

Black Box Software Testing

(Professional Seminar)

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

Professor of Computer Sciences
Florida Institute of Technology

Section:12

Reusable Test Matrices (and Domain Testing)

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.

Using Test Matrices for Routine Issues

After testing a simple numeric input field a few times, you've learned the drill. The boundary chart is reasonably easy to fill out for this, but it wastes your time.

Use a test matrix to show/track a series of test cases that are essentially the same.

- For example, for most input fields, you'll do a series of the same tests, checking how the field handles boundaries, unexpected characters, function keys, etc.
- As another example, for most files, you'll run essentially the same tests on file handling.

The matrix is a concise way of showing the repeating tests.

- Put the objects that you're testing on the rows.
- Show the tests on the columns.
- Check off the tests that you actually completed in the cells.

Typical Uses of Test Matrices

- You can often re-use a matrix like this across products and projects.
- You can create matrices like this for a wide range of problems. Whenever you can specify multiple tests to be done on one class of object, and you expect to test several such objects, you can put the multiple tests on the matrix.
- Mark a cell green if you ran the test and the program passed it.
- Mark the cell red if the program failed and write the bug number of the bug report for this bug.
- Write (in the cell) the automation number or identifier or file name if the test case has been automated.

Matrices

Problems?

- **What if your thinking gets out of date? (What if this program poses new issues, not covered by the standard tests?)**
- **Do you need to execute every test every time? (or ever?)**
- **What if the automation ID number changes? -- We still have a maintenance problem but it is not as obscure.**

Examples of integer-input tests

Nothing

Valid value

At LB of value

At UB of value

At LB of value - 1

At UB of value + 1

Outside of LB of value

Outside of UB of value

0

Negative

At LB number of digits or chars

At UB number of digits or chars

Empty field (clear the default value)

**Outside of UB number of digits or
chars**

Non-digits

**Wrong data type (e.g. decimal into
integer)**

Expressions

Space

Non-printing char (e.g., Ctrl+char)

**DOS filename reserved chars (e.g., "\
* . :")**

Upper ASCII (128-254)

Upper case chars

Lower case chars

**Modifiers (e.g., Ctrl, Alt, Shift-Ctrl,
etc.)**

Function key (F2, F3, F4, etc.)

Error Handling when Writing a File

full local disk

almost full local disk

write protected local disk

damaged (I/O error) local disk

unformatted local disk

**remove local disk from drive after
opening file**

**timeout waiting for local disk to
come back online**

**keyboard and mouse I/O during save
to local disk**

**other interrupt during save to local
drive**

power out during save to local drive

full network disk

almost full network disk

write protected network disk

damaged (I/O error) network disk

**remove network disk after opening
file**

timeout waiting for network disk

**keyboard / mouse I/O during save to
network disk**

**other interrupt during save to
network drive**

**local power out during save to
network**

**network power during save to
network**

Homework: File Name Matrix

Tomorrow, we'll illustrate the process of creating test matrices by brainstorming a specific one.

- File name field
- Windows 95 / 98 / NT / 2000
- Treat the path as a separate issue

Please spend 15 minutes at home writing a list of file name tests. Bring your notes with you.

Homework: File Name Matrix

You could do this for almost any type of variable. For example, imagine listing all of the hardware (including connection, power, etc.) error conditions that could cause failure of a file save operation.

At this point, though, it is interesting to look at variables that have values that can be treated as equivalent. Simpler examples are:

- string fields, numeric (rational numbers), percentages
- date fields, time fields

Matrix Construction Brainstorm

Take up the assignment with a brainstorming session.

Expect to spend 1-1.5 hours if the students are prepared and 2-3 hours if they are not.

Brainstorming Rules:

- Don't criticize others' contributions
- Jokes are OK, and are often valuable
- Goal is to get lots of ideas, filter later.
- Recorder and facilitator keep their opinions to themselves.

Matrix Construction Brainstorm

Facilitating and Recording Rules:

- Exercise patience: Goal is to get lots of ideas.
- Encourage non-speakers to speak.
- Use multiple colors when recording
- Echo the speaker's words.
- Record the speaker's words
- The rule of three 10's. Silence is OK.
- Switch levels of analysis.
- Some references:
 - » S. Kaner, Lind, Toldi, Fisk & Berger, *Facilitator's Guide to Participatory Decision-Making*
 - » Freedman & Weinberg, *Handbook of Walkthroughs, Inspections & Technical Reviews*
 - » Doyle & Straus, *How to Make Meetings Work*.

