The Metrics Minefield

Michael Bolton, DevelopSense
Toronto, Ontario, Canada
michael@developsense.com
http://www.developsense.com
+1 (416) 656-5160

STARWest 2012

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

I'm Michael Bolton
Not the singer. Not the guy in Office Space. No relation.

Michael Bolton
http://www.developsense.com
michael@developsense.com
Twitter: @michaelbolton

Notes & Acknowledgements
• Acknowledgements to my colleagues and mentors:
  • James Bach
  • Jon Bach
  • Cem Kaner
  • Jerry Weinberg
• Thanks to our hosts at STAR West
  • Griffin Jones
• Get the complete version of this presentation
  • from my USB key
  • with a business card
  • from http://www.developsense.com/past.html

These notes are rough drafts. I won’t talk about everything in them, and I will talk about stuff NOT in them.

I Don’t Hate Numbers
• I love numbers so much that I can’t stand to see them abused as they are by people in our profession.
• This workshop is designed to help you spot critical thinking errors that might cause you to miss observations and mislead your client—or yourself.
• The intention is not to suggest that measurement is useless—on the contrary—but instead to expand our notions of what measurement might be.

Imperfections in measurement are always a problem, but they're a devastating problem only when we don’t recognize them.
—Daniel Gilbert, Stumbling on Happiness

Copyright © Michael Bolton 2012
Exercise
Test Your Chair

Why Do We Measure?
- self-assessment and improvement
- evaluating project status
- evaluating staff performance
- informing others about the characteristics of the product
- informing external authorities about the characteristics of the product

—Kaner and Bond

See Robert Austin, Measuring and Managing Performance in Organizations

But there’s an over-riding reason...

Fear of irrationality

Quantifying something complex provides a kind of observational integrity or consistency, but with a side effect: information loss. Think of all testing and measurement as providing partial answers that might be useful.

Ignore the risks of measurement, and distortion and dysfunction are likely. Manage to your metric, and dysfunction is guaranteed.

What Is Measurement?

Measurement is the empirical, objective assignment of numbers, according to a rule derived from a model or theory, to attributes of objects or events with the intent of describing them.

—Jerry Weinberg

We waste time and effort when we try to obtain six-decimal-place answers to whole-number questions.

A Synthesis

Measurement is the art and science of making reliable observations, based on modeling and comparison of objects, attributes, or events, for the purposes of understanding distinctions, making assessments, and informing evaluations.

—Jerry Weinberg

What do we want?

What Is Measurement?

Measurement is the art and science of making reliable and significant observations.

—Jerry Weinberg

What do we want?

Since the time of Aristotle (at least), we’ve known about two kinds of measurement that inform decisions:
- “Two pounds of meat”
- “Too much”, “too little”, “just right”.

Measurement might be qualitative or quantitative, but assessment and evaluation are always qualitative:

We waste time and effort when we try to obtain six-decimal-place answers to whole-number questions.

• Source: “Software Engineering Metrics: What Do They Measure and How Do We Know?”
  - Cem Kaner and Walter P. Bond
  - Note that this is a description of quantitative measurement
How Do We Measure?

- Third-order measurement
  - highly instrumented, used to discover natural laws
  - "What will happen? What always happens?"

- Second-order measurement
  - often instrumented, used to refine first-order observation
  - used to tune existing systems
  - "What’s really going on here, exactly? What’s happening?"

How Else Do We Measure?

- First-order measurement
  - minimal fuss, direct observation, minimal instrumentation
  - used to inform a control action OR to prompt search for more refined information
  - "What’s going on? What should we do? Where should we look?"

Weinberg suggests that, in software development, we’re obsessed with trying to make third- and second-order measurements when first-order measurements might be all we need—and are cheaper in every way.

Why Prefer First-Order Measures?

