A Ridiculously Rapid Introduction to Rapid Software Testing

James Bach, Satisfice, Inc.
james@satisfice.com
www.satisfice.com
+1 (360) 376-4367

Michael Bolton, DevelopSense
michael@developsense.com
www.developsense.com
+1 (416) 992-8378

Quality Jam 2017

I’m Michael Bolton

I help people solve testing problems they didn’t know how to solve, and teach them how they can do that too.

Slides and Updates

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

Thank Yous

- Jeff Perkins, Gina Kawalek, and the QA Symphony team
- The audio-visual people
  - like testers, you never notice them when everything all goes well
- James Bach, Keith Klain, Jerry Weinberg, Cem Kaner, Harry Collins
- …and you. Because this may get a little rough!

Why Are We Here?

“We’re making a product!”
“We need you to start testing it right now!”

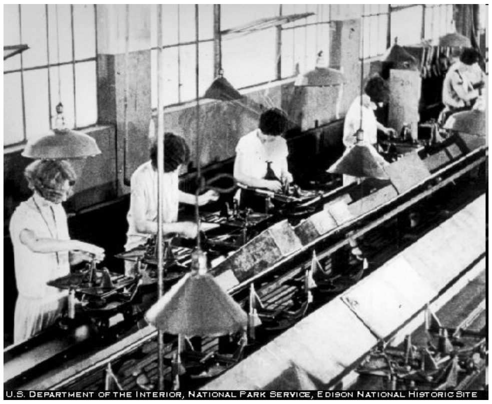
What do you do?

Testing in two easy steps!

1. Prepare test cases.
2. Execute test cases.

Maybe it's more like this…

1. Read the specification.
2. Identify specific items to be checked.
3. Prepare test cases.
4. Execute test cases.



U.S. Department of the Interior, National Park Service, Edison National Historic Site

Or maybe it's more like this…

1. Read the spec.

1.1. OMG there is no spec!

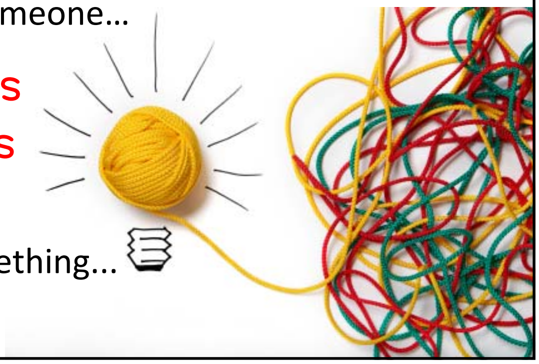
1.2 Oh wait, there is a spec! I'll just read it.

1.2.1 OMG the spec is old and confusing and maybe WRONG…

1.3 Maybe I should ask someone…

1.3.1. OMG Nobody seems to know how this thing is supposed to work!

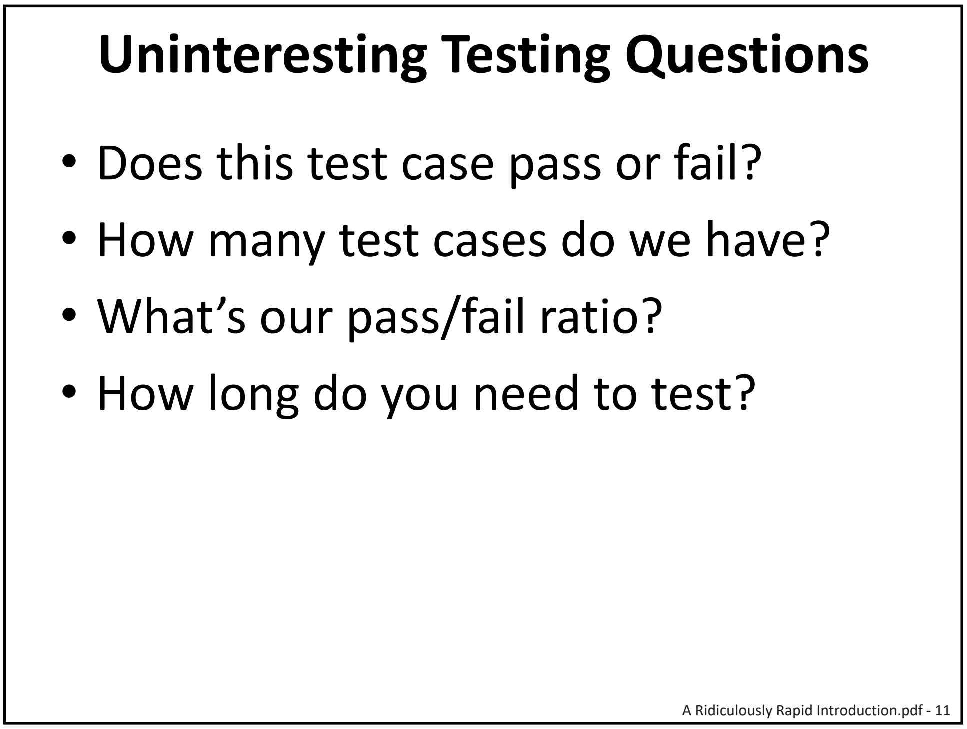
1.3.2. Wait… is there something… *anything* I can test?



Yes! You CAN test…

- …the product
- …a mockup of the product
- …some document describing the product
- …a diagram that models the product
- …a product *like* this product
- …somebody's ideas about the product

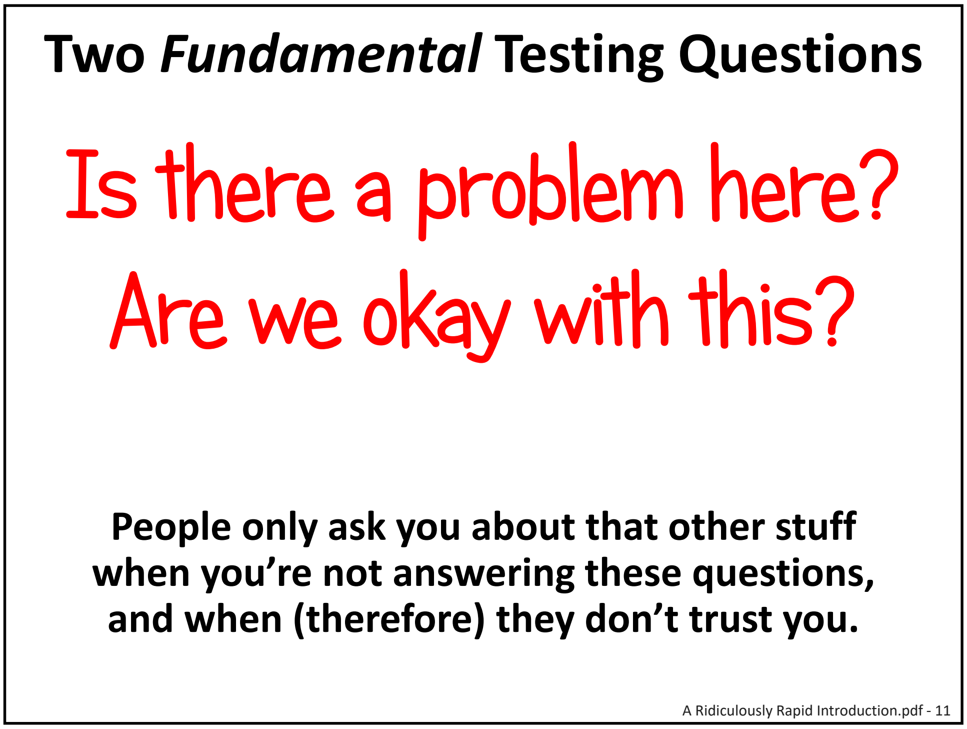
Testing is the process of evaluating a product
by learning about it
through exploration and experimentation.



Uninteresting Testing Questions

- Does this test case pass or fail?
- How many test cases do we have?
- What’s our pass/fail ratio?
- How long do you need to test?

A Ridiculously Rapid Introduction.pdf - 11



Two *Fundamental* Testing Questions

Is there a problem here?

Are we okay with this?

