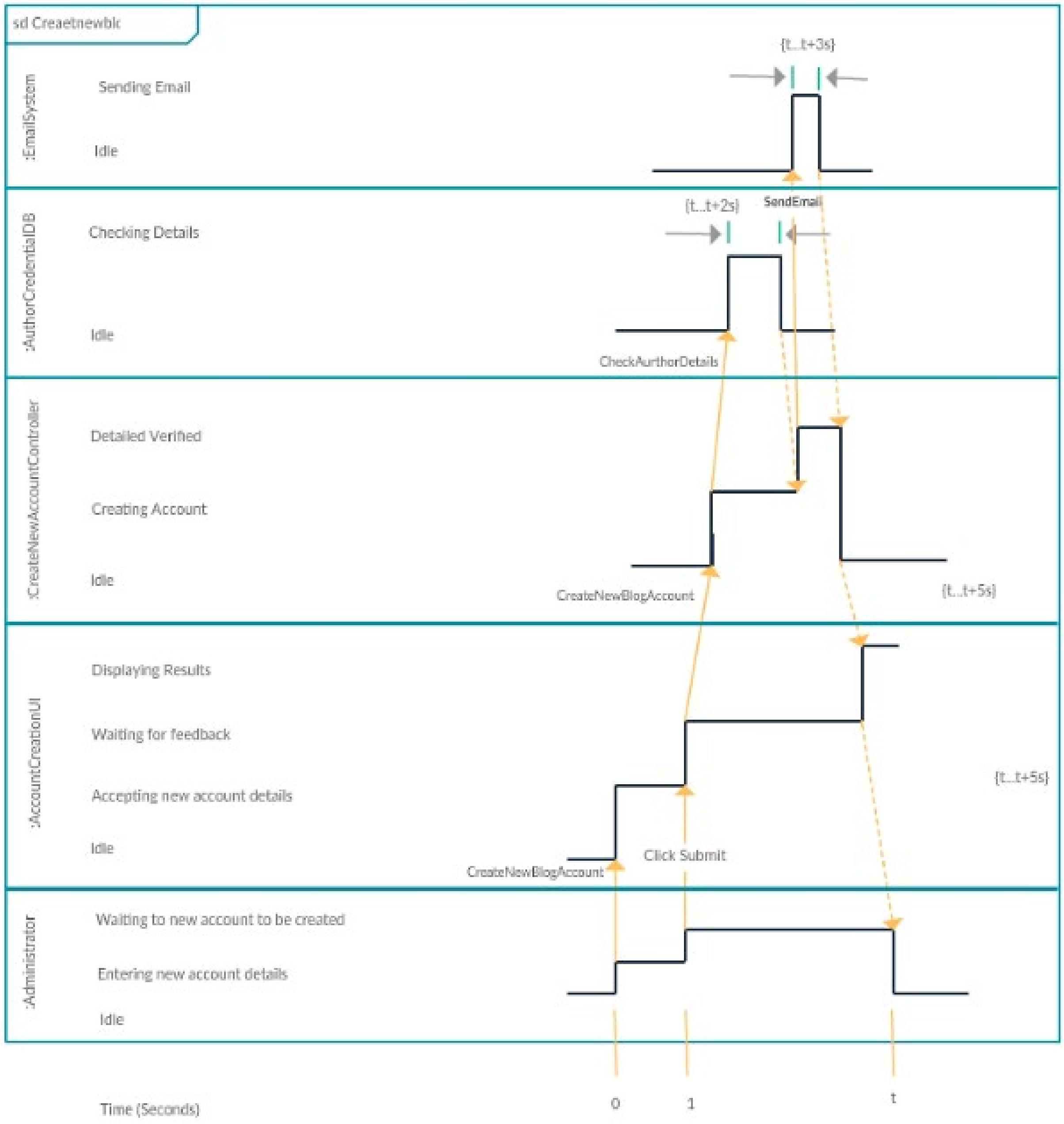
- ***Communication diagram*** model the interactions between objects in sequence. They describe both the static structure and the dynamic behavior of a system. In many ways, a communication diagram is a simplified version of a collaboration diagram introduced in UML 2.0.



- ***Interaction overview diagram*** It is the one that has ability to picture control flow along with nodes that contain interaction diagrams. It is the same as the activity diagram in the sense that both of them visualize the sequence of activities.