- When you’re driving, are you mostly concerned about…
  - your velocity, acceleration, vehicle mass, drag co-efficient, frictional force? (third-order)
  - your engine temperature, RPMs, and current rate of gas consumption? (second-order)
  - looking out the window to avoid hitting something?

I’ve observed many projects that have crashed because managers were focused on the dashboard instead of the traffic and obstacles around them, and the road ahead.

What kind of driver do you trust?

Exercise

How Do You Measure When You’re Driving?

Questioning Measurement

What might we observe?
  - What are we choosing NOT to observe?
  - What are we not observing by accident?
  - What comparisons can we make?
    - are those comparisons valid? relevant?
    - are those comparisons reliable?
  - What are we trying to assess?
    - How will we respond to the information?
    - Will we use it to control? Or to learn?

Quality measurement depends upon our skill at observation, what we’re comparing, and the validity of the models that we’re using for assessment.

What Is a Metric?

- A metric is a measurement function that maps a number onto an observation.
  - like a mathematical function that establishes a relationship between one number and another
  - Constructs are ideal objects or attributes whereby we relate something observable to a category
    - like a label for a class, pattern, model…
    - when measuring, we apply ideas that allow us to classify something as an instance of or consistent with that construct (or not)
  - A construct is typically described in terms of its fit with several variables
**Construct Validity & External Validity**

- **Construct validity** is (informally) the degree to which your attributes and measurements can justified within an experiment or observation.
  - How do you know that you’re measuring what you think you’re measuring?
- **External validity** is the degree to which your experiment or observation can be generalized to the world outside.
  - How do you know that your experiment or observation will be relevant at other times or in other places?

**Kaner & Bond’s Tests For Construct Validity**

- What is the purpose of your measurement? The scope?
- What is the attribute you are trying to measure?
- What are the scale and variability of this attribute?
- What is the instrument you’re using? What is its scale and variability?
- What function (metric) do you use to assign a value to the attribute?
- What’s the natural scale of the metric?
- What is the relationship of the attribute to the metric’s value?
- What are the natural, foreseeable side effects of using this measure?

The essence of good measurement is a model that incorporates answers to questions like these.

If you don’t have solid answers, you aren’t doing measurement; you are just playing with numbers.

**Darrell Huff’s Tests for Statistical Validity**

- Who says so?
- How do they know?
- What’s missing?
- Did somebody change the subject?
- Does it make sense?

*From How to Lie with Statistics, originally published in 1954.*

**Control vs. Inquiry Measurement**

- A control measurement is a measurement that drives decisions.
  - Any measurement you use to control a self-aware system will be used by that system to control YOU.
- An inquiry measurement is any measurement that helps you ask the right questions at the right time.
  - Inquiry measurements are also vulnerable to gaming, but the stakes are far lower, so there’s less incentive for manipulation.

This slide is taken from the work of my colleague, James Bach. http://www.satisfice.com

**Obstacles**

- In software testing (my field) we talk about finding bugs, and reporting them.
- There’s something worse than a bug: an issue—something that slows down or prevents your ability to find a problem.
- Mechanisms intended to provide management information often ignore issues.
- Bug reporting systems focus on bugs, and suppress reporting of issues.
- Mechanisms intended to provide management control often create issues.

**A Thought Experiment: Test Session Effectiveness**

- A “perfectly effective” testing session is one entirely dedicated to test design, test execution, and learning
  - a “perfect” session is the exception, not the rule
- Test design and execution tend to contribute to test coverage
  - varied tests tend to provide more coverage than repeated tests
- Setup, bug investigation, and reporting take time away from test design and execution
- Suppose that testing a feature takes two minutes
  - notice that “testing” and “feature” are weak constructs, but we use them to model an issue and make a point
- Suppose also that it takes eight extra minutes to investigate and report a bug
  - “investigation and reporting” and “bugs” are also weak constructs
- In a 90-minute session, we can run 45 feature tests—as long as we don’t find any bugs

Copyright © Michael Bolton 2012
How Do We Spend Time?
(assuming all tests below are good tests)

