Use Case Fundamentals

By Alistair Cockburn (http://alistair.cockburn.us)

Note from Alistair: This Use Case Fundamentals mini-document was sent by someone who had read the article "Structuring Use Cases with Goals", and wrote a digest of it for his group (thanks, Peter). It provides the following information.

- *A brief description of Use Cases, Actors and how they are used in the development process.*
- *A section on the Use Case methodology used along with a description of the terms used.*
- *A detailed description of the Use Case Reports generated during the Use Case analysis of the proposed system.*
- *A brief description of how to validate these Use Cases.*

What are Actors and Use Cases?

Actors are basically users of the system. They are actually user types or categories. Actors are external entities (people or other systems) who interact with the system to achieve a desired goal.

Use Cases are what happens when actors interact with the system. An actor uses the system to achieve a desired goal. By recording all the ways our system is used ("cases of use" or Use Cases) we accumulate all the goals or requirements of our system.

Therefore: A use case is a collection of possible sequences of interactions between the system under discussion and its Users (or Actors), relating to a particular goal. The collection of Use Cases should define all system behavior relevant to the actors to assure them that their goals will be carried out properly. Any system behavior that is irrelevant to the actors should not be included in the use cases.

There are many methods of defining how to pick or create a use case. The use cases in this report are generated using a goal oriented Structuring Methodology presented by Alistair Cockburn of Humans and Technology. Examining all the Actor's goals that the system satisfies yields the functional requirements. Goals summarize system function in understandable verifiable terms of use that users, executives and developers can appreciate and leave little open to interpretation.

Use Cases:

- Hold Functional Requirements in a easy to read, easy to track text format.
- Represents the goal of an interaction between an actor and the system. The goal represents a meaningful and measurable objective for the actor.
- Records a set of paths (scenarios) that traverse an actor from a trigger event (start of the use case) to the goal (success scenarios).
- Records a set of scenarios that traverse an actor from a trigger event toward a goal but fall short of the goal (failure scenarios).
- Are multi-level, one use case can use/extent the functionality of another.

Use Cases Do Not...

- Specify user interface design. They specify the intent, not the action Detail
- Specify implementation detail (unless it is of particular importance to the actor to be assured that the goal is properly met)

Use Cases are used during many stages of software development.

- To Capture the Requirements of the systems.
- To act as a spring board for the software design.
- To validate the software design against.
- For Software Test and Quality Assurance. (Tests are performed to validate proper and complete implementation of the use cases)
- Potentially as an initial framework for the on line help and user manual.

Use Case Definitions

Primary Actors:

The Actor(s) using the system to achieve a goal. The Use Case documents the interactions between the system and the actors to achieve the goal of the primary actor.

Secondary Actors:

Actors that the system needs assistance from to achieve the primary actors goal.

Use Case:

A collection of possible scenarios between the system under discussion and external actors, characterized by the goal the primary actor has toward the system's declared responsibilities, showing how the primary actor's goal might be delivered or might fail.

Use cases are goals (use cases and goals are used interchangeably) that are made up of scenarios. Scenarios consist of a sequence of steps to achieve the goal, each step in a scenario is a sub (or mini) goal of the use case. As such each sub goal represents either another use case (subordinate use case) or an autonomous action that is at the lowest level desired by our use case decomposition.

This hierarchical relationship is needed to properly model the requirements of a system being developed. A complete use case analysis requires several levels. In addition the level at which the use case is operating at it is important to understand the scope it is addressing. The level and scope are important to assure that the language and granularity of scenario steps remain consistent within the use case.

There are two scopes that use cases are written from: Strategic and System. There are also three levels: Summary, User and Sub-function.

Scopes: Strategic and System

Strategic Scope:

The goal (Use Case) is a strategic goal with respect to the system. These goals are goals of value to the organization. The use case shows how the system is used to benefit the organization.

These strategic use cases will eventually use some of the same lower level (subordinate) use cases.

System Scope:

Use cases at system scope are bounded by the system under development. The goals represent specific functionality required of the system. The majority of the use cases are at system scope. These use cases are often steps in strategic level use cases

Levels: Summary Goal , User Goal and Sub-function.

Sub-function Level Use Case:

A sub goal or step is below the main level of interest to the user. Examples are "logging in" and "locate a device in a DB". Always at System Scope.

User Level Use Case:

This is the level of greatest interest. It represents a user task or elementary business process. A user level goal addresses the question "Does your job performance depend on how many of these you do in a day". For example "Create Site View" or "Create New Device" would be user level goals but "Log In to System" would not. Always at System Scope.

Summary Level Use Case:

Written for either strategic or system scope. They represent collections of User Level Goals. For example summary goal "Configure Data Base" might include as a step, user level goal "Add Device to database". Either at System of Strategic Scope.

The Use Case Forms

Use Case Header: The forms contain a use case header that describes the use case using the following fields:

- Name: the name of the use case
- Goal: the goal of the use case in context of the scope and level
- Scope: the scope that the use case is covering
- Level: the level that the use case is written to
- Preconditions: any prerequisites before the use case can be started
- Success Condition: what is considered a successful end to the use case
- Failure Condition: what is considered a failed end to the use case
- Trigger: what external event happens that triggers the start of the use case
- Notes: Misc. notes regarding performance issues, frequency of use and other issues.

Following the Use Case Header is the collection of scenarios.

Scenario Header:

For Each Scenario in the Use Case the Scenario header lists the Scenario Name. Each scenario presents its list of steps.

Steps Section:

The Steps section of the report is defined by the following fields:

- Description: Description of the step usually written in a narrative form.
- Variation: Any variations to the step that don't substantially change the nature of the goal but are required for completeness.
- Notes: Misc. notes regarding performance issues, frequency of use and other issues
- Use Case: Since steps are essentially sub goals, if they requires more detailed elaboration a Use Case will be generated to detail (or satisfy) the step. This indicates which Use Case satisfies the step.

Each Step is assumed to be successful, if there is a significant failure condition for the step that will prohibit the continued success of the scenario then the step may contain and exception. The exception either ends the use case in failure or details the steps required to recover.

Exception Header:

The exception header can be present for each step and has two fields:

- Name: Describes the nature of the exception
- Return Step: Indicates for recovery cases which step in the scenario to return to.

The exception section then details each step required to recover.

Validating Use Cases

As stated before, Use Cases are used during many stages of software development.

- To Capture the Requirements of the systems.
- To act as a spring board for the software design.
- To validate the software design against.
- For Software Test and Quality Assurance. (Tests are performed to validate proper and complete implementation of the use cases)
- Potentially as an initial framework for the on line help and user manual.

Needless to say, it is very important that the Use Cases are validated thoroughly. As you are reading these use cases, ask the following questions...

- Is the Use Case complete? Are there any details that need to be added?
- Do I feel confident that the actor's goal is going to be properly met?
- Are there any procedural or requirement changes I can suggest that would simplify the process depicted in the Use Case.?
- Are there any additional Goals of the Actors that are not addressed?
- Are there any additional Actors that are not represented (directly or indirectly)?