

Sample

Use Case

Step Method

**VERSION 1.0**

This template was created to enable departments to more easily develop their project plans. The Department of Technology, Consulting and Planning Division, created this template based on its experiences. The template relies on industry best practices combined with decades of experience on California state information technology projects. The way it was structured is to enable a department to complete the information related to its project without having to write background information related to the discipline. A department may use as much or as little of the template as it wishes.

**Template Instructions:**

* ***Instructions for completing*** this template – written for the author of the project plan - are encased in **[ ]** and the text is ***italicized*** *and* ***bolded.***
* *Examples* are provided as a guideline to the type of sample information presented in each section and the text is *italicized*.
* Boilerplatestandard language for each section is written in the document font and may be used or modified, as necessary.
* A department’s project specific information goes within the brackets ***<< >>***.
* *Informational text is italicized* within square brackets [ ] for informational purposes to the person who has to create the plan and includes background information, explanation, rationale, etc.

VENDOR PROJECT MANAGEMENT APPROVAL

DOCUMENT HISTORY

| **DOCUMENT APPROVAL HISTORY** | |
| --- | --- |
| Prepared By |  |
| Reviewed By |  |
| Approved By |  |

| **DOCUMENT REVISION HISTORY** | | | |
| --- | --- | --- | --- |
| Date | Document Version | Revision Description | Author |
| 01/06/2015 | 1.0 | Initial Version | R. Peterson |
|  |  |  |  |
|  |  |  |  |

TABLE OF CONTENTS

1. introduction 1

1.1. Overview 1

1.2. Background 2

2. Use Case Element descriptions 4

2.1. Summary section 4

2.2. Main Flow section 5

2.3. Alternate Flow Section 6

2.4. Exception Flow Section 7

3. State Requirements Use case Example 9

3.1. Use Case Diagram 9

3.2. UC-03 Approve XYZ Application 9

3.2.1. Summary Section 10

3.2.2. Main Flow 11

3.2.3. Alternative Flows 12

3.2.4. Exception Flows 13

3.2.5. use case Revision History 13

4. Detailed Requirements Use case example 14

4.1. Use Case Diagram 14

4.2. UC-o3 Approve XYZ Application 15

4.2.1. Summary Section 15

4.2.2. Main Flow 17

4.2.3. Alternative Flows 19

4.2.4. Exception Flows 20

4.2.5. Use Case Revision History 22

Appendix A: BPD-A-01 XYZ Applicant Approval Process (State Requirements Level) A-1

Appendix B: BPD-A-01 XYZ Applicant Approval Process (Detailed Requirements Level) B-1

Appendix c: Use Case Template C-1

# 

# introduction

## Overview

[*The objective of this document is to present the generally required content and format for two (2) different levels of Use Case instances[[1]](#footnote-1), though both Use Case instances represent the exact same functionality. The difference between the two is the level of specificity. The main body of this document provides information, instructions, and guidance on how to develop a Use Case; Appendix C provides a template for a Use Case.*

*The first Use Case, the State Requirements Level Use Case, is specified at a higher level and omits system specific “how” type information; this is the level of Use Case that would/should generally be written for a Request for Proposal (RFP) or other solicitation document by the state. The second level, Detailed Requirements Level, is more detailed and would/should be typically written by the contractor. This Detailed Requirements Level Use Case begins to 1) blend system specific information into the Use Case description, 2) reference more external specifications, and 3) integrate non-functional requirements in with the functional requirements; in general, the requirements and defined system behavior in a Detailed Requirements Level Use Case becomes more “concrete” rather than abstract. While this second level generally will not be written by state resources[[2]](#footnote-2), except for perhaps on internal projects, it is being presented here for two primary reasons: 1) to demonstrate the relationship that should exist between Use Cases written at different levels, and 2) to highlight the expected level of detail that a contractor/integrator-developed Use Case should contain.*

*In addition to the Use Case instance examples for two different levels of specificity, the appendices contain two business process diagrams, also written at different levels of specificity for the same process flow. One business process diagram is specified at the same level as, and is consistent with, the State Requirements Level Use Case and the other is at the same level as, and is consistent with, the Detailed Requirements Level Use Case.*

*As identified above, the objective of this document is to provide an understanding of Use Cases, what the components of a Use Case are, how a Use Case relates to business processes and business process diagrams, how Use Cases may document requirements at different levels of detail/specificity, and how different levels should be compared to ensure no state requirements for functionality, capabilities, behavior, etc. are lost.*

*It is important to note that Use Cases typically do not document all system requirements. Other types of requirements documents must also exist in order to fully cover the state’s needs. Additional requirements documents may include such items as: non-functional requirements, User Interface requirements, system interface requirements, training requirements, maintenance and operations (M&O) documentation requirements, etc., which may all be separate documents. All of these requirements must be documented to ensure the state’s requirements are complete. The commonly used and recommended approach to ensure all of the requirements have been identified is to use a product-oriented Work Breakdown Structure.]*

## Background

*[Use Cases, by far, are the current industry standard method for defining requirements. Use Case instances, which are a specific set of steps within a collection of a larger sequence of activities or events, not only identify the stakeholders’ specific needs and requirements but also put those requirements in a context of what is trying to be achieved by the stakeholders. Therefore, as a communications means to express and specifically identify what is required, Use Cases far exceed the older methods of just listing a large set of requirements, which typically do not have context between requirements (e.g., 1) The system shall …, 2) The system shall … , 3) The system shall ….). The bottom line is that a requirements document must communicate the needs of the stakeholders to the developers, who likely do not have in-depth knowledge of the stakeholders’ needs; Use Cases, and other scenario-based approaches do a better job at communicating these needs/requirements.*