<table>
<thead>
<tr>
<th>Module</th>
<th>Bug reporting/investigation (time spent on tests that find bugs)</th>
<th>Test design and execution (time spent on tests that find no bugs)</th>
<th>Number of tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>A (good)</td>
<td>0 minutes (no bugs found)</td>
<td>80 minutes (45 tests)</td>
<td>45</td>
</tr>
<tr>
<td>B (okay)</td>
<td>10 minutes (1 bug, 1 test)</td>
<td>80 minutes (40 tests)</td>
<td>41</td>
</tr>
<tr>
<td>C (bad)</td>
<td>80 minutes (8 bugs, 8 tests)</td>
<td>10 minutes (5 tests)</td>
<td>13</td>
</tr>
</tbody>
</table>

In the first instance, our coverage is great—but if we’re being assessed on the number of bugs we’re finding, we look bad.
In the second instance, coverage looks good, and we found a bug, too.
In the third instance, we look good because we’re finding and reporting lots of bugs—but our coverage is suffering severely. A system that rewards us or increases confidence based on the number of bugs we find might mislead us into believing that our product is well tested.

What Happens The Next Day?
(assume 6 minutes per bug fix verification)

<table>
<thead>
<tr>
<th>Fix verifications</th>
<th>Bug reporting and investigation today</th>
<th>Test design and execution today</th>
<th>New tests today</th>
<th>Total over two days</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 min</td>
<td>0 min</td>
<td>45 min</td>
<td>45 min</td>
<td>90 min</td>
</tr>
<tr>
<td>6 min</td>
<td>10 min (1 new bug)</td>
<td>74 min (37 tests)</td>
<td>38 min</td>
<td>138 min</td>
</tr>
<tr>
<td>48 min</td>
<td>40 min (4 new bugs)</td>
<td>2 min (1 test)</td>
<td>5 min</td>
<td>57 min</td>
</tr>
</tbody>
</table>

Finding bugs today means…
VERIFYING FIXES LATER…
which means…
EVEN SLOWER TESTING or…
EVEN LESS COVERAGE...or both.

Testing vs. Investigation

- I just gave you a compelling-looking table
- notice that, in real life, without evaluation we don’t know anything about…
  - the quality and relevance of the tests
  - the quality and relevance of the bug reports
  - the skill of the testers in finding and reporting bugs
  - the complexity of the respective modules
  - the role of luck

…but if we ask better questions, instead of letting data make our decisions, we’re more likely to make progress.

What Are The Factors of a “Test Case”??

- Supports debugging: Will it provide useful results for the programmer?
- Repeatability: does the test reveal a problem?
- Mutability: can the test be adapted to other test ideas?
- Complexity: are there interesting interactions between components?
- Simplicity: does the test successfully isolate a problem of interest?
- Accountability: can you explain, justify, and prove that you run the test?
- Equipment cost: do you need specialized gear to run the test?
- Development cost: what resources are required to design the test?
- Setup cost: what time and resources are required to prepare for the test?
- Execution time: how long does it take the test to run?
- Reporting time: what effort is required to communicate the results?
- Opportunity cost: what valuable work could you do instead of this test?

Many of these ideas come from Kaner & Bach’s Black Box Software Testing Course

Don’t Just Count Them!

- You’ve just seen at least 24 factors by which we might describe or evaluate a given test
- Bugs have similar numbers of factors, if only we pause to think about it
- Many factors aren’t usefully quantifiable
  - yet they might be supremely important
  - people base decisions on politics and emotions
  - people have emotional reactions to software
- Models may leave out many dimensions
  - some of which might also be very important
- Testers are even more complex
  - tester effectiveness needs multi-dimensional measures
Counting Ideas

• Don’t count test cases
  • test cases are ideas; ideas aren’t things
  • counting ideas is reification error
• Don’t judge product health by bug counts
  • try bug stories instead
• Don’t measure testers by counting bug reports
  • testers may be doing other things of great value besides writing bug reports

“If you evaluate testers by counting bug reports, I guarantee that your testers are misleading you.”
“Test cases are like briefcases; I’m a ‘famous testing expert’, and I can’t tell whether 1000 test cases are good or bad until I see them.

