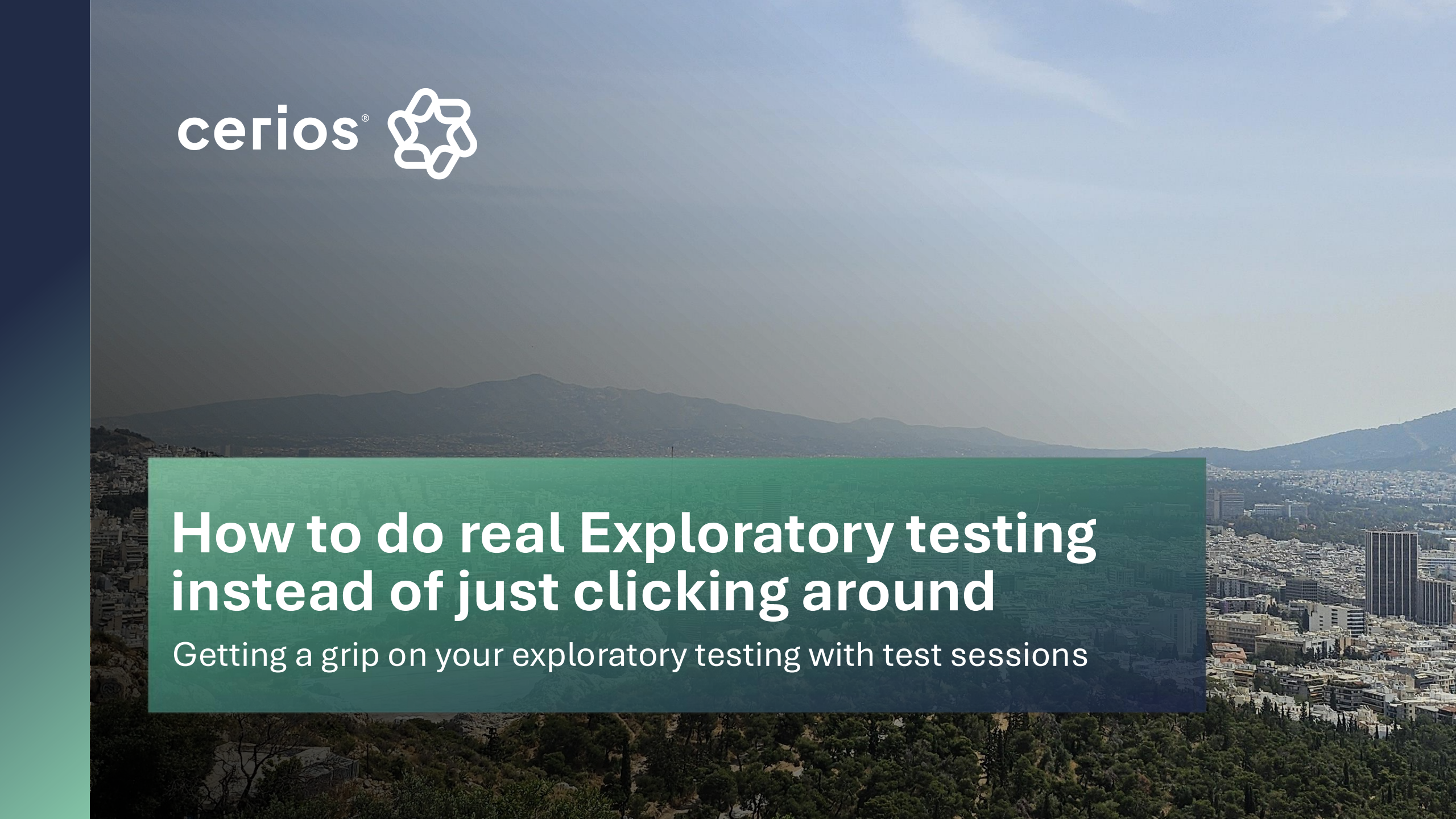
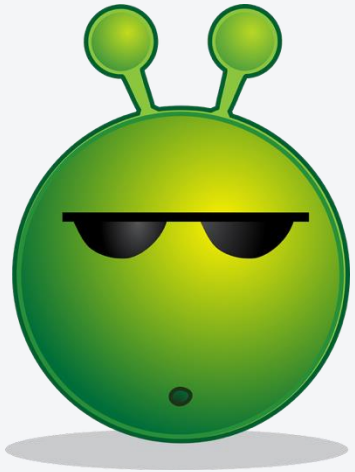


How to do real Exploratory testing instead of just clicking around

Getting a grip on your exploratory testing with test sessions

What is exploratory testing?



Definition

“Exploratory software testing is a style of software testing that emphasizes the personal freedom and responsibility of the individual tester to continually optimize the value of her work by ***treating test-related learning, test design, test execution, and test result interpretation as mutually supportive activities*** that run in ***parallel*** throughout the project.” – Cem Kaner 2006

1. Introduction
2. Explore the software
3. Exploratory Testing
4. Testing tours
5. Test charter basics
6. Let’s go deeper
7. Evaluate and learn

Rob van Steenbergen

- Release date 15th of June 1972
- Since 1996 in testing
- World champion software testing
- Movies, music, drawing, robots, learning, walking, testing, …

- Working for Cerios.com
- Trainer for “Agile united” - united-certifications.org
  - Certified Practitioner in Agile Testing 2.0
  - Certified Specialist in Agile Testing

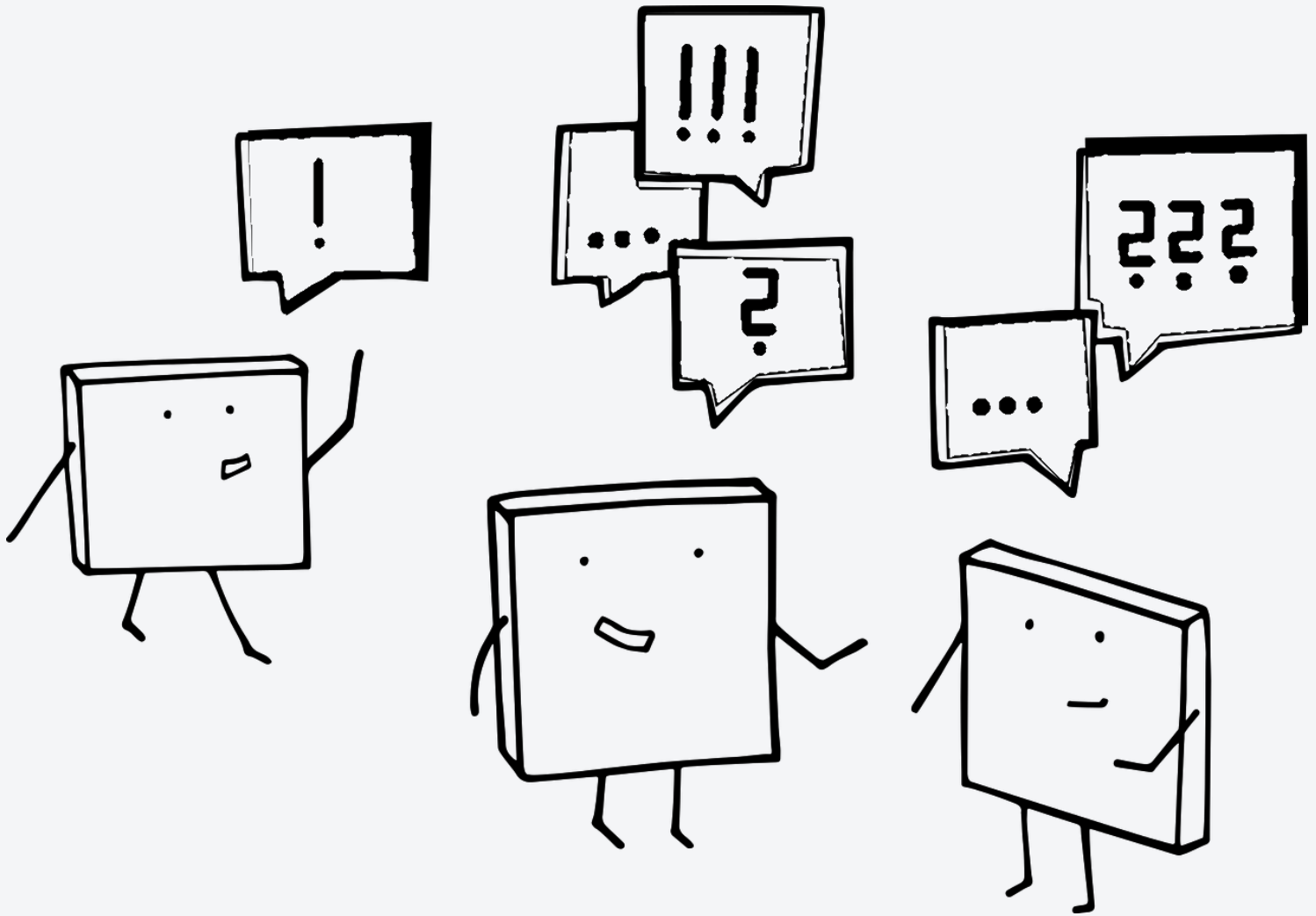
Explore the software

Start testing – The Pulper!

