Open Universiteit

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

The future of automating regression testing at the Graphical User Interface (GUI)

Tanja Vos

Open Universiteit

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

The future of automating regression testing at the Graphical User Interface (GUI)

(or how to stop flushing money away doing it the way we currently do)

Tanja Vos

5 billion euros

annual cost of
maintaining broken
GUI test scripts

Estimation based on EU automation testing market ≈ €10–11 B (2024-25) × 30–50 % maintenance share (sources: Market Data Forecast, Fortune Business Insights, Capgemini World Quality Report 2023-24, IT Convergence 2023.)

Testing at the GUI a user story to ”apply for a loan”

Step 1
Open Bank Application

Step2
Login with a valid user account

Step 3
Click on Request Loan

Step 4
Type a valid amount of money

Step 5
Click to apply now the loan

Testing at the GUI a user story to ”apply for a loan”

Step 1
Open Bank Application

Step2
Login with a valid user account

Step 3
Click on Request Loan

Step 4
Type a valid amount of money

Step 5
Click to apply now the loan

Testing at the GUI a user story to ”apply for a loan”

Step 1
Open Bank Application

Step2
Login with a valid user account

Step 3
Click on Request Loan

Step 4
Type a valid amount of money

Step 5
Click to apply now the loan

Testing at the GUI a user story to ”apply for a loan”

Step 1
Open Bank Application

Step2
Login with a valid user account

Step 3
Click on Request Loan

Step 4
Type a valid amount of money

Step 5
Click to apply now the loan

Testing at the GUI a user story to ”apply for a loan”

Step 1
Open Bank Application

Step2
Login with a valid user account

Step 3
Click on Request Loan

Step 4
Type a valid amount of money

Step 5
Click to apply now the loan

Testing at the GUI a user story to ”apply for a loan”

After last step:

- The user story succeeds
- The loan is approved

After each step:

- No failure occurs
- No error message appears

We need to repeat this every time the software changes!

Manual regression testing

- Expensive
- Boring
- Error-Prone

Needs to be automated

State of automation practice: make scripts

Step 1
Open Bank Application

Step2
Login with a valid user account

Step 3
Click on Request Loan

Step 4
Type a valid amount of money

Step 5
Click to apply now the loan

```python
from playwright.sync_api import sync_playwright

with sync_playwright() as p:
    # Launch browser
    browser = p.chromium.launch(headless=False)
    page = browser.new_page()

    # Step 1: Open bank application
    page.goto("https://parabank.parasoft.com/parabank/index.htm")

    # Step 2: Login with a valid user account
    page.fill("input[name='username']", "john")
    page.fill("input[name='password']", "demo")
    page.click("input[value='Log In']")

    # Step 3: Click on 'Request Loan'
    page.click("a[href='requestloan']")

    # Step 4: Type a valid amount of money
    page.fill("input[name='amount']", "999")
    page.fill("input[name='downPayment']", "123")

    # Step 5: Click to apply for the loan
    page.click("input[value='Apply Now']")

    # Optional: check that loan was approved
    if "Loan Request Processed" in page.content() or "approved" in page.content().lower():
        print("Loan approved successfully.")
    else:
        print("Loan approval failed or not displayed.")

    # Close browser
    browser.close()
```

When the GUI changes……

Step 1
Open Bank Application

Step2
Login with a valid user account

Step 3
Click on Request Loan

Step 4
Type a valid amount of money

Step 5
Click to apply now the loan

Writing scripts for automated test execution is over in the era of AI.

from SCRIPTED
to SCRIPTLESS

SCRIPTLESS

≠

low-code, no code, codeless, code less

from SCRIPTED to SCRIPTLESS

How did we get there?

What this means for:


testers


companies

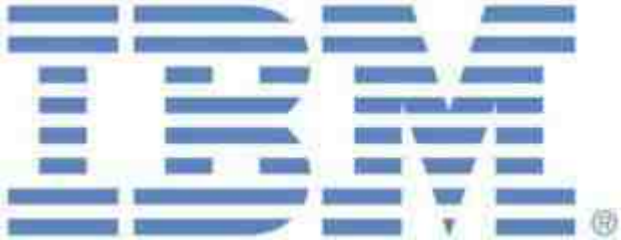

research


AI

2010 - 2013

EU funding: 5.845.000 euros

It all started in 2010…..



Universiteit Utrecht

Conditions for doing the research

❌ Use testers' time

❌ Access the source code

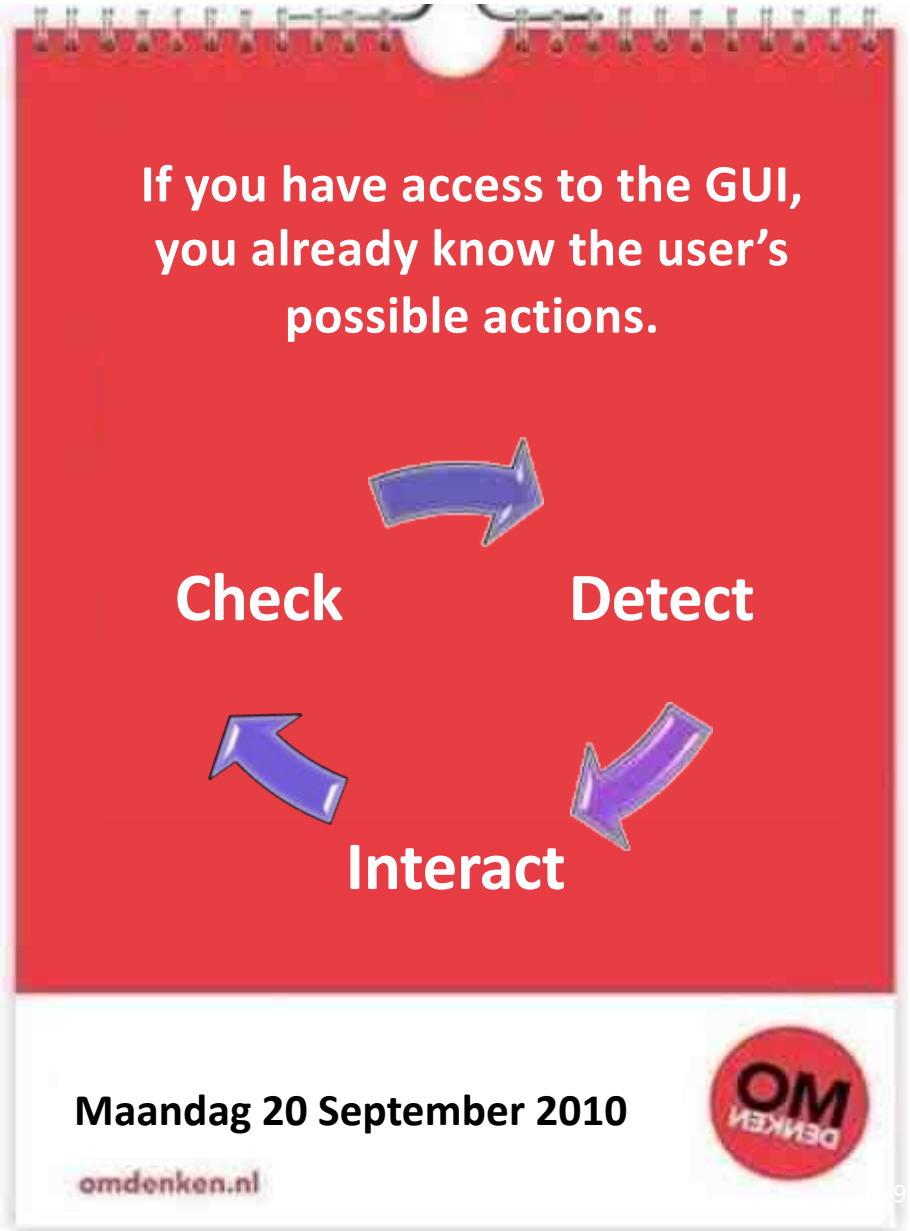
❌ Access the bug database

❌ Publish how many bugs we found

✅ Access to the GUI of the system in production

GUIs

- **Simple**
  - always the same building blocks
  - buttons, menus, text fields, etc.
- **Universal**
  - every app (web, desktop, mobile) has them
- **Accessible**
  - browsers and OS APIs reveal GUI state
    - **GUI ripping** (Atif Memon - GUITAR)
  - we can perform actions automatically
    - **GUI crawling** (Ali Mesbah – CRAWLJAX)

