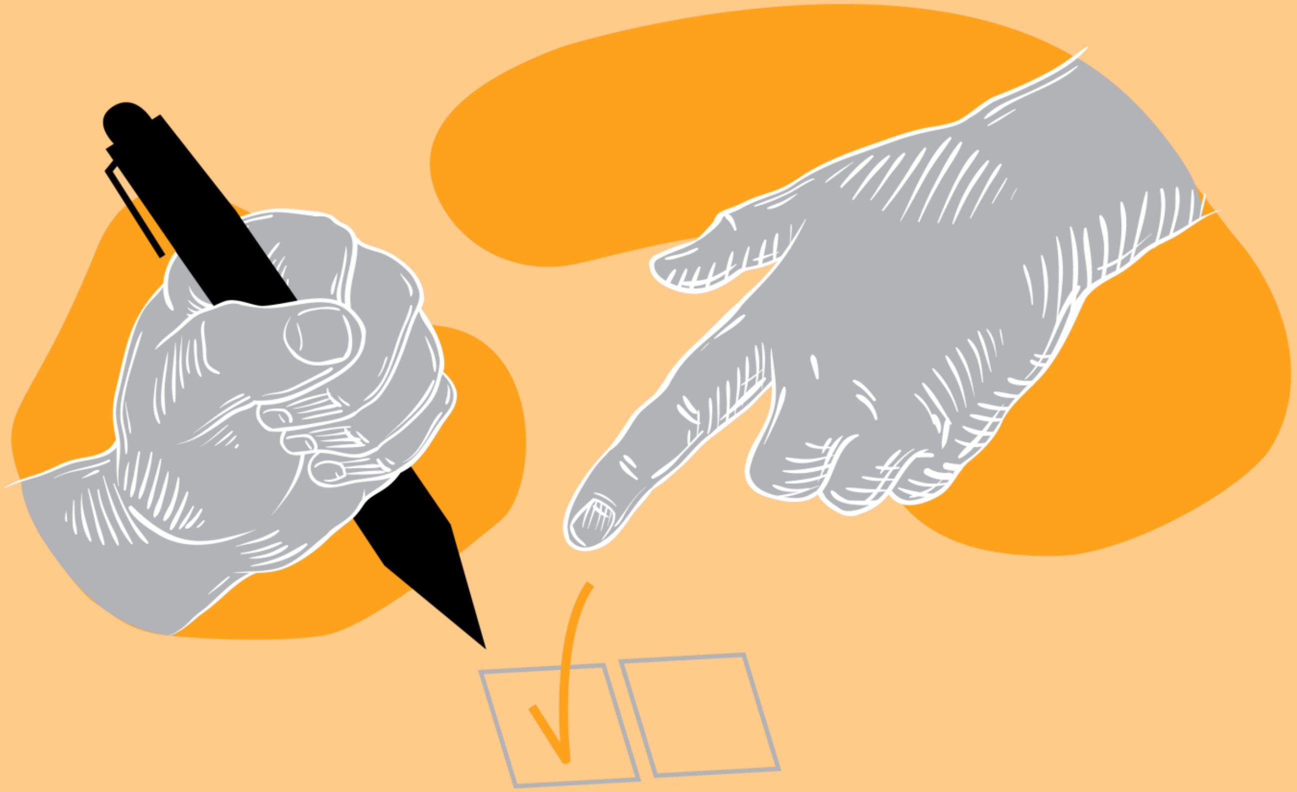
Quality Assurance

Mette Hande and Tina Syversen
13.09.2023



About us

Mette Hande

Economics NTNU

Senior Test Consultant

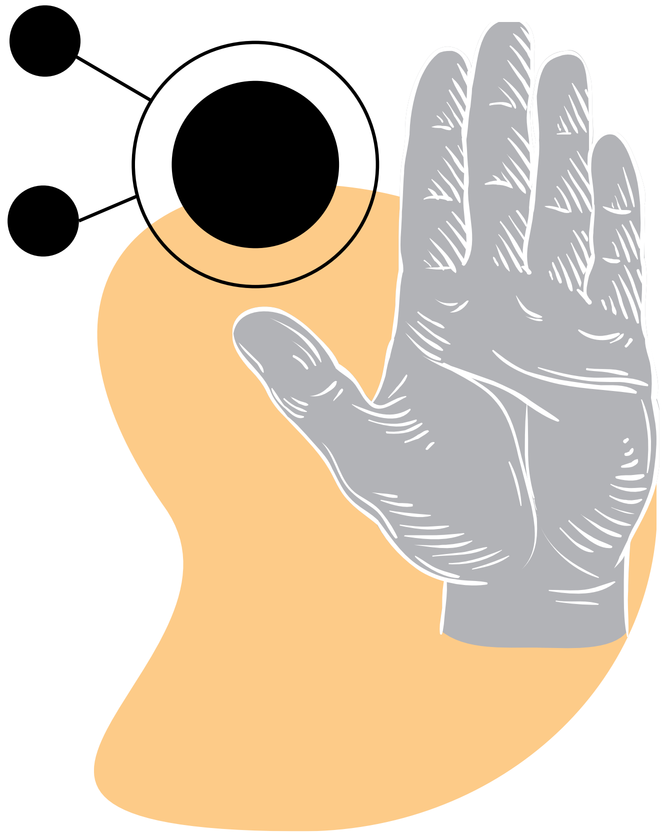
Tina Syversen

Computer Science NTNU

Quality manager

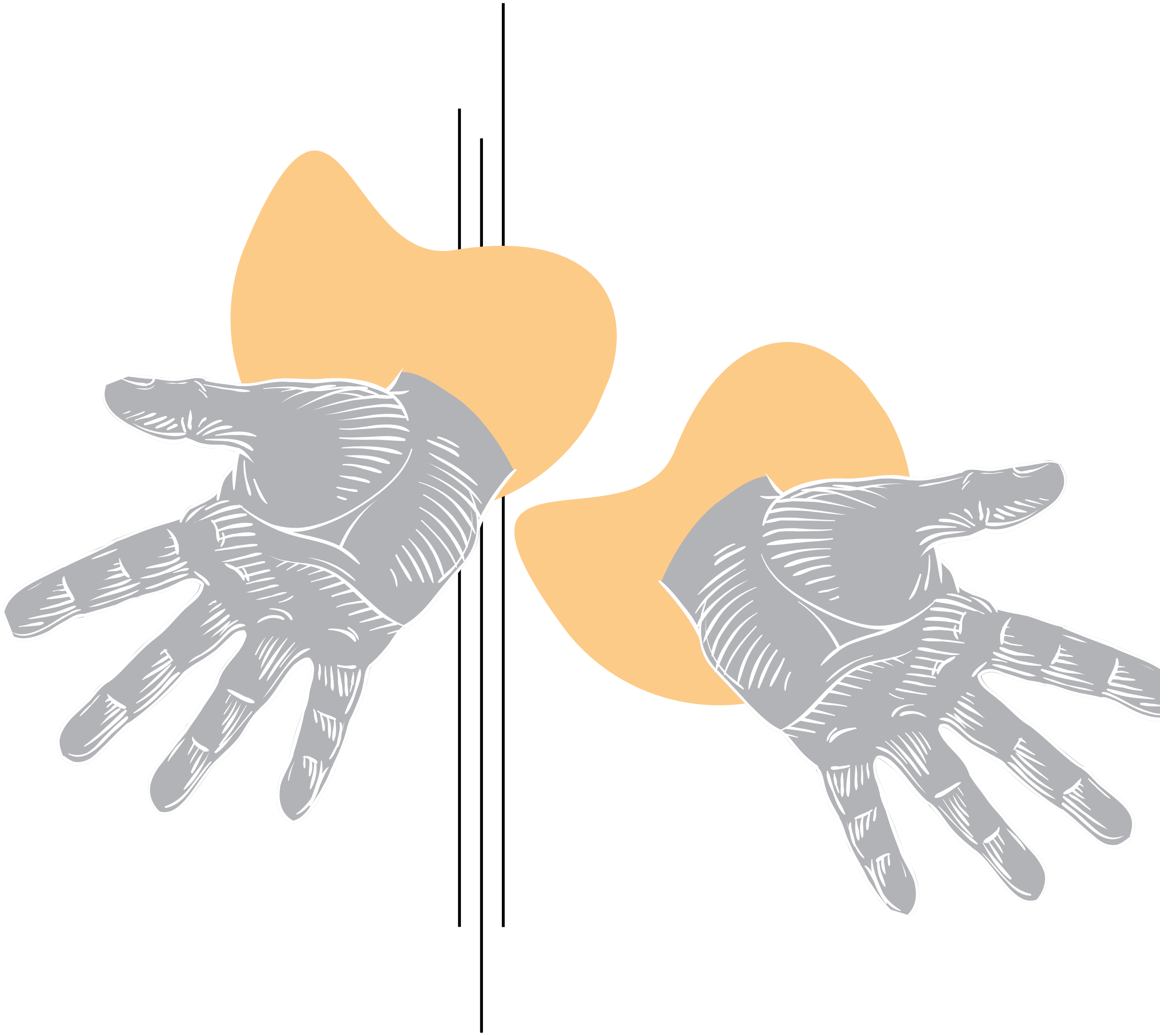
Learning Objectives

Learn about the complexity and challenges of Quality in software projects and how to deal with them



