Agile Modeling

Usage Scenarios: An Agile Introduction

A usage scenario, or scenario for short, describes a real-world example of how one or more people or organizations interact with a system. They describe the steps, events, and/or actions which occur during the interaction. Usage scenarios can be very detailed, indicating exactly how someone works with the user interface, or reasonably high-level describing the critical business actions but not the indicating how they're performed.

Usage scenarios are applied in several development processes, often in different ways. In derivatives of the Unified Process (UP) they are used the help move from use cases to sequence diagrams. The basic strategy is to identify a path though a use case, or through a portion of a use case, and then write the scenario as an instance of that path. For example, the text of the "Withdraw Funds" use case would indicate what should happens when everything goes right, in this case the funds exist in the account and the ATM has the funds. This would be referred to as the "happy path" or basic course of action. The use case would include alternate paths describing what happens when mistakes occur, such as there being insufficient funds in the account or the ATM being short of cash to disburse to customers. You would write usage scenarios that would explore the happy path, such as the first scenario above, as well as each of the alternate courses. You would then develop a sequence diagram exploring the implementation logic for each scenario.

High-Level Example

Scenario: ATM banking for the week.

1. Sally Jones places her bank card into the ATM.
2. Sally successfully logs into the ATM using her personal identification number.
3. Sally deposits her weekly paycheck of $350 into her savings account.
4. Sally pays her phone bill of $75, her electric bill of $145, her cable bill of $55, and her water bill of $85 from her savings account
5. Sally attempts to withdraw $100 from her savings account for the weekend but discovers that she has insufficient funds
6. Sally withdraws $40 and gets her card back

Detailed Example

Scenario: A successful withdrawal attempt at an automated teller machine (ATM).

1. John Smith presses the "Withdraw Funds" button
2. The ATM displays the preset withdrawal amounts ($20, $40, and so on)
3. John chooses the option to specify the amount of the withdrawal
4. The ATM displays an input field for the withdrawal amount
5. John indicates that he wishes to withdraw $50 dollars
6. The ATM displays a list of John's accounts, a checking and two savings accounts
7. John chooses his checking account

8. The ATM verifies that the amount may be withdrawn from his account
9. The ATM verifies that there is at least $50 available to be disbursed from the machine
10. The ATM debits John's account by $50
11. The ATM disburses $50 in cash
12. The ATM displays the "Do you wish to print a receipt" options
13. John indicates "Yes"
14. The ATM prints the receipt

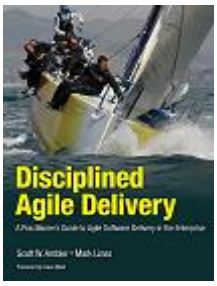
As you can imagine, there are several differences between use cases and scenarios. First, a use case typically refers to generic actors, such as Customer, whereas scenarios typically refer to examples of the actors such as John Smith and Sally Jones. There's nothing stopping you from writing a generic scenario, but it's usually better to personalize the scenarios to increase their understandability. Second, usage scenarios describe a single path of logic whereas use cases typically describe several paths (the basic course plus any appropriate alternate paths). Third, in UP-based processes use cases are often retained as official documentation whereas scenarios are often discarded after they're no longer needed.

Usage scenarios are a major artifact in the Agile Microsoft Solutions Framework (MSF). They are used in combination with personas, descriptions of archetypical users such as John Smith or Sally Jones to explore the requirements for your system. With the Agile MSF you would write either style of usage scenario as appropriate, and you would keep the usage scenario only if your stakeholders are willing to make that investment in the documentation. The Agile MSF has been influenced by Agile Modeling, including the practice Discard Temporary Models.

Let Us Help

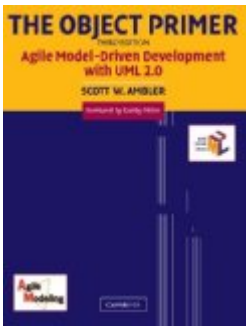
We actively work with clients around the world to improve their information technology (IT) practices, typically in the role of mentor/coach, team lead, or trainer. A full description of what we do, and how to contact us, can be found at Scott Ambler + Associates.

Recommended Reading



This book, Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise describes the Disciplined Agile Delivery (DAD) process decision framework. The DAD framework is a people-first, learning-oriented hybrid agile approach to IT solution delivery. It has a risk-value delivery lifecycle, is goal-driven, is enterprise aware, and provides the foundation for scaling agile. This book is particularly important for anyone who wants to understand how agile works from end-to-end within an enterprise setting. Data professionals will find it interesting because it shows how agile modeling and agile database techniques fit into the overall solution delivery process. Enterprise professionals will find it interesting beause it explicitly promotes the idea that disciplined agile teams should be enterprise aware and therefore work closely with enterprise teams. Existing agile developers will find it interesting because it shows how to extend Scrum-based and Kanban-based strategies to provide a coherent, end-to-end streamlined delivery process.

The Object Primer 3rd Edition: Agile Model Driven Development with UML 2 is an important reference book for agile modelers, describing how to develop 35 types of agile models including all 13 UML 2 diagrams. Furthermore, this book describes the fundamental programming and testing techniques for successful agile solution delivery. The book also shows how to move from your agile models to source code, how to succeed at implementation techniques such as refactoring and test-driven development(TDD). The Object Primer also includes a chapter overviewing the critical database



development techniques (database refactoring, object/relational mapping, legacy analysis, and database access coding) from my award-winning Agile Database Techniquesbook.

DISCIPLINED AGILE | Agile Modeling | Agile Data | Enterprise Unified Process | Agile Unified Process

Ambysoft | The Enterprise Transformation Advisor™ | SCOTT AMBLER + Associates | @scottwambler

---