Synergistic Virtualized Crowdsourced Agile Testing in the Cloud as a Service

Michael Bolton
DevelopSense
http://www.developsense.com
Denver SQuAD
March 2011

Testers: Get Out of the Quality Assurance Business!

Michael Bolton
DevelopSense
http://www.developsense.com
Denver SQuAD
March 2011

Updates

- This presentation is ALWAYS under construction
- Updated slides at http://www.developsense.com/past.html
- All material comes with lifetime free technical support

Let's Start With a Simple Question:

What is "quality"?

Quality Answers

- Quality is "value to some person(s)"
  – Jerry Weinberg
- (with respect to testing) "…who matter."
  – James Bach and Michael Bolton
- Decisions about quality are always political and emotional
  - made by people with the power to make them
  - made with the desire to *appear* rational
  - yet ultimately based on how those people *feel*

If you're a tester, do you…

design the product?
negotiate customer contracts?
write the code?
hire the programmers?
decide which bugs to fix?
allocate staff?
set the schedule?
set the product scope?
fix problems in the code?
decide on raises?
allocate training budgets?
produce manuals?
choose the development model?
fire some programmers?
control the budget?
set the company's strategic direction?

No?

Then how, exactly, do you ASSURE quality?

How Can You, Tester, Assure Quality?

YOU CAN'T.

But not to worry. That's not the tester's job.

We Can't Assure Quality

but we can TEST.

A Computer Program

```
A set of instructions
for a computer.
```

See the Association for Software Testing's
Black Box Software Testing Foundations course, Kaner & Bach

A House

A set of building materials, arranged in the "House" design pattern.

A House

Something for people to live in.

Kaner's Definition of a Computer Program

- A computer program is
- a *communication*
- among several people
- and computers
- separated over distance and time
- that contains instructions that can be run on a computer.

The purpose of a computer program is to provide **value** to **people**

The code is not the product.
The code is part of the product.
The product is
a problem solved for a customer.

Implications of Kaner's Definition

- A computer program is **far more** than its code
- A software product is **far more** than the instructions for the device
- Quality is **far more** than the absence of errors in the code.
- Testing is **far more** than writing code to assert that other code returns some "correct" result
- Testing is **not about** "writing test cases".

Quality is value to some person(s).

Testing is an **investigation** of code, systems, people, and the relationships between them.

What Is Testing?

Software testing is the investigation of *systems* composed of people, computer programs, and related products and services.

- Excellent testing is not a branch of computer science
  - focus only on program code and functions, and you leave out questions of *value* and other relationships that include people
- To me, excellent testing is more like *anthropology*
  - highly multidisciplinary
  - doesn't look at a single part of the system
- Anthropologists investigate many things
  - biology (human mechanisms; human "code" and "hardware")
  - archaeology (human history)
  - linguistics (human communication)
  - cultures (what it means to be human)

So What Is Testing?

- "Questioning a product in order to evaluate it"
  - James Bach
- "Gathering information with the intention of informing a decision."
  - Jerry Weinberg
- "A technical, empirical investigation of a product, done on behalf of stakeholders, with the intention of revealing quality-related information of the kind that they seek."
  - Cem Kaner

No assurances!

Testing Is More Than *Checking*

- Checking is a process of confirming and verifying existing beliefs
  - Checking can (and I argue, largely should) be done by automation
  - It is a *non-sapient* process

See http://www.developsense.com/2009/08/testing-vs-checking.html

Oh no! What Does "*Non-Sapient*" Mean?

- A ***non-sapient*** activity can be performed

by a machine
that *can't* think
(but it's quick and precise)

by a human who has been
instructed NOT to think
(and that's slow and erratic)

What Is *Sapience*?

- A ***sapient*** activity is one that requires a thinking human to perform
- "Manual" testing is a misnomer
- We test not only for repeatability, but also for *adaptability*, *value*, and *threats to value*

This kind of testing CAN NOT be scripted

What else *don't* we script?

```
Management Case #3412
=====================
Preconditions:

Ensure date is March 21; time 9:23am
Ensure staffing level = 4 members
Set coffee cup to full

Management Steps:
1) Receive annual departmental budget for $752,688.
2) Allocate $501,472 to burdened employee cost.
3) Allocate remaining $251,256 to equipment and tools.
3a) Leave training and book budgets at $0.
4) Receive email from development manager requesting 75 hours
   of testing work on Confabulator IV project. Offer 40.
5) Turn down 3:30pm meeting requested by lead programmer.
6) 3:15 leave office.

Postcondition: Observe whether par has been achieved
   on 4th hole.
```

Like a good manager,

- A good tester doesn't simply follow scripts asking

Pass or Fail?

- A good tester *investigates* and asks

Is there a problem here?