Question

What is quality?



Garvin's Five Definitions

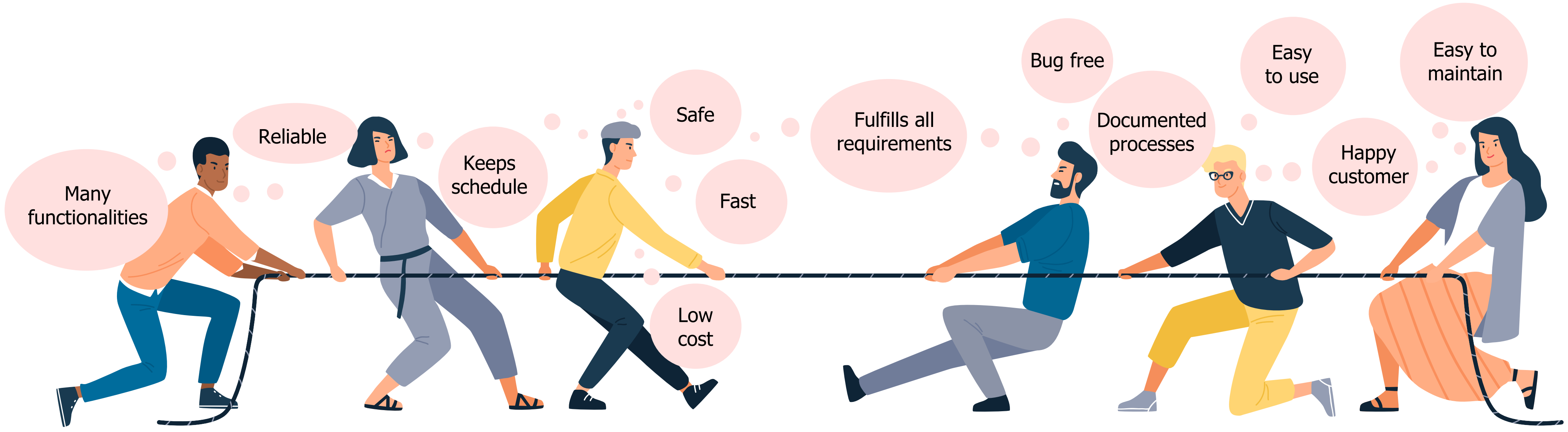
1 **The transcendal perspective:**
Quality is something you feel or that we learn to recognize through experience

2 **The product perspcetive:**
Quality is precise and measurable

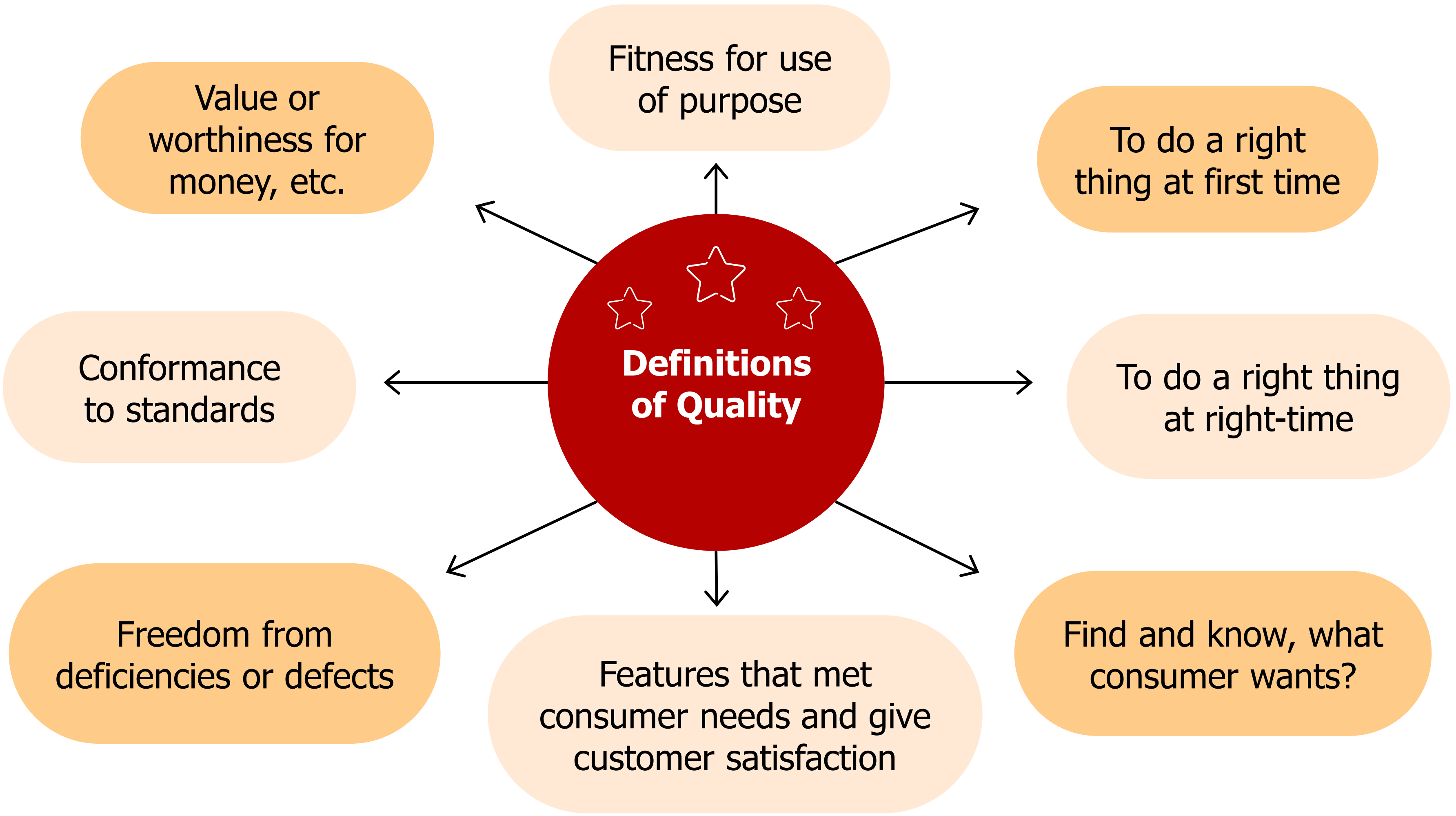
3 **The user perspcetive:**
Quality is defined by In terms of fitness for use