*It is important to point out that Use Cases are primarily used to communicate functional requirements, though some non-functional requirements may also be communicated. However, in practice and reality, more information is needed for non-functional requirements than what can readily be provided in a Use Case in order to communicate the necessary non-functional requirement components (i.e., the source, stimulus, environment, artifact, response and response measurement components). The means for capturing and documenting non-functional requirements is documented in a separate template.*

*As stated above, this document presents two different levels of Use Cases, though both for the exact same functionality. One Use Case, called the State Requirements Use Case, represents the stakeholders’ requirements without regards to “how” the functionality will be implemented. In contrast, the second Use Case, called the Detailed Requirements Use Case, is a system-specific variation of the first Use Case as it begins to identify “how” certain functions will be performed (e.g., by clicking the “Save” button on screen GUI-XYZ-APRVL). This version is what the state should expect to receive from the developer/integrator upon their completion of the Requirements/Analysis phase of a project. At this point, there is a much clearer understanding and consistency between what the state requires and what the contractor is going to provide and how it will work/function.*

*One criticism of the Detailed Requirements Use Case is that it begins to include “design” work within the “requirements” document, which is a somewhat true statement. However, some level of “design” detail is required (e.g., there needs to be a save capability on a data entry screen), as it will, must, and should constrain the design. For example, a Detailed Requirements Use Case may identify a screen (specifically or generally) within an application, such as GUI-XYZ-APRVL, that performs a specific required function and one of the command buttons or features on the screen may be identified as “Save” to signify the saving of the data. The extent of this level of requirement constraining the design is that there must be a screen that displays the required information and somewhere on the screen there must be a “Save” command button or other feature that saves the data. The referenced design document (e.g., GUI-XYZ-APRVL, 1) must be consistent with the Use Case requirements, 2) may not be fully designed (e.g., the layout, list of elements being displayed, final names or labels for each element, etc.) and 3) may continue to evolve or change though when it is formally approved it must still be consistent with the requirements. So yes, the design is being somewhat constrained by the contractor themselves, though the full design is not constrained; careful naming through the use of pseudonyms or aliases greatly simplifies and reduces constraining the design.*

*Another major benefit of the Detailed Requirements Use Case document is that it greatly aids the development of test cases/scripts, which can be done in parallel with the remaining design and build efforts. During testing, the final design criteria will be examined to verify compliance (e.g., examine the GUI-XYZ-APRVL description to verify that all of the screen elements are displayed and the “Save” command button or other feature as defined in the GUI design once approved is present).]*

# Use Case Element descriptions

*[The following describes the four (4) primary sections of a Use Case, and the specific elements within each section: Summary, Main Flow, Alternate Flow, and Exception Flow. The intent of providing this information is to present an overview of what is involved in defining and creating a Use Case as well as to highlight key elements that must be identified and related within a Use Case, and between a Use Case and other Use Cases and respective documentation.]*

## Summary section

*[The Summary section is used to put the Use Case instance in context with stakeholder needs, a Use Case Diagram, showing multiple Use Case instances, and/or a Business Process Diagram. The Summary section identifies the Use Case by assigning it a unique identification number and name. The remainder of the Summary section is to put the Use Case in context with user interactions, identify the associations with other Use Case instances to place this Use Case in context in order to increase the understanding of the requirements defined within the Use Case, and identify separately defined items (e.g., letter layouts, screen views, interface specifications, etc.).]*

Summary Section:

[Use Case ID: A system-wide unique identifier for this Use Case; no two Use Cases should have the same ID. The ID should be consistent with a higher-level diagram, such as a Use Case Diagram and/or a Business Process Model reference. In practice, a Use Case ID will be used more than the Use Case Name; therefore, building some intelligence into the Use Case ID is beneficial (e.g., UC-G-01 for the first Use Case in the General category).

Use Case Name: A unique name for the Use Case that should begin with an action verb and associate to the business process step being described. The name should be consistent with a higher-level diagram, such as a Use Case Diagram and/or a Business Process Model reference.

Description: A textual description of the process step being described within this Use Case. Remember, this Use Case is about communicating the business needs to another person who does not know anything about the business need or how this step fits in so be complete but also be succinct.

Actors: Identify the people or things, outside of the system, that have a role or function during the execution of this specific Use Case. Be cautious when multiple actors are involved as it may signify that too many separate functions are being performed within one Use Case.

Triggers: Identify what triggers the execution of this Use Case. For example, sometimes a system generated event will trigger the start of a Use Case such as a batch job executes and moves a work item into someone’s work queue, an interface changes the state of an item and notifies someone that work can continue, an item is received by mail and needs to be entered into the system, etc. The best way to think of triggers is from a process flow perspective, meaning what in the previous process flow step(s) must have occurred in order to execute this next step of the process? Triggers are not the same thing as Pre-Conditions; Pre-Conditions set the stage while Triggers say “go”.

Pre-Conditions: In addition to the Trigger event occurring, identify what other pre-conditions are required to begin the execution of the Use Case. What you are doing here is setting the stage for the first step of the Use Case. So, if you expect an authorized user to be logged in, with sufficient security privileges to perform the action planned by this Use Case, and be at a specific screen, it should be identified here. Also include pre-conditions for the work that needs to be done within the Use Case.

