Illustrating the Quality Story

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

EuroSTAR
November 2011

Who I Am

Michael Bolton
DevelopSense, Toronto
Canada

mb@developsense.com

(416) 992-8378

http://www.developsense.com/past.html

Notes & Acknowledgements

- Acknowledgements to my colleagues and mentors:
  - James Bach
  - Jon Bach
  - Cem Kaner
  - Jerry Weinberg
- Thanks to our hosts at EuroSTAR
- 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.

Example: A Test Project

Total Bugs Reported

190

4

15 weeks
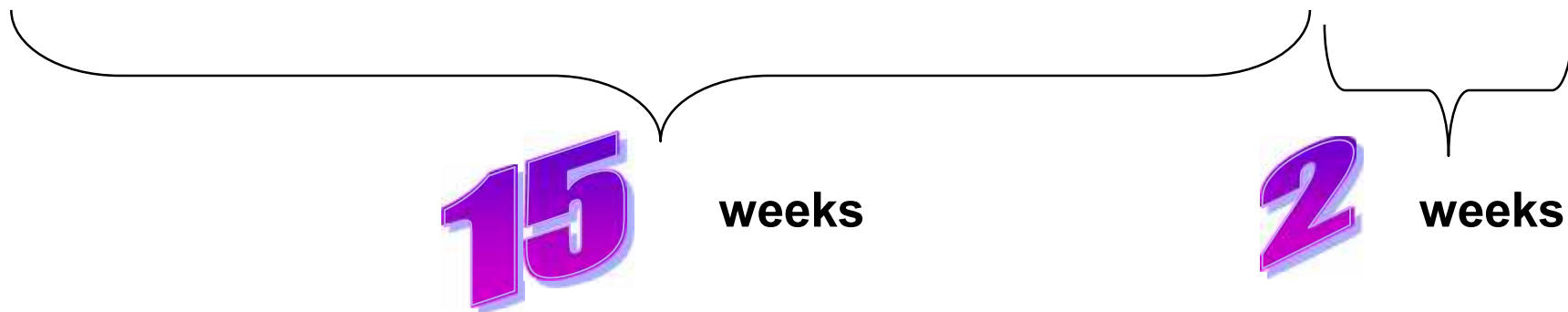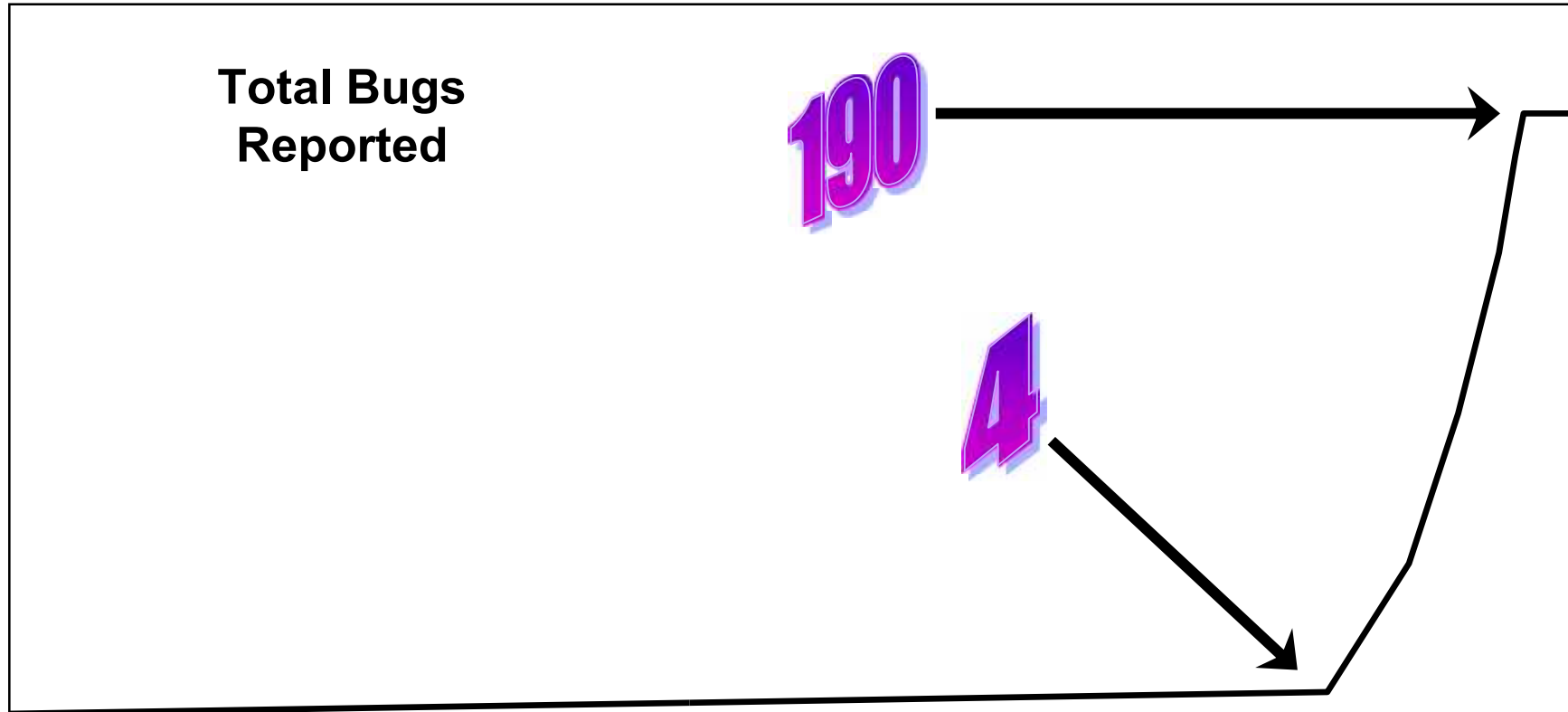
2 weeks

Example: A Test Project

- The original tester was faking it.

- A tiger team took over for the last 2 weeks.

- The reporting system broke down in the last several days before the beta was shipped. That's why the line goes flat– but reports were still made by hand.

Lesson:

- **Pay attention.**

- **Don't assume that all known problems are being reported.**

Measurement Skills

- What measurement skills can we list that will be useful for testers?

- What tactics can testers use in measurement?

- What issues must testers confront in measurement?

- What IS measurement anyway?

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…

…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.**

If testing is telling stories…

- Measurement is a *tool* or *medium* that can help give warrant to those stories.
- Measurement *illustrates* the stories of the quality of the work and the quality of the product.
- Measurement *is not* the story itself.
- As a tool, measurement must be subject to *its* own quality story.
- Tools reflect *what we are*.