James Bach

Lines of Code Metrics

Here are two lines of code from the same program. Are they equivalent?

```javascript
obj.visibility=v;
for(i=0;!x&&d.layers&&i<d.layers.length;i++)
  x=MM_findObj(n,d.layers[i].document);
```

In the first example, it appears as though one bit is being set. In the second, multiple values are (conditionally) being initialized, compared, set, incremented, referenced, or dereferenced.

This is like counting tricycles and space shuttles as equivalent items.

Beware: Precision vs. Accuracy

• What time is it, exactly, right now?
• My watch say 5:35:21, which is precise, but it’s wrong.
• Many people will offer precise numbers—but will they be accurate?

Heuristic: In software development processes, the more precise the number provided, the more likely it is to be based on an unsupportable model.

Testing is a Structured Process

In excellent testing, one structure tends to dominate all the others:

The Testing Story

Expert testers construct a compelling story of their testing. It is this story that gives testing a backbone.

To test is to compose, edit, narrate, and justify THREE stories.

A story about the status of the PRODUCT...
  ...about how it failed, and how it might fail...
  ...in ways that matter to your various clients.
A story about HOW YOU TESTED it...
  ...how you configured, operated and observed it...
  ...about what you haven’t tested, yet...
  ...and won’t test, at all...
A story about how GOOD that testing was...
  ...what the risks and costs of testing are...
  ...what made testing harder or slower...
  ...how testable (or not) the product is...
  ...what you need and what you recommend.

Hold It!

Aren’t stories subjective by nature?

Sure.
All observations—including quantitative observations—depend on observers, their choices about what to observe, and the quality of their observations.
Break Down the Testing Story

- Who are the characters? (People? Bugs?)
- What matters to them?
- What happens over time?
- Why do they do what they do?
- When did the story happen?
- Where did they go?
- What did they do?

Why should we care?

The Power of Limited Information

- Snap judgments and heuristics are central to our decision-making process
- Paradoxically because they’re fast and frugal, they may be more accurate than complex and rigourous analysis. Too much information can add noise to an evaluation.

People often make snap (first-order) observations or decisions and then use (second-order) numbers to test them.

See Malcolm Gladwell, Blink, and Gerd Gigerenzer, Gut Feelings

Quantifying vs. Qualifying

- Comparisons and assessments aren’t necessarily numerical (ask Goldilocks).
- Numbers aren’t as descriptive as words and stories.
- Words can be vague or ambiguous, but numbers without clarifying words are just as bad or worse.
- Could you tell convincing, motivating stories?
- Could you use ranking or laddering?
- Could you use reports and opinions from multiple people?

Assessment Without Numbers

- Observation can go directly to first-order assessment without quantified measurement
- Ask what other modes, beside numerical ones, you could use for evaluation
  - start by asking what problem you want to solve or what situation you’d like to assess, what attributes to pay attention to, and why they matter
  - Instead of using numbers to tell the story, use numbers to prompt questions about the story
- If you’re worried that observations and assessments are subjective, ask several people who matter

Possibly Useful Measures

- The Binary Metric
  - “Any showstoppers?”
  - showstopper (n.) A problem that, in management’s judgment, makes more sense to fix than to ship.
- The Issues List
  - list bugs and issues by importance to some stakeholder; optionally rank them first
  - note that testability issues may be most important; problems that prevent or slow down testing mean that other problems have more places to hide
- The Furrowed Brow Test (a.k.a. Squirm Test)
  - announce that we’re planning to ship on schedule
  - observe postures, grimaces, furrowed brows, and squirming

