Testability - lessons learnt the hard way

Quality Meetup 14.10.2020

Tomasz Wierzchowski

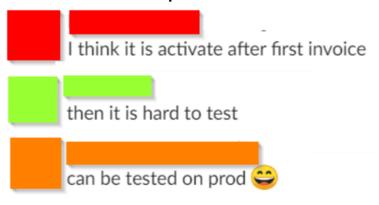
About me

- 10+ years in QA
- QA Lead @ Northmill Bank
- Program commitee @ Quality Excites
- in /tomasz-wierzchowski



When you hear:

- We need more time for tests (just before the deadline)
- We have legacy system, thus it cannot be tested
- This feature can only be 'tested' on production



When you hear:

- Our system is special one (special domain / business / technology / etc.),
 thus it cannot be tested
- There is too many bugs on production

Then you probably have issue with testability

Testability? What?

Theory - <u>ISTQB</u> / <u>ISO 25010</u>

The degree of effectiveness and efficiency with which tests can be designed and executed for a component or system.

In practise

How fast and easy can we test?

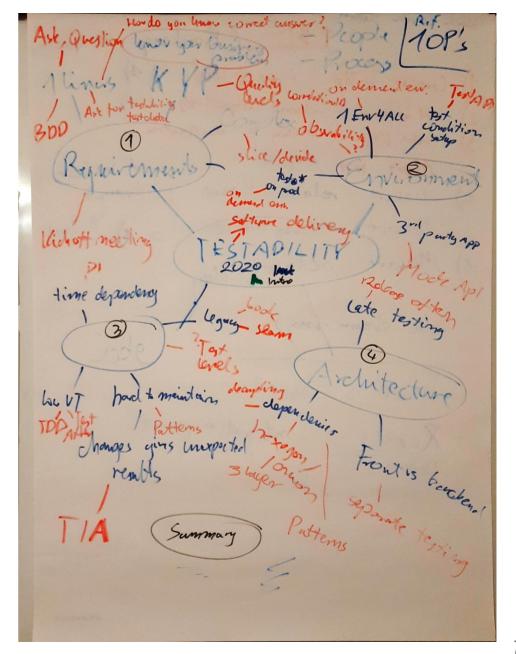
Why should you care?

- Speed
- Efficiency
- Control

Time is money

Challenges:

- Requirements
- Environments
- Code
- Architecture



First challenge (requirements)

- One liner requirement
 - Do the same system as in X, but for Y country
 - Make sure the actual calculations works as in project A
- Team doesn't understand the business problem
- Complex/complicated systems are hard to test

- Ask & question about the domain and possible solutions
- Do small features (Decomposition & INVEST!)

HOW TO SPLIT A USER STORY PREPARE THE THE SPLIT **WORKFLOW STEPS INPUT STORY** Can you take a thin slice through the workflow first and enhance it with more stories later? Are the new stories Does the big story satisfy Can you split the story so you do roughly equal in size? INVEST* (except, perhaps, small)? the beginning and end of the workflow first and enhance with stories **OPERATIONS** from the middle of the workflow? Is each story about **BUSINESS RULE** 1/10 to 1/2 of your velocity? Can you split the operations Combine it with another story VARIATIONS into separate stories? or otherwise reformulate it to get Try another pattern on the a good, if large, starting story. Can you split the story so you original story or the larger do a subset of the rules first and Do each of the post-split stories. Does the story describe enhance with additional rules later? stories satisfy INVEST? PERFORMANCE a workflow? Is the story size 1/10 to Does the story include multiple Could you split the story % of your velocity? Try another pattern. operations? (e.g. is it about "managing" to just make it work first and or "configuring" something?) Are there stories you then enhance it to satisfy the Does the story have a variety of can deprioritize or delete? non-functional requirement? business rules? (e.g. is there a domain You're done. Continue, You term in the story like "flexible dates" Try another pattern. need to split it. that suggests several variations?) Does the story get much of its You probably have waste complexity from satisfying in each of your stories. Is there an obvious story non-functional requirements like **APPLY THE** to start with that gets you performance? **SPLITTING** early value, learning, risk IN DATA Does the story do the same mitigation, etc.? **PATTERNS** thing to different kinds of data? Can you split the story to Does the story have a simple Try another pattern to process one kind of data Could you split the story to core that provides most of the first and enhance with the see if you can get this. do that simple core first and You're done, though you value and/or learning? other kinds later? enhance it with later stories? could try another pattern to see if it works better. Does the story have a SIMPLE/COMPLEX When you apply the obvious complex interface? split, is whichever story you do first the most difficult? **BREAK OUT A SPIKE** Could you group the later stories and defer the decision Are you still baffled about Does the story get the same about which story comes first? how to split the story? kind of data via multiple interfaces? * INVEST - Stories should be: MAJOR EFFORT Independent



Negotiable

Valuable

Estimable

Testable

Small

Visit http://www.richardlawrence.info/splitting-user-stories/ for more info on the story splitting patterns Copyright @ 2011-2013 Agile For All. All rights reserved.