Don't Be A Turkey

- Every day the turkey adds one more data point to his analysis *proving* that the farmer has the best interests of turkeys at heart.

- Hundreds of observations support his theory.

- Then, a few days before Thanksgiving…

Based on a story told by Nassim Taleb, who stole it from Bertrand Russell, who stole it from David Hume.

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 take you deeper into measurement, spotting 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, but 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

Exercise

Measure The Height of 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

<div align="right">

—Kaner and Bond

</div>

See http://www.kaner.com/pdfs/metrics2004.pdf
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. Expressing things in numbers comes with a side effect: *information loss*. As Cem Kaner describes the social sciences, think of 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."

- Source: "Software Engineering Metrics: What Do They Measure and How Do We Know?"
  - Cem Kaner and Walter P. Bond
  - http://www.kaner.com/pdfs/metrics2004.pdf

What happens when we walk through that definition?

Exercise

Using Kaner and Bond's definition, identify potential problems with
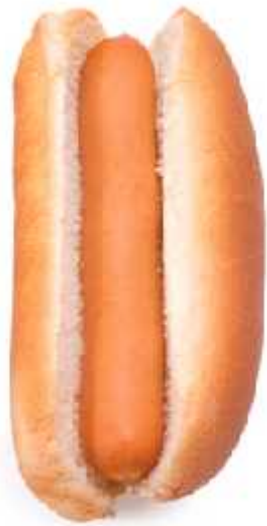a) the definition; and
b) measurement generally.

Exercise

Identify factors of measurement.

Measurement vs. Metrics

- A *measurement* is an observation that is related to something, based on some model or theory, for the purpose of making some distinction.

- The *metric* is the operation and the mathematical function that establishes or (or "maps") the relationship.

<

A

B

Theory:

If len(A) < len(B) then A is not really a foot-long dog.

Exercise

How could this go wrong?

Measurement Scales

equality (🔑 identity; constructs; this IS that)

nominal (🔑 labels; the number isn't quantity at all)

ordinal (🔑 order; number suggests a rank)

interval (🔑 differences, and equalities in them)

ratio (🔑 differences and proportions in differences)

derivative (🔑 combinations of the above)

**Mistake one kind of scale for another, and
you can guarantee invalid, unsupportable conclusions.**

See S.S. Stevens, "On the Theory of Scales of Measurement", *Science*, June 17, 1946

Nominal Scale

Ordinal Scale

Interval Scale

Gretzky is 2 minutes and 12 seconds faster than Liut.

2:21:45.91   2:33:56.74

1st

Ratio Scale

Gretzky is 8% faster than Liut.
In *this* race.

2:21:45.91

2:33:56.74

1st

What Statistics Can You Do for Each Scale?

- Nominal
  - assign labels
  - count an instance of a construct, the total instances of each, the grand total, the greatest number of instances (a.k.a. the mode)
- Ordinal
  - order by rank
  - identify the median (the middle element), divide into percentiles
- Interval
  - count units of distance between items
  - doesn't require a true zero point: think calendars, thermometers
  - identify the mean, calculate the standard deviation, identify why they've earned a given rank, etc. –but no zero-based stats
- Ratio
  - find multiples of distance between items; magnitudes
  - *does* require a true zero point
  - includes *fundamental* and *derived* measures

What Is Measurement?

> Measurement is the art and science of making reliable observations.
>
> —Jerry Weinberg, Quality Software Management Vol. 2

- 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".

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

- http://www.developsense.com/articles/2009-05-IssuesAboutMetricsAboutBugs.pdf
- http://www.developsense.com/articles/2009-07-ThreeKindsOfMeasurement.pdf
- http://www.developsense.com/articles/2007-11-WhatCounts.pdf

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?

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

What Might We Want to Measure?

What do you
want to measure?

How do we
measure it?

Exercise from Kaner, 2009

What Makes a Measurement *Good*?

What is Objectivity?

Three Important Kinds of Reliability

- Quixotic reliability
  - the measurement yields consistent results in different circumstances
  - could be the result of a broken instrument, "party line" answers
- Diachronic reliability
  - the measurement yields the same result when taken multiple times
  - only reliable for things that don't change in a changing world; "may deny history"
- Synchronic reliability
  - similarity of observations within the same time period
  - reveals potentially significant questions when it fails

- Kirk and Miller

Two Important Kinds of Validity

- Construct validity: to what degree is what you're measuring an instance of what you think you're measuring?
  - The "counting to one" problem
- External validity: to what degree are can your measurements support generalizations to cases outside your study?
  - The "so you measured; so what?" problem

Measurement without consideration of validity is playing with numbers.

Kaner & Bond on Measurement Validity

1. What is the purpose of your measurement?
2. What is the scope of the measurement?
3. What is the attribute you are trying to measure?
4. What are the scale and variability of this attribute?
5. What is the instrument you're using
6. What are the scale and variability of the instrument ?
7. What function (metric) do you use to assign a value to the attribute?
8. What's the natural scale of the metric?
9. What is the relationship of the attribute to the metric's output?
10. 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 Five Questions

- 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.

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? What's happening right now?"

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 tend to be much cheaper.

A Synthesis

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

- My intention here is to highlight
  - why we measure (assessment and evaluation)
  - how we measure (observation, and comparison based on models)
- Measurement might be qualitative or quantitative, but assessment and evaluation are always qualitative:

What do we want?

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 things (first-order)?

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

What kind of driver do you trust?

Possibly Useful First-Order Measures

- The Binary Release Metric
  - "Any showstoppers?" (A showstopper is a problem that makes more sense to fix than to release.)
  - IF answer == YES, then don't 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 posture, furrowed brows, and squirming

The Power of Limited Information

- Snap judgments and heuristics are central to our decision-making process
- Because they're fast and frugal, they may sometimes be more valuable than complex and rigourous analysis

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

Exercise: A Release Decision
## at Quarterdeck

- Karla (tester): Our product, QEMM-386 crashes at startup on a certain machine. Changing the disk controller fixes the problem.

- Phil (programmer): I've been working on this problem for three weeks, and I'm stuck. It's only Karla's cobbled-together machine anyway.

- What more information would help to inform this decision?