Post-Conditions: Identify what the expected state of the work item and system is at the end of the Use Case. It is important is to cover all possible end states for the main flow; Alternate and Exception Flows may also be covered here or they may be covered in their respective sections of the Use Case.

Includes: Identify Use Cases that “would be” embedded into this Use Case but were broken out into separate Use Cases due to their commonality across multiple Use Cases. This normally occurs during Use Case Modeling. For example, the function to print a letter or correspondence may be a common function consisting of many steps that may appear in many different Use Cases. Therefore, a separate Use Case describing the steps to print a letter may be created and referenced as ‘Included’ in the other Use Cases versus repeating all of the steps in each Use Case.

Extension Points: Identify the separate Use Cases that are optional that extend the steps within this Use Case. Often, there are steps within a process that are optional and therefore not documented within the main flow of a Use Case or in another flow but are documented in a separate Use Case. For example, a process may include an optional step to call an applicant to provide information. Due to the sub-steps that need to be done when this optional step is performed, a separate Use Case may be created and that Use Case extends this Use Case.

(Note: The difference between Includes and Extends is that Includes Use Cases are generally executed as if they are actually part of the main Use Case while Extends Use Cases are not always executed but only optionally or conditionally executed (i.e., they extend the main Use Case when necessary.)

References: Identify specific references to other requirements that are defined to support the Use Case, such as Business Process Diagrams, Use Case Diagrams, Screen designs, Data Dictionary, letters, files, etc. Also include predecessor and successor Use Cases to help maintain business context for the big picture goal or objective.]

## Main Flow section

*[The Main Flow section is where the step-by-step description of the stakeholders’ interactions with the system and the system’s required responses are documented. There are three main columns for this section.]*

Main Flow Section:

[Step Column: A unique and ordered number for each step should be identified, which must be sequential, (e.g., 1 🡪 2🡪3🡪4, or 1🡪3🡪7🡪12, or A🡪B🡪C🡪D, or A🡪D🡪G🡪M, or 1.0🡪1.1🡪1.2🡪2.0, etc.). Many organizations and tools will increment by more than 1 between successive steps to allow the insertion of additional steps without forcing a renumbering of all of the steps after the newly inserted step (e.g., instead of 1 🡪 2 🡪3 they may use 1 🡪4 🡪7). This is useful when there is a significant amount of branching between the Main Flow and Alternate or Exception Flows to avoid re-numbering for all of the branching points and when performing detailed traceability between Use Cases and Test Scripts, however, this is not absolutely necessary.

Action/Cause/Stimulus Column: As stated previously, the Trigger indicates that this Use Case should be executed and the Pre-Condition defines all of the conditions necessary to get to this first step. Within this column, all of the steps necessary to complete the defined Use Case must be identified. When documenting the steps, they must be in process-step order and they must include every step, not just the steps that involve the IT system. To the extent possible, attempt to keep the actions as small as possible yet large enough to cause a measureable, testable, verifiable reaction. For example, a step to read an applicant’s name and another step to enter an applicant’s name into the system is too small because the first step is not measureable.

*Reaction/Effect/Response Column: For each Action/Cause/Stimulus, there must be a Reaction/Effect/Response that is measureable, testable, and/or verifiable. Also note that the response may or may not be an IT system generated response but something from outside the IT system. Also, the response may be subtle, such as the information appears on the screen, or something more significant, such as a payment gets authorized and queued for submission to the SCO. Within this column is where any redirection to Alternate Flows, Exception Flows, Included and/or Extended Use Cases may be identified.]*

## Alternate Flow Section

*[An Alternate Flow is simply an alternate path through a business process that is too large to be handled with a single step within the Main Flow and not an Exception. For example, a business process may include a decision point to determine if an applicant is a California resident. The Main Flow may assume the applicant is a California Resident. An Alternate Flow may require some additional steps be performed and then continue on with processing the applicant.]*

Alternate Flow Section:

*[Branched From: Identify the flow and the step that invokes this Alternate Flow. If more than one point branches to this flow, be clear and consistent with respect to naming the Branch-From and Branch-To fields.*

*Flow Scenario: Specify a Number and Name for this flow that is unique and conveys meaning related to the function being performed within the flow.*

*Post-Condition: Identify what the expected state of the work item and system is at the end of the Alternate Flow. It is important to cover all possible end states for Alternate Flow.*

*Branch To: Once this Alternate Flow completes, it must return to the branching flow, though not necessarily to the same step. Clearly identify to which step in the flow this Alternate Flow returns.*

Step Column: A unique and ordered number for each step should be identified, which may or may not be sequential, as long as they are ordered.

Action/Cause/Stimulus Column: Within this column, all of the steps necessary to complete the Alternate Flow must be identified. When documenting the steps, they must be in process-step order and they must include every step, not just the steps that involve the IT System. To the extent possible, attempt to keep the actions as small as possible yet large enough to cause a measureable, testable, verifiable reaction. For example, a step to read an applicant’s name and another step to enter an applicant’s name into the system is too small because the first step is not measureable.