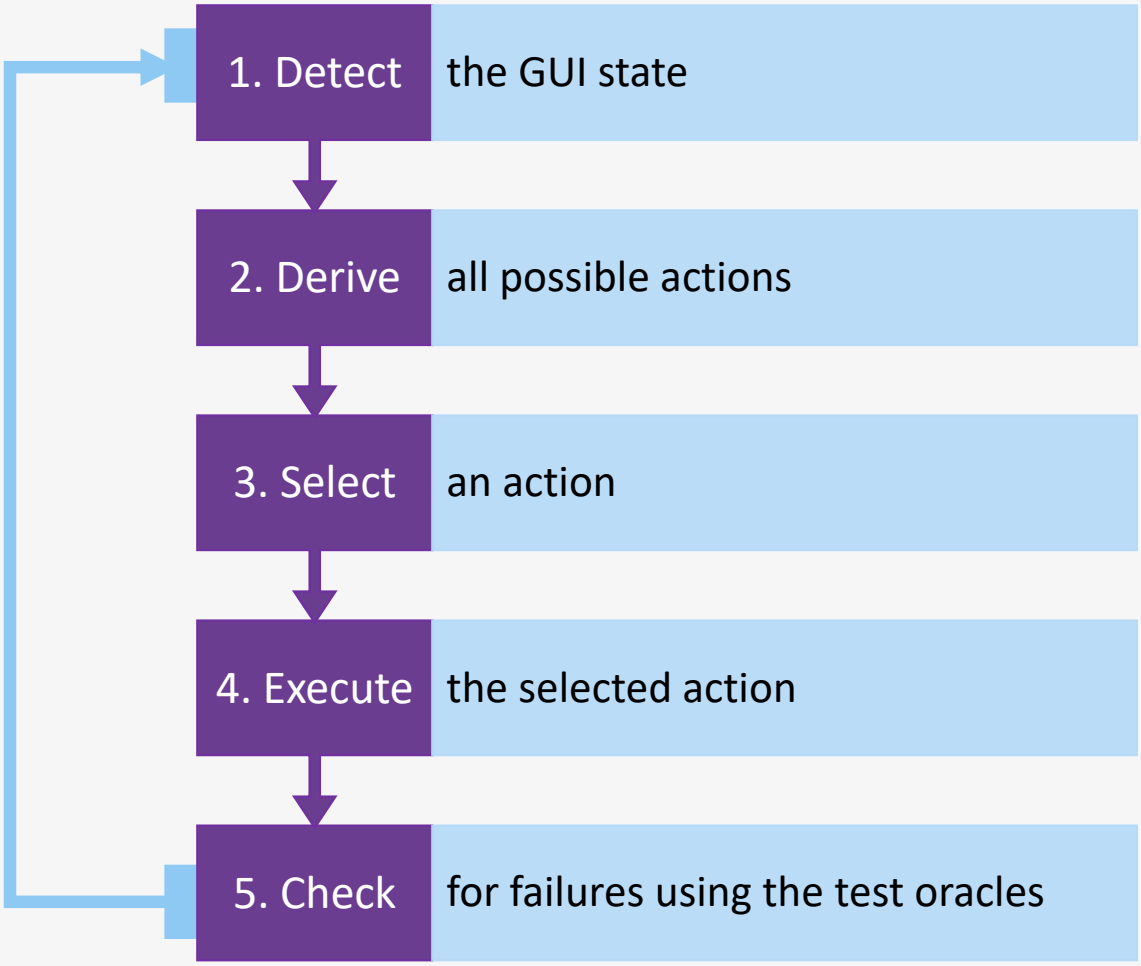


Then we can just start testing

at the GUI level

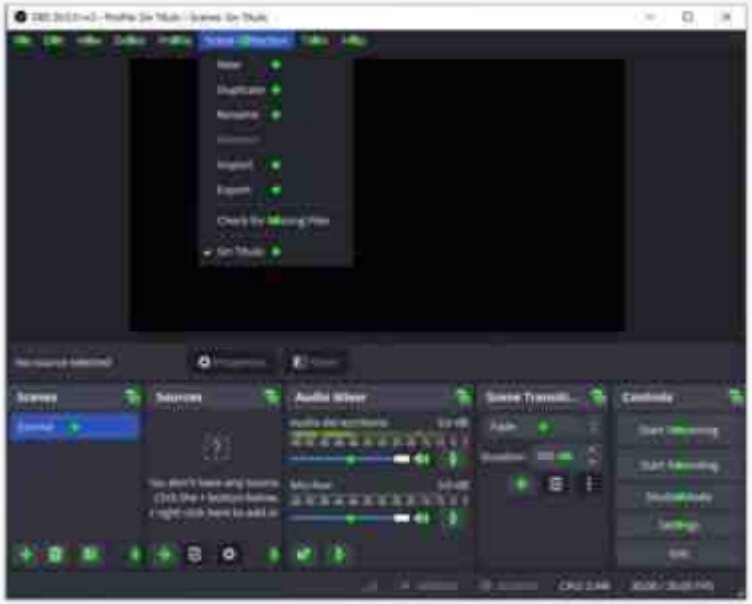
no need for scripts!

How does scriptless testing work?



1. Get current state of the SUT

- Accessibility API of Windows (UIAutomation)
- Webdriver for web applications
- Appium for mobile
- iv4XR for XR games



- The state consists of all widgets + their properties

2. Derive actions

- Click (Home)
- Click (About Us)
- Click (Contact Us)
- Type (text, 'Loan Amount'-field)
- Type (text, 'Down Payment'-field)
- Click (From Account)
- Click (Apply Now)
- Click (Services)
- Click (Open New Account)
- Click (Accounts Overview)
- Click (Transfer Funds)
- Click (Bill Pay)
- More actions...

3. Select action

- Click (Home)
- Click (About Us)
- Click (Contact Us)
- Type (text, 'Loan Amount'-field)
- Type (text, 'Down Payment'-field)
- Click (From Account)
- Click (Apply Now)
- Click (Services)
- Click (Open New Account)
- Click (Accounts Overview)
- Click (Transfer Funds)
- Click (Bill Pay)
- More actions...

4. Execute the action, go to new state

Click (Apply Now)

Derive actions again

5. Check oracles

- Oracle = Mechanism to determine whether we found a failure
- In TESTAR: Oracles give verdicts about each state of the GUI
- Per default: general system failures:
  - crashes
  - hangs
  - suspicious outputs

It worked! We found new bugs!


Evaluating Rogue User Testing in Industry: an Experience Report

Clave

	TESTAR
Preparation	26 hour
Testing	91 hour
Post testing	1,5 hour
Critical faults	10


Scriptless Testing at the GUI Level in an Industrial Setting

ProRail

Capgemini

	TESTAR	Manual
Preparation	44 hour	43 hour
Testing	51 hour	6 hour
Post testing	5 hour	2 hour
Critical faults	3	0

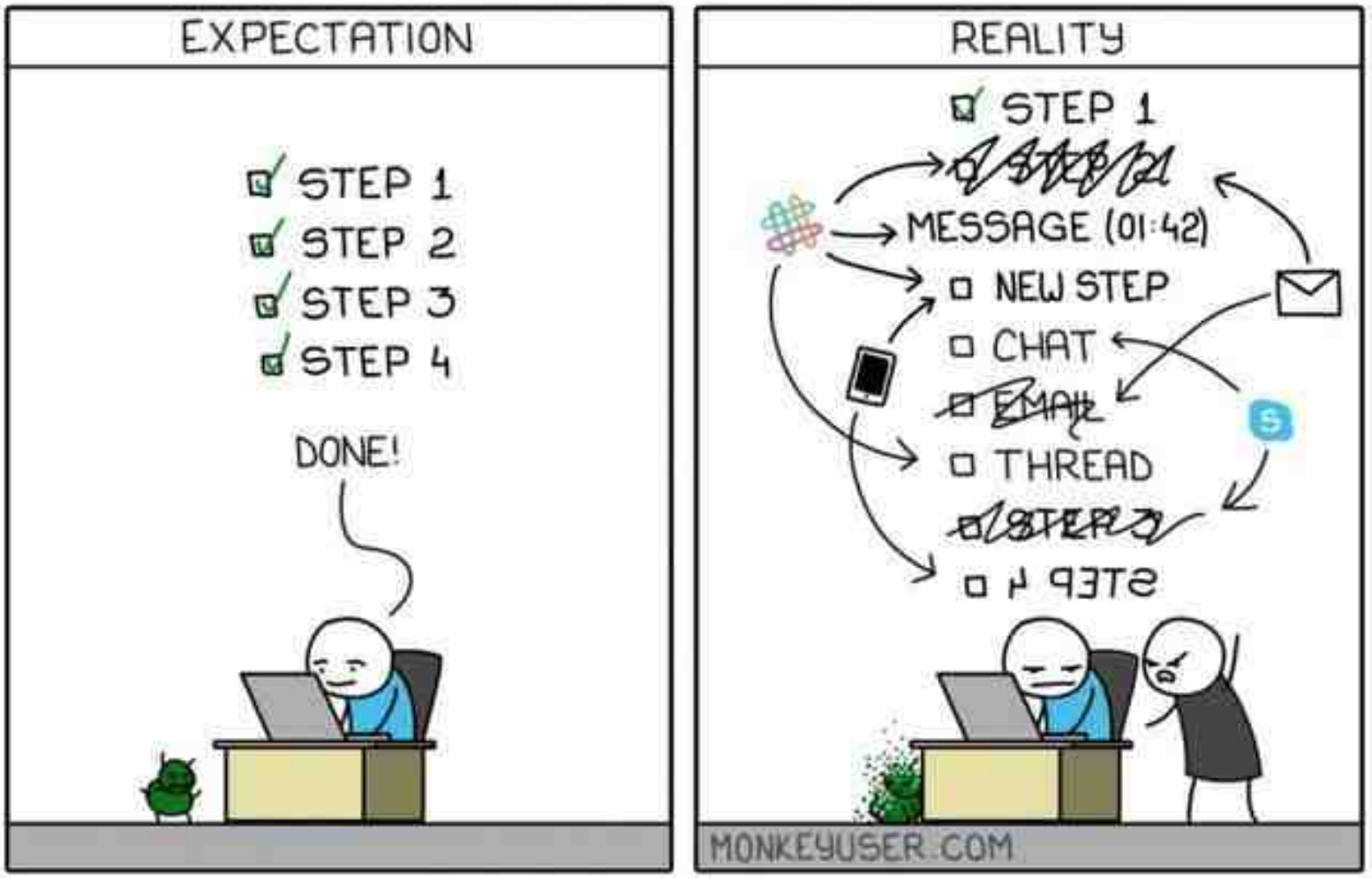
Finding bugs is one thing…

How can we test user stories?

Well, with random.. you can´t!

