HOW TO WRITE BETTER TEST CASES

Refs: Presentation by by Dianne L. Runnels, CQA, CSTE

How to Write Better Test Cases

- Test cases and software quality
- Anatomy of a test case
- Improving testability
- Improving productivity
- The seven most common mistakes
- Case study

Test cases and software quality

- Test cases are valuable assets
- What is the risk of bad cases?
- How do you lower costs by improving quality?
- Is quality subjective?
- How can quality be measured?

What is a test Case ?

- **Definition:** A test that (ideally) executes a single well defined test objective (*Testing Computer Software – Kaner, Faulk, Nguyen*)
- **Definition:** A specific set of test data and associated procedures developed for a particular objective (*IEEE 729-1983*)

Why write test case?

- Accountability
- Reproducibility
- Tracking
- Automation
- To find bugs
- To verify that tests are being executed correctly
- For compliance
- To measure test coverage

What is a good test case?

- Accurate - tests what it's designed to test
- Economica- no unnecessary steps
- Repeatable, reusable - keeps on going
- Traceable - to a requirement
- Appropriate - for test environment, testers
- Self standing - independent of the writer
- Self cleaning - picks up after itself

Test Case Criteria

- An excellent test case satisfies the following criteria:
  - Reasonable probability of catching error
  - Neither too simple nor too complex
  - Not redundant with other tests
  - Best of its breed
  - Makes program failures obvious

Test Case Criteria

How is a good test ?

- Refers and maintains tight control over specific test data
- Has detailed enough Test Design Steps.
- Is aware of the Tester's experience
- Has clear criteria for Pass or Fail

Test Case Criteria

A Not-so-Good Test ?

- Leaves it up to the user to find test data
- Gives very high level instructions that leave too much room for "artistic interpretation"
- Does not consider the Tester's experience
- Leaves out follow-up verification steps which make it difficult to determine Pass or Fail criteria.

Anatomy of a test case

- Statement of purpose, what is being tested
- Method, how it will be tested
- Setup, environment, data
- Steps - actions and expected results, implied in a
- table or matrix
- Proofs, files, printouts, reports, screen grabs (optional)

Test Case Essentials

Things usually to include in a test case:

- Tracking Information
- **Test Case ID**
- **Test Description**
  - **Purpose/Objective/Title**
  - **Methods, how it will be tested**
  - **Environments**
  - **Procedures/Steps**
  - **Script**
- Parameters/Data
  - Input
  - Output
  - Default
- Call to system (printer, system clock, available RAM, APIs)
- **Expected Result**
- ***Observed Result***
- ***Pass/Fail/Blocked/Skipped***
- ***Bug ID***
- Notes/Comments
- Environment

Test Case Essentials

Test case objective

- The most important part of a test case is the 1-line title describing the objective of the test
- That 1-line title can be called
  - Test Title
  - Test Name
  - Test Case
  - Test Objective
  - Test Goal/Purpose

Test Case Essentials

Test case objective

- It is most important because
  - It gives the reader a description and idea of the test
  - A good test name makes review easier
  - Easier to pass to another person
  - Easier to pass to automation team
  - In many cases, may be the only part of a test case documented

Test Case Objective Syntax

Action + Function + Operating Condition

In which:

- Function may be function, feature, validation point
- Operating Condition may be data
- Action:
  - Verify
  - Test
  - Validate
  - Prove
  - Execute
  - Print
  - Calculate
  - Run
  - …any action verb

Test Case Examples

Action	Function	Operating Condition
Run	annual report	from standard data (file location)
Run	annual report	on Day 1 of fiscal year
Run	annual report	from empty spreadsheet
Run	annual report	on last day of fiscal year

Validation Points

Tests need validation points

- This is the expected result.
- Write it.
- Define clearly state: what behavior, result or point you are attempting to validate.

