Purpose of Use Case Diagram

Use case diagrams are typically developed in the early stage of development and people often apply use case modeling for the following purposes:

* Specify the context of a system
* Capture the requirements of a system
* Validate a systems architecture
* Drive implementation and generate test cases
* Developed by analysts together with domain experts

|  |
| --- |
|  |
| **Notation Description** | **Visual Representation** |
| **Actor**   * Someone interacts with use case (system function). * Named by noun. * Actor plays a role in the business * Similar to the concept of user, but a user can play different roles * For example:   + A prof. can be instructor and also researcher   + plays 2 roles with two systems * Actor triggers use case(s). * Actor has a responsibility toward the system (inputs), and Actor has expectations from the system (outputs). | Use Case Diagram Notation - Actor |
| **Use Case**   * System function (process - automated or manual) * Named by verb + Noun (or Noun Phrase). * i.e. Do something * Each Actor must be linked to a use case, while some use cases may not be linked to actors. | Use Case Diagram Notation - Use Case |
| **Communication Link**   * The participation of an actor in a use case is shown by connecting an actor to a use case by a solid link. * Actors may be connected to use cases by associations, indicating that the actor and the use case communicate with one another using messages. | Use Case Diagram Notation - Communication Link |
| **Boundary of system**   * The system boundary is potentially the entire system as defined in the requirements document. * For large and complex systems, each module may be the system boundary. * For example, for an ERP system for an organization, each of the modules such as personnel, payroll, accounting, etc. * can form a system boundary for use cases specific to each of these business functions. * The entire system can span all of these modules depicting the overall system boundary |  |

Sequence diagram

### Purpose of Sequence Diagram

* Model high-level interaction between active objects in a system
* Model the interaction between object instances within a collaboration that realizes a use case
* Model the interaction between objects within a collaboration that realizes an operation
* Either model generic interactions (showing all possible paths through the interaction) or specific instances of a interaction (showing just one path through the interaction)

### Sequence Diagram Notation

|  |  |
| --- | --- |
| **Notation Description** | **Visual Representation** |
| **Actor**   * a type of role played by an entity that interacts with the subject (e.g., by exchanging signals and data) * external to the subject (i.e., in the sense that an instance of an actor is not a part of the instance of its corresponding subject). * represent roles played by human users, external hardware, or other subjects.   Note that:   * An actor does not necessarily represent a specific physical entity but merely a particular role of some entity * A person may play the role of several different actors and, conversely, a given actor may be played by multiple different person. | Actor |
| **Lifeline**   * A lifeline represents an individual participant in the Interaction. | Lifeline |
| **Activations**   * A thin rectangle on a lifeline) represents the period during which an element is performing an operation. * The top and the bottom of the of the rectangle are aligned with the initiation and the completion time respectively | Activation |
| **Call Message**   * A message defines a particular communication between Lifelines of an Interaction. * Call message is a kind of message that represents an invocation of operation of target lifeline. | Call Message |
| **Return Message**   * A message defines a particular communication between Lifelines of an Interaction. * Return message is a kind of message that represents the pass of information back to the caller of a corresponded former message. | Return Message |
| **Self Message**   * A message defines a particular communication between Lifelines of an Interaction. * Self message is a kind of message that represents the invocation of message of the same lifeline. | Self-Message |
| **Recursive Message**   * A message defines a particular communication between Lifelines of an Interaction. * Recursive message is a kind of message that represents the invocation of message of the same lifeline. It's target points to an activation on top of the activation where the message was invoked from. | Recursive Message |
| **Create Message**   * A message defines a particular communication between Lifelines of an Interaction. * Create message is a kind of message that represents the instantiation of (target) lifeline. | Create Message |
| **Destroy Message**   * A message defines a particular communication between Lifelines of an Interaction. * Destroy message is a kind of message that represents the request of destroying the lifecycle of target lifeline. | Destroy Message |
| **Duration Message**   * A message defines a particular communication between Lifelines of an Interaction. * Duration message shows the distance between two time instants for a message invocation. | Duration Message |
| **Note**  A note (comment) gives the ability to attach various remarks to elements. A comment carries no semantic force, but may contain information that is useful to a modeler. |  |

Activity diagram:-

Activity Diagrams describe how activities are coordinated to provide a service which can be at different levels of abstraction. Typically, an event needs to be achieved by some operations, particularly where the operation is intended to achieve a number of different things that require coordination, or how the events in a single use case relate to one another, in particular, use cases where activities may overlap and require coordination. It is also suitable for modeling how a collection of use cases coordinate to represent business workflows

1. Identify candidate use cases, through the examination of business workflows
2. Identify pre- and post-conditions (the context) for use cases
3. Model workflows between/within use cases
4. Model complex workflows in operations on objects
5. Model in detail complex activities in a high level activity Diagram

Flowchart:-

A flowchart is a visual representation of the sequence of steps and decisions needed to perform a process. Each step in the sequence is noted within a diagram shape. Steps are linked by connecting lines and directional arrows. This allows anyone to view the flowchart and logically follow the process from beginning to end.

A flowchart is a powerful business tool. With proper design and construction, it communicates the steps in a process very effectively and efficiently.

Class diagram:-

The main purpose of class diagrams is to build a static view of an application. It is the only diagram that is widely used for construction, and it can be mapped with object-oriented languages. It is one of the most popular UML diagrams. Following are the purpose of class diagrams given below:

1. It analyses and designs a static view of an application.
2. It describes the major responsibilities of a system.
3. It is a base for component and deployment diagrams.
4. It incorporates forward and reverse engineering.

Object diagram:-

Purpose of Object Diagram

The use of object diagrams is fairly limited, mainly to show examples of data structures.

* During the analysis phase of a project, you might create a class diagram to describe the structure of a system and then create a set of object diagrams as test cases to verify the accuracy and completeness of the class diagram.
* Before you create a class diagram, you might create an object diagram to discover facts about specific model elements and their links, or to illustrate specific examples of the classifiers that are required.