user stories

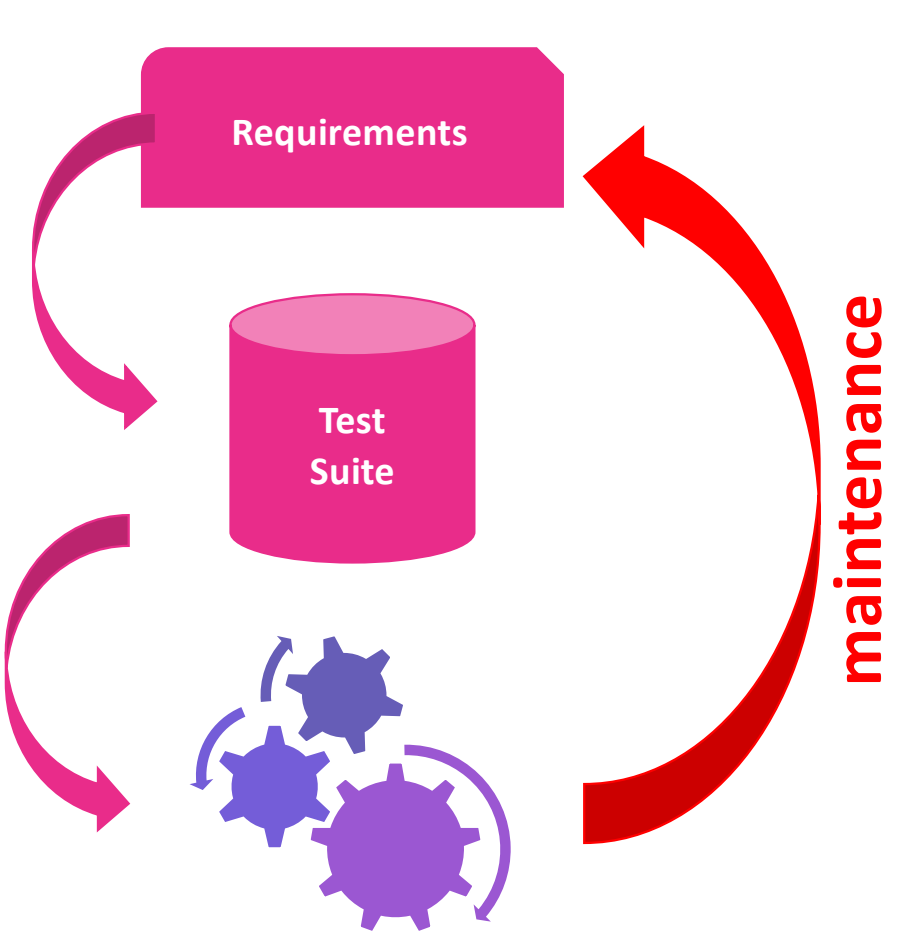
non-user stories

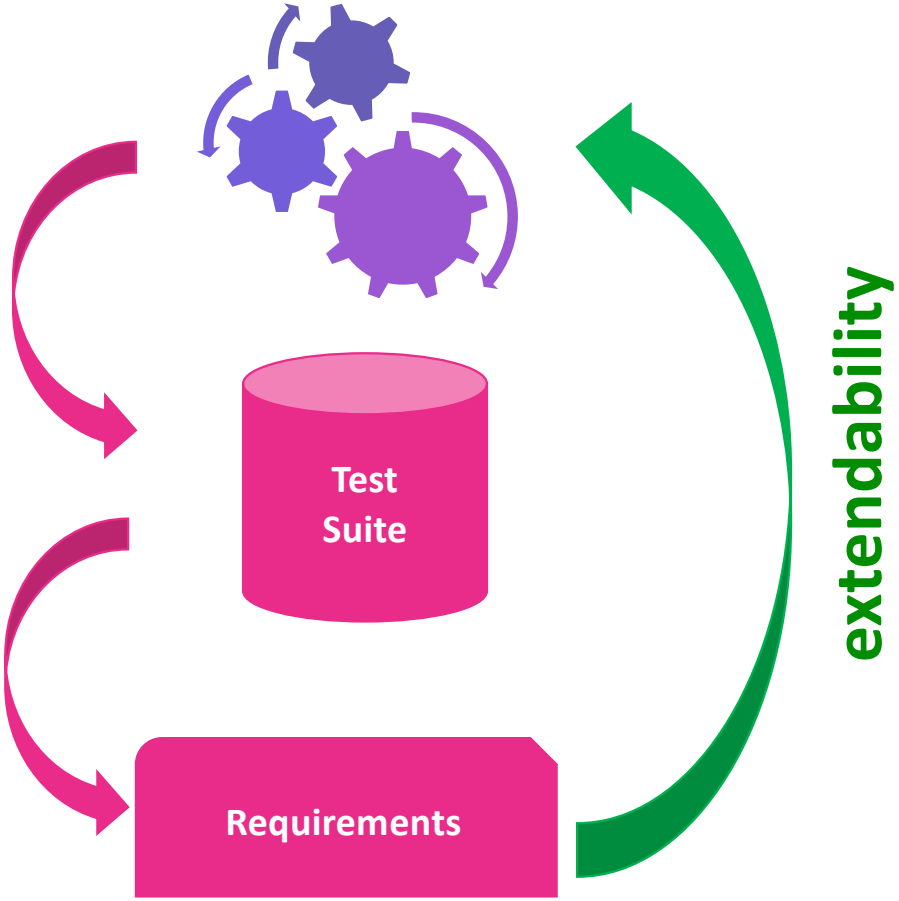


Complementary approaches


Research dot on the horizon: get rid of scripts!

