Maximizing Test Case Efficiency

August 2007 - Pragmatic Software Newsletters

It seems we are busier today than ever. With increasing workloads, it is important to maximize the time spent creating test cases. A well thought out testing approach will pay dividends by reducing the time spent creating test cases and by improving the quality of your software releases.

Develop Test Cases During the Requirements Phase

As requirements are developed, create your test cases once each requirement is at the approval stage. This has many benefits. First, it forces the test team to scour the requirement and brings to light inadequate or inaccurate detail for the requirement. Additionally, it ensures the requirement is fully testable. Last, it allows the developers to review the test cases prior to coding. This illuminates your test plan and developers will spend time testing each scenario before it enters QA, saving valuable re-work time.

Implement Smart Requirement Numbering and Test Case Titles

Each requirement should be numbered in logical sequences and if there are multiple sections of the requirement, each section should be numbered separately. For example, let's assume you were creating a newsletter system for ABC client. Let's assume this is the first work you have done for ABC client. Consider numbering the requirement as ABC-0001. If you later do more work for this client, name that requirement ABC-0002. This allows you to quickly determine what client the requirement pertains to as well as how much work you have done for them. Let's also assume the first requirement for the newsletter system is to provide an Opt-In newsletter signup screen, but it has several level of details for the requirement. Consider identifying each level of detail with sections within the requirement:

- **ABC-0001.001 Newsletter Opt-In Screen**
- **ABC-0001.002 Send Opt-In Subscriber an Email Verifying Opt-In Profile**

By having several sections within the requirement and numbering them accordingly, it is easy to determine the distinct areas of the requirement.

First Priority is Positive Test Cases

When creating test cases, focus on **positive test cases** first. These are test cases that test the behavior of the requirement as it is designed. For example, in the Newsletter example above, these test cases will ensure that you can opt-in to the newsletter system. When creating test cases, include the section number in the title each test case. This allows you quickly determine sets of test cases for each requirement (and each section with the requirement). Below are some example positive test cases for the Newsletter example above:

- **ABC-0001.001 Check Opt-in Screen format** (Does the format of the screen match the requirement?)
- **ABC-0001.001 Check Opt-in Tab Order** (When tabbing between fields, does it

go from top to bottom, left to right?)

- **ABC-0001.001 Save Opt-In record with all required fields** (this will ensure that you can opt-in to the newsletter system)
- **ABC-0001.002 Check Subscriber Verification Email** (did we receive an email upon opting-in?)
- **ABC-0001.002 Check Subscriber Verification Email format** (Is the email formatted per the requirement?)

Notice how we included the requirement section in each test case title, providing quick traceability. This makes it easier for project managers to ensure there is adequate test case coverage for each section of the requirement and allows programmers to quickly determine what area of the requirement is covered by the test case.

Second Priority is Negative Test Cases

To ensure that the software works gracefully when users try things that were not intended, create a set of **negative test cases**. These are test cases that test the requirement in ways the developer may not have thought of. Below are some example negative test cases for the Newsletter example above:

- **ABC-0001.001 Enter an invalid email address** (It should ask them to re-enter a valid email address)
- **ABC-0001.001 Leave the required fields blank** (Last name, first name and email are required, should give a friendly error message if any of these are left blank)
- **ABC-0001.001 Bounds testing for each field** (Try entering 51 characters for the first and last name knowing that 50 characters is the maximum size allowed. Try entering 256 characters for the email address to ensure it only allows 255 characters.)
- **ABC-0001.001 Opt-in with an existing Opt-In account** (Opt-in with your email address, then try to opt-in again after the opt-in is successful, it should tell you that your email address is already opted-in.)

Other Test Cases

Once the positive and negative test cases are defined, consider adding additional test cases for:

1. **Performance** - Performance test cases ensure that the code will not become unusable with large amounts of data. For example, import 50,000 items and record the timings. Compare those timings to when you only have 50 items. For most applications, acceptable response time is anything under 5 seconds, while good response time is anything under 2 seconds. This may vary depending on the application and your own performance guidelines. The review should ensure that there are an adequate number of test cases for this.
2. **Security Testing** - If the feature is a secured feature, there should be security test cases to ensure that the correct rights are granted before specific actions can occur. The review should ensure that there are an adequate number of test cases for this.

3. **Regression Testing** - If the feature will become a base part of your product, identify specific test cases from this test suite that would be a good candidate for Regression Testing in future releases. The review should ensure that there are an adequate number of test cases for this.

Helpful Templates

Below are some helpful templates to aid you in developing software solutions on-time and on-budget:

- **Pragmatic Agile Development (PAD) Overview** - http://www.PragmaticSW.com/PADOverviewPresentation.pdf
- **PAD Road Map** - http://www.PragmaticSW.com/Pragmatic/Templates/RoadMap_Template.pdf
- **PAD Best Practices Excerpt** - http://www.PragmaticSW.com/PADBestPracticesExcerpt.pdf
- **Additional PAD Information** - http://www.pragmaticsw.com/PADOverview.pdf
- **Project Management Guidelines** - http://www.PragmaticSW.com/Pragmatic/Templates/ProjectMgtGuidelines.rtf
- **Functional Specifications** - http://www.PragmaticSW.com/Pragmatic/Templates/FunctionalSpec.rtf
- **Architectural Overview** - http://www.PragmaticSW.com/Pragmatic/Templates/ArchitectureOverview.rtf
- **Detailed Design** - http://www.PragmaticSW.com/Pragmatic/Templates/DetailedDesign.rtf
- **Strategic Planning Document** - http://www.PragmaticSW.com/Pragmatic/Templates/StrategicPlanning.rtf
- **Test Design** - http://www.PragmaticSW.com/Pragmatic/Templates/TestDesign.rtf
- **Risk Assessment** - http://www.PragmaticSW.com/Pragmatic/Templates/Risk%20Assessment.rtf
- **Weekly Status** - http://www.PragmaticSW.com/Pragmatic/Templates/WeeklyStatusRpt.rtf
- **User Acceptance Test Release Report** - http://www.PragmaticSW.com/Pragmatic/Templates/UATRelease.rtf
- **Post Mortem Report** - http://www.PragmaticSW.com/Pragmatic/Templates/PostMortem.rtf
- **All Templates** - http://www.PragmaticSW.com/Templates.htm
- **Prior Newsletters** - http://www.PragmaticSW.com/Newsletters.htm
- **Software Planner** - http://www.SoftwarePlanner.com/SoftwarePlannerPro.asp
- **Defect Tracker** - http://www.DefectTracker.com
- **Remoteus (Remote Desktop Sharing)** - http://www.PragmaticSW.com/Remoteus.asp

About the Author

Steve Miller is the President of **Pragmatic Software** (http://www.PragmaticSW.com). With over 21 years of experience, Steve has extensive knowledge in project management, software architecture and test design. Steve publishes a monthly newsletter for companies that design and develop software. You can read other newsletters at http://www.PragmaticSW.com/Newsletters.asp. Steve's email is steve.miller@PragmaticSW.com.

Pragmatic Software Co., Inc.
383 Inverness Parkway
Suite 280
Englewood, CO 80112

Phone: 303.768.7480
Fax: 303.768.7481
Web site:
http://www.PragmaticSW.com
E-mail: info@PragmaticSW.com