Besides…

- Automation cannot
  - program a script
  - investigate a problem you've found
  - determine the meaning or significance of a problem
  - decide that there's a problem with a script
  - escape a script problem you've identified
  - determine the best way to phrase a report
  - unravel a puzzling situation

But automation CAN help YOU do those things.

Is Regression Your Biggest Risk?

- Before the Agile Manifesto was declared, a group of experienced test managers reported that regression problems ran from 6-15% of discovered problems
- In Agile shops, we now (supposedly) have
  - TDD
  - unit tests
  - pairing
  - configuration management
  - build and version control
  - continuous integration
- With all this regression testing, is regression a big risk?
- If so, is high-level scripted checking (whether we call it ATDD or not) a good way to fix it?

Regression Problems Are *Symptoms*

- If you see a consistent pattern of house fires
  - no individual fire is the city's biggest problem
  - bigger problems: shoddy construction, and/or arsonists
- If you see a consistent pattern of regression
  - the failing tests are not the organization's biggest problem
  - you might want to raise awareness that there's a favourable environment for regression

Testing Is More Than Checking

- Testing is an ongoing, continuously re-optimizing process of

exploration, discovery, investigation, and learning

Testing is *Exploring*

- Our community* sees testing as exploration, discovery, investigation, and learning
  - Testing can be *assisted* by machines, but can't be done by machines alone
  - Testing is a *sapient* process

I can't test, but I can help *you* act on test ideas.

See http://www.developsense.com/2009/08/testing-vs-checking.html
See also Kaner & Hoffman's work on exploratory test automation.

*** The Context-Driven Testing community**

What IS Exploratory Testing?

- I follow (and to some degree contributed to) Kaner's definition, which was refined over several peer conferences through 2007:

Whoa. Maybe it would be a good idea to keep it brief *most* of the time…

Exploratory software testing is…

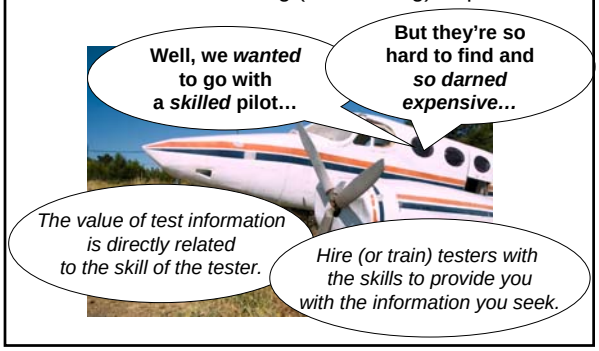
- a style of software testing
- that emphasizes the personal freedom and responsibility
- of the individual tester
- to continually optimize the value of his or her work
- by treating test design, test execution, test result interpretation, and test-related learning
- as mutually supportive activities
- that run in parallel
- throughout the project.

"Parallel test design, test execution, and learning."

See Kaner, "Exploratory Testing After 23 Years", www.kaner.com/pdfs/ETat23.pdf

Yes, Exploratory Testing Requires Skill

- But doesn't ANY testing (worth doing) require skill?



Irony Alert!

- We talk about *checking* with test cases
- We often manage *testing* with checklists

Oh well!

Smart people can deal with stuff like this.

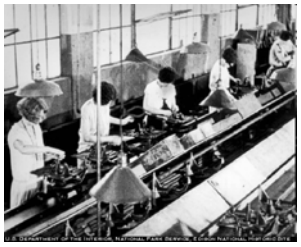
So What Are We Testers?

Skilled investigators

The tester doesn't have to reach conclusions or make recommendations about how the product *should* work. **Her task is to expose credible concerns to the stakeholders**.
- Cem Kaner, *Approaches to Test Automation*, 2009 (my emphases)

We Are Sensory Instruments

Software Development Is Not Much Like Manufacturing



- In manufacturing, the goal is to make zillions of widgets *all the same*.
- Repetitive checking makes sense for manufacturing, but…
- In software, creating zillions of identical copies is not the big issue.
- If there is a large-scale production parallel, it's with *design*.

Software Development Is More Like Design

- If existing products sufficed, we wouldn't create a new one, thus…
- Each new software product is novel to some degree, and that means a new set of relationships and designs every time.
- New designs cannot be checked only; they must be *tested*.