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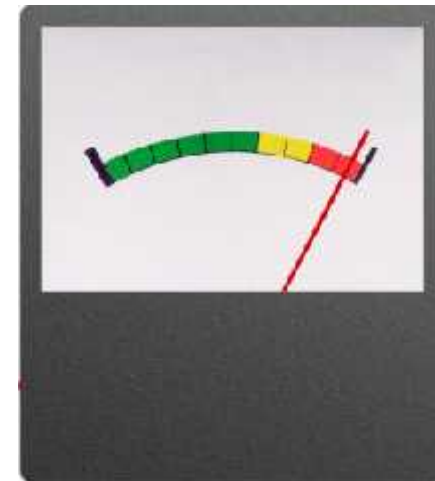
Text here is taken from the work of my colleague, James Bach.
http://www.satisfice.com

Control vs. Inquiry

- Remove *control metrics* that are linked to pay, bonuses, performance evaluation, etc.
  - control metrics trigger some action, usually automatically
  - a metric that is used to control something will eventually be used to control you
- Foster *inquiry metrics*
  - inquiry metrics prompt us to ask questions
- Relax measurement when the metric stops changing
  - if you're not obtaining new information, try measuring something else of interest for a while

Counting Bugs? Or Identifying Issues?

- In software testing 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.

- For a description of issues and how we might report them, see http://www.developsense.com/blog/2011/01/youve-got-issues/.

