How to Get What You Really Want from Testing (Managers' Edition)

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

James Bach
Satisfice
http://www.satisfice.com
@jamesmarcusbach
james@satisfice.com

How to Get What You Really Want from Testing (Managers' Edition)

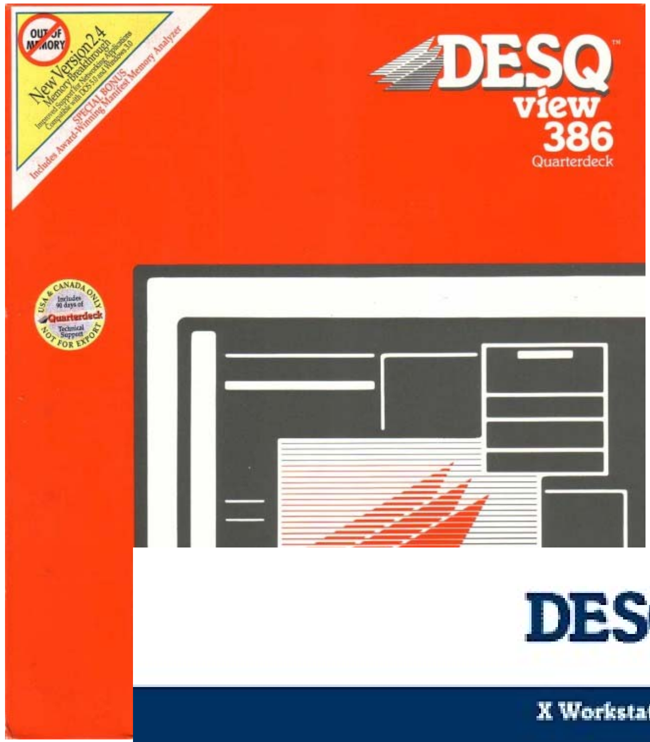
Michael Bolton
DevelopSense
http://www.developsense.com
@michaelbolton
michael@developsense.com

James Bach
Satisfice
http://www.satisfice.com
@jamesmarcusbach
james@satisfice.com

Notes and Updates

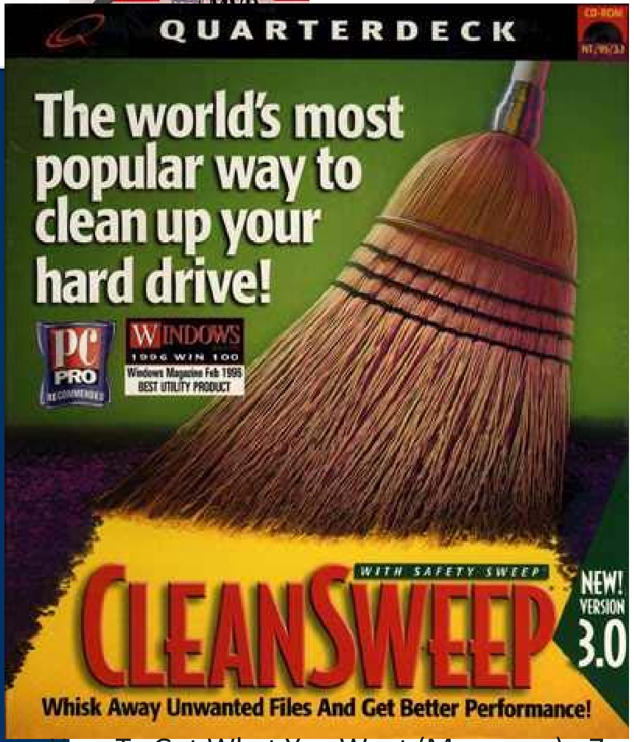
- This presentation is ALWAYS under construction
- Updated slides at http://www.developsense.com/past.html
- All material comes with lifetime free technical support

I’m Michael Bolton


DESQ view 386
Quarterdeck
New Version 2.4
OUT OF MEMORY

Quarterdeck expanded memory manager 386
New Version 6.0
OUT OF MEMORY
MEMORY

DESQview/X
Quarterdeck
X Workstation Software for DOS PCs
Version 2
Cross-platform integration from Quarterdeck
(C) 1985-1994 Quarterdeck Office Systems, Inc. U.S. Patent No. 4,823,108.
For other proprietary notices see the DESQview/X Tune Performance Menu.


QUARTERDECK
The world's most popular way to clean up your hard drive!
PC PRO
WINDOWS 1996 WIN 100
Windows Magazine Feb 1996 BEST UTILITY PRODUCT
WITH SAFETY SWEEP
CLEANSWEEP
NEW! VERSION 3.0
Whisk Away Unwanted Files And Get Better Performance!

What Do I Do?

- I help people to solve testing problems they didn't know they could solve, and I teach them how they can do that themselves.
- I teach Rapid Software Testing
  - http://www.developsense.com
- I'm focused on advancing the craft of software testing, and its value to organizations
- And I need *your* help!

RAPID SOFTWARE TESTING

RST

A CONTEXT-DRIVEN METHODOLOGY

everybody, in my opinion, should

I think How to Get What ~~You~~ Really Want from Testing (Managers' Edition)

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

James Bach
Satisfice
http://www.satisfice.com
@jamesmarcusbach
james@satisfice.com

What are YOU doing?

Directing people

Acquiring resources

Applying judgement

Making decisions

Gathering information

Massaging egos

Weighing opinions

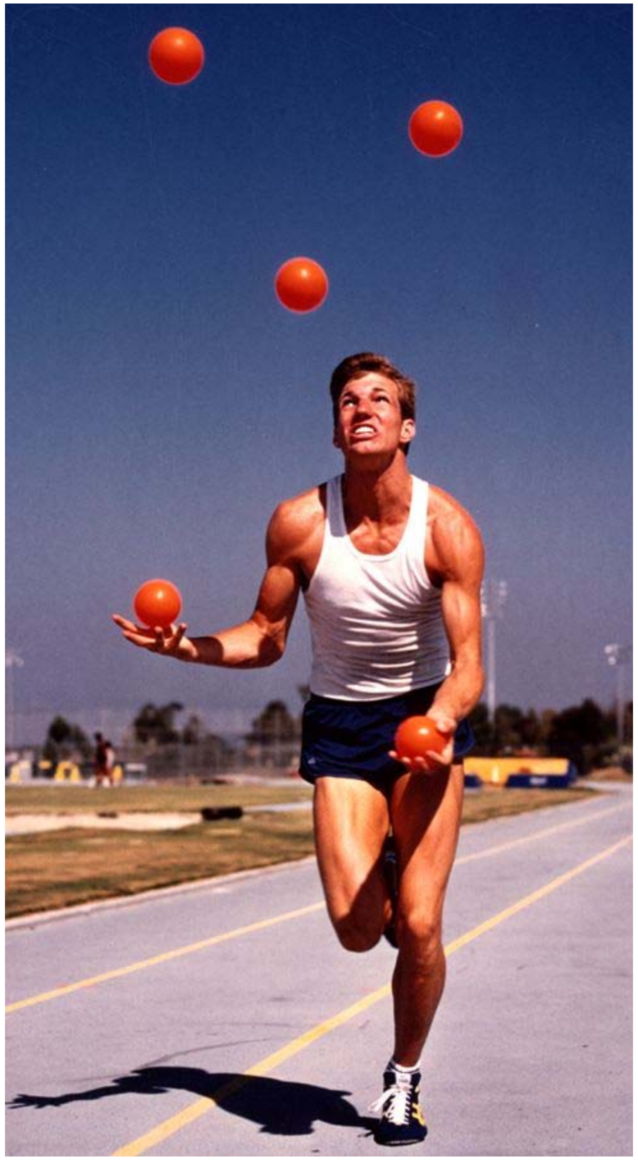
Balancing budgets

Co-ordinating work

Dealing with emotions

Handling problems

Delegating



A Key Question

Are there problems
that threaten
the on-time
successful
completion of the project*?

* At any level of granularity

What Testing IS

- “Gathering information in order to inform a decision” (Weinberg)
- “An empirical, technical investigation of software, done on behalf of stakeholders, with the intention of revealing quality-related information of the kind that they seek” (Kaner)
- Applied critical thinking… “thinking about thinking with the intention of avoiding being fooled.” (Bach and Bolton)
- (but there’s more, later…)

What Testing IS NOT

Confirmation

Demonstration

“Breaking the product”

Testing is not Quality Assurance

"QA" = Questions + Answers

- Testing does not *assure quality*
  - *YOU, dear Project Manager, do that!*
- Testing does not *improve* quality
  - *unless someone changes something,*
    quality stays the same
- Testing *informs decisions* about quality
- Testing *raises questions*
  - "Is there a problem here?"
  - "Is everyone OK with this?"
- Testing *gets answers*
  - but not *complete* answers
  - "partial answers that might be useful" (Cem Kaner)

Testing is not test cases!