4 **The producer perspcetive:**
Quality is the degree a product/service shows conformity with a project or specification

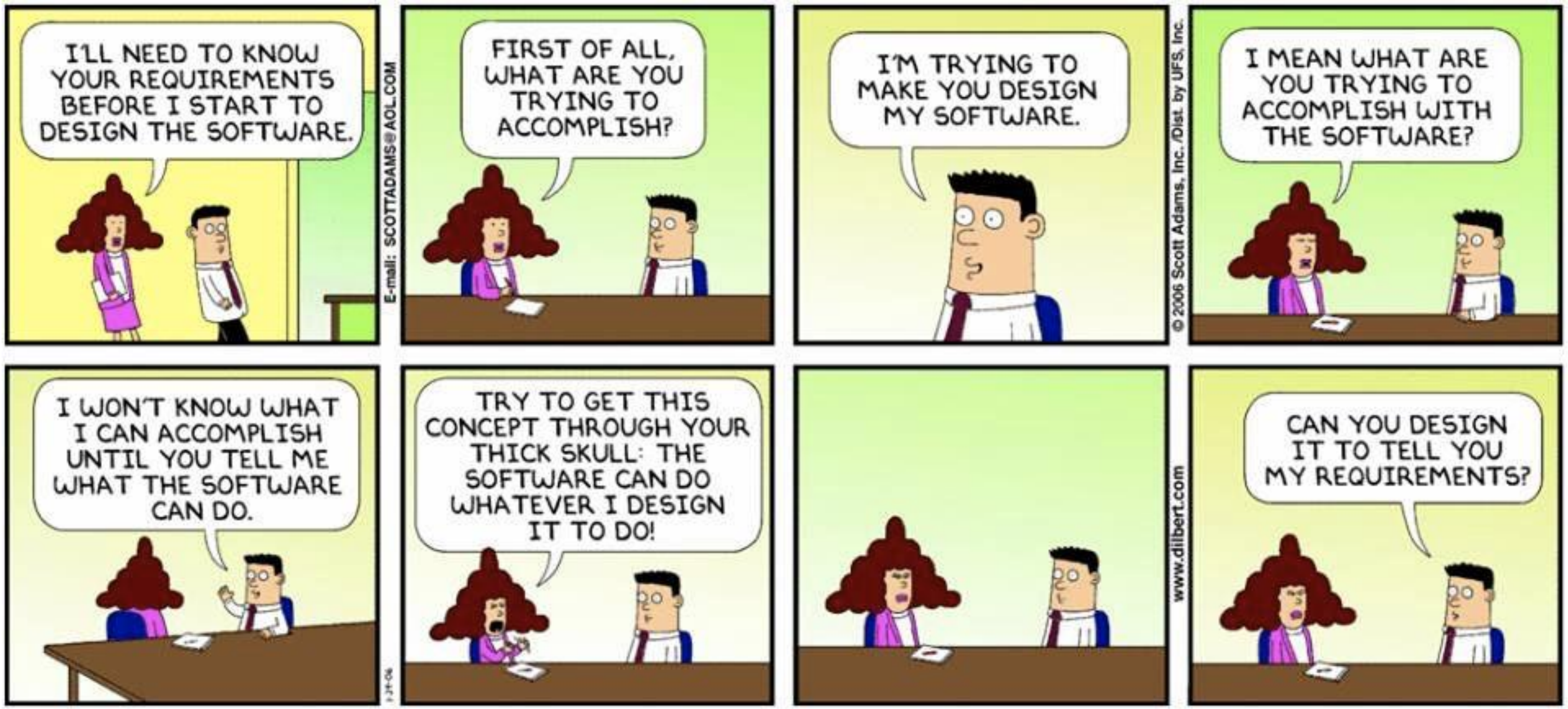
5 **The value perspcetive:**
Quality is defined in terms of cost and prices. Have you recieved what you paid for?



