Myths of Exploratory Testing

Uses and Misuses, Origin and Origin Stories

Introduction

- Exploratory testing: a technique of using knowledge, experience, and skills to test software in a non-linear, investigatory fashion
- What are the origins of exploratory testing?
- Is it magic quality pixie dust?
- Is it the only real way of testing?
- Isn't it just an unmanageable, unaccountable, random bug hunt?
- Can we live without it?
- Let's see…

What Is Exploratory Testing

- When testers use their skills, knowledge, and experience as their primary test basis and test oracle, this is experience-based testing
- When such testing is non-linear and investigatory, it you can call it exploratory
- Two-thirds of test teams report using exploratory testing, higher than any other technique except use cases (70%), though that probably says more about the lack of proper training of testers than the value of exploratory testing
- Exploratory testing is a form of validation, as it is about measuring the software against user and customer expectations (as understood by the tester)
- Contrast verification, which is about measuring the software against defined requirements and design specification
- We'll look more closely at the use of exploratory testing later, but first, some myth-busting…

Origin Myth

- Myth: exploratory testing was invented in the 1990s in Silicon Valley
- In fact, it was probably invented by the first person to write and run a program
- Widely used at IBM starting in the 60s, as independent test teams are mentioned by Brooks in The *Mythical Man-month*
- Described, as error guessing, in Myers's book *The Art of Software Testing*, published in the mid-70s

Completeness Myth

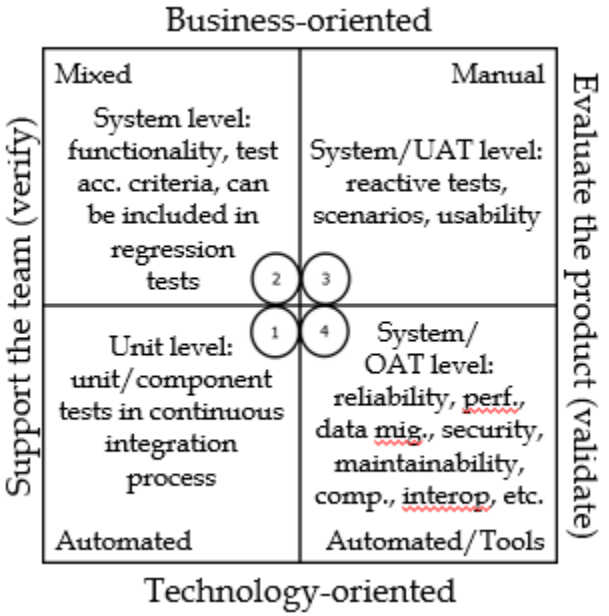
- Myth: just play around with the software, and you can make sure it works before we ship it
- Common misconception among organizations with no exposure to formal testing
- Eventually, in an organization, this myth is self-dispelling, given enough releases
- Absence of testing education in computer science curricula means the myth keeps coming back with each new crop of programmers
- Absence of standards for getting a job in IT means even test education in CS programs wouldn't fully dispel the myth for good

Sufficiency Myth

- Myth: we've been trained by a mighty Jedi of exploratory testing, and therefore we can find every bug that matters
- Studies done at Microsoft show:
  - The set of bugs found by exploratory testing and other reactive test strategies is different than those bugs found by other test strategies
  - These different sets are only partially overlapping
- Think of testing like a water purification system: use multiple stages of filtration (i.e., different test strategies and techniques) to achieve maximum effectiveness
- Understand: no set of test strategies yields 100% defect detection effectiveness

Irrelevance Myth

- Myth: we have TDD, ATDD, and BDD, so we have 100% confidence in our software
- Reliance on verification alone leads to missing a number of important bugs
- Remember no set of specifications can be perfect, just as code is never perfect
- In Agile projects, rely on the balanced use of the test quadrants
- In non-Agile, apply the test types and levels as discussed in ISTQB Foundation syllabus
- This includes experience-based testing