People only ask you about that other stuff when you’re not answering these questions, and when (therefore) they don’t trust you.

A Ridiculously Rapid Introduction.pdf - 11

What do managers and developers really want from testers?

An answer to this question:

Are there problems

that threaten

the on-time successful

completion of the project?

Premises of Rapid Software Testing

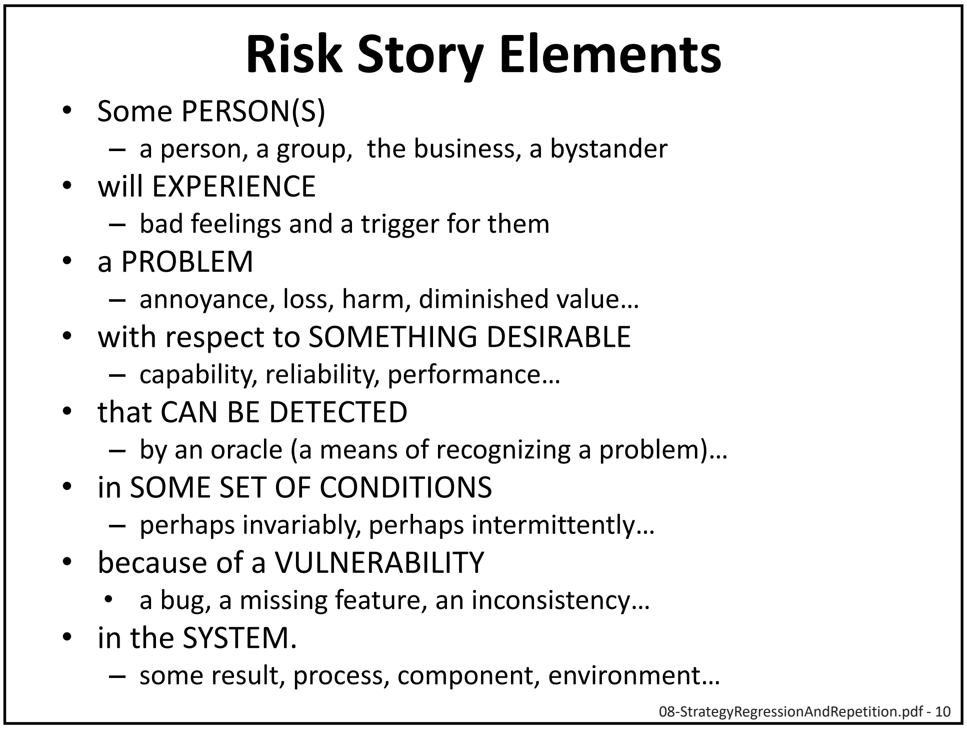
Social Context

1. Software projects and products are **relationships between people**.
2. Every project occurs under conditions of **uncertainty and time pressure**.
3. Despite our best hopes and intentions, some degree of **inexperience, carelessness, and incompetence is normal**. (Plus, people don't follow "the rules".)

Practical limitations

Therefore, software products and projects are fraught with risk.
It is testing's mission to investigate risk.

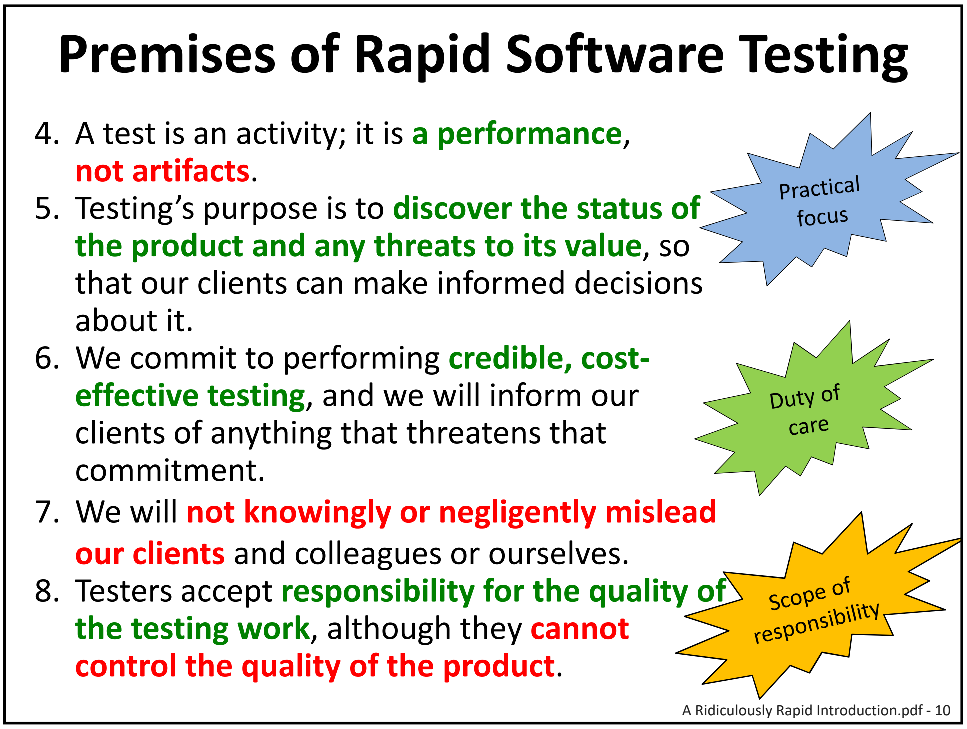
See http://www.developsense.com/blog/2012/09/premises-of-rapid-software-testing-part-1/, and the following two posts.



Risk Story Elements

- Some PERSON(S)
  - a person, a group, the business, a bystander
- will EXPERIENCE
  - bad feelings and a trigger for them
- a PROBLEM
  - annoyance, loss, harm, diminished value…
- with respect to SOMETHING DESIRABLE
  - capability, reliability, performance…
- that CAN BE DETECTED
  - by an oracle (a means of recognizing a problem)…
- in SOME SET OF CONDITIONS
  - perhaps invariably, perhaps intermittently…
- because of a VULNERABILITY
  - a bug, a missing feature, an inconsistency…
- in the SYSTEM.
  - some result, process, component, environment…

08-StrategyRegressionAndRepetition.pdf - 10



Premises of Rapid Software Testing

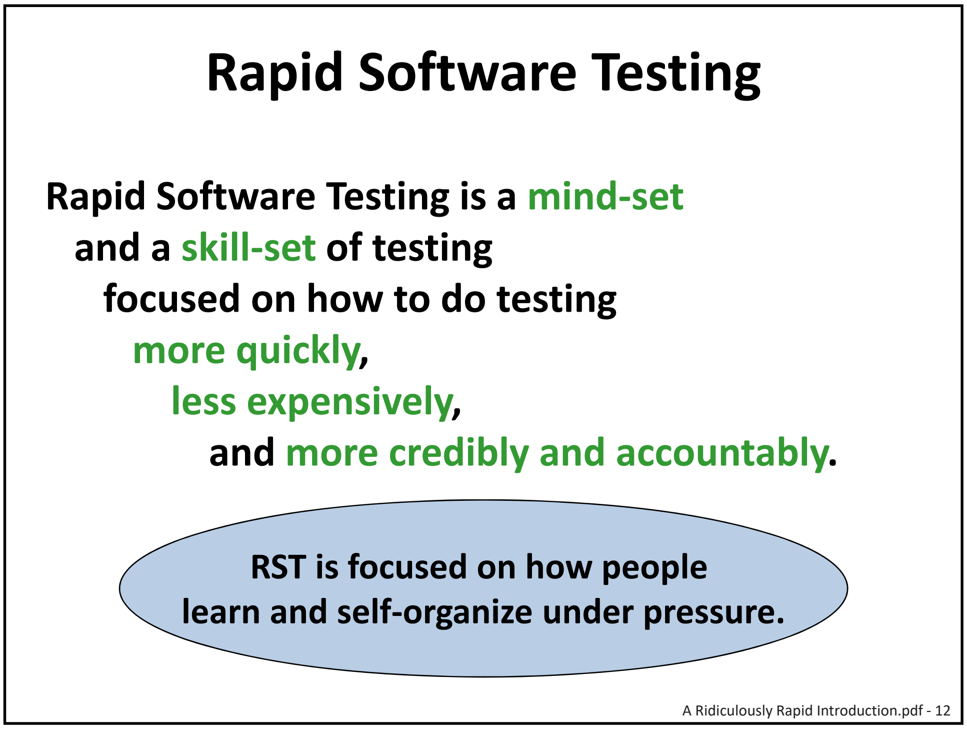
4. A test is an activity; it is **a performance**, **not artifacts**.
5. Testing's purpose is to **discover the status of the product and any threats to its value**, so that our clients can make informed decisions about it.
6. We commit to performing **credible, cost-effective testing**, and we will inform our clients of anything that threatens that commitment.
7. We will **not knowingly or negligently mislead our clients** and colleagues or ourselves.
8. Testers accept **responsibility for the quality of the testing work**, although they **cannot control the quality of the product**.

