Exploratory Testing

14.04.2020

Overview

- Test design techniques
- Exploratory Testing in Theory
  - What?
  - When?
  - Benefits and Challenges
- Activities
  - Learn | Design | Execute | Interpret
- Exploratory Testing in Practice
  - Heuristics: Techniques + Examples
- Summary

Test Design Techniques

How and where Exploratory Testing fits in

Test Design Techniques

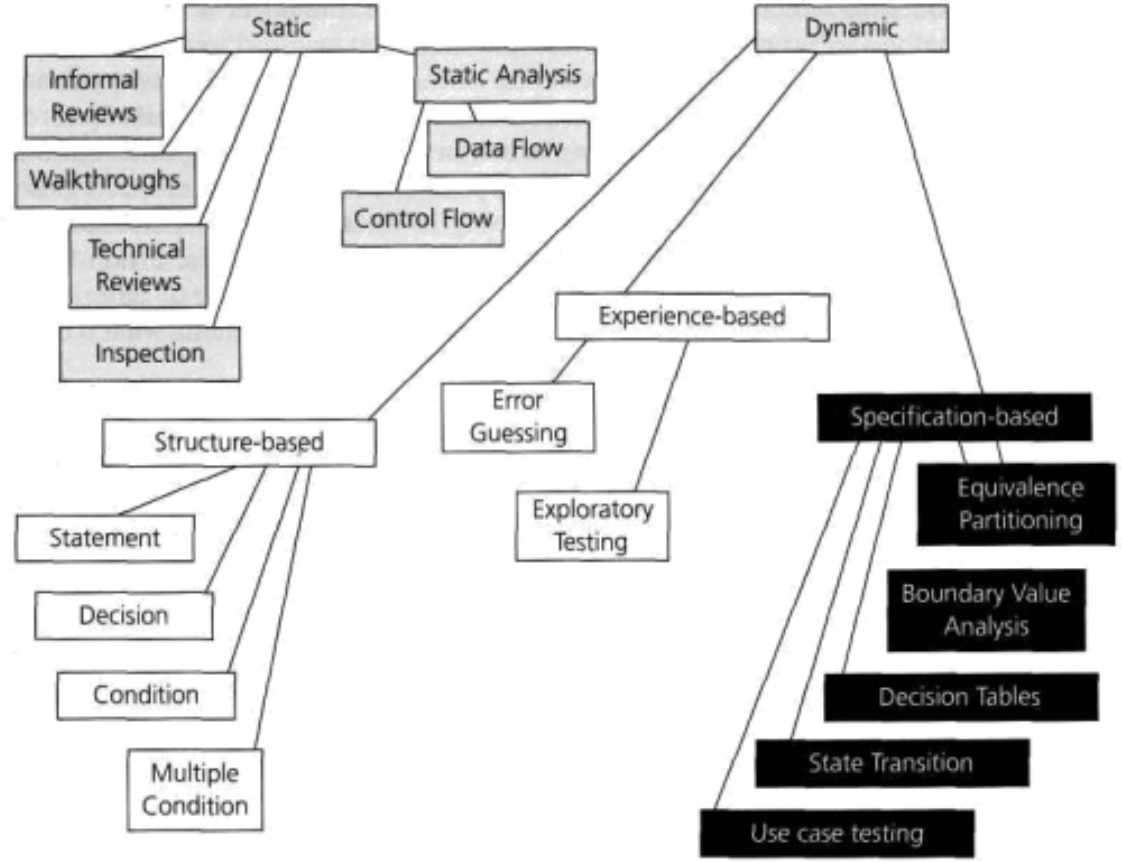


FIGURE 4.1 Testing techniques

ISTQB Foundation Home. December 26th 2016. "Chapter 4: Test Design Techniques". [Online]: https://istqbinformation.wordpress.com/2016/12/23/chapter-4-test-design-techniques/

Static vs. Dynamic Testing

Static testing

- Code not executed
- Inspect code, requirements documents, design documents
- Purpose: Improve quality of the software product

ISTQB Foundation Home. December 26th 2016. "Chapter 4: Test Design Techniques". [Online]: https://istqbinformation.wordpress.com/2016/12/23/chapter-4-test-design-techniques/