*Reaction/Effect/Response Column: For each Action/Cause/Stimulus, there must be a Reaction/Effect/Response that is measureable, testable, and/or verifiable. Also note that the response may or may not be an IT system generated response but something from outside the IT system. Also, the response may be subtle, such as the information displays on the screen, to something more significant, such as a payment gets authorized and queued for submission to the SCO.]*

## Exception Flow Section

*[The Exception Flow section is similar to the Alternate Flow section with one major difference. Exception Flows are used when something goes wrong, either in the Main Flow or any other flow. For example, if the system cannot write data to the database an Exception Flow may be executed. For this example, while database read errors may be handled as an Extension to the Use Case, database write errors are commonly handled as Exceptions because they must be dealt with locally on a case-by-case basis (e.g., what to do if some data was written from a previous step, but not the data from the current step). Generally, Exception Flows terminate the Use Case Main Flow and therefore do not specify a Branch-To point to return to in order to continue execution; otherwise, they are identical.]*

Exception Flow Section:

*[Branched From: Identify the flow and the step that invokes this Exception Flow.*

*Flow Scenario: Specify a Number and Name for this flow that is unique and conveys meaning related to the function being performed within the flow.*

*Post-Condition: Identify what the expected state of the work item and system is at the end of the Exception Flow. It is important to cover all possible end states for Exception Flow.*

Step Column: A unique and ordered number for each step should be identified, which may or may not be sequential, as long as they are ordered.

Action/Cause/Stimulus Column: Within this column, all of the steps necessary to complete the Exception Flow must be identified. When documenting the steps, they must be in process-step order and they must include every step, not just the steps that involve the IT System. To the extent possible, attempt to keep the actions as small as possible yet large enough to cause a measureable, testable, verifiable reaction. For example, a step to read an applicant’s name and another step to enter an applicant’s name into the system is too small because the first step is not measureable.

*Reaction/Effect/Response Column: For each Action/Cause/Stimulus, there must be a Reaction/Effect/Response that is measureable, testable, and/or verifiable. Also note that the response may or may not be an IT system generated response but something from outside the IT system. Also, the response may be subtle, such as the information appears on the screen, to something more significant such as a payment gets authorized and queued for submission to the SCO.]*

# State Requirements Use case Example

## Use Case Diagram

*[An abbreviated Use Case Diagram is presented in Figure 1. Within this figure there are two specific Use Cases instances identified, UC-03 Approve XYZ Application and UC-13 Print and Mail Letter, and one general Use Case, Others. While a full discussion of a Use Case Diagram is beyond the scope of this document, at a high-level, a Use Case Diagram is a diagram that shows the actors (someone or something outside the system that interacts with the system), the use cases (a complete task of a system that provides a measurable result of value for an actor), and their relationships (of which there are four primary types used in a Use Case Diagram: Association, Generalization, Includes, and Extends).*

*The Use Case Diagram presented here shows that an actor (User) interacts with UC-03 Approve XYZ Application and that this Use Case Includes UC-13 Print and Mail Letter in order to accomplish a complete task within the System. The following is the Use Cases instance description of UC-03.]*



Figure 1: State Requirements Use Case Diagram

## UC-03 Approve XYZ Application

*[The following is a Use Case instance for UC-03 Approve XYZ Application. As noted above, this Use Case is written from a State Requirements level and it defines the requirements for the business process identified in Appendix A: BPD-A-01 XYZ Applicant Approval Process (State Requirements Level). The actual Use Case Instance document is between the State Requirements Use Case UC-03 Beginning label and the State Requirements Use Case UC-03 End label.]*

***State Requirements Use Case UC-03 Beginning***

### Summary Section

| Item | Description |
| --- | --- |
| Use Case ID: | UC-03 |
| Use Case Name: | Approve XYZ Application |
| Description: | This use case describes the process steps that must be performed in order to approve an XYZ application. |
| Actors: | * Internal User – XYZ Review Staff with privileges to approve the application |
| Triggers: | 1. The application has been assigned to the Staff member for approval by their Supervisor |
| Pre-Conditions: | 1. The Staff member assigned to approve the application has access to the IT system and has sufficient privileges to approve the application. 2. The internal user is logged into the system and is viewing a screen that shows a list of work items ready for approval. |
| Post-Conditions: | **Main Flow:**   1. The stored application data will be marked as Approved. 2. The Approval Letter will be sent to be printed and mailed to the applicant. 3. Application will be removed from Staff members list of items waiting for approval.   **Alternate Flow:**  See A1 – Application Deficient  **Exception Flows:**  None |
| Includes: | UC-13 Print and Mail Letters |
| Extension Points: | None |
| References | BPD-A-01 XYZ Application Process (State Requirements Level)  XYZ Application Approval Policy (External written policy for manual review)  XYZ Application Approval Letter  XYZ Application Deficiency Letter |