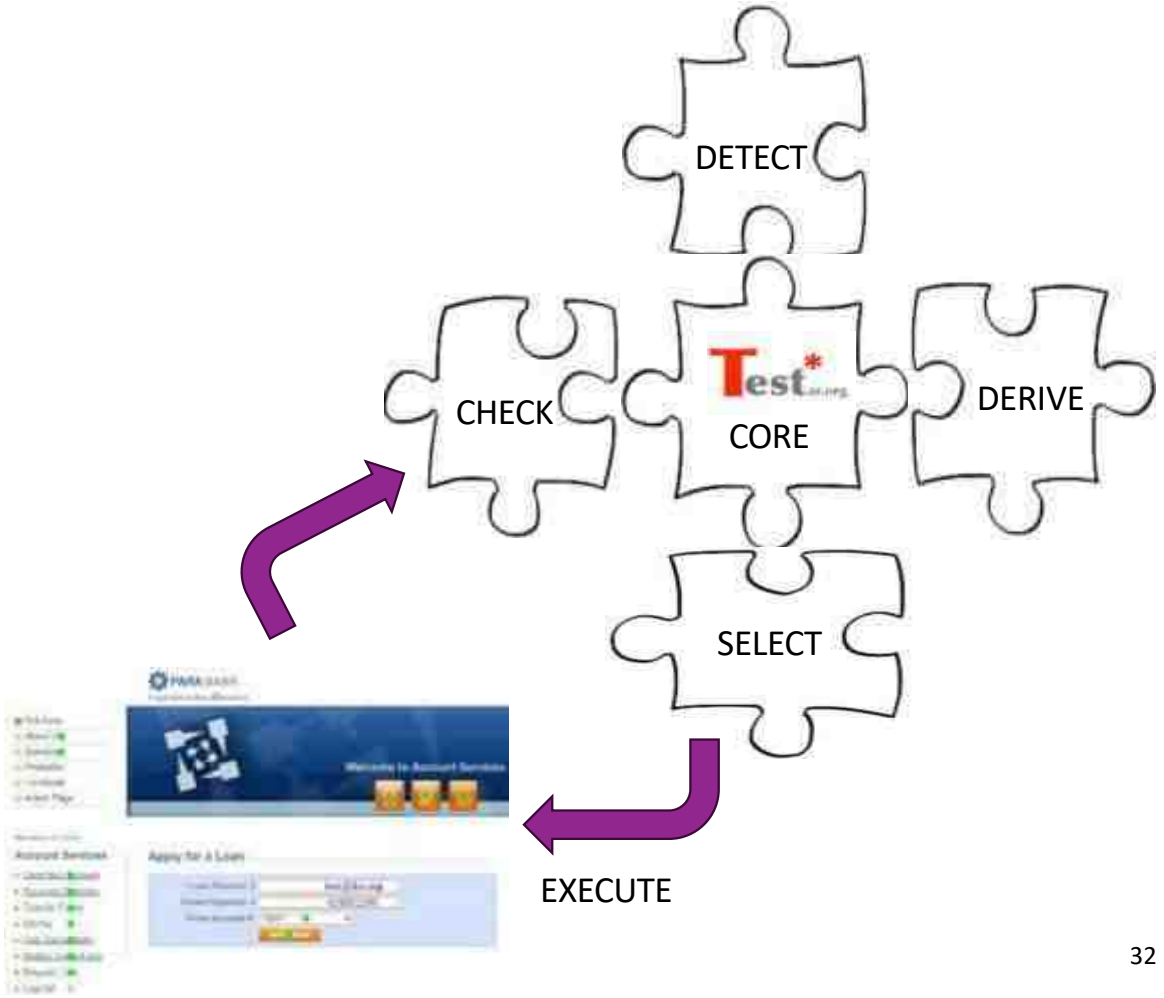
Flip test automation



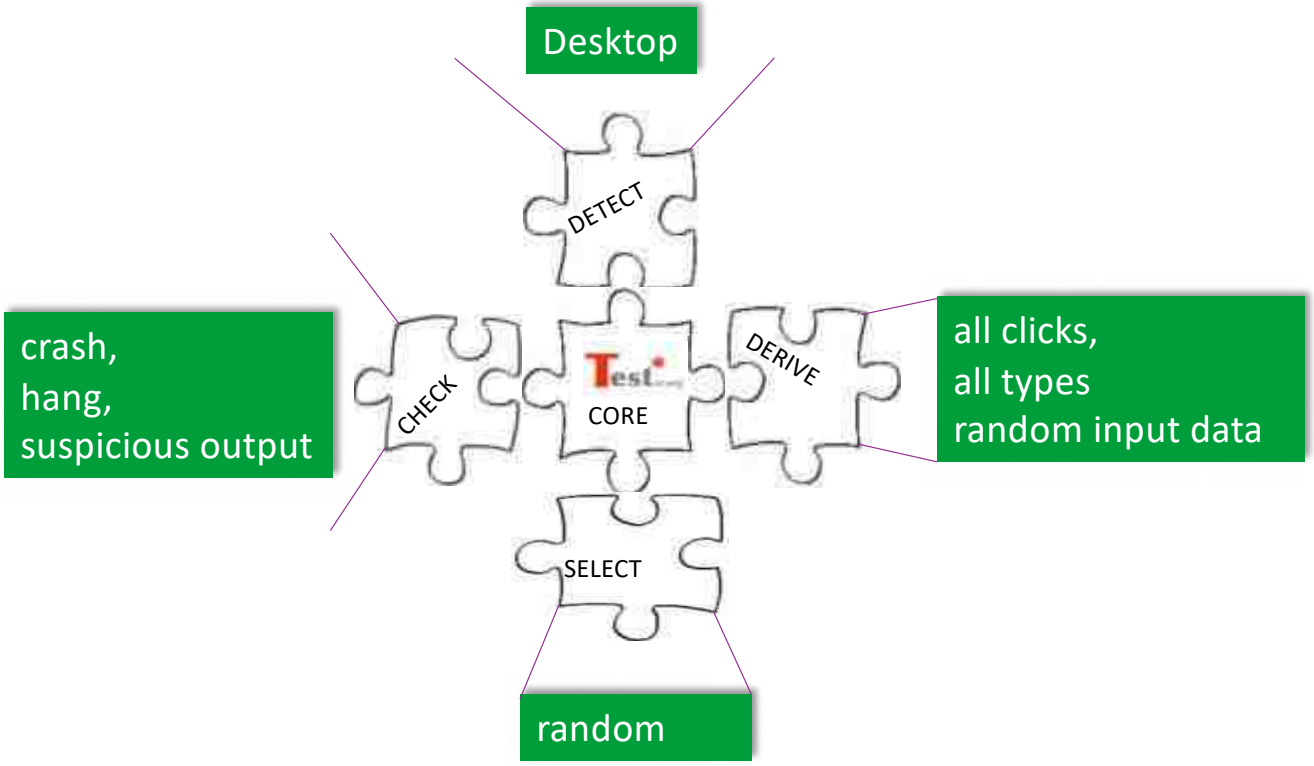


Modular plug-in based implementation

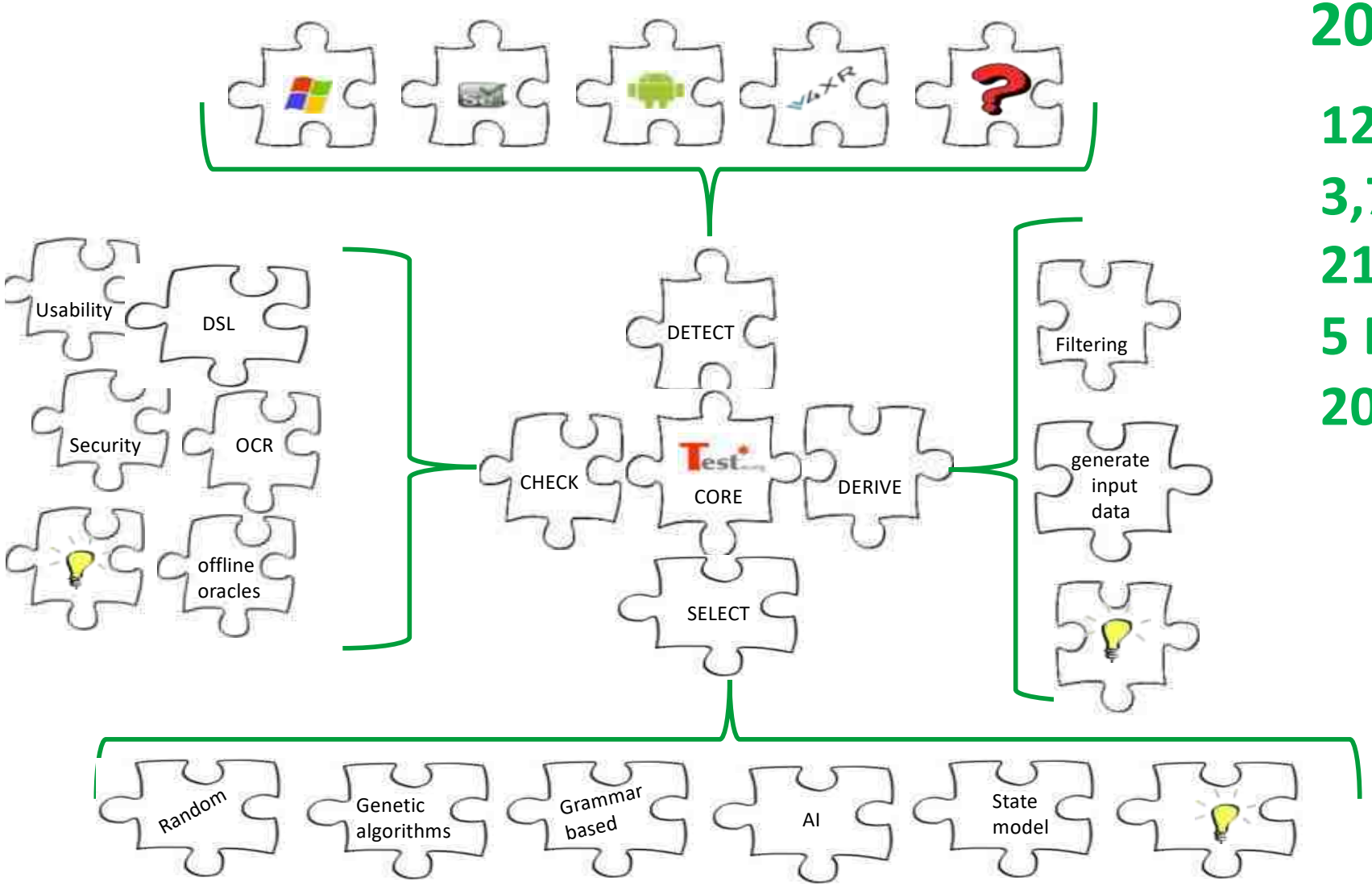
1. Detect the GUI state
2. Derive all possible actions
3. Select an action
4. Execute the selected action
5. Check for failures using oracles



In 2010….


Desktop
DETECT
crash,
hang,
suspicious output
CHECK
Test
CORE
DERIVE
all clicks,
all types
random input data
SELECT
random

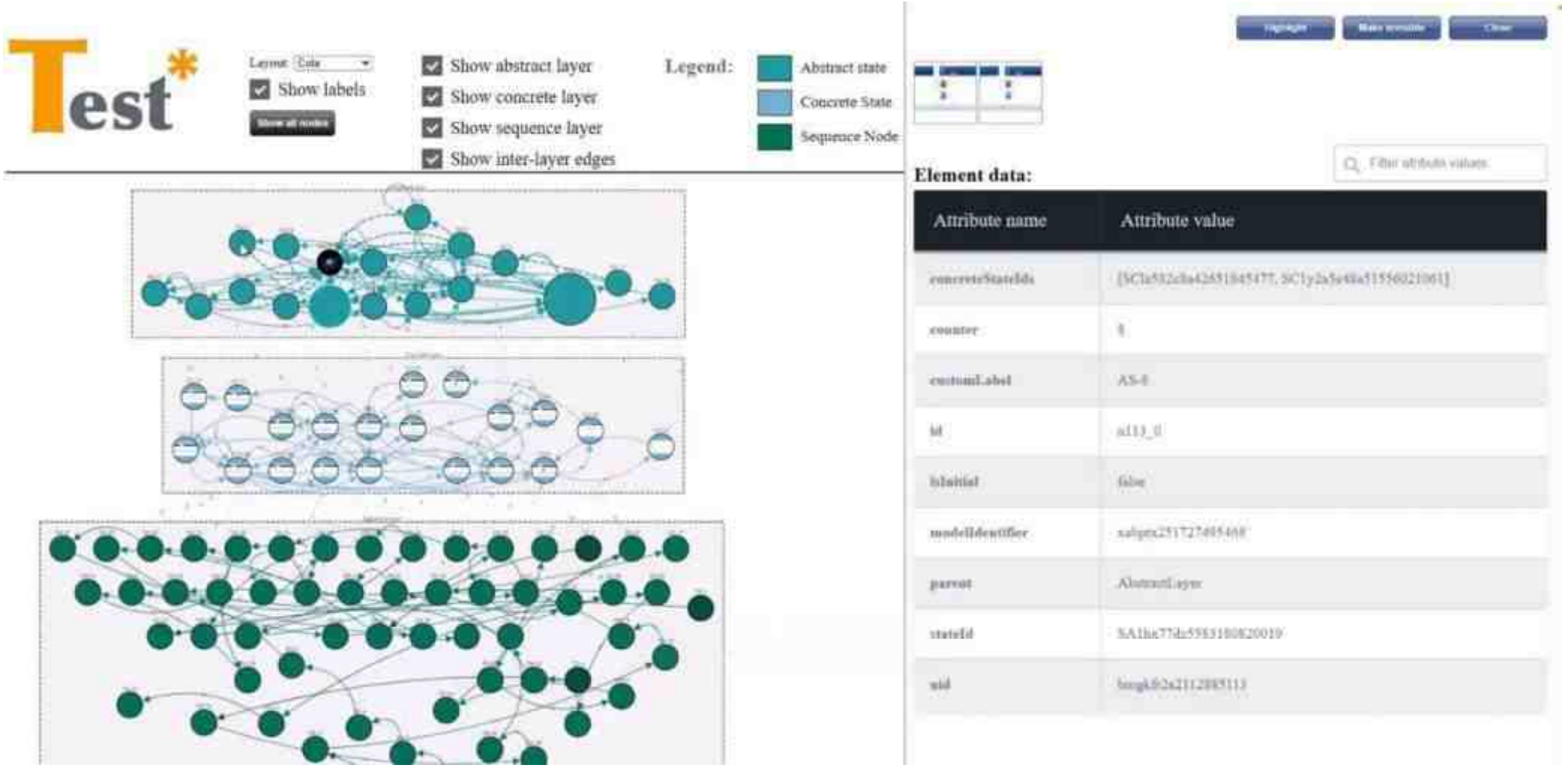

look at all these critical bugs!



2010 - 2022

120k LOC
3,7 PhD thesis
21 Ba/MSc thesis
5 EU projects
20 companies

Infer state-models



Test*

Layout: Klay

Show labels

Show all nodes

Show abstract layer

Show concrete layer

Show sequence layer

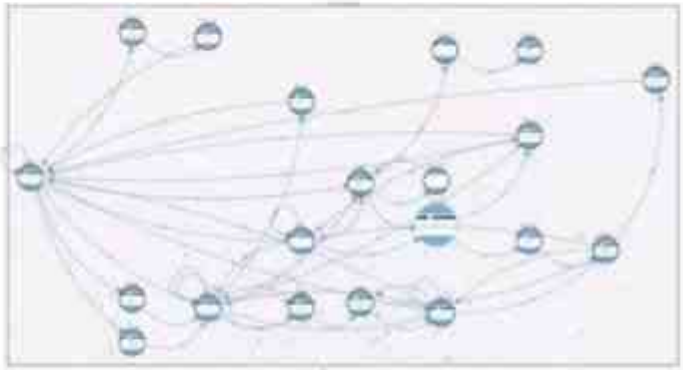
Show inter-layer edges

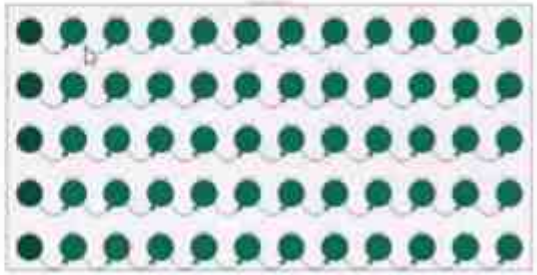
Legend:

Abstract state

Concrete State

Sequence Node





Element data:

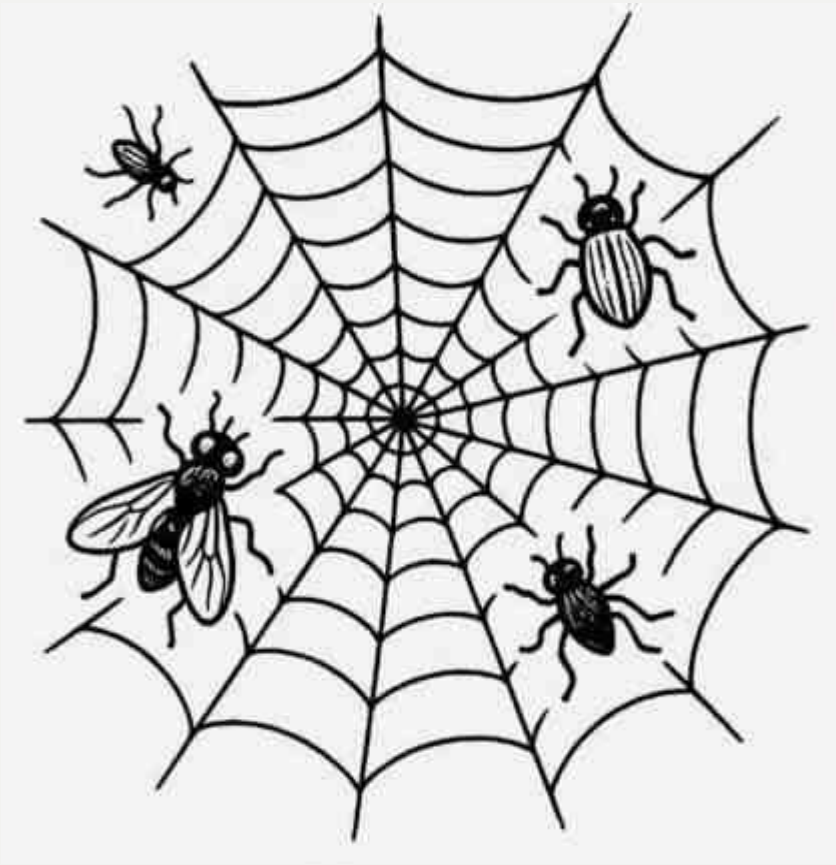
Filter attribute values

Attribute name	Attribute value
Abs(R)ID	5Rzm9vyd4x13064191173
Abs(R_k_T)ID	5T2gpdsh0f93060229513
Abs(R_k_T_k_P)ID	5Pfbgxiz7b01979918752
AbstractID	5Aaxp8j253b3105190985
Blocked	false

Failure Detection Potential of scriptless oracles

Industrial Bug database
N=2500

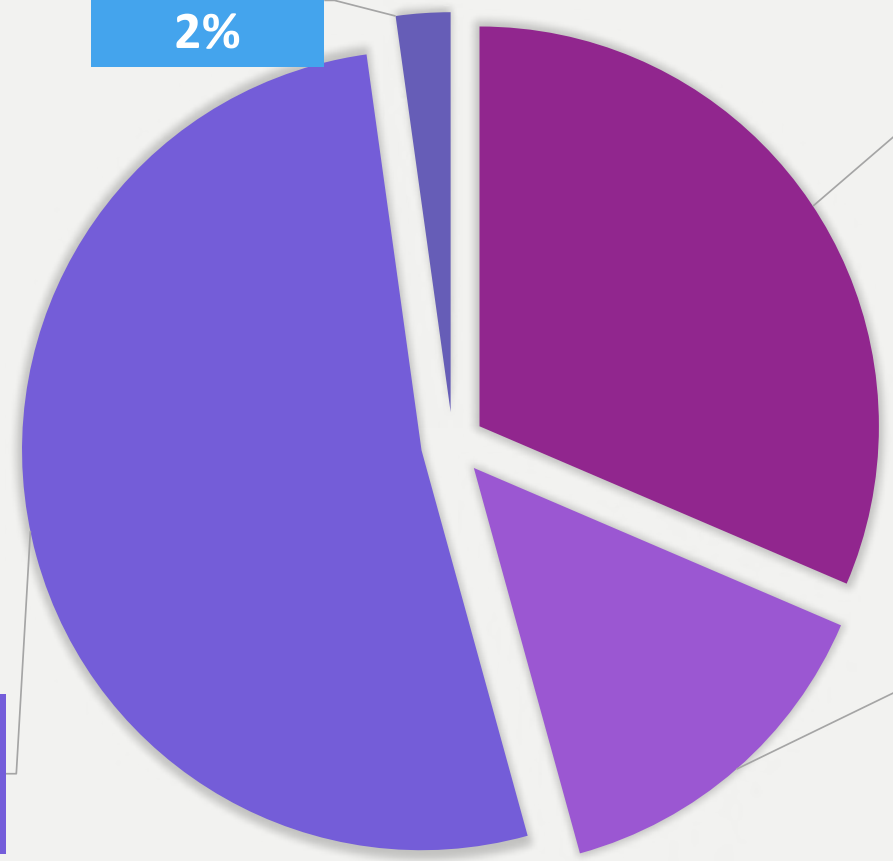
DigiOffice



Industrial Bug database
N=2500

DigiOffice

Failure Detection Potential of scriptless oracles



Adding oracles like GUI state predicates

```
1  public Verdict widgetImageWithoutAlternativeText(State state) {
2    // Loop through all the widgets of the state
3    for(Widget widget : state) {
4      // Check whether the widget is an image (<img>)
5      // and if it lacks alternative text
6      if(widget.get(Tags.Role, Roles.Widget).equals(WdRoles.WdImg))
7          &&
8          (widget.get(WdTags.WebAlt, null) == null
9          ||
10         widget.get(WdTags.get(WdTags.WebAlt, "").isBlank())) {
11
12             String verdictMsg = String.format("Detected web image
                   ↪ widgets '%s' without alternative text!",
                   ↪ widget.get(WdTags.WebTitle, ""));
13
14             return new
                   ↪ Verdict(Verdict.Severity.WARNING_ACCESSIBILITY,
                   ↪ verdictMsg);
15         }
16     }
17     return Verdict.OK;
18 }
```

```java
public Verdict widgetImageWithoutAlternativeText(State state) {
  // Loop through all the widgets of the state
  for(Widget widget : state) {
    // Check whether the widget is an image (<img>)
    // and if it lacks alternative text
    if(widget.get(Tags.Role, Roles.Widget).equals(WdRoles.WdImg))
       &&
       (widget.get(WdTags.WebAlt, null) == null
       ||
       widget.get(WdTags.get(WdTags.WebAlt, "").isBlank())) {

          String verdictMsg = String.format("Detected web image
               ↪ widgets '%s' without alternative text!",
               ↪ widget.get(WdTags.WebTitle, ""));

          return new
               ↪ Verdict(Verdict.Severity.WARNING_ACCESSIBILITY,
               ↪ verdictMsg);
       }
    }
    return Verdict.OK;
}
```

Oracle language

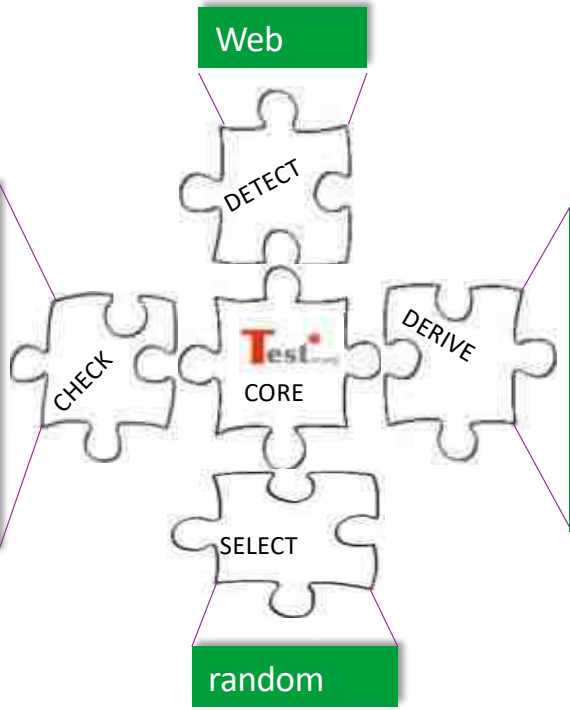

```
assert image "img" has nonempty label
       "Image Without Alternative Text".
```

Academia/Industry studies: E-Dynamics + Yoobi Web app

yoobi

Tool	New bugs	Known bugs
Selenium	2 high	1 high
TESTAR	2 high 4 low	-

added 3 SUT specific oracles for:
- *errors on browser console*
- *specific class selector*
- *fallback text for missing translations*

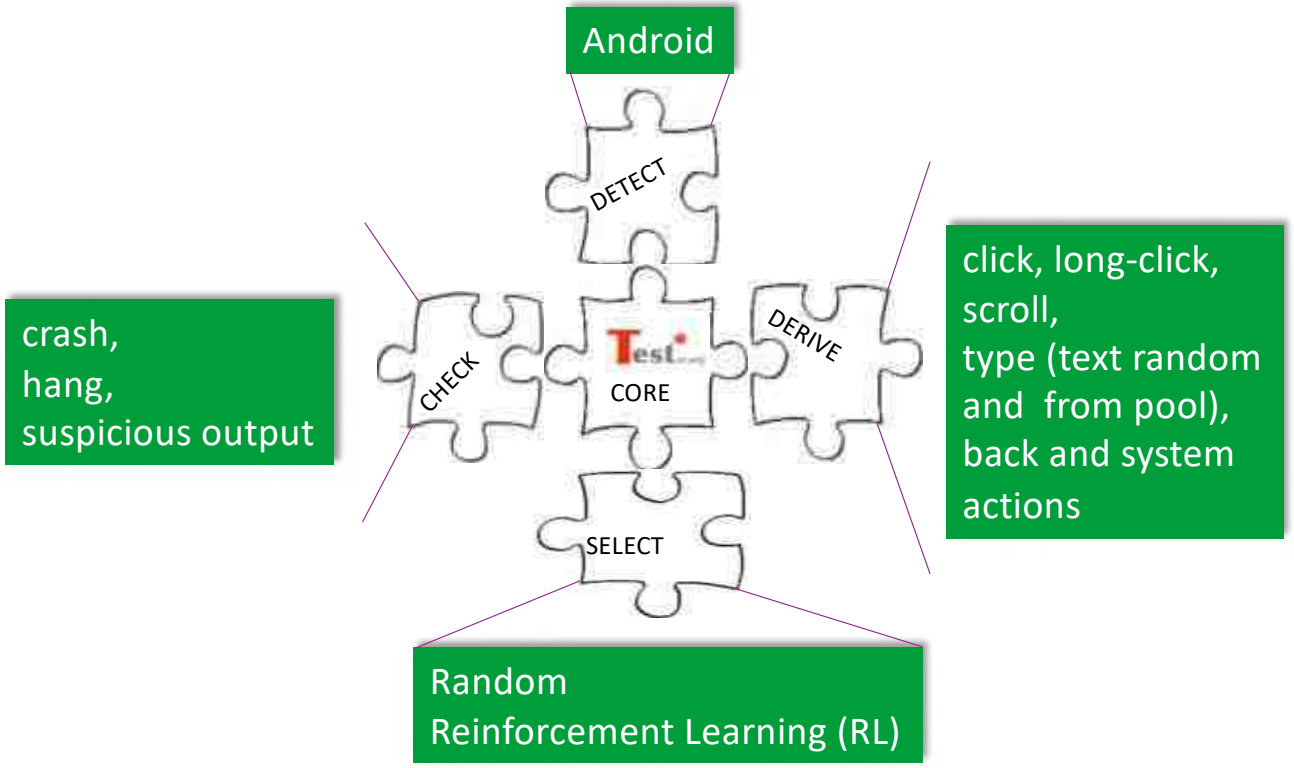


added rules for:
- only SUT allowed domains
- the cookie consent
- the login
- the elements that can be ignored.