Possible definitions

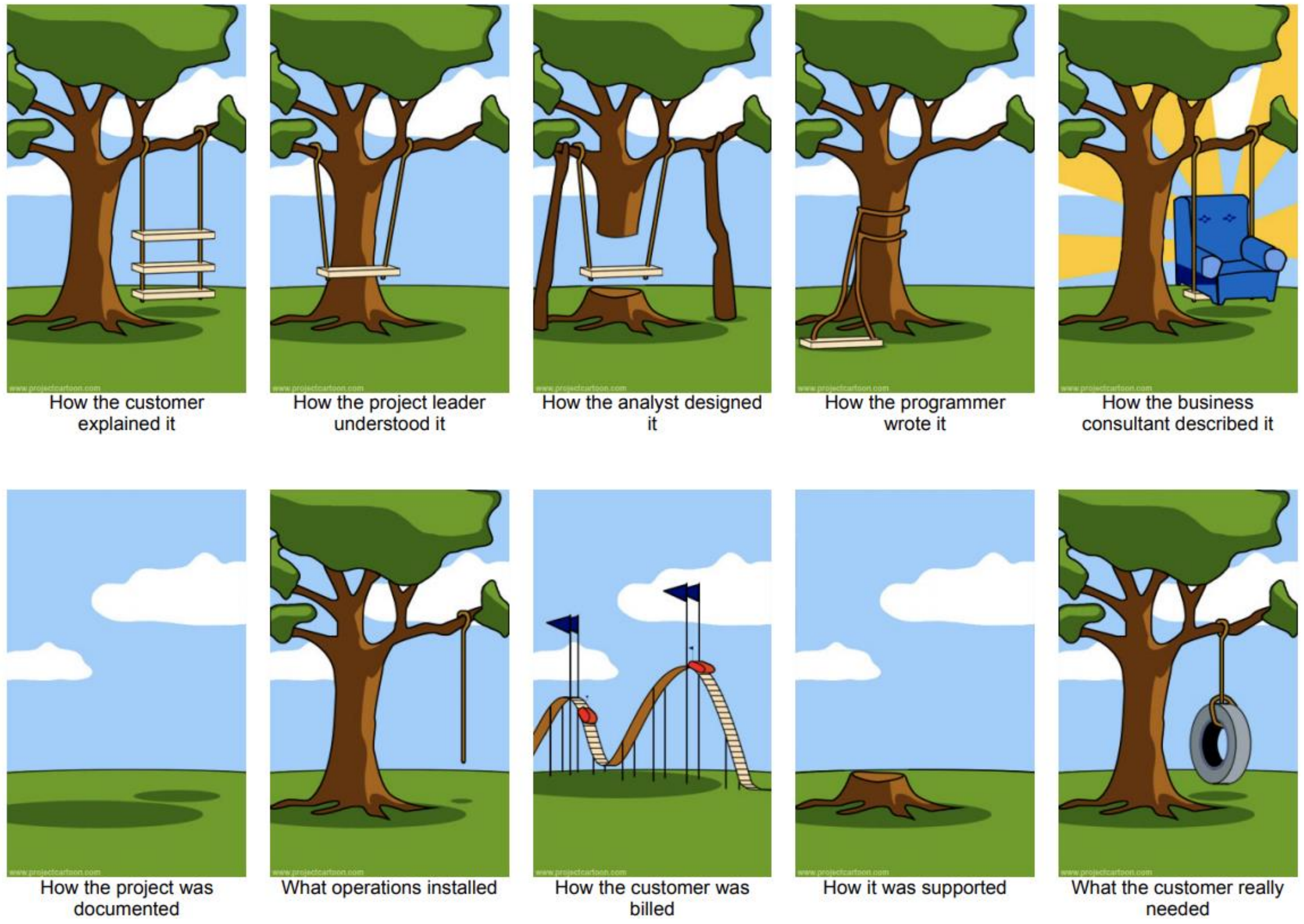

Definitions of Quality
Fitness for use of purpose
To do a right thing at first time
To do a right thing at right-time
Find and know, what consumer wants?
Features that met consumer needs and give customer satisfaction
Freedom from deficiencies or defects
Conformance to standards
Value or worthiness for money, etc.

Do the customer know what they want?

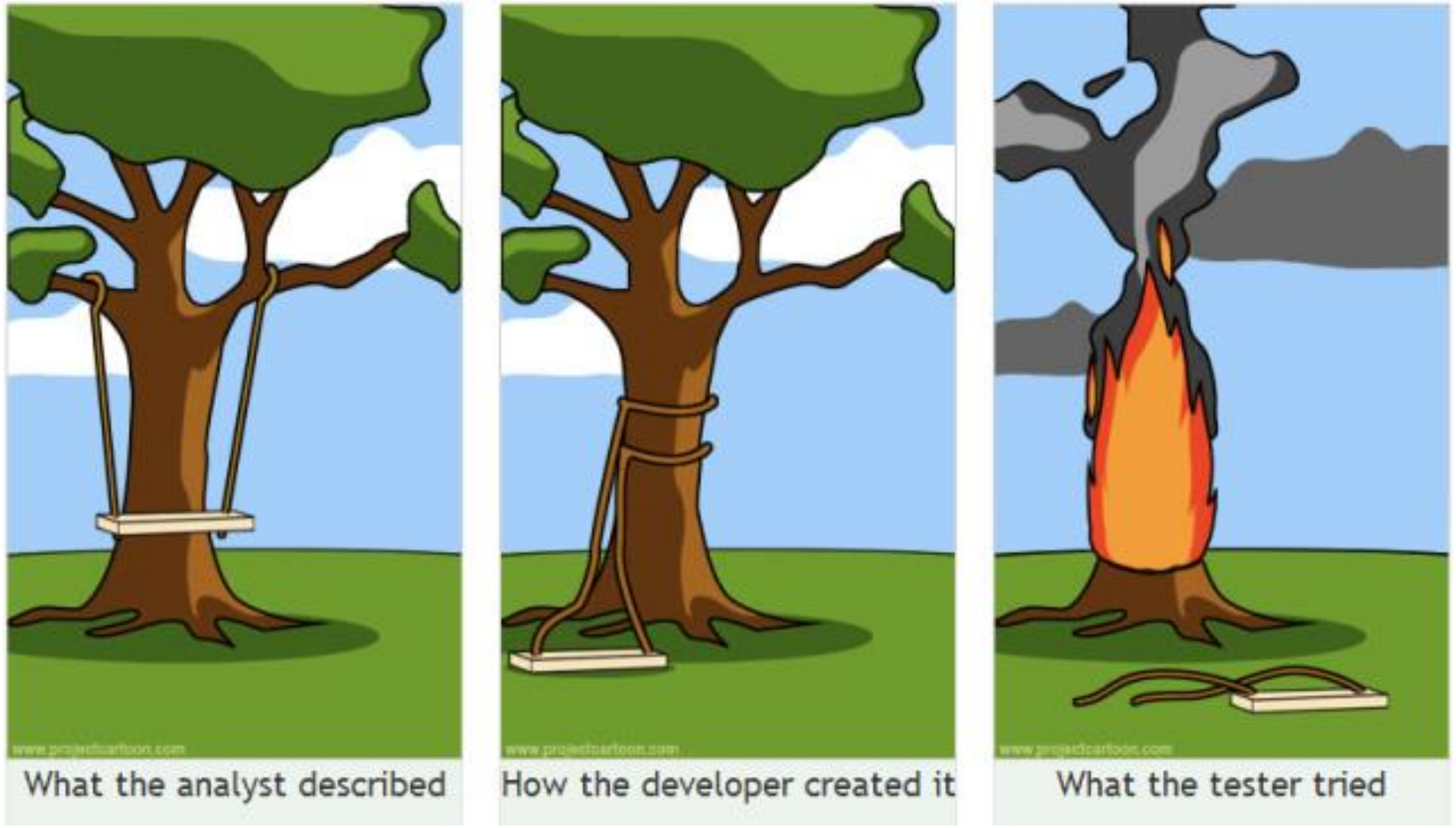

I'LL NEED TO KNOW YOUR REQUIREMENTS BEFORE I START TO DESIGN THE SOFTWARE.
FIRST OF ALL, WHAT ARE YOU TRYING TO ACCOMPLISH?
I'M TRYING TO MAKE YOU DESIGN MY SOFTWARE.
I MEAN WHAT ARE YOU TRYING TO ACCOMPLISH WITH THE SOFTWARE?
I WON'T KNOW WHAT I CAN ACCOMPLISH UNTIL YOU TELL ME WHAT THE SOFTWARE CAN DO.
TRY TO GET THIS CONCEPT THROUGH YOUR THICK SKULL: THE SOFTWARE CAN DO WHATEVER I DESIGN IT TO DO!
CAN YOU DESIGN IT TO TELL YOU MY REQUIREMENTS?
E-mail: SCOTTADAMS@AOL.COM
© 2006 Scott Adams, Inc. /Dist. by UFS, Inc.
www.dilbert.com

Informal Requirements

From requirements to final product


How the customer explained it
How the project leader understood it
How the analyst designed it
How the programmer wrote it
How the business consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

From requirements to final product



From requirements to final product

THE PROBLEM ABOUT BEING A PROGRAMMER

MY MOM SAID:

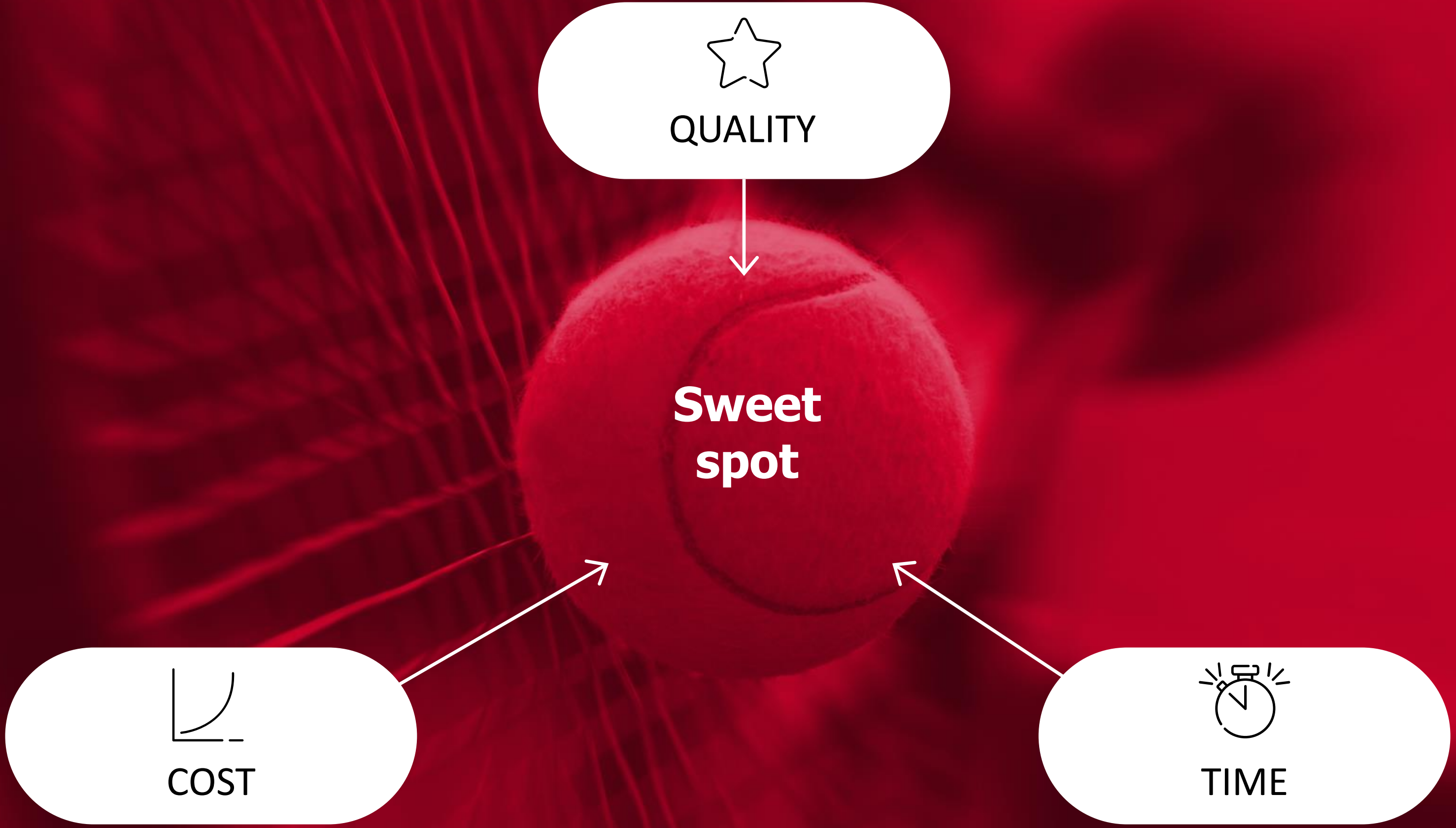
"HONEY, PLEASE GO TO THE MARKET AND BUY 1 BOTTLE OF MILK. IF THEY HAVE EGGS, BRING 6."

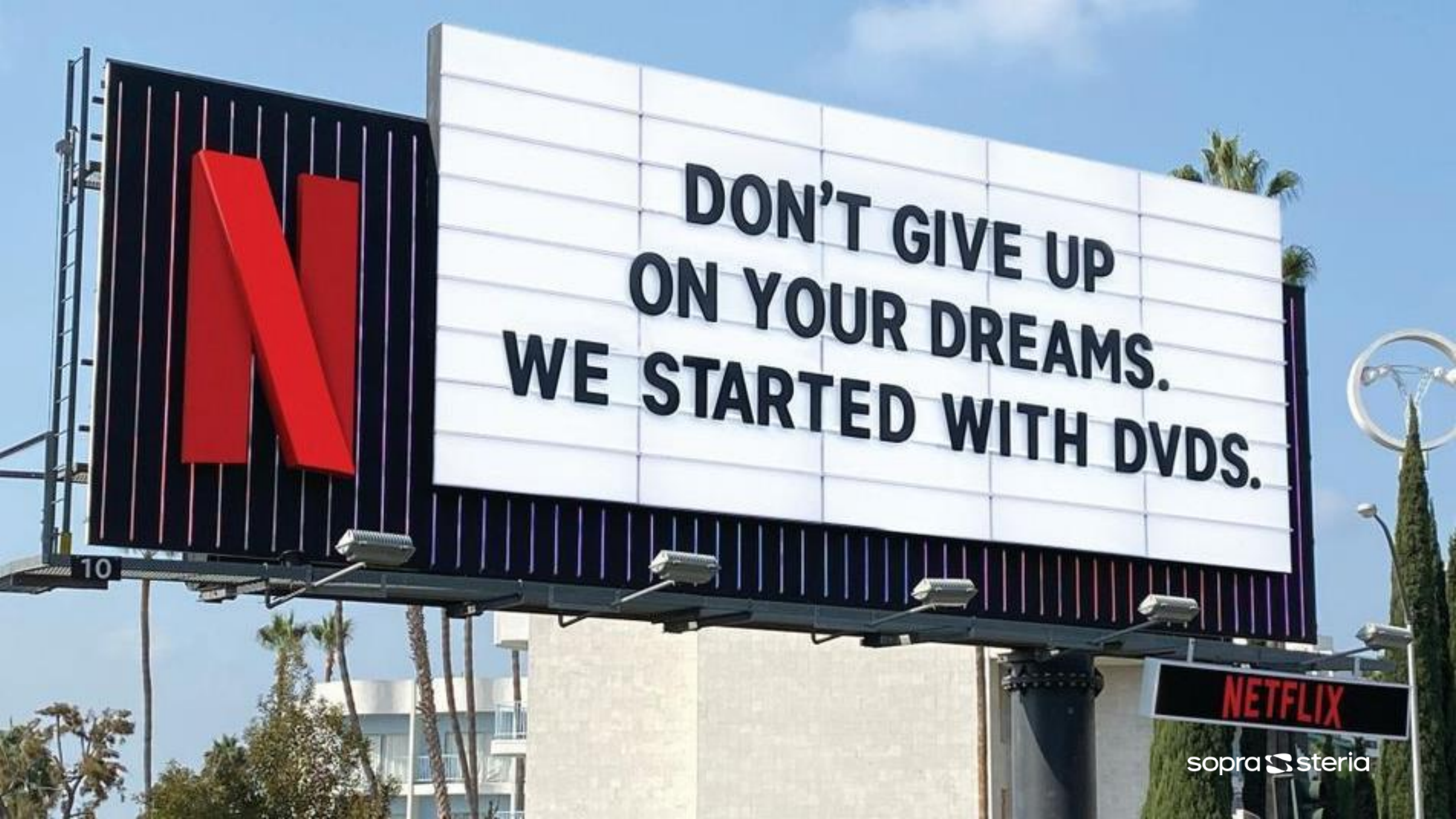
I CAME BACK WITH 6 BOTTLES OF MILK.