### Main Flow

| Step | Action/Cause/Stimulus | Reaction/Effect/Response |
| --- | --- | --- |
| 1 | The user shall select an application from the list of applications waiting for approval that have been assigned to the user. | The system shall display the selected application data necessary for the user to verify that the application data complies with the requirements identified in the XYZ Application Policy document |
| 2 | The user shall record the results of his/her manual review of the application in the system by identifying if the application passed the manual review or by identifying that the application is deficient. | If the application passed the manual review, the system shall display the user confirmation that the manual review passed but shall not yet store this data with the application record at this time.  If the user identified that the application is deficient, **Alternate Flow** A1 – Application Deficient shall be executed and this Main Flow terminated. |
| 3 | If the application manual review passed, the user shall initiate an automated review of the application data by the system to verify that the system stored data complies with the XYZ Application Approval Policy requirements and system defined data consistency, integrity, quality, and security requirements. | The system shall verify that the stored data complies with XYZ Application Approval Policy requirements and system derived data consistency, integrity, quality, and security requirements and shall display the results for each verification check to the user as well as an overall pass or fail result. |
| 4 | The User shall review the results of the automated verification and shall elect to either save the results of the verification, or to not save the results. | If the user elects to save the results, the system shall save the results of both the manual and automated verification with the application record.  If the user elects to not save the results, no verification data shall be stored. |
| 5 | The user shall then elect to approve the application or to not approve the application. | If the user elects to approve the application, the system shall verify that all required data for an application has been stored, shall verify that all stored data meets the requirements for approval as defined in the XYZ Application Approval Policy, system derived data consistency, integrity, quality, and security requirements, and that manual and automated verification have passed. The system shall then generate an XYZ Application Approval letter for printing and mailing to the applicant, UC-13, shall remove the application for the list of applications waiting for approval, and shall return the user to the display showing an updated list of applications waiting for approval. The system then proceeds to Step 7.  If the user elects not to approve the application, execute Step 6. |
| 6 | If the user elects to not approve the application, the user shall be prompted by the system to identify if the application is deficient or if they want to stop the approval process for this application. | If the user identifies that the application is deficient, **Alternate Flow** A1 – Application Deficient shall be executed.  It the user identifies that they want to stop processing this application, the system shall return the user to the display showing an updated list of applications waiting for approval. The system then proceeds to Step 7. |
| 7 | Flow ends |  |

### Alternative Flows

#### Alternate Flow A1 – Application Deficient

| Branched From: | Main Flow, Step 3. |
| --- | --- |
| Flow Scenario: | A1 – Application Deficient |
| Post-Condition: | 1. A letter shall be generated that identifies the application’s deficiencies that shall be printed and mailed to the applicant. 2. The Main Flow terminates. |
| Branch To: | Main Flow Step 7 |

| Step | Action/Cause/Stimulus | Reaction/Effect/Response |
| --- | --- | --- |
| A1-1 | Upon entry into this Alternate Flow, the user shall be presented a display that shows all of the possible causes for an application to be deficient based on the verification rules defined in the XYZ Application Approval Policy and the manual and automated verification checks performed. | The system shall update the display to identify which items are deficient based on the stored data and the results from the manual and automated verification checks that include the verification of the XYZ Application Approval Policy requirements. |
| A1-2 | The user shall override the system identified areas of deficiency in order to 1) identify additional areas as deficient from the displayed list even though they were not identified by the system, 2) de-identify or de-select system identified areas of deficiency, and 3) add textual language to identify the specific deficiencies related to the manual review of the application. | The system shall update the displayed areas of deficiency presented on the screen based on the user’s selections to add areas of deficiency, remove/delete areas of deficiency, and provide a textual description of the deficiencies related to the manual review. |
| A1-3 | The user shall then request for the system to generate an XYZ Application Deficiency Letter to be printed and mailed to the applicant. | Upon the user requesting the letter be printed and mailed, UC-13, the system shall save the data displayed on the screen with the application, save the request for letter generation action, and then print and mail the letter. |
| A1-4 | Flow ends |  |

### Exception Flows

| Branched From: | None |
| --- | --- |
|  |  |
| Post-Condition: | None |

| Step | Action/Cause/Stimulus | Reaction/Effect/Response |
| --- | --- | --- |
| E1-1 | None | None |
| E1-2 |  |  |
| E1-3 |  |  |

### use case Revision History

| Date | Version | Description | Author |
| --- | --- | --- | --- |
| 11/03/2014 | 1.0 | Initial Baseline Version | R. Peterson |
|  |  |  |  |

**State Requirements Use Case UC-03 End**

# Detailed Requirements Use case example

## Use Case Diagram

*[Figure 2 below is an abbreviated Use Case Diagram for the Detailed Requirements level. Notice that this Use Case Diagram is different from the diagram for the State Requirements level shown in Figure 1. The major difference is that the one Use Case and Includes relationship shown in Figure 1 for printing and mailing the letters, UC-13, is no longer shown in Figure 2; however, a new Use Case, UC-G-01, is Included in Figure 2 that was not Included in Figure 1. Also, Figure 2 shows and Extends Use Case (UC-07) and relationship that was not shown in Figure 1.*

*While there may be numerous reasons for the differences, all of the differences must be reconciled and understood during the State’s review of the Detailed Requirements level Use Cases to ensure requirements are not lost. For example, the State should question what happened to the included Use Case for printing and mailing letters (UC-13) identified in the State Requirements level Use Case Diagram; in this example, a business rule referenced in the Detailed Requirements Use Case addresses this functionality (i.e., it is performed as a business rule and not as a separate Use Case). Similarly, the State should question where UC-07 View XYZ Application Details came from; this may be answered that per user interviews, the users identified that sometimes they need to review the application details when approving an application (i.e., it is a derived requirement after talking with the stakeholders and creating the Detailed Requirements Level Business Process Flow for the Use Case). And since the functionality may have already existed as part of another Use Case, UC-07 View XYZ Application Details, and the functionality is not always required to be performed but only under special circumstances, it was identified as an extended Use Case. The State must reconcile and understand all differences to ensure the defined requirements are correct and consistent with the State requirements specified in the State Requirements Level documents and no requirements are lost or un-necessary requirements added.]*