Academia/Industry studies: ING + Android app

ING

Tool	% IC	% LC	% MC
Espresso	43,9	43,4	45,9
TESTAR	41,0	40,7	40,8
combined	**52,3**	**52,1**	**52,3**

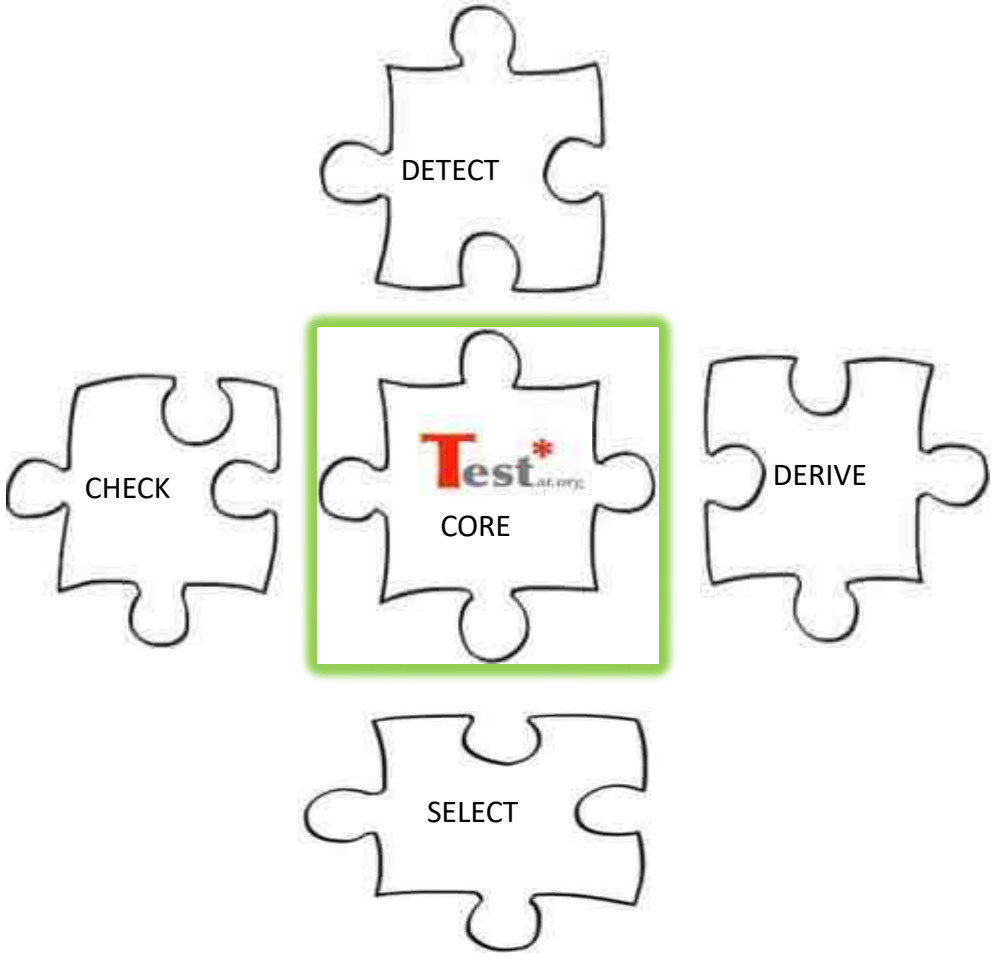


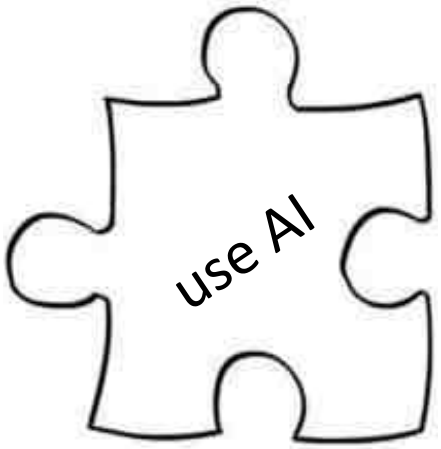

2010 – 2022, still…..
Scripted
Scriptless
Test*

Not for long….

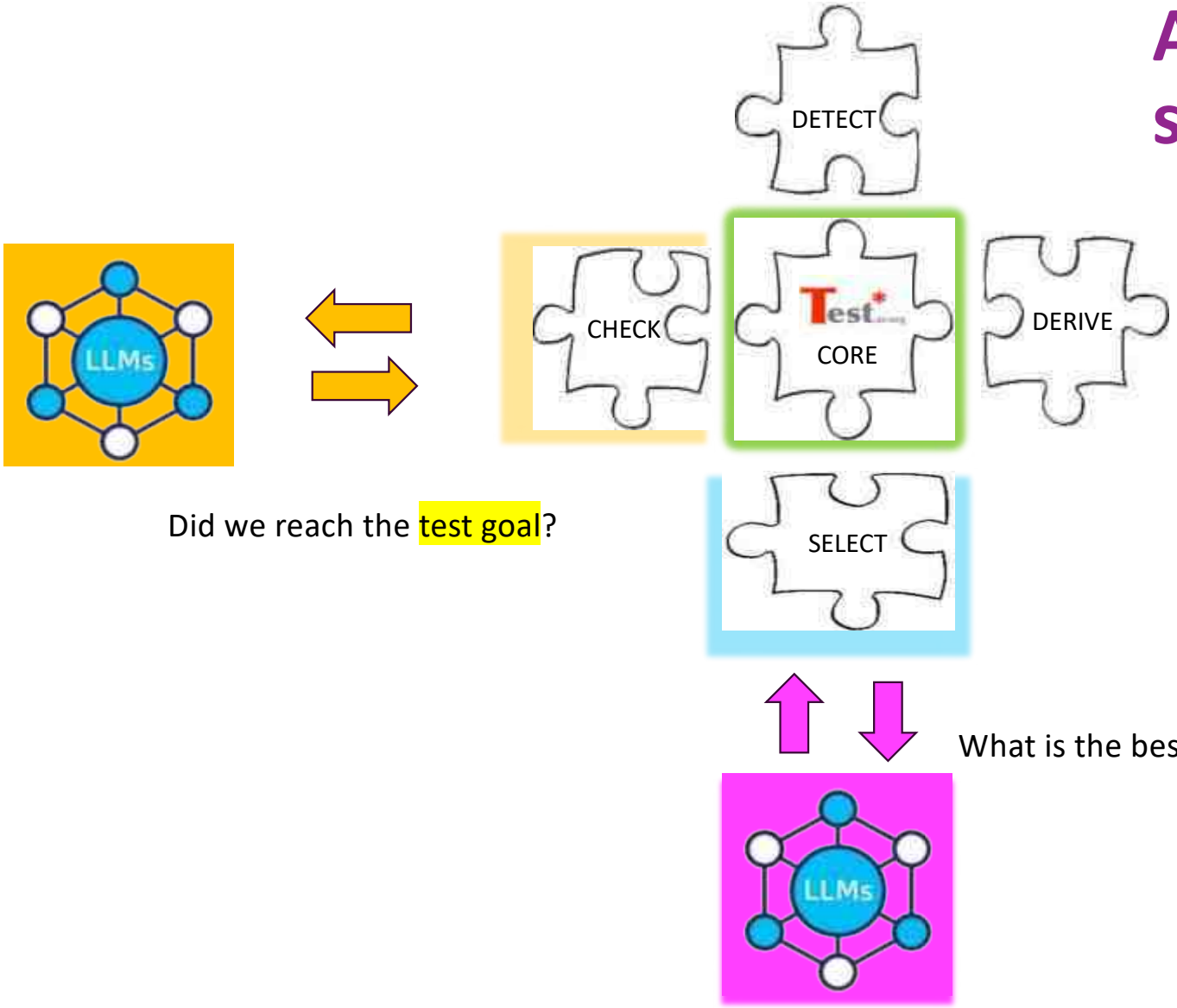
- November 2022
  - launch chatGPT and genAI to general public
  - everything started to become AI-powered
- Just when we started a new project: AUTOLINK
- Funded by NWO
- LINKing requirements and testing
- Open Universiteit, Utrecht University, Universitat Politècnica de València

AI-powered scriptless testing





AI-powered scriptless testing



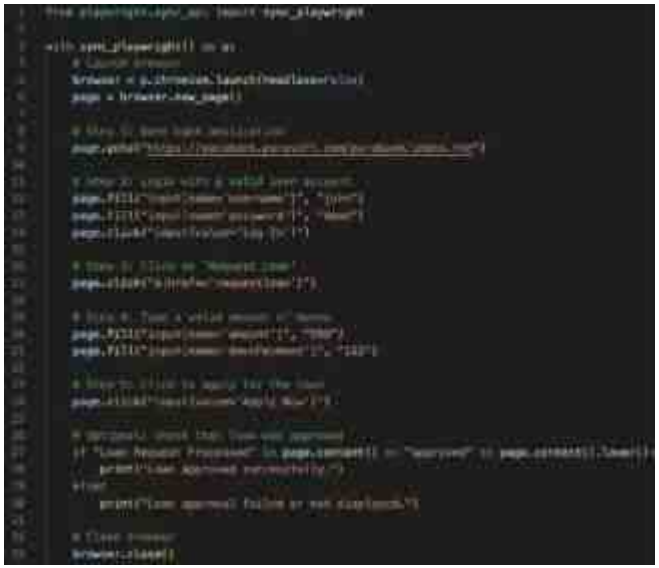
The test goals (can be anything!)