Static vs. Dynamic Testing

Static testing

- Code not executed
- Inspect code, requirements documents, design documents
- Purpose: Improve quality of the software product

Dynamic testing

- Code is executed
- Check functional behaviour of the SUT
- Purpose: Confirm software adheres to business requirements

ISTQB Foundation Home. December 26th 2016. "Chapter 4: Test Design Techniques". [Online]: https://istqbinformation.wordpress.com/2016/12/23/chapter-4-test-design-techniques/

Test Design Techniques

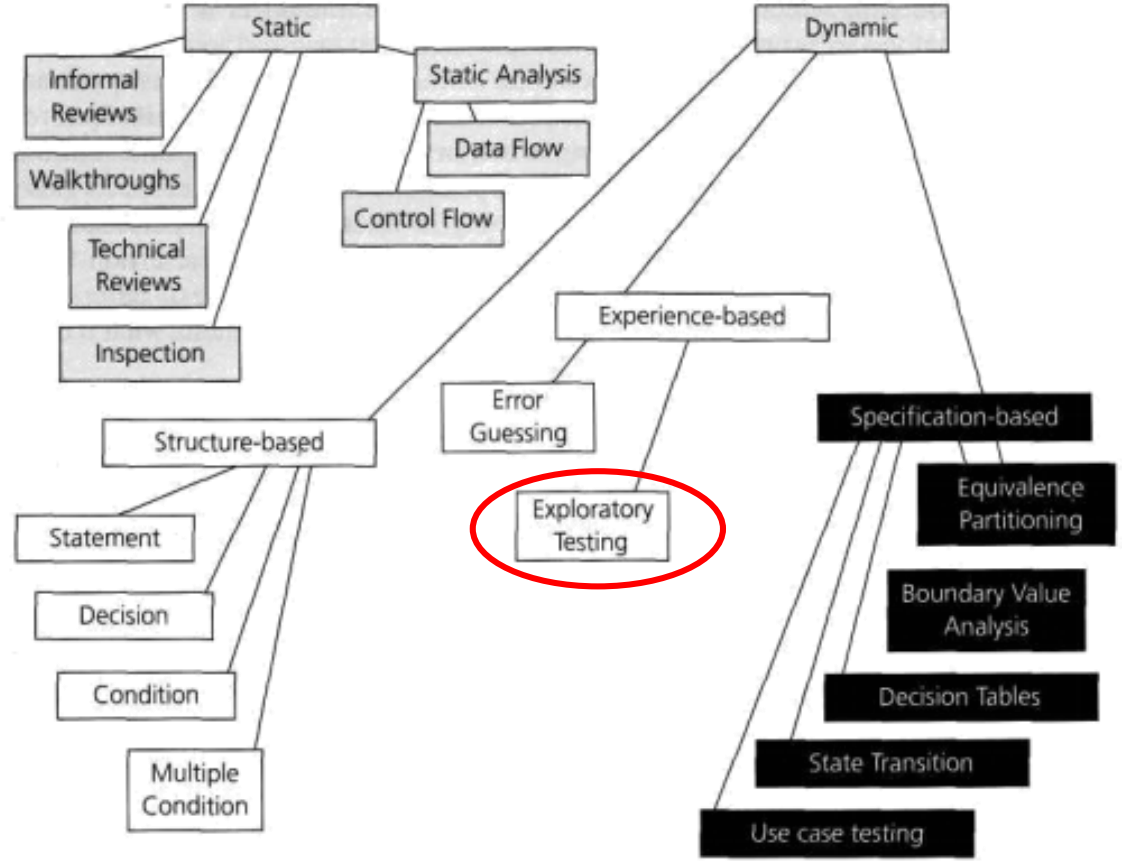


FIGURE 4.1 Testing techniques

ISTQB Foundation Home. December 26th 2016. "Chapter 4: Test Design Techniques". [Online]: https://istqbinformation.wordpress.com/2016/12/23/chapter-4-test-design-techniques/