http://bit.ly/3PlO4LA

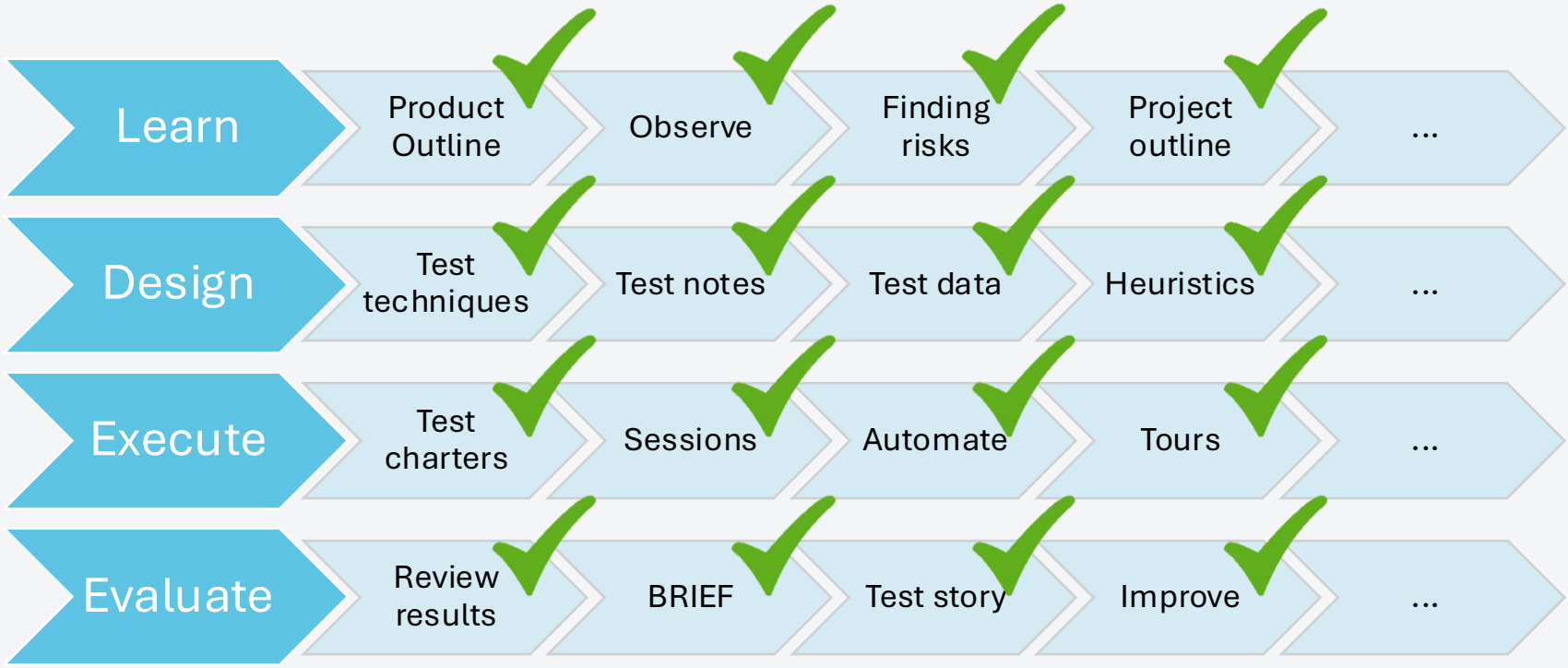
Created by Alan Richardson (aka The Evil Tester)

