Testing in production

Losing fear, one test at a time

Jorge Marín
November 7th, 2019 – Bristech 🇬🇧

https://jorgemarin.me
@chipironcin

Hidden slide

Welcome to Bristech and thanks for coming to my talk 🤩

I know we are in the second time, you are a bit tired already after attending to many interesting talks but I promise this talk won’t be too heavy.

breathe

no, srsly, breathe

My name is Jorge, and I am an engineer 👋🏼 🇪🇸

Testing in production

Losing fear, one test at a time

https://jorgemarin.me
@chipironcin

Poll time - Raise your hands if you…

- Are having a good day
- Often forget to squash before merge

- Know what testing is
- Know about the testing pyramid or any other way of classifying different testing levels

- Have a production environment
- Are testing your services in production

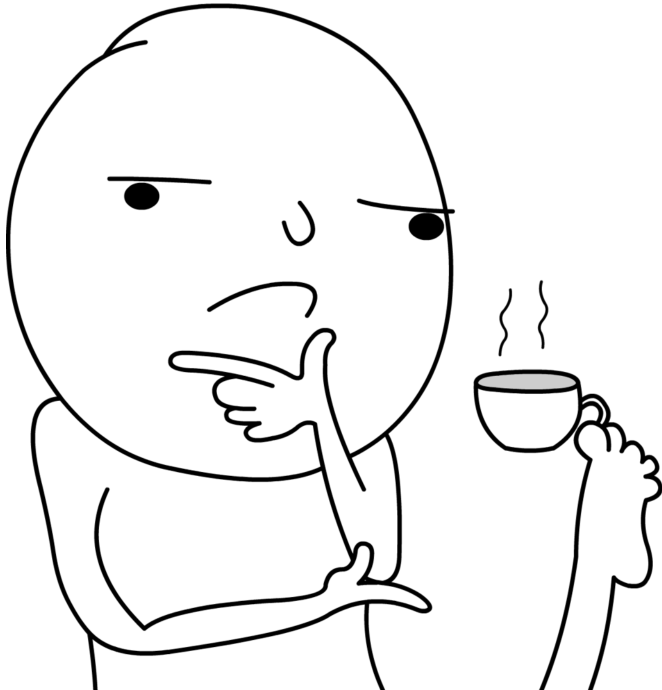
- Prefer spaces to tabs 💥

What is this talk about

- Mostly answers to Frequently Asked Questions
- Not going to be too technical
- Based on the experience gained working at Dyson

Frequent Asked Questions

- Why do you need tests in production?
- What is the right testing level against production systems?
- How can we define an end to end test?
- How to avoid disrupting real users?
- How to keep your tests out of business reports?
- Cleanup
    - Should you clean after yourself? What if you don't?



Let’s dive in!

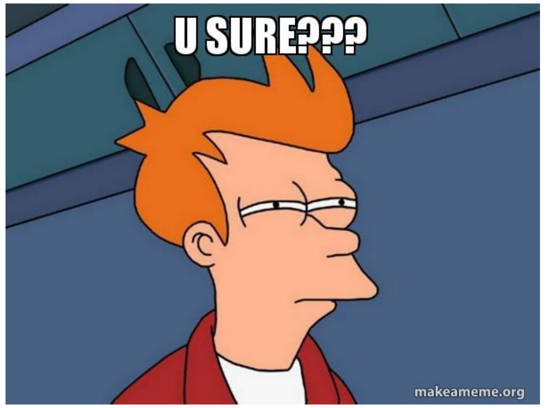
Why do you need testing in production? #1

What do you mean by “testing in production”?

lambda
service
To guarantee that your application continues to work
consistently after the release/deployment

Why do you need testing in production? #1

- But I have the best real-time health dashboards in the world and will tell me when something is wrong



Why do you need testing in production? #1

- Imagine a lambda function
  - One function that returns the current date
  - Unit tested
    - Upon call, returns a formatted string in the form ‘’
  - Integration tested
    - API endpoint works as expected and other services are getting the string
  - Manually tested
  - No new versions, deploy, verify and forget
  - Dashboards are healthy
    - Service healthy
    - Metrics healthy
    - Lots of 200OK