Practical focus

Duty of care

Scope of responsibility

A Ridiculously Rapid Introduction.pdf - 10

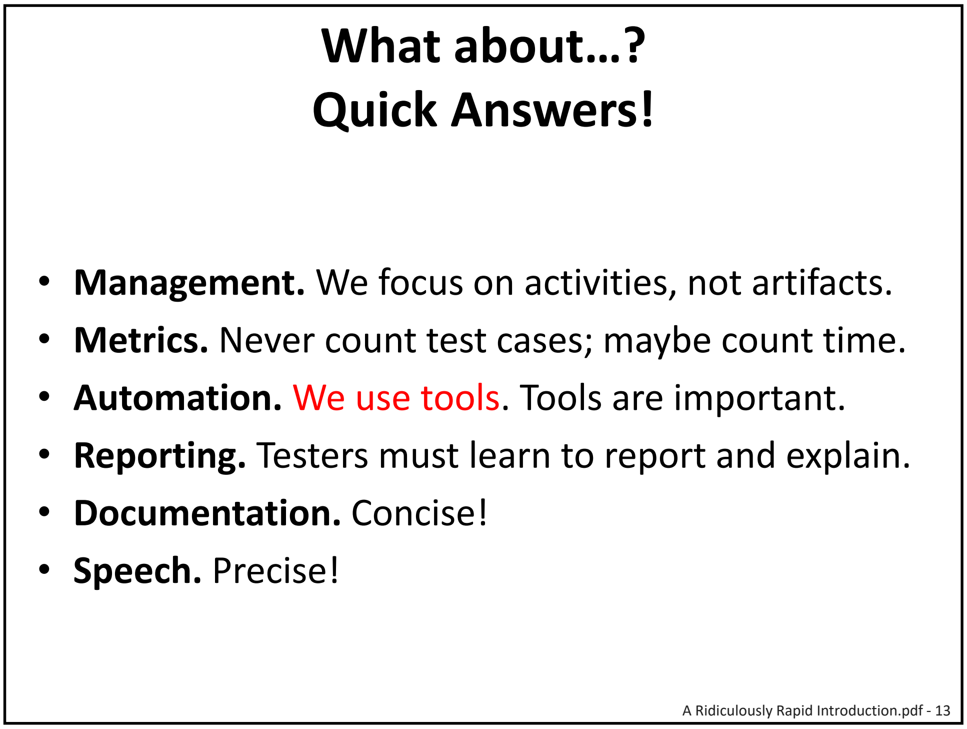


Rapid Software Testing

Rapid Software Testing is a mind-set
and a skill-set of testing
focused on how to do testing
more quickly,
less expensively,
and more credibly and accountably.

RST is focused on how people learn and self-organize under pressure.

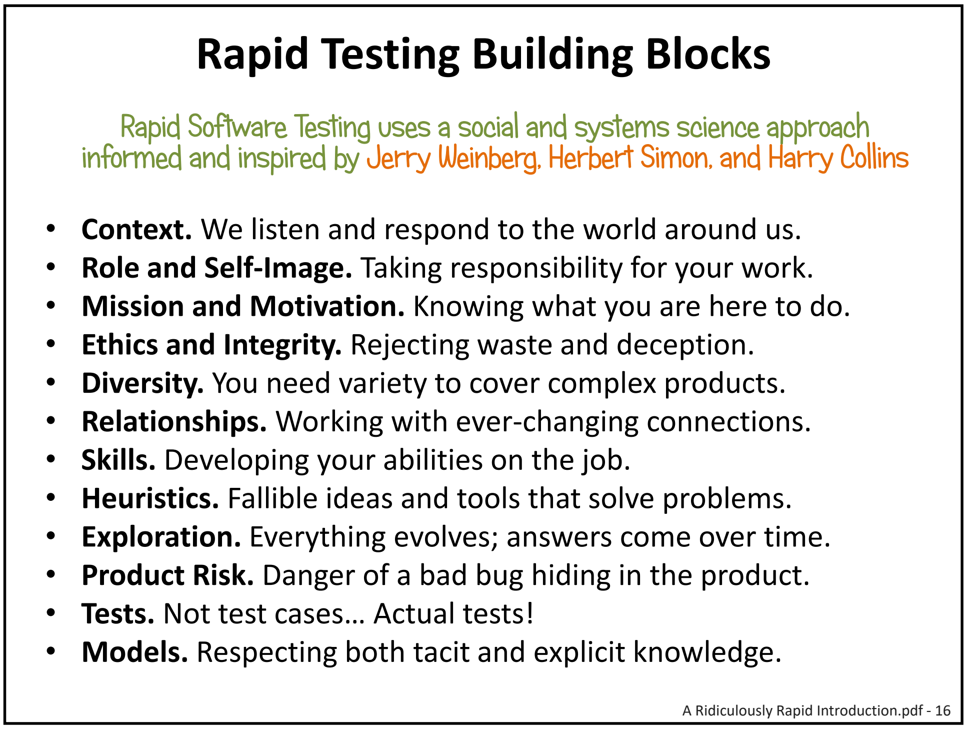
A Ridiculously Rapid Introduction.pdf - 12



What about…? Quick Answers!

- **Management.** We focus on activities, not artifacts.
- **Metrics.** Never count test cases; maybe count time.
- **Automation.** We use tools. Tools are important.
- **Reporting.** Testers must learn to report and explain.
- **Documentation.** Concise!
- **Speech.** Precise!

A Ridiculously Rapid Introduction.pdf - 13

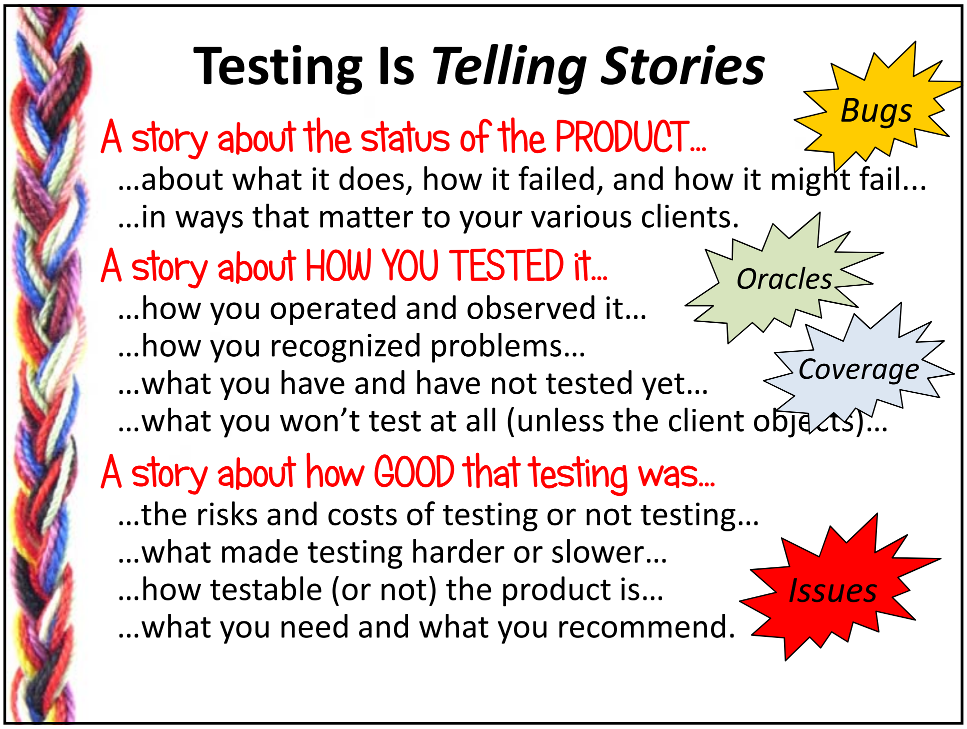


Rapid Testing Building Blocks

Rapid Software Testing uses a social and systems science approach informed and inspired by Jerry Weinberg, Herbert Simon, and Harry Collins