Lines of Code

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

```
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.

Exercise

What are the factors of a test case?

What Are The Factors of a "Test Case"?

Power: will this test reveal a problem?

Validity: is problem revealed a genuine problem?

Value: is the information is useful to your product, project, or client?

Credibility: will clients believe it represents things people will actually do?

Representative: is it a good example of plausible similar tests?

Motivating: will the test make us want to fix the problem it reveals?

Performable: Can the test be performed as designed?

Maintainable: Can the test be revised easily when the product changes?

Reusable: It is easy and inexpensive to reuse the test for other products?

Pop: Will the test challenge our assumptions and reveal new information?

Coverage: Will the test exercise the product in a unique way?

Easy to evaluate: Is there a clear answer to the question the test poses?

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

http://www.wtst.org/WTST7/BBSTwtst2008kanermeeting.pdf

What Are The Factors of a "Test Case"?

Supports debugging: Will it provide useful results for the programmer?

Repeatable: does the test reveal a problem consistently?

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**

http://www.wtst.org/WTST7/BBSTwtst2008kanermeeting.pdf

Questions

1) What are the easily measurable aspects of a test case?

2) What might such measures tell us?

3) How such measures mislead us?

Surrogate Measures

- We don't know how to measure
  - the quality of the product
  - the quality of testing
  - the quality of a tester
- So we count something easily countable
  - Requirements
  - Bug counts
  - Test cases
  - Test cases per requirement
  - Test cases per day
  - Defect escapes
- These are called *surrogate measures*
- Do they measure quality?

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 them
- 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 emphasizing bug stories
- 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

Let's Measure a Development Group

- DCE - Defect Containment Effectiveness
- DEP - Defect Escape Percentage
- UA - Urgent Action
- CSI - Customer Satisfaction Indicator
- TES - Team Effectiveness Scorecard
- DKLOC – Defects Per Thousand Lines of Code
- **If you think these are bogus measurements, can you explain why? Can you critique them effectively?**

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*?
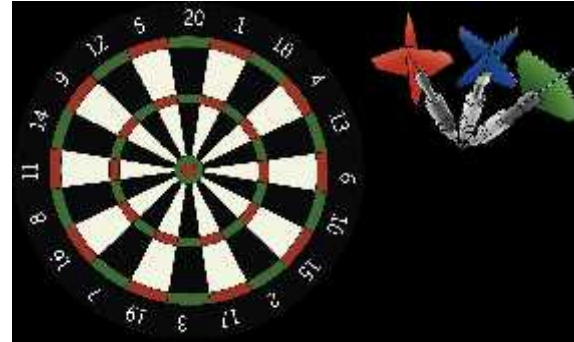
Precise!

Accurate!

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

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.

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?

What if you asked for "things you liked and things you didn't like"?

Assessment Without Numbers

- Observation can go directly to assessment without quantified measurement
  - i.e. first order 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*
- If you're worried that observations and assessments are subjective, ask *several* people who matter
- The object of the game is usually NOT mathematical precision; it's reliable observation

Qualifying Coverage

- We can compose, edit, narrate, and justify *stories* about our coverage

- We can use product elements, quality criteria, test coverage outlines, and risk lists to frame those stories

- We can even quantify coverage *with respect to a particular model* if we provide and illustrate a clear story about the model

Test Coverage

