

Black Box Software Testing

(Professional Seminar)

Cem Kaner, J.D., Ph.D.

Professor of Computer Sciences
Florida Institute of Technology

Section:24

Requirements Analysis For Test Documentation

Summer, 2002

Contact Information:

kaner@kaner.com

www.kaner.com (testing website)

www.badsoftware.com (legal website)

I grant permission to make digital or hard copies of this work for personal or classroom use, with or without fee, provided that (a) copies are not made or distributed for profit or commercial advantage, (b) copies bear this notice and full citation on the first page, and if you distribute the work in portions, the notice and citation must appear on the first page of each portion, (c) each page bear the notice "Copyright (c) Cem Kaner" or if you changed the page, "Adapted from Notes Provided by Cem Kaner". Abstracting with credit is permitted. The proper citation for this work is Cem Kaner, *A Course in Black Box Software Testing (Professional Version)*, Summer-2002, www.testing-education.org. To copy otherwise, to republish or post on servers, or to distribute to lists requires prior specific permission and a fee. Request permission to republish from kaner@kaner.com.

Problems with the (allegedly) standard approach

It is essential to understand your requirements for test documentation.

Unless following a “standard” helps you meet your requirements, it is empty at best, anti-productive at worst.

Problems with the (allegedly) standard approach

IEEE Standard 829 for Software Test Documentation

- Test plan
- Test-design specification
- *Test-case specification*
 - » *Test-case specification identifier*
 - » *Test items*
 - » *Input specifications*
 - » *Output specifications*
 - » *Environmental needs*
 - » *Special procedural requirements*
 - » *Intercase dependencies*
- Test-procedure specification
- Test-item transmittal report
- Test-log

*We often see
one or more
pages per
test case.*

Problems with the (allegedly) standard approach

- What is the documentation cost per test case?
- What is the maintenance cost of the documentation, per test case?
- If software design changes create documentation maintenance costs, how much inertia do we build into our system? How much does extensive test documentation add to the cost of late improvement of the software? How much should we add?
- What inertia is created in favor of invariant regression testing?
- Is this incompatible with exploratory testing? Do we always want to discourage exploration?

Problems with the (allegedly) standard approach

- What is the impact on high-volume test automation?
- How often do project teams start to follow 829 but then give it up mid-project? What does this do to the net quality of the test documentation and test planning effort?
- WHAT REQUIREMENTS DOES A STANDARD LIKE THIS FULFILL?
- WHICH STAKEHOLDERS GAIN A NET BENEFIT FROM IEEE STANDARD DOCUMENTATION?
- WHAT BENEFITS DO THEY GAIN, AND WHY ARE THOSE BENEFITS IMPORTANT TO THEM?

Standards:

ISO 9000-3

The basic thinking underlying ISO 9000 goes like this:

- Decide what you're going to do (create and document a process).
- Do what you said you'd do.
- Document what you've done in a way that lets an auditor check what you've done against your process.

Risks

- It can be difficult to build in room for exploratory testing. To some degree (perhaps a large degree), this depends on the auditor.
- The style and extent of documentation needed to clue in a third party (auditor) might go beyond your internal needs. On finite budgets, where do we draw the line?
- What is the best tradeoff between process development and test execution? This must be a practical question, not a religious question.

Requirements

- There are many different notions of what a good set of test documentation would include. Before spending a substantial amount of time and resources, it's worth asking what documentation should be developed (and why?)
- *Test documentation is expensive and it takes a long time to produce. If you figure out some of your main requirements first, you might be able to do your work in a way that achieves them.*

Defining documentation requirements

- Stakeholders, interests, actions, objects
 - » Who would use or be affected by test documentation?
 - » What interests of theirs does documentation serve or disserve?
 - » What will they do with the documentation?
 - » What types of documents are of high or low value?
- Asking questions
- Context-free questions
- Context-free questions specific to test planning
- Evaluating a plan

Discovering Requirements

Requirements

- Anything that drives or constrains design

Stakeholders

- Favored, disfavored, and neutral stakeholders

Stakeholders' interests

- Favored, disfavored, and neutral interests

Actions

- Actions support or interfere with interests

Objects

Exercise

1. List the Stakeholders

- Favored
- Disfavored
- Neutral stakeholders

2. For each Stakeholder, list her Interests

- Favored
- Disfavored
- Neutral interests

3. For each Interest, list Actions

- Actions support an interest
- Actions interfere with an interest

Exercise

Objects: The Stuff You Create

- Such as features, data of the product

For each object, what is its relationship

- to a stakeholder,
- a stakeholder's interest, or
- in the actions the stakeholder wants to take or will have taken on her?

What is your group's mission?

Find important problems

Assess quality

Certify to standard

Fulfill process mandates

Satisfy stakeholders

Assure accountability

Advise about QA

Advise about testing

Advise about quality

Maximize efficiency

Minimize time

Minimize cost

The quality of testing depends on which of these possible missions matter and how they relate.

Many debates about the goodness of testing are really debates over missions and givens.

Test Docs Requirements Questions

- Is test documentation a product or tool?
- Is software quality driven by legal issues or by market forces?
- How quickly is the design changing?
- How quickly does the specification change to reflect design change?
- Is testing approach oriented toward proving conformance to specs or nonconformance with customer expectations?
- Does your testing style rely more on already-defined tests or on exploration?
- Should test docs focus on what to test (objectives) or on how to test for it (procedures)?
- Should the docs ever control the testing project?

Test Docs Requirements Questions

- If the docs control parts of the testing project, should that control come early or late in the project?
- Who are the primary readers of these test documents and how important are they?
- How much traceability do you need? What docs are you tracing back to and who controls them?
- To what extent should test docs support tracking and reporting of project status and testing progress?
- How well should docs support delegation of work to new testers?
- What are your assumptions about the skills and knowledge of new testers?
- Is test doc set a process model, a product model, or a defect finder?

Test Docs Requirements Questions

- A test suite should provide prevention, detection, and prediction. Which is the most important for this project?
- How maintainable are the test docs (and their test cases)? And, how well do they ensure that test changes will follow code changes?
- Will the test docs help us identify (and revise/restructure in face of) a permanent shift in the risk profile of the program?
- Are (should) docs (be) automatically created as a byproduct of the test automation code?

Example: Early vs. Late Planning

Benefits of early planning

- Scheduling and staffing (*unless things change*)
- Early opportunity to review (*if people can understand your work*)

Costs of early planning

- Any delays in finding bugs are expensive. You should start *finding* bugs ASAP.
- Early investment in test case design is risky if the product design is subject to change.
- You will keep learning over time. Planning early sometimes precludes groups from inventing new tests later.
- Depth of testing should reflect risks, many of which are unclear until mid-project.

MY GOAL: Do just-in-time test planning without losing key benefits of early work.

Ultimately, write a mission statement

Try to describe your core documentation requirements in one sentence that doesn't have more than three components.

Examples:

- The test documentation set will primarily support our efforts to find bugs in this version, to delegate work, and to track status.
- The test documentation set will support ongoing product and test maintenance over at least 10 years, will provide training material for new group members, and will create archives suitable for regulatory or litigation use.