Figure 2: Detailed Requirements Use Case Diagram Example

## UC-o3 Approve XYZ Application

*[The following is a Use Case instance for UC-03 Approve XYZ Application. As noted above, this Use Case is written from a Detailed Requirements level and it defines the requirements for the business process identified in Appendix B: BPD-A-01 XYZ Applicant Approval Process (Detailed Requirements Level). The actual Use Case Instance document is between the Detailed Requirements Use Case UC-03 Beginning label and the Detailed Requirements Use Case UC-03 End label.]*

***Detailed Requirements Use Case UC-03 Beginning***

### Summary Section

|  | Description |
| --- | --- |
| Use Case ID: | UC-03 |
| Use Case Name: | Approve XYZ Application |
| Description: | This use case describes the process steps that must be performed in order to approve an XYZ application. |
| Actors: | * Internal User – XYZ Review Staff with privileges to approve the application |
| Triggers: | 1. The application has been assigned to the Staff member’s Work Queue for approval by the Supervisor. |
| Pre-Conditions: | 1. The Staff member assigned to approve the application has access to the IT system and has sufficient privileges to approve the application. 2. The internal user is logged into the system and is viewing the GUI-XYZ-WORKQ screen that shows work items ready for approval. |
| Post-Conditions: | **Main Flow:**   1. The application will be marked as Approved with the XYZ Applicant data record, DD-XYZ-APP, updated in accordance with BR-XYZ-APP-APRVL-UPDATE XYZ Application Approval Update. 2. Letter LTR-XYZ-APP-APRVAL will be queued up for printing to be generated and mailed to the applicant in accordance with PRINT-XYZ-APP-APRVAL. 3. Application will be removed from Staff members Pending Approval work queue.   **Alternate Flow:**  See A1 – Application Deficient  **Exception Flows:**  See E1 – Application Save Error  See E2 – Approval Rule Violation  See E3 – Deficiency Letter Violation |
| Includes: | UC-G-01 Database Read Access (Exceptions) |
| Extension Points: | UC-07 View XYZ Application Details |
| References | BPD-A-01 XYZ Application Process (Detailed Requirements Phase)  BR-XYZ-APP-VERIFY XYZ Application Verification  BR-XYZ-APP-APRVL-UPDATE XYZ Application Approval Update  BR-XYZ-APP-DEFCNT-UPDATE XYZ Application Deficient Update  BR-XYZ-APP-VER-RESULT-UPDATE XYZ Applicant Verification Results Update  BR-XYZ-APP-DEFCNT-SEND Application Deficiency Check for Letter  GUI-XYZ-APP-APRVL XYZ Application Approval Screen  GUI-XYZ-APP-VERIFY XYZ Automated Application Verification Screen  GUI-XYZ-WORKQ User Work Queue Screen  GUI-XYZ-APP-DEFCNT XYZ Application Deficiency Screen  GUI-XYZ-APP-DEFCNT-MSG Application Deficiency Message Screen  LTR-XYZ-APP-APRVAL XYZ Application Approval letter  LTR-XYZ-APP-DEFCNT XYZ Application Deficient Letter  PRINT-XYZ-APP-DEFCNT Print XYZ Application Deficiency Letter  PRINT-XYZ-APP-APRVAL Print XYZ Application Approval Letter  ERR-LOG-DB-SAVE Report for Database Read Errors  XYZ Application Approval Policy (External written policy for manual review)  DD-XYZ-APP XYZ Data Dictionary for Applicant Record (Non-specific reference) |

