CaseComplete 2010 Modeling Process

Contents

Preparation

Use Case Modeling Process

This document contains all of the modeling process topics that appear on CaseComplete's start page. It is provided for your convenience so you are able to read them all in one document or print them out. Note that the implied hyperlinks in this document will only work within CaseComplete.

Welcome to CaseComplete®, the better way to gather requirements. The items on this page describe a process for creating a use case model that CaseComplete is tailored to. Generally you should follow the steps from the top of the list to the bottom, but it is not a strict order. You may even choose to leave some steps out.

Move your mouse over the topics in the list to learn more about each step in the process. If the topic is a hyperlink, you can click on it to take you to the appropriate location in CaseComplete (even if you can't click on it, you can still move your mouse over the item for more information). To return to this page, select the "Start Page" tab along the top.

Many of the steps of this process are described in the following texts:

- *Advanced Use Case Modeling*, Frank Armour and Granville Miller, Addison-Wesley, 2001
- *Writing Effective Use Cases*, Alistair Cockburn, Addison-Wesley, 2001

Create a Vision Document

The **Vision document** provides an overview of the problem being solved and answers two key questions:

- Are we solving the right problem?
- Are we solving the same problem?

When beginning a new development effort, a key risk is misunderstanding the scope of the system under development. The vision document mitigates that risk by establishing a shared understanding of the business problem that the system should solve.

The vision document need not be long to be effective. You may choose to include the following items in your document:

- A terse problem statement
- The business opportunity
- Product positioning
- The stakeholders and users of the system
- High level goals or features

Click here or on the "create vision document" topic to create or view your project's vision document.

Identify Stakeholders

Stakeholders include anyone who has an interest in the use case project. It is important to identify and engage stakeholders as early as possible so their feedback can be incorporated into the use cases. The benefits of doing stakeholder analysis include:

- Prevents unwelcome surprises late in the development process
- More accurate requirements
- Promotes "buy-in" from all stakeholders including end users

Possible stakeholders include:

- End users
- Funding authority
- Project manager
- Developers
- Testers

Notice there are two kinds of stakeholders: non-users (such as the funding authority) and users. Each stakeholder will have a different set of goals for the project which is why it is important to include as many stakeholders as possible during the use case modeling process. Since each stakeholder has a different perspective, including all stakeholders *early* in the process helps identify issues that otherwise might not have been found until late in the development process.

A good place to keep track of stakeholders is in your vision document.

Introduction to Use Cases Presentation

Even though most stakeholders involved in your project will be not *writing* use cases, it's likely they will be involved in information gathering sessions, as well as reviewing use case documents, so it is important that everyone has at least a basic understanding of the use case modeling process. If some of your stakeholders are unfamiliar with use cases, you should set aside some time to teach them the basics of use case modeling.

Click here or on the "Teach stakeholders" topic to start a PowerPoint presentation that covers the benefits of use case modeling, some of the terminology, and the basic structure of a use case. (If you don't have PowerPoint, you can download a free PowerPoint viewer at microsoft.com).

The First Steps

Create a Context Diagram

A **context diagram** is an informal, short-lived artifact that is typically produced in a group setting. It identifies:

- The system boundary - identifies what is included in the system being built and what is external to the system.
- The external entities that interact with the system
- The nature of the interactions with each external entity

The context diagram is particularly useful for teams that are dealing with an unclear problem or a domain that is new to them. It is a good way to kick-start the analysis activity. For teams with sufficient domain knowledge, this activity can be safely skipped. Click here to see an example of a context diagram.

You can create context diagrams within CaseComplete using basic and line connector shapes. Click here or on the "Create Context Diagram" topic to import a generic context diagram that you can use as a starting point.

Define Actors

An **actor** has three essential traits:

- External to the system
- Interacts with the system
- Tries to achieve some goal

An actor is a role, not a specific person or thing, for example "Bank Customer", not "Jane Smith". There can be both human actors such as users or administrators, and non-human actors such as other software systems, and hardware devices.