10 minutes

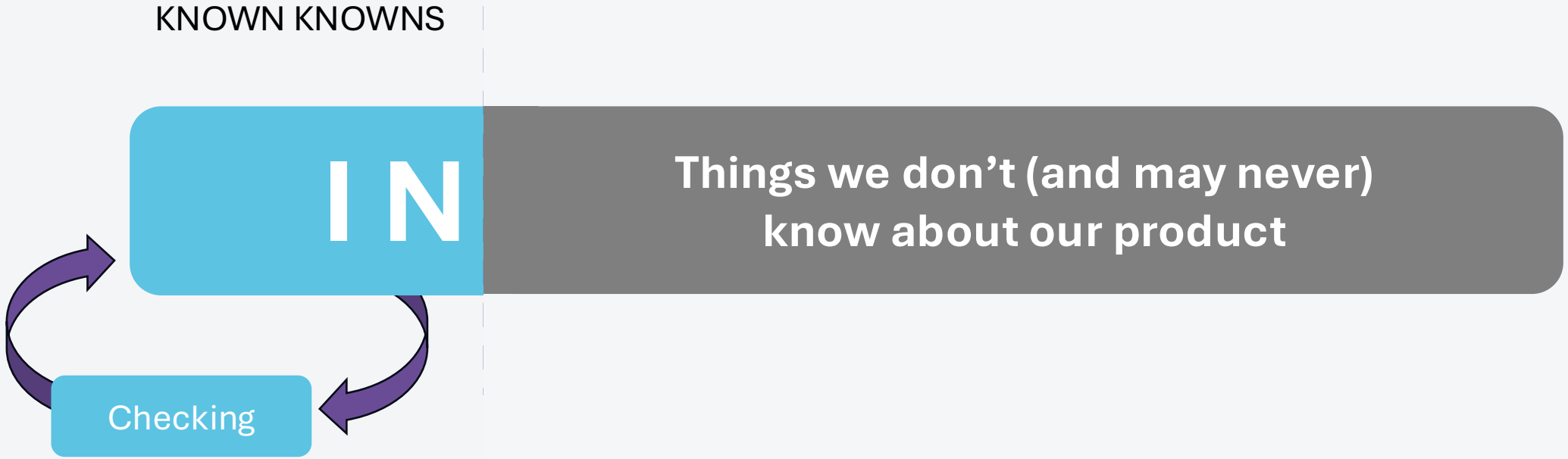


So: Exploratory testing

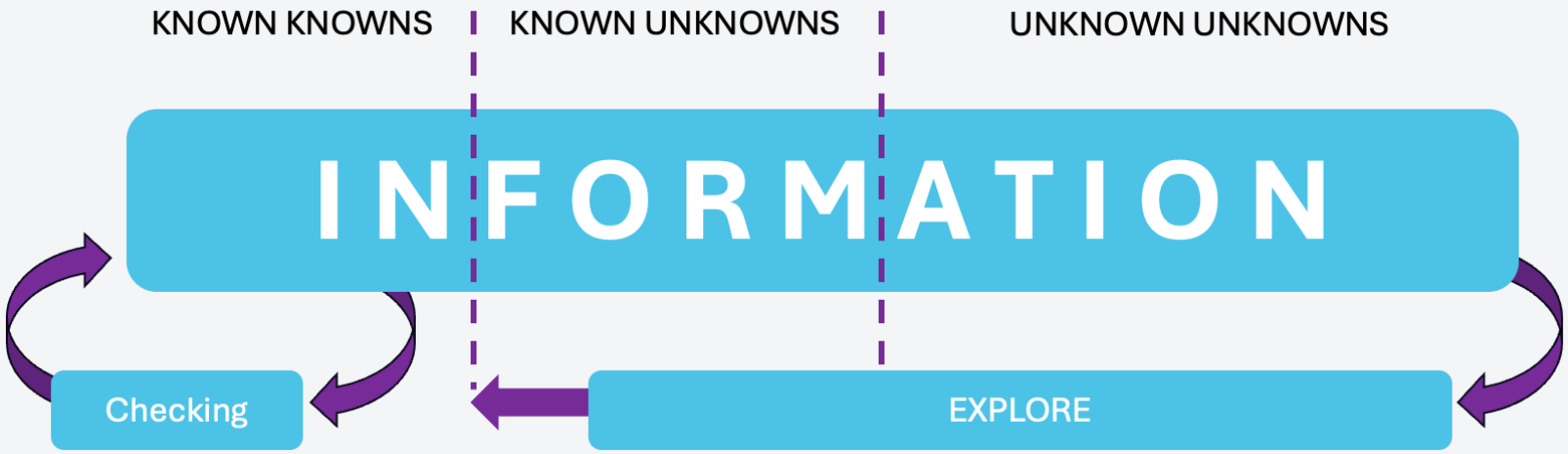
What is Exploratory testing?



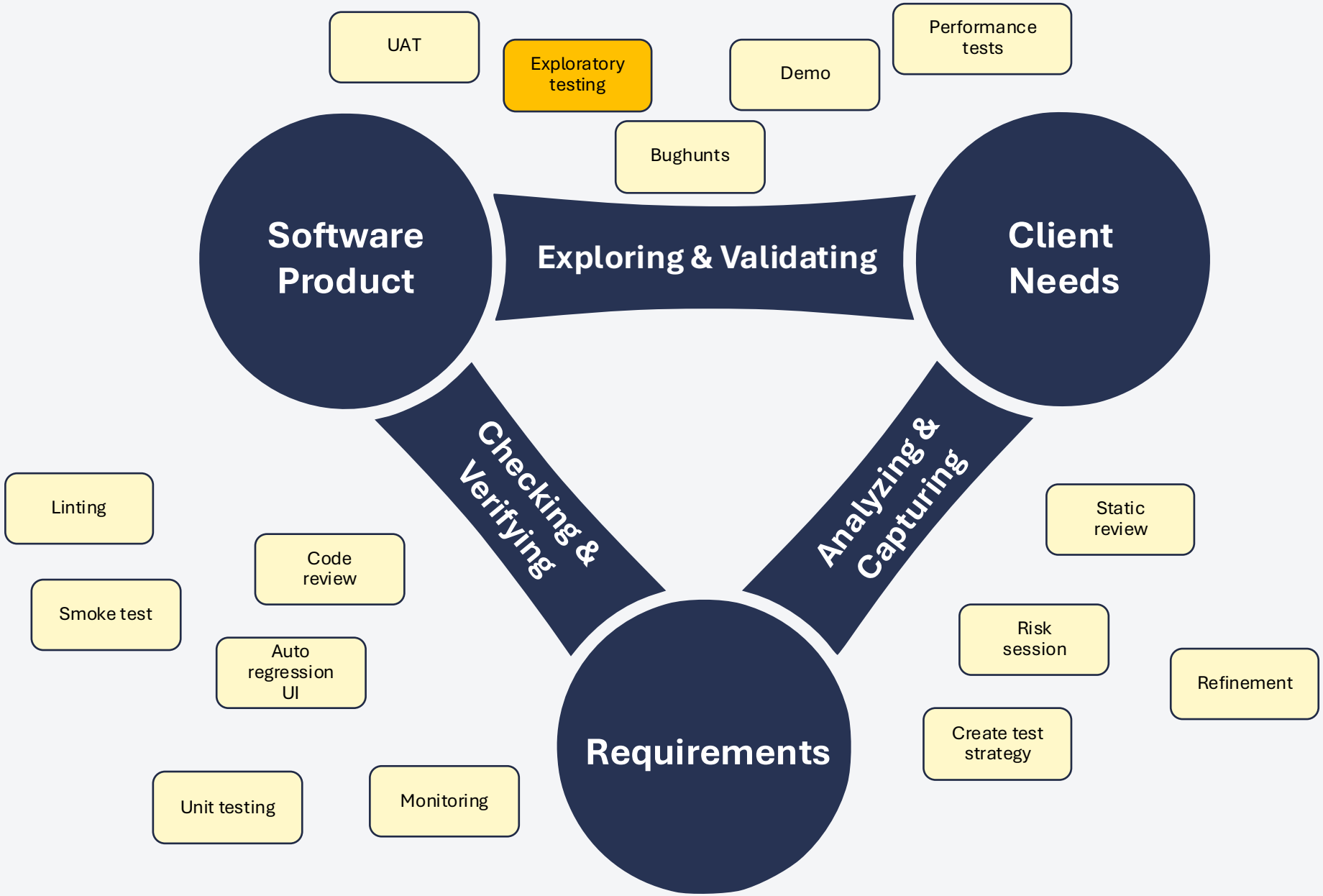
Exploratory testing: why?



Exploratory testing: why?



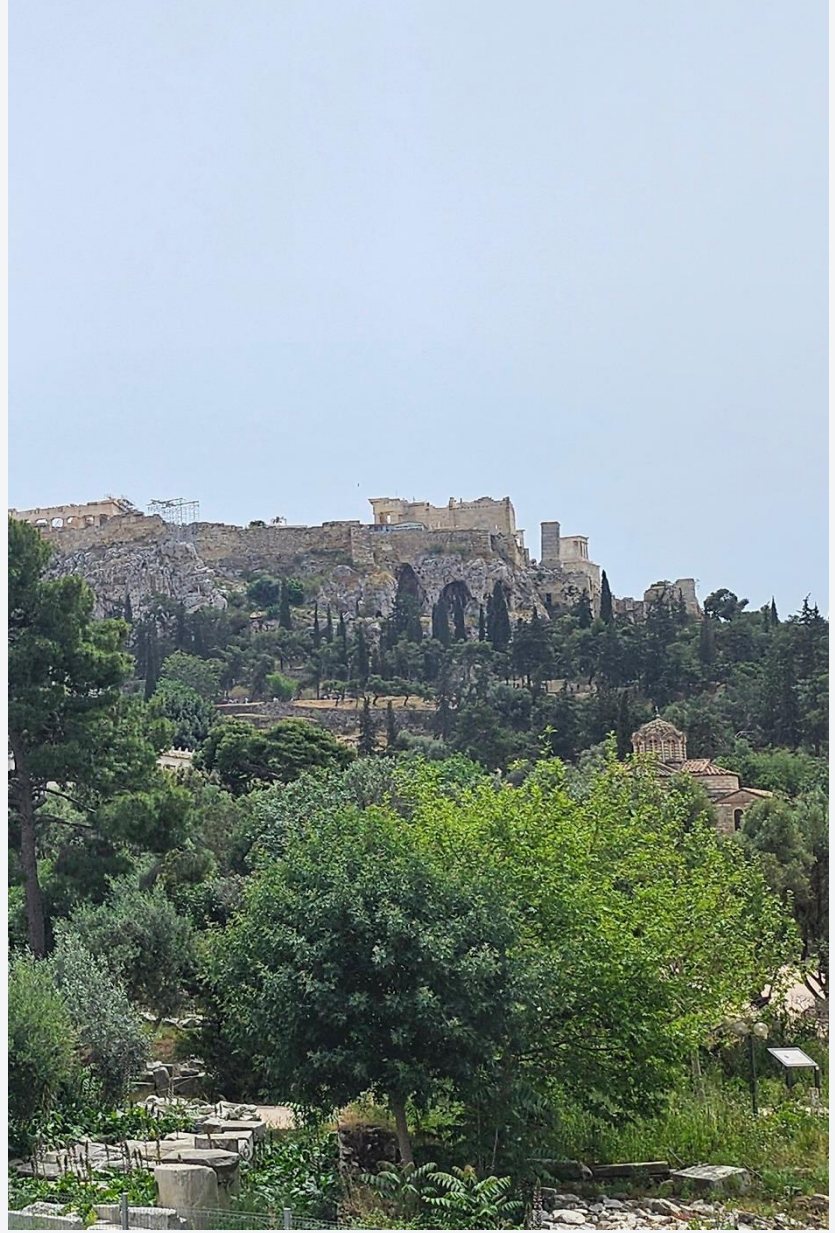
- Checking only confirms what we know
- Checking informs exploring
- Exploring uncovers unknown Information
- Exploring brings unknowns to the known domain