Types of test cases

- "A rose by any other name smells as sweet"
- Step-by-step or word/action instructions
- Table, matrix
- Script for record/playback or performance test
- All need the same structure (anatomy)

Test Case Template

TC ID	Description	Steps	Expected Result	Observed result	Status
TC00 1	*/*Test objective*/*	*/*Very clear and specific steps*/* <u>Pre-condition:</u> <u>Steps:</u> 1. Action 1 2. Action 2 …….	*/*you need to pre determine what your program is supposed to do*/*	*/* write down the result that you get when excecute the test case*/*	Passed/Failed

Step-by-Step

Step	Action	Result
1	Enter new name and address. Press <OK>.	Displays screen 008 new name details.
2	Fill all blanks with natural data. Make screen grab. Press <OK>.	Displays screen 005 maintenance.
3	Click on <Inquiry> button.	Displays screen 009 inquiry details.
4	Enter name from screen grab. Press <OK>.	Displays screen 010 record detail.
5	Compare record detail with screen grab.	All details match exactly.

Data Matrix

Date	Hired After 1/96	401K	Life Insurance	Payment Computation
10/25/99	Y	1	3	$24.50
1/4/98	Y	3	1	$34.00
3/6/96	N	2	5	$48.00
8/15/96	Y	2	5	$86.25
8/15/96	N	2	5	$105.00

Automated script

```
# Open the Fax Form
    menu_select_item ("File;Fax Order...");
    set_window("Fax Order");

    # Retrieve the Fax Order information and compare it to data from
    the main window
    edit_get_text("Arrival:", text);
    if(main_data["arr_time"] != text)
    {
            failure_msg = arrival_fr_mismatch;
            result = FAIL;
```

Improving testability - language

- Testability = easy to test
- Use active case, do this, do that
- System displays this, does that
- Simple, conversationalanguage
- Exact, consistent names of fields, not generic
- Don’t explain Windows basics

Improving testability - length

- 10-15 steps or less, unless user cannot save work
- Uniform time to test
- Wide range of testers
- Pros and cons of cumulative cases
- Order of cases follows business scenarios

Improving productivity

- with templates
- Prevents blank page panic
- Assists the disorganized
- Builds in standards
- Prints spiffy looking tests
- Assists testers to find information

Improving productivity-with clones

- "Save As" - the sweetest command
- Start seeing variables
- Use "Replace" and proofread
- Use stored text, macros
- Don't forget to plagiarize, piggyback

Improving productivity with test management software

- Single best investment to improve productivity
- More than templates on steroids
- Makes outlining and writing easier
- Allows cloning (copying) of steps, cases, sets
- Easy to add, move, delete cases and steps
- Automatically renumbers

The seven most common mistakes

- Making cases too long
- Incomplete, incorrect, or incoherent setup
- Leaving out a step
- Naming fields that changed or no longer exist
- Unclear whether tester or system does action
- Unclear what is a pass or fairesult
- Failure to clean up

Sudden project changes

- Requirements changes - be informed, build in variables, make deals
- Schedule changes - impact of delay, quick shifts, skinny dipping
- Staff changes - software knowledge first,requirements, prototypes, test cases

Protecting assets

- Maintain test cases
- Configuration management standards
- Be sure to include data files
- Keep your own index
- Leverage them for product knowledge

Test case checklist

Quality Attributes
o Accurate - tests what the description says it will test.
o Economical - has only the steps needed for its purpose
o Repeatable, self standing - same results no matter who tests it.
o Appropriate - for both immediate and future testers
o Traceable - to a requirement
o Self cleaning - returns the test environment to clean state

Structure and testability

o Has a name and number

o Has a stated purpose that includes what requirement is being tested

o Has a description of the method of testing

o Specifies setup information - environment, data, prerequisite tests, security access

o Has actions and expected results

o States if any proofs, such as reports or screen grabs, need to be saved

o Leaves the testing environment clean

o Uses active case language

o Does not exceed 15 steps

o Matrix does not take longer than 20 minutes to test

o Automated script is commented with purpose, inputs, expected results

o Setup offers alternative to prerequisite tests, if possible

o Is in correct business scenario order with other tests

Configuration management

- o Employs naming and numbering conventions
- o Saved in specified formats, file types
- o Is versioned to match software under test
- o Includes test objects needed by the case, such as databases
- o Stored as read
- o Stored with controlled access
- o Stored where network backup operates
- o Archived off-site