Actors are usually defined before use cases because they provide the motivation for the use case. A use case is the direct result of some goal the actor has. Describing actors in a system puts a focus on *how* the system will be used, one of the benefits that use case modeling provides. Click here for tips on finding actors.

If the actor represents an end user, it is useful to include the background and skills of that actor in the actor description. This way, user interfaces can be designed to match the users' abilities.

Define Actor Goals

Actors have **goals** and the system under development helps satisfy those goals. The initial set of goals defined for your actors will give a good overview of what the system will look like and is the shortest summary of the system's function. To help find goals, ask the following questions for each actor:

- What measurable value is needed by the actor?
- What business event might the actor initiate?
- What service does the actor need from the system?
- Does the actor need to receive information from the system?

Once the set of goals has been defined, you can generate your initial use cases directly from those goals. Note however, there won't necessarily be a 1 to 1 correspondence between actor goals and use cases. For example, some use cases may fulfill several goals.

Ongoing Activities

Maintain a Dictionary

Early in the use case modeling process, you should start a **Dictionary**. Use the dictionary to identify key terms that have specific meaning in the domain that you're analyzing. Clearly defining key terms will help reduce miscommunication between stakeholders by giving clear definition to important terms.

As your analysis takes shape, use the dictionary to define Domain Objects. Identifying domain objects helps you move from use cases into the early stages of object-oriented design, and can capture important data requirements for your system (see the "Find Domain Objects" topic below).

Terms and elements to include in your dictionary:

- Tangible things
- Concepts
- Events
- Units
- Composite terms
- Acronyms
- Data objects

Document the purpose and source of the entry using the description field. If the entry is a domain object whose definition is still a bit sketchy, document as much as is known. Try to capture

- Description
- Format
- Possible values

Each dictionary entry may optionally specify one or more aliases (synonyms used to identify the same thing). When specifying more than one alias, separate each one with a comma.

As you learn more about the domain objects, add fields to the dictionary entry to capture the attributes of the domain object. Describe the type of information represented by the field and add constraints that limit the characteristics of the information represented by the fields.

Maintain Project Level Details

There are many details in a use case project that apply to the **project as a whole**, not just an individual use case. As you think of project level notes, requirements, or related external documents, add them to the top level package (packages are discussed below in "organize the project"). Click here or on the "add project level details" topic to view or add project level details now (or at any time you can double click on the top level package in the project browser on the right).

Organize Your Project Using Packages

When all of your use cases are presented as one large list, stakeholders have a difficult time understanding the "big picture". For this reason, it makes sense to **organize your use cases** into groups and subgroups, known as "packages". How you choose your packages is up to you and can vary between projects. One common way is to create packages based on the business functions of the use cases, for example, Shipping or Customer Management. Another way is to create them based on the primary actors, e.g. all use cases that have the Customer actor as their primary actor would get grouped into the same package.

To create a new package, right mouse click on an existing package in the project browser and select "Add Package", or click on the New Package icon in the main toolbar. You can then move existing use cases, requirements, and actors into the new package by dragging and dropping them in the project browser, or by right clicking on selected items in the main list and selecting the Move To menu time. You may even nest packages within a package.

Tip: if you are primarily working with use cases in a particular package, select that package in the toolbar drop-down list. This will cause only the use cases, requirements, or actors in that package to be shown, and in addition, any new items that you create will automatically get added to that package.

If more than one person needs to work on the same CaseComplete project, save each package as a separate file (via right mouse click on the package in the project browser). This way, each team member can be simultaneously working on the contents of a package without interfering with others. This will also allow check-outs and check-ins of a package if your project is shared. See Working in Teams with CaseComplete for more information about shared projects.

Write Use Cases

Develop Use Cases Iteratively

The process of writing use cases is **iterative**. It is not practical to write complete use case descriptions right from the start since writing use cases is a process of discovery. Writing a use case can be broken down into 3 phases: brief description, primary scenario, and fully dressed (see next topic).

