When quality is just a cost

Useful approaches to testing

Jani Grönman

17.10.2015 Tampere Goes Agile 2015

Introduction: you

Introduction: me

15 years of experience in various positions in software development

First a developer, then manager and then again a developer

Quality and testing related things

Always trying to learn something new

Twitter: @JGronman

(https://www.linkedin.com/profile/view?id=12221179)

Attitudes and experiences

”Testing is expensive.”

So is development. Can you afford not to test?

What if you thought of it as an investment?

”Testing is a buffer.”

Are you being honest to your customer and to yourself?

”10 percent of the development budget is enough for testing.”

Have you thought that the budget might depend on the product you are building?

”Your code coverage is just 40%, so it’s bad quality.”

I can make it 100% if you like. It will still have bugs.

”We have tested every test case. Therefore the product is completed.”

So you went through every line in the manuscript and you are sure nothing was missed?

What if there was something in between those test cases?

”What automation? I can quickly click through the app myself.”

Ok, fine. But can you do it every time the product ships?

What if we tighten the delivery cycle to four weeks?

Or to one week?

”The best quality of a tester is the ability to follow orders.”

If that’s what you like, imagine what you could get with a thinking tester.

”We don’t need testers, the developers can do it. The code came from them after all.”

Because subjectivity is the best tool to assess your work as a whole?

”Those are not tests. Where are all the step definitions?”

Is it really worth the while to break test cases into gazillion steps so any guy out of the woodwork could test it?

Wouldn’t it be nice if you knew enough about the domain in question so you could start actually testing instead of writing about it?

What do you think?
Sound familiar?

A joke

Schools of testing

Controversial topic, lots of discussion going on (some of which not civilized)

Term coined by Bret Pettichord, Cem Kaner and James Bach in 90’s

Following definitions are adapted from Bret Pettichord’s presentation.

What is a school (of thought)?

Defined by: intellectual affinity, social interaction, common goals

Made up of: hierarchies of values, exemplar techniques, standards of criticism, organizing institutions, common vocabulary

Not defined by: common doctrine, specific techniques

Analytical

Testing is a branch of computer science, mathematical approach

Methods: code coverage, techniques and metrics

Requires detailed requirements. Testing ensures that the product corresponds to these requirements. Anything else isn’t testing.

Approach to risk: calculate the reliability

Approach to sertifications: no. University degree preferred.

Standard (or Factory)

Testing must be managed, predictable, repeatable, planned, cost-effective

Encourages standards, best practices, and certification

Testing validates the product and measures development progress

Resistance to change (complicates tracking), require clear boundaries between testing and other activities (start/stop criteria)

Approach to risks: key risk is failure to meet schedules (project risk)

Approach to certifications: yes (testers are easier to manage, hire and train)

Quality

Quality is a process and testing is part of this process

Testing determines whether development processes are being followed.

Product is not ready until QA says it’s ready. Testers are the gatekeepers.

Methods: rules, inspections, reviews

Key Question: Are we following a good process?

Approach to risks: uncover project risks, prove that project is out of control

Approach to certifications: yes (they increase status)

Context-driven

Testing is a skilled activity, testing is part of development. Testing finds bugs. A bug can be in the screen as well as in business cases. Testing provides information to the project

Methods: context based mix of different techniques, exploratory testing, automatization, checking vs testing

Expect changes. Adapt testing plans based on test results

Key Question: What testing would be most valuable right now?

Approach to risks: produce information to the team about risks. Orientate testing according to those risks

Approach to certifications: no (they test doctrine, not actual skill)

Agile

Software is an ongoing conversation, testing tells us that a development story is complete

Methods: automated testing, unit tests, TDD

Presumes that the developers provide test automation

Approach to risks: fail fast

Approach to certifications: no

What is software quality?

My thoughts

“Software quality is measured with your customer's success, not development project metrics and quality processes. While quality processes have their use, most organizations use them to measure wrong things and because of this they often hinder the development that they should foster.

Good quality is the right tools for the right job. Just as you have to choose the correct tech stack for the project, you need to choose the right set of QA tools for the job. One size fits all -solutions are evil.

Good quality is not about cloning practices, it's about taking responsibility.”

Is quality just a cost?

Is it right to save on the quality costs by not testing?

What would be a better way? What is the cost structure of quality?

Cost of doing bad quality

Internal costs (delays, service breaks, rework, difficult change management, unnecessary meetings)

External costs (bug reports, reclamations, warranty work, declining customer satisfaction)

The 1:10:100 rule (requirements, development, production)

Agile way of failing fast reduces risk and costs

Is quality an investment?

It is difficult to define ROI to quality

What is ROI? Quality is not definable in the same quantitative metrics like amount of profit or sold products

Testing costs and testing more costs even more

It is the not testing that can prove to be the most costly option

The problem is in the *narrative*

Example: You need a new shirt.
A shirt is on -30% sale.

How much do you save by buying the shirt?
Nothing, you just used less money than could have normally.

Most money could be saved by not buying the shirt at all.
But then you wouldn’t have a new shirt.

What do you tell when come back home: do you tell how much money you spent or how much you saved?

Both options are valid: *They just have different narrative, depending on the motives of this person.*

The same goes for all economic reporting in general

The context and the way it is presented always serves some purpose

Testing has its price

One of the outcomes of testing is that it produces more information about the product

This can assist different stakeholders in their decision making

Tools like automatization can bring new views to them or make decisions easier, but direct monetary gain is hard to prove.