Why do you need testing in production? #1

- Imagine a lambda function
  - One function that returns the current date
  - Unit tested
    - Upon call, returns a formatted string in the form ‘’
  - Integration tested
    - API endpoint works as expected and other services are getting the string
  - Manually tested
  - No new versions, deploy, verify and forget
  - Dashboards are healthy
    - Service healthy
    - Metrics healthy
    - Lots of 200OK

UNTIL YEAR 2038

y2k38

- Storing dates (seconds since Jan 01 1970) as a signed 32-bit binary integer

32-bit binary representation	Seconds since Jan 01 1970	Date	
00000000000000000000000000000000	0	Thursday, 01-Jan-1970 00:00:00 UTC	Beginning epoch
01011101110000111111011000000011	1573123587	Thursday, 07-Nov-2019 10:46:27 UTC	Today
01111111111111111111111111111111	2147483647	Tuesday, 19-Jan-2038 03:14:07 UTC	y2k38
10000000000000000000000000000000	-2147483648	Friday, 13-Dec-1901 20:45:52 UTC	One day after Guglielmo Marconi received the first trans-Atlantic radio signal, sent 1,700 miles from Poldhu in Cornwall, England to Signal Hill, St. John's in Newfoundland in Canada

y2k38

Why do you need testing in production? #1

- Or just imagine one of your stateful services becomes inconsistent and starts failing for all customers

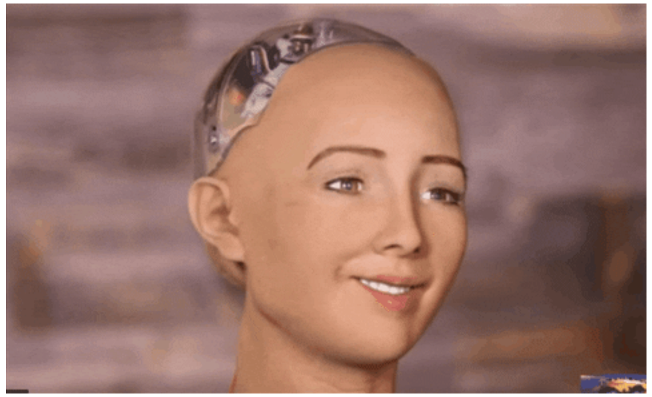
- At that moment, lots of customers will call you saying:
  - Cannot login
  - Cannot register
  - Cannot change profile picture
  - Cannot launch the app
  - Cannot order chocolate milkshakes takeaway

Why do you need testing in production? #1

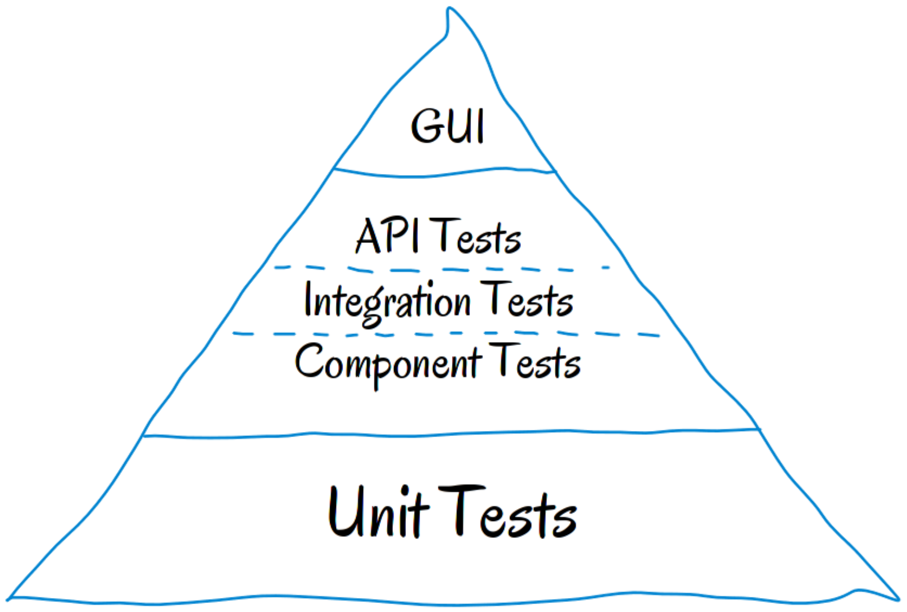
- What do you prefer:

Real angry **paying** customers calling you because what they paid for is not working

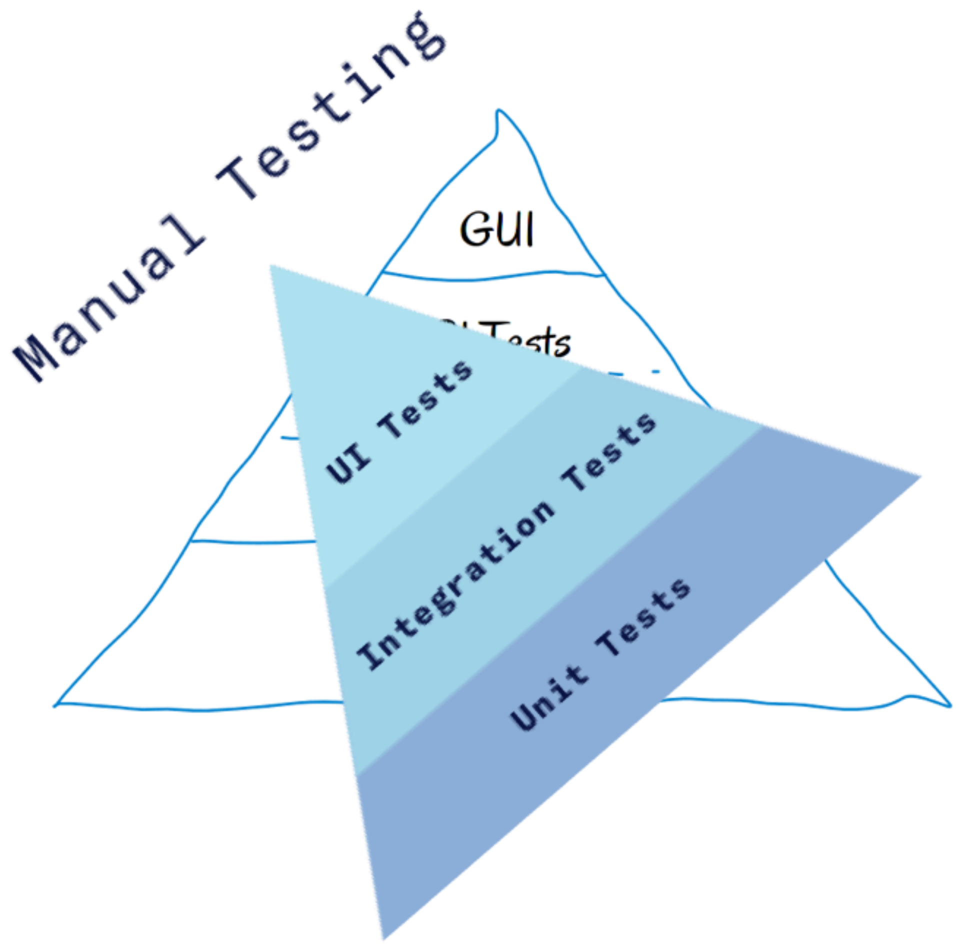
An automated process acting as the first real user noticing an issue and notifying the team right away