Tip: CaseComplete provides two "views" of your use case and actor lists - details view and description view. Switch between the views by clicking on the appropriate toolbar button, or use

the View menu. The *description view* shows just the name and description and is especially useful when you want to quickly add new use cases or actors, e.g. during brainstorming. (To add a new item, just press enter while editing or click anywhere below the last entry in the list). The *details view* is useful when you need to sort by various criteria, prioritize, or see more fields for each use case. You may add and remove columns to the details view by right clicking anywhere in the list and choosing select columns, or use the View menu.

Make sure you manage your energy while writing your use cases. It is best to write them with the lowest level of precision that is necessary for your purposes. That level varies from organization to organization and from project to project. Trying to detail all possible variations and exceptions for a use case takes a great deal of effort and can actually decrease the level of comprehension of the reader if too much detail is added.

Define Use Cases

As mentioned in the previous topic, use cases are not written in one sitting, rather they go through various phases that gradually add more detail.

Brief Description: After identifying and naming use cases, add a brief description of the use case to ensure that its purpose isn't forgotten and to provide a quick, high level understanding of the system. The description is usually short and can include the following:

- A summary of the business goal motivating the use case
- The purpose of the use case
- A summary of the major behaviors that occur in the use case
- A final sentence that includes the result of the action

Primary Scenario: This phase adds the main success scenario to the use case, which describes the *normal* flow of events where everything goes as planned. You may also want to add preconditions and the success guarantee at this point, and you may start to reveal non-functional requirements (see "Add requirements" below). It's premature to focus on very low levels of detail during this stage - you are still in discovery mode.

Fully Dressed: The fully dressed description of a use case adds extensions or alternative flows to the main success scenario. During this phase, you make the use case sufficiently precise to drive other aspects of the project, such as estimation, scheduling, testing and development.

To include **extra details** in your use case as described by Alistair Cockburn and others (such as trigger and stakeholders) that are not part of CaseComplete's set of predefined fields, you can import the *AdditionalDetailsCustomFields* file from the custom fields dialog. The file is located in the *Templates* folder of the CaseComplete installation directory.

Set Use Case Priority

As you add use cases to your project, you should be considering their **priority** relative to each other. Use the priority to determine which use cases to add details to first (remember, writing use cases is iterative). It can also be used to specify development priority. In general, you should give higher priority to use cases that are architecturally significant or have higher levels of risk.

Priority is a numeric value, with a lower number designating higher priority. To easily change the priority, select one or more use cases and press the + or - keys, or the +/- buttons on the toolbar.

Make sure your priorities are well defined. Here is one example:

1. (High) - Important to describe and implement these use cases first. They may have impact on architecture and/or user interface. They may be high risk
2. (Medium) - Will be part of release, but can wait until later iterations
3. (Low) - Consider leaving out of release if time or resources are short
4. (Future) - Leave out of current release; consider for next release

Add Details

Reference Other Use Cases

You may notice that some of your use cases perform a common set of steps. For example, you may have several use cases in an online catalog system that need to verify a credit card. Instead of copying the same set of steps over and over, you may wish to break out those steps into a use case of their own, and then **create a reference** to that use case from the others.

To reference another use case, select "Insert Use Case Reference" from the right mouse menu (or type Ctrl+U). This will bring up a list of existing use cases from which to choose from (**tip**: find it quickly by typing the first few characters of its name). When you select the use case by double clicking or pressing enter, its name and ID will be inserted into the text you are editing. You are free to modify the name to make it fit grammatically and match the specific situation, for example you may want to change "verify credit card" to "Clerk verifies alternate credit card". The ID shown provides a link to the referenced use case: hover the mouse over it to see its brief description, and Ctrl+click on it to navigate to the use case.