- ***Timing diagram*** These diagrams are basically needed in order to represent relations between objects whenever the center of attention is resting on time. However, even though we aren't interested to know how do objects interact or even change each other, despite we wish to represent how to do these objects as well as actors would act along a time axis that is linear.



4. **Install any CASE tool for UML and write about its basic features.**

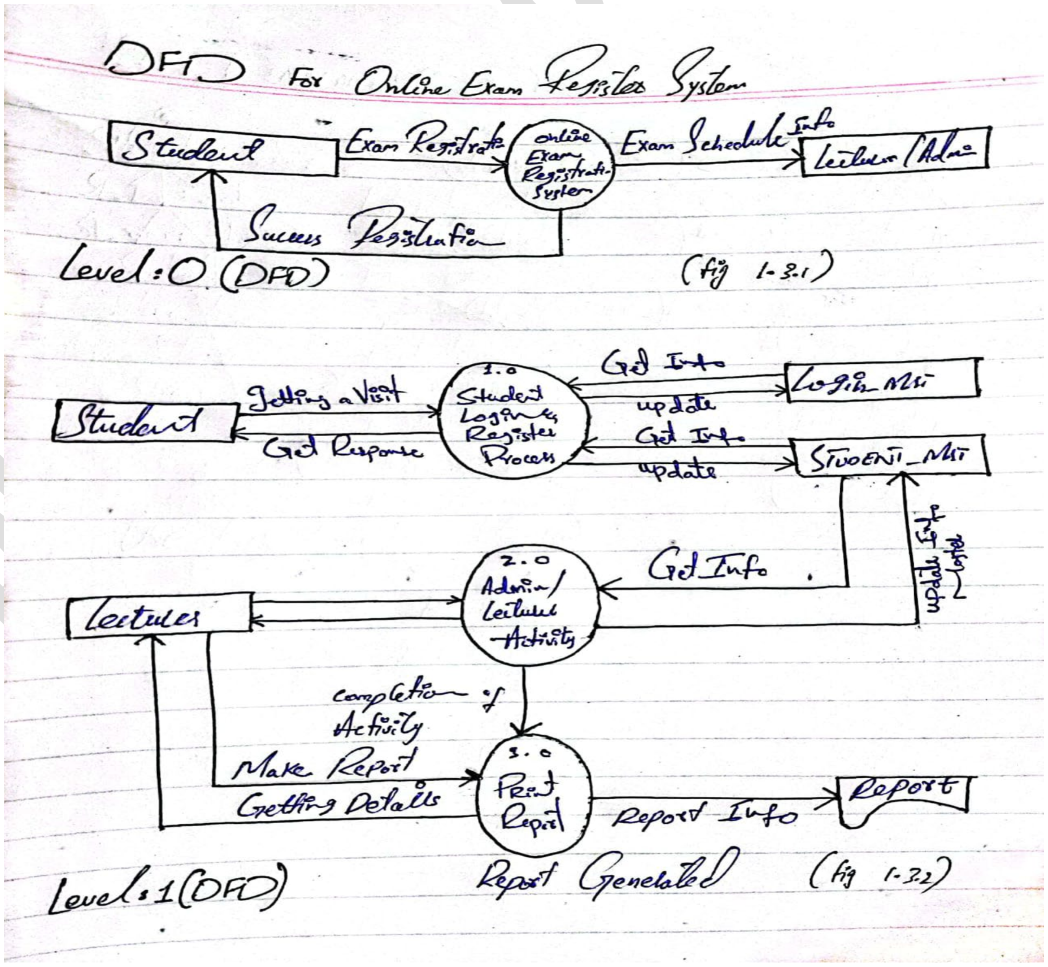
UML Designer Tool: The UML designer tool helps in modifying and envisioning UML2.5 models. It allows you to create all of the UML diagrams.