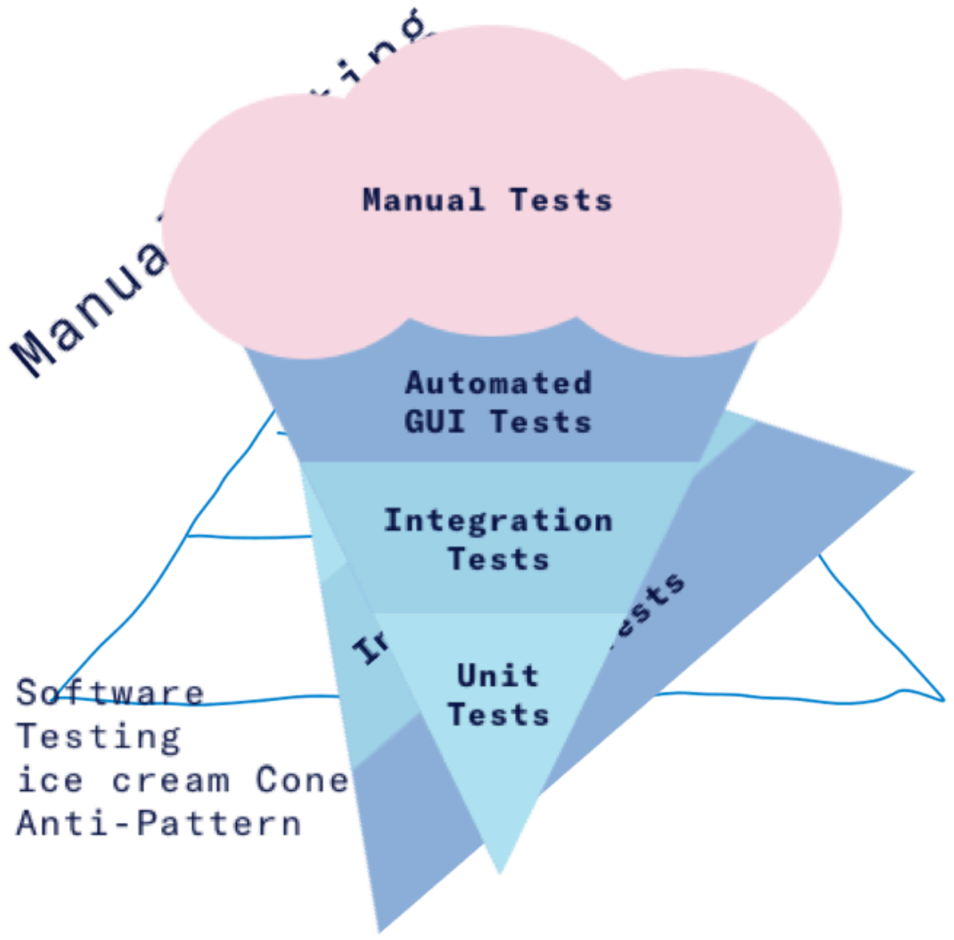
What is the right testing level against production systems? #2



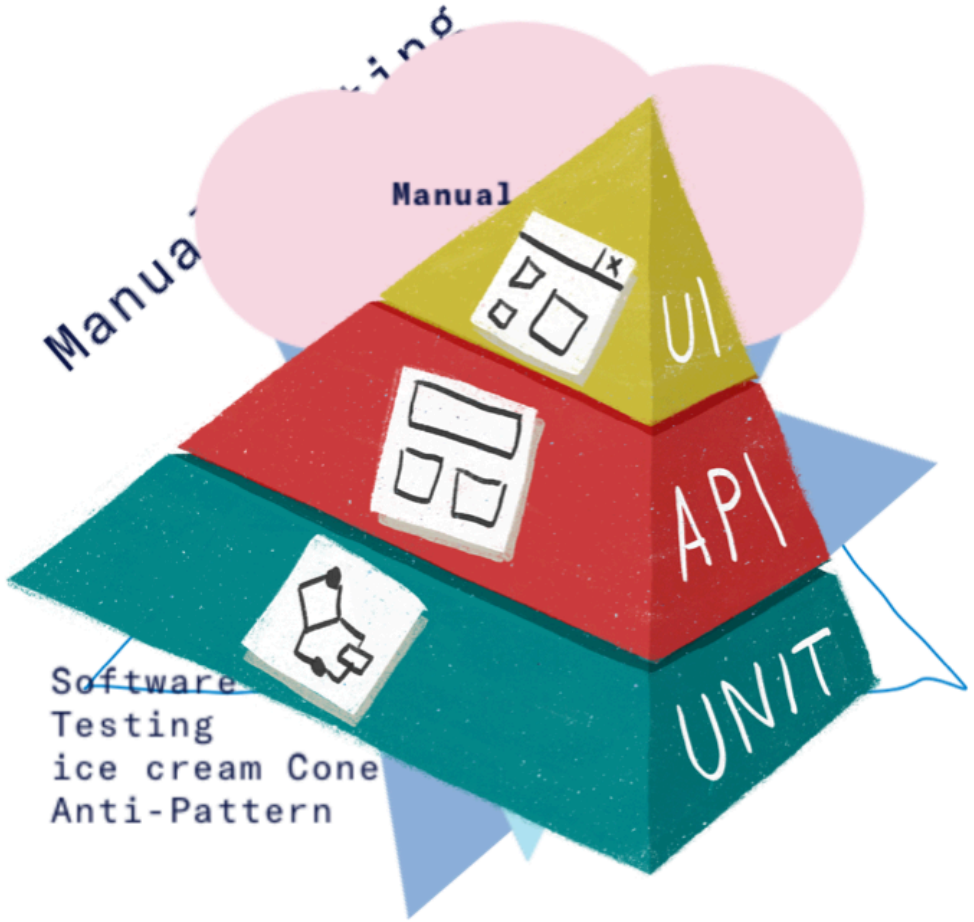
What is the right testing level against production systems? #2