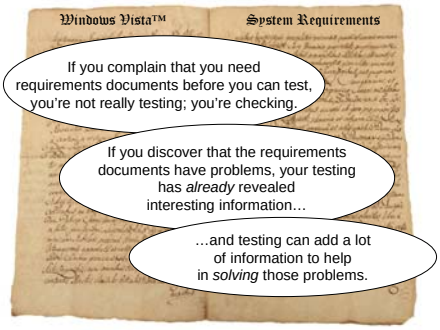
Testing of Design Is Like CSI

- There are many tools, procedures, sources of evidence.
- Tools and procedures don't *define* an investigation or its goals.
- There is too much evidence to test anything like all of it
- Tools are often expensive
- Investigators are working under conditions of uncertainty and extreme time pressure
- Our clients (not we) make the decisions about how to proceed based on the available evidence

These ideas come largely from Cem Kaner, *Software Testing as a Social Science*
http://www.kaner.com/pdfs/KanerSocialScienceSTEP.pdf

Viewing Testing as an Investigative Service Solves Many Problems

Viewing Testing as an Investigative Service Solves Many Problems



The problem is not that testing is the bottleneck. The problem is that you don't know what's in the bottle. That's a problem that testing addresses.

More Relevant Comparisons

- Investigative reporters and journalists
  - What's actually going on? What's the story?
- Anthropologists
  - What do people in the real world *actually do*?
- Historians
  - What can we learn from the past?
- Field botanists
  - Why does this thrive over here, but not over there?
- Philosophers
  - What do we know? How do we know we know it?
- Film critics
  - Will this movie appeal to its intended audience?

Can't We *Help* With Quality Tasks?

- Sure; (to me, at least) development teams should be autonomous and self-organizing
  - when you're providing other services to your team, that might be good and very useful.
  - but that could be a problem if you're not also *testing.*
- To the extent that your work is
  - requested by your colleagues
  - appreciated by your colleagues
  - not busy work
  - not busybody work

  …rock on! Help out! But also *test*.

Where Do We Go From Here?

We must build knowledge and skills

What Skills and Knowledge?

- Critical thinking
- General systems thinking
- Design of experiments; threats to validity
- Visualization and data presentation
- Observation
- Reporting
- Rapid learning
- Programming

Not a comprehensive list!

What Skills and Knowledge?

- REAL Measurement
- Anthropology
- Teaching
- Risk analysis
- Cognitive psychology
- Economics
- Epistemology
- *Test framing*

Not a comprehensive list!

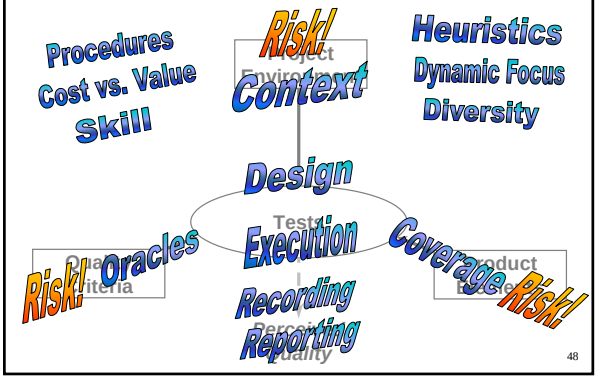
What is test framing?

Test framing is
the set of logical connections
that structure and inform a test.

Framing ~= Traceability

- Framing is, in essence, traceability…
- …but typically we hear people talk of traceability in an impoverished way: between *tests* and requirements *documents*
- Can you demonstrate traceability between tests and implicit requirements?
- Can you demonstrate traceability between the test result and the mission?

A Heuristic Test Strategy Model



Book References: Cem Kaner

- The Ongoing Revolution in Software Testing
  - http://www.kaner.com/pdfs/TheOngoingRevolution.pdf
- Software Testing as a Social Science
  - http://www.kaner.com/pdfs/KanerSocialScienceSTEP.pdf
- Software Engineering Metrics: What Do They Measure and How Do We Know? (with Walter P. Bond)
  - www.kaner.com/pdfs/metrics2004.pdf
- Approaches to Test Automation
  - http://www.kaner.com/pdfs/kanerRIM2009.pdf
- Lessons Learned in Software Testing
  - Kaner, Bach, & Pettichord

Book References: Jerry Weinberg

- Perfect Software and Other Illusions About Testing
- Quality Software Management
  - Volume 1: Systems Thinking
  - Volume 2: First Order Measurement
- Quality Software Management: Requirements Before Design
- An Introduction to General Systems Thinking
- The Psychology of Computer Programming
  - Jerry Weinberg

Book References

- The Black Swan
- Fooled by Randomness
  – Nassim Nicholas Taleb
- Secrets of a Buccaneer Scholar
  – James Bach
- Everyday Scripting in Ruby
  – Brian Marick
- How To Program
  – Chris Pine
- Sciences of the Artificial
  – Herbert Simon
- How Doctors Think
  – Jerome Groopman

Book References

- Blink
  – Malcolm Gladwell
- Tools of Critical Thinking
  – David Levy
- Mistakes Were Made (But Not By Me)
  – Carol Tavris and Eliot Aronson
- How to Lie With Statistics
  – Darrell Huff
- The Visual Display of Quantitative Information
- Envisioning Information
- Visual Explanations
- Beautiful Evidence
  – Edward Tufte