Agile Model:

More appropriate. It adopts changes in requirements. Models like incremental model, waterfall model were less corporate towards change.
Agile Model use incremental approach and is used for fast development.
Advantages:

- No team structure. All at same level
- Frequent delivery
- Face to face communication every week with client.
- Less time required in development.

Disadvantes:

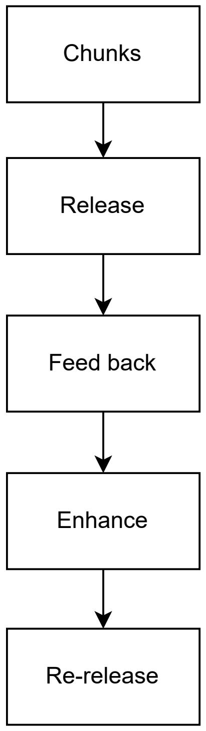
- Less documentation. New person will have dificulties
- Maintenance problem

Agile methodology has 6 types:

- Scrum
- XP (Extreme Programming)
- FDD (Feature Driven Development)
- ASD (Adaptive Software Development)
- DSDM (Dynamic Systems Development Method)
- LSD (Lean Software Development)

Scrum

More common. It is used for large projects.



Scrum Model

We divde the large project to small chunks called iteration.

After one chunk development we release it.

Get feedback of Stackholders

After feedback, enhance that chunk

Re-release

Procedure of Scrum Model

Product Owner(PO): PO tells us about requirements. Then we make a **proper document**. That is know as **Product Backlock**

Sprint Planing(SP): In sprint planing we arrange the **requirements by prioriety.**

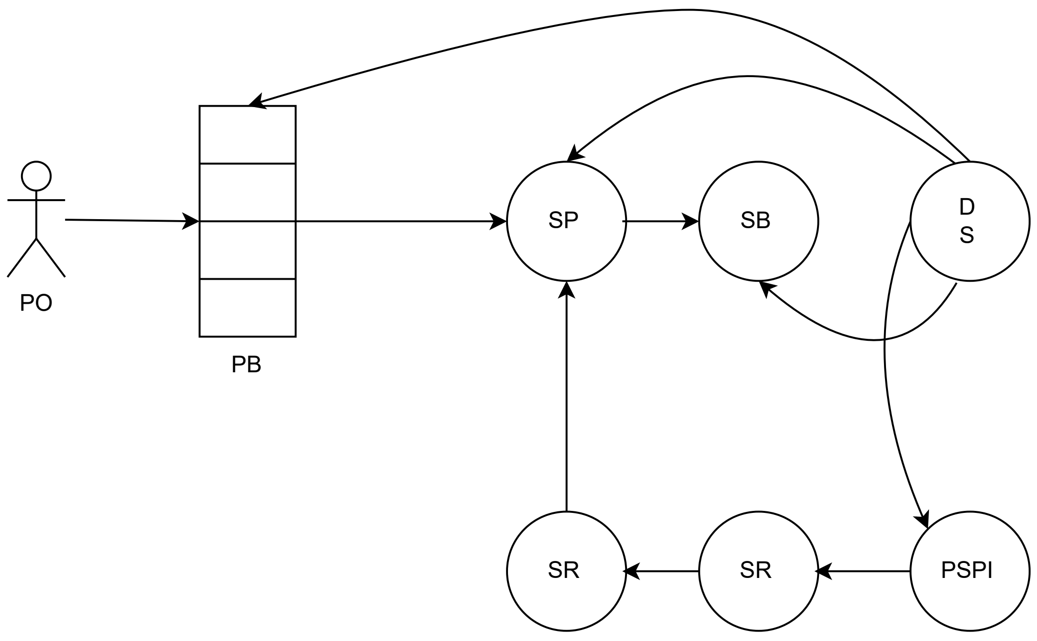
Sprint Backlock(SB): The scrum team decides which requirements to build within **2-4 weeks**

Daily Sprint(DS): The Scrum team has a **daily 15 minute meeting** to discuss any issues that come up

Potential Shipable Product Increment(PSPI): First sprint release is known as PSPI.