References can be created in essentially any text field of any use case, actor, requirement or package. However references that are created in a use case's flow of events are special in that they represent **include relationships**. This is significant if you generate UML diagrams: arrows will be shown between use cases indicating that one use case includes another.

CaseComplete provides an easy way to **refactor** your steps into new use cases. If you have an existing set of steps that you would like to put in their own use case, select the steps, right click on one of them and select copy/move to new use case. If you choose to move the steps (as opposed to copying them), a reference to the new use case will automatically be added to your flow of events.

If you would like to reference a use case in **a different project**, enter the name of the file, followed by a question mark followed by the ID of the use case, all within square brackets. Example:

[C:\SomeFolder\MyProject.ucd?UC-10]

The text will appear as a hyperlink that you can Ctrl-click on to start a new instance of Case

Complete and bring up the specified use case. You may also do this for Actor and Requirement IDs.

In fact, you may do this in any application that supports hyperlinks by using the **custom Case Complete hyperlink**, cc://. Using the example from above, if you wanted to reference a use case from Excel, you would choose Insert/Hyperlink and type the following:

cc://C:\SomeFolder\MyProject.ucd?UC-10

Note that hyperlinks in Microsoft Office products don't support UNC syntax (e.g. \\SomeServer\...) so if your file is on a shared server, you must substitute "UNC" for the first backslash:

cc://UNC\SomeServer\SomePath\MyProject.ucd?UC-10

Add Requirements

Use cases capture *functional* requirements of your system, but systems have other important ***non-functional*** **requirements** that must also be captured (a.k.a. non-behavioral requirements). Consider the FURPS+ requirements model*. FURPS+ is a mnemonic for:

- Functional - features and capabilities
- Usability - human factors, interfaces, help, documentation
- Reliability - frequency of failure, recoverability
- Performance - response times, throughput, availability, resource usage
- Supportability - adaptability, maintainability, internationalization, configurability
- + "ancillary factors" - implementation, packaging, legal, etc.

You add new requirements the same way you create new actors and use cases, e.g. from the toolbar, in the requirements tab, or in the project browser. Just as for use cases and actors, you can add a description, notes, related documents and open issues to a requirement. In addition, requirements have various properties such as priority, type and status which you can assign using the properties window or by opening the requirement details page. You may also create your own custom fields for requirements. And since some organizations prefer to use a mix of use cases and requirements, CaseComplete also allows you to create other types of requirements besides non-functional, including functional and testing, as well as business rules. Specify the type of your requirement in the Type field of the requirement.

You can nest requirements in CaseComplete, e.g. when **child requirements** are variations of a generic parent. You can add a new child requirement by right mouse clicking on a requirement in the requirements tab or project browser and selecting "Add Child Requirement". Or, drag and drop existing requirements onto other requirements to "reparent" them, making them child requirements. You can also use the requirement form to change the Owner of an existing requirement to another requirement, making it a child requirement.

Tip: Use drag and drop or the Owner drop down list box to move child requirements out of their current parent requirement and into packages or other requirements, changing them into top level requirements or children of other requirements.

* Robert Grady, *Practical Metrics for Project Management and Process Improvement*, Prentice-Hall, Englewood Cliffs, NJ, 1992

Reference Requirements

In CaseComplete, requirements are referenced by use cases or requirements. More than one use case may refer to the same requirement, for example if two use cases have the same performance requirements. You may want to create packages that only hold requirements to keep them separate from use cases, perhaps a package that holds only performance requirements, or one that holds business rules.

You can indicate which requirements are applicable for a use case by selecting them on the details tab of the use case form, or by drag and dropping requirements onto a use case (or vice-versa). Although this is not required, it can be helpful in understanding what use cases will be affected when a requirement is changed. You can see the list of affected use cases for each requirement in the Requirements Cross Reference Word report. You may also find the Matrix - Use Cases and Referenced Requirements Excel report useful.