UAT
Exploratory testing
Demo
Performance tests
Bughunts
Software Product
Exploring & Validating
Client Needs
Checking & Verifying
Analyzing & Capturing
Linting
Code review
Smoke test
Auto regression UI
Unit testing
Monitoring
Requirements
Static review
Risk session
Refinement
Create test strategy

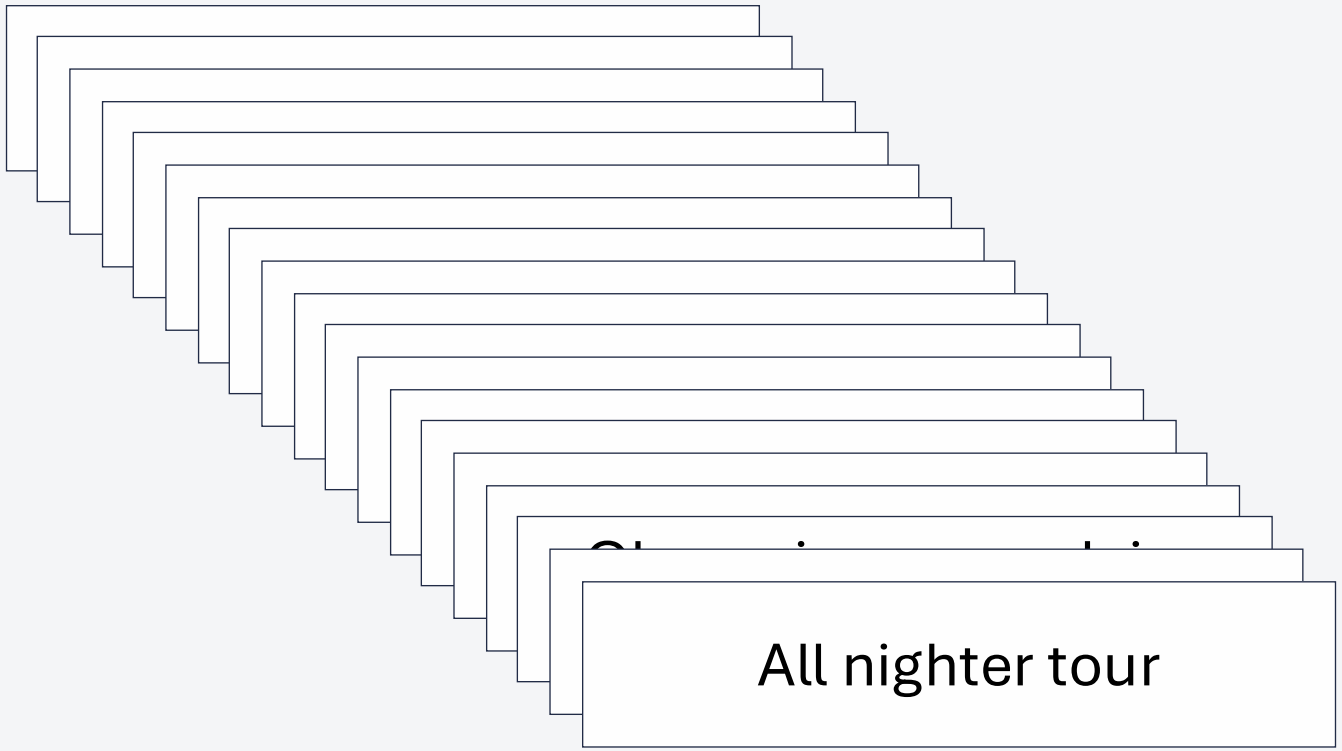
Testing tours

Testing tour

Testing is organized around a specific theme or goal, similar to a tourist tour. As a tester, you are a tourist exploring a city (the software) using a route (the tour).



Different testing tours



Excercise

- Choose a testing tour
- Perform a test
- Discuss what you did in teams

App: **http://bit.ly/3PlO4LA**

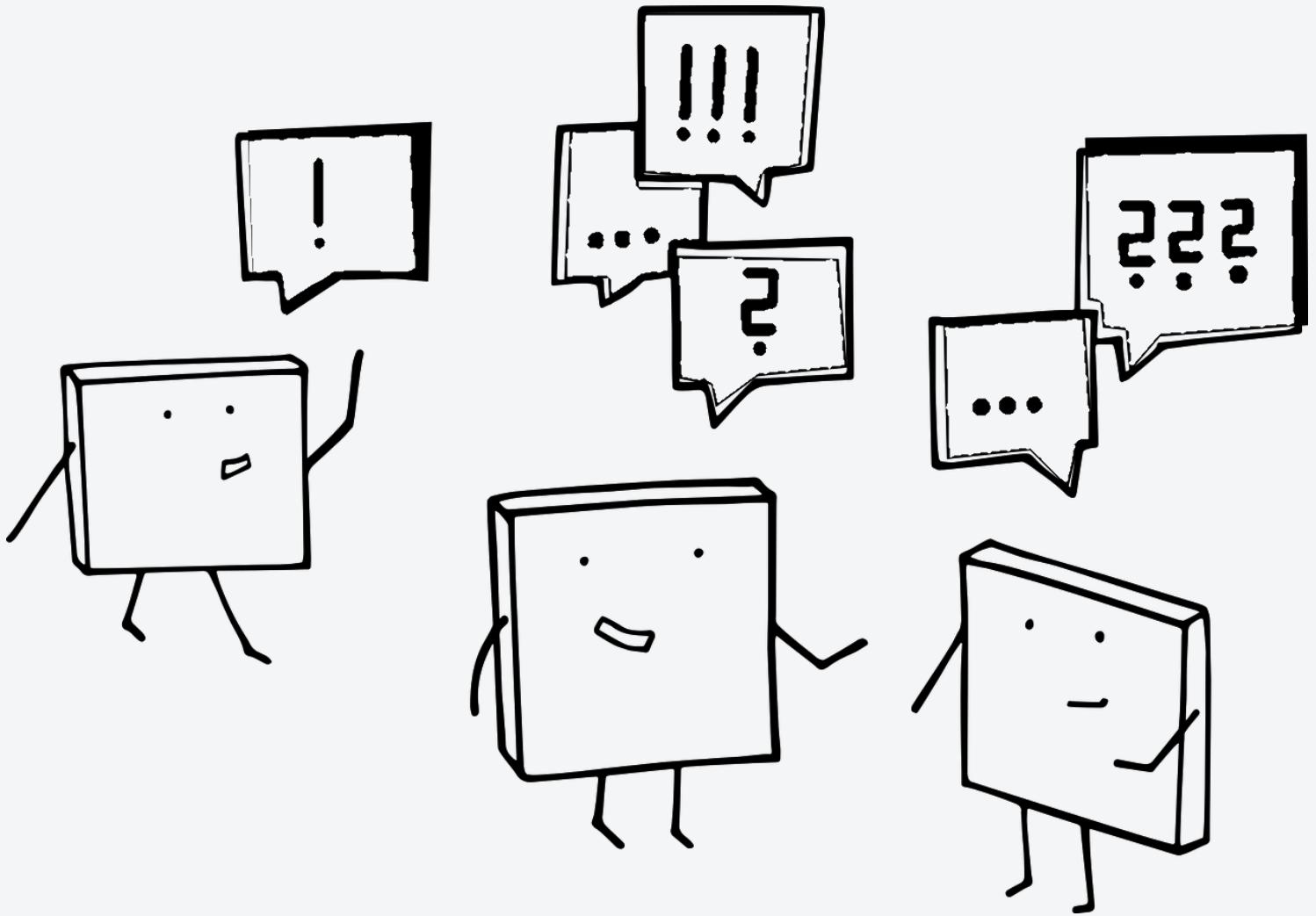
20 minutes

Michael Bolton on testing tours: https://bit.ly/testing_tours

Of Testing Tours and Dashboards

Back in the 1980s and 1990s, Quarterdeck Office Systems (later Quarterdeck Corporation)—a company...

bit.ly



Test charter basics

Watch out when exploring

- It’s easy to “wander off” and get lost
  - You should have a specific goal
  - You will need to prepare
- Bring a survival kit
  - the right tools, techniques

Your **goal and plan** for your "exploratory test" = **a test charter**

Define your goal: Charter template

- Target
  - Where are you exploring?
- Resources
  - What resources do you need?
- Information
  - What kind of information do you want to discover?