Sprint Review(SR): For the review of PSPI, it is **presented to the stackholders**. After that suggestions are incorporated.

Sprint Retrospective(SR): Critical review of PSPI by technical team. Then new suggestions are also incorporated.



XP(Extreme Programing)

Five values of XP:

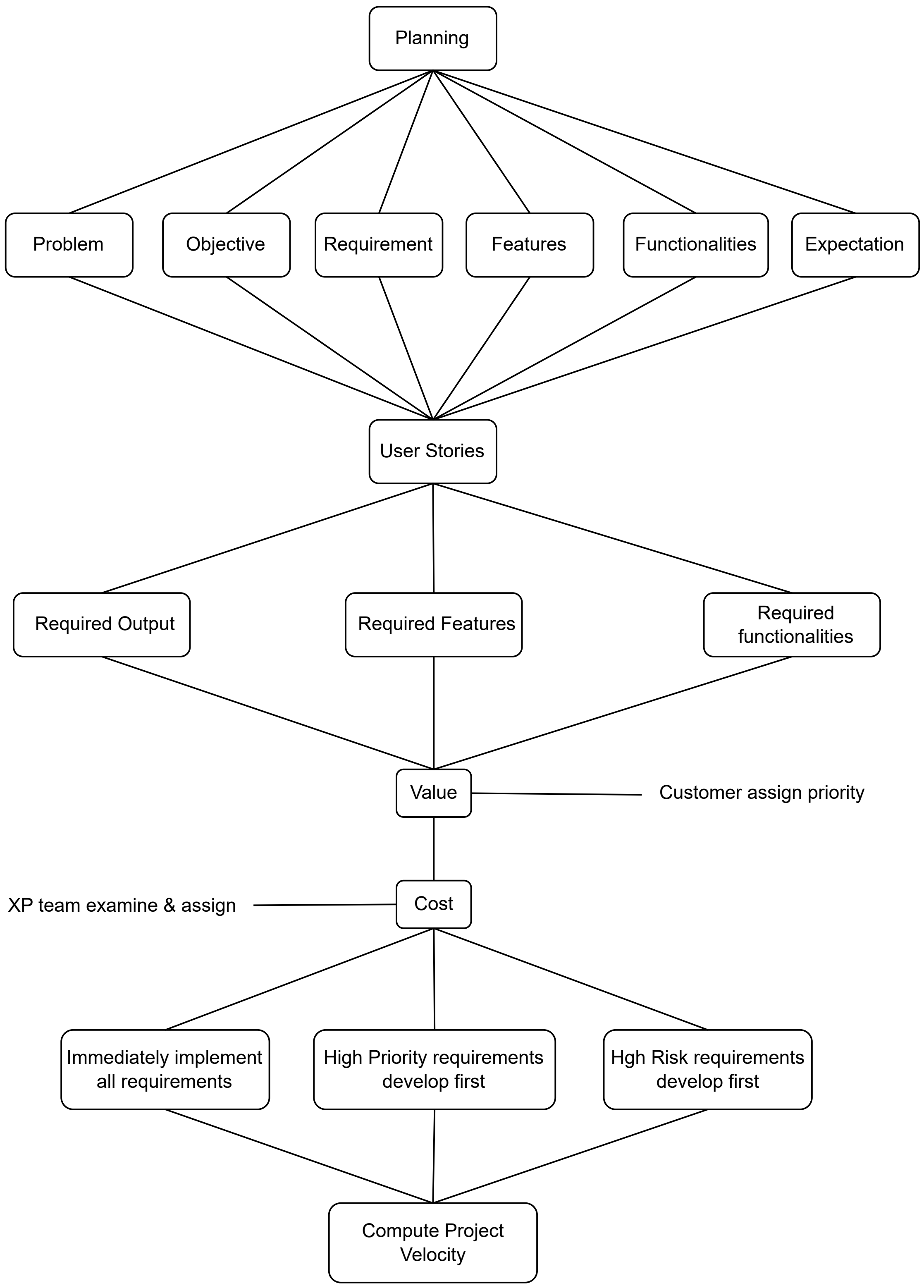
1. **Communication:** Communication between the software development team and stakeholders.
2. **Simplicity:** Eliminate the least important requirements. Do not concentrate on future requirements. Focus on immediate requirements by priority.
3. **Feedback:** Error margin from stakeholders. The difference between actual and expected requirements is known as the error margin. Feedback from developers about budget and deadline.
4. **Courage:** Courage in each member of the development team to raise a particular problem.
5. **Respect:** Respect others' feelings, help out others.