- **Context.** We listen and respond to the world around us.
- **Role and Self-Image.** Taking responsibility for your work.
- **Mission and Motivation.** Knowing what you are here to do.
- **Ethics and Integrity.** Rejecting waste and deception.
- **Diversity.** You need variety to cover complex products.
- **Relationships.** Working with ever-changing connections.
- **Skills.** Developing your abilities on the job.
- **Heuristics.** Fallible ideas and tools that solve problems.
- **Exploration.** Everything evolves; answers come over time.
- **Product Risk.** Danger of a bad bug hiding in the product.
- **Tests.** Not test cases… Actual tests!
- **Models.** Respecting both tacit and explicit knowledge.

A Ridiculously Rapid Introduction.pdf - 16



Testing Is *Telling Stories*

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 your various clients.

A story about HOW YOU TESTED it…

…how you operated and observed it…
…how you recognized problems…
…what you have and have not tested yet…
…what you won't test at all (unless the client objects)…

A story about how GOOD that testing was…

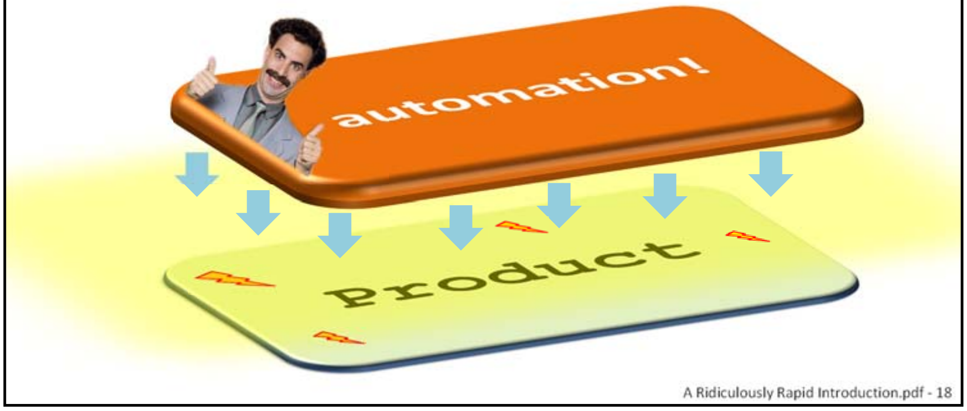
…the risks and costs of testing or not testing…
…what made testing harder or slower…
…how testable (or not) the product is…
…what you need and what you recommend.

Those who develop and foster testing and reporting skill can account for their test coverage.

Those who do not focus on those skills will inevitably resort to ass coverage.

Many believe that to test well means to build a great big test case re-running machine!

“I rerun my old tests to ensure that nothing has broken.”



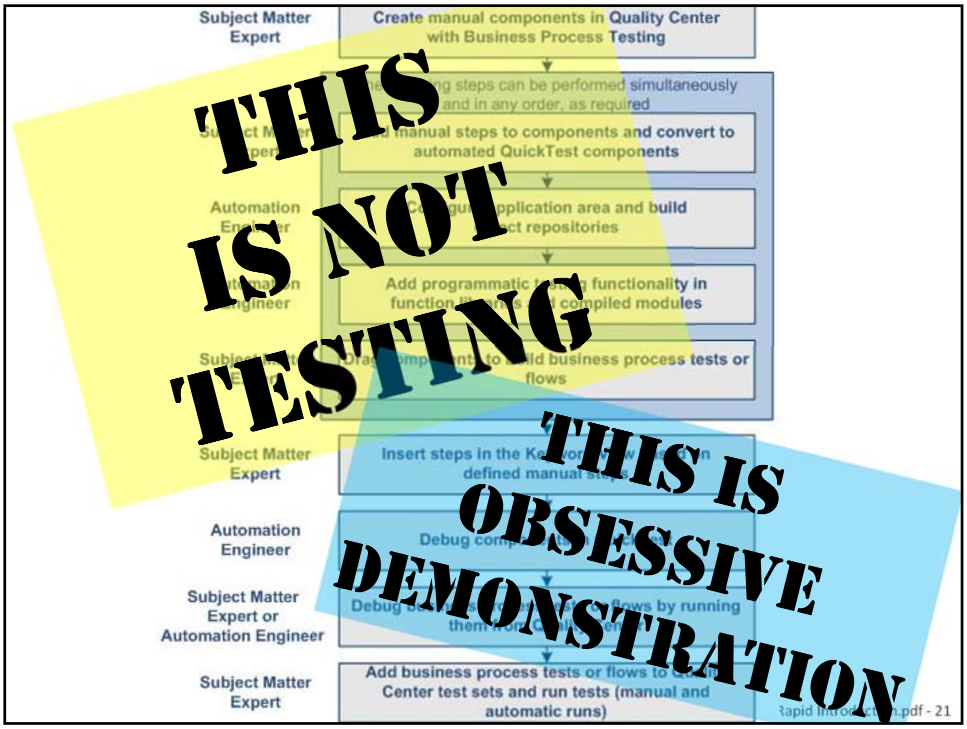
This is based on the belief that testing is easy and that we should treat it like "factory work."

"I rerun my old tests to ensure that nothing has broken."

This can only be true if your old tests cover **everything** completely with **perfect** oracles so that **all** conceivable bugs are detected…

Test factories (whether "manual" or "automated") are costly to build and maintain. And they miss bugs!

"We tour the product a little bit…
Maybe find some obvious bugs…
And hope the bugs we miss aren't bad ones!"



Subject Matter Expert — Create manual components in Quality Center with Business Process Testing

The following steps can be performed simultaneously and in any order, as required

Subject Matter Expert — Add manual steps to components and convert to automated QuickTest components

Automation Engineer — Configure application area and build object repositories

Automation Engineer — Add programmatic testing functionality in function libraries and compiled modules

Subject Matter Expert — Drag components to build business process tests or flows

Subject Matter Expert — Insert steps in the Keyword View based on defined manual steps

Automation Engineer — Debug components in QuickTest

Subject Matter Expert or Automation Engineer — Debug business process tests or flows by running them from Quality Center

Subject Matter Expert — Add business process tests or flows to Quality Center test sets and run tests (manual and automatic runs)

THIS IS NOT TESTING

THIS IS OBSESSIVE DEMONSTRATION

A Ridiculously Rapid Introduction.pdf - 21

Testing

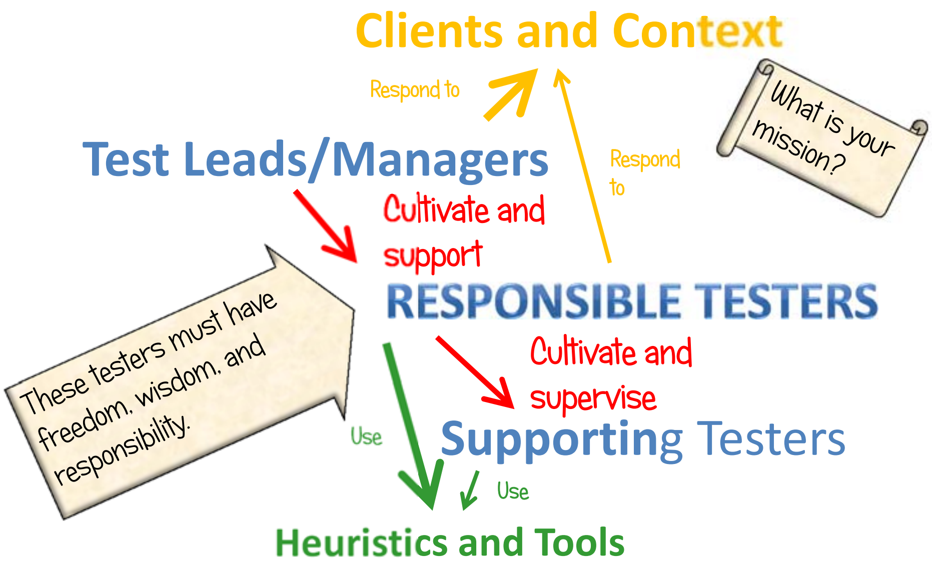
Scripted Testing

A Ridiculously Rapid Introduction.pdf - 22

Why not say "exploratory testing"?