Piloting Cases

Parenting Cases

Research Cases

Management Cases

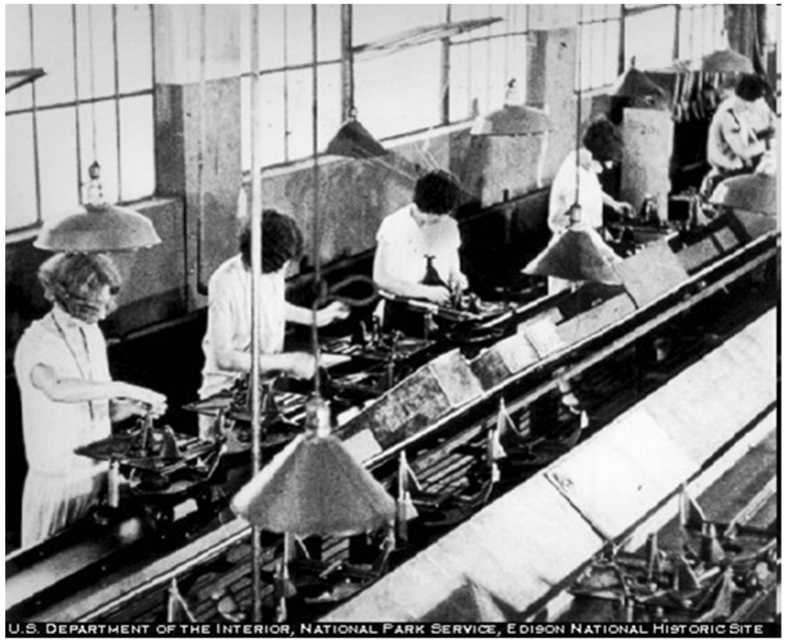
Then why do testers use test cases?

Even worse… why COUNT them?

One Answer

- Testers keep talking about test cases because managers keep asking about them.
    - because testers keep talking about them
        - because managers keep asking about them
            - because testers keep talking about them
                - because managers keep asking about them
                    - because testers keep talking about them
                        - because managers keep asking about them
                            - because testers keep talking about them
                                - because managers keep asking about them
                                    - because testers keep talking about them
                                        - because managers keep asking about them
- Some managers like test cases because they make testing “legible”—readable
    - discrete outcomes; “pass or fail”
    - units of production; widgets

Testing is not factory work!

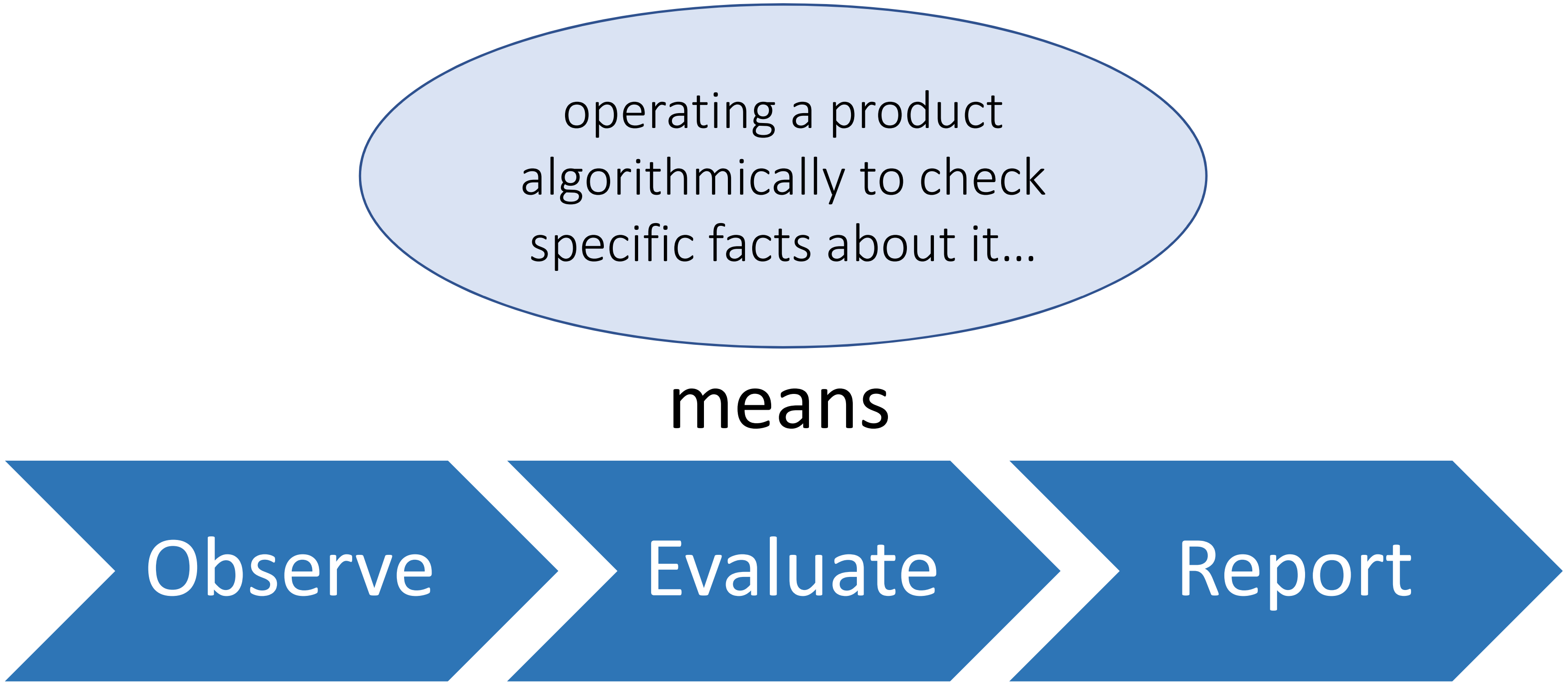

U.S. DEPARTMENT OF THE INTERIOR, NATIONAL PARK SERVICE, EDISON NATIONAL HISTORIC SITE

Testing is not factory work

Testing is *design* work

Design work is neither "manual" nor "automated" —and testing isn't either.

Call this “Checking” not Testing



operating a product algorithmically to check specific facts about it…
means
Observe
Evaluate
Report

Interact with the product in specific, *algorithmic* ways to collect specific observations.

Apply *algorithmic* decision rules to those observations.

Report any failed checks *algorithmically*.

A check can be performed…

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

by a human who has
been instructed *not to*
think (and who is slow
and variable)

Tools Can Do More Than Checking, Too!