XP Process:
There are four activities of XP process:

1. Planning
2. Design
3. Coding
4. Testing

Planning
Objective: Defined boundries
Requirements: Functional requirements
Expectation: Non-functional requirements.

Goal VS Objective
Goal: Without Boundry. e.g. large house
Objective: Defined Boundries. e.g: 2 canal house.



XP team examin and assign cost
Afterward, the development begins. XP development team and the customer collaborate to decide which approach they are going to follow for development.

Project Velocity
Estimate the delivery date.
Schedule & Plan the subsequent release.
Manage change in the project requirement.

XP Process:

There are four activities of XP process:

1. Planning
2. Design
3. Coding
4. Testing

1. Planning:

1. **User Stories:**
    - **Definition:** A user story is the smallest unit of work or requirement in software. We consider the tiny unit of software and write a story for it.
    - **Features:** (Hint: SMART)
        - Smart (S): A User Story must yield a productive output.
        - Measurable (M): A User Story should be measurable.
        - Achievable (A): A User Story must be achievable.
        - Real (R): A User Story must be real.
        - Time-bound (T): We can define a time limit for the user story.
    - **Template:** In the template, we will describe three points:
        1. Who
        2. What
        3. Why
        - As a [who] i want [what] so that [why]. For example: As a registered user i want to login. So that i can access my account.
2. **Acceptance Criteria:**
    - **Definition:** A set of predefined requirements that must be met to mark a user story complete.
    - **Features:** (Hint: SMART)
        - Smart (S): An acceptance criteria must yield a productive output.
        - Measurable (M): An acceptance criteria should be measurable.
        - Achievable (A): An acceptance criteria must be achievable.
        - Real (R): An acceptance criteria must be real.
        - Time-bound (T): We can define a time limit for the acceptance criteria.
    - **Template:** In the template, we will describe three points:
        1. Given
        2. When
        3. Then
        - Given that [I am logged out]. When [I am on home page & enter valid user name, & password, & enter loging button]. Then [I am oged into my account].

2. Design:

1. **Spike Solutions:**
    - **Definition:** Spike means prototype. Smallest part of project is known as spike. Spike solution is the simplest solution of project. For example, if we are developing a scientific calculator, we will firstly implement simple arithmetic functions.
2. **CRC Card:**
    - CRC Card is a tool used in brainstorming sessions to better help team's collaboration.
    - CRC stands for Class Responsibility Collaboration
        - Class represents a collection of similar objects. e.g. customer
        - Responsibilities are actions the class knows how to perform. e.g. order book
        - Collaboration refers to any other parties the class is working with. e.g: publisher

3. Coding:

- **Refactoring:**
    - **Definition:** we change the code of software to improve its efficiency without changing the external behaviour of the system known as refactoring.
- **Pair Programming:** There are two persons for the task.

4. Testing:

- **Unit testing:** The testing of the smallest unit to check whether it is working properly or not.
- **Integration testing:** When all the smallest units are integrated together then we perform integration testing. After the integration testing, we release the software.
- **Acceptance testing:** Final testing by the stackholder known as acceptance testing.

