**USE CASE UC01 – SIGN UP**

**Primary Actor:** Potential members

**Level:** Blue (Use goal)

**Stakeholder:**

+ Potential members – want to join the gym

+ Trainer – wants to issue the membership card in a timely manner

+ Epay System – collect the payment

+ Finance Manager – wants to receive payments from all members provided

**Preconditions:** None

**Minimal Guarantee:** System will roll back any changes

**Success Guarantee:**

+ New member has been sign up, membership details have been recorded and payment has been received

+ Confirmation of membership has been provided to the new member

+ Trainer (more specific of a staff member) has been notified that access token shall be provided to the new member

**Main success Scenario:**

1. User enter personal details
2. System verifies that person was not already a member -> allow to create account
3. User could have their selections of various membership types and their coresponding cost
4. Systems prompt user for preferred method of delivering the access card
5. User enter payment details
6. System use ePay system to collect payment
7. System provides confirmation via provided method of contact
8. System notifies trainer of manual access card creation and user’s preferred method of delivery
9. User are able to access to gym classes information after sign up
10. User are able to access their booking records and their previous as well as planning payment
11. User would be able to manage their membership, book sessions, and pay  
    fees (via Credit Card) (require an external payment system)

NOT BELONGS TO SIGN UP -> SAVE IT FOR ANOTHER ONE

**Extention or Alternative Flows:**

1. No user interaction for 120 sec

a.1) System aborts use case and revert to home screen

1a) User is already known to the system

1a.1) System perform UC Authentical User

1a.2) System continues with UC Update member details

1b) Member is not permitted to join

1b.1) System reject the application and the UC ends unsuccessfully

6a) Payment unsuccsessful (bank system decline or time out session)

6a.1) System notifies user and reverts to step 4

**USE CASE UC13 – BOOK SERVICE**

**Primary Actor:** GymMember

**Level:** Blue (Use goal)

**Stakeholder and Interests:**

+ Gym Member – wants to book a service then receive confirmation

+ Employee – want to avoid double bookings and want to view existing bookings

+ Epay System – want to charge for bookings that have to be paid in addition to their member fees

**Preconditions:**

+ Member has been authenticated by the system

+ Calender has been populated with the available time slots

**Minimal Guarantee:** System will roll back any changes

**Success Guarantee:**

+ New booking b has been created -> b has been associated with the time slot t and the gym member

+ Time slot t has entered state “booked” in the calender

**Main success Scenario:**

1. Gym member opens calender system
2. System presents available time slots
3. Gym member selects a time slots
4. System verifies that the tiem slot is still available change the status of the time slot to “booked” amd records the booking (and time)
5. System provides booking confrimation to g member

**Extention or Alternative Flows:**

3a) Time slot si no longer available

3a.1) System inform g member

3a.2) UC resumes at step 2

**USE CASE 101**

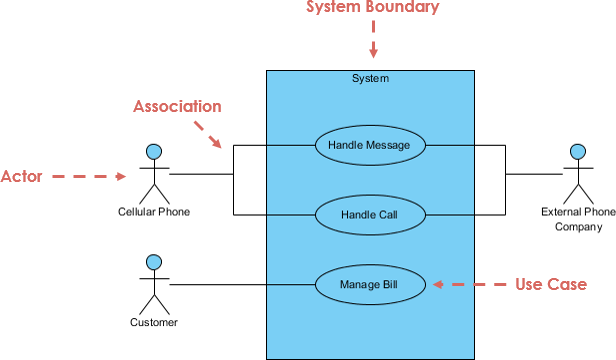
Purpose of Use Case Diagram

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

Use Case Diagram at a Glance

A standard form of use case diagram is defined in the Unified Modeling Language as shown in the Use Case Diagram example below:



|  |  |
| --- | --- |
| 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 | Use Case Diagram Notation - System Boundary |

Structuring Use Case Diagram with Relationships

Use cases share different kinds of relationships. Defining the relationship between two use cases is the decision of the software analysts of the use case diagram. A relationship between two use cases is basically modeling the dependency between the two use cases. The reuse of an existing use case by using different types of relationships reduces the overall effort required in developing a system. Use case relationships are listed as the following:

|  |  |
| --- | --- |
| Use Case Relationship | Visual Representation |
| 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 | Use Case Diagram Notation - Extend |
| 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. | Use Case Diagram Notation - Include |
| 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. | Use Case Diagram Notation - Generalization |

Use Case Examples

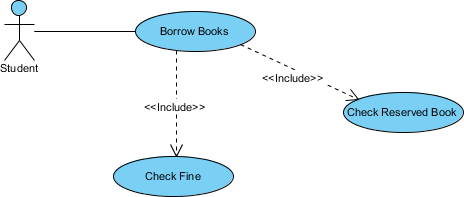
Use Case Example - Association Link

A Use Case diagram illustrates a set of use cases for a system, i.e. the actors and the relationships between the actors and use cases.