restore system parameters

set systems to a particular state

create containers

check builds

calculate statistics

simulate sub-systems

perturb test data

monitor live systems

raise alerts

provide a parallel algorithm

vary environmental conditions

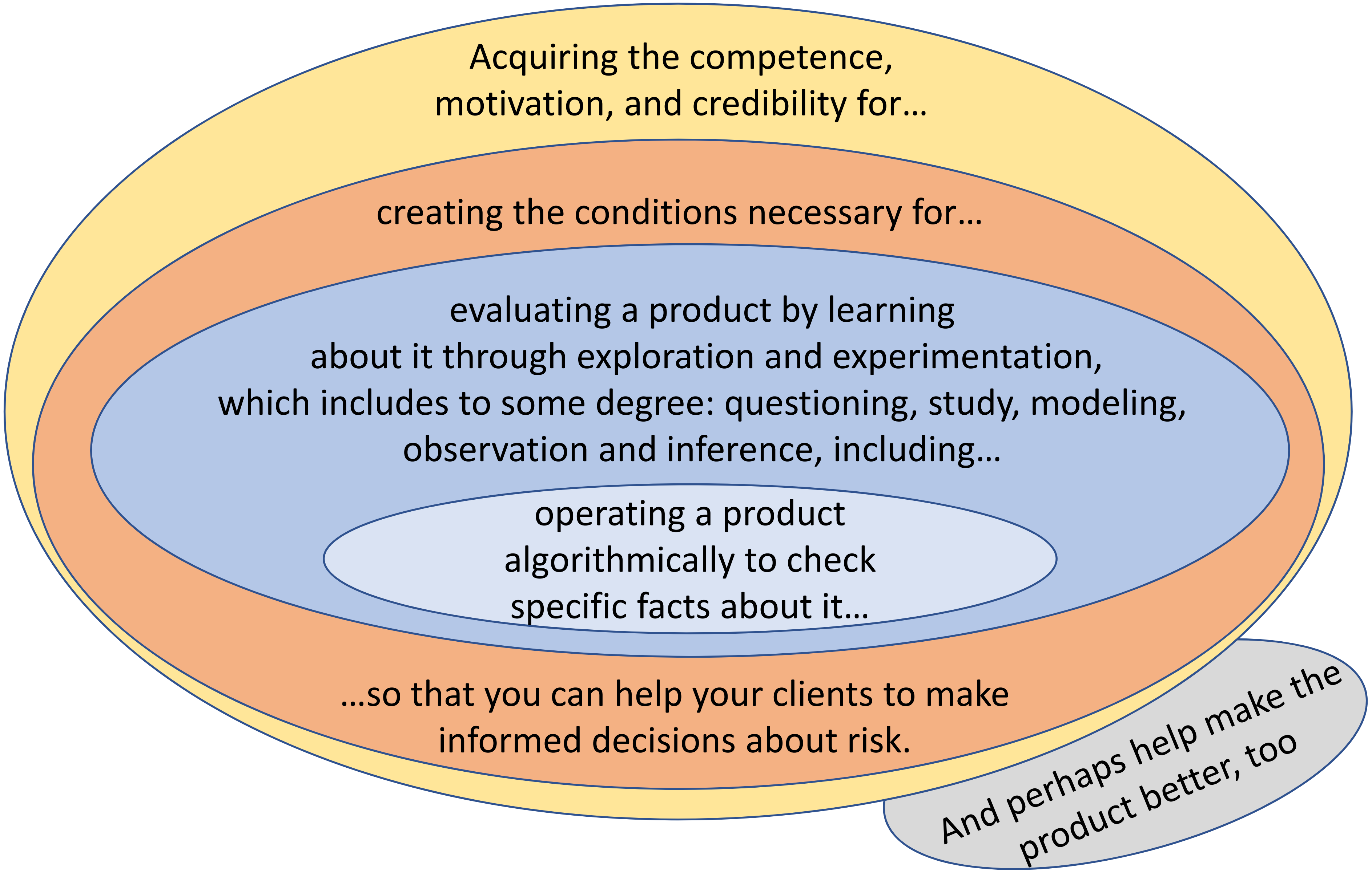
track results

Testing Is *More Than Checking*

- *Checking* is okay, but it's mostly focused on *verification*; confirming what we know or hope to be true.
- To escape problems with verification, we must do more than checking; we must *test*.

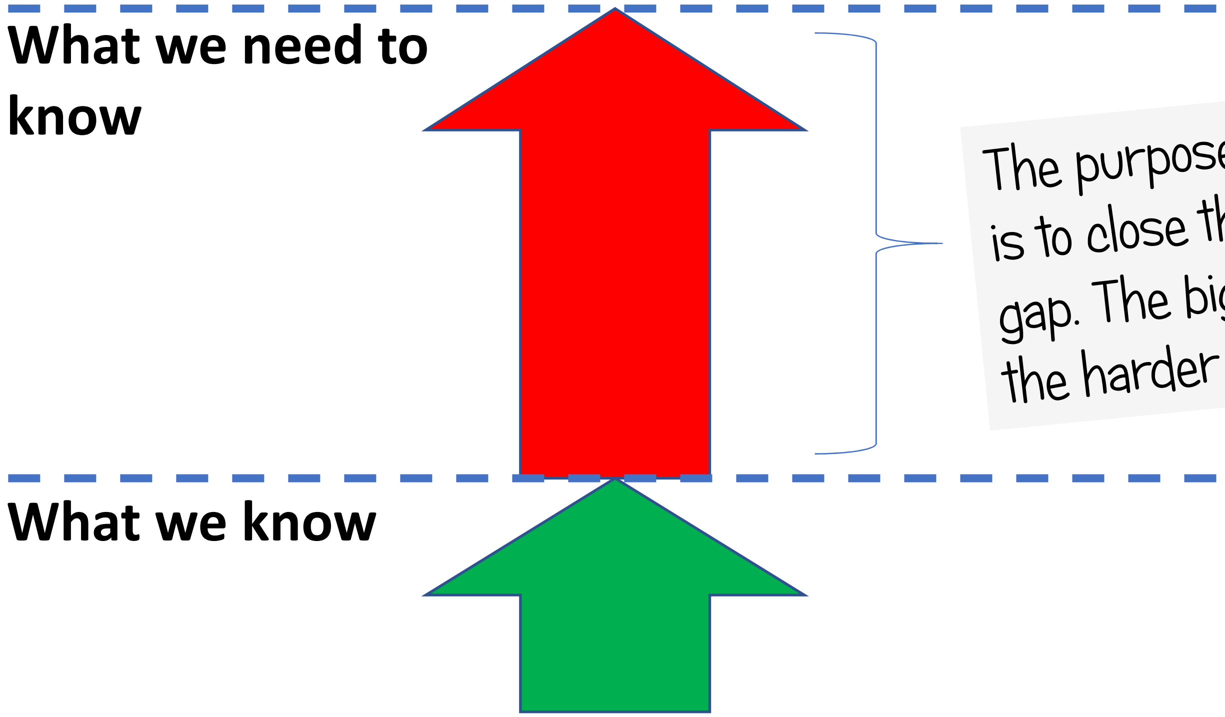
See http://www.developsense.com/2009/08/testing-vs-checking.html

Testing Is *Learning About a Product*


Acquiring the competence, motivation, and credibility for…
creating the conditions necessary for…
evaluating a product by learning about it through exploration and experimentation, which includes to some degree: questioning, study, modeling, observation and inference, including…
operating a product algorithmically to check specific facts about it…
…so that you can help your clients to make informed decisions about risk.
And perhaps help make the product better, too

Testing is *Investigating and Exploring Risk*

Problems that threaten the value of the product



The purpose of testing is to close the risk gap. The bigger this is, the harder it is to test.

Our knowledge of the status of the product

Four Kinds of Risk Drivers

- problems in the *product*

Bugs

- *unawareness* of problems in the product
- problems in the *project*

Issues

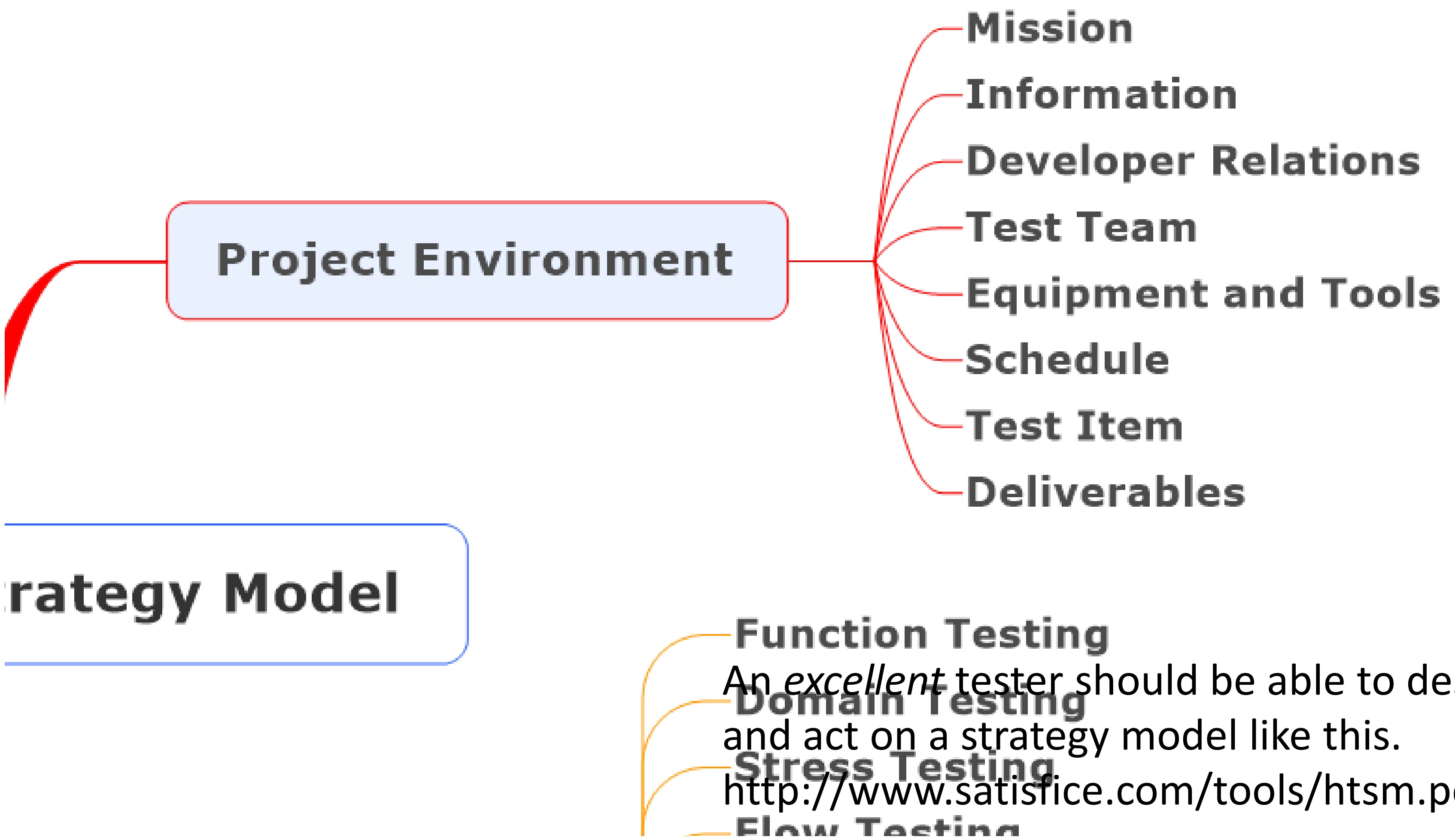
- *unawareness* of problems in the project

Risk (n.) Some person(s) will experience a problem with respect to something desirable that can be detected in some set of conditions because of a vulnerability in the system.