When you want to show any sort of dependency between requirements, CaseComplete allows you to create explicit references between requirements. For example, when one requirement can't be fulfilled unless another requirement is implemented. Requirement references can be created and removed using the details tab of the requirements form.

To reference a requirement within the description or any other text field of a use case, actor, requirement, or package, you can create an id link reference. Select "Insert Requirement ID Link" from the right mouse menu (or type Ctrl+R). This will bring up a list of existing requirements to choose from. When you select the requirement, its ID will be inserted into the text you are editing.

Add Related Documents

You may choose to describe **supplemental information** for your system in separate documents. You can attach these documents to any actor, use case, requirement, or package by selecting the appropriate item in the main list or project browser, selecting the "related documents" tab along the bottom, and dragging the file from Windows explorer into the list. Or you can add URL's and files manually via the right mouse button. Examples of supplemental information include:

- User Interface specification - since you want versatility in the design of the user interface, you should try not to assume any particular UI when documenting a use case. The creation of the use case generally precedes user interface specification.
- Screen mockups - note that standard image files can automatically be inserted as pictures into Word reports.
- Machine Interface specification - could include legacy interfaces and new interfaces
- Requirements - if your requirements are complex, you may choose to document them separately instead of in packages as described above.

Communicate

Generate Reports

When you are ready to produce output of your project, you have 3 options: generate a Microsoft Word report, a Microsoft Excel report, or an HTML report. You may find any of these report styles are useful, depending on the particular situation.

Word and Excel reports use an easily customizable template document to produce output that can be tailored to your organization. Click to view the Custom Reports User's Guide to see how to customize and create your own templates.

HTML reports are viewed using your web browser, which CaseComplete will bring up automatically. From there, you can save the report as an html file, copy/paste it to another document or an email, or print it. HTML reports are produced using XSLT scripts, so are also customizable.

If you create new template documents or XSLT scripts and place them in the Reports folder of your installation directory, they will automatically show in the Generate Reports list.

Incorporate Diagrams into your Project

You may find it useful to include diagrams to aid in the reader's comprehension of the material. Every Use Case, Actor, Requirement and Package can own one or more diagrams. CaseComplete supports several types of diagrams:

- **UML Use Case diagrams**: These diagrams show use cases (ovals), actors (stickmen) and the relationships between them (directed arrows). You can learn more about UML at www.uml.org.
- **Generic free-form diagrams**: CaseComplete provides many different shapes and arrow styles allowing you to create diagrams such as flow-charts and activity diagrams.
- **User Interface mockups, or wireframes**: CaseComplete provides many common UI controls such as buttons and checkboxes allowing you to mock up screen layouts for your web or desktop applications.

To create UML Use Case diagrams:

- Drag an item from the project browser into a diagram to create a shape representing an *existing* use case, actor or package.
- Or, drag shapes from the shapes library to create *new* use cases, actors or packages.
- CaseComplete does not distinguish between the diagram types described above, so you can add use cases and actors to any diagram, and vice-versa, you can add any free-form or UI shape to a use case diagram.

Relationships between Actors and Use Cases:

- Primary actors for a use case are represented as an arrow from the actor to the use case. Arrows for supporting actors point the other direction, from the use case to the actor.

- Include relationships are represented as a dashed arrow from the including use case to the included use case. See the *Reference Use Cases* topic above for more information about how to create include relationships in your flow of events.
- When relationships exists between an item dragged in from the project browser and use cases and actors already in the diagram, relationship arrows are automatically added to and from the existing shapes.
- Right click on any use case or actor shape and select "Add Related Shapes/Connectors" to add all actors and use cases that are related to that item, as well as the arrows between the shapes added, and between existing shapes in the diagram not already connected.
- You can specify new primary and supporting actors directly in the diagram by using the arrow tool from the toolbox above the diagram.
- Likewise, you can add include relationship arrows between use cases. However, unlike primary and supporting actor arrows, adding or deleting an include arrow has no effect on the underlying use case. For example, if you add an include arrow, CaseComplete will not add a step to your flow of events that contains a reference to the included use case.