Its Basic Features which I notice when working on it:

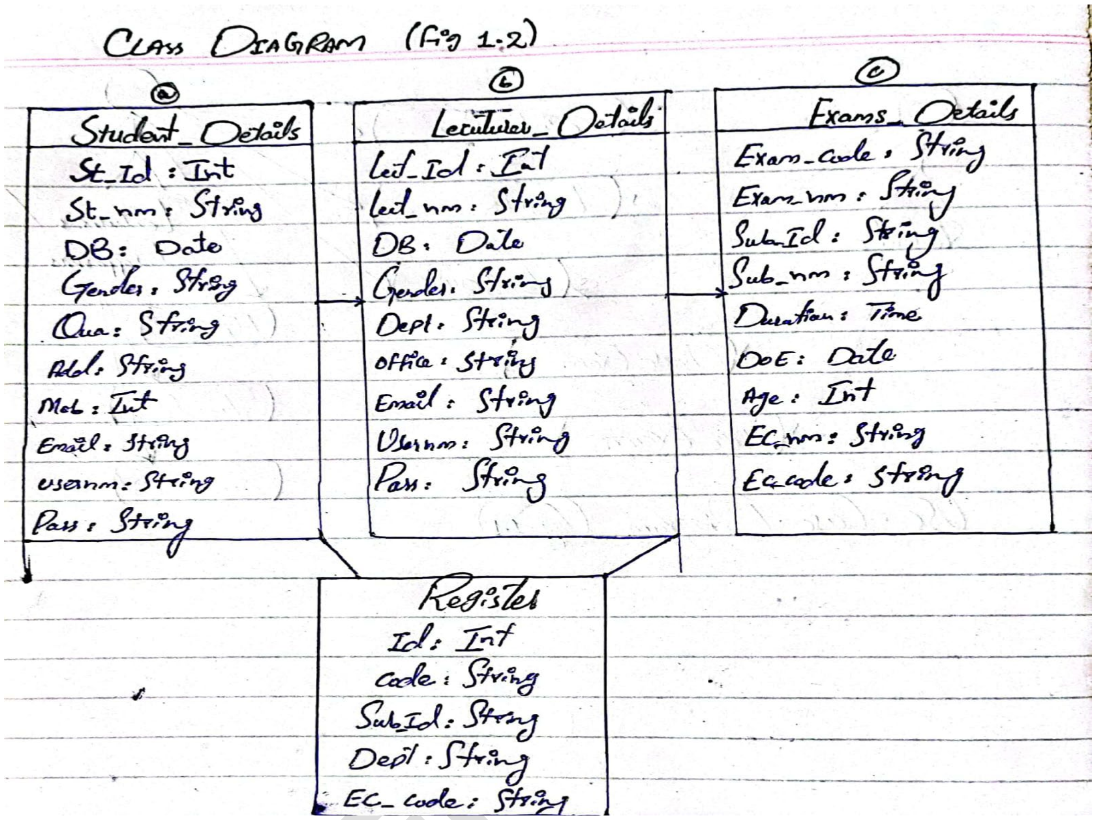
- To start working with DSL, you can use UML legacy models.
- With the UML designer tool, the user can reuse the provided presentations.
- It provides transparency to work on DSL as well as UML models.
- It implements Component, Class, and Composite structure diagrams.

5. **Draw Data Flow Diagram (DFD), Class Diagram and Use-case Diagrams for the system selected in part 1.**

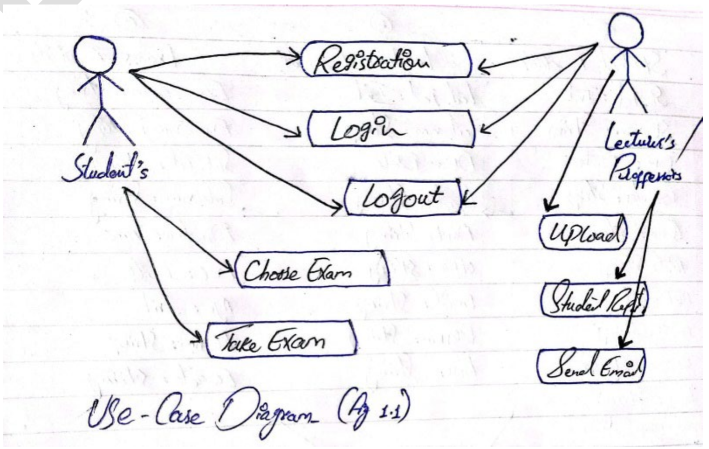
DFD:



Class Diagram:



Use-Case Diagram:



References:

- ✓ https://cloud.smartdraw.com
- ✓ https://creately.com/blog
- ✓ www.educba.com
- ✓ www.smartdraw.com
- ✓ https://www.visual-paradigm.com
- ✓ https://www.uml-diagrams.org