VARIATIONS

Can you split the story to

handle data from one

interface first and enhance

with the others later?

Is there a simple version

you could do first?

Can you find a small

piece you understand

well enough to start?

Write that story first,

build it, and start again

at the top of this process

Can you define the 1-3

questions most holding you back?

Write a spike with those

questions, do the minimum to answer them, and start again at the top of this process

Last updated 3/26/2013

Take a break

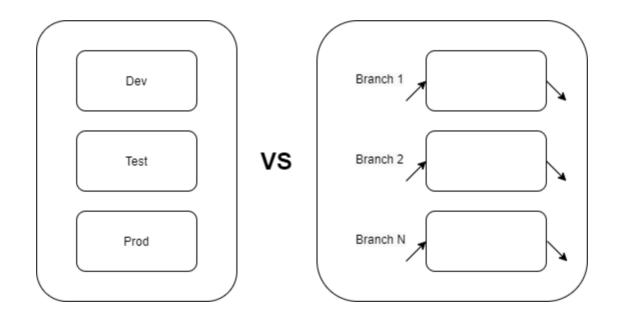
and try again.

- Prepare test data during feature planning
- Think of BDD / ATDD / Specification By Example

Second challenge (environments)

- Everyone is using the same environment
- Hard to setup system in required test condition
- 3rd party apps
- Tests on production*

- Script creation of environments (Infrastructure as a code)
- Make environments on demand

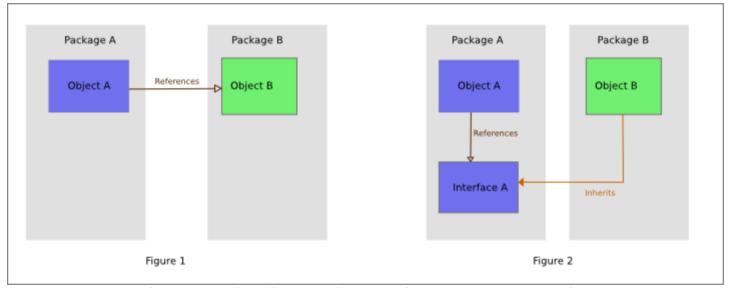


- Fake services
- Use dedicated API for testing purposes
- Log & monitor

Third challenge (code)

- Changing code in one area breaks functionality in another
- Team doesn't have confidance in changing code
- 'Hard' to write unit test -> Low level of unit tests -> low code coverage
- Testing time-dependent features
- Legacy code

• Use SOLID rules (dependency inversion)



https://en.wikipedia.org/wiki/Dependency_inversion_principle

• Don't mix objects creation with business logic

```
(...)
public Offer CalculateOffer()
    var checker = new scoreCalculator(); // Wrong !
    var score = checker.calculate();
    if (score >= 90)
        MaxLoanAmount = 5000;
    if (score >= 70)
    (\ldots)
```

• Don't mix objects creation with business logic

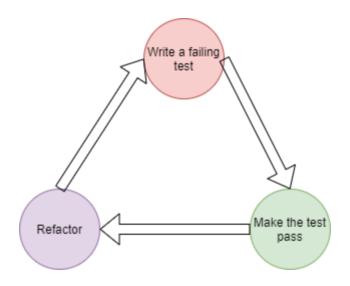
```
(...)
public Offer CalculateOffer(IScoreCalculator scoreCalculator)
   var score = scoreCalculator.calculate(); // Better
    if (score >= 90)
       MaxLoanAmount = 5000;
   if (score >= 70)
    (...)
```

• Make abstrtaction for time, and all external/system dependencies

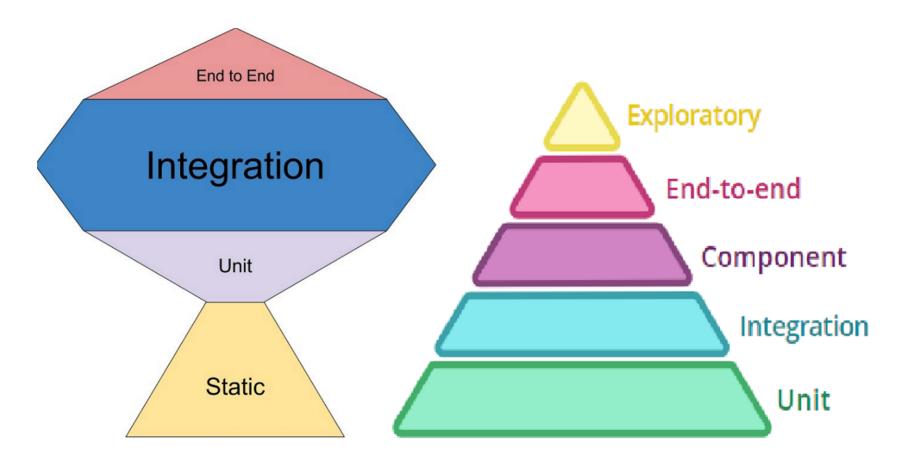
```
public interface IDateTimeService
{
    DateTime Now {get;}
}
```

```
public class FakeDateTimeService : IDateTimeService
{
    public DateTime Now {get; set;}
}
(...)
var longTimeAgo = new FakeDateTimeService {
    Now = new DateTime(2019,10,14) }
19
```