Exploratory Testing

According to theory

Exploratory Testing: **What?**

- Dynamic technique
- Experience-based

- Test cases not created in advance
  - However, useful to note down ideas before execution
- Popular in agile models
  - Discover | Investigate | Learn
- "Thinking" activity

ISTQB Foundation Home. December 26th 2016. "Chapter 4: Test Design Techniques". [Online]: https://istqbinformation.wordpress.com/2016/12/23/chapter-4-test-design-techniques/

Exploratory Testing: **When to use?**

- Experienced tester
- Poor or absent requirements specifications
- Limited time

- Complement other (more formal) testing techniques
  - "Verify" the formal test effort
  - Establish increased confidence in the SUT

ISTQB Foundation Home. December 26th 2016. "Chapter 4: Test Design Techniques". [Online]: https://istqbinformation.wordpress.com/2016/12/23/chapter-4-test-design-techniques/

Exploratory Testing: **Benefits**

- Limited to no preparation needed
- Bugs are found quickly and during execution
- Uncover bugs that would otherwise be ignored by other testing techniques
- Intellectually appealing
  - Creativity + Intuition

ISTQB Foundation Home. December 26th 2016. "Chapter 4: Test Design Techniques". [Online]: https://istqbinformation.wordpress.com/2016/12/23/chapter-4-test-design-techniques/

Exploratory Testing: **Challenges**

- Must learn to use the application / system
  - Domain knowledge
  - Skill of the tester
- Sometimes difficult to replicate failures / certain behaviour
- Determining the "best" test cases
- Reporting the test results
- Unclear when to stop the test effort

ISTQB Foundation Home. December 26th 2016. "Chapter 4: Test Design Techniques". [Online]: https://istqbinformation.wordpress.com/2016/12/23/chapter-4-test-design-techniques/

Exercise

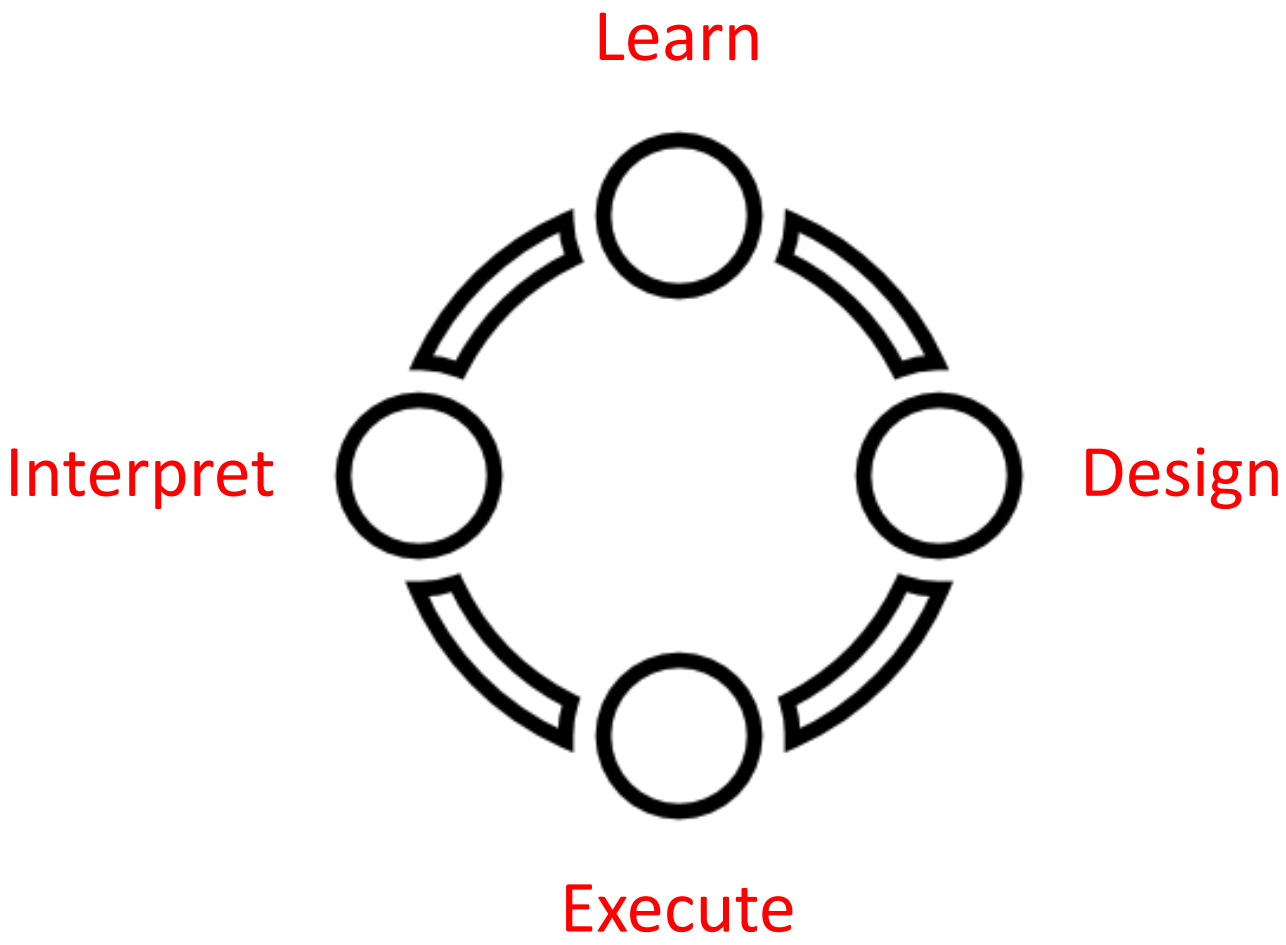
"Exploratory testing uncovers bugs that would otherwise be ignored by other testing techniques".