Risk Story Elements

- Some PERSON(S) Stakeholders
  - user, customer, developer, tester, businessperson, bystander…
  - (a group, a business, a community, society at large…)
- will EXPERIENCE Context
  - be affected, in the context of an event or situation, at least once by …
- a PROBLEM Problem
  - that leads to bad feelings (annoyance, frustration, confusion), loss, harm, or diminished value…
- with respect to SOMETHING DESIRABLE Quality Criteria
  - like capability, reliability, performance…
- that CAN BE DETECTED Oracles
  - by a feeling, principle, tool, comparison to a document or picture…
- in SOME SET OF CONDITIONS Test Conditions
  - perhaps always, perhaps only sometimes,…
- because of a VULNERABILITY Theory of Error
  - a bug, a missing feature, an inconsistency…
- in the SYSTEM Product Elements
  - some result, process, component, feature, environment…

Testing Includes *Developing Strategy*



An *excellent* tester should be able to describe and act on a strategy model like this.
http://www.satisfice.com/tools/htsm.pdf

Testing Includes *Telling Stories*

Bugs

A story about the status of the PRODUCT…

…about what it does, how it failed, and how it might fail...

…in ways that matter to the various clients.

Oracles

A story about HOW IT WAS TESTED…

…how testers operated and observed it…

…how testers recognized problems…

Coverage

…what testers have and have not tested yet…

…what testers won't test at all (unless the client objects).

A story about how GOOD that testing is/was…

…what made testing harder or slower…

Issues

…how testable (or not) the product is…

…the risks and costs of testing or not testing…

…what testers need and what they recommend.

Why Is Part 3 So Important?

- *Bugs* threaten the value of the product.
- Testers need to be able to find important problems (especially bugs) quickly, but…
- *Issues* (things that make testing harder or slower) give bugs more time and more opportunity to hide.
- Testers must recognize, identify, and resolve issues (and might need help with that)…
- …especially when those issues involve *testability*.
- Testability helps to pre-empt the question you don't want to ask and testers don't want to hear:

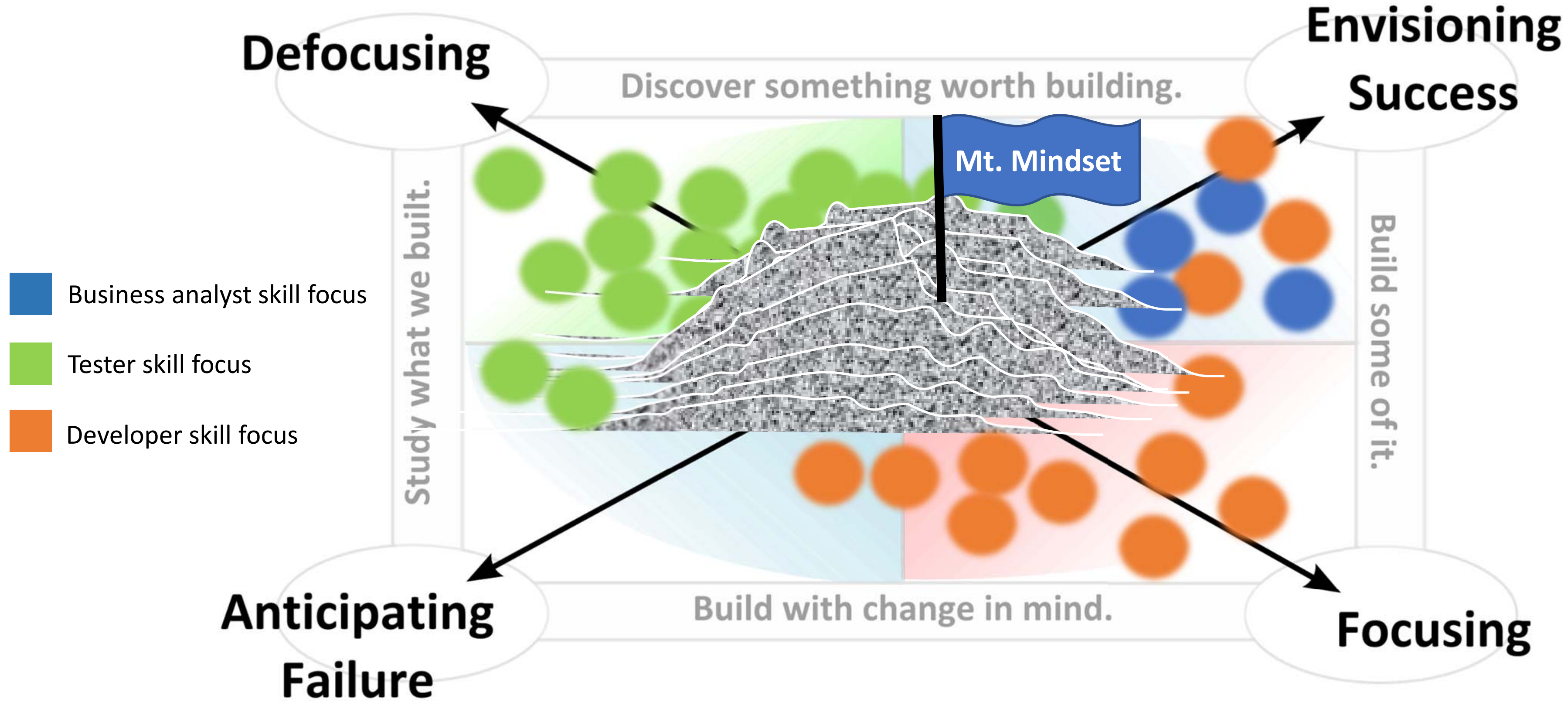
  "Why didn't you find that bug?"

Testing Includes *Advocating for Testability*

- Epistemic testability
  - What is the gap between what we know and what we need to know? Is there a real risk here?
- Value-related testability
  - What is the appropriate standard of correctness?
  - Should we be more concerned with risks other than this one?
- Intrinsic testability
  - Is there internal error detection and handling?
  - Does the product produce logs? Is there an API?
  - Is the product "unbuggy"?
- Subjective testability
  - Do we have a good understanding of the test space?
  - Do we have the skills we need to develop and apply useful tools?
- Project-related testability
  - Are we working in close collaboration with the programmers?
  - Could programmer checks and tests now be faster and cheaper than testing later?

Why Have Dedicated Testers?

A Central Obstacle Divides Development Work



🟦 Business analyst skill focus

🟩 Tester skill focus

🟧 Developer skill focus

NOTE: We do NOT claim that different kinds of work *must* be done by different people, or that the people *must* have different permanent roles. We DO claim that roles on a development team (collaborating with each other) provide a powerful heuristic for solving the mindset switching problem.

Organizing and Accounting for Testing Work

Consider focusing on *activities*, rather than artifacts. It's what people think and do that's important.

Try organizing testing in terms of *sessions* that produce coverage—experience with and knowledge about the product.

- Try granting testers freedom AND responsibility.
  - Require testers to keep credible, professional lab notes.
  - Help testers focus on a three-part testing story whether in a formal report, a debriefing, a morning standup, or at the water cooler.

Beware of excessive or premature formalization.

- Bugs don't follow formal processes or standards!
- Testing is about discovery and investigation, not demonstration.

Avoid reducing testing to test cases.