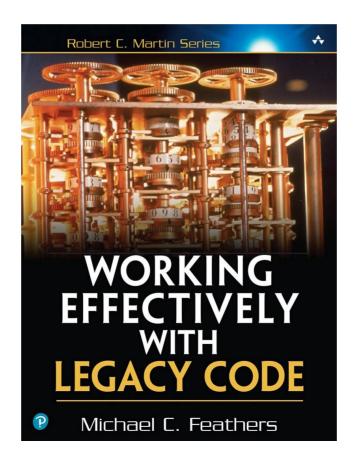
• TDD



Know & use your test levels



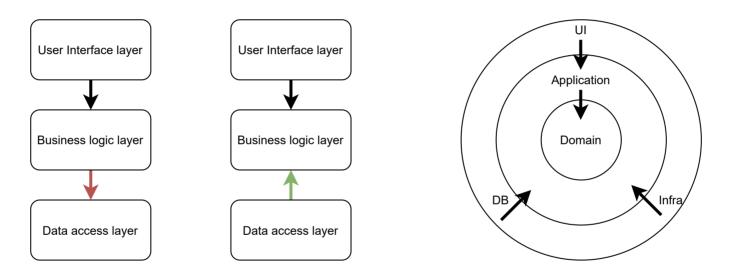
 Legacy is not an excuse – read book "Working effectively with legacy code" Michael Feathers (published in 2004!)



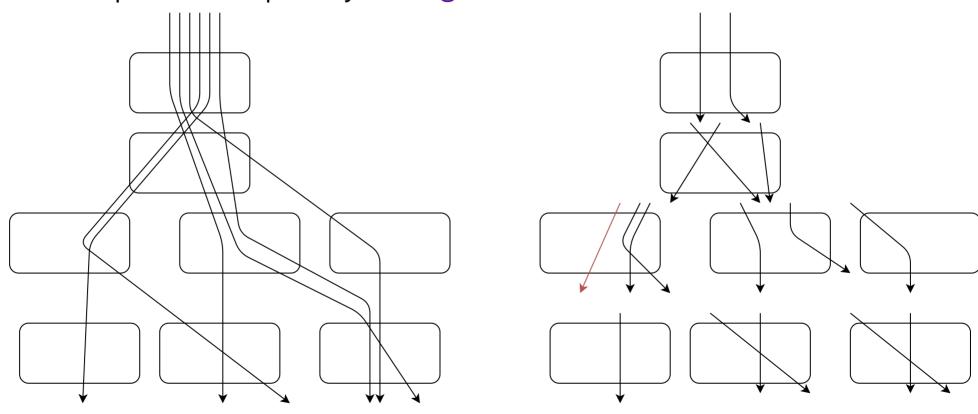
Fourth challenge (architecture)

- To test component A you have to run components B & C (and a few more)
- Cannot test front-end without testing whole application
- Tests can be done only at the end of the project schedule

• Keep business rules decoupled from UI / Database / Servers / etc.



• Test components separetly <u>#integratedtestsareascam</u>



- Release often
- Cooperate with developers/architects

What to do when you're back in project?

- Know your domain & gain expirience
- Question requirements status quo
- Control aplication and environments
- Enhance clean code & clean architecture
- Promote testability on all levels



Additional materials:

- <u>User Stories Applied</u>
- https://www.satisfice.com/download/heuristics-of-software-testability
- https://www.slideshare.net/AshWinter/10-ps-of-testability
- https://www.slideshare.net/AshWinter/architectural-testability-workshop-for-test-academy-barcelona?next_slideshow=1
- <u>Aslak Hellesøy Testable Architectures</u>
- https://understandlegacycode.com/
- https://martinfowler.com/articles/microservice-testing/
- http://misko.hevery.com/attachments/Guide-Writing%20Testable%20Code.pdf
- https://agileforall.com/resources/how-to-split-a-user-story/
- https://blog.thecodewhisperer.com/permalink/integrated-tests-are-a-scam

Questions?

We are looking for

QA Engineer











10 000 - 15 000 PLN (gross)