### Main Flow

| Step | Action/Cause/Stimulus | Reaction/Effect/Response |
| --- | --- | --- |
| 1 | The Internal User is viewing the GUI-XYZ-WORKQ screen and selects an applicant from their Work Queue of Pending Approval applications by double-clicking the mouse cursor on the applicant’s name. | The XYZ Applicant Approval screen, GUI-XYZ-APP-APRVL, displays the showing the GUI defined information for the applicant selected. |
| 2 | The Internal User reviews the displayed information to determine if the applicant’s information complies with the current business policy requirements defined in the XYZ Application Approval Policy. | The user is able to review all required data referenced in the XYZ Application Approval Policy and to determine if the application complies with the Policy requirements based on the data displayed.  **Extension:** UC-07 View XYZ Application Details – If the user chooses to review additional details of the application, the user may select the “Application Information” option on the GUI-XYZ-APP-APRVL screen that will execute this Extension. (Upon the users review, the user will return to this Step.) |
| 3 | Upon the user’s manual review of the application, the user will confirm that the review was completed and the application meets the current XYZ Application Approval Policy requirements by clicking on the “Manual Review” checkbox. | If the user confirmed that the application meets the Policy requirements, a check mark will be displayed next to the “Manual Review” item shown on the screen.  If the user determines that the application does not comply with the current Policy requirements, the user will click the “Deficient” command button on the screen and **Alternate Flow:** A1 – Application Deficient shall be followed. |
| 4 | If the application complies with the XYZ Application Approval Policy and the “Manual Review” confirmation is displayed on the GUI-XYZ-APP-APRVL screen, the Internal User shall initiate the automated verification of the application by selecting the “Verify” command button on the screen to ensure the business rules have been met by this application. | The system verifies that the application is complete by verifying the requirements identified in BR-XYZ-APP-VERIFY XYZ Application Verification and a new window opens that displays the GUI-XYZ-APP-VERIFY screen showing the required screen data for this screen and the specific application being verified. (Note: no additional exceptions generated for this Business Rule (BR).) |
| 5 | The Internal User selects to either save the results or to not save the results of the automated verification by selecting the “Save” or “Cancel” (not save) command on the GUI-XYZ-APP-VERIFY screen. | Upon selection to save the results, the results of the manual and automated verification are stored with the applicant’s record in accordance with BR-XYZ-APP-VER-RESULT-UPDATE XYZ Applicant Verification Results Update, GUI-XYZ-APP-VERIFY closes, the user is returned to the GUI-XYZ-APP-APRVL screen, and the applicant’s data displayed on the screen is refreshed from the database showing the correct Manual and Automated verification status.  **Exception:** E1 Application Save Error – If there is an error in saving the application data, this exception shall be executed.  Upon selection of “Cancel”, the applicant’s record is not updated, GUI-XYZ-APP-VERIFY closes, the user is returned to the GUI-XYZ-APP-APRVL screen, and the applicant’s data displayed on the screen is refreshed from the database showing the correct Manual and Automated verification status. |
| 6 | The Internal User will then confirm that the application is approved by clicking on the “Approved” command button or if the application is deficient by clicking on the “Deficient” command button or if the user wants to stop and abandon the approval process by clicking the “Stop” command button. | Upon selecting “Approved”, BR-XYZ-APP-APRVL-UPDATE Application Approval Update will be executed to verify that the currently saved data meets the requirements for approval and the application record will be update to record the approval action. The LTR-XYZ-APP-APRVAL letter will be generated to notify the applicant of the approval action, which will be printed in accordance with PRINT-XYZ-APP-APRVAL. The GUI-XYZ-APP-APRVL screen will close and the user will be returned to the GUI-XYZ-WORKQ screen, the displayed data will be refreshed, and the applicant no longer appearing on the Pending Approval Work Queue.  **Exception:** E1 Application Save Error – If there is an error in saving the application data, this exception shall be executed.  **Exception:** E2 Approval Rule Violation – If the requirements for approval as defined in BR-XYZ-APP-APRVL-UPDATE are not met then this exception shall be executed.  If “Deficient” is selected, Alternate Flow A1 – Application Deficient shall be followed.  If “Stop” is selected, the GUI-XYZ-APP-APRVL screen will close and the user will be returned to the GUI-XYZ-WORKQ screen. |
| 7 | Flow ends |  |

### Alternative Flows

#### Alternate Flow A1 – Application Deficient

| Branched From: | Main Flow, Step 3. |
| --- | --- |
| Flow Scenario: | A1 – Application Deficient |
| Post-Condition: | 1. LTR-XYZ-APP-DEFCNT, Application Deficient Letter, will be generated identifying the deficiencies within the application preventing it from approval. 2. A record of the Letter generation and the deficiencies shall be stored with the application in accordance with BR-XYZ-APP-DEFCNT-UPDATE. 3. The Main Flow terminates, the user is returned to the GUI-XYZ-WORKQ screen, the screen is refreshed from the database that no longer shows the applicant in the Pending Approval queue. |
| Branch To: | Main Flow Step 7 |

| Step | Action/Cause/Stimulus | Reaction/Effect/Response |
| --- | --- | --- |
| A1-1 | Upon entry into this Alternate Flow, BR-XYZ-APP-VERIFY executes, verifying the application information, and GUI-XYZ-APP-DEFCNT opens in a new window and displays the status of all GUI required application deficiency items. | The GUI-XYZ-APP-DEFCNT screen opens in a new window and all application deficiencies are correctly identified in accordance with BR-XYZ-APP-VERIFY. |
| A1-2 | The Internal User reviews the screen items, verifies that they are correct, clears the check mark for identified deficiencies that are not correct by clicking on the deficiency item checkbox, and adds deficiency check marks to deficiency items that are not checked by clicking on the deficiency item. | The check marks for the deficient items are set or not set based on the user’s actions. |
| A1-3 | If the “Manual Review” deficiency item is checked, the use shall enter a description of the deficiency identified during the manual review into the text box field displayed on the screen. | The user shall be able to enter a textual description of the deficiency up to the maximum length of the text field as defined in GUI-XYZ-APP-DEFCNT. |
| A1-4 | Upon the Internal User final review, the user shall elect to either generate a letter to be mailed to the applicant, by clicking “Send” or to stop all work on the application and abandon all work performed. | If the user selects “Send”, the system will verify that all conditions have been met by executing BR-XYZ-APP-DEFCNT-SEND and if met then BR-XYZ-APP-DEFCNT-UPDATE shall be executed, and the print job will be processed to generate XYZ-LTR-APP-DEFCNT in accordance with PRINT-XYZ-APP-DEFCNT.  **Exception:** E3 Deficiency Letter Violation – If BR-XYZ-APP-DEFCNT-SEND fails, then this exception shall be executed.  **Exception:** E1 Application Save Error – If there is an error in saving the application data, this exception shall be executed.  If the user selects “STOP”, no additional data shall be saved and this Alternate Flow shall terminate. |