Unmanageable Myth

- Myth: when people do exploratory testing, there's no way to manage it or capture test coverage
- Logging tools, open-source and other (e.g., Rapid Reporting), can capture:
  - What data, risks, user stories, etc. have been covered
  - Notes about interesting discoveries, system stability, defects found, next steps
  - Details to be escalated to relevant stakeholders
  - Concerns or questions about expected behavior, test efficiency, test environments, test data, or the system
  - The actual behavior (e.g., screen captures or video clips)
- The use of time-boxing a well-defined scope of testing (called chartered testing sessions) allows management and direction

Exploratory Testing and Reactive Strategies

- Exploratory testing and other techniques associated with reactive test strategies are useful in all situations, since requirements are never perfect
- In Agile projects especially, limited documentation and on-going change make these reactive strategies even more useful
- Blend reactive testing with other strategies (e.g., analytical risk-based, analytical requirements-based, regression-averse)
- For exploratory testing:
  - Analysis produces the test condition(s) for the test charter which will guide a test session (30-120 minutes)
  - Test design and test execution occur at the same time, covering the charter, once software is delivered to testers
- Test design can use formal test techniques (e.g., ISTQB Foundation, Advanced Test Analyst, and Advanced Technical Test Analyst), influenced by the results of the previous tests

Managing Exploratory Test Sessions

- Time-boxed, chartered test sessions help you manage it, preventing overlap and runaway testers
- Session includes time to setup, design/execute tests, log results, and investigate bugs, and actual time should be recorded
- Sessions should include activities for basic familiarization, evaluation of the feature or function, and (where risk justifies it) deep coverage
- Session charter may list actor (intended user), purpose (test conditions), setup (pre-conditions), priority, references (test basis), test data, activities (suggested actions), test oracle, variations to try

Questions and Heuristics

- Questions to ask while exploring:
  - What is most important to discover?
  - How might the system fail?
  - What does/should happen if/when?
  - Will customer/user be satisfied (validation)?
  - Does it install/upgrade/uninstall?
- Utilize all creativity, intuition, ideas, and skills with…
  - The system
  - The business domain
  - The ways people use the software
  - The ways the software fails
- Also, consider heuristics such as boundaries, CRUD (Create, Read, Update, Delete), configuration variations, and possible interruptions

Example: Exploratory Test Charter

- Actor: customer
- Purpose: check if filtering allows access to pornographic, objectionable, lewd, obscene, or violent sites, or blocks access to sites that aren't
- Setup: initiate a browsing session
- Priority: high
- References: MRD section 3.1.6, user story "XYZ"
- Test data: none needed
- Activities: To get started, think of words that qualify as or are associated with pornography, lewd behavior, obscenity, violence, or are otherwise objectionable or could be objectionable to some people; use a search engine to locate websites based on these words; try to visit the site; repeat; watch for patterns of failure
- Test oracle: As Justice Stewart said, you'll know it when you see it. However, some people might have less tolerance for such things than you, so err on the side of reporting questionable sites.
- Variations to try: the possibilities are endless

Conclusions

- Exploratory testing was almost certainly invented by the first programmer
- A number of myths surround exploratory testing, which limit the effectiveness and smart use of this proven best practice
- Done properly, exploratory testing is an essential element—but only one element—of a successful test approach
- Use structuring techniques such as time-boxes and charters to keep it productive
- Exploit the skills and experience of your testers to find bugs other techniques miss

To Contact RBCS

For over twenty years, RBCS has delivered consulting, outsourcing and training services to clients, helping them with software and hardware testing. Employing the industry's most experienced and recognized consultants, RBCS advises its clients, trains their employees, conducts product testing, builds and improves testing groups, and hires testing staff for hundreds of clients worldwide. Ranging from Fortune 20 companies to start-ups, RBCS clients save time and money through improved product development, decreased tech support calls, improved corporate reputation and more. To learn more about RBCS, visit www.rbcs-us.com.

Address:	RBCS, Inc. 31520 Beck Road Bulverde, TX 78163-3911 USA
Phone:	+1 (830) 438-4830
E-mail:	info@rbcs-us.com
Web:	www.rbcs-us.com
Twitter:	@RBCS, @LaikaTestDog
Facebook:	RBCS-Inc.