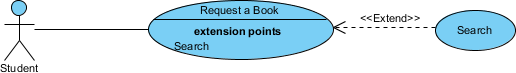
Use Case Example - Include Relationship

The include relationship adds additional functionality not specified in the base use case. The <<Include>> relationship is used to include common behavior from an included use case into a base use case in order to support the reuse of common behavior.



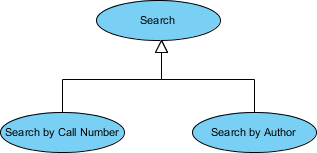
Use Case Example - Extend Relationship

The extend relationships are important because they show optional functionality or system behavior. The <<extend>> relationship is used to include optional behavior from an extending use case in an extended use case. Take a look at the use case diagram example below. It shows an extend connector and an extension point "Search".



Use Case Example - Generalization Relationship

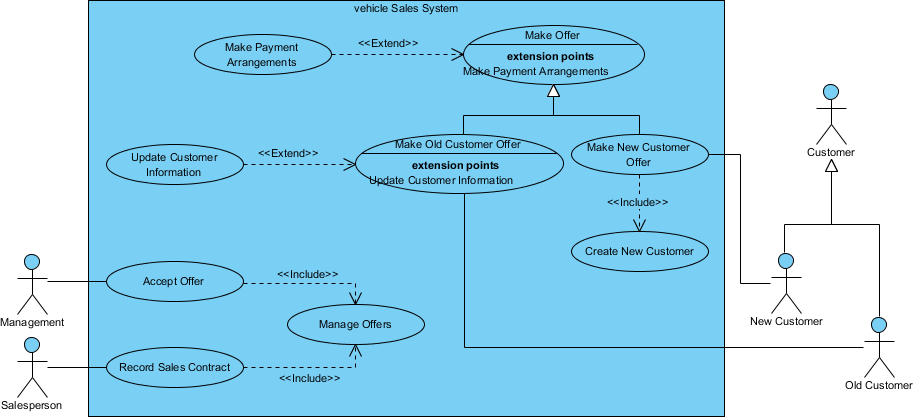
A generalization relationship means that a child use case inherits the behavior and meaning of the parent use case. The child may add or override the behavior of the parent. The figure below provides a use case example by showing two generalization connectors that connect between the three use cases.



Use Case Diagram - Vehicle Sales Systems

The figure below shows a use case diagram example for a vehicle system. As you can see even a system as big as a vehicle sales system contains not more than 10 use cases! That's the beauty of use case modeling.

The use case model also shows the use of extend and include. Besides, there are associations that connect between actors and use cases.



How to Identify Actor

Often, people find it easiest to start the requirements elicitation process by identifying the actors. The following questions can help you identify the actors of your system (Schneider and Winters - 1998):

* Who uses the system?
* Who installs the system?
* Who starts up the system?
* Who maintains the system?
* Who shuts down the system?
* What other systems use this system?
* Who gets information from this system?
* Who provides information to the system?
* Does anything happen automatically at a present time?

How to Identify Use Cases?

Identifying the Use Cases, and then the scenario-based elicitation process carries on by asking what externally visible, observable value that each actor desires. The following questions can be asked to identify use cases, once your actors have been identified (Schneider and Winters - 1998):

* What functions will the actor want from the system?
* Does the system store information? What actors will create, read, update or delete this information?
* Does the system need to notify an actor about changes in the internal state?
* Are there any external events the system must know about? What actor informs the system of those events?