SHE SAID: "WHY THE HELL DID YOU BUY 6 BOTTLES OF MILK?"

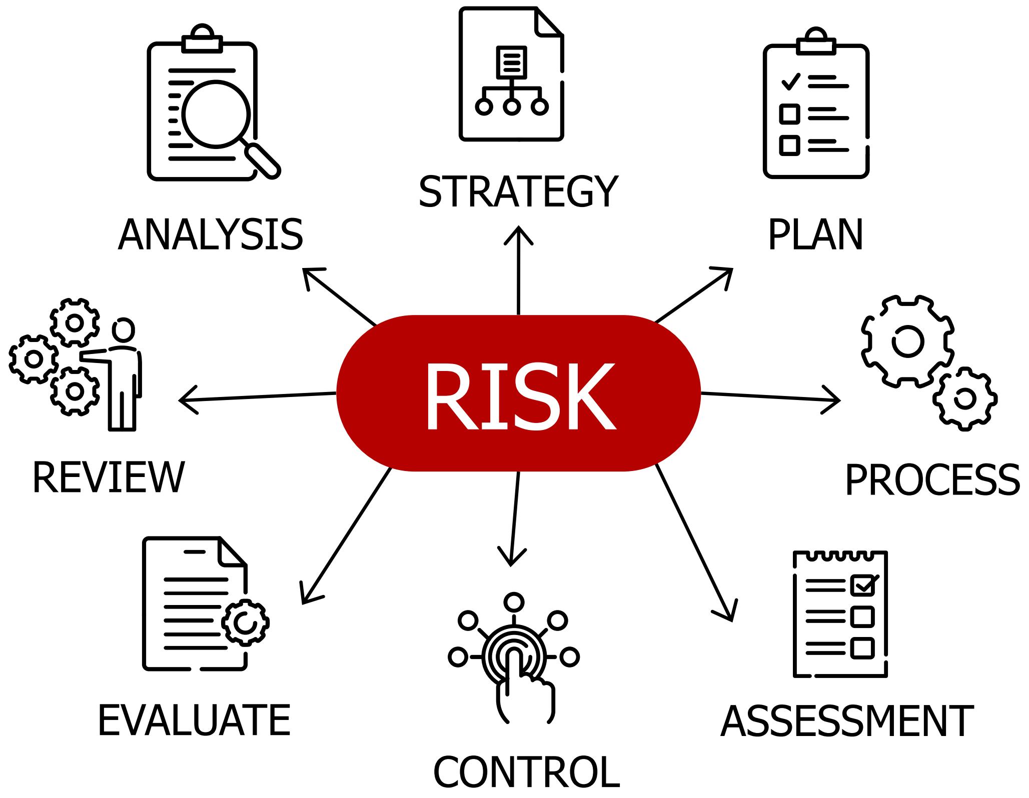
I SAID: "BECAUSE THEY HAD EGGS!"