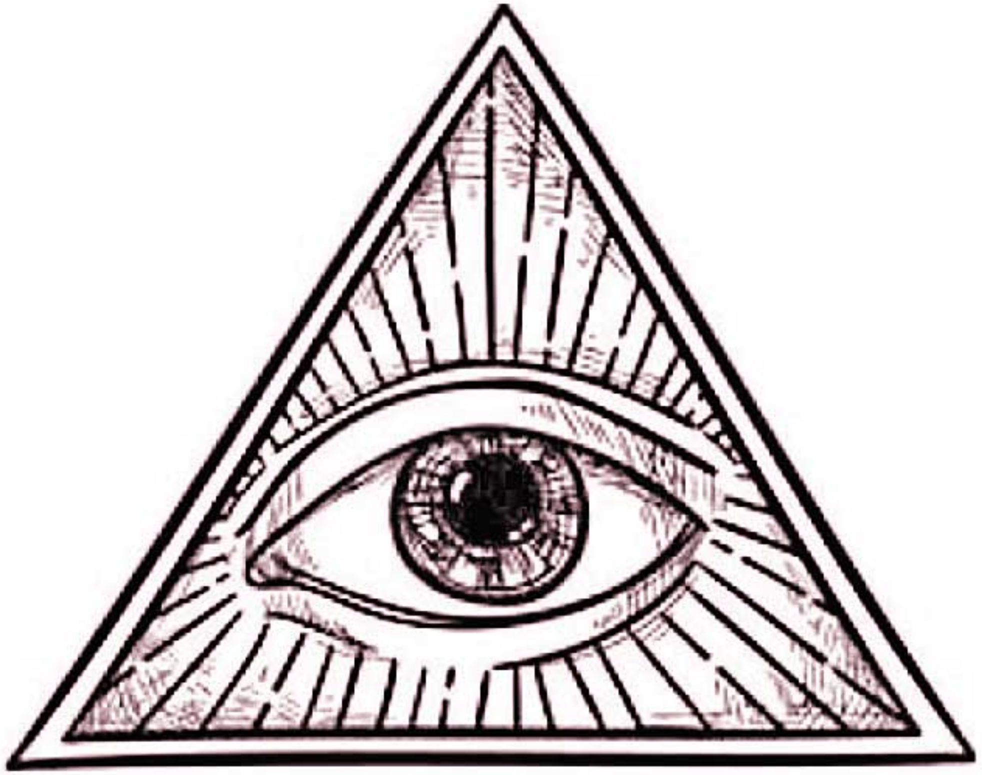


What is the right testing level against production systems? #2

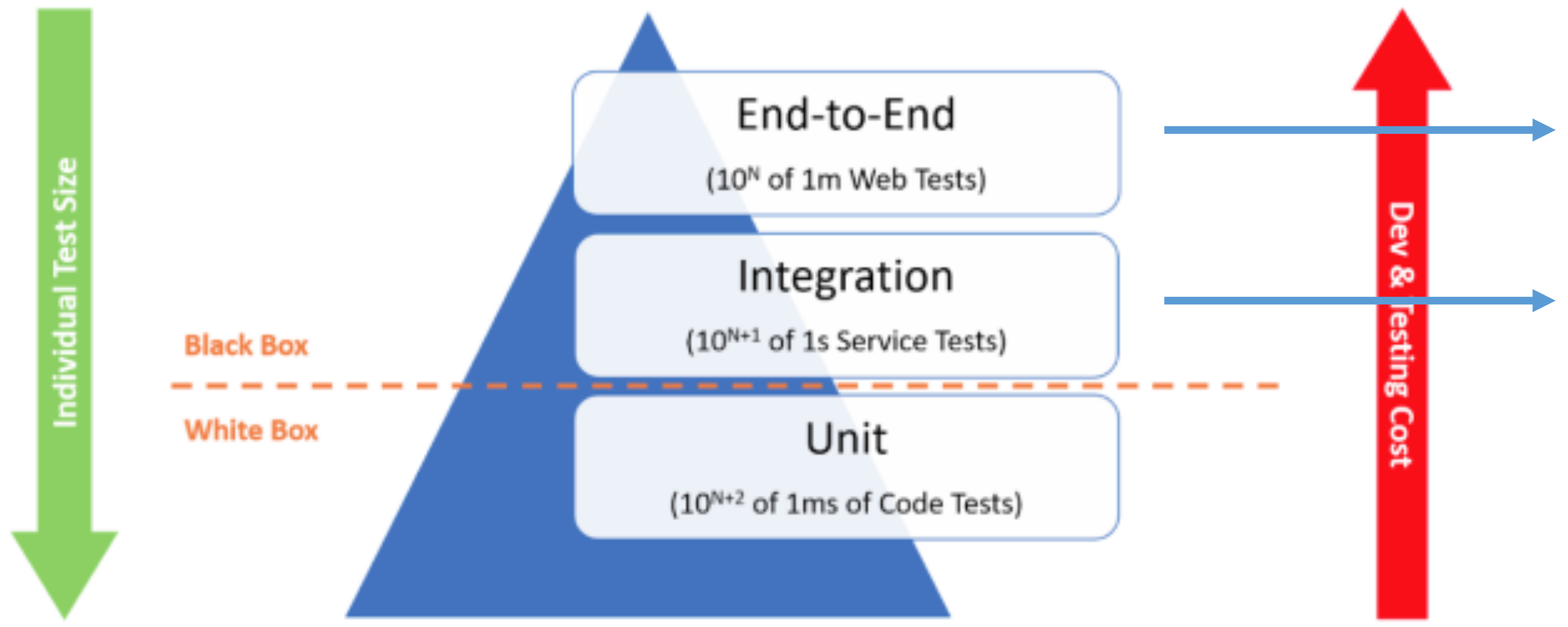


What is the right testing level against production systems? #2





What is the right testing level against production systems? #2



Minimum unit user sees

- User intent
- User action
- User journey

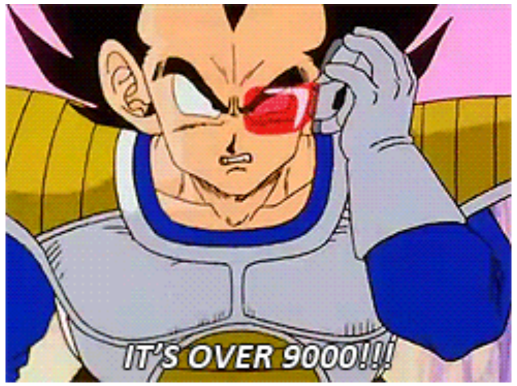
Minimum unit after deployment

https://automationpanda.com/2018/08/01/the-testing-pyramid/

How can we define an end to end test? #3

- More than 1 million connected machines performing automated actions like scheduled cleans
- Users all over the world
  - actions as simple as renaming your cleaning robot from ‘Dyson 360 Eye’ to ‘RoboMop9000’