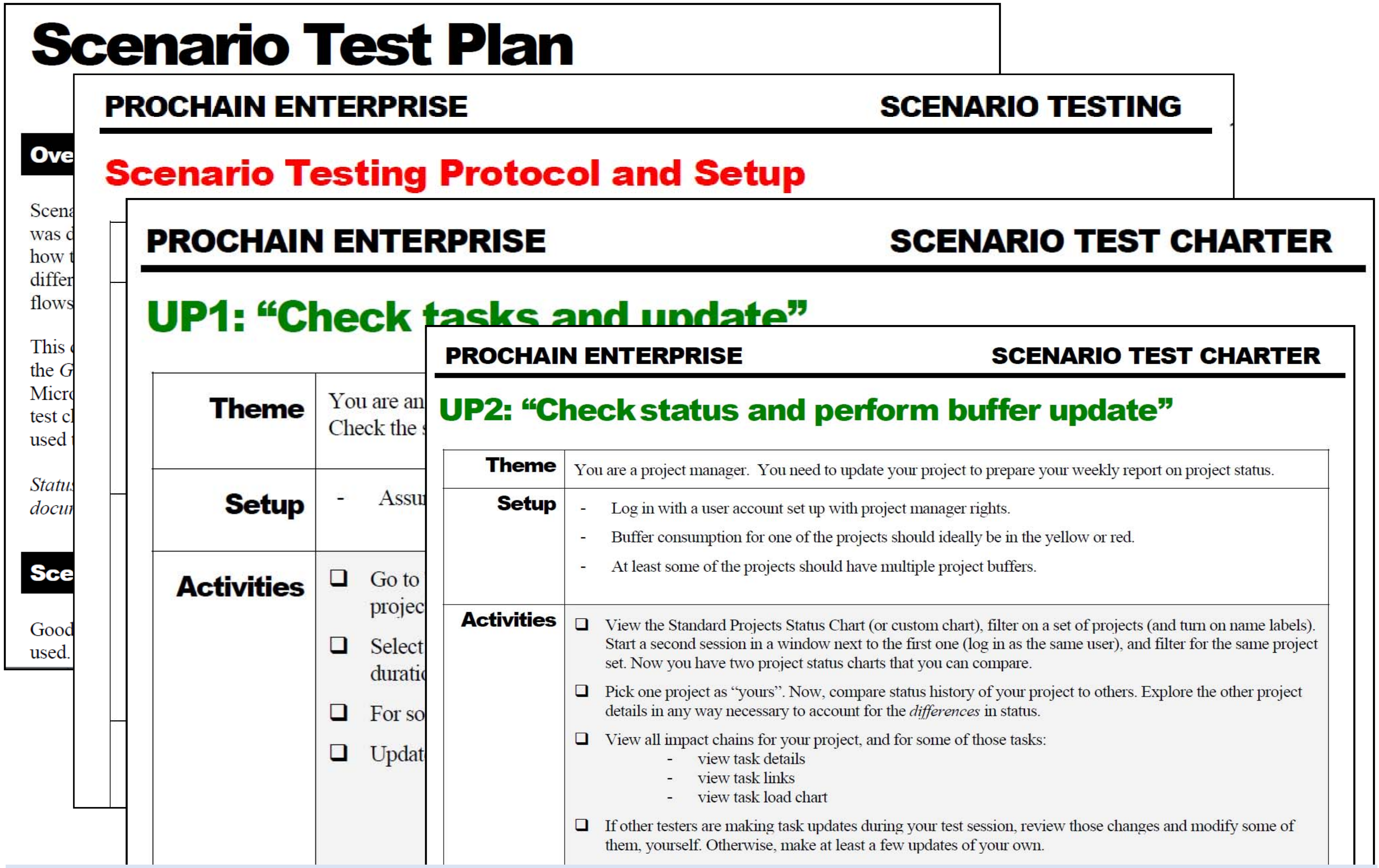
One Big Problem in Testing

Formality Bloat

- Typically, your testing doesn’t need to be too formal*
  - Even when your testing *does* need to be formal, you’ll need to do substantial amounts of informal testing in order figure out how to do *excellent* formal testing.
  - Who says? The FDA. See http://www.satisfice.com/blog/archives/602
- Even in a highly regulated environment, you do *formal* testing primarily for the auditors.
- You do *informal* testing to help identify if you’re going to lose money, blow things up, or kill people.
- “Testing shouldn’t be too formal… unless you want miss lots of bugs.” —James Bach

* *Formal testing means “testing that must be done in a specific way, or to investigate a specific fact.”*

Want Lightweight Formality? Try Scenarios.



Scenario Test Plan

Ove

Scena was d how t differ flows

This the *G* Micr test c used

Status docu

Sce

Good used.

PROCHAIN ENTERPRISE SCENARIO TESTING

Scenario Testing Protocol and Setup

PROCHAIN ENTERPRISE SCENARIO TEST CHARTER

UP1: "Check tasks and update"

Theme	You are an Check the
Setup	- Assu
Activities	❑ Go to projec ❑ Select duratic ❑ For so ❑ Updat

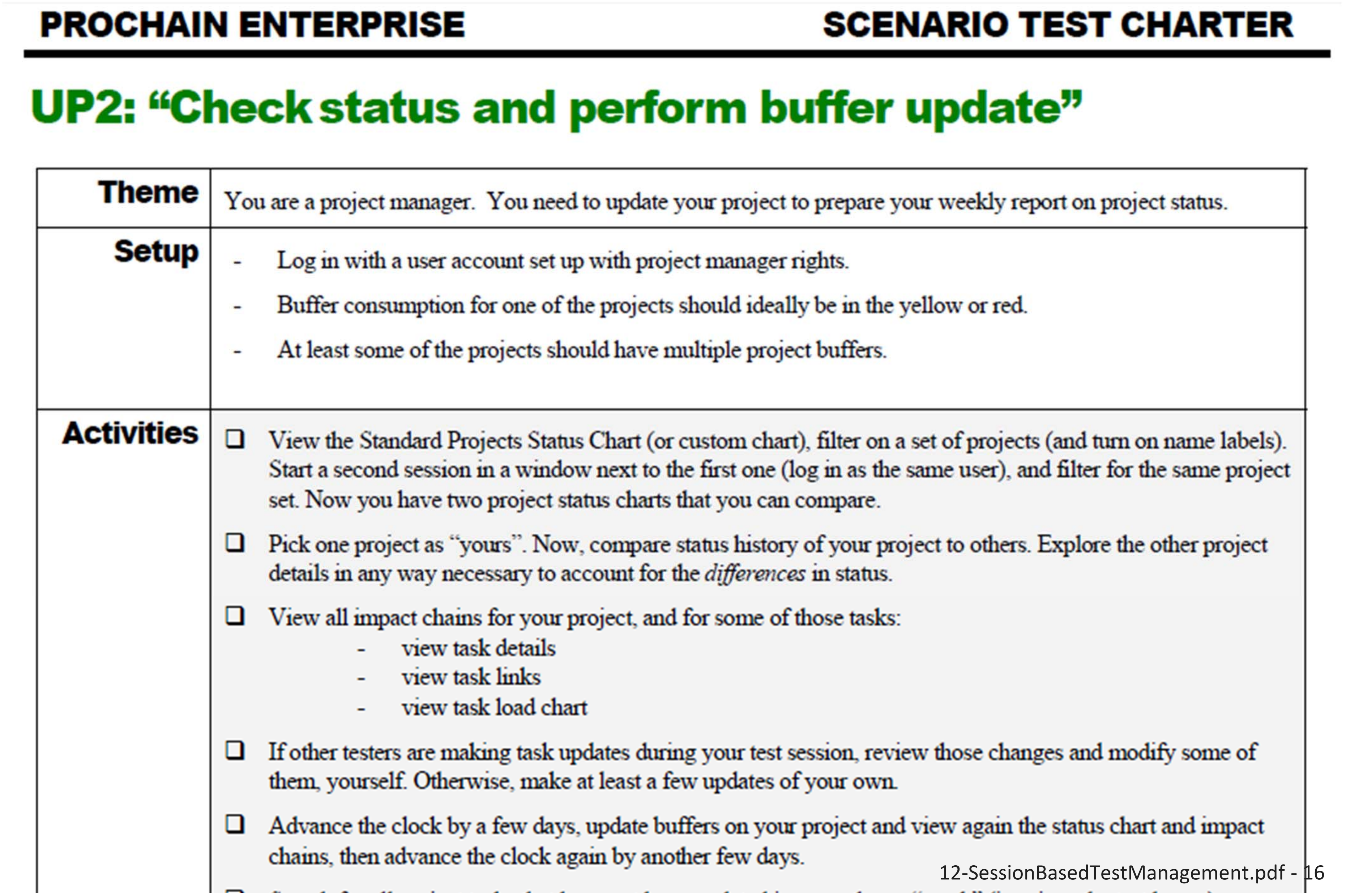
PROCHAIN ENTERPRISE SCENARIO TEST CHARTER

UP2: "Check status and perform buffer update"

Theme	You are a project manager. You need to update your project to prepare your weekly report on project status.
Setup	- Log in with a user account set up with project manager rights. - Buffer consumption for one of the projects should ideally be in the yellow or red. - At least some of the projects should have multiple project buffers.
Activities	❑ View the Standard Projects Status Chart (or custom chart), filter on a set of projects (and turn on name labels). Start a second session in a window next to the first one (log in as the same user), and filter for the same project set. Now you have two project status charts that you can compare. ❑ Pick one project as "yours". Now, compare status history of your project to others. Explore the other project details in any way necessary to account for the *differences* in status. ❑ View all impact chains for your project, and for some of those tasks: - view task details - view task links - view task load chart ❑ If other testers are making task updates during your test session, review those changes and modify some of them, yourself. Otherwise, make at least a few updates of your own.

