

CBD/e White Papers

Use Case Techniques and Considerations

Introduction to Use Case Analysis

Richard Woolridge, Castek

[Editor's Note: Welcome to a new series of articles about the CBD/e analysis and design methodology for developing component-based applications. If you are unfamiliar with CBD/e, click here to see an introduction.]

This series begins by describing how CBD/e employs use cases in the analysis of systems to discover the business processes and system requirements. This, the first article in the series, provides an introduction to use cases. It defines a use case and the types of use cases that exist, and it outlines the objectives of use case analysis.]

What is a Use Case?

Use Case analysis is one of the first and primary means of gathering requirements in the behavioral methodology. Use cases are a standard technique for gathering requirements in many modern software development methodologies. In fact, use cases are included in the Unified Modeling Language (UML) which has become the *de facto* industry standard software artifact notation.

The UML is "a language for specifying, visualizing and constructing the artifacts of software systems . . ." [Booch97]. UML is not a methodology. Any method can be used to gather software artifacts that are represented in UML as long as the meaning of those artifacts comply with the definition in the used notation.

Definition

Use cases in UML are defined in various but similar ways within the literature.

"A use case is a narrative document that describes the sequence of events of an actor (an external agent) using a system to complete a process." [Jacobson92]

"They are stories or cases of using a system. Use case are not exactly requirements or functional specifications, but they illustrate and imply requirements in the stories they tell." [Larman98]

"...domain processes can be expressed in use cases – narrative descriptions of domain processes in a structure prose format." [Larman98]

"A description of set of sequences of actions, including variants, that a system performs that yield an observable result of value to an actor." [Booch99]

"You apply use cases to capture the intended behavior of the system you are developing, without having to specify how that behavior is implemented. Use cases provide a way for your developers to come to a common understanding with your system's end users and domain experts. In addition, use cases serve to help validate



your architecture and to verify your system as it evolves during development.”[Booch99]

This paper focuses on the development of business applications and, as such, the definition of a use case for this paper is specific to the business application domain. Given that perspective, the following is the definition of a use case:

“The list of steps, manual and automated, necessary to accomplish the business goal of the use case.”

The definition is made clearer by defining types of use cases and explaining when each type is valuable.

Types of Use Cases

There are two types of use cases. One type of use case is called an **Essential Use Case** [Constantine97] and the other type of use case is called a **Real Use Case** [Larman98]. These use case types are defined below:

“**Essential Use Cases** . . . are expressed in an ideal form that remains relatively free of technology and implementation detail; design decisions are deferred and abstracted, especially those related to the user interface.”[Larman98]

“ . . . a **Real Use Case** concretely describes the process in terms of its real current design, committed to specific input and output technologies, and so on. When a user interface is involved, they often show screen shots and discuss interaction with the widgets.”[Larman98]

Essential use cases are of primary importance early in a project’s analysis. Their purpose is to document the business process that the system must support without bias to technology. Later, during project design, real use cases become important since they document how a specific set of user interfaces will support the business process documented in the essential use case. This paper will focus entirely on the development of essential use cases, referred to simply as use cases.

Use Cases and CBD/e

CBD/e employs use cases for its analysis and requirements gathering work. Use cases describe the business process, which documents how the business works and what the business goals are of each interaction with the system. These use cases are then extended to show how the system will support the business goals.

The benefits of this style of use for use cases are twofold: the business processes are well documented, and the system requirements are described in terms of the processes they support. This makes for a close mapping between business process and requirements.

What are the objectives of use case analysis?

In general terms, the purpose of use case analysis is to document the business process that is to be supported by the system under development. However, to effectively develop a component-based application for that process, use case analysis must have a much more specific purpose. Use cases must document the business process to be supported in such a way as to facilitate the identification of *operations* that support the business process. Use cases must achieve the following goals in order to be effective for this stated purpose.

- Use cases must be an **Effective Communication Tool**
- Use cases must be scoped to a **Specific Business Goal**, which means they must identify **Business Decisions and Actions**.
- Use case steps must be identified as **Automated or Manual**

Effective Communication Tool

Use cases are a tool for customers to communicate the business requirements to software developers. For this to be an *effective* tool, the software analyst must be able to coax the customers to give them the right information. It is possible to perform business analysis by way of use cases and still not get the information necessary to build a good software solution; however, if the information does not communicate the requirements, or if the software developers find them impossible to use, then the use cases have been done improperly.

Use cases are a tool for software developers to communicate how the system meets the customer's business requirements. The use case documents the business process that the software solution is designed to support. Software developers must be able to map the specific features and functionality of the system to the use case. This mapping allows developers to relate requirements to system functions to prove that the system meets the requirements of the system. If the software solution cannot be effectively mapped to the use case, then the software solution does not meet the business requirements.

