# Presentation of the Use Case

## Use Case Name

A *use case name* provides a unique identifier for the use case. It should be written in verb-noun format (e.g., *Borrow Books*, *Withdraw Cash*), should describe an achievable goal (e.g., *Register User* is better than *Registering User*) and should be sufficient for the end user to understand what the use case is about.

Goal-driven use case analysis will name use cases according to the actor's goals, thus ensuring use cases are strongly user centric. Two to three words is the optimum. If more than four words are proposed for a name, there is usually a shorter and more specific name that could be used.

## Version

Often a *version* section is needed to inform the reader of the stage a use case has reached. The initial use case developed for business analysis and scoping may well be very different from the evolved version of that use case when the software is being developed. Older versions of the use case may still be in current documents, because they may be valuable to different user groups.

## Goal

Without a *goal* a use case is useless. There is no need for a use case when there is no need for any actor to achieve a goal. A goal briefly describes what the principal actor intends to achieve with this use case.

## Summary

The *summary* is used to capture the essence of a use case before the main body is complete. It provides a quick overview, which is intended to save the reader from having to read the full contents of a use case to understand what the use case is about. Ideally, a summary is just a few sentences or a paragraph in length and includes the goal and principal actor.

# Detailed Description of the Use Case

## System (Added by Etienne)

The *system* section must describe the *boundaries* of the system for which the current use case captures one behavior, but should *not* describe its internal functioning.

For example, in case of activity migration, it is most important to specify if the execution service managing the migrated activity is considered *inside* or *outside* the system.

## Actors (or Stakeholders, or Participants, or Roles)

An *actor* is someone or something outside the system that either acts on the system – a primary actor – or is acted on by the system – a secondary actor. An actor may be a person, a device, another system or sub-system, or time. Actors represent the different roles that something outside has in its relationship with the system whose functional requirements are being specified. An individual in the real world can be represented by several actors if they have several different roles and goals in regards to a system. These interact with system and do some action on that.

A stakeholder is an individual or department that is affected by the outcome of the use case, and might be called on to provide input, feedback, or authorization for the use case. The stakeholder description can include a brief presentation of which of these functions the stakeholder is assigned to fulfill.

## Preconditions

A *preconditions* section defines all the conditions that must be true (i.e., describes the state of the system) for the *trigger* (see below) to meaningfully cause the initiation of the use case. That is, if the system is not in the state described in the preconditions, the behavior of the use case is indeterminate. Note that the preconditions are not the same thing as the "trigger" (see below): the mere fact that the preconditions are met does NOT initiate the use case.

## Triggers

A *triggers* section describes the event that causes the use case to be initiated. This event can be external, temporal or internal. If the trigger is not a simple true "event" (e.g., the customer presses a button), but instead "when a set of conditions are met", there will need to be a triggering process that continually (or periodically) runs to test whether the "trigger conditions" are met: the "triggering event" is a signal from the trigger process that the conditions are now met.

## Basic course of events

At a minimum, each use case should convey a *primary scenario*, or typical course of events, which are interactions between the actors and the system (excluding internal events of the system). This is also called "basic flow", "normal flow," "happy flow" and "Happy path". The main basic course of events is often conveyed as a set of usually numbered steps, specifying *who* performs *what* and sends *which message* to *whom*. For example:

1) The system prompts the user to log on,

2) The user enters his name and password,

3) The system verifies the logon information,

4) The system logs user on to system.

## Postconditions

The *post-conditions* section describes what the change in state of the system will be after the use case completes. Post-conditions are guaranteed to be true when the use case ends. For example persistent data, logs for security audits.

# Additional Information for the Use Case

## Alternative paths or Extensions

Use cases may contain *secondary paths* or *alternative scenarios*, which are variations on the main theme. In order to keep the use case readable, please list only the most important alternative paths, and give only a short description for each of them (if an alternative path really requires a detailed description, then this is a separate use case).

## Business rules

*Business rules* are written (or unwritten) rules or policies that determine how an organization conducts its business with regard to a use case. Business rules are a special kind of requirement. Business rules may be specific to a use case or apply across all the use cases, or across the entire business. Use cases should clearly reference business rules that are applicable and where they are implemented.

Example of business rule for grid computing: If the system detects that an activity is keeping costly computing resources too long without really using them, the system may terminate the activity and free all its resources.

## Notes

Experience has shown that however well-designed a use case template is, the analyst will have some *important information* that does not fit under a specific heading. Therefore all good templates include a section (e.g. "Notes to Developers") that allows less-structured information to be recorded.

## Author and date

This section should list *when* a version of the use case was created and *who* documented it. It should also list and date any versions of the use case from an earlier stage in the development which are still current documents. The author is traditionally listed at the bottom, because it is not considered to be essential information; use cases are intended to be collaborative endeavors and they should be jointly owned.