Use Case diagram

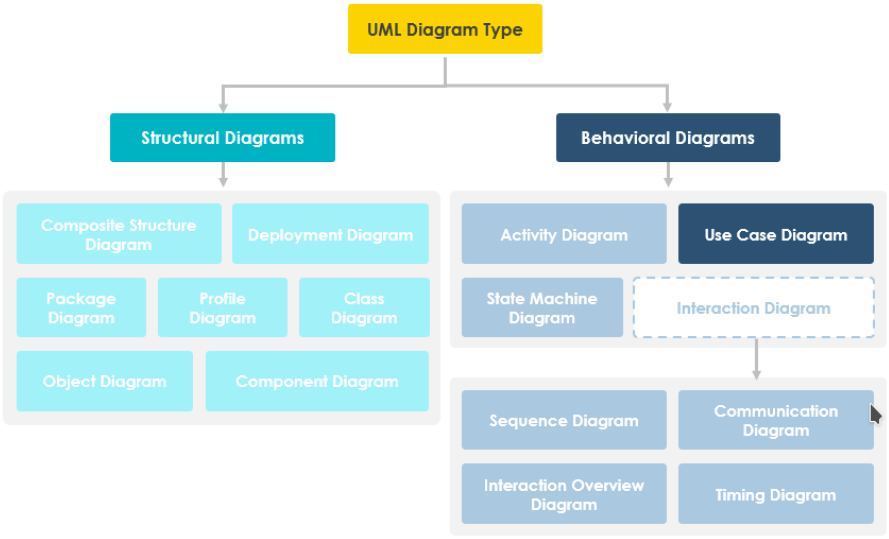
A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

A use case diagram is usually simple. It does not show the detail of the use cases:

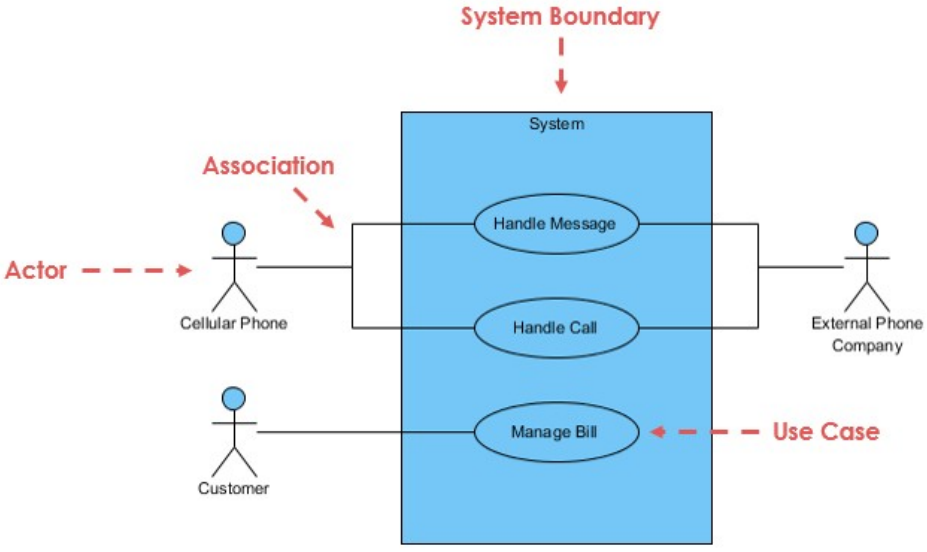
It only summarizes some of the relationships between use cases, actors, and systems.

It does not show the order in which steps are performed to achieve the goals of each use case.

As said, a use case diagram should be simple and contains only a few shapes. If yours contain more than 20 use cases, you are probably misusing use case diagram.

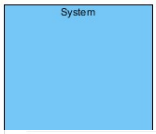


Use cases represent only the functional requirements of a system. Other requirements such as business rules, quality of service requirements, and implementation constraints must be represented separately, again, with other UML diagrams



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 • 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.	
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.	
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	
Generalization • A generalization relationship is a parent-child relationship between use cases. • The child use case is an enhancement of the parent use case. • Generalization is shown as a directed arrow with a triangle arrowhead. • The child use case is connected at the base of the arrow. The tip of the arrow is connected to the parent use case.	
Extends • Indicates that an **"Invalid Password"** use case may include (subject to specified in the extension) the behavior specified by base use case **"Login Account"**. • Depict with a directed arrow having a dotted line. The tip of arrowhead points to the base use case and the child use case is connected at the base of the arrow. • The stereotype "<<extends>>" identifies as an extend relationship	
Include • When a use case is depicted as using the functionality of another use case, the relationship between the use cases is named as include or uses relationship. • A use case includes the functionality described in another use case as a part of its business process flow. • A uses relationship from base use case to child use case indicates that an instance of the base use case will include the behavior as specified in the child use case. • An include relationship is depicted with a directed arrow having a dotted line. The tip of arrowhead points to the child use case and the parent use case connected at the base of the arrow. • The stereotype "<<include>>" identifies the relationship as an include relationship.