Why do you think that is? (Justify your answer)

Exploratory Testing

Activities (according to C. Kaner)

Activities (according to C. Kaner)



Kaner, C. (2008)

Learn | Design | Execute | Interpret

Learn

- What to test? How to test? What is the problem?

Design

- Not equal to scripting!

Execute

- Executing the actual tests and collecting the result

Interpret

- What does the resulting behaviour tell us?
  - About the product | About the manner in which we are testing the product

Kaner, C. (2008)

Learn | Design | Execute | Interpret

- Project context
- Risks
- Study similar products + their history
- Review documentation / written sources

Kaner, C. (2008)

Learn | Design | Execute | Interpret

- Outline areas of interest
  - Based on the "learning" activity
- Create a rough outline of the test effort
- Derive testing techniques from test ideas
- Decide on the testing tools

Kaner, C. (2008)

Learn | Design | Execute | Interpret

- Execute test cases on the fly
- Vary activities to stay engaged
- Tester is free to adhere or discard the plan (design)
- Collect interesting results in a somewhat orderly fashion

Kaner, C. (2008)

Learn | Design | Execute | Interpret

- Analyse the main takeaways from the execution activity
  - Bugs?
  - Interesting results?

Other things to consider

  - Consistency (similar products)
  - Adherence to specifications
  - Adherence to claims / expectations

- Assess need for further testing
  - Are you still finding bugs / interesting results?

Kaner, C. (2008)

Exercise

What do you consider to be the most valuable points or activities in the LEARN | DESIGN | EXECUTE | INTERPRET cycle for effective exploratory testing? (Justify your answer)

Exploratory Testing

In practice

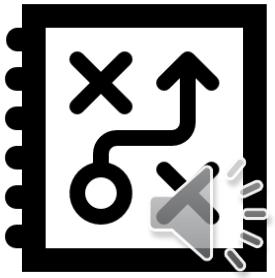
Heuristics

- Simple guidelines
  - "Rule of thumb"
- Spark associations
  - "This reminds me of…"
- Contextual alignment
  - "Okay, se we have…"
- Based on experience
  - "I've seen this before"

CRUD

CREATE | READ | UPDATE | DELETE ←→ ZERO | ONE | MANY

- Zero / No data
  - Check how the system behaves without data
- One
  - Check behaviour when deleting the only record
- Many
  - Check behaviour when updating several records



CONORFI. (2017)

Personas

Different users interact with the software in different ways

- Create various personas
- Provides insight into test case design

- Expert / Power user
  - Shortcuts, fast navigation, etc.