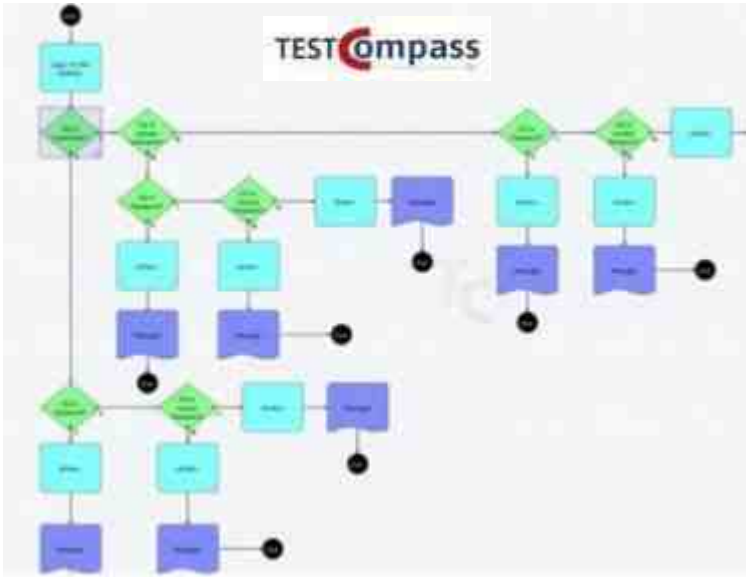
```
Scenario Outline: Request Loan Parabank
  Given I want to log in
  And fill in username 'john'
  And fill in password 'demo'
  When login
  Then Request Loan link is shown

  When fill in amount '999'
  And fill in downPayment '123'
  And Apply Now
  Then the loan has been approved.
```

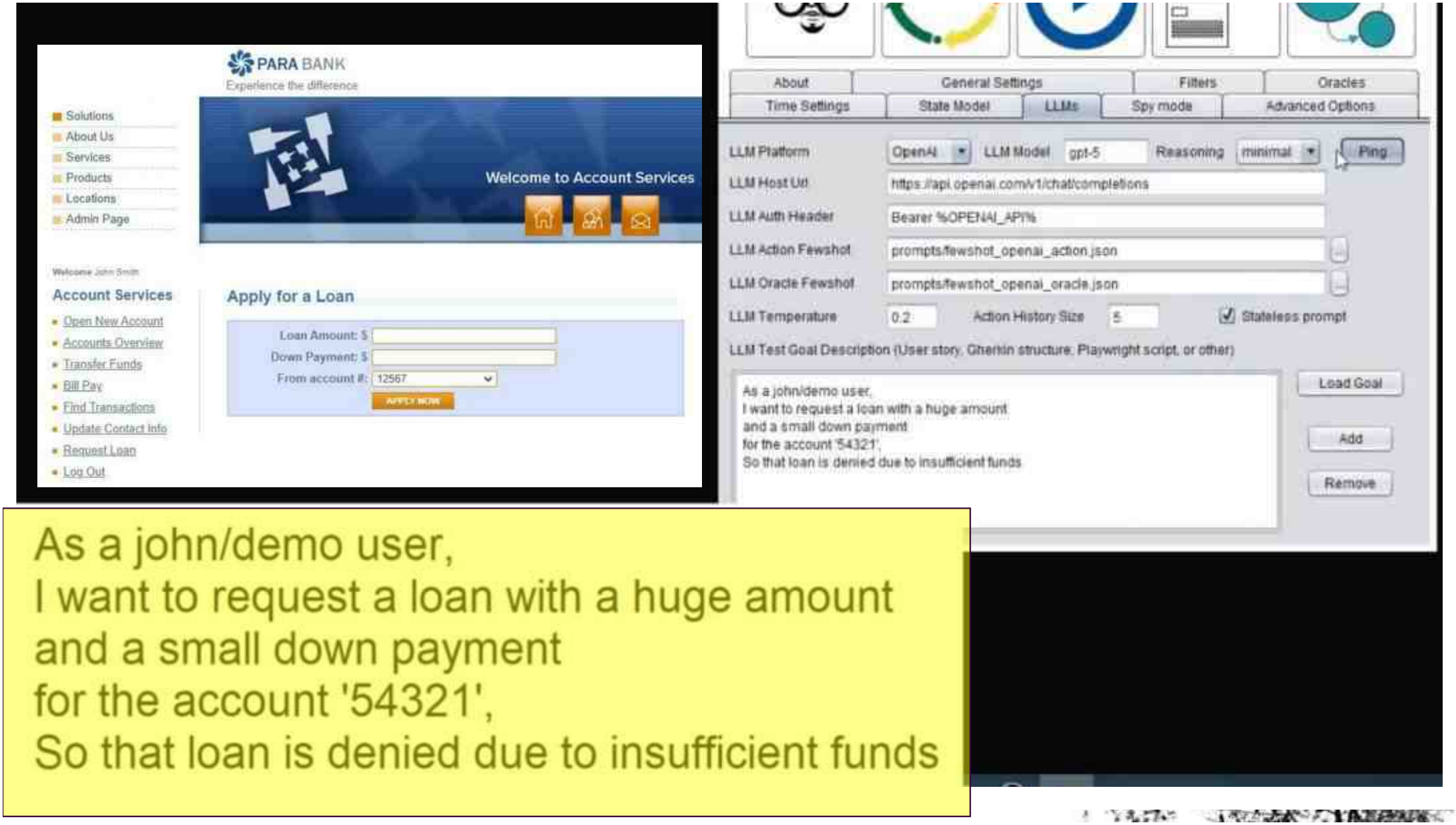
- Natural language
- GWT (Given-When-Then), like Gherkin
- User stories (As a < WHO >, I want < WHAT > so that < WHY>)
- Use cases (Actors, Preconditions, Flow)
- Models

Or re-use your old (broken) scripts!

- Playwright
- Selenium
- Robot’s keywords
- etc..

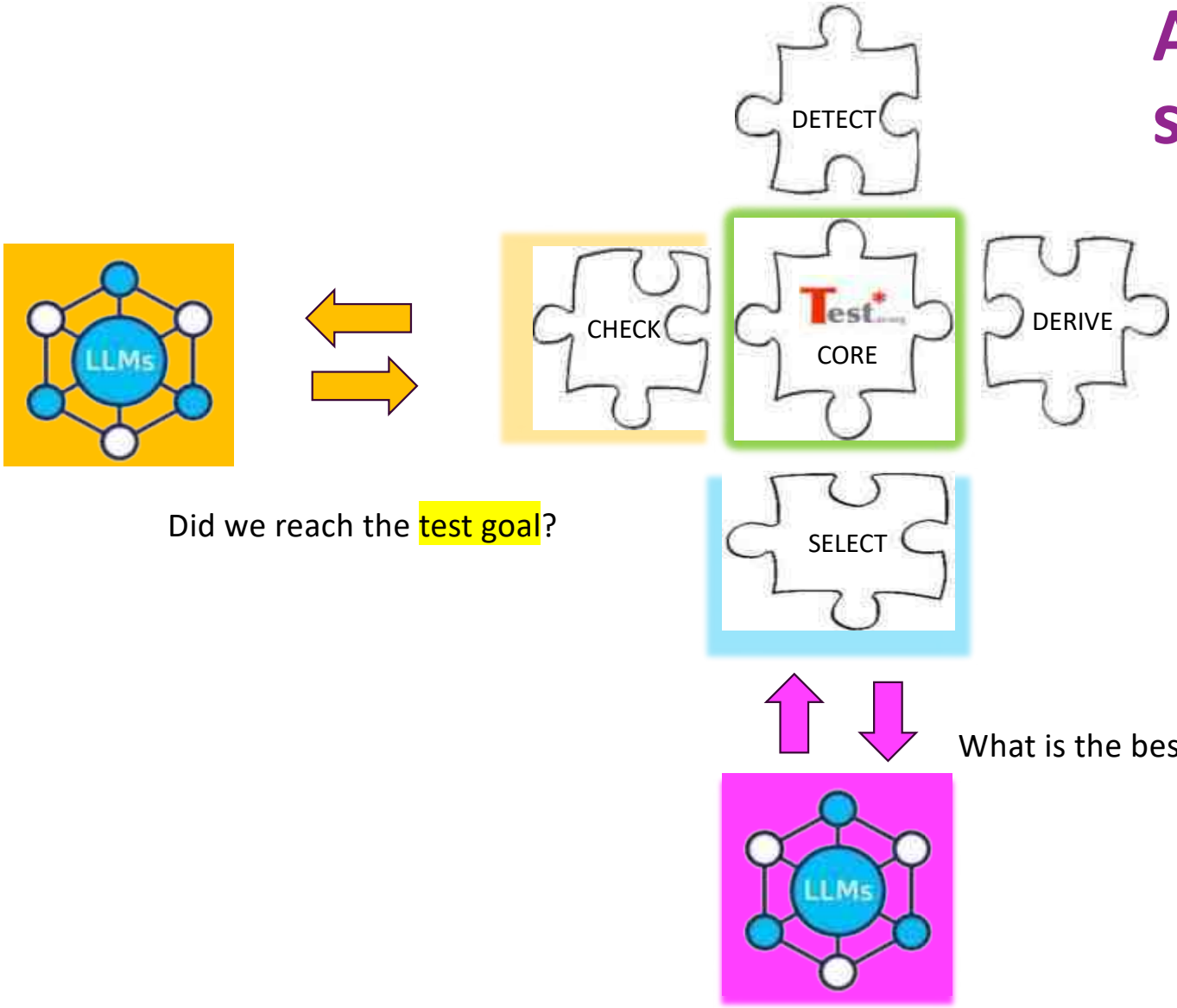






As a john/demo user,
I want to request a loan with a huge amount
and a small down payment
for the account '54321',
So that loan is denied due to insufficient funds