Use Case Diagram Tips

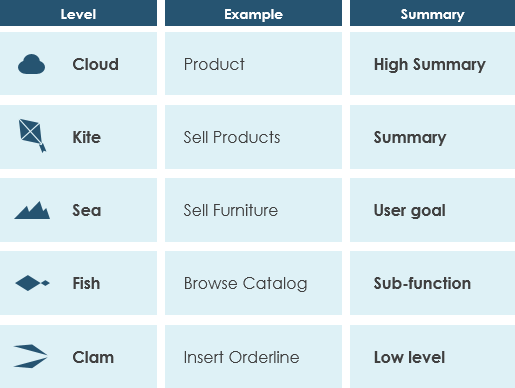
Now, check the tips below to see how to apply use case effectively in your software project.

* Always structure and organize the use case diagram from the perspective of actors.
* Use cases should start off simple and at the highest view possible. Only then can they be refined and detailed further.
* Use case diagrams are based upon functionality and thus should focus on the "what" and not the "how".

Use Case Levels of Details

Use case granularity refers to the way in which information is organized within use case specifications, and to some extent, the level of detail at which they are written. Achieving the right level of use case granularity eases communication between stakeholders and developers and improves project planning.

Alastair Cockburn in *Writing Effective Use Cases* gives us an easy way to visualize different levels of goal level by thinking in terms of the sea:



Note that:

* While a use case itself might drill into a lot of detail about every possibility, a use-case diagram is often used for a higher-level view of the system as blueprints.
* It is beneficial to write use cases at a coarser level of granularity with less detail when it's not required.

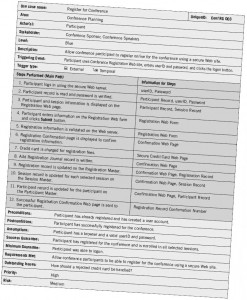
**Use Case Levels (Use case Modeling)**

You may want to create use cases for different levels. One method (defined by Alistair Cockburn) uses the following altitude metaphors:

1. White is the highest level, like clouds. This is the enterprise level, and there may only be four to five for the entire organization. Examples might be to advertise goods, sell goods to customers, manage inventory, manage the supply chain, and optimize shipping.
2. Kite is lower than white but still a high level, providing an overview. The kite use case may be at the business unit or department level and is a summary of goals. Examples would be to register students, or if working with a travel company: make an airline, hotel, car, or cruise reservation.
3. Blue is at sea level, and is usually created for user goals. This often has the greatest interest for users and is easiest for a business to understand. It is usually written for a business activity and each person should be able to do one blue level activity in anywhere from 2 to 20 minutes. Examples are register a continuing student, add a new customer, place an item in a shopping cart, and order checkout.
4. Indigo or fish is a use case that shows lots of detail, often at a functional or subfunctional level. Examples are choose a class, pay academic fees, look up the airport code for a given city, and produce a list of customers after entering a name.
5. Black or clam, like the bottom of the ocean, are the most detailed use cases, at a subfunction level. Examples might be a secure logon validation, adding a new field using dynamic HTML, or using Ajax to update a Web page in a small way.

A use case scenario example is shown in Figure 1 below. Some of the areas included are optional, and may not be used by all organizations. The three main areas are:

1. An area header containing case identifiers and initiators.
2. Steps performed.
3. A footer area containing preconditions, assumptions, questions, and other information.

[](https://www.w3computing.com/systemsanalysis/wp-content/uploads/2014/08/2.16.jpg)Figure 1: A use case scenario is divided into three sections: identification and initiation; steps performed; and conditions, assumptions, and questions.

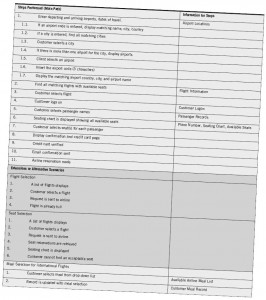
The first area, use case identifiers and initiators, orients the reader and contains the use case name and a unique ID; the application area or system that this use case belongs to; the actors involved in the use case; and the stakeholders that have a high level of interest in the use case. Some stakeholders never interact directly with the system, such as the stockholders, the board of directors, or the sales manager. Each primary actor is a stakeholder, but not listed in the stakeholder area. Include the level (blue, kite, and so on) and a brief description of what the use case accomplishes.

The header concludes with the initiating (triggering) event, that is, what caused the use case to start, and the type of trigger, either external or temporal. External events are those started by an actor, either a person or another system requesting information, such as an airline reservation system requesting flight information from an airline system. Temporal events are those that are triggered or started by time. Events occur at a specific time, such as sending an email about special offers once a week on a Sunday evening, sending bills on a specific day, or generating government statistics on a specified date every quarter.