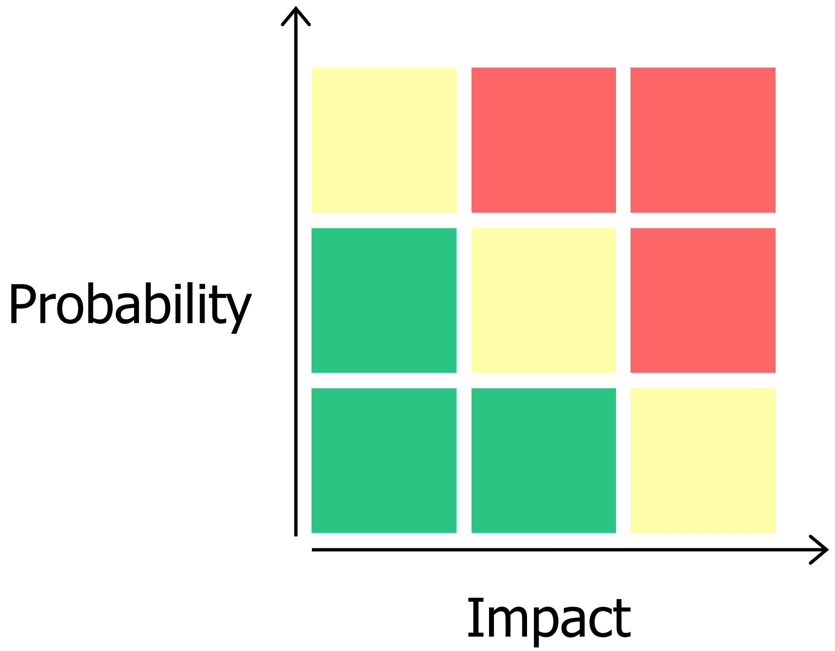
sopra steria


QUALITY
Sweet spot
COST
TIME


DON'T GIVE UP
ON YOUR DREAMS.
WE STARTED WITH DVDS.
10
NETFLIX

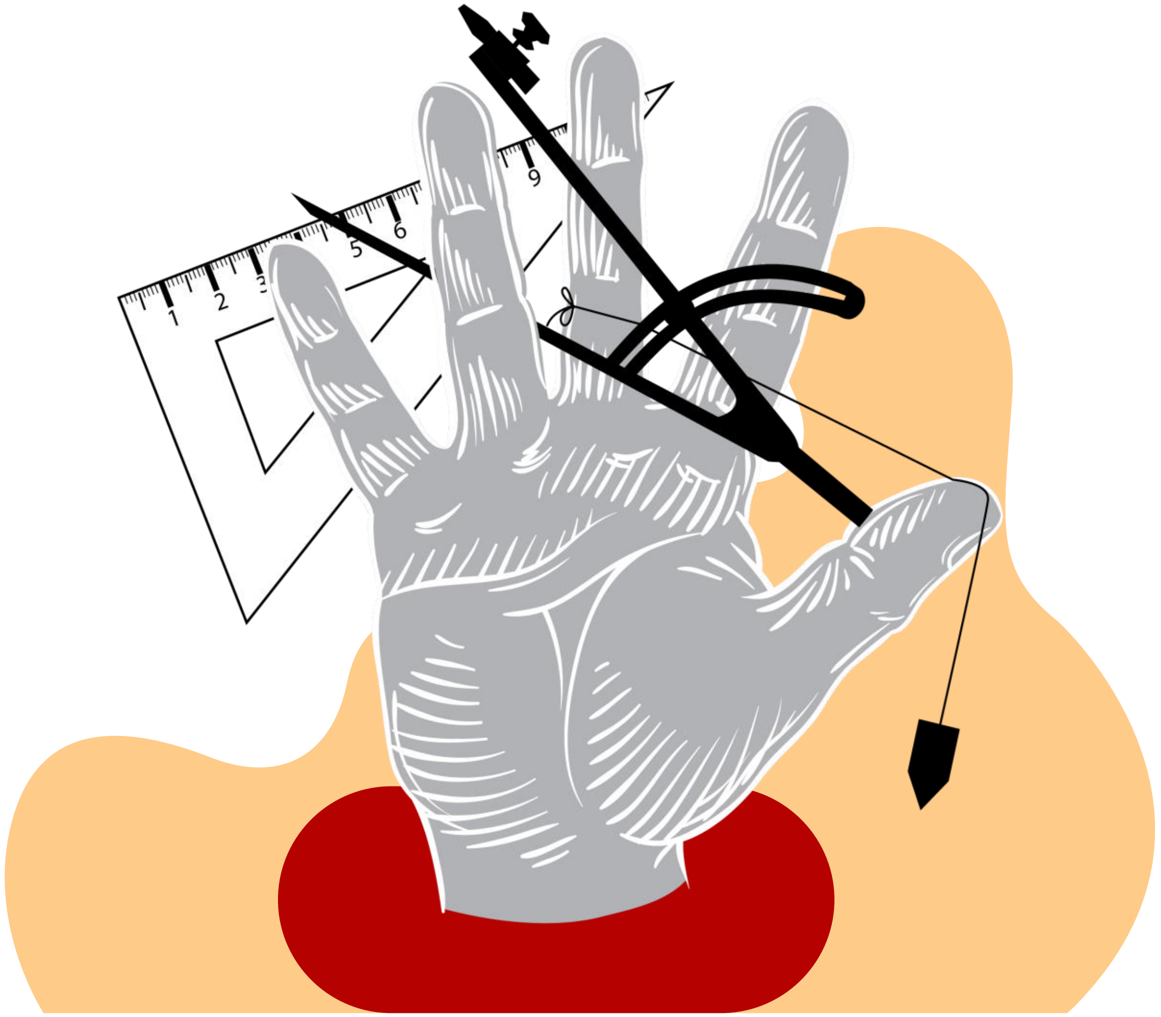
Strategy


RISK
ANALYSIS
STRATEGY
PLAN
REVIEW
PROCESS
EVALUATE
CONTROL
ASSESSMENT


Probability
Impact

What to measure?

- Static testing
- Dynamic testing
- Code Reviews
- Code Coverage
- Deployment times
- Number of tests going green
- Up-time
- Performance
- Number of defects in production