Why not say "vegetarian cauliflower"?

Three Testing Roles



Clients and Context

Respond to

Respond to

What is your mission?

Test Leads/Managers

Cultivate and support

RESPONSIBLE TESTERS

These testers must have freedom, wisdom, and responsibility.

Cultivate and supervise

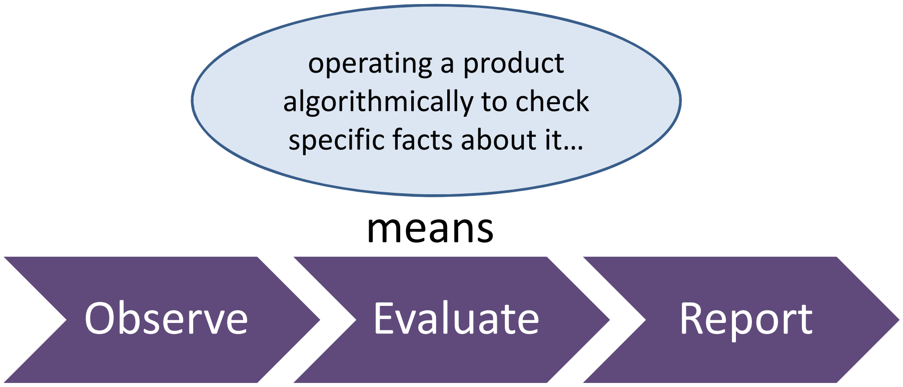
Supporting Testers

Use

Use

Heuristics and Tools

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.

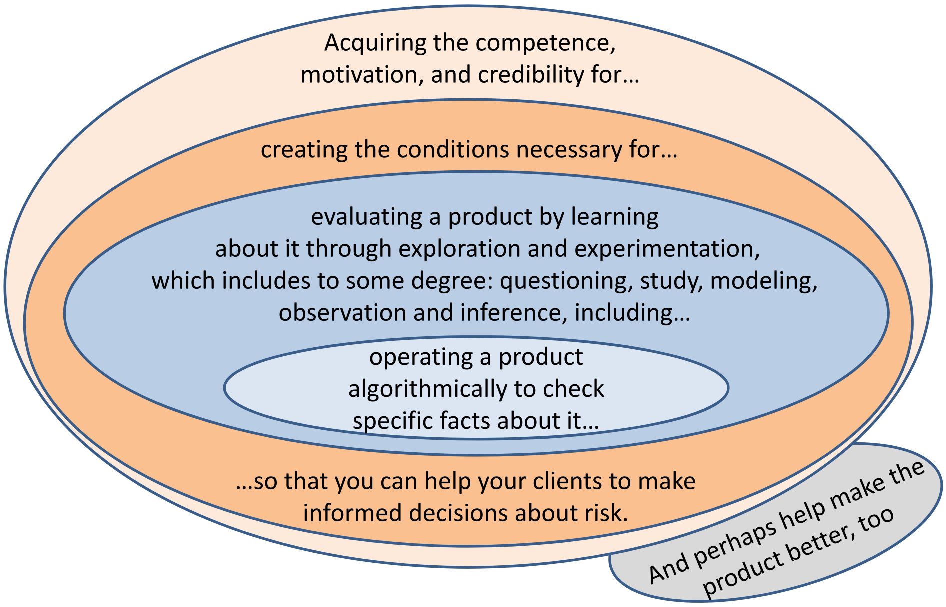
Report any failed checks *algorithmically*.

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)

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

A Ridiculously Rapid Introduction.pdf - 27

They say…

“Automate all the testing!”

They might have meant…

“Automate all the checking!”

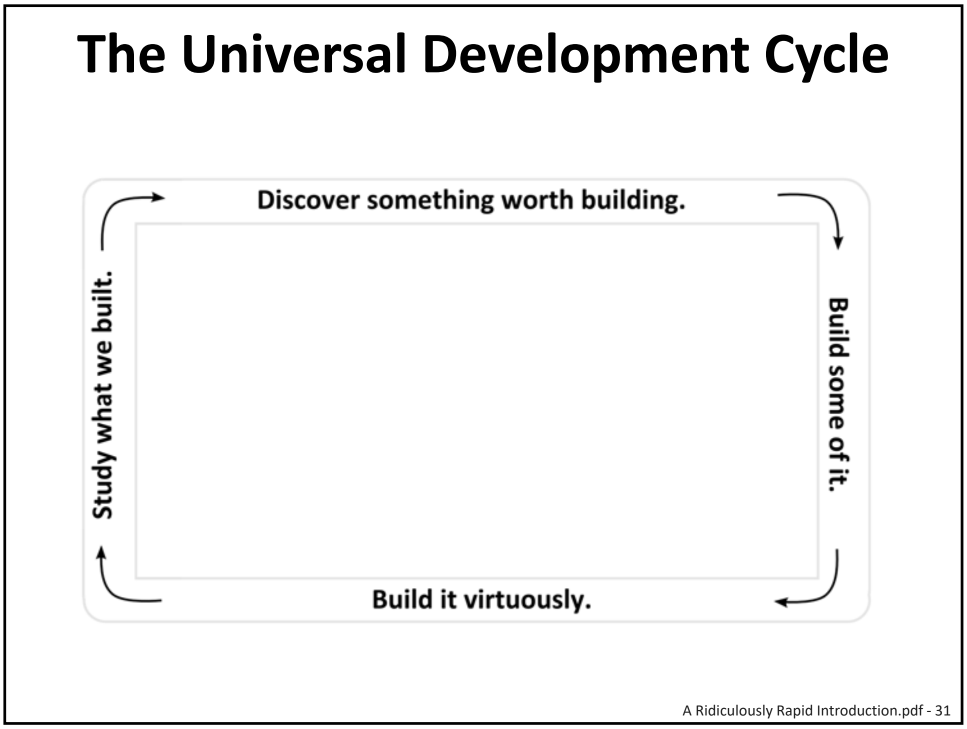
or…

“Use tools!”

…which means

and risk assessment
and task prioritization
and coverage analysis
and pattern recognition
and decision making
and design of the test lab
and preparation of the test lab
and sensemaking
and test code development
and tool selection
and recruiting of helpers
and making test notes
and preparing simulations
and interacting with developers
and bug advocacy
and triage

