Application Use Case Template

Working with templates

This document serves as the basis for detailing an application use case. The sections of this document should not be altered in any way. If a section is not applicable type the word 'N/A' into that section.

Project Name
Application Name
Document Name
Application Use Case
Version
July 6, 2009

Table of Contents

Table of Figures

1 Introduction

This section serves as an overview of the use case.

1.1 Purpose

The Application Use Case (AUC) document captures all the requirements for the Document Name use case.

1.2 Audience

For each role that is expected to review this document describe how this document is useful to their role.

1.3 Scope

This document contains a single use case for the Application Name system and its supporting requirements.

Contains an overview of the information in this document.

1.4 Definitions, Acronyms, and Abbreviations

See the glossary in ReqPro for a complete list of terms used on this project.

1.5 Goals

This section describes the purpose of this use case.

2 Document Name Use Case Details

This section details the use case.

2.1 Brief Description

A short description of the use case focusing on how it enables actors to achieve their goals.

Insert a use case diagram.

2.2 Usage

Indicate the normal and peak frequency that this use case can be expected to execute.

2.3 Primary Actor

State the primary actor's role in this use case.

2.4 Supporting Actors

'List all supporting actor roles in this use case.

2.5 Precondition(s)

This is the state of the system before the use case start. As such it is not checked during execution of the use case, instead it is assumed that it is not possible to execute the use case it the precondition has not been satisfied. (If the system displays an error because something has not previously occurred, do not list it as a precondition, but describe it in an exception flow.) In general a precondition indicates that some other use case has run to completion prior to this use case executing. In that sense a precondition can be seen as the post condition of some other use case. A common example of a precondition is the actor has logged onto the system.

2.6 Successful Post Condition

This is the condition of the system after the completion of the basic flow.

2.7 Alternate Post Conditions

This is the condition of the system after traversing an extension flow.

2.8 Basic Flow

1. <enter use case steps here>

Type each line of the use case using the 'Normal text' style. When complete, set each step to style 'Use Case Step'. This will automatically number the steps.

The basic flow is the normal flow of events that this use case is intended to execute.

If system does multiple things in no particular order in response to the actor's action, further split those into several substeps if it increases clarity, and, use bullet points to format the substeps (indicating that there is no order to the sequence in which the steps are executed).

Indicate the end of a flow by typing the words 'End use case'.

If several steps are to be repeated, write the repetition instruction after the repeating steps.

How to include a different use case:

When including another use case, reference the use case name and identifier in the step.

2.9 Alternate Flows

1.1.1.1.1.1.1.1.1 **<Enter alternate flow name here>**

2. <enter alternate steps here>

Alternative flow is as the name suggests a different path a use case can follow to reach a successful post condition. It starts with a condition (which is stated in the Alternative Flow title) and contains a set of steps describing what happens under that condition.

Usually an alternative flow ends by simply remerging with the basic flow at a particular step. When that is not the case, clearly indicate the end of the use case with the statement "Use case ends". Make sure the post condition of this flow is consistent wit the Successful Post Condition(s).

Enter alternate flow heading and text using the 'Normal body' style. Do not reference steps in the use case yet. Set the alternate flow heading to style 'Alternate Flow'. Set each alternate flow step to style 'Alternate Step'.

Where an alternate flow step makes reference to a step in the use case body, use the 'Insert->Reference->Cross Reference' command to reference the paragraph number.

Where an alternate flow is referenced from a basic step, use the 'Insert->Reference->Cross Reference' command to reference the paragraph number.

2.10 Extension Points

i. **<Enter extension flow name here>**

3. <enter extension steps here>

A list of locations within the flow of events of the use case at which additional behavior can be inserted.

Unless the action is trivial, document extensions to a use case in their own use case document. Indicate that the use case is extended by using the extend-relationship to connect the use cases in a use case diagram.

[Example use case:

2. Document Name Use Case

This section details the use case.

2.1 Description

The primary function of the dematerializer is to breakdown the cargo into particles that may be sent by the transmitter. A blueprint of the cargo is sent to the blueprint manager.

Figure 1 : Example Use Case Diagram

2.2 Peak Usage

The Dematerializer may need to be used up to 5 times a day.

2.3 Primary Actor

Cargo handler.

2.4 Secondary Actors

Transmitter

2.5 Precondition(s)

The Dematerializer is in a 'Ready To Transport' state.

2.6 Post Condition(s)

The Dematerializer is in a 'Ready To Transport' state.

2.7 Basic Flow

1 .System receives an 'Open Door' command.

2.The system opens the door.

3.(The cargo is loaded), The system receives a command to transport the cargo.

4. The transmitter is 'Ready to Transmit' and the system closes the door.

5. The door is closed and the system:

- *. secures the cargo.*
- *. creates a vacuum.*

6. The cargo is ready to be deconstructed and the system:

. sends a blueprint of the deconstructed cargo to the Blueprint Manager.

. deconstructs the cargo and sends the deconstructed matter to the Transmitter.

7. The cargo has been sent to the transmitter and system removes the vacuum.

The use case ends.

2.8 Alternate Flows

A.1 The transmitter is not ready:

8. At step 4, the system waits for the transmitter to indicate that it is ready.

9. Transmitter indicates that it is ready.

Return to step 4.

2.9 Extension Points

E.1 The system is configured to automatically shutdown after successful transmission.

10. At step 7 the system informs the user that it is shutting down.

11. The system shutsdown.

The use case ends.]

2.11 Activity Diagram

The following diagram represents the steps of the use case. The inputs are represented by control flows between activities. The activities represent the work done by the system. The initial state represents the precondition of the use case and the final states are the postconditions. Alternative and extension flows are represented by decisions. Where an alternative flow returns to the basic flow it is represented by a merge.

Place an activity diagram here that describes the use case flow.

Figure 2 : Example Activity Diagram

3 Use Case Performance Requirements

The following requirements apply to the steps of the use case. Every system step in the use case should have an associated timing. That timing may apply to an individual step or to a group of steps, so long as every step is covered.

<insert performance requirement here>

[Example

At step 2, the system door will open within 10 seconds.]

4 Use Case Supplementary Requirements

The following supplementary requirements impact this use case. They are gathered here during the detailing of the use case and should be moved to the supplementary folder of ReqPro once the use case has been imported to ReqPro.

<insert supplementary requirement here>

1 Interface Requirements

Where the use case interfaces with another actor, reference the appropriate standards for developing that interface. The reference may be to an industry standard, company standard or a 'to be developed' interface document.

[An interface should be described in its own separate interface document, and not in the use case. Interface requirements are not imported into ReqPro.]

<insert interface requirement here>