Explore
<target>

With
<resources>

To discover
<information>

Exercise: create test charters

- Team up!
  - Create at least three test charters
  - Discuss those in your team

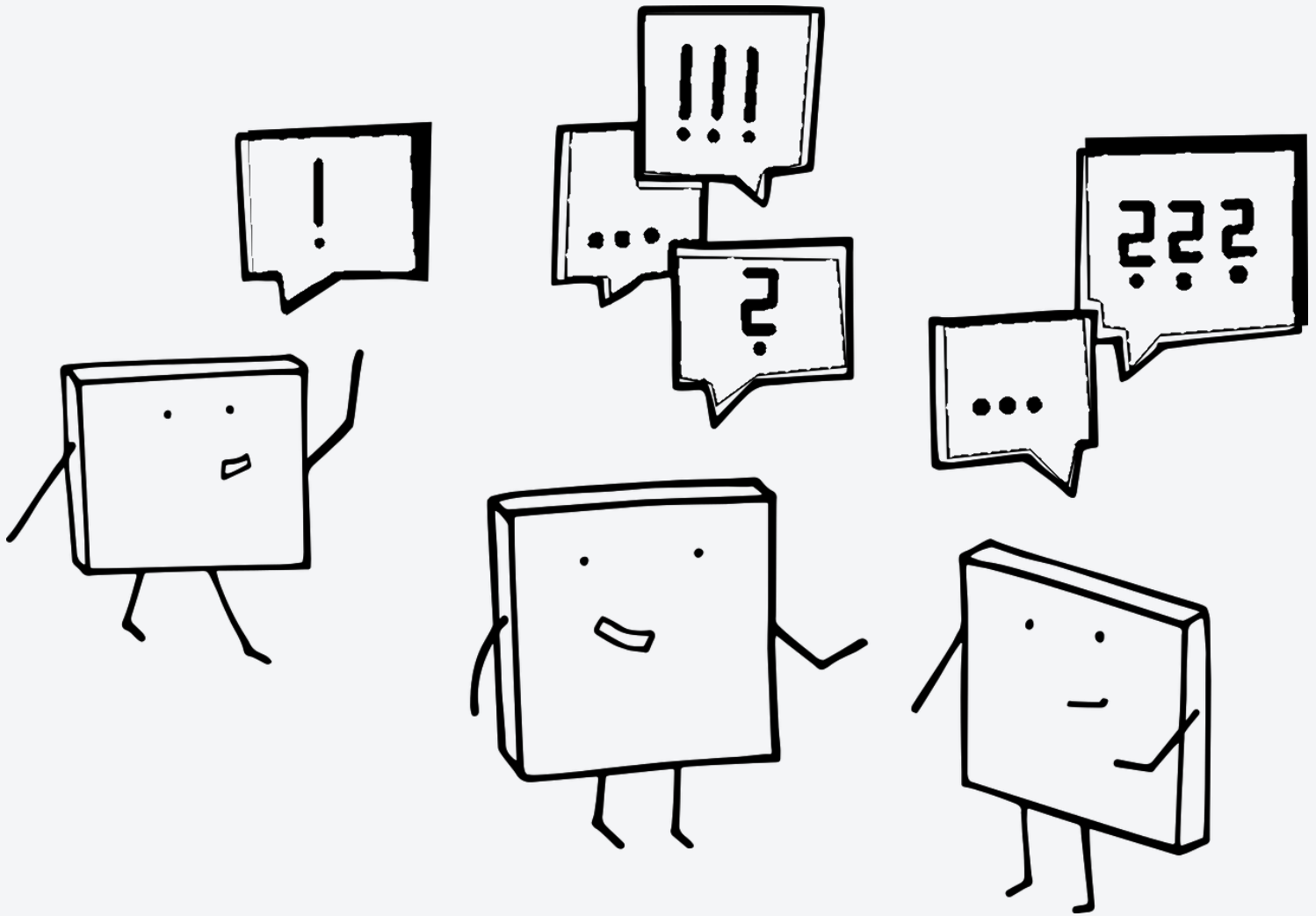
- **Application url: http://bit.ly/3PlO4LA**