- The amount of the product that has been tested with respect to coverage criterion (that is, some idea about completeness).

- Model the product

- Cover the map!
  - "Maps" can be lists, tables, matrices, diagrams, graphs, mind maps,…

Extent of Coverage

1. Smoke and sanity

   - Can this thing even be tested at all?

2. Common, core, and critical

   - Can this thing do the things it *must* do?

   - Does it handle happy paths and regular input?

   - *Can* it work?

3. Complex, harsh, extreme and exceptional

   - Will this thing handle challenging tests, complex data flows, and malformed input, etc.?

   - *Will* it work?

Quality Criteria Coverage Matrix

Visualizing Behaviour: Blink Tests
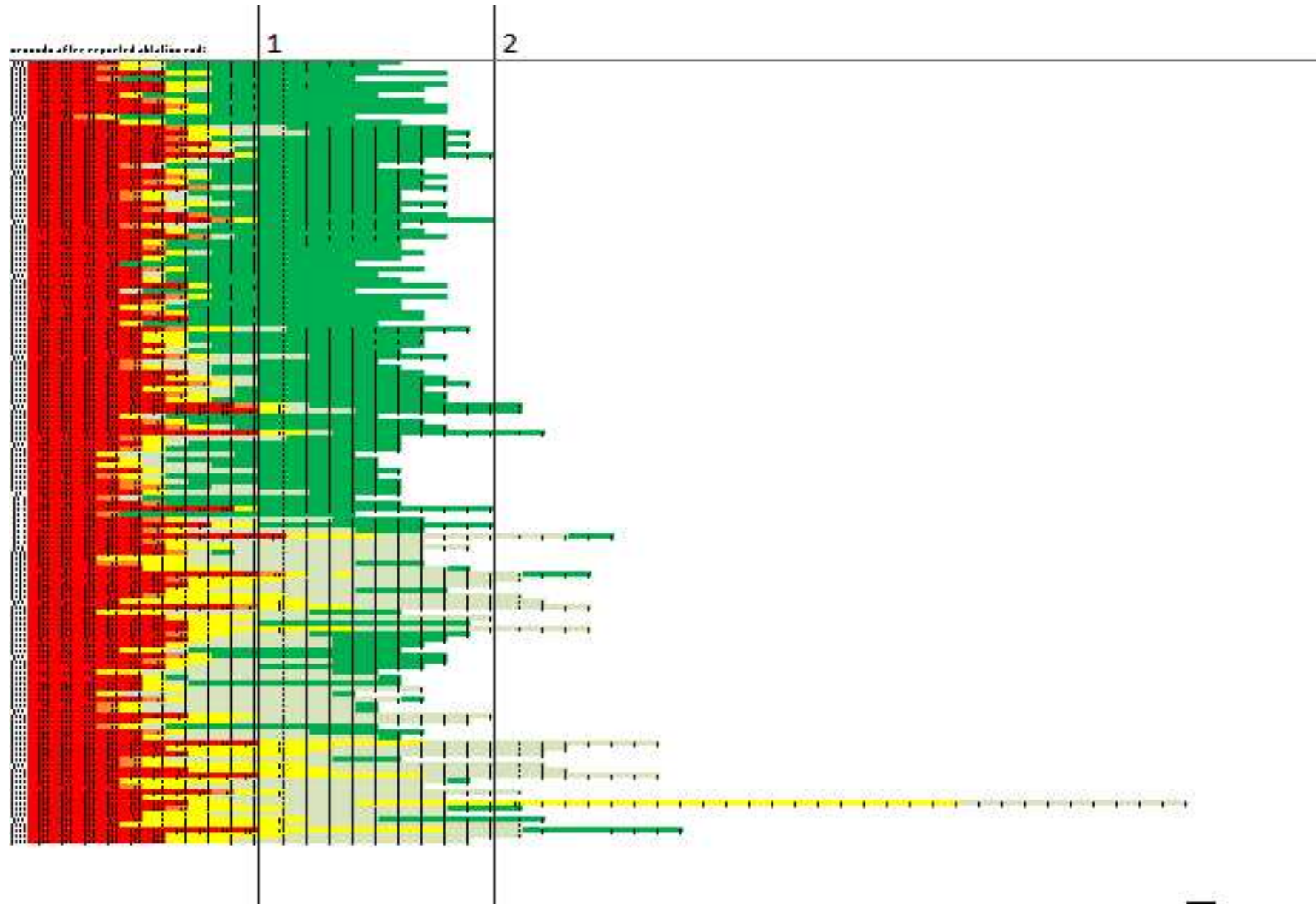
Visualizing Estimation

- project size (obvious size vs. hidden)
- project complexity
- development staff (people/skills)
- developer relations (are you in good collaboration?)
- desired quality level
- desired coverage level
- availability of requirement and design information
- testability
- routine interruptions (meetings, email, etc.)
- illnesses
- vacations
- extraordinary interruptions
- accounting and auditing
- change control processes
  - design
  - code
  - configuration
  - build
- test staffing (skills required, needed hiring)
- test administration (status reports, documentation, coordination cost)
- tester training and learning curve
- test data preparation
- test platform availability (delivery, installation, stability)
- test tooling
- the time scale of the estimate
  - required accuracy
  - required precision
- implicit or explicit commitments
- relationship between tester and client
- time required to prepare the estimate
- revision triggers
- number and significance of bugs
  - investigation and reporting slows down testing
- number of test cycles
  - driven by development work
- specific new features and design changes
  - driven by business needs
- recognition of implicit project or test scope
  - we may not have realized everything that was necessary
- specific new risks
  - emergent from test activity; may require new testing
- local dependencies
  - programmers "can't start X until we finish Y"
  - testers may be waiting on programmers
- third-party dependencies
  - someone can't finish Y until we *receive* Z
  - programmers or testers may be waiting on outsiders
- implementation/testing sequencing and delays
  - what is being delivered, and in what order
  - how quickly are fixes turned around?
- specific slippages
  - mini-disasters and Black Swans
- competing projects
  - when our project's schedule slips, what will happen?

In this list, testing has

• plenty of control over green factors;

• some control over yellow factors, and

• very little or none over the red ones.

*So why ask testers for an estimate?*

Estimation

- Question 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
    - It could continue forever
    - It can be stopped at any time

- Ask "When do you want to ship?"
 - that's how much time you have, *and need*, to test

http://www.developsense.com/blog/2010/10/...
…project-estimation-and-black-swans-part-5-test-estimation

Visualizing Test Results
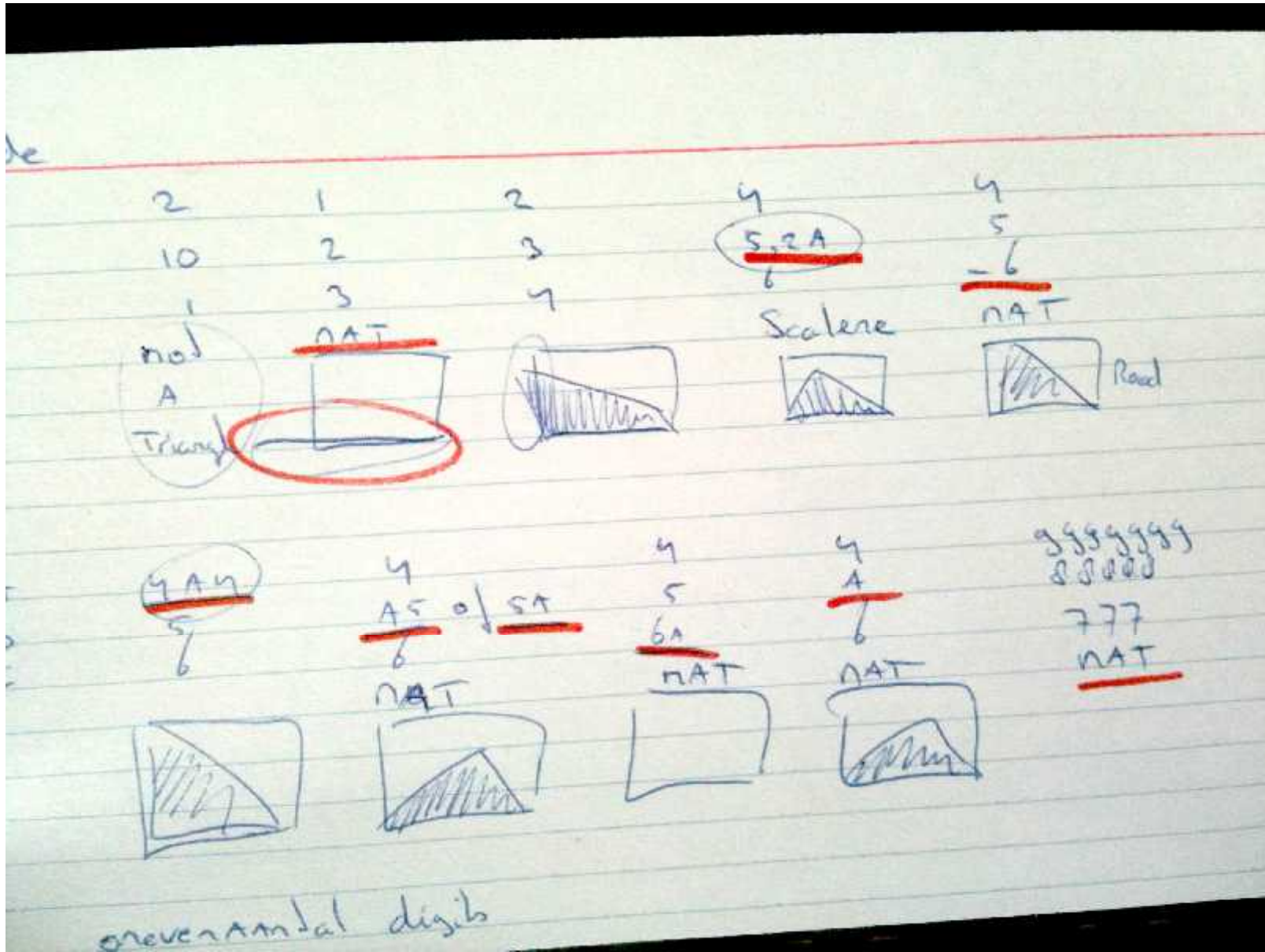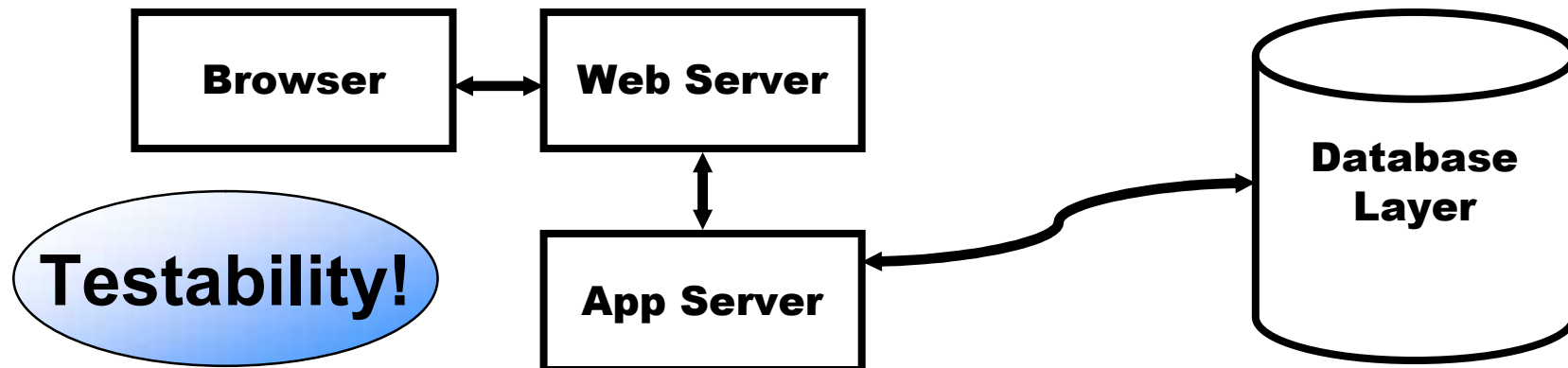
Visualizing Risk: *Analysis*

- [pointing at a box] *What if the function in this box fails?*

- *Can this function ever be invoked at the wrong time?*

- [pointing at any part of the diagram] *What error checking do you do here?*

- [pointing at an arrow] *What exactly does this arrow mean? What would happen if it was broken?*
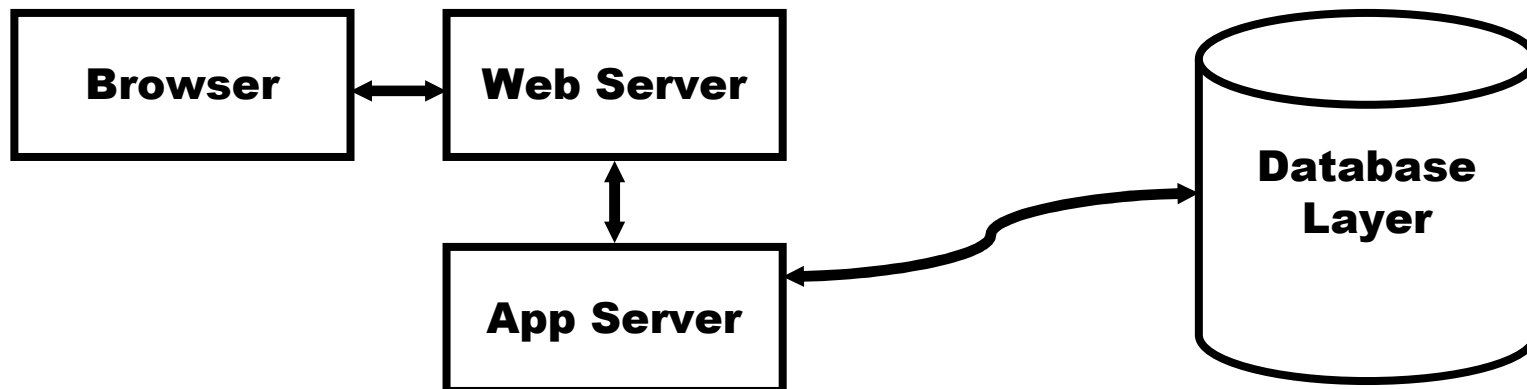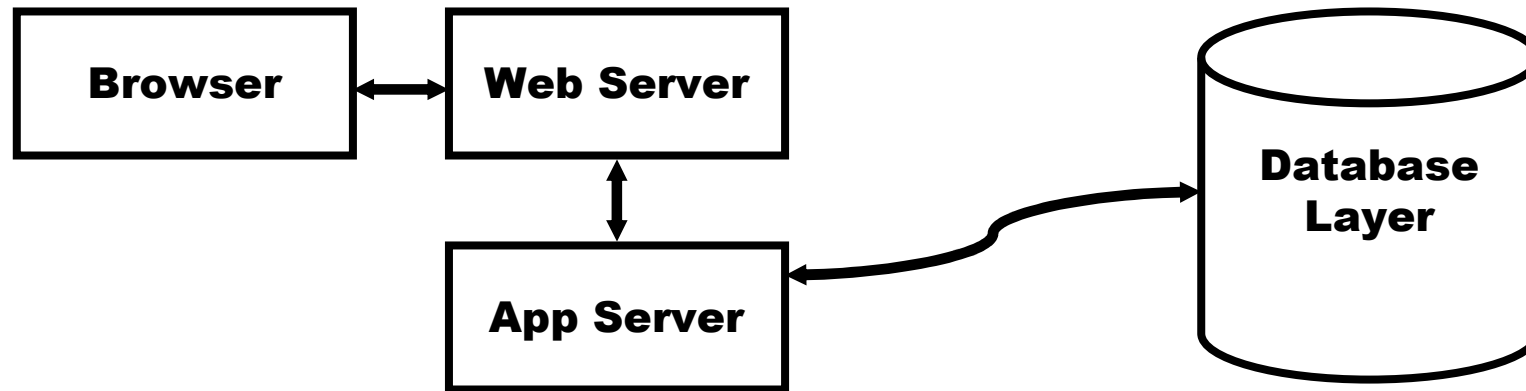
