Unified Modelling Language (UML)

Family of graphical notations

Helps in describing and designing software systems especially object oriented systems

Provides high level of abstraction for Design(blueprint) discussions

UML diagrams represent two different views of a system model

Static (or structural) view:

Static structure of the system using objects, attributes, operations and relationships.

Includes class diagrams and composite structure diagrams.

Dynamic (or behavioral) view:

Dynamic behavior of the system by showing collaborations among objects and changes to the internal states of objects.

Includes sequence diagrams, activity diagrams and state machine diagrams**.**

Design includes

any activities (jobs)

individual components of the system

how components interact with other software components

how the system will run

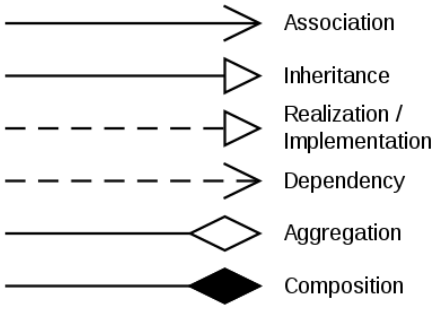
how entities interact with others (components and interfaces)

Use cases

external user interface

Class Diagrams (Structural)

- Describes the types of objects in the System and the various kinds of static relationships that exist among them
- Shows the properties and operations of a Class and the constraints that apply to the way objects are connected



Private members : - symbol

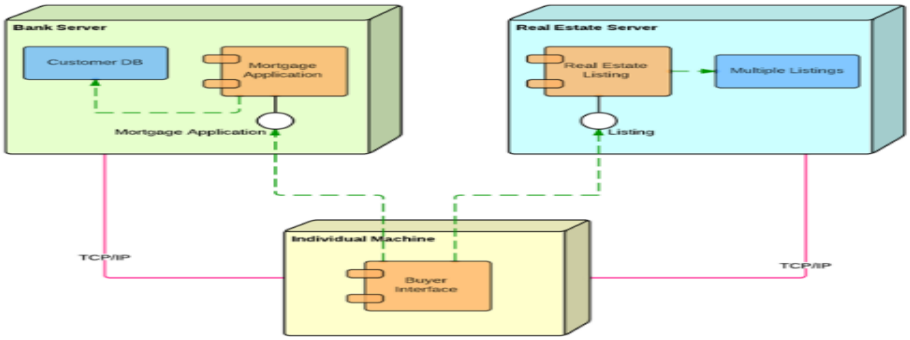
Public members: + symbol

Deployment Diagram (Structural)

Models the physical deployment of artifacts on nodes

deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server),

what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).



Use case Diagram (Behavior Diagram)

Graphical depiction of a user's possible interactions with a system.

User represents the actor ie. the one who is doing actions

A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well.

The use cases are represented by either circles or ellipses.

The actors are often shown as stick figures**.**

Foreg. In a Banking system Actors can be Customer, Bank Manager, Clerk

Activity Diagram (Behavior)

Graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency.

activity diagrams are intended to model both computational and organizational processes(workflows)

Activity diagrams are constructed from a limited number of shapes, connected with arrows.[4] The most important shape types:

Construction

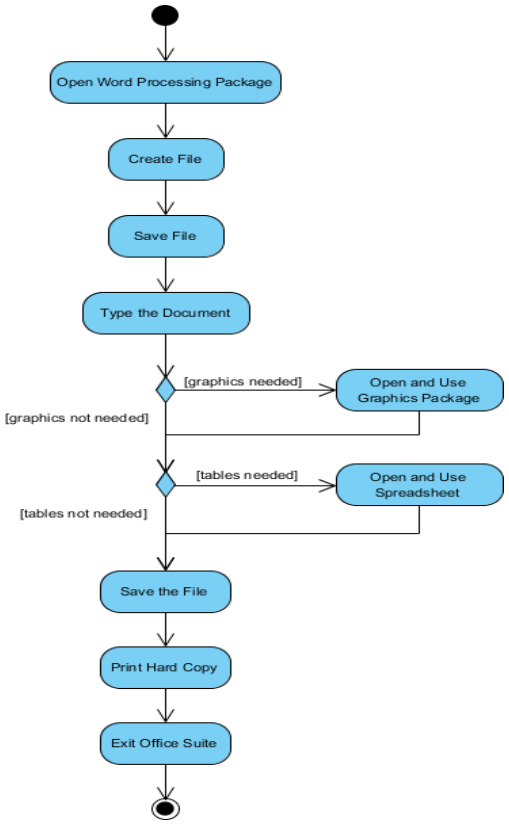
Lines represent actions

diamonds represent decisions

bars represent the start (split) or end (join) of concurrent activities;

a black circle represents the start (initial node) of the workflow;

an encircled black circle represents the end (final node).



Sequence Diagram

Shows process interactions arranged in time

Depicts the processes and objects involved and the sequence of messages exchanged between the processes and objects needed to carry out the functionality.

Construction

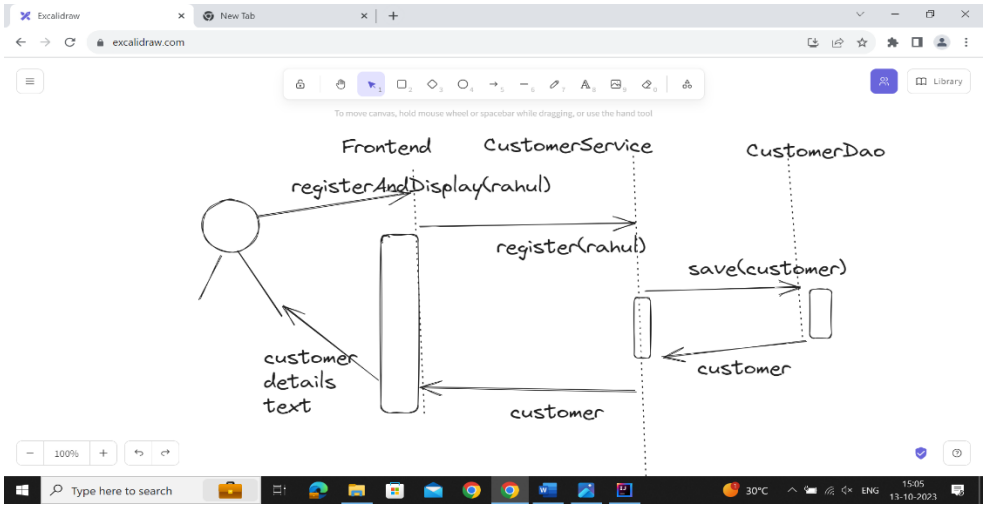
A sequence diagram shows, as parallel vertical lines (lifelines)

Different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

A system sequence diagram should specify and show the following:

1) External actors
2) methods invoked by these actors
3) Return values of methods
4) Indication of any loops or iteration area

Customer registration sequence



Entity Relationship Diagram

illustrates how entities relate to each other.
Used to design or debug relational databases

Uses defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes.

Entities are nouns , relationships are verbs

Represents data or information structure which can be implemented in a database foreg rdbms