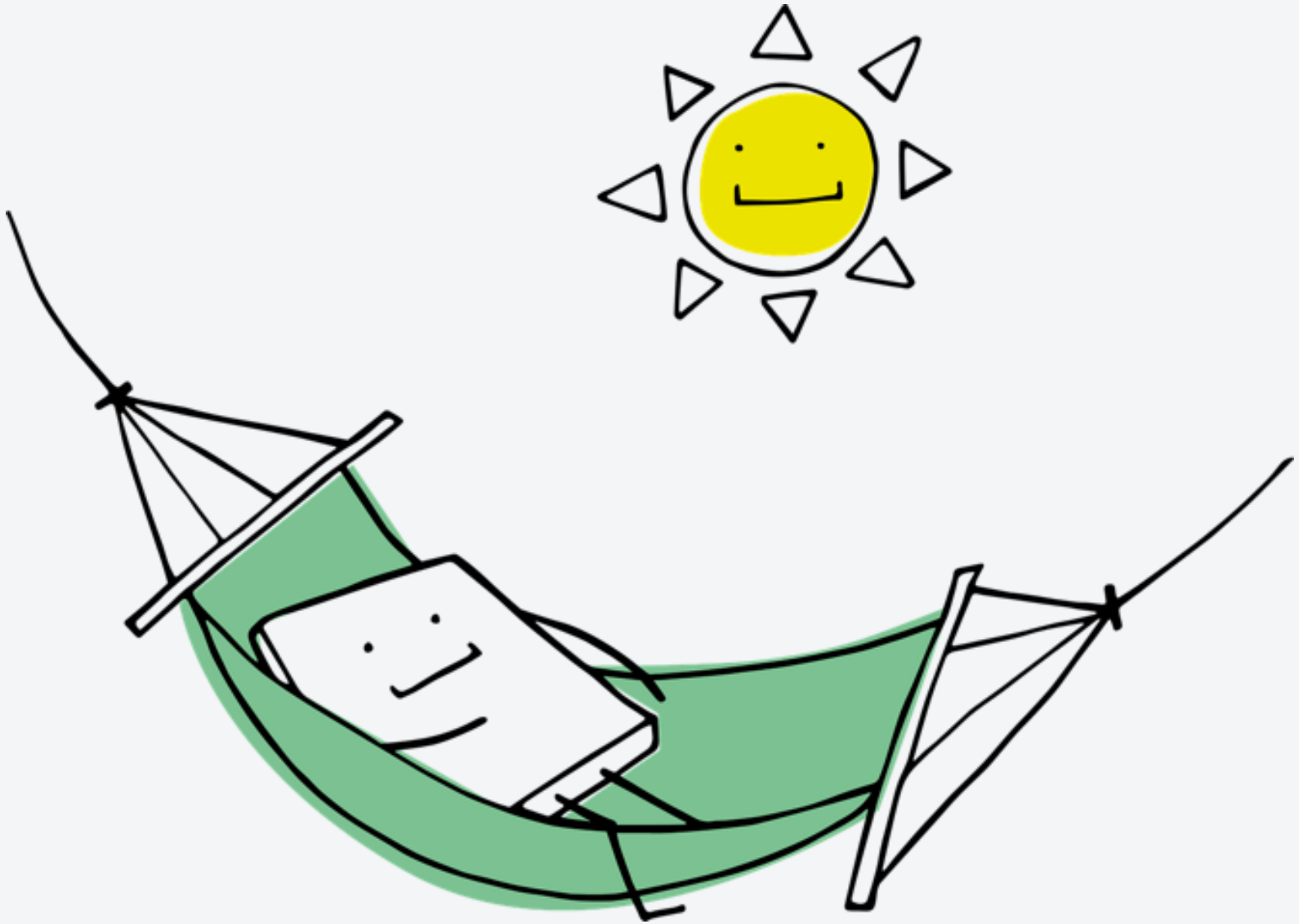
20 minutes

Explore
<target>

With
<resources>

To discover
<information>





Let’s go deeper & test

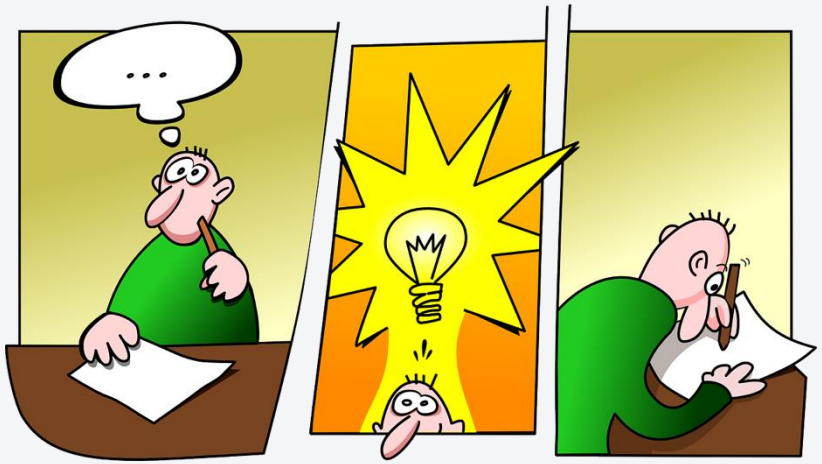
Why use test charters

- Helps you with critical thinking
- Provides direction and structure
- Enables interaction and cooperation
- Encourage continuous improvement
- Shows the value of testing

Image from the book: Explore it!

Inspiration for test charters

- Requirements
- Implicit expectations
- Existing Artifacts
- Product risks
- Quality characteristics (ISO-25010)
- Personas
- Test tours
- Heuristics



Inspiration for test charters

- Use bug taxonomies
- API specifications
- Revisit classic test techniques
- The product itself
- Check the project context
- Connect with stakeholders
- External
- Et cetera...

How to continue?

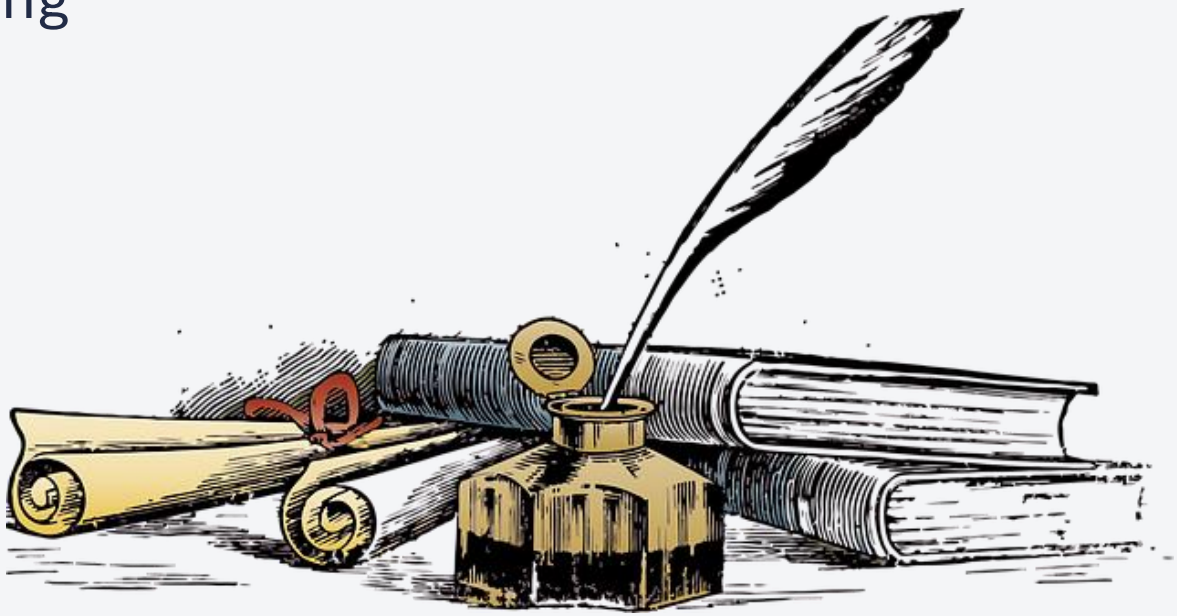
To get a limitation on content and to be efficient, you limit a test session timewise to a period of about 1 - 2 hours

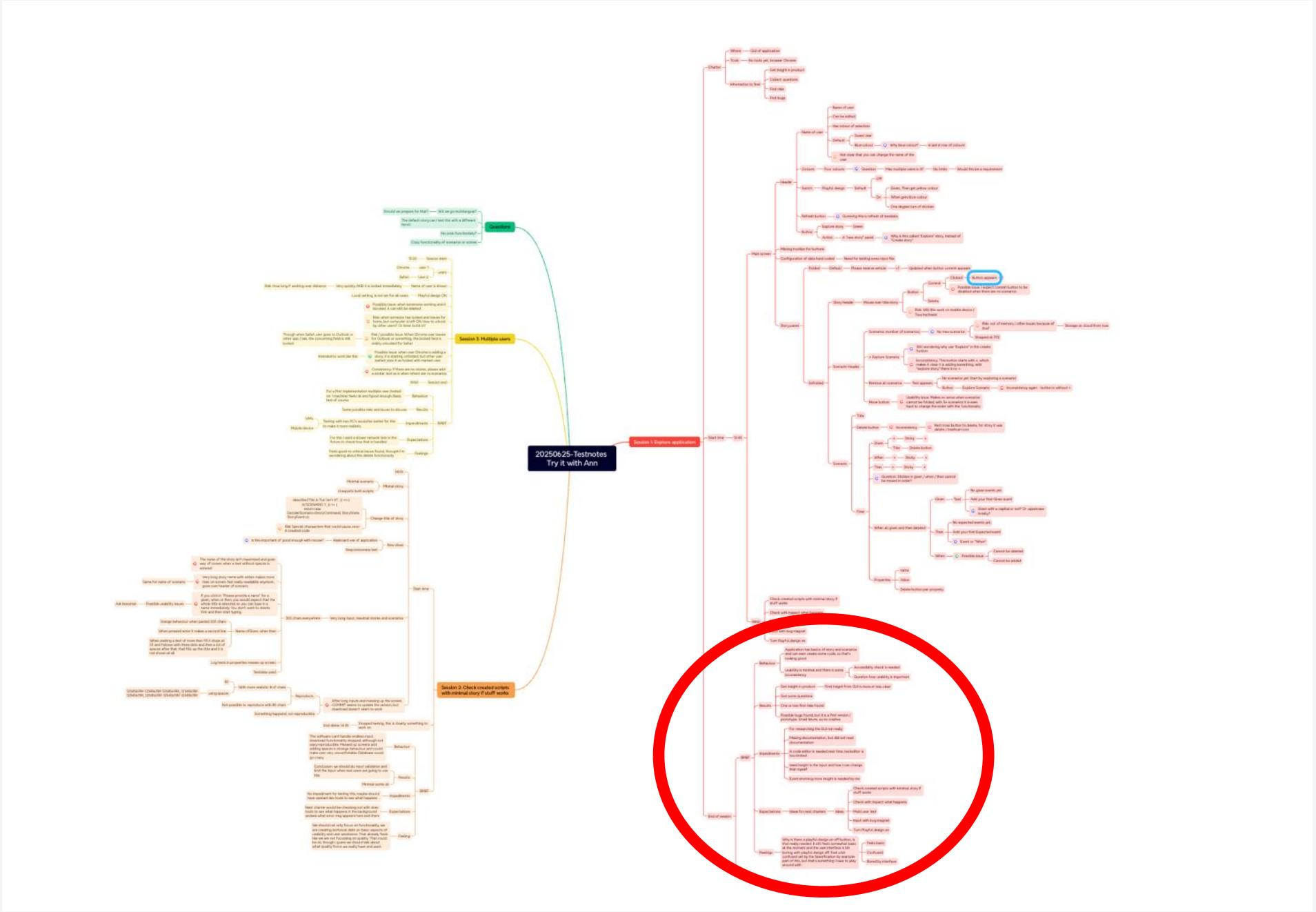
- Depending on your attention span
- With a team to get back together
- Focused on your goal/mission
- Continuously uninterrupted

