Use-Case Specification: <UC-Name>

Note: this template is provided to help on writing a use case specification. The text in blue is included to guide the authors.]

1 General Information

1.1 Use-case description

The description communicates the reason(s) of the use case. In general, a well-written description comprises: (a) the problem which the use case solves; (b) the importance that the use case has in the whole system; and (c) what is achieved by executing the use case. A single or two paragraphs will suffice for this description.

2 Preconditions

A precondition of a use case represents the state of the system that must be present (i.e., must be true) prior to a use case being performed. A well-defined precondition can be easily checked. For example, instead of saying that a system B must available, it should tell the system's name, accessing address, among others.

2.1 Precondition one

A first precondition.

2.2 Precondition two

A second precondition.

3 Scenarios

A use case consists of a number of scenarios, each representing specific instances of the use case that correspond to specific inputs from the actor or to specific conditions in the environment. Each scenario describes alternative ways that the system provides a behavior, or it can describe failure or exception cases., as depicted in figure 1.

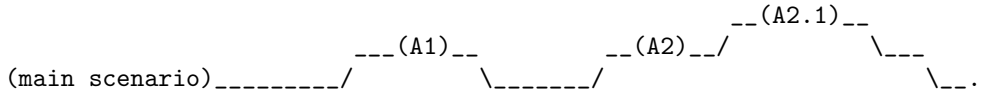

Figure 1. Use-case scenarios

3.1 Main Scenario

The main flow follows a step-by-step description of the use-case. This means that the main flow describes what the actor does and what the system does in response; i.e., it needs to be phrased in the form of a dialog between the actor and the system. In general, a scenario description explores:

- How and when the use case starts
- When the use case interacts with the actors, and what data they exchange
- When the use case uses data stored in the system or stores data in the system
- How and when the use case ends

It does not describe:

- The graphical user interface (GUI)
- Technical details of hardware or software
- Design issues

As you detail the main scenario, identify alternate flows by asking these questions for each step:

- Are there different options available, depending on input from the actor? (for example, if the actor enters an invalid PIN number while accessing an ATM)
- What business rules can come into play? (For instance, the actor requests more money from the ATM than is available in her account)
- What could go wrong? (such as no network connection available when required to perform a transaction) . It is best to develop these scenarios iteratively, as well. Begin by identifying them. Examine each possible scenario to determine whether it is relevant, that it can actually happen, and that it is distinct from other scenarios. Eliminate redundant or unnecessary scenarios, and then start elaborating on the more important ones.

If any information is exchanged, be specific about what is passed back and forth. For example, it is not interesting to say that the client/user enter his/her account data to access his/her bank account. It is better to say the client gives the account's number and the password.
A picture is sometimes worth a thousand words, though there is no substitute for clean, clear prose. If it improves clarity, feel free to paste graphical depictions of user interfaces, process flows or other figures into the use case. If a flow chart is useful to present a complex decision process, by all means use it! Similarly for state-dependent behavior, a state-transition diagram often clarifies the behavior of a system better than pages upon pages of text. Use the right presentation medium for your problem, but be wary of using terminology, notations or figures that your audience may not understand. Remember that your purpose is to clarify, not obscure.

3.2 Alternative Scenario

3.2.1 < First Alternative Scenario >

More complex alternatives are described in a separate section, referred to in the Main Scenario subsection of the Scenarios section. Think of the Alternative Scenario subsections like alternative behaviors. In this case, each alternative flow represents an alternative behavior usually due to exceptions that occur in the main flow.
An alternative flow may be as long as necessary to describe the events associated with its behavior. When an alternative flow ends, the events of the main flow of events are resumed unless otherwise stated.

3.2.2 < Second Alternative Scenario >

There may be, and most likely will be, a number of alternative flows in a use case. Keep each alternative flow separate to improve clarity. Using alternative flows improves the readability of the use case, as well as preventing use cases from being decomposed into hierarchies of use cases. Keep in mind that use cases are just textual descriptions, and their main purpose is to document the behavior of a system in a clear, concise, and understandable way.

4 Postconditions

A postcondition of a use case is a list of possible states that the system can be in immediately after the use case has finished.

4.1 < Postcondition one >

4.2 < Postcondition two >