Acceptance/Customer Tests as Requirements Artifacts: An Agile Introduction

Acceptance tests (also called Customer tests or Customer Acceptance Tests) describe black-box requirements, identified by your project stakeholders, which your system must conform to. In traditional software development acceptance tests are typically thought of as testing artifacts, which they are, but when you step back and think about it acceptance tests really are first class requirements artifacts because they describe the criteria by which stakeholders will determine whether the system meets their needs. In short, they're executable specifications. Business rules, features, technical (non-functional) requirements, and even detailed usage requirements can easily be captured as acceptance tests. Mike Cohn likes to call acceptance tests "conditions of satisfaction". Ron Jeffries measures project progress via the Tested Feature Metric, and the only way you can capture this metric is by having an acceptance regression test suite.

Figure 1 and Figure 2 each depict acceptance test descriptions (both test descriptions are shortened for the sake of brevity, both would need to be expanded with more steps, and/or support by other test cases, to truly validate the functionality described). As you'd expect, each test has instructions for setting up and then running it. Additionally, a description, test ID (optional), and expected results are also indicated. In the case of Figure 2, the definition of a "standard wildcard search" would appear in your organizations user interface guidelines (remember, follow the practice Apply Modeling Standards).

Figure 1. Acceptance test for validating a business rule.

ID	T0014
Description	Checking accounts have an overdraft limit of $500. As long as there are sufficient funds (e.g. -$500 or greater) within a checking account after a withdrawal has been made the withdrawal will be allowed.
Setup	1. Create account 12345 with an initial balance of $50 2. Create account 67890 with an initial balance of $0
Instructions	1. Withdraw $200 from account #12345 2. Withdraw $350 from account #67890 3. Deposit $100 into account #12345 4. Withdraw $200 from account #67890 5. Withdraw $150 from account #67890 6. Withdraw $200 from account #12345 7. Deposit $50 into account #67890 8. Withdraw $100 from account #67890
Expected Results	Account #12345: • Ending balance = -$250 • $200 Withdrawal transaction posted against it • $100 Deposit transaction posted against it • $200 Withdrawal transaction posted against it Account #67890: Ending balance = -$500 • $350 Withdrawal transaction posted against it

	• $150 Withdrawal transaction posted against it • $50 Deposit transaction posted against it Errors logged: • Insufficient funds in Account #67890 (balance -$350) for $200 Withdrawal • Insufficient funds in Account #67890 (balance -$450) for $100 Withdrawal

Figure 2. Acceptance test for validating part of a user interface.

ID	T0015
Description	The customer search screen allows user to perform standard wildcard searches on first and last name.
Set Up	1. Remove all customer records from the database. 2. Add the following customers: • John Smith • James Doe • Robin Saunders • Jim Saunders • Sally Smith • Scott Davidson • Beverley Williams • Bob Roberts • Rob Williams • Robert Smithers • Bobby Snookerby • Sandy Davington • Janice Sinters
Instructions	1. Display the Customer search screen. 2. In the First Name entry field, enter "%ob*" 3. In the Last Name entry field, enter "S*" 4. Press the search button.
Expected Results	The following names should be displayed in the search result box: • Robin Saunders • Robert Smithers • Bobby Snookerby

Acceptance tests should be fully automated so that you can run them as part of your application's regression test suite. The FITNesse testing framework is a popular choice for doing so.

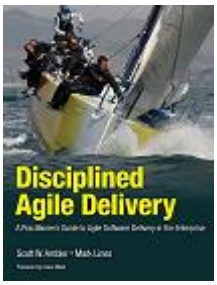
Why is it important to recognize that acceptance tests are first-class requirements artifacts? Because if you follow the single source information practice and capture requirements once, as acceptance test, instead of twice (as some other requirements artifact and as acceptance tests) you can dramatically reduce your traceability needs on your project. Furthermore, your acceptance/customer tests are effectively executable specifications, providing value both in the form of detailed specifications and as tests which validate whether your software conforms to those specifications.

Share with friends: Tweet LinkedIn Facebook StumbleUpon Digg Baidu Google +

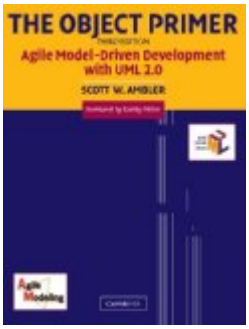
Let Us Help

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

---

Copyright 2003-2014 Scott W. Ambler

This site owned by Ambysoft Inc.