Design Principle

The universal principle of design is [kis].

- Keep It Simple

Advantages:

- It will be easy to incorporate features like modifications, updating, and reusability.

Testing Techniques

1. Acceptance testing
    1. Alpha testing
    2. Beta testing
2. Black box testing
3. White box testing
4. Regression testing
5. System testing
6. Performance testing

1. Acceptance testing: Acceptance testing is performed by customer to determine whether the software is according to his requirements or not.

1. Alpha testing: Type of acceptance testing performed by group of internal testers from the development side.
2. Beta testing: Type of acceptance testing performed by the external testors of the customer side.

2. Black box testing: Testing technique where the testor does not know the internal structure of the software application. That is known as black box testing. e.g: we can use the lms but we don't have access to the internal structure.

3. White box testing: Testing technique where the testor knows and has the access to the internal structure of the application. That is known as white box testing. e.g: If we are in the development team of the lms. We have the access to the internal structure.

4. Regression testing: Testing technique which is used when there is a bug in our system or when updates are required.We perform updates keeping in mind that the external behaviour of the existing system does not get affected.

5. System testing: System testing is connducted by integrating all components of the entire system collectively.

6. Performance testing: Performance testing involves evaluating four key aspects.

- Stability: Accessing how a system peroforms under a load.
- Speed
- Scalability: The recovery time after a failure.
- Throughput

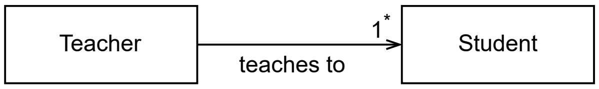
Relationships

There are five types of relationship

- Association relationship
- Dependency relationship
- Aggregation relationship
- Composition relationship
- Generalization relationship

Association relationship:

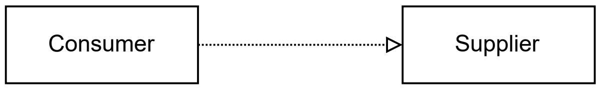
- Association relationship is b/w two classes.
- Association relationship is represented by straight line end up with arrow sign.
- Example:



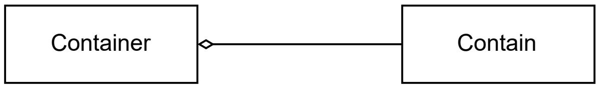
Notations
1 (exact 1)
0..1 (0 or 1)
0^* or * (0 or more)
1^* (1 or more)

Dependency relaltionship:

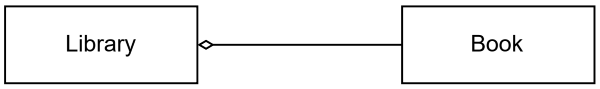
- Dependency relationship is represented by dotted line with unfilled arrow.
- Example:



Aggregation relationship:

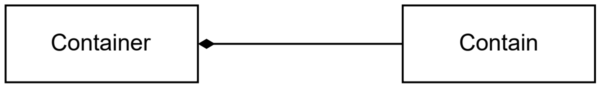


- In aggregation relationship contain class does not dependent on container class
- In other words part of class is not dependent of whole class.
- Aggregration relationship is represented from part to whole with empty diamond sign.
- Example:

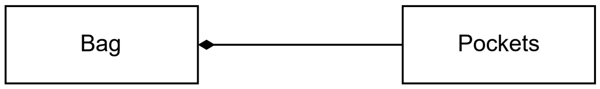


agr library khtm be ho jai to books present rehti hein.