Possibly Useful Measures

- Spend no more than a couple of minutes per day tracking time spent on
  - test design and execution
  - bug investigation and reporting
  - test setup
  - none of the above (non-testing work such as meetings, email, administrivia, etc.)
- Try tracking this to 5% - 10% granularity
  - finer granularity means that “time tracking” requires significant non-testing effort
  - do you include “time spent on time tracking” in your time tracking?
  - do you include “time spent on estimation” in your estimates?
The Metrics Minefield

Framing The Testing Story
For a given test, be prepared to describe…
• the question you wanted to ask and answer
• the test techniques you used
• the coverage you sought and obtained
• the oracles you used
• the results you observed

Tell The Testing Story
For a test cycle, be prepared to describe…
• the testing mission
• specific risks to address
• the diversity of your coverage, oracles, and techniques
• the tests you performed
• what you might be missing, and why it’s okay

Try Rubrics and Checklists
• Diversify your criteria
• Create checklists of desired behaviours
• Produce multidimensional tables for comparison
• Qualitative evaluation is okay
  • after all, evaluation is based on what we value
• Subjective evaluation is okay too, but again …
  • get multiple opinions from multiple sources; or
  • consider the differing values and observational modes of multiple constituencies

Try Other Modes of Assessment
• Try temperature readings
  • appreciations
  • new information
  • puzzles
  • complaints
• Recognize the ways in which data can be converted to information, and vice versa
• When pushed to provide numbers, provide several alternative interpretations

Reporting
• “Never give a number to a bureaucrat”
• Plum’s Second Law
• Emphasize stories and narratives
• Don’t produce, offer or accept a number without a story
  • lead with the story
  • show the multivariate nature of data
  • annotate charts or tables
  • note several possible interpretations
• Prefer direct observations and simple comparisons over derivative metrics
The Metrics Minefield

Estimation

- Change assumptions
  - Testing is a responsive activity
  - Think of prospecting: we don’t know when we’re going to find the gold
  - Most people, when estimating the test cycle, are really estimating the fix cycle
  - Testing could continue forever
  - Testing can be stopped at any time
- Ask “When do you want to ship?”
  - that’s how much time you have, and need, to test

Let’s Get Serious

- If our numbers are
  - multivariate
  - based on incomplete models
  - rough, first-order approximations

Let’s not overwhelm ourselves wasting time and money seeking bogus precision.

Three Heuristics

The numbers are not the story.
Seek information, not just data.
Always ask “Compared to what?”

Summary

- Testing is about noticing
- What haven’t you noticed today?
- Measurement is not analysis
- Measurement is a tool for analysis

“We shape our tools; thereafter, our tools shape us.”
---Marshall McLuhan

Finale: A Useful Measurement

- Weinberg’s Management Measurement
  - “How preoccupied the managers are with overly precise measurements for which they have no working models”

If you watch where people are looking, it can tell you something about where they’re not looking.
That may be where the biggest risks live.

Readings

- Exploring Requirements: Quality Before Design
- Gerard M. Weinberg
- “Software Engineering Metrics: What Do They Measure and How Do We Know?”
- Cem Kaner and Walter P. Bond
- How to Lie with Statistics
  - Darrell Huff
- The Black Swan
- Fooled by Randomness
  - Nassim Nicholas Taleb
- Reliability and Validity in Qualitative Research
  - Jerome Kirk and Mark Miller
Readings

- Experimental and Quasi-Experimental Designs for Generalized Causal Inference
  - Shadish, Cook, and Campbell
- Visual Explanations
  - Edward Tufte
- Stumbling on Happiness
  - Daniel Gilbert
- Why Does Software Cost So Much?
  - Tom DeMarco
- What’s Your Story?
  - Craig Wortmann
- Measuring and Managing Performance in Organizations
  - Robert D. Austin
- Freakonomics
  - Stephen Leavitt
- Perfect Software and Other Illusions About Testing
  - Gerald M. Weinberg