See RST Appendices, "PCE Scenario Testing": http://www.satisfice.com/rst-appendices.pdf

(Optional) Formalize Scenario Charters

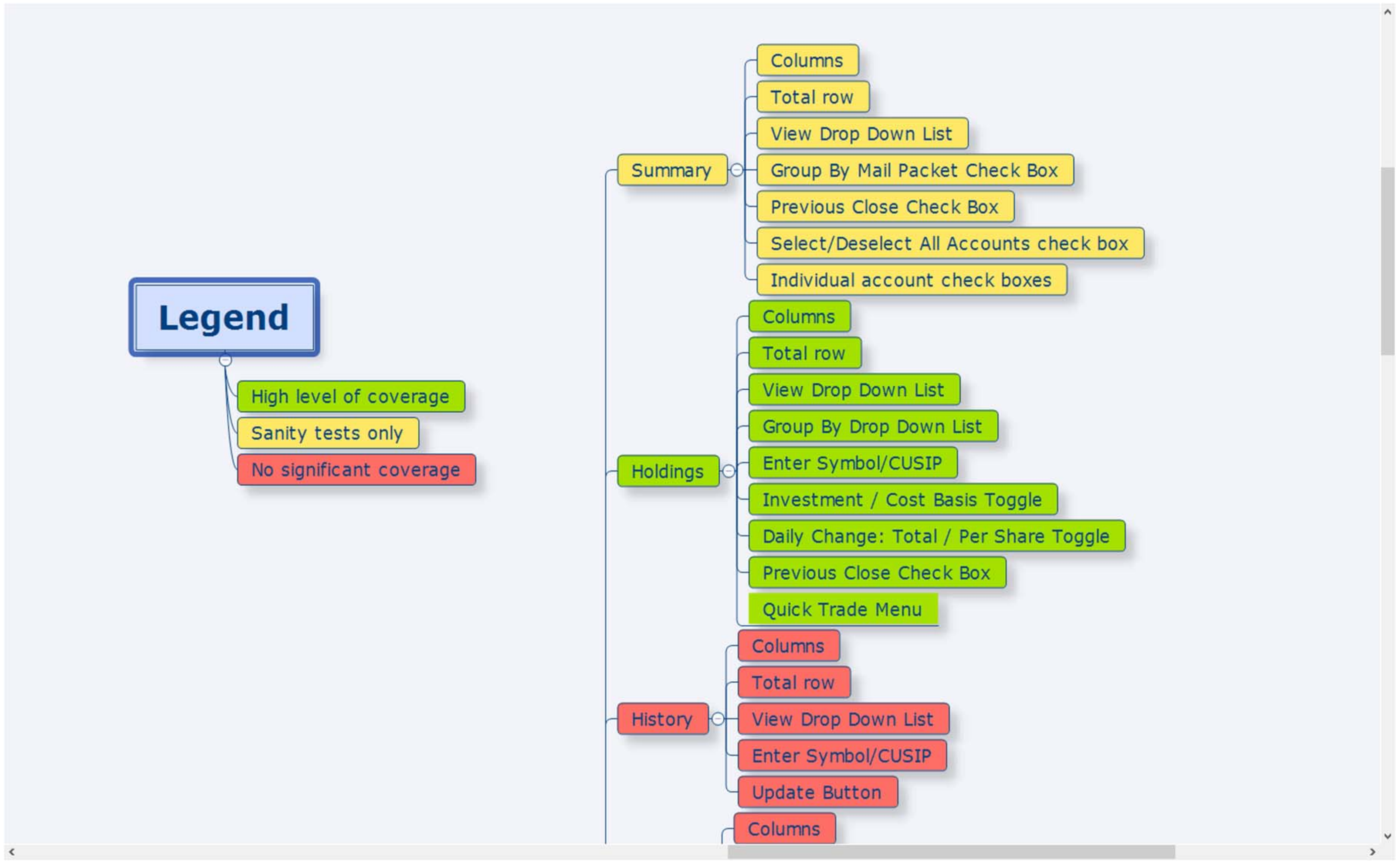


PROCHAIN ENTERPRISE **SCENARIO TEST CHARTER**

UP2: "Check status and perform buffer update"

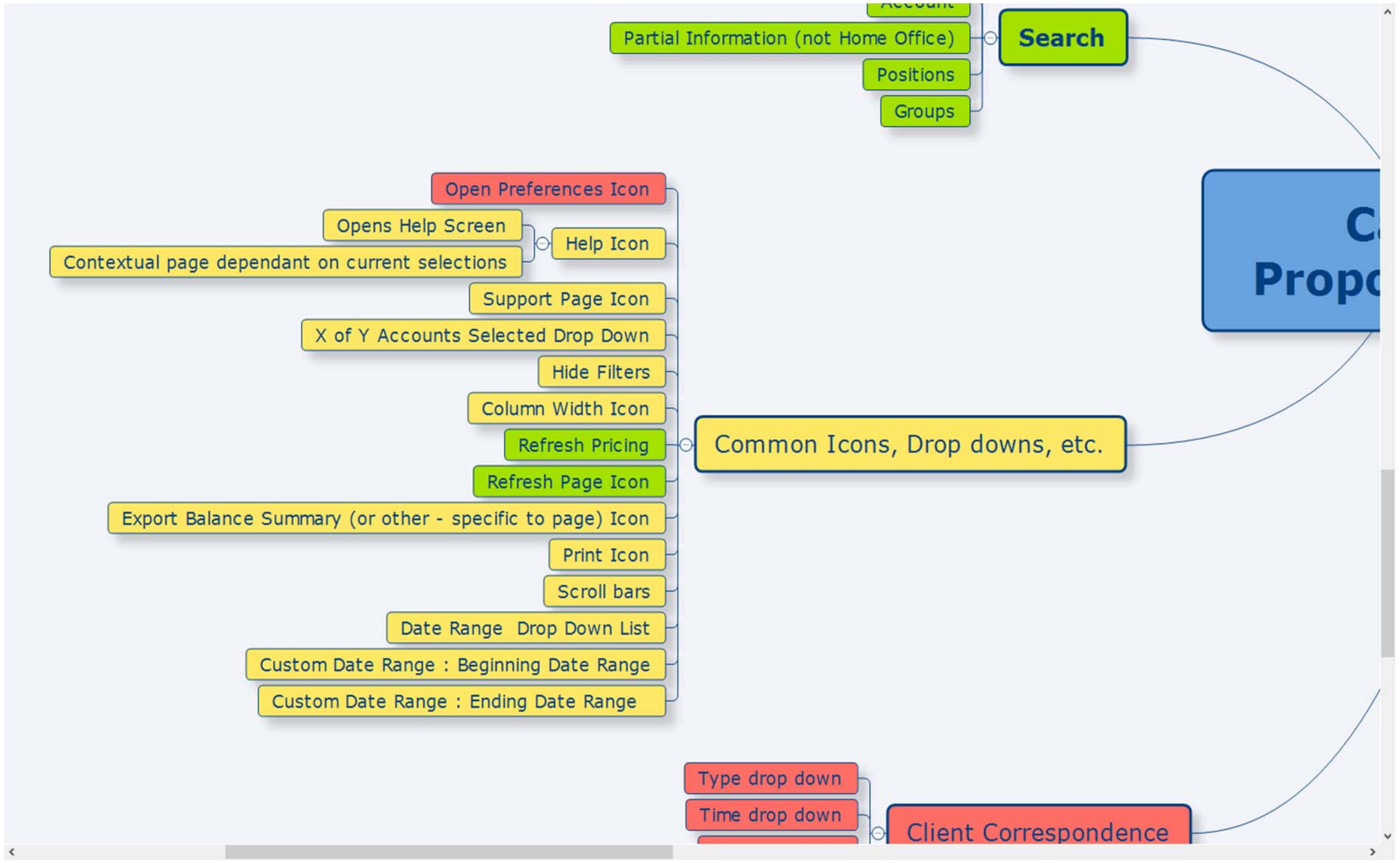
Theme	You are a project manager. You need to update your project to prepare your weekly report on project status.
Setup	- Log in with a user account set up with project manager rights. - Buffer consumption for one of the projects should ideally be in the yellow or red. - At least some of the projects should have multiple project buffers.
Activities	❑ View the Standard Projects Status Chart (or custom chart), filter on a set of projects (and turn on name labels). Start a second session in a window next to the first one (log in as the same user), and filter for the same project set. Now you have two project status charts that you can compare. ❑ Pick one project as "yours". Now, compare status history of your project to others. Explore the other project details in any way necessary to account for the *differences* in status. ❑ View all impact chains for your project, and for some of those tasks: - view task details - view task links - view task load chart ❑ If other testers are making task updates during your test session, review those changes and modify some of them, yourself. Otherwise, make at least a few updates of your own. ❑ Advance the clock by a few days, update buffers on your project and view again the status chart and impact chains, then advance the clock again by another few days.

Alternative to Test Cases: Coverage Maps
Feature Coverage



See "No Test Cases Required", https://www.youtube.com/watch?v=vN3E_jjbBpc

Alternative to Test Cases: Coverage Maps
Interface Coverage



Account
Partial Information (not Home Office)
Positions
Groups
Search
Open Preferences Icon
Opens Help Screen
Contextual page dependant on current selections
Help Icon
Support Page Icon
X of Y Accounts Selected Drop Down
Hide Filters
Column Width Icon
Refresh Pricing
Refresh Page Icon
Export Balance Summary (or other - specific to page) Icon
Print Icon
Scroll bars
Date Range Drop Down List
Custom Date Range : Beginning Date Range
Custom Date Range : Ending Date Range
Common Icons, Drop downs, etc.
C
Propo
Type drop down
Time drop down
Client Correspondence

See “No Test Cases Required”, https://www.youtube.com/watch?v=vN3E_jjbBpc

Product Status Reporting

Consider that bugs aren’t properties of a product, but a relationship between the product and some person(s).