- Impatient user
  - Repeatedly clicking when experiencing slow response times

CONORFI. (2017)

Exercise

Create a persona representing an older person and their envisioned traits when interacting with an online banking app.

Derive test cases from these associations (heuristics).

Nouns and Verbs

Identify nouns and verbs characterising the system

- From experience / own knowledge / requirements specification

Use said nouns and verbs to envision test cases

- Guide on-the-fly test effort

- E-mail
  - Verbs: Create, edit, send, forward, copy, delete, move
  - Nouns: Messages, attachments, contacts, accounts, folders, flags

CONORFI. (2017)

Exercise

Use the "nouns and verbs" technique to identify suitable test cases for testing a website selling concert tickets.

Summary

- Dynamic, experience-based testing technique
- Overcome the limitations of scripted testing
- Investigative work
  - Thinking both outside and inside the box
- Encourages creativity and continuous learning
- Dependent on tester's skill and knowledge
- Effective testing using heuristics

Further Reading/Challenges

Heuristics Cheat Sheet

Heuristics

Variable Analysis	Identify anything whose value can change. Variables can be obvious, subtle, or hidden.
Touch Points	Identify any public or private interface that provides visibility or control. Provides places to provoke, monitor, and verify the system.
Boundaries	Approaching the Boundary *(almost too big, almost too small)*, At the Boundary
Goldilocks	Too Big, Too Small, Just Right
CRUD	Create, Read, Update, Delete
Follow the Data	Perform a sequence of actions involving data, verifying the data integrity at each step. *(Example: Enter → Search → Report → Export → Import → Update → View)*
Configurations	Varying the variables related to configuration *(Screen Resolution; Network Speed, Latency, Signal Strength; Memory; Disk Availability;* **Count** *heuristic applied to any peripheral such as 0, 1, Many Monitors, Mice, or Printers)*
Interruptions	Log Off, Shut Down, Reboot, Kill Process, Disconnect, Hibernate, Timeout, Cancel
Starvation	CPU, Memory, Network, or Disk at maximum capacity
Position	Beginning, Middle, End *(Edit at the beginning of the line, middle of the line, end of the line)*
Selection	Some, None, All *(Some permissions, No permissions, All permissions)*
Count	0, 1, Many *(0 transactions, 1 transactions, Many simultaneous transactions)*
Multi-User	Simultaneous create, update, delete from two accounts or same account logged in twice.
Flood	Multiple simultaneous transactions or requests flooding the queue.
Dependencies	Identify "has a" relationships *(a Customer has an Invoice; an Invoice has multiple Line Items).* Apply **CRUD**, **Count**, **Position**, and/or **Selection** heuristics *(Customer has 0, 1, many Invoices; Invoice has 0, 1, many Line Items; Delete last Line Item then Read; Update first Line Item; Some, None, All Line Items are taxable; Delete Customer with 0, 1, Many Invoices)*
Constraints	Violate constraints *(leave required fields blank, enter invalid combinations in dependent fields, enter duplicate IDs or names).* Apply with the **Input Method** heuristic.
Input Method	Typing, Copy/Paste, Import, Drag/Drop, Various Interfaces *(GUI v. API)*
Sequences	Vary Order of Operations ▪ Undo/Redo ▪ Reverse ▪ Combine ▪ Invert ▪ Simultaneous
Sorting	Alpha v. Numeric ▪ Across Multiple Pages
State Analysis	Identify states and events/transitions, then represent them in a picture or table. Works with the **Sequences** and **Interruption** heuristics.
Map Making	Identify a "base" or "home" state. Pick a direction and take one step. Return to base. Repeat.
Users & Scenarios	Use Cases, Soap Operas, Personae, Extreme Personalities