### Exception Flows

#### Exception Flow E1 – Application Save error

| Branched From: | Main Flow, Step 5 and 6  Alternate Flow A1 – Application Deficient, Step A1-4 |
| --- | --- |
| Flow Scenario: | If any or all of the data is not saved, then none of the data is to be saved. |
| Post-Condition: | 1. The data within the database is unchanged and consistent with the state of the database prior to the step being executed. 2. An exception report is written into an error log in accordance with ERR-LOG-DB-SAVE requirements. 3. All future steps within the Use Case are bypassed and the Main Flow ends. |

| Step | Action/Cause/Stimulus | Reaction/Effect/Response |
| --- | --- | --- |
| E1-1 | The system shall detect a failure to write data to the database for the identified steps for this exception. | The system shall roll back any data previously written for the Main Flow step being executed, not for all of the Main Flow. |
| E1-2 | The system shall detect the reason for the failure and all additional data necessary to generate the error log in accordance with ERR-LOG-DB-SAVE. | The system shall write the error log in accordance with ERR-LOG-DB-SAVE. |
| E1-3 | Flow ends |  |

#### Exception Flow E2 – Approval rule violation

| Branched From: | Main Flow, Step 6. |
| --- | --- |
| Flow Scenario: | The user attempted to approve an XYZ application, after performing manual and automated verification but the system identified that the requirements for the approval of an XYZ application as defined in BR-XYZ-APP-APRVL-UPDATE were not met, which generated this exception. |
| Post-Condition: | 1. No data is written to the database for the Main Flow step being executed. 2. No additional Main Flow steps are executed. 3. Alternate Flow Application Deficient is initiated. |

| Step | Action/Cause/Stimulus | Reaction/Effect/Response |
| --- | --- | --- |
| E2-1 | BR-XYZ-APP-APRVL-UPDATE identifies a deficiency with the application. | Any data written as part of the Main Flow step is rolled back and no changes to the database will have been made.  The system captures the deficiency detected by BR-XYZ-APP-APRVL-UPDATE and then begins the execution of Alternate Flow A1 – Application Deficient. |
| E2-2 | Flow ends |  |

#### Exception Flow E3 – Deficiency Letter Violation

| Branched From: | Alternate Flow A1 – Application Deficient, Step A1-4. |
| --- | --- |
| Flow Scenario: | If the conditions necessary to generate an XYZ Application Deficiency letter fails, as verified by BR-XYZ-APP-DEFCNT-SEND, then this exception shall be executed. |
| Post-Condition: | 1. The user will be notified of the deficiency causing the letter not to be generated. 2. The user will be returned to Alternate Flow A1 – Application Deficient, Step A1-4. 3. No changes shall be made to the database or any log files. |

| Step | Action/Cause/Stimulus | Reaction/Effect/Response |
| --- | --- | --- |
| E3-1 | The system detects that the data stored in the database for the application and the checkboxes checked on the GUI-XYZ-APP-DEFCNT screen are not consistent or do not meet the approval requirements as defined by BR-XYZ-APP-DEFCNT-SEND. | No data is written to the database or any log files.  The system displays GUI-XYZ-APP-DEFCNT-MSG in a new window the reason(s) for the failure in accordance with the GUI requirements. |
| E3-2 | The user reviews the reasons for the failure and closes the message screen by selecting “Close”. | The GUI-XYZ-APP-DEFCNT-MSG screen closes and the user is returned to the GUI-XYZ-APP-DEFCN screen where the user can then correct the problem or terminate sending the letter. |
| E3-3 | Flow ends and the user is returned to Alternate Flow Step A1-4. |  |

### Use Case Revision History

| Date | Version | Description | Author |
| --- | --- | --- | --- |
| 11/3/2014 | 1.0 | Initial Baseline Version | R Peterson |
|  |  |  |  |

***Detailed Requirements Use Case UC-03 End***

# Appendix A: BPD-A-01 XYZ Applicant Approval Process (State Requirements Level)



# Appendix B: BPD-A-01 XYZ Applicant Approval Process (Detailed Requirements Level)



# Appendix c: Use Case Template

**Summary Section**

|  | Description |
| --- | --- |
| Use Case ID: |  |
| Use Case Name: |  |
| Description: |  |
| Actors: |  |
| Triggers: |  |
| Pre-Conditions: |  |
| Post-Conditions: | **Main Flow:**  **Alternate Flow:**  **Exception Flow:** |
| Includes: |  |
| Extension Points: |  |
| References: |  |

**Main Flow**

| Step | Action/Cause/Stimulus | Reaction/Effect/Response |
| --- | --- | --- |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |
| 6 |  |  |
| 7 |  |  |

**Alternative Flows**

#### Alternate Flow A1 – XYZ

| Branched From: |  |
| --- | --- |
| Flow Scenario: |  |
| Post-Condition: |  |
| Branch To: |  |

| Step | Action/Cause/Stimulus | Reaction/Effect/Response |
| --- | --- | --- |
| A1-1 |  |  |
| A1-2 |  |  |
| A1-3 |  |  |
| A1-4 |  |  |

**Exception Flows**

#### Exception Flow E1 – ABC

| Branched From: |  |
| --- | --- |
| Flow Scenario: |  |
| Post-Condition: |  |

| Step | Action/Cause/Stimulus | Reaction/Effect/Response |
| --- | --- | --- |
| E1-1 |  |  |
| E1-2 |  |  |
| E1-3 |  |  |

**Use Case Revision History**

| Date | Version | Description | Author |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

1. A Use Case instance is simply a specific instance of a use case that documents the behavior of the subjects and interactions between actors and the use case. [↑](#footnote-ref-1)
2. For certain acquisition strategies when a legacy system is being replaced, the state might write Detailed Use Cases for the legacy system plus new requirements and require the contractor to transform these into new Detailed Use Cases for the new system. [↑](#footnote-ref-2)