Composition relationship:



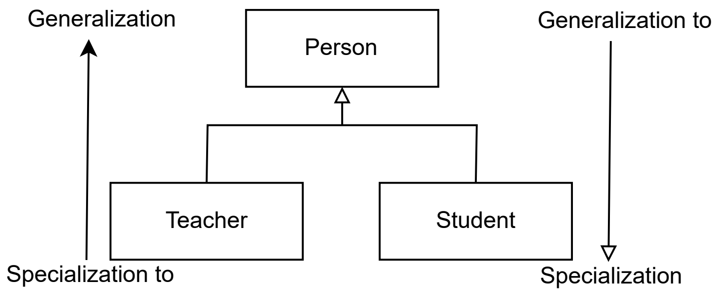
- In composition relationship contain class dependent on container class
- In other words part of class is dependent of whole class.
- Composition relationship is represented from part to whole with filled diamond sign.
- Example:



agr bag destroy ho ga to us kei andar pockets ke existance be nie rhy gie.

Generalization relationship:

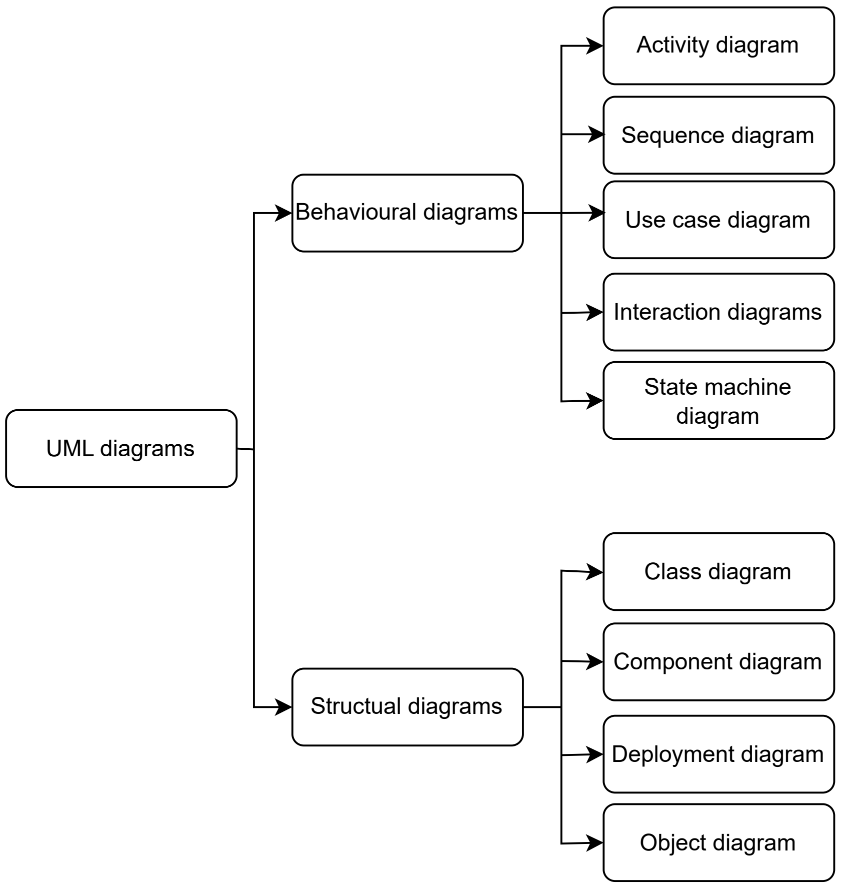
- Generalization is represented by the arrow from bottom to top.
- If the arrow direction is from top to bottom. It mean we are going from
- generalization class to Specialization classses.



UML(Unified Modelig Language)

It's a Notational language.

It is a pictorial diagram(blue print) of a system.



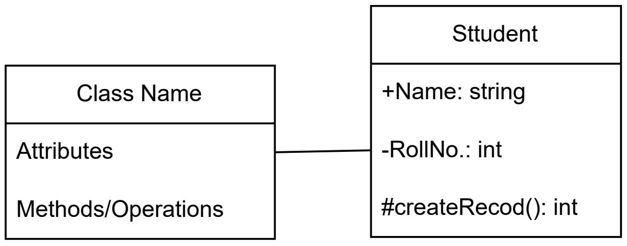
Structual diagrams (Structural diagrams represent the static aspect of a system.)