A Ridiculously Rapid Introduction.pdf - 28



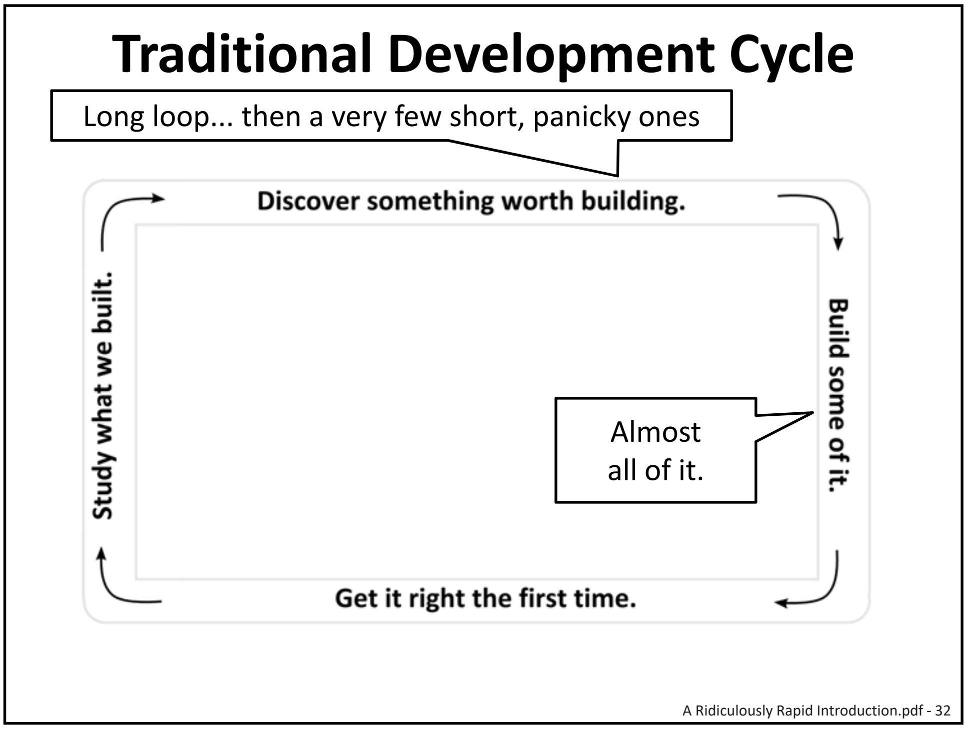
The Universal Development Cycle

Discover something worth building.

Build some of it.

Build it virtuously.

Study what we built.



Traditional Development Cycle

Long loop... then a very few short, panicky ones

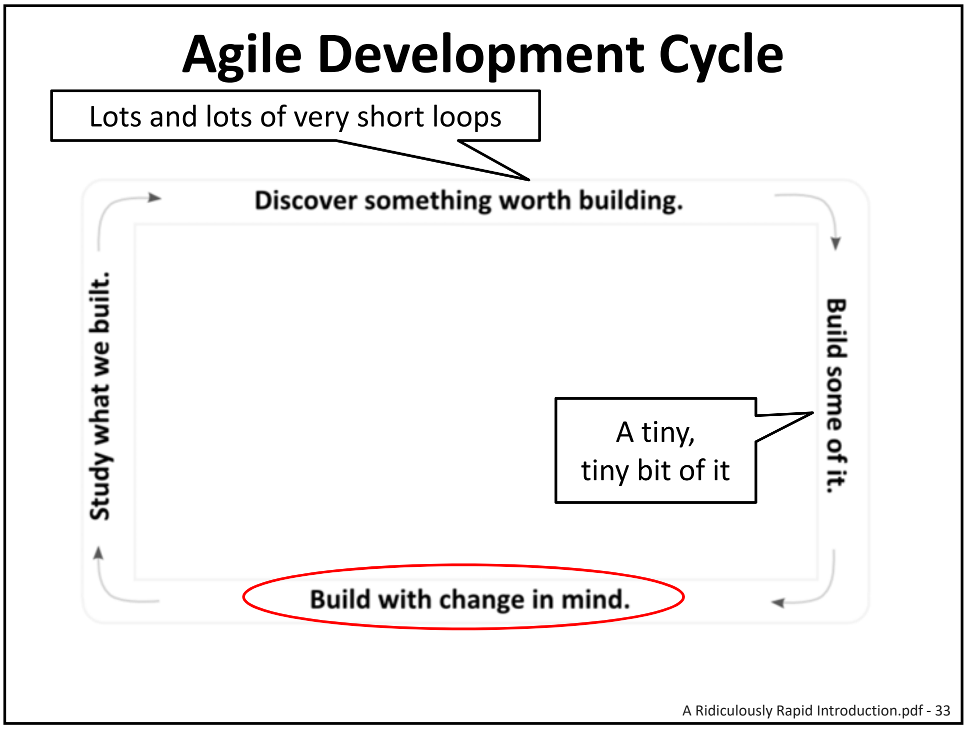
Discover something worth building.

Build some of it.

Almost all of it.

Get it right the first time.

Study what we built.



Agile Development Cycle

Lots and lots of very short loops

Discover something worth building.

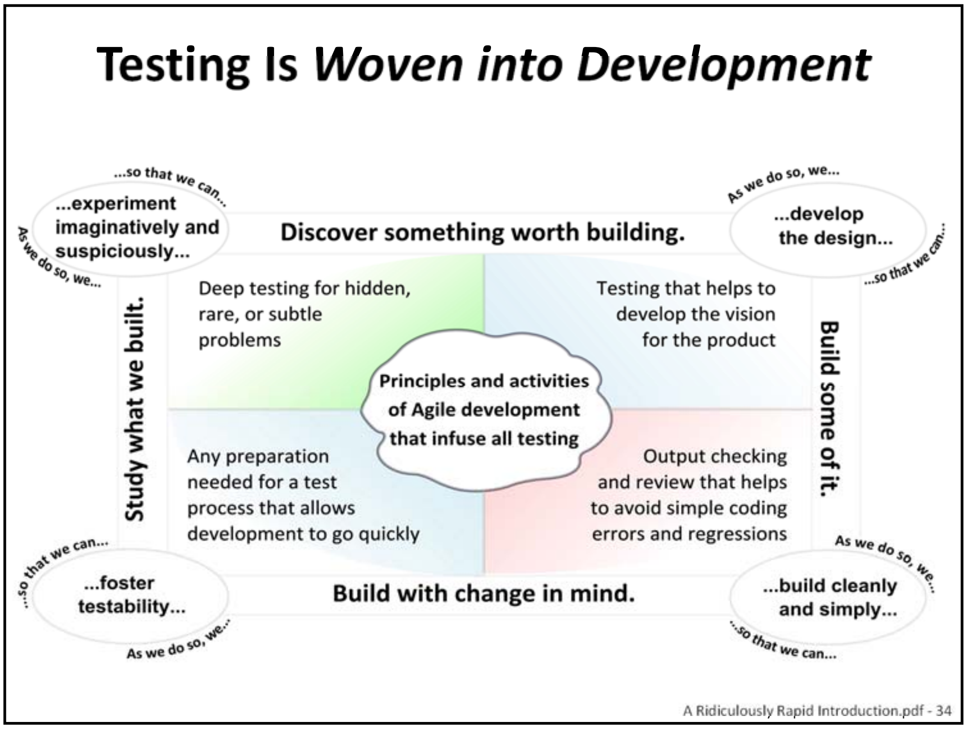
Build some of it.

A tiny, tiny bit of it

Build with change in mind.

Study what we built.

A Ridiculously Rapid Introduction.pdf - 33



Testing Is *Woven into Development*

...so that we can...

...experiment imaginatively and suspiciously...

As we do so, we...

Discover something worth building.

As we do so, we...

...develop the design...

...so that we can...

Deep testing for hidden, rare, or subtle problems

Testing that helps to develop the vision for the product

Principles and activities of Agile development that infuse all testing

Any preparation needed for a test process that allows development to go quickly

Output checking and review that helps to avoid simple coding errors and regressions

Study what we built.

Build some of it.

...so that we can...

...foster testability...

As we do so, we...

Build with change in mind.

As we do so, we...