Create test notes

Register the following elements during your execution

- What you are doing
- The relevant test data you are using
- Potential bugs
- Questions you have
- New product risks
- Off charter topics




20250625-Testnotes
Try it with Ann
Questions
Session 1: Multiple users
Session 2: Check created scripts with minimal story if stuff works
Session 1: Explore application

The application and my test story

Application behavior

- The application has basic functionality and I can create scenarios.
- User-friendliness does not seem optimal (inconsistency)

What are my results?

- Gained insight into the product, although there are still a few questions I want to address later
- Found one or two initial risks and potential bugs: no crashes

- **Obstacles**
- Missing documentation
- A code editor is needed next time to interpret the code
- Insight into the input (import) is needed

The application and my test story

Expectations

- Accessibility test / research is needed
- Use Devtools to see what happens
- Test with multiple users

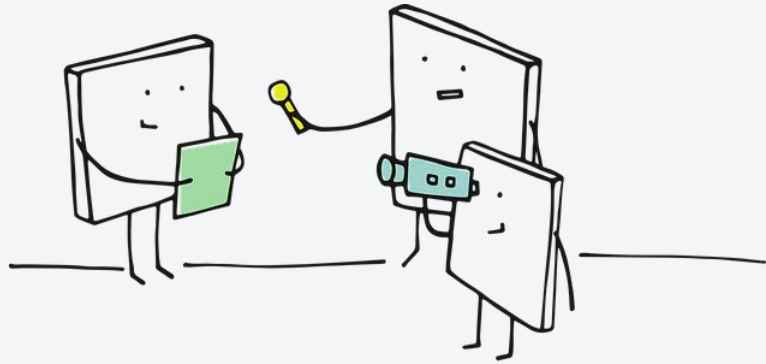
Feeling

- It still feels a bit basic right now
- Bored by the user interface, which is very dull

Debriefing

Adopt the following elements (BRIEF)

- Behavior — how did the software behave
- Results — what has been achieved
- Impediments — what stood in your way
- Expectations — what work has still to be done
- Feelings — what does my intuition tell me



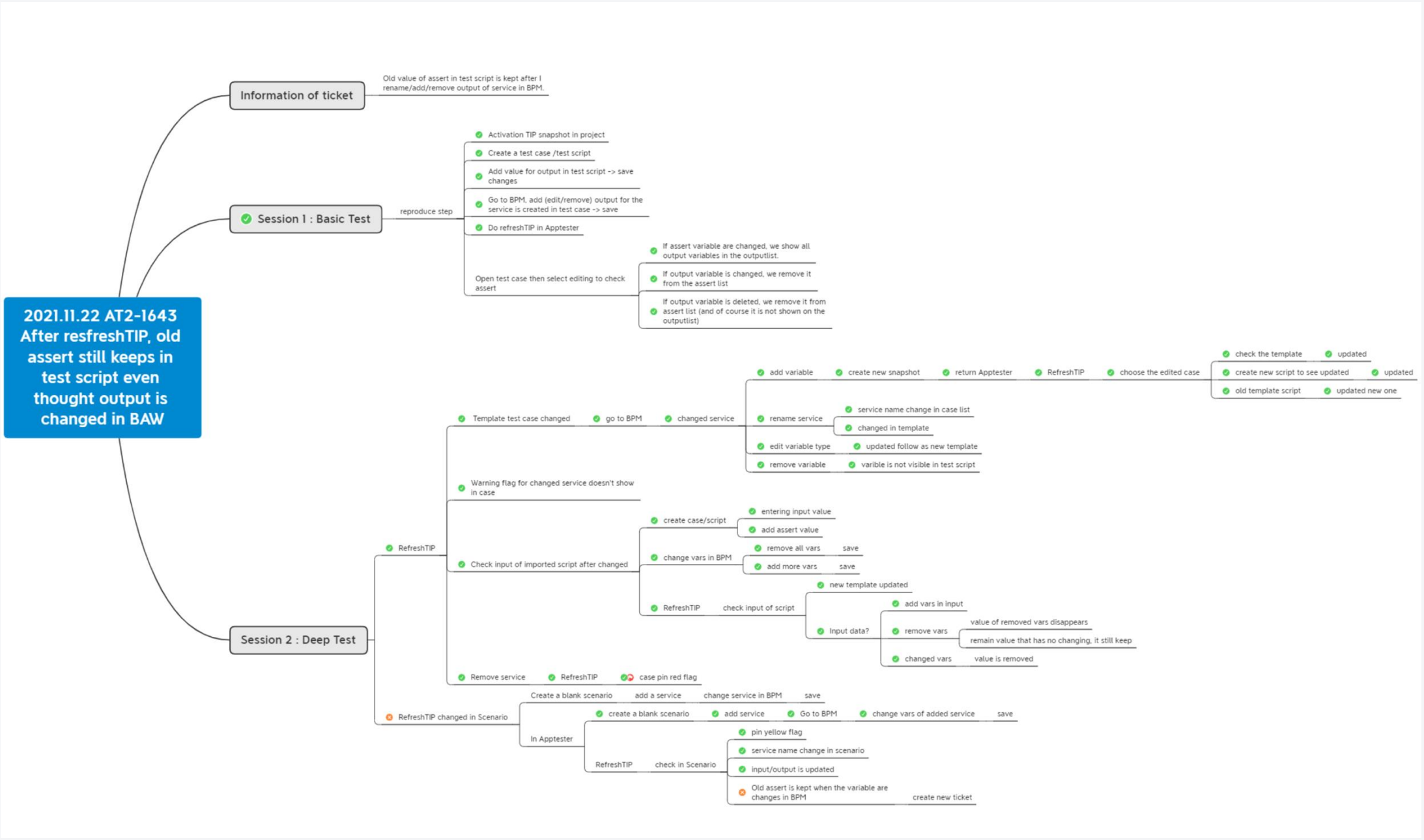
Exercise: Test with charters

- Write test notes
- Discuss findings with BRIEF
  - Behaviour - how did the software behave
  - Results - what has been achieved
  - Impediments - what stood in your way
  - Expectation - what work has still to be done
  - Feelings - what does my intuition tell me