Let's understand structural diagrams using the keyword CAR.
C for components (wheel, steering, seat, etc.).
A for attributes (material/color/size of the seat).
R for relations (relations between components and attributes).

Behavioural diagrams (Behavioural diagrams represent the dynamic behaviour of a system.)

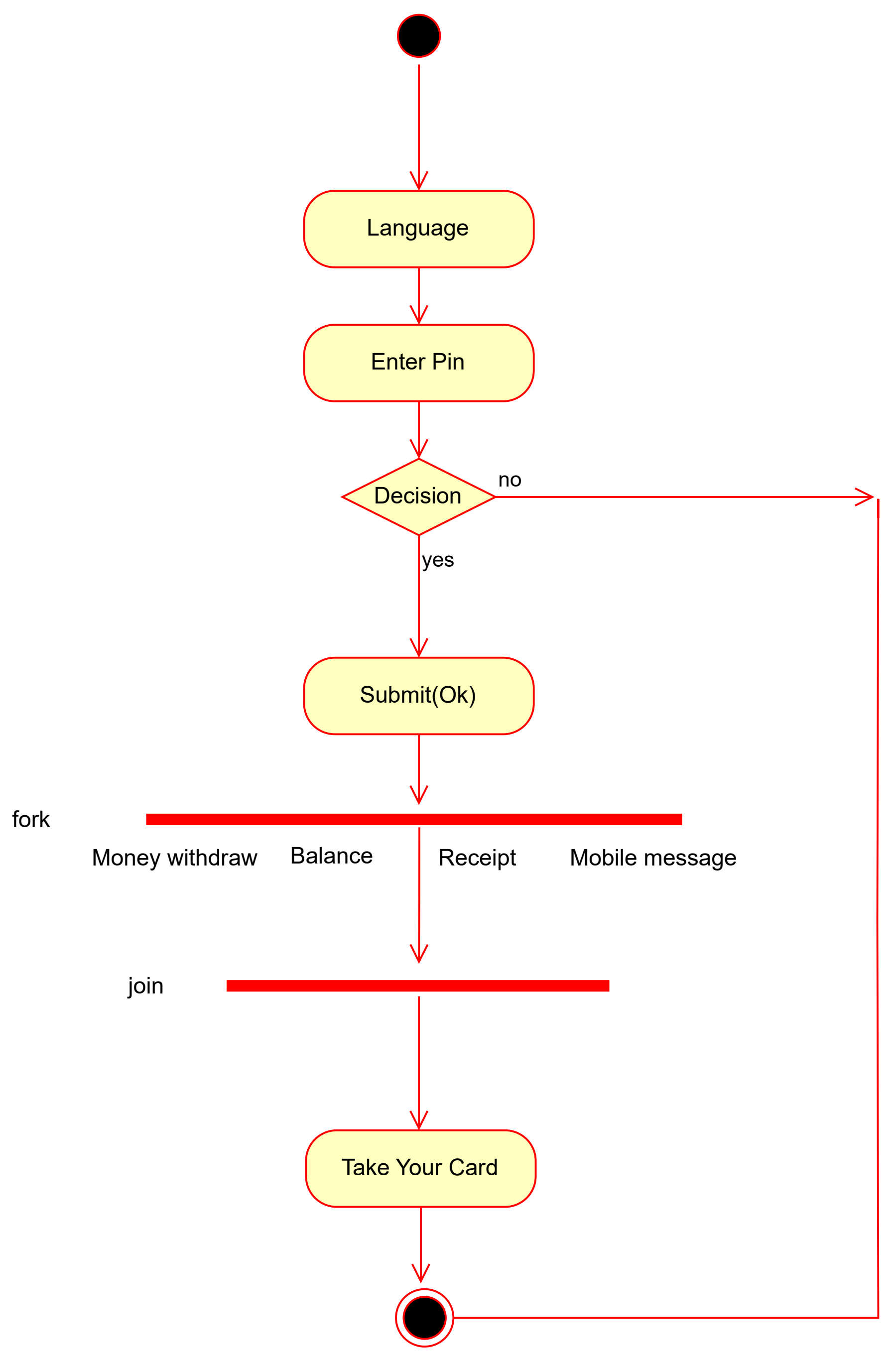
Starting the car by turning the key initiates the activity, and the engine starts in response to that action.