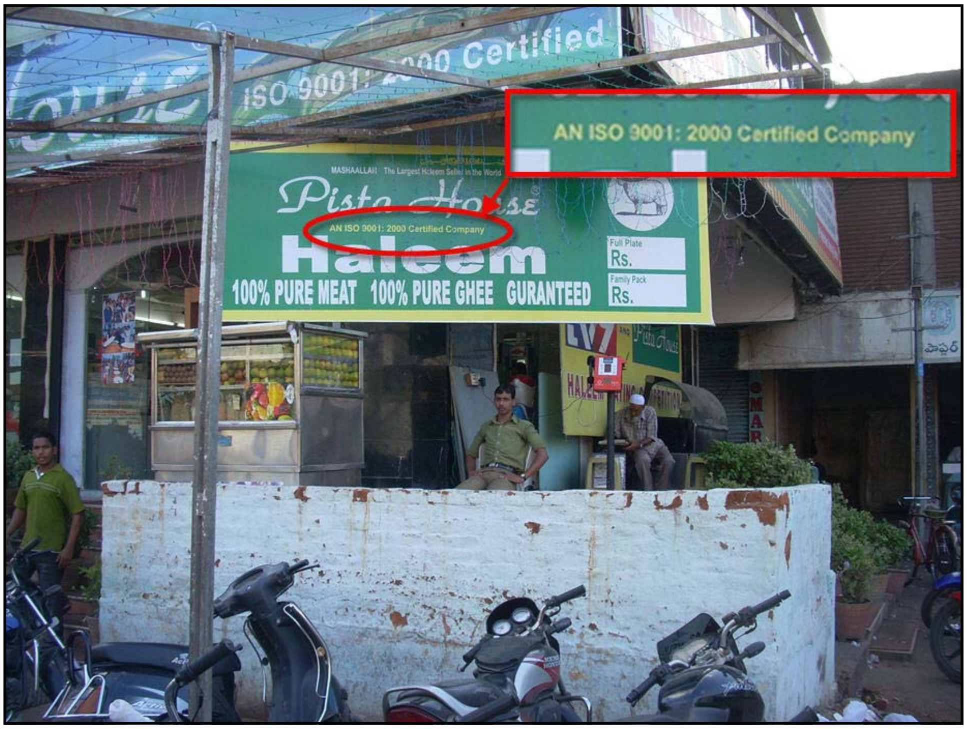
...build cleanly and simply...

...so that we can...

A Ridiculously Rapid Introduction.pdf - 34

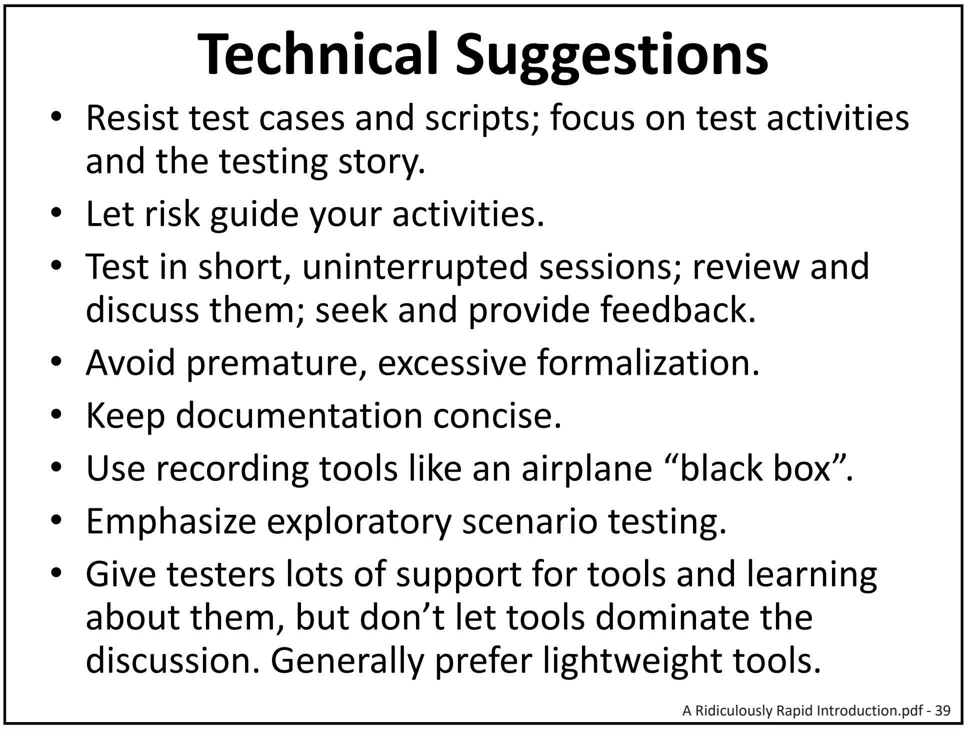
What about standards?

A Ridiculously Rapid Introduction.pdf - 36



AN ISO 9001: 2000 Certified Company

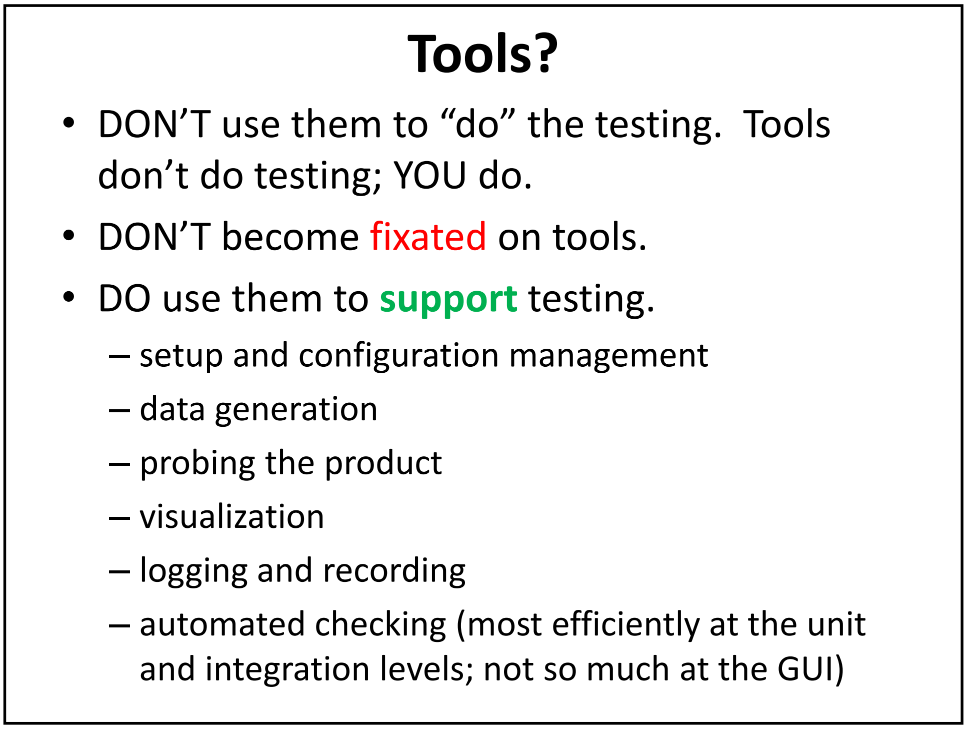
ISO 9001: 2000 Certified
MASHAALLAH · The Largest Haleem Seller in the World
Pista House
AN ISO 9001: 2000 Certified Company
Haleem
100% PURE MEAT 100% PURE GHEE GURANTEED
Full Plate
Rs.
Family Pack
Rs.



Technical Suggestions

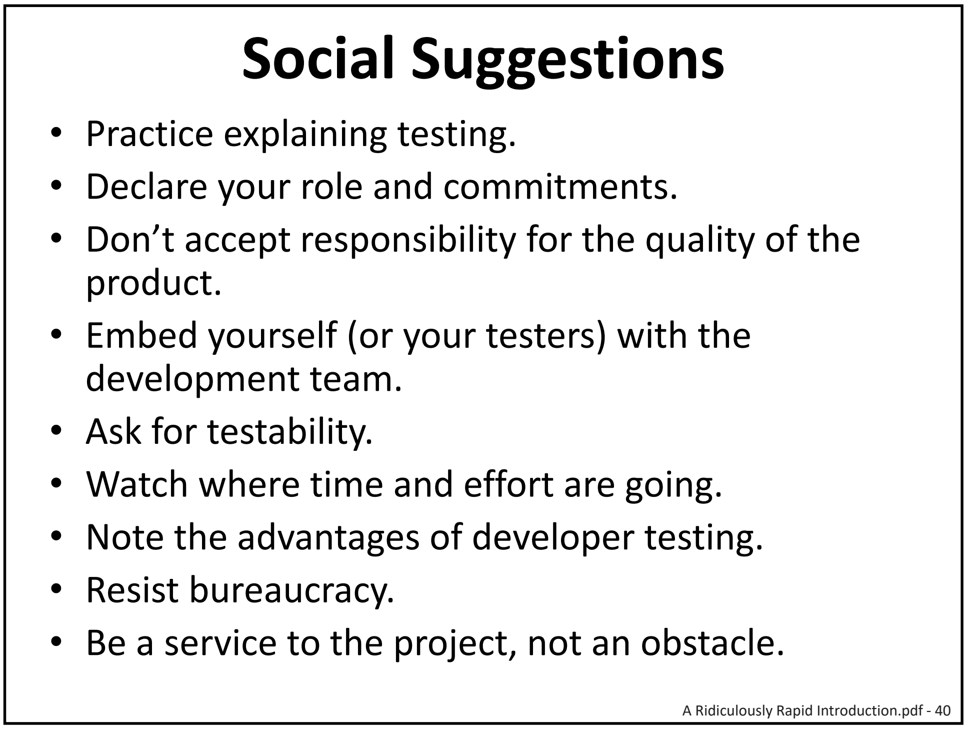
- Resist test cases and scripts; focus on test activities and the testing story.
- Let risk guide your activities.
- Test in short, uninterrupted sessions; review and discuss them; seek and provide feedback.
- Avoid premature, excessive formalization.
- Keep documentation concise.
- Use recording tools like an airplane “black box”.
- Emphasize exploratory scenario testing.
- Give testers lots of support for tools and learning about them, but don’t let tools dominate the discussion. Generally prefer lightweight tools.