Guideword Heuristics for Diagram Analysis

- **Boxes**
  - *Missing/Drop-out*
  - *Extra/Interfering*
  - *Incorrect*
  - *Timing/Sequencing*
  - *Contents/Algorithms*
  - *Conditional behavior*
  - *Limitations*
  - *Error Handling*

- **Lines**
  - *Missing/Drop-out*
  - *Extra/Forking*
  - *Incorrect*
  - *Timing/Sequencing*
  - *Status Communication*
  - *Data Structures*

- **Paths**
  - *Simplest*
  - *Popular*
  - *Critical*
  - *Complex*
  - *Pathological*
  - *Challenging*
  - *Error Handling*
  - *Periodic*

Browser ↔ **Web Server**

Testability!

App Server

Database Layer

Visualizing Test Coverage: *Annotation*

Browser ↔ **Web Server**	**Database Layer**
↕	
App Server	

🔍 **Observation or Oracle**

🔨 **Tool Focus**

📱 **Control/Vary/Modify**

📄 **Data or Object**

⚡ **Force Fail**

🔄 **Activity**

Visualizing Test Coverge: Annotation

Performance history

Performance data

Inspect reports

Build history oracle

Survey

Server stress

Build stressbots

History oracle

Browser

Man-in-middle

Web Server

Build
Error Monitor

Data generator

Datagen Oracle

Force fail

Force fail

**Database
Layer**

Build
Error Monitor

Review Error
Output

App Server

Table consistency oracle

Coverage analysis

History oracle
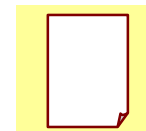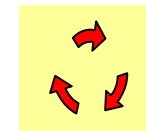
Beware Visual Bias!

Browser ← → W...

↯	setup
▦	browser type & version
⚡	cookies
▦	security settings
▦	screen size
🔍	review client-side scripts & applets
🔍	usability
↯	specific functions

Things that don't appear on the diagram are easy to forget.

Visualizing Test Progress
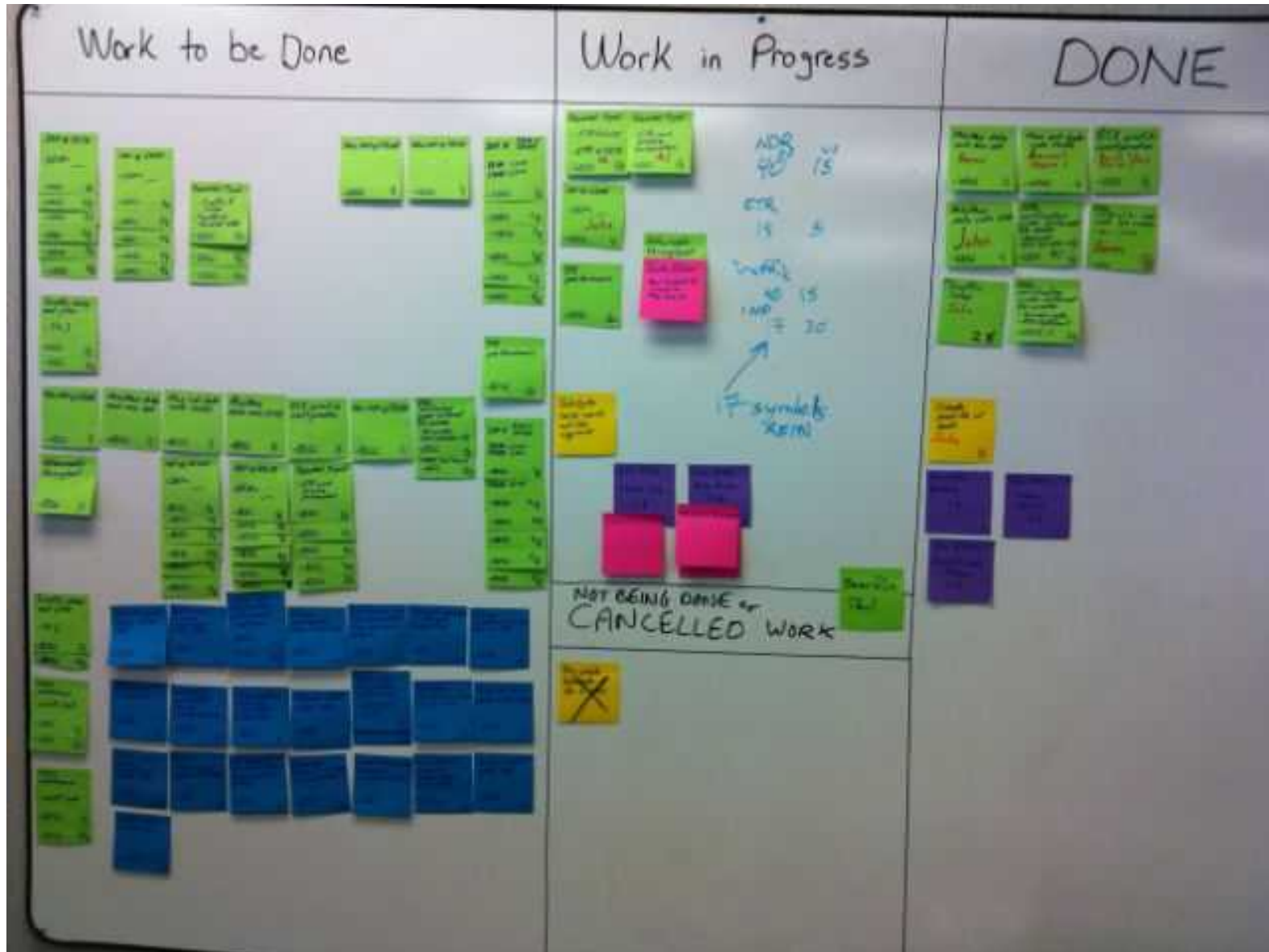
Visualizing Test Progress

Visualizing Test Progress
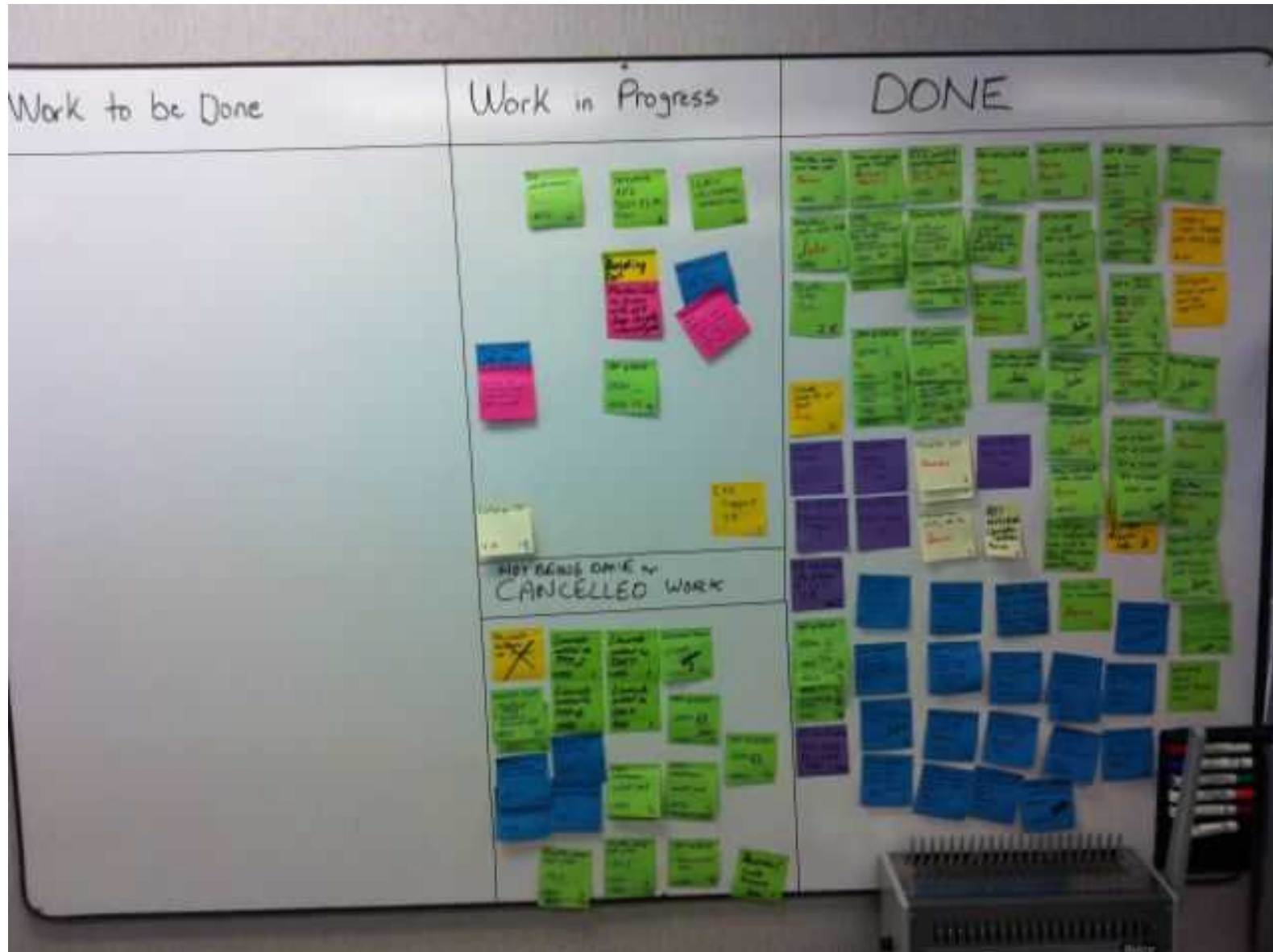
The Low-Tech Testing Dashboard

Testing Dashboard				Updated 21/2 Build 38
Area	**Effort**	**C**	**Q**	**Comments**
File/edit	High	1	🙂	
View	Low	1+	😐	1345, 1363, 1401
Insert	Low	2	🙂	
Format	Low	2+	😐	automation broken
Tools	Blocked	1	☹️	crashes bug 1407, 1423
Slideshow	Low	2	☹️	animation memory leak
Online help	Blocked	0		new files not delivered
Clip art	Pause	1	😐	need help to test
Connectors	None	1	😐	need help to test
Install	Start 20/3	0		
Compatibility	Start 13/3	0		compatibility lab time scheduled
General GUI	Low	3	🙂	

Product Area
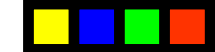
Area
file/edit
view
insert
format
tools
slideshow
online help
clipart
converters
install
compatibility
general GUI

- 15-30 areas (keep it simple)
- Avoid sub-areas: they're confusing.
- Areas should have roughly equal value.
- Areas together should be inclusive of everything reasonably testable.
- "Product areas" can include tasks or risks- but put them at the end.
- Minimize overlap between areas.
- Areas must "make sense" to your clients, or they won't use the board.

Test Effort

How much testing focus each area is getting right now?

None	Not testing; not planning to test.
Start	No testing yet, but expecting to start soon.
Low	Regression or spot testing only; maintaining coverage.
High	Focused testing effort; increasing coverage.
Pause	Temporarily ceased testing, though area is testable.
Blocked	Can't effectively test, due to blocking problem.
Ship	Going through final tests and wrap-up procedure.

Test Coverage

How much information do we have about each area so far?

0	We don't have good information about this area.
1	**Sanity Check:** major functions & simple data.
1+	More than sanity, but many functions not tested.
2	**Common & Critical:** all functions touched; common & critical tests executed.
2+	Some data, state, or error coverage beyond level 2.
