Top 10 Negative Test Cases

September 2007 - Pragmatic Software Newsletters

Negative test cases are designed to test the software in ways it was not intended to be used, and should be a part of your testing effort. Below are the top 10 negative test cases you should consider when designing your test effort:

1. **Embedded Single Quote** - Most SQL based database systems have issues when users store information that contain a single quote (e.g. John's car). For each screen that accepts alphanumeric data entry, try entering text that contains one or more single quotes.
2. **Required Data Entry** - Your functional specification should clearly indicate fields that require data entry on screens. Test each field on the screen that has been indicated as being required to ensure it forces you to enter data in the field.
3. **Field Type Test -** Your functional specification should clearly indicate fields that require specific data entry requirements (date fields, numeric fields, phone numbers, zip codes, etc). Test each field on the screen that has been indicated as having special types to ensure it forces you to enter data in the correct format based on the field type (numeric fields should not allow alphabetic or special characters, date fields should require a valid date, etc).
4. **Field Size Test -** Your functional specification should clearly indicate the number of characters you can enter into a field (for example, the first name must be 50 or less characters). Write test cases to ensure that you can only enter the specified number of characters. Preventing the user from entering more characters than is allowed is more elegant than giving an error message after they have already entered too many characters.
5. **Numeric Bounds Test -** For numeric fields, it is important to test for lower and upper bounds. For example, if you are calculating interest charged to an account, you would never have a negative interest amount applied to an account that earns interest, therefore, you should try testing it with a negative number. Likewise, if your functional specification requires that a field be in a specific range (e.g. from 10 to 50), you should try entering 9 or 51, it should fail with a graceful message.
6. **Numeric Limits Test -** Most database systems and programming languages allow numeric items to be identified as integers or long integers. Normally, an integer has a range of -32,767 to 32,767 and long integers can range from -2,147,483,648 to 2,147,483,647. For numeric data entry that do not have specified bounds limits, work with these limits to ensure that it does not get an numeric overflow error.
7. **Date Bounds Test -** For date fields, it is important to test for lower and upper bounds. For example, if you are checking a birth date field, it is probably a good bet that the person's birth date is no older than 150 years ago. Likewise, their birth date should not be a date in the future.
8. **Date Validity -** For date fields, it is important to ensure that invalid dates are not allowed (04/31/2007 is an invalid date). Your test cases should also check for leap years (every 4th and 400th year is a leap year).

9. **Web Session Testing -** Many web applications rely on the browser session to keep track of the person logged in, settings for the application, etc. Most screens in a web application are not designed to be launched without first logging in. Create test cases to launch web pages within the application without first logging in. The web application should ensure it has a valid logged in session before rendering pages within the application.
10. **Performance Changes -** As you release new versions of your product, you should have a set of performance tests that you run that identify the speed of your screens (screens that list information, screens that add/update/delete data, etc). Your test suite should include test cases that compare the prior release performance statistics to the current release. This can aid in identifying potential performance problems that will be manifested with code changes to the current release.

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