



www.primaryview.org

from the notebooks of John Artim

Use Cases and Software Requirements Specification

Pragmatics of Use Case Authoring

John M. Artim, ©2003, 2004. Permission to copy, modify, or distribute is granted.

Acknowledgements

- ✓ This notebook was developed in collaboration with Chuck Walrad of Davenport Consulting, Inc (www.daven.com)
Any errors are solely the responsibility of John Artim!
- Feel free to use this material, in whole or part, for your own training needs but please indicate that you based your work, with permission, on a copyrighted work by John M. Artim that is available from www.primaryview.org.
- A Microsoft® PowerPoint file of this notebook is available by request. Send an email to John Artim (jartim@uistyle.com) requesting the Use Case Basics Notebook PowerPoint file in the Use Cases and Software Requirements Specification series. Please include your name, affiliation, and your intended use of the material.

Goals of this Notebook

- This notebook discusses use case authors' common questions and problems
- One or more rules-of-thumb are presented addressing each of these questions or problems

Browsing time for this notebook is about three hours depending on the reader's experience level

Use Case Pragmatics

www.primaryview.org

from the notebooks of John Artim

What is a Rule-of-Thumb?

- A *Rule-of-Thumb* provides an approach or set of guidelines for solving a problem
- Also known as a *Heuristic* in some circles

Pragmatics of Use Case Authoring

- The pragmatics of use case authoring are presented in this notebook as a set of rules-of-thumb to guide authoring, extending, reviewing, and managing sets of use cases



Background for this Notebook

- You should know what a use case is, what makes up a use case, and have a basic understanding of how individual use cases work together to form a model as outlined in the *Use Case Basics* notebook



Identify Your Primary Purpose

- Before you begin any use case work, you must identify your primary purpose:
 - Describing business processes?
 - Exploring and documenting a potential product or system?
 - Specifying a product or system?
- Each of these tasks focuses on a different level and scope of use case affecting many of the rules-of-thumb in this notebook

Use Case Problem Areas Addressed

- Authoring Use Cases
- Extending Use Cases
- Reviewing Use Cases
- Managing Use Cases

Problems Authoring Use Cases

- How do I get started?
- When do I re-factor?
- How do I know when I'm done?
- *Analysis Paralysis* has struck! How do I get “unstuck”?
- How do I prepare my use cases prior to review or publication?
- How do I deal with domain terminology?

How Do I Get Started?

- Pick a simple story to tell. Then tell it in a Use Case!
 - Use no more than a dozen or so steps; close to seven is about right
 - Most steps in this summary use case will become *included* use cases
 - Where possible, get down to a task use case as soon as you can
 - The steps in your summary use case say something like, “One actor does this. Another actor does something else.” And so on

How Do I Get Started?

(continued)

- Add more stories in the same way...
- But make sure the stories hang together
 - After adding each story, check its fits with prior stories
- Don't go crazy!
 - You are looking for one or a few key stories
 - Use these stories to illustrate where you are going with your set of use cases

How Do I Get Started?

(continued)

- Find reference sources that describe the domain
 - Encyclopedia Britannica and other general references provide excellent high-level introductions to many domain areas
 - Ask SMEs for white papers and reference models
- Sketch first, fill out the picture later!
 - Create a straw-person to test out this sketch
 - This is critical!
 - Find your misunderstandings of basic domain concepts early by discussing the sketch with a SME

Re-factor Defined

- Re-factoring is the process of restructuring something so that it better serves its purpose
- When you re-factor something, you make it easier to understand and to work with

Re-Factoring in Use Cases

- In use case modeling this means breaking up the tasks making up a set of use cases and reforming them in a number of ways:
 - Changing which steps go into which use cases sometimes even shifting the names and goals of use cases in the set
 - Adding or deleting either steps or use cases as needed
 - Re-factoring can also change pre-conditions, success end conditions, triggers, and everything else in the case

Re-Factoring in Use Cases

(continued)

- Re-factoring has tremendous power to improve a model but adds the cost of re-checking everything you change and everything those changes affect!
 - **Never underestimate the difficulty and sheer cost of rechecking a set of use cases for consistency and accuracy!**

When Do I Re-Factor?

- When you think you need a new use case to add to your set of cases, ask yourself:
 - Is the goal of this new use case the same as or very similar to any other existing use case ?

or

 - Are the pre-conditions and success conditions for the new use case the same or very similar to any other existing use case ?
- If the answer is, *yes*, then try to re-write the existing use case as needed to match to meet the new needs as well as the original need

When Do I Re-Factor?