For detailed information about diagramming, take some time to run through the diagramming tutorial (close your current project and then open the tutorial project from the Getting Started box on the Start Page).

Generate UML Diagrams

Once you have use cases and actors defined, you may find it useful to visualize your use case project using **UML diagrams**. In addition to using built-in diagramming as described above, you can also do this by exporting your use case project to an XMI file, which can then be imported by most UML drawing tools. Case Complete can export both UML 1.3 and 1.4 formats. Consult documentation for your UML tool to determine which version is appropriate, or simply try them both to see which yields the best results.

CaseComplete will export the following information:

- Use Cases, Actors, Packages, and their descriptions
- Uni-directional associations between use cases and their primary and supporting actors
- Include relationships between use cases (determined by use case references found in the flow of events, see the *Reference Use Cases* topic above for more information).
- Use Case details (e.g. priority and status) (exported as tagged values)
- Dictionary entries (exported as domain objects - see next subject)

Find Domain Objects

Domain objects are the real-world entities that your system is responsible for, such as Orders and Loan Applications. By finding and defining your system's domain objects you will gain an even greater understanding of your system and set the foundation for an object-oriented design. Data modelers can also use the information you capture for domain objects as the foundation of a database model.

CaseComplete tracks domain objects in the dictionary. As you identify a domain object during the course of writing your use cases, right click on the text to add it to the dictionary.

CaseComplete helps with domain object modeling in several ways. First, when you export to UML as described in the previous subject, CaseComplete will generate a class for each entry in the dictionary. You can use these classes in a UML modeling tool as a starting point to add design details such as the object's attributes and its relationships to other domain objects.

Second, CaseComplete can generate a domain analysis report. CaseComplete searches for domain objects (i.e. dictionary terms) in the description and flow of events of each use case and creates two cross reference tables: one showing domain objects for each use case and the other showing use cases for each domain object. This gives you a quick overview of the domain objects that each use case touches and a means of tracing which use cases are affected by changes to a particular domain object. You can also add to each table how the use cases interact with the objects, such as whether they Create, Read, Update or Destroy the object. Often referred to as a "CRUD" matrix, this helps document each object's life cycle.

Finally, CaseComplete can generate a Dictionary report containing each domain object, the specific fields and constraints defined for each domain objects and any special constraints that apply. This can be used by data modelers and information specialists as the basis of an information model.

Generate Project Plan

You may find it useful to export your use cases and requirements to **Microsoft Project** in order to plan and track their development. Note however that use cases don't always map directly to development tasks, for example one use case may necessitate multiple tasks that span several subsystems and are assigned to multiple individuals. Despite this, exporting your use cases can serve as a good starting point for your project plan.

CaseComplete exports each use case and requirement as a task and each package as a summary task. Since Microsoft Project and CaseComplete use different priority schemes, you can optionally map CaseComplete priority to an equivalent Project priority (by default, priority 1 maps to Project priority 1000, priority 2 maps to 900, etc.)

You can also optionally assign a rough estimate of duration based on use case complexity. You are free to set your own duration estimates for each level of complexity by selecting a complexity value and assigning a duration in Project format, e.g. 5d, 5 days, 1w, etc. Note that CaseComplete will not attempt to link tasks, i.e. they are exported as concurrent.

CaseComplete exports project data as a tab delimited text file. To import the file into Project, select File/Open and select the "text, tab delimited" file type. This will start the import wizard, which you can quickly go through by accepting all defaults.

Several CaseComplete fields don't map directly to Microsoft Project fields. CaseComplete exports these fields as generic Project text fields. If you wish, you can add them as columns in Project and give them meaningful column names at that time. These fields are mapped as follows:

Text1 = ID
Text2 = Complexity
Text3 = Use Case Status
Text4 = Implementation Status
Text5 = Release