The second area of the use case includes the steps performed, and the information required for each of the steps. These statements represent the standard flow of events and the steps taken for the successful completion of the use case. It is desirable to write up a use case for the main path, and then to write up one for each of the alternative paths separately, rather than using IF . . .THEN . . . statements. Steps are numbered with an integer. The steps may come from a detailed interview with users or may be derived from agile modeling stories (as described in Chapter 6). These steps should be reviewed with the users for clarification.

The analyst should examine each of the steps and determine the information required for each step. If the analyst cannot determine the information, he or she should schedule a follow-up interview with the user. Some use case descriptions include extensions or alternative scenarios, with the exceptions as additional sections following the standard flow of events. These are numbered with an integer, decimal point, and another integer, such as 3.1, 3.2, 3.3, and so on. These are steps that may or may not be used. Analysts and users can brainstorm what can go wrong with the main path, and may uncover important details and conditions. It is necessary to work with the users to determine what to do when these conditions occur. This helps to detect errors earlier in the life cycle.

Figure 2 illustrates how logic and alternative scenarios can be included in the middle section of a use case. In this airline example, notice that step 1 is made up of smaller steps, many of which are preceded by an “if.” These are still on the main path, but only occur if the condition is met. For example, if there are many airports that serve a city, then all the airports will be displayed. Extensions or alternate scenarios can also appear here. For this airline, other scenarios include flight selection, seat selection, and meal selection. Use cases may even include iterative or looping steps.

[](https://www.w3computing.com/systemsanalysis/wp-content/uploads/2014/08/2.17.jpg)Figure 2: Use cases can include conditional steps as well as extensions or alternative scenarios.

The third area of the use case includes:

* Preconditions, or the condition of the system before the use case may be performed, which may be another use case. An example might be, “The viewer has successfully logged into the system,” or it might be the successful completion of another use case.
* Postconditions, or the state of the system after the use case has finished, including output people have received, transmissions to other systems, and data that have been created or updated. These relate to the goals or user requirements from a problem definition (described in Chapter 3) or to agile stories (described in Chapter 6).
* Assumptions made that would affect the method of the use case and that could stipulate required technology, such as the minimum technology requirements in a browser or even a specific or higher version of a browser. An assumption might be that cookies or JavaScript are enabled. The analyst must determine what to do if the assumptions are not met. When using Google Maps, JavaScript must be enabled. If it is not enabled, the map will not display. Cookies are required by Netflix. Good Web pages will detect that an assumption has not been met and notify the viewer with a message, including information on how to turn on cookies or JavaScript for different browsers.
* Minimal guarantee is the minimum promised to the users. They may not be happy with this result and it may be that nothing happens.
* Success guarantee is what would satisfy the users, and it is usually that the goal of the use case has been met.
* Any outstanding issues or questions must be answered before implementation of the use case.
* An optional statement of priority of the use case, which may come from a problem definition or user requirements.
* An optional statement of risk involved in creating the use case.

The “requirements met” area links the use case to user requirements or objectives from a problem definition. Once you develop the use case scenarios, be sure to review your results with the business experts to verify and refine the use cases if needed.

In this particular use case scenario, called Register for Conference, the only actor involved is the Participant. The overall area is Conference Planning, and the use case is triggered by the participant logging on to the Registration Web page. The Steps Performed area lists the sequence of events that must occur for a successful conference registration. Notice that the information needed to perform each of the steps is listed on the right. This may include Web pages and forms, as well as database tables and records.

The Preconditions area in the footer section of the use case scenario lists what must occur before the participant can register for a conference. In this example, the participant must have already signed up as a member of the society and have a valid userID and password. The Postconditions area lists what has been accomplished by the use case. The Assumptions area lists any basic premises the analyst assumes are fulfilled by the actor beforehand. The Requirements Met area shows why this use case is important and necessary for the business area to be successful. Priority is an indication of which use cases should be developed first and which may be delayed. Risk is a rough assessment of whether there may be problems or difficulties developing the use case. In this case, the risk is medium because the registration use case requires a secure server and is accepting credit card information.

Creating Use Case Descriptions

Use the following four steps to create use case descriptions:

1. Use agile stories, problem definition objectives, user requirements, or a features list as a starting point.
2. Ask about the tasks that must be done to accomplish the transaction. Ask if the use case reads any data or updates any tables.
3. Find out if there are any iterative or looping actions.
4. The use case ends when the customer goal is complete.