- When you start to add a new use case that seems like another one you already have, ask yourself:
 - Is the Goal of this new use case the same as the other existing use case ?
 - or*
 - Are the pre-conditions and success conditions for the new use case the same as in the other use case ?

If the answer to either question is *yes*, try to re-write the existing use case as needed to match the new needs as well as the original need

Beware though, you won't find this level of sameness often!

When Do I Re-Factor?

(continued)

- If a use case is triggered by two or more other use cases:
 - Is the goal of the triggered use case trying to serve two or more needs but not really filling *all* of these needs well?
 - or*
 - Are some of the pre-conditions or success conditions related to one of the triggering use cases and some to another?

That is, do you only use part of the triggered use case depending on who is triggering it?

If the answer to either is, *yes*, you may need to split the one use case into two or more use cases

When Do I Re-Factor?

(continued)

- Are all the steps in each use case parallel to each other? Reword steps as needed to place all steps at about the same level of detail
- Are all use cases included *within* a use case at roughly the same level of detail. If not, take a close look at each use case that is not at the same level
 - Is it the right use case for the situation?
 - Is it written at the correct level? Why was it written at that level?
- Parallelism is a common trigger for re-factoring and rewards a small to moderate investment in time with a *much* more readable use case model

How Do I Know When I'm Done?

- This question is really a series of four questions:
 - Have I Gone High Enough?
 - Have I Gone Low Enough?
 - Have I Gone Wide Enough?
 - Are there any Gaps?

When Am I Done—Have I Gone High Enough?

- Do the Goals of the highest use cases explain why all the lower cases are there?
 - If you are doing a proof-of-concept or business process analysis, are the highest cases summary use cases that explain business goals in the domain?
 - If you are doing a product specification, are the highest cases summary use cases that explain how your product fits in to the client business?

When Am I Done—Have I Gone Low Enough?

- Do the lowest use cases document all in-scope user tasks?
 - If you are doing a proof-of-concept or business process analysis, are the lowest cases task use cases that explain the user goals that must be satisfied to complete the business processes?
 - This is the minimum level you need to get to.
 - Hint: You are describing how individual user goals (or a sub-organization's goals) contribute to achieving an organization's goals. For example:
 - How do staff members' tasks achieve the department goals?
 - Or, how do the subsidiary companies' processes achieve the parent company's goals?

Have I Gone Low Enough?

(continued)

- Do the lowest use cases document all in-scope user tasks?
 - If you are doing a product specification, are the lowest cases task use cases that document where end users will use your product to complete their tasks and satisfy their goals?
 - This is the minimum level you need to get to
 - Hint: Task use cases that involve the SUD should feature only the SUD as a supporting agent to the primary actor so that you can clearly spell out the SUD's responsibilities

When Am I Done—Have I Gone Wide Enough?

- Do the use cases document all in-scope user tasks?
 - If you are doing a proof-of-concept or business process analysis, have all business processes been captured as a summary use case? And for these summary use cases, are the task use cases included within them necessary and sufficient?
 - If you are doing a product specification, are all of the (areas of) task use cases that are to be supported by your product included?

When Am I Done—Are there any Gaps?

- Read your stories through the Use Cases

For a given level of detail, can you flow **from one use case to the next** without a significant gap in:

- Goals from one case to the next
 - Success conditions to pre-conditions
 - Triggers whose origin isn't explained by prior use cases or obvious behavior of actors
- Fill any gaps as needed!



Analysis Paralysis!

How Do I Get “Unstuck”?

- Tell a story!
- Understand the motivation of your characters!
- Sketch it, don't try to paint the Mona Lisa from the very first!
- Always do breadth before depth—never go too deep too fast!
- When all else fails, try switching to another part of the use case set and come back later
 - Something may be missing in your understanding of one area that may become apparent as you work with adjacent use cases
- Think of this as the analyst's version of writer's block

How Do I Prepare for Review or Publication?

- Do the use case name, goal-in-context, and success end conditions consistently describe the goal?
- Do the use case name, goal-in-context, and scenario steps consistently describe the same action?
- Have all embedded to-do notes to and among the use case authors been resolved and removed?
- Is the status of all open issues correct and up-to-date?
Are all issues listed?