- Fast
- Easy to run
- Extendable
- Behavior focused

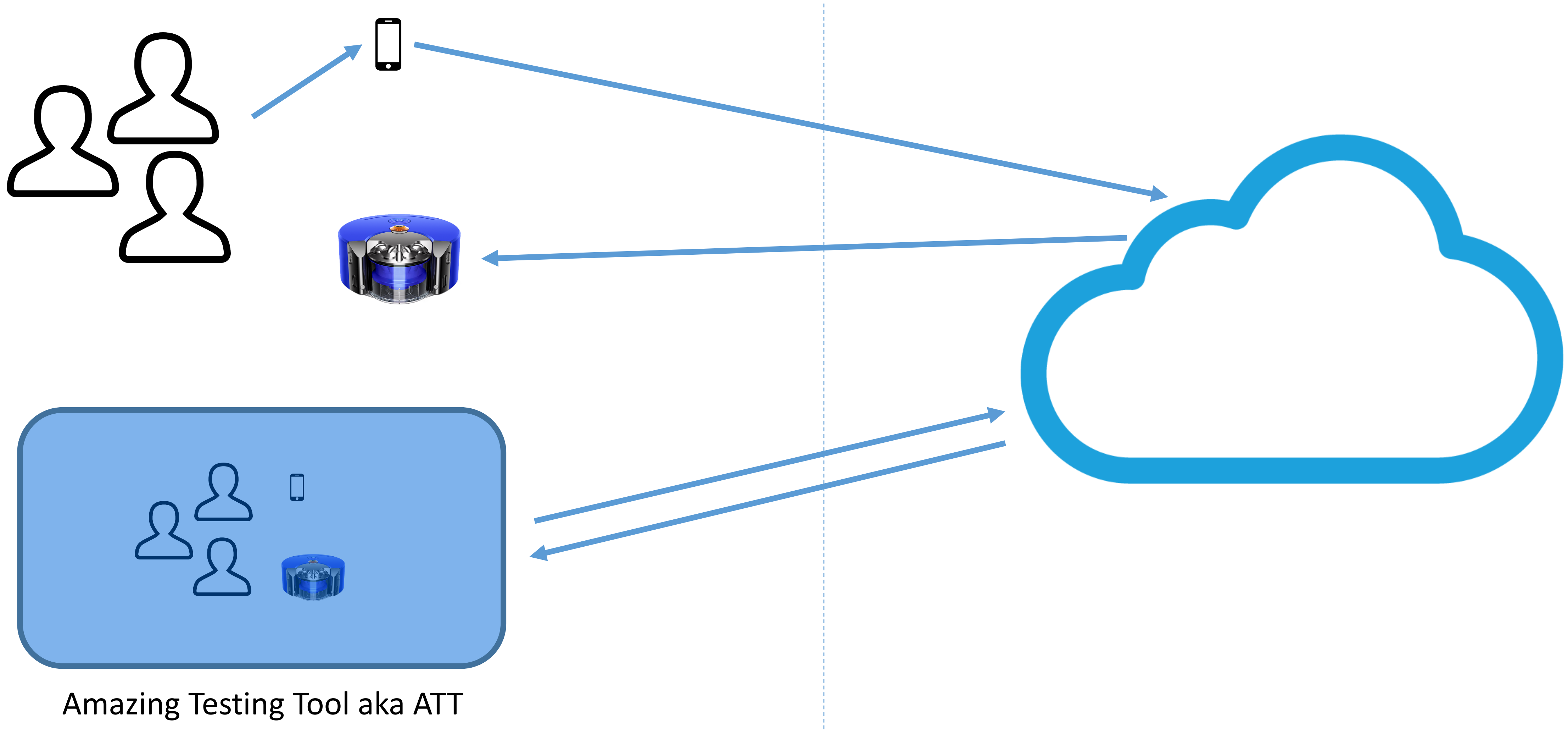


How can we define an end to end test? #3

- Given an existing user with this configuration
- And the user logs in
- And the user changes robot name to ‘RoboMop9000’
- Then the robot name has changed to ‘RoboMop9000’

https://themvacuums.com/79-best-robot-vacuum-names/

How can we define an end to end test? #3



How can we define an end to end test? #3

- Combination of:
  - CucumberJS
  - Request library
  - Assert library
  - Lots of single steps defining calls to APIs and data transformations
    - Mocking machine (virtual machine)
    - Mocking user (virtual user)
    - But going to the cloud and back
  - Grouped together under user journeys or scenarios