Use cases prove their worth when several things happen upon completion of the software solution. First, the customer, upon seeing the use cases again, agrees that the use cases properly describe the business process to be supported. Second, the software developers can show exactly how the system explicitly supports that business process. Third, the customer agrees that the system supports the business process as expected. Last of all, when a *really* good job was performed on the use cases, the customer will state, "I would like to take these use cases and use them as our procedure manual. We never have had this process documented so well and it would really help to train our staff."

Business Goals, Decisions, and Actions

The use case definition provided above mentions that a use case must accomplish a business goal. This concept is very important to use case development and is illustrated in the following quotes.

"An important issue I've come across with use cases is the difference between what I call user goals and system interactions." [Fowler97]

"Both styles of use cases have their applications. System interaction use cases are better for planning purposes; thinking about user goals is important so that you can consider alternative ways to satisfy the goals. If you rush too quickly toward system interaction, you will miss out on creative ways to satisfy user goals more effectively than you might by using the obvious first choice. In each use case it is a good idea to as yourself, "why did we do that?" That question usually leads to a better understanding of the user goal." [Fowler97]

In CBD/e, these two styles, user goals and system interactions, are combined. Each use case must document the steps to accomplish an identifiable, specific, measurable business goal. The system interactions are documented in terms of business decisions and business actions necessary to accomplish that goal. This approach gives each interaction a purpose and focus.

The identification of the business goal provides the analyst with the invaluable insight of knowing why each of the steps is to be performed. This leads to a system that better supports the business because the analyst can offer alternative solutions and, as a result, creates a system that adds more business value.

Once the goal of the use case has been defined, each of the steps, manual and automated, necessary to achieve that goal are documented. While a use case is supposed to describe the interactions between an actor (user or other system) and the system, it is too early in the process to distinguish between manual and automated steps. In addition, the documentation of manual steps forms a complete picture by which a user can understand exactly where the system supports the business process and where manual work is required. This documentation of manual steps makes the use case a more effective communication tool.

The best way to perform use case development for a business application is to focus on identifying business decisions and business actions in the use case steps. All steps are documented, but it is important to understand how each one supports a business decision or business action that in turn helps accomplish the use case goal. This approach will help to weed out unnecessary steps and it will cause each of the steps to have a clear purpose.

Identifying business goals, decisions, and actions is the second objective of use case development (the first is being an effective communication tool). Focusing on these items during use case development will greatly enhance the business value of the delivered system. Unfortunately, properly identifying the goals, decisions, and actions can present a challenge to analysts. Other articles in this series will discuss how to identify these items.

Automated vs. Manual Steps

Each use case step must be identified as automated or manual. The focus of each step is to make a business decision or execute a business action. Assigning responsibility for each business decision and business action to either the **system** (automated) or the **actor** (manual) directly impacts the system delivered to support the business process because the automated steps will result in system operations to make these decisions or execute these actions.

The system operations will be named according to the decisions or actions for which they are responsible. Naming operations this way will aid the ability to trace requirements to the delivered system because the operation name will reflect its business purpose, which should map to a business requirement.

Summary

Use cases are a very useful analysis tool. They can be used to define the business processes and the system requirements that are necessary to support that process, which leads to a natural mapping between the business processes and the requirements.

There are different types of use cases that can be used in different situations. Certain use cases describe business processes and the system response at a high level (essential use cases). These use cases can be refined to describe the interaction that takes place in a particular implementation of a system (real use cases).

The primary goal of use case analysis is to be an effective communication tool that describes the business processes and assigns responsibility to the steps of the process to either the system (an automated step) or to an actor that is outside of the system (a manual step).

[Editor's note] In the next installment of this series, we will discuss how and when to employ use cases. We will also look at the details of some of the common problems or issues that may arise when developing use cases, such as the scope of a use case, or the relationships that can exist between use cases.

References

- [Booch97] Booch, G., Jacobson, I., and Rumbaugh, J. 1997. UML Specification v1.3. Download from the Rational website: www.rational.com/uml/
- [Booch99] Booch, G., Jacobson, I., and Rumbaugh, J. 1999. *Unified Modeling Language – Users Guide*. Addison Wesley Longman, Inc. Reading, MA
- [Constantine97] Constantine, L. 1997. The case for Essential Use Cases. *Object Magazine* May 1997. SIGS Publications. NY, NY.
- [Fowler97] Fowler, M & Scott, K. *UML Distilled*. 1997. Addison Wesley Longman, Inc. Reading, MA
- [Jacobson92] Jacobson, I., et al. *Object-oriented Software Engineering: A Use Case Driven Approach*. Addison Wesley. Reading, MA.
- [Larman98] Larman, Craig 1998. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design*. Prentice Hall. Upper Saddle River, NJ.