3	**Complex Cases:** strong data, state, error, or stress testing.

Quality Assessment

Does management see threats to the ship date?

🙂 (green smiley face)	"We know of no problems in this area that threaten to stop ship or interrupt testing, nor do we have any definite suspicions about any."
😐 (neutral face)	"We know of problems that are possible showstoppers, or we suspect that there could be important problems not yet discovered."
☹️ (red frowning face)	"We know of problems in this area that definitely stop ship or interrupt testing."
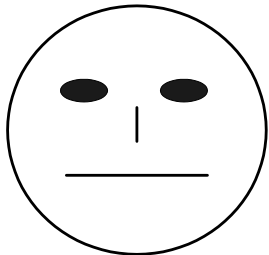
Why Is Testing Taking So Long?

- Try 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, time-tracking, waiting, 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 to estimate" in their estimates?

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
  - this is a highly arbitrary and artificial assumption—that is, it's *wrong*, but we use it to model an issue and make a point
- Suppose also that it takes eight extra minutes to investigate and report a bug
  - another stupid, sweeping generalization in service of the point
- In a 90-minute session, we can run 45 feature tests—*as long as we don't find any bugs*

How Do We Spend Time?
(assuming all tests below are *good* tests)

Module	Bug reporting/investigation (time spent on tests that find bugs)	Test design and execution (time spent on tests that find no bugs)	Number of tests
A (good)	0 minutes (no bugs found)	90 minutes (45 tests)	45
B (okay)	10 minutes (1 bug, 1 test)	80 minutes (40 tests)	41
C (bad)	80 minutes (8 bugs, 8 tests)	10 minutes (5 tests)	13

Investigating and reporting bugs means….

SLOWER TESTING or…

REDUCED COVERAGE …or both.

• 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)

Fix verifications	Bug reporting and investigation today	Test design and execution today	New tests today	Total over two days
0 min	0	45	45	90
6 min	10 min (1 new bug)	74 min (37 tests)	38	79
48 min	40 min (4 new bugs)	2 min (1 test)	5	18

Finding bugs today means….
VERIFYING FIXES LATER
…which means….

EVEN SLOWER TESTING or…
EVEN LESS COVERAGE …or both.

• …and note the optimistic assumption that all of our fixed verifications worked, and that we found no new bugs while running them. Has this ever happened for you?

Testing vs. Investigation

- Note that I just gave you a compelling-looking table, using simple measures, but notice that we still 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
  - luck

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

Principles for Data Presentation from Tufte

- The point of display is to aid thinking

  - What is the thinking task?

- Data is about relationships; show comparisons

- Where causality is known, show causality

  - where causality isn't know, seek patterns

- Show more than one variable and changes in the variables

  - which may expose previously unnoticed patterns

- Integrate word, number and image

  - "don't make a report replicate the chaotic mode of production"

from notes taken at Tufte's one-day class, Dearborn, MI, 2004

Principles for Data Presentation from Tufte

- Documentation quality shapes credibility
  - make the source of the data part of the display
- Show as many (or as long) intervals adjacent in space
  - not stacked in time
- Use as large a display as possible
  - large paper, big monitors, multiple monitors
- Use small multiples
  - use the same scale consistently

from notes taken at Tufte's one-day class, Dearborn, MI, 2004

Principles for Data Presentation from Tufte

- At least six dimensions available: height, width, length, time, revealed/concealed, and colour