How avoid disrupting real users? #4

- If your test is too close to real users and real actions…

How avoid disrupting real users? #4

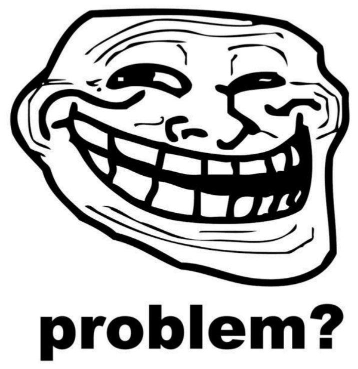
- Spacing these end to end tests
  - The more often you run them the faster you will notice an issue
  - Noise
  - Slowing down your production systems
  - Balance

- Virtual users registered in production like real users with their own configuration

How to keep your tests out of statistics? #5

- But testing in production can also skew some of the metrics you record like:
  - Number of users interacting with your system
  - API usage
  - Error count

- Even mixed logs
  - How to discern what was part of a test or part of a real interaction?



How to keep your tests out of statistics? #5

- Correlation id

  fffffff-0000-0000-0000-000000000000

- User agent

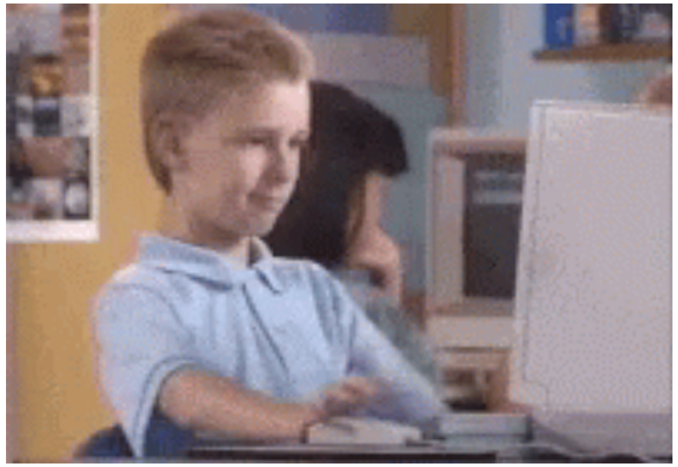
  AppId/AppVersionId (Language=LanguageNameAndOptionallyVersion)

- Custom HTTP headers

  TestToolVersion=1.1.2

Cleanup? #6

- Running tests in production 👍🏼



Cleanup? #6

- But… what about the all the fake data they generated?

Cleanup? #6

- Automatically clean as much testing items as possible after each test
- Objective: Leave the system as you found it, as if the tests never ran there

Cleanup? #6

- Helps starting tests from a clean state
  - Avoiding false negatives
- Avoids cluttering your production databases
- Avoids exhausting other limited resources
- Avoids making real data more difficult to retrieve and search

That’s interesting, new questions!

- Can your E2E Tests pass and you still have problems?
- By their nature E2E Tests span multiple services and teams

  Who is responsible for them and how do teams collaborate to write them?
- How do E2E Tests compare to dark or canary Deployments?

  Do you need both?
- How do you avoid E2E Tests becoming fragile?

Recap

- Testing in production is good and necessary 🎉
- Think like an user, act like an user 🧠
- CucumberJS is a good start
- Think about your system’s capacity 📈
- Mark test intents to differentiate them from real user interactions
- Clear your test data and reset connections/status after each test

That’s all folks

https://jorgemarin.me
@chipironcin

Before Q&A

THE FINAL

BIT-BORING-TALK-BUT-WILL-SMILE-FOR-THE-GRAM

SELFIE WITH ATTENDEES!

and you?
are you brave enough to test in production?

Questions round

https://jorgemarin.me
@chipironcin