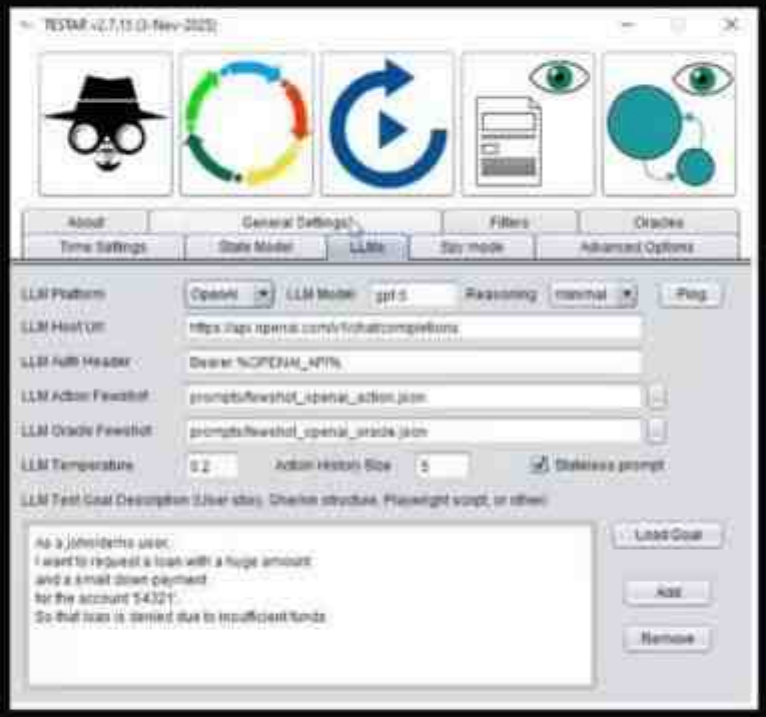
AI-powered scriptless testing

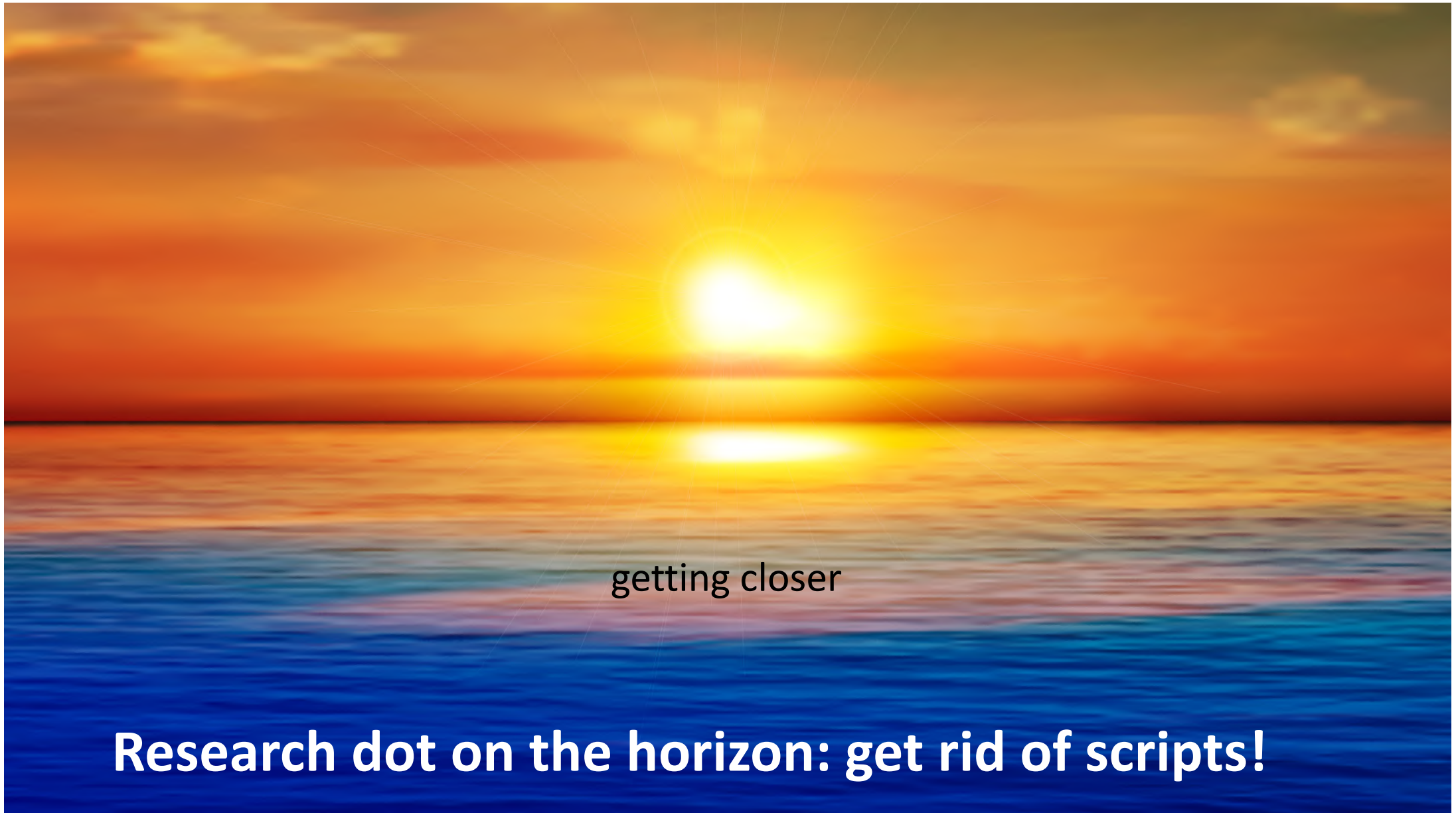


Did we reach the test goal?

What is the best next action to reach the test goal?

Test goal in natural language




getting closer
Research dot on the horizon: get rid of scripts!

What began as random action selection in 2010 is now goal-driven intelligent action selection.

What changes for the tester?

- No more fixing broken scripts
- Know how to write good test goals, not scripts!
- Maintain test goals as the software requirements evolve
- Keep tests relevant, not executable

From keeping scripts in sync with a changing GUI

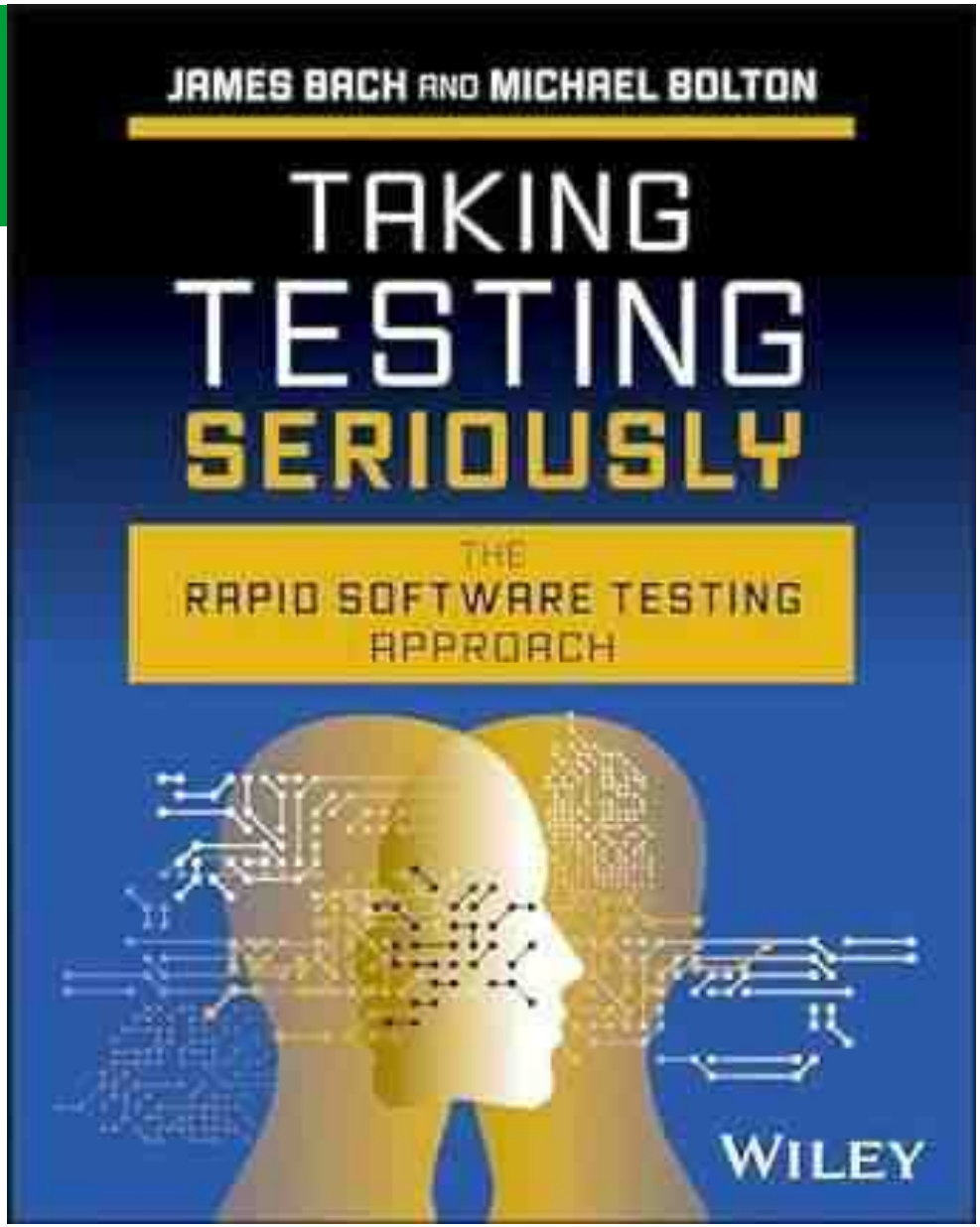
to keeping testing in sync with changing requirements.

(as it should be!)

Finally, people will start.....

The authors of this book, which came out this month, have been saying it for years:

test scripts are not testing!



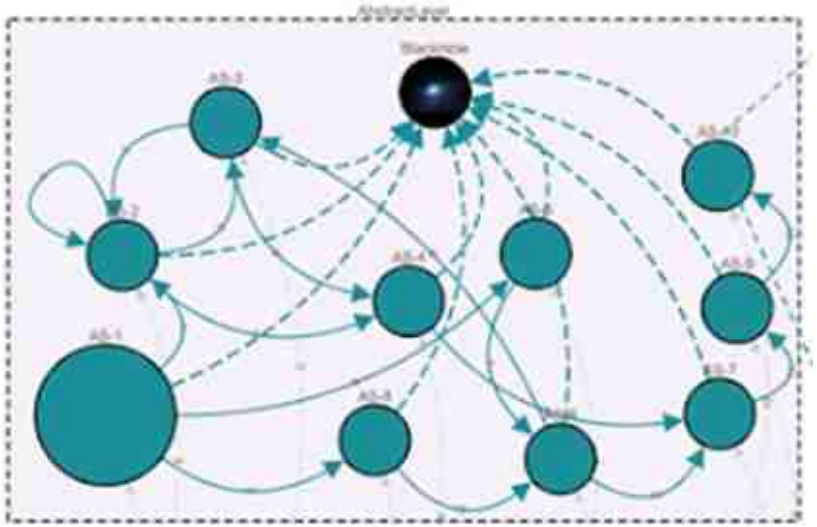
What changes for companies?

- Stop losing money on maintaining old scripts
- Put testers' time where it adds value: improving quality, not fixing automation.

From maintenance cost to product value!

What changes for our research?

- Agents explore stochastically: every run is different
- Product owners (still) want deterministic evidence for acceptance tests
- Traceability will need to link each test sequence to its goals and requirements



From exploration to explanation

(AUTOLINK)

What changes for the AI we use?

1. **Give it better context** *(lightweight, no retraining)*
2. **Teach it new skills** *(medium effort, partial retraining)*
3. **Build it for your domain (SUT)** *(heavy, specialized)*

About testing,
your SUT,
your users,
your test

From general purpose to domain-specific

No more scripts.
Just goals and serious testing.

Contact:

+34 690 917 971

info@testar.org

https://testar.org/

https://github.com/TESTARtool/TESTAR_dev

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Open Universiteit
www.ou.nl