Class Diagram



Activity Diagram

- Activity diagrm represent dynamic behaviour of system and user.
- Activity diagram depicts flow of action of system action or user action.

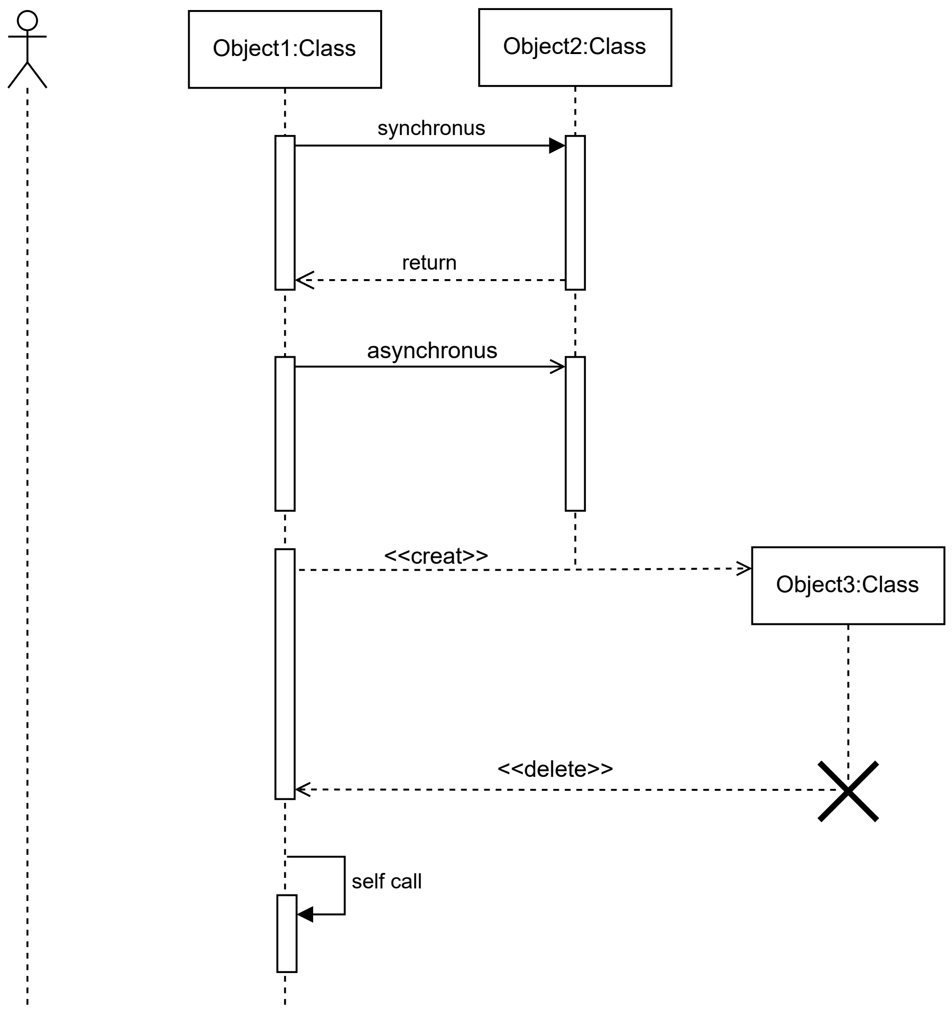


Symbols
start: filled circle rounded rectangles for proecesses Decision: diamond end: cirlce within filled circle. fork: when proecess work parallely. fork: represented by staraight line join: represented by staraight line