A Ridiculously Rapid Introduction.pdf - 39



Tools?

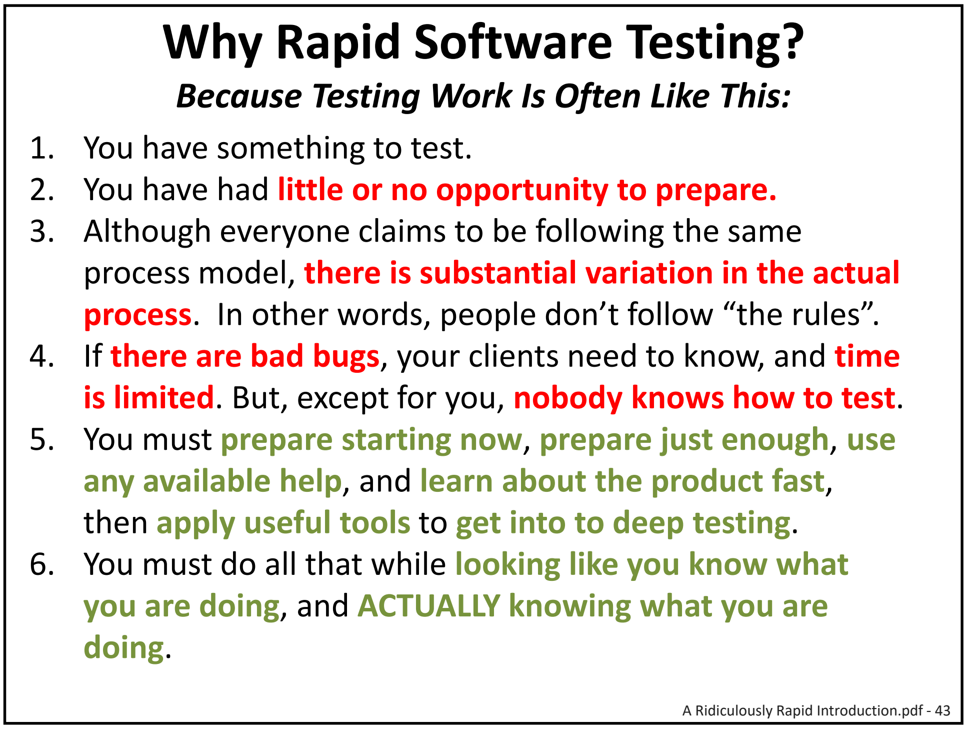
- DON’T use them to “do” the testing. Tools don’t do testing; YOU do.
- DON’T become fixated on tools.
- DO use them to **support** testing.
  - setup and configuration management
  - data generation
  - probing the product
  - visualization
  - logging and recording
  - automated checking (most efficiently at the unit and integration levels; not so much at the GUI)



Social Suggestions

- Practice explaining testing.
- Declare your role and commitments.
- Don’t accept responsibility for the quality of the product.
- Embed yourself (or your testers) with the development team.
- Ask for testability.
- Watch where time and effort are going.
- Note the advantages of developer testing.
- Resist bureaucracy.
- Be a service to the project, not an obstacle.

A Ridiculously Rapid Introduction.pdf - 40

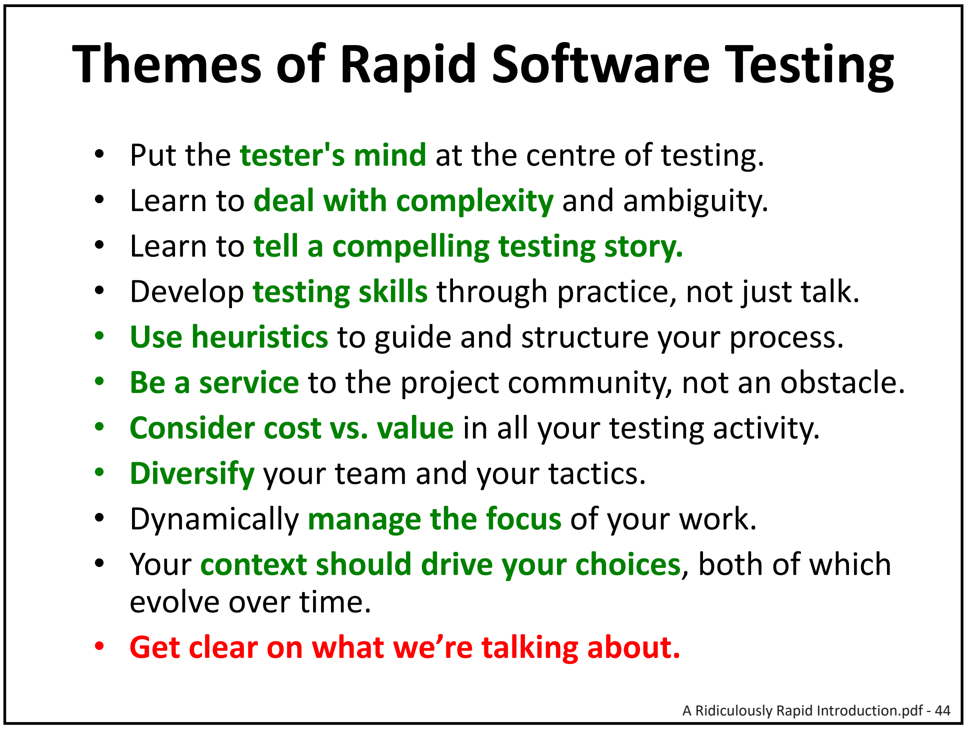


Why Rapid Software Testing?

Because Testing Work Is Often Like This:

1. You have something to test.
2. You have had **little or no opportunity to prepare.**
3. Although everyone claims to be following the same process model, **there is substantial variation in the actual process**. In other words, people don’t follow “the rules”.
4. If **there are bad bugs**, your clients need to know, and **time is limited**. But, except for you, **nobody knows how to test**.
5. You must **prepare starting now**, **prepare just enough**, **use any available help**, and **learn about the product fast**, then **apply useful tools** to **get into to deep testing**.
6. You must do all that while **looking like you know what you are doing**, and **ACTUALLY knowing what you are doing**.

A Ridiculously Rapid Introduction.pdf - 43



Themes of Rapid Software Testing

- Put the **tester's mind** at the centre of testing.
- Learn to **deal with complexity** and ambiguity.
- Learn to **tell a compelling testing story.**
- Develop **testing skills** through practice, not just talk.
- **Use heuristics** to guide and structure your process.
- **Be a service** to the project community, not an obstacle.
- **Consider cost vs. value** in all your testing activity.
- **Diversify** your team and your tactics.
- Dynamically **manage the focus** of your work.
- Your **context should drive your choices**, both of which evolve over time.
- **Get clear on what we’re talking about.**

A Ridiculously Rapid Introduction.pdf - 44