How Do I Prepare for Review or Publication? (continued)

- Is everything Spell Checked?
- Are all domain-specific terms in a glossary?
 - Where appropriate, are they in the glossary as the preferred term?
- Are all super use cases listed? Are the names correct?
- Are all sub use cases listed? Are the names correct?
- Have you double-checked all cross references between use cases?

How Do I Prepare for Review or Publication? (continued)

- If this is a major revision, and if the readership is consistent with the last revision, make sure you have change bars if they are appropriate for the audience
- Consider adding a major section titled, *Revision History*, that lists each release version, release date, and release change content. This can make it much easier for your reviewers and easier for you to manage!

Domain Terminology Issues

- Ambiguity
 - Domain terms, having evolved organically, often use *ambiguity* to cover over inconsistent use of terminology
 - When it proves difficult to get a good glossary definition of a term try focusing on the Steps in the use cases that use that term. Force the language to be as unambiguous as possible and try to tease out the nuances in the term's usage

Domain Terminology Issues

(continued)

- Consistency
 - Between Use Case Authors
 - When you use a Glossary, how can this be a problem? ☺
 - Between SMEs or between domain contexts
 - This is harder to control
 - Use cases can help identify where these discrepancies occur
 - By localizing them to individual use cases
 - Adjust language use to reduce or remove the inconsistency
 - Satisfy the SME with language that seems natural
 - But don't paper over the problem
 - If all else fails, emphasize the inconsistency and press for a resolution!

Domain Terminology Issues

(continued)

- Sometimes domain terminology discrepancies across various aspects of the domain can be difficult to get rid of
 - Try to investigate how plain English solves the problem
 - Use a good, unabridged dictionary (or even two—second opinions are helpful)
 - Non-specialist language has evolved for a longer period of time (mostly!) and so tends to be less brittle, more extensible
 - Non-specialist language may give you a clue as to where the problem lies if not how to fix the problem
 - Or the problem may be in the structure of your use case set
 - If this is the case, you may need to re-factor your use cases

Brittle Defined

- A brittle model or terminology is one that, when extended, fails ungracefully
- A set of use cases is brittle if the structure of the use case set does not allow the set to be extended to cover the required scope

Extending Use Cases

- Going from Proof-of-Concept to Product Specification
- Adding New Features

Proof-of-Concept to Product Specification

- POC concentrates on demonstrating business value of automation
 - The lowest-level use cases are high-level task use cases
- When going from POC to product specification:
 - Widen high-level task use cases, as needed, to verify accuracy of the POC use cases
 - Deepen the set of use cases so that all task use cases involving the SUD are highlighting the role of automation in aiding the user in satisfying their goal
 - See slide 21

Adding New Features

- There should be an existing use case model covering the product into which new features are being added
- Check that this use case model is wide enough at a summary use case level, then at a task use case level
 - Apply *Have I Gone Wide Enough* Rules-of-Thumb
- Check the use case model for gaps
 - Apply *Are there any Gaps* Rules-of-Thumb

Reviewing Use Cases

- Where do I start?
- How do I move through the set of use cases?
- Who reviews what? When do they review it?

Review—Who Does What?

Review Type	Role	Responsibilities
Informal Walkthrough	Author	<ul style="list-style-type: none">•Write Cases•Create UC Diagram•Lead SME through a Story
	SME or Informant	<ul style="list-style-type: none">•Respond to Story
Formal Review	Author	<ul style="list-style-type: none">•Write Complete Cases•Create UC Diagrams•Outline Review Strategy
	Reviewers	<ul style="list-style-type: none">•Use Strategy for First Read•Then Dig Deep, as Needed

Review—Where Do I Start?

- Is there an overall use case diagram?
 - If so, check the use cases in the upper-left corner of the diagram—these should be most important
 - And look at the actors
 - For each role, what use cases does that actor perform?
 - For each concrete actor, what roles does that actor play?

How Do I Move Through the Set of Use Cases?

- Look for a simple story. Do any of the highest-level summary use cases seem like a story?
 - If so, read through all its included use cases as though it were a story
 - Then go back and evaluate all of these use cases

Who Reviews What? When?

- Early On
 - Give an early draft of what you believe is a typical low-level summary or high-level task use case (or maybe two use cases) to a SME and get some feedback
- When You Have Significant Domain Questions
 - Take a draft of the affected use case or cases to the SME and walk through the case with them
 - Check other use cases employing similar domain concepts

Stakeholders Defined

- The project *Stakeholders* are the individuals who have a substantial vested interest in a project
 - This can include the project sponsor, project manager, lead analyst, development manager, development team lead, chief architect, SME, business analyst, end-user, and customer representative
 - Sometimes a stakeholder represents an entire population with a vested interest as when a team lead represents their team members or one end-user represents all end-users
 - Stakeholders are not necessarily boosters of a project
 - But as a general rule, all stakeholders must be satisfied for a project to continue and succeed

Who Reviews What? When?