Try asking for a list when you’re tempted to ask for a number.

- Try asking your testers about problems and risks that threaten the value of the product (instead of asking “Is everything OK?”).
- *Reward* problem reports; don’t punish them.

Beware bug and product reports that reduce information to data.

Avoid establishing a culture where people are reluctant to speak up. Don’t criminalize the reporting of problems!

Estimating Testing Effort

Consider that testing is not separate from development, but part of it.

- When on a journey from here to there, there's no "driving phase" followed by a "looking-out-the-windshield phase".

Try asking developers and testers as a group, not separately, if you need an estimate.

- If you've got a release date in mind, that's fine.

Beware of the belief that *developing* a product is the same as *assembling* one.

- The proportion of testing work to programming work can vary a lot.

Avoid treating estimates as commitments

- Development unfold in ways that are to some degree unpredictable.

Evaluating Testing Efficiency

Consider that bug investigation, reporting, and setup may be *interrupting* test coverage.

Try asking testers to account for these hidden aspects of testing work, and make them visible.

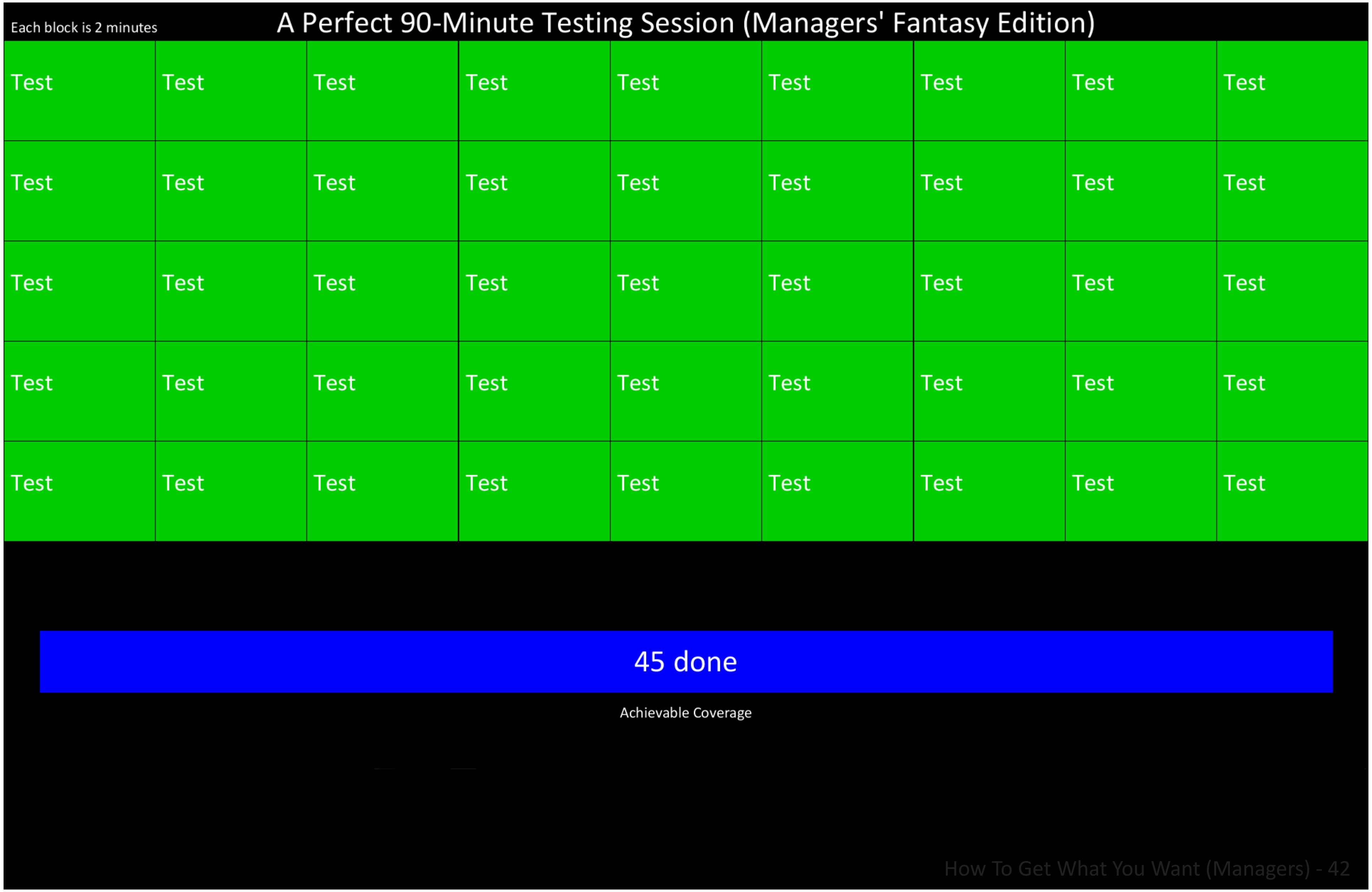
- Consider short experiments of detailed accounting or observation
- If you do gather test/bug/setup data, use it for *inquiry* rather than for *control*.

Beware "measurements" based on counting test cases or bugs.

- If they measure anything at all, "defect escape ratios" apply to project managers far more than they apply to testers.
- Too much scrutiny can rip the social fabric. Testers need freedom and responsibility to investigate and to stay engaged.

Avoid KPIs—reducing engineering to scorekeeping.

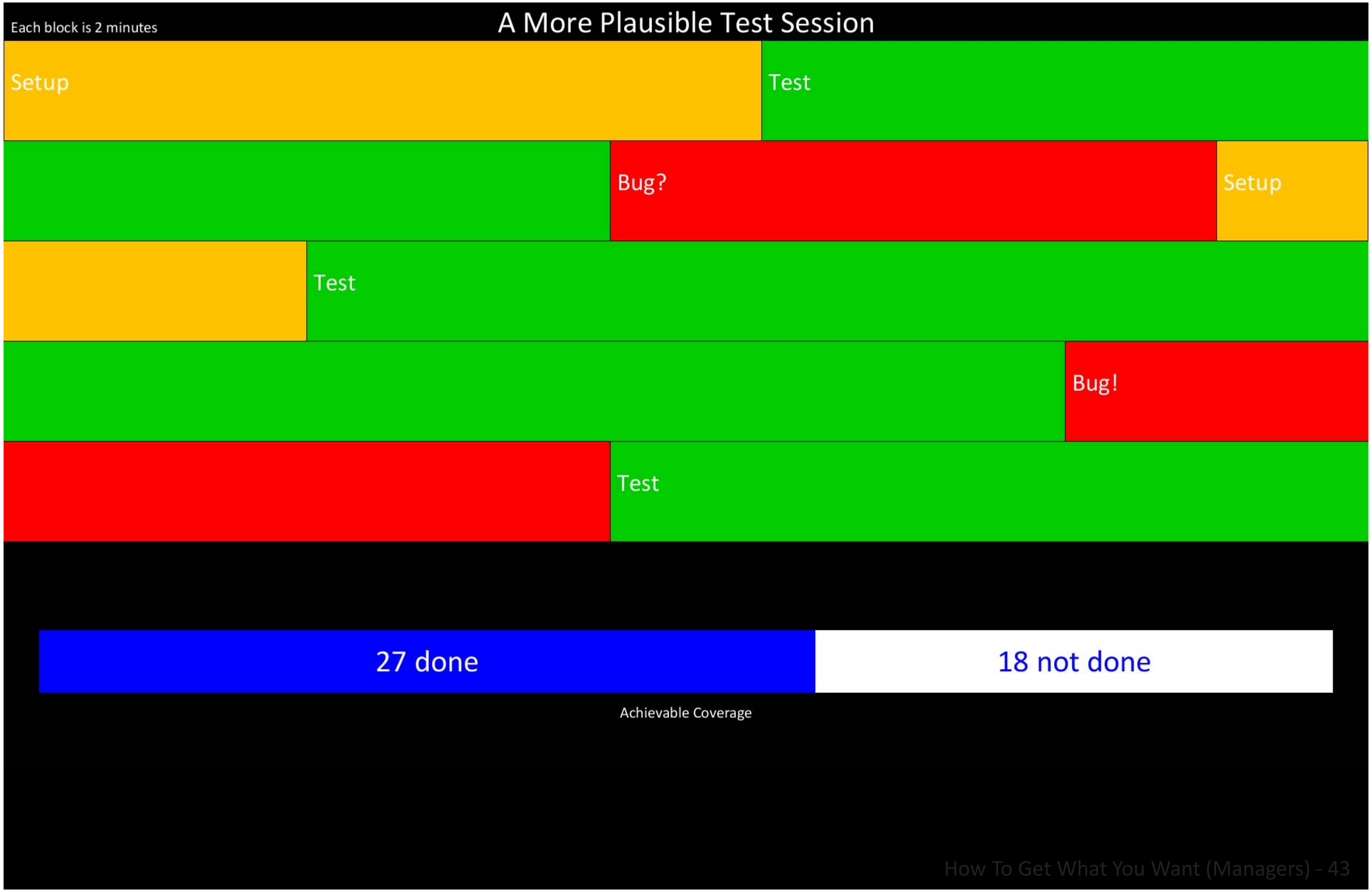
What Testing Sessions *Don't* Look Like


Each block is 2 minutes
A Perfect 90-Minute Testing Session (Managers' Fantasy Edition)