By testing less you gain nothing

But you can ensure the given budget is spent as wisely as possible to make the most of it

This is the link between testing costs, schools of thought and the goals of testing

Costs of doing good quality

Prevention measures: architecture, responsivity to change, quality thinking, professional pride

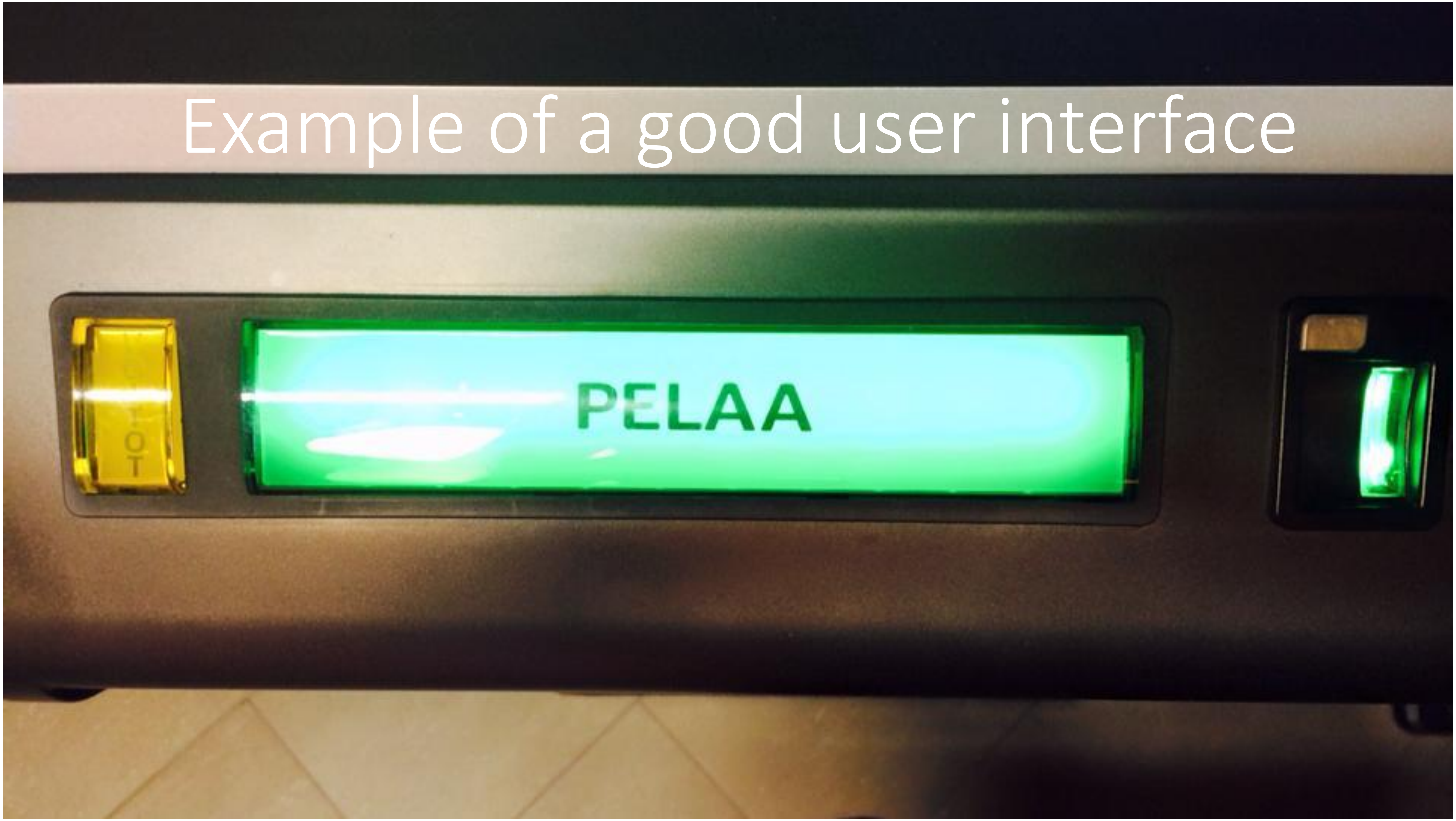
Assessment measures: testing, (code) inspections, risk analysis

Lifecycle approach: money spent in development is money saved or even gained during product lifecycle

Better customer satisfaction, easier to make changes, lower warranty costs

Another joke

Example of a good user interface



”One size fits all” solutions are evil

What is the skillset of a good tester like?

Skills: soft ones

Be a good listener

Make good notes

Be brave and question everything

Be nosy (but not obnoxious)

Rapid testing, exploratory testing

Skills: hard ones

SaaS and cloud services: usetrace.com, loadimpact.com, testcloud.io

Subcontract skills you don’t have

Automation: jMeter, Robot Framework and libraries, Cucumber, SpecFlow, Rspec, Selenium

Specialist tools: Galen Framework, Sikuli, browser tools, OWASP tools (ZAP), gauntlt.org

Low level: programming langs, scripting

Skills: weird ones

Befriend DevOps, we really can learn a lot from them

Have a knack of breaking things or make them do stuff they weren’t supposed to

Culture over Rules

Best practices are like a sales person telling you ”I have this same model home” or ”My brother has one too.”

Standards are best served with a hint of doubt

Culture eats Rules

Om nom nom