Sequence Diagram

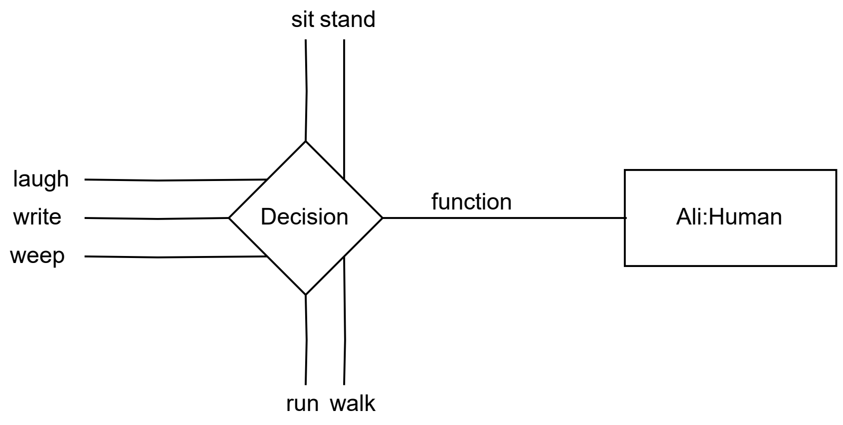
Important features of Sequence diagram:

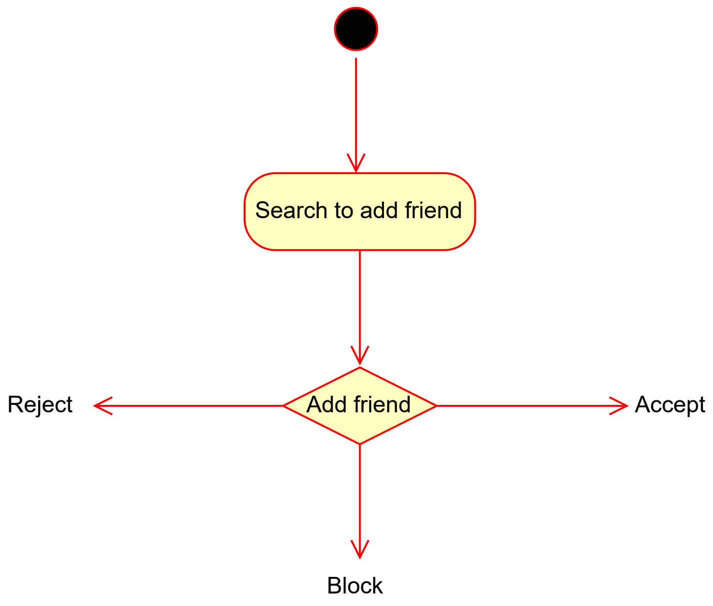
1. **Object:** Represented with rectangle with rounded corners.
2. **Life line:** Show presence of a particular object in the system.
3. **Actor**
4. **Activation Bar:** Represent the active time of the object in the system.
5. **Messages**
    - **Synchronus message:** Sender wait for the receiver's responce. Represented with filled arrow.
    - **Asynchronus message:** Sender does not wait for the receiver's responce and sends the next message.
    - **Return:** Represented with dotted line and unfilled arrow.
    - **Creat message:** Used when during program if there is need to create an object.
    - **Delete message:** Used when during program if there is need to delete an object.
    - **Self message:** self call



State Machine Diagram(State Transition Diagram)

State transition diagram depicts all the possible states of system when a particular function is performed.


sit
stand
laugh
write
weep
Decision
function
Ali:Human
run
walk


Search to add friend
Add friend
Reject
Accept
Block