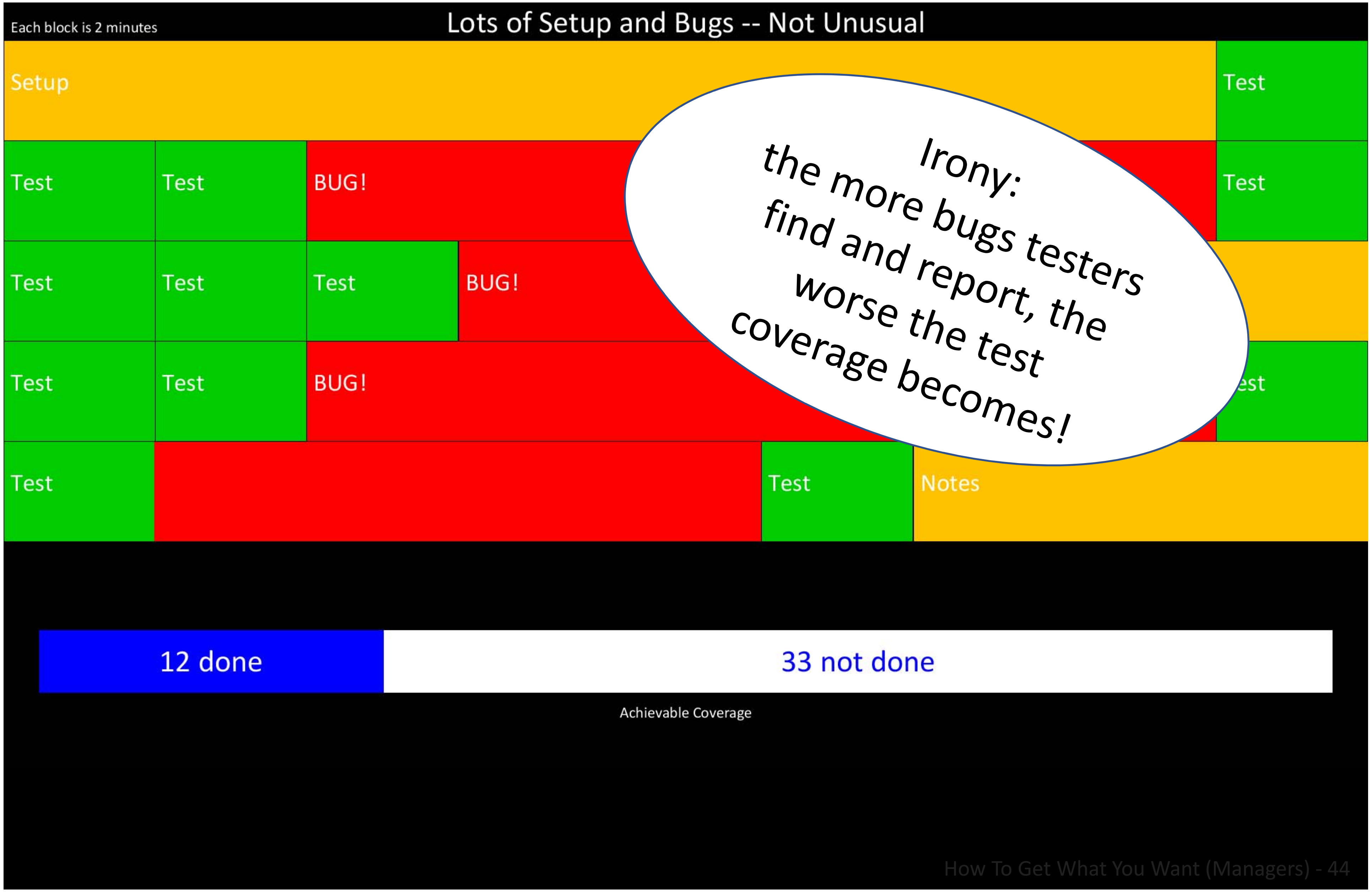
Test	Test	Test	Test	Test	Test	Test	Test	Test
Test	Test	Test	Test	Test	Test	Test	Test	Test
Test	Test	Test	Test	Test	Test	Test	Test	Test
Test	Test	Test	Test	Test	Test	Test	Test	Test
Test	Test	Test	Test	Test	Test	Test	Test	Test

45 done
Achievable Coverage

What Testing Sessions *Might* Look Like



What Testing Sessions *Often* Look Like



Automation and Tools

Consider the power of tools for far more than automated checking.

Try asking "how could tools help extend, enhance, accelerate, intensify testing work?"

Beware testers (or others) who overfocus on automated checking.

Avoid thinking that you can "automate the testing".

- Machines don't evaluate, learn, experiment, explore, model, study, question,
- You can't automate the management, either!

Avoid allowing tools to overstructure the testing work.

Not-So-Good Questions for Testers

- Is the product done?
- Are we ready to ship?
- Is it good enough?
  - All three of these are *your* decision, Dear Project Manager.
- How much time do you need to test?
  - This is like asking "How much time do you need to learn about the product?"
- How many tests cases have you run?
- How many test cases are passing and failing?
- How many bugs are in the product?
  - These numbers don't mean anything without a story — and once you have the story, the numbers probably aren't important.

Better Questions for Testers

- What is the product story? What can you tell me about important problems in the product?
- What have you done to obtain the product story?
- What risks I should be aware of?
- What important testing remains to be done?
- What problems are slowing testing down or making it harder to find out what we might need to know?
- What help do you need to speed things up?
- What *specific* aspects of testing are taking time?
- How does your tests link what I need to know?

That Key Question Again

Are there problems
that threaten
the on-time
successful
completion of the project*?

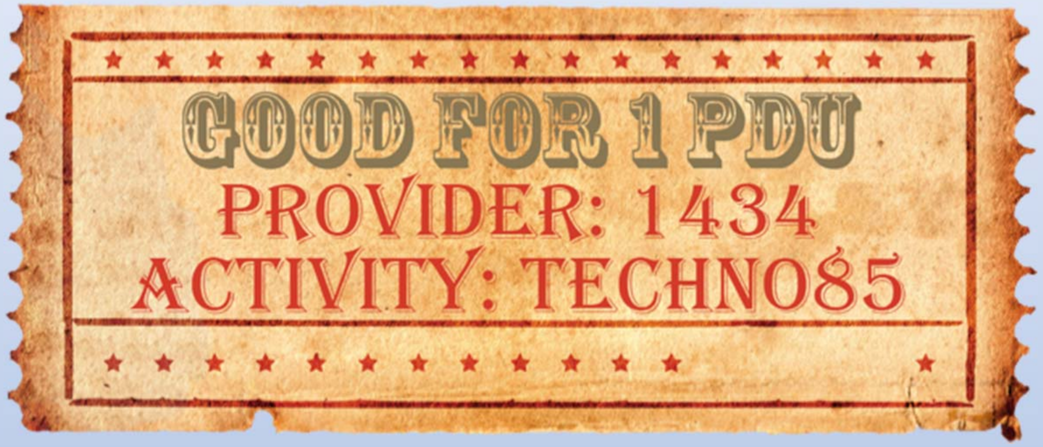
* At any level of granularity

Lots more stuff on…

- Rapid Software Testing methodology and training
- RST in Agile contexts
- Test Strategy
- Oracles (how to recognize problems)
- Coverage (how much testing has been done)
- Recording and Reporting
- Regression Testing
- Session-Based Test Management
- Measurement and Metrics
- Estimation
- …

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

James Bach
Satisfice
http://www.satisfice.com
@jamesmarcusbach
james@satisfice.com



GOOD FOR 1 PDU
PROVIDER: 1434
ACTIVITY: TECHNO85

PLEASE NOTE: All PDU claims are subject to audit by PMI.
This PDU can only be claimed by those who have viewed all of the presentation

For details on claiming your PDU? Download this file.
www.technobility.com/docs/reportingpdus.pdf

Vimeo.com/Technobility

Peter de Jager

pdejager@technobility.com

www.technobility.com

@pdejager

Mark Mullaly

mark.mullaly@interthink.ca

www.interthink.ca

@markmullaly