20 minutes

Examples

	A	B
1	**Testcharter 3D-printer Model 3**	
2	**Missie**	Error handling material input until print starts
3	**Doel/Plan**	There are many reports of problems with error handling when importing materials. There seems to be little to no error handling. We focus on undesirable situations when entering parameters in order to trigger error handling as much as possible
4	**Time**	2 hours max
5	**Notes**	Starting point is a printer with no material entered
6		Printer started up. Printer asks for material to be entered. Tried to continue without material. That is not possible.
7		Material 1 entered (PLA). Material 2 not entered (is not required). Indicated that material has been entered. This is accepted. Strange. See what happens if we are going to print with 2 materials
8		Print started with 2 materials (the 2 color frog). Print refuses to start because material 2 has not been entered. That is in line with expectations. This should be captured during setup …
9		Remove material 1 and replace this with material 3 (ABS). Choose a temperature of 350 degrees (much too hot for ABS). I now expect an error message or warning, but it is not coming.
10		Start a print (robot) and see what the printer will do with the much too high temperature. The printer starts the print. The plastic burns in the printhead and the material gets stuck. The printer overheats because the heat cannot dissipate. Error message 'temperature too high' appears on the screen with the option <ok>. Lots of smoke and smells from the printer. I confirm the ok with the expectation that the printer will turn off. This is not happening. It stays on and returns with the error message. Again confirmed, but the loop continues. Finally pulled the plug.
11		……
12	**Debriefing**	
13	Behaviour	Error handling is clearly not ok, even in such a way that the print head became defective
14	Results	Confirmation that this must be worked on. Half of the company stood around the smoking printer
15	Impediments	Only 1 printer available, which slows down because starting a print takes quite a long time. Multiple printers to test in parallel would be handy.
16	Expectation	Error handling during the rest of the process will also have to be tested, and this is expected to be missing. The input side is also certainly not ready because many variables have not been performed in this test.
17	Feelings	I have had a double feeling about the test. On the one hand, it is pleased that the problems have emerged before the printer goes to customers. On the other hand, it is surprising that this has never been discovered before because the consequential damage can be considerable. It has made the organization realize that a good ET is important


2021.11.22 AT2-1643 After resfreshTIP, old assert still keeps in test script even thought output is changed in BAW
Information of ticket
Old value of assert in test script is kept after I rename/add/remove output of service in BPM.
Session 1 : Basic Test
reproduce step
Activation TIP snapshot in project
Create a test case /test script
Add value for output in test script -> save changes
Go to BPM, add (edit/remove) output for the service is created in test case -> save
Do refreshTIP in Apptester
Open test case then select editing to check assert
If assert variable are changed, we show all output variables in the outputlist.
If output variable is changed, we remove it from the assert list
If output variable is deleted, we remove it from assert list (and of course it is not shown on the outputlist)
Session 2 : Deep Test
RefreshTIP
Template test case changed
go to BPM
changed service
add variable
create new snapshot
return Apptester
RefreshTIP
choose the edited case
check the template
updated
create new script to see updated
updated
old template script
updated new one
rename service
service name change in case list
changed in template
edit variable type
updated follow as new template
remove variable
varible is not visible in test script
Warning flag for changed service doesn't show in case
Check input of imported script after changed
create case/script
entering input value
add assert value
change vars in BPM
remove all vars
save
add more vars
save
RefreshTIP
check input of script
new template updated
Input data?
add vars in input
remove vars
value of removed vars disappears
remain value that has no changing, it still keep
changed vars
value is removed
Remove service
RefreshTIP
case pin red flag
RefreshTIP changed in Scenario
Create a blank scenario
add a service
change service in BPM
save
In Apptester
create a blank scenario
add service
Go to BPM
change vars of added service
save
RefreshTIP
check in Scenario
pin yellow flag
service name change in scenario
input/output is updated
Old assert is kept when the variable are changes in BPM
create new ticket

UAT / bug hunts – information and goal

Test charter: I am testing <goal> with <tools, techniques, other resources> to discover <information>
Name tester(s):
Contact person project: << To discuss testing and who to send this form. Name and email address of contact person >>
Session duration Date: Duration: Short (30-60 min) / Normal (60-90 min) / Long (90-120 min)
Objective: Software / functionality - Functionality - Components **Tools, recourses:** - What do you need - List **Test data preparation / needed:** - … - ….
Remark

UAT / bug hunts – Test Notes

Test notes

Notes:

<< Indicate in this section how the actual test was performed and which steps you actually took.

Describe simply the actions that were taken during the test

- You could present this as a checklist
- Describe what you observed and what stood out

If there is anything that needs to be reported and you need a screenshot, you can also place it here. >>>

UAT / bug hunts – BRIEF & bugs

Behavior of the software
<< Did the software behave as you expected? >>

Results
<< Were you able to achieve the correct results? >>

Blockers
<< Were there any issues that prevented you from proceeding, and did you need to apply a workaround, for example? >>

Expectations
<< What else should be done to improve the part of the software that was tested? >>

Intuition
<< What is your feeling after performing these tests? Satisfied, happy, or disappointed? Do you think there's anything else you would like to check based on this? >>

Bugs

<< Describe the bugs found, the nature of each bug, at what point in the test they appeared, what deviations occurred with each bug, and the steps to reproduce them. This would mostly include information about bugs that have been recorded through screen captures, screenshots, text logs, etc. >>

Describe the bug as thoroughly as possible and try to include the following elements:
Clear title: What exactly went wrong

Description

- Test data used: specific account, specific BSN number, any other details you want to

Evaluate and learn

Evaluate and learn

- Change teams, go to another table
  - Discuss the Pro’s and cons
- Stickies -> on the walls
  - Anything you learned
  - Tips and ideas
  - All other things you want to mention

Thanks!

www.linkedin.com/company/cerios/

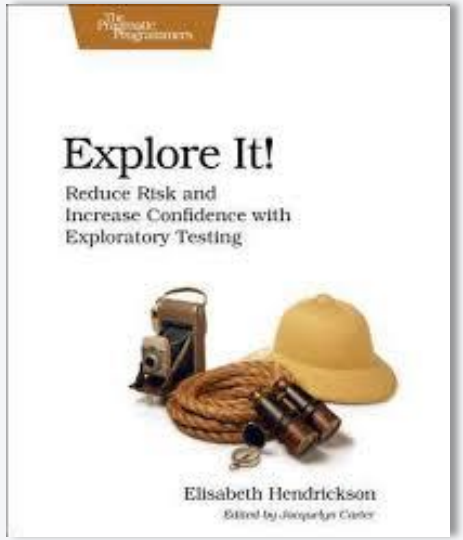
Rob.van.steenbergen@cerios.nl

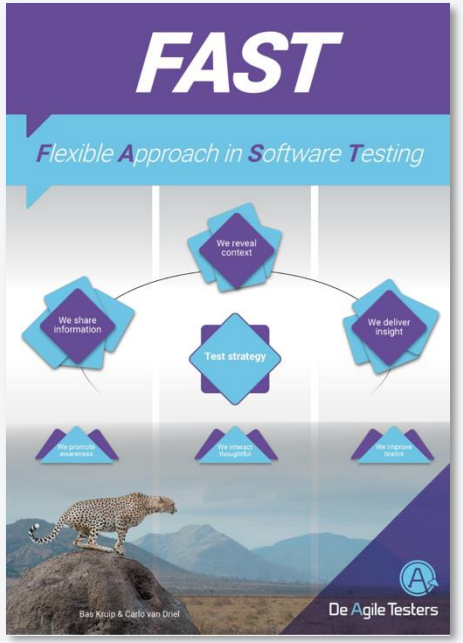
Helvoirt, Utrecht, Rotterdam, Hoogeveen

www.cerios.nl

References

- Book: Explore it!
  - Elisabeth Hendrickson
- Elisabeth Hendrickson about her book
  - https://www.youtube.com/watch?v=9FKY1Is0lgs&t
- Book: FAST – Flexible Approach in Software Testing
  - Bas Kruip / Carlo van Driel
- Session based test management
  - Jonathan en James Bach
  - https://www.satisfice.com/download/session-based-test-management
- James Bach: Checking, Testing, and Mutation
  - https://www.youtube.com/watch?v=TnExUCCTFf4





References

- Exploratory testing: Del Dewar
  - https://findingdeefex.com/2016/05/20/the-testing-checking-synergy/
- Blog Michael Bolton about testing tours
  - http://www.developsense.com/blog/2009/04/of-testing-tours-and-dashboards/
- Generate test ideas misc sources
  - https://www.testrail.com/blog/generate-quick-test-ideas/
  - https://www.jpourdanis.com/2024/07/23/37-sources-for-test-ideas/
  - http://www.testingeducation.org/BBST/foundations/OracleHeuristicsLab6b.pdf

- Most of images used are free images from https://pixabay.com/nl/