

# *Black Box Software Testing*

## *(Academic Course - Fall 2001)*

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

Florida Institute of Technology

**Section: 19 :**

Exploratory Testing in Pairs

Contact Information:

**kaner@kaner.com**

**www.kaner.com (testing practitioners)**

**www.badsoftware.com (software law)**

**www.testingeducation.org (education research)**

Copyright (c) Cem Kaner 2001.

I grant permission to make digital or hard copies of this work for personal or classroom use, 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. Abstracting with credit is permitted. The proper citation for this work is Cem Kaner, *A Course in Black Box Software Testing (Academic Version)*, Fall 2001, [www.testing-education.org](http://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.

Copyright (c) Cem Kaner 2001.

# *Paired Exploratory Testing--Acknowledgment*

- The following, paired testing, slides developed out of several projects.
- We particularly acknowledge the help and data from participants in the First and Second Workshops on Heuristic and Exploratory Techniques (Front Royal, VA, November 2000 and March 2001, hosted by James Bach and facilitated by Cem Kaner), those being Jon Bach, Stephen Bell, Rex Black, Robyn Brilliant, Scott Chase, Sam Guckenheimer, Elisabeth Hendrickson, Alan A. Jorgensen, Brian Lawrence, Brian Marick, Mike Marduke, Brian McGrath, Erik Petersen, Brett Pettichord, Shari Lawrence Pfleeger, Becky Winant, and Ron Wilson.
- Additionally, we thank Noel Nyman and Ross Collard for insights and James Whittaker for co-hosting one of the two paired testing trials at Florida Tech.
- A testing pattern on paired testing was drafted by Brian Marick, based on discussions at the Workshop on Patterns of Software Testing (POST 1) in Boston, January 2001 (hosted primarily by Sam Guckenheimer / Rational and Brian Marick, facilitated by Marick). The latest draft is at "Pair Testing" pattern) (<<http://www.testing.com/test-patterns/patterns/pair-testing.pdf>>).

# *Paired Exploratory Testing*

- Based on our (and others') observations of effective testing workgroups at several companies. We noticed several instances of high productivity, high creativity work that involved testers grouping together to analyze a product or to scheme through a test or to run a series of tests. We also saw/used it as an effective training technique.
- In 2000, we started trying this out, at WHET, at Satisfice, and at one of Satisfice's clients. The results were spectacular. We obtained impressive results in quick runs at Florida Tech as well, and have since received good reports from several other testers.

# *Paired Programming*

- Developed independently of paired testing, but many of the same problems and benefits apply.
- The eXtreme Programming community has a great deal of experience with paired work, much more than we do, offers many lessons:
  - Kent Beck, *Extreme Programming Explained*
  - Ron Jeffries, Ann Anderson & Chet Hendrickson, *Extreme Programming Installed*
- Laurie Williams of NCSU does research in pair programming. For her publications, see <http://collaboration.csc.ncsu.edu/laurie/>

# *What is Paired Testing*

- Two testers and (typically) one machine.
- Typically (as in XP)
  - Pairs work together voluntarily. One person might pair with several others during a day.
  - A given testing task is the responsibility of one person, who recruits one or more partners (one at a time) to help out.
- We've seen stable pairs who've worked together for years.
- One tester strokes the keys (but the keyboard may pass back and forth in a session) while the other suggests ideas or tests, pays attention and takes notes, listens, asks questions, grabs reference material, etc.

# *A Paired Testing Session*

- **Start with a charter**
  - Testers might operate from a detailed project outline, pick a task that will take a day or less
  - Might (instead or also) create a flipchart page that outlines this session's work or the work for the next few sessions.
    - An exploratory testing session lasts about 60-90 minutes.
  - The charter for a session might include what to test, what tools to use, what testing tactics to use, what risks are involved, what bugs to look for, what documents to examine, what outputs are desired, etc.

# *Benefits of Paired Testing*

- Pair testing is different from many other kinds of pair work because testing is an \*idea generation activity\* rather than a plan implementation activity. Testing is a heuristic search of an open-ended and multi-dimensional space.
- Pairing has the effect of forcing each tester to explain ideas and react to ideas. When one tester must phrase his thoughts to another tester, that simple process of phrasing seems to bring the ideas into better focus and naturally triggers more ideas.
- If faithfully performed, we believe this will result in more and better ideas that inform the tests.

# *Benefits of Paired Testing*

- Generate more ideas
  - Naturally encourages creativity
  - More information and insight available to apply to analysis of a design or to any aspect of the testing problem
  - Supports the ability of one tester to stay focused and keep testing. This has a major impact on creativity.
- More fun

# *Benefits of Paired Testing*

- Helps the tester stay on task. Especially helps the tester pursue a streak of insight (an exploratory vector).
  - A flash of insight need not be interrupted by breaks for note-taking, bug reporting, and follow-up replicating. The non-keyboard tester can:
    - Keep key notes while the other follows the train of thought
    - Try to replicate something on a second machine
    - Grab a manual, other documentation, a tool, make a phone call, grab a programmer--get support material that the other tester needs.
    - Record interesting candidates for digression
- Also, the fact that two are working together limits the willingness of others to interrupt them, especially with administrivia.

# *Benefits of Paired Testing*

- **Better Bug Reporting**

- Better reproducibility
- Everything reported is reviewed by a second person.
- Sanity/reasonability check for every design issue
  - (example from Kaner/Black on Star Office tests)

- **Great training**

- Good training for novices
- Keep learning by testing with others
- Useful for experienced testers when they are in a new domain

# *Benefits of Paired Testing*

- **Additional technical benefits**
  - Concurrency testing is facilitated by pairs working with two (or more) machines.
  - Manual load testing is easier with several people.
  - When there is a difficult technical issue with part of the project, bring in a more knowledgeable person as a pair

# *Risks and Suggestions*

- Paired testing is *not* a vehicle for fobbing off errand-running on a junior tester. The pairs are partners, the junior tester is often the one at the keyboard, and she is always allowed to try out her own ideas.
- Accountability must belong to one person. Beck and Jeffries, et al. discuss this in useful detail. One member of the pair owns the responsibility for getting the task done.
- Some people are introverts. They need time to work alone and recharge themselves for group interaction.
- Some people have strong opinions and don't work well with others. Coaching may be essential.

# *Risks and Suggestions*

- Have a coach available.
  - Generally helpful for training in paired testing and in the conduct of any type of testing
  - When there are strong personalities at work, a coach can help them understand their shared and separate responsibilities and how to work effectively together.