(continued)

- When you have a breadth pass through the use cases:
 - Compile a set of drafts for these use cases and walk through them with a SME. If SMEs with significantly different expertise or type of experience are involved, review with each separately
- When you complete a full pass through the use case set:
 - Compile a set of drafts for these use cases and conduct a walk-through with a cross-section of stakeholders

Who Reviews What? When?

(continued)

- When you think the use cases are ready, do a full-blown stakeholder review:
 - Divide the use cases into logically connected sets (packages)
 - This is usually all use cases needed to support a high-level summary use case as well as the summary case itself
 - Make sure each package has its own use case diagram
- Note:** consider packaging use cases as you author them to save time and effort at the end of review cycles

Who Reviews What? When?

(continued)

- To conduct a full-blown stakeholder review:
 - Do one package per session if time permits
 - Spread the sessions out so that no more than one session is held per day, if possible
 - When stakeholders are brought to one location from across the globe you may have to do a single, two-to-three day review with sessions held back-to-back
 - Budget more time for the first session
 - Even if everything is right, the first review session, that is, review of the first package of use cases, will take longer

Managing Use Cases

- Coordinating Use Case Change
- Controlling Changes to Existing Use Cases
- Managing Large Use Case Sets

Coordinating Use Case Change

- If more than two people are authoring use cases nominate one use case czar
 - The use case czar mediates changes that affect other use cases, primarily:
 - Use case name and goal-in-context
 - Success end conditions

Controlling Changes to Existing Use Cases

- Use a change request process to manage changes when:
 - Changing an existing the shared aspects of a use case—most especially the name, goal-in-context, success end conditions, and preconditions
 - Deleting a use case
 - Adding a summary use case
- Before formal review is complete these changes must be cleared with the use case czar
- After formal review and approval use normal change control procedures



Managing Large Use Case Sets—What to Keep Track Of

- What is Done within Each UC?
- Who has Requested Changes to Each UC?
- What are the Open Issues for Each UC?
- Are all References to Each UC Correct?
- Is it named correctly everywhere?
- Is it correctly named as a Super- or Sub-Case when it should be?
- Do the diagrams agree with the text?

Managing Large Use Case Sets—How to Do This

- Using Word and Visio
 - Try keeping a summary table of use case status:
 - Use case name, ID, goal-in-context, status (by section), change requests pending, and open issues
 - If you save files as Rich Text (RTF) you can use text search tools such as GREP or Perl or a good plain-text editor to check for consistency, old names, and so on.
 - This is *very* labor intensive!

Managing Large Use Case Sets—How to Do This (continued)

- Using a CASE tool
 - You should be able to produce reports to aid in finding incomplete sections, outstanding change requests, inconsistent naming, and what not
 - Doing this sort of consistency checking and coordination with a CASE tool saves enough in labor costs that it makes the case for buying the tool in the first place
 - If your CASE tool does not support this kind of reporting it isn't a very good tool for managing use case models and, depending on total cost of ownership and use, probably isn't worth using for such

How Does An Analyst Get Started, Revisited

- How do I get the SME or user to give me the entire process?
 - Focus on telling a story
 - Make it the SME's story; get them involved!
 - Write a high-level draft of the story and review it with the SME
 - Revise and iterate
 - Use your rules-of-thumb for completeness to look for holes in the story!

How Does An Analyst Get Started, Revisited (continued)

- A user-centered focus ties it together
 - From business goals to end-user tasks a user-centered focus makes it a compelling story
 - The story is your users' story
 - The user, the analyst, the developer, and any other stakeholders can all see why everything is being done and why it should be done
- Use cases sell themselves, then sell the project.
If they don't, then rethink the project!

Wrap-Up

- The tips and rules-of-thumb described in this notebook will only really make sense in the context of your daily work
 - See each new project you work on as an opportunity to put these rules-of-thumb into practice—that’s the only way to really work out what them mean!
 - Develop a peer network of fellow practitioners and help each other!
 - Share what you learn with others. You’ll learn it more thoroughly and you’ll contribute your own insights to the way we practice use cases!