- More data horizontally provides the usual basis for comparison

- Design of information usually recapitulates

  - the corporate hierarchy

  - computer systems

- Visual metaphors should be based on the content

from notes taken at Tufte's one-day class, Dearborn, MI, 2004

Measurement Skills

What measurement skills
can we list now?

Framing The Test 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

Focusing

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?

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

- what you might be missing, and why it's okay

Defocusing

Try Rubrics and Checklists

- Identify a diversified set of factors
- Create checklists of observable behaviours and outcomes
- Use multidimensional tables for comparison
- Qualitative evaluation may be okay
- Subjective evaluation is okay too, but…
  - if it's a concern, obtain multiple opinions from multiple sources; and
  - consider the differing values and observational modes of multiple constituencies

Other Modes of Assessment

- Try temperature readings; ask people for
    - 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

Try Other Modes of Assessment

- Try private chats, standup meetings, or scrums
  - short meetings that identify needs for further meetings between smaller groups, to lessen wasted time
- Try laddering exercises
  - ranking, rather than measuring
  - if they're influenced by feelings, that may be OK
  - emotions are signals; look into them
- Try *talking to people*
  - try asking for descriptions instead of numbers
  - try retrospectives

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 or surrogate measures

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.

For a valid *predictive* software development metric, you need a stable team and a stable project.

Readings

- *Quality Software Management, Vol. 2., "First Order Measurement"*
- *Exploring Requirements: Quality Before Design*
  - Gerald M. Weinberg
- *"Software Engineering Metrics: What Do They Measure and How Do We Know?"*
  - Cem Kaner and Walter P. Bond
  - online at http://www.kaner.com/pdfs/metrics2004.pdf
- *How to Lie with Statistics*
  - Darrell Huff
- *The Black Swan*
- *Fooled by Randomness*
  - Nassim Nicholas Taleb

Readings

- *Experimental and Quasi-Experimental Designs for Generalized Causal Inference*
  - Shadish, Cook, Campbell
- *Reliability and Validity in Qualitative Research*
  - Kirk & Miller
- *Tools of Critical Thinking: Metathoughts for Psychology*
  - David Levy
- "From Neurons to Neighborhoods: The Science of Early Childhood Development"
  - Shonkoff and Phillips, http://www.nap.edu/openbook.php?isbn=0309069882
- *How to Measure Anything: Finding the Value of "Intangibles" in Business*
  - Douglas Hubbard

Readings

- *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

Quick Summary

- Measurement: of the talks, books, and articles on metrics you find, what proportion fail to mention validity?

- Measurement: in discussions on measurement, what proportion fail to distinguish between "measurement" and "metric"?

- Try measuring this: in presentations on measurement, what proportion say "It's not easy" without providing any how-tos?

- *Measurement is the art and science of making reliable observations*.

- A metric is a measurement function that relates an observation with a number. --Cem Kaner

  - A number *or a value*.

- Construct validity involves questioning the function relating an observation and a number. Thus construct validity also includes questioning our observations and our classifications.