Frameworks

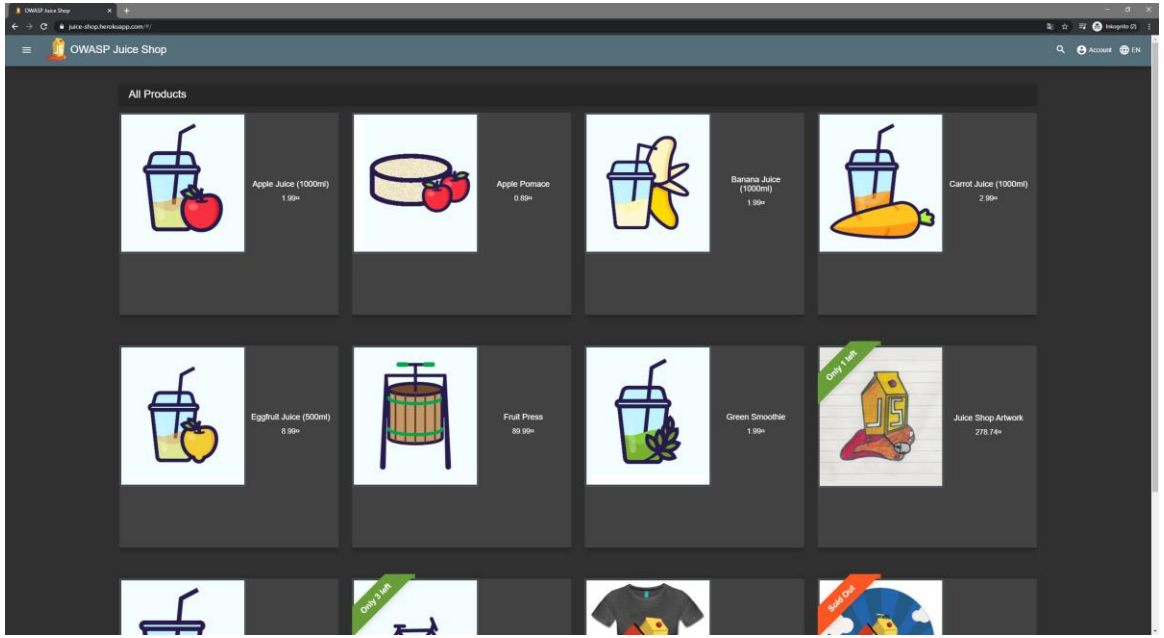
Judgment	Inconsistencies, Absences, and Extras with respect to Internal, External – Specific, or External – Cultural reference points. (James Lyndsay, Workroom Productions)
Observations	Input/Output/Linkage (James Lyndsay, Workroom Productions)
Flow	Input/Processing/Output
Requirements	Users/Functions/Attributes/Constraints (Gause & Weinberg *Exploring Requirements*)
Nouns & Verbs	The objects or data in the system and the ways in which the system manipulates it. Also, Adjectives (attributes) such as Visible, Identical, Verbose and Adverbs (action descriptors) such as Quickly, Slowly, Repeatedly, Precisely, Randomly. Good for creating random scenarios.
Deming's Cycle	Plan, Do, Check, Act

Hendrickson, Lyndsay, Emergy. (2006)

OWASP Juice Shop

For those wanting a security-related exploratory testing challenge.

Read more here: https://owasp.org/www-project-juice-shop/



https://juice-shop.herokuapp.com/#/

Resources

- ISTQB Foundation Level Syllabus. [Online]:
- TestMatick. June 4th 2019. "Usage of heuristics and mnemonics in software testing". [Online]: https://testmatick.com/usage-of-heuristics-and-mnemonics-in-software-testing/
- ISTQB Foundation Home. December 26th 2016. "Chapter 4: Test Design Techniques". [Online]: https://istqbinformation.wordpress.com/2016/12/23/chapter-4-test-design-techniques/
- CONORFI. May 7th 2017. "Why use Heuristics in Software Testing?". [Online]: https://conorfi.com/software-testing/software_testing_heuristics/
- Hendrickson, E.; Lyndsay, J.; Emery, D. 2006. "Test Heuristics Cheat Sheet". Quality Tree Software. [Online]: http://testobsessed.com/wp-content/uploads/2011/04/testheuristicscheatsheetv1.pdf
- Kaner, C. (2008). "A Tutorial in Exploratory Testing". [Online]: http://www.kaner.com/pdfs/QAIExploring.pdf