Quality Attributes

Functionality

Reliability

Performance

Compatibility

User experience

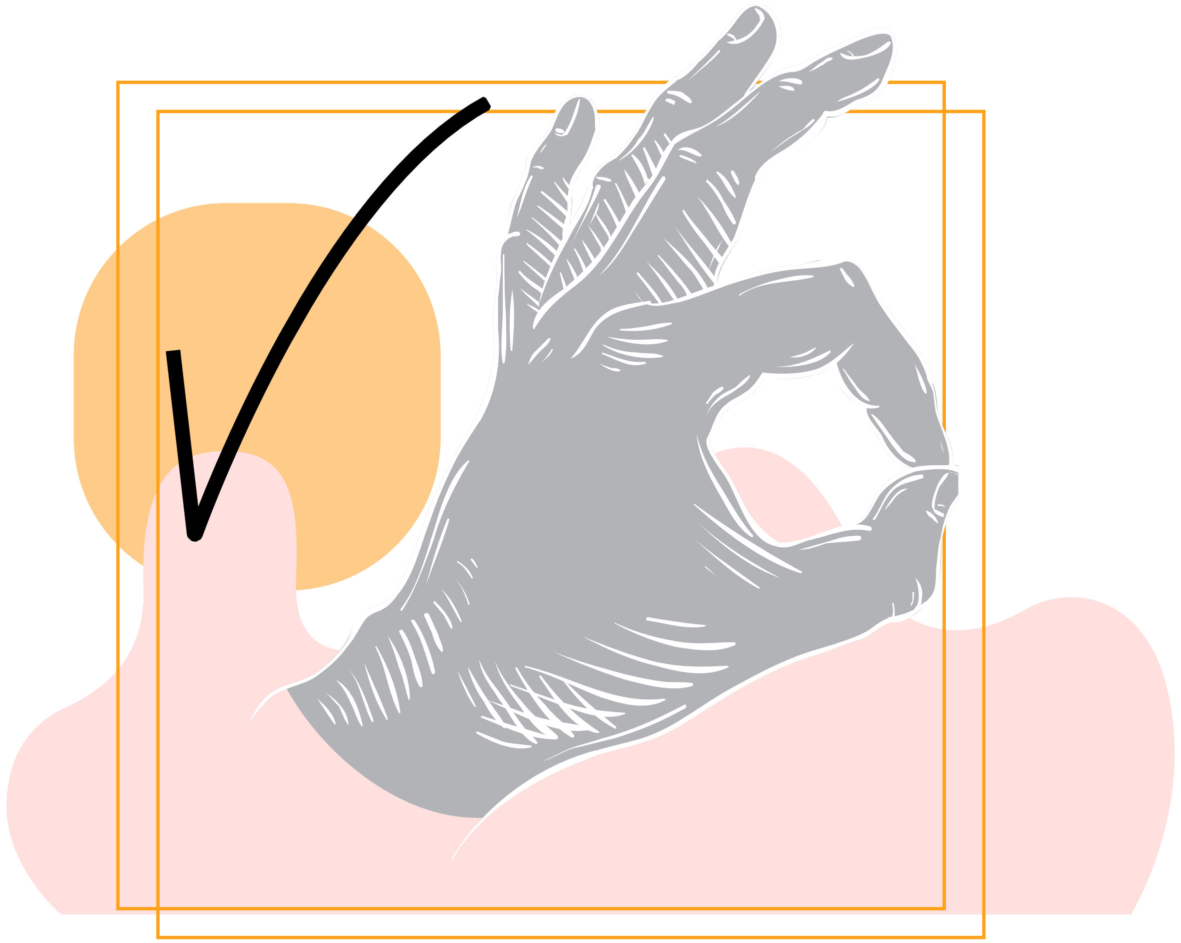
Security

Maintainability

Portability

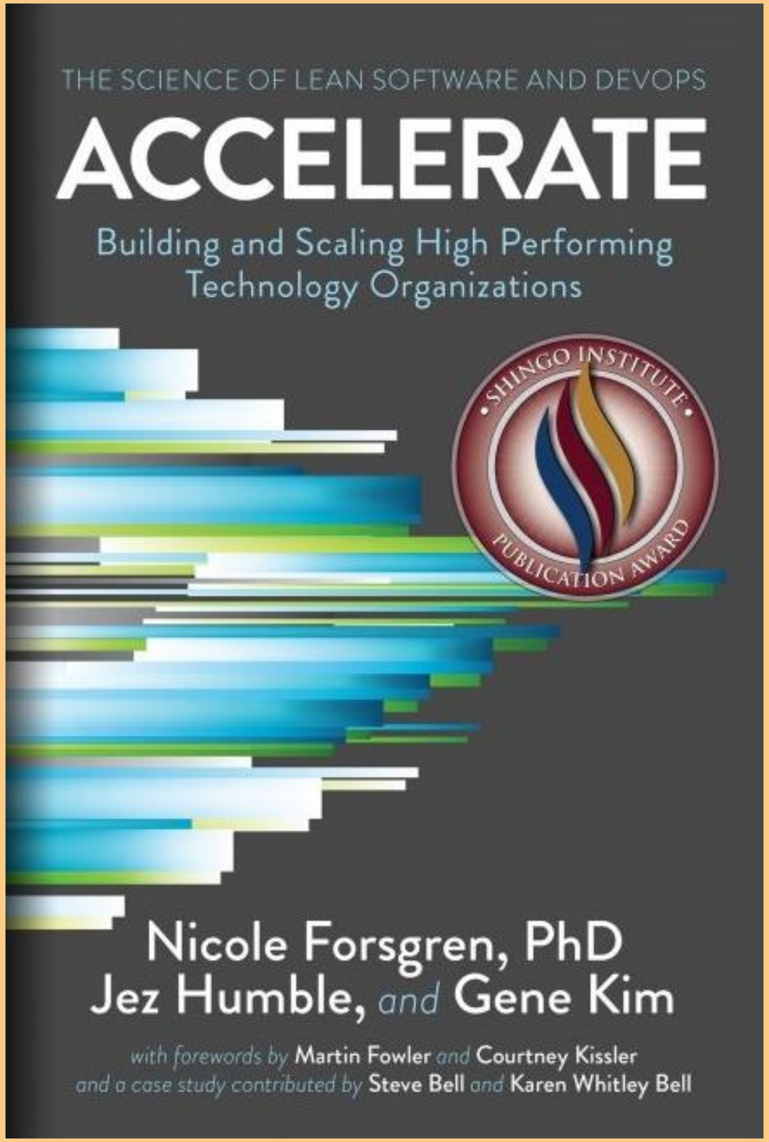
Documentation

Compliance

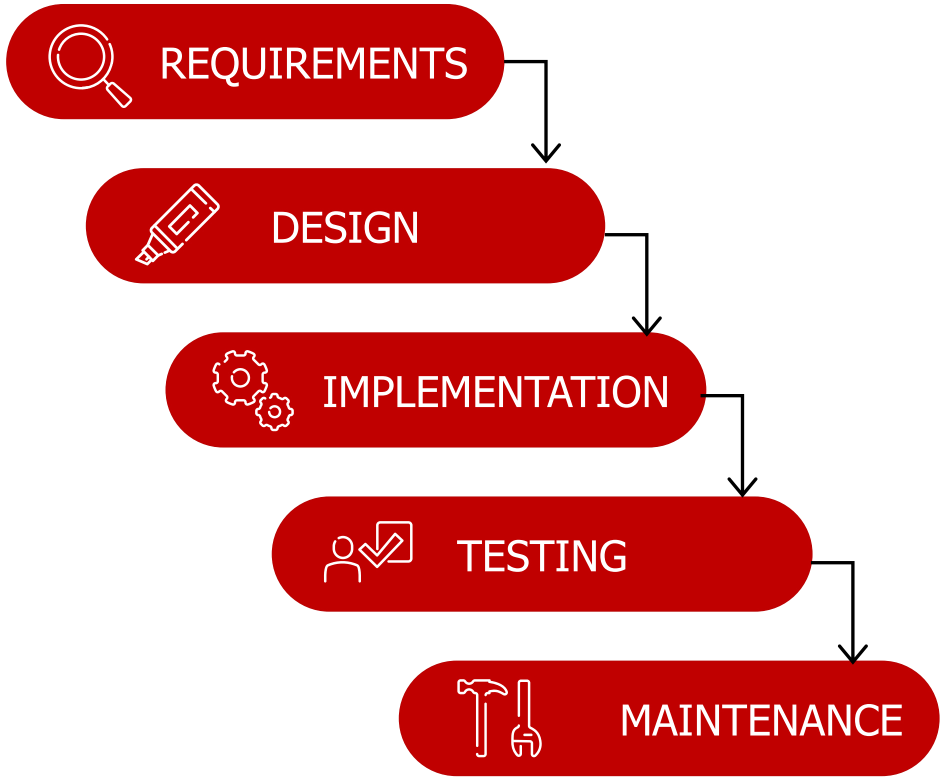


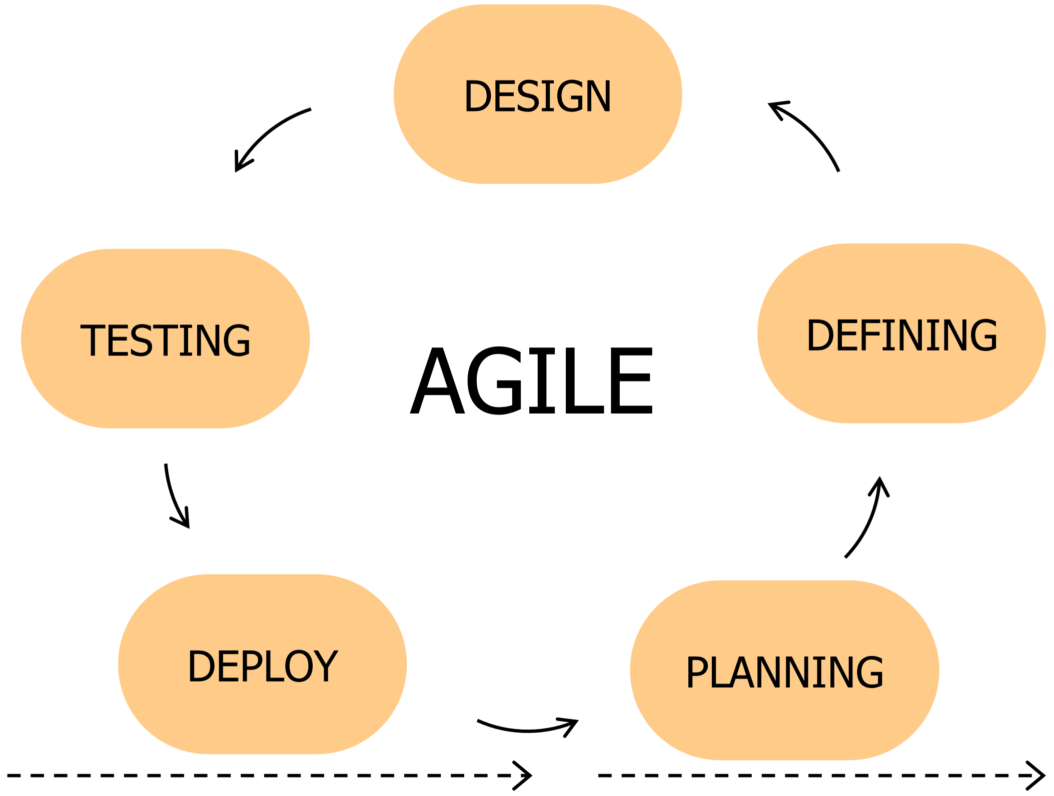
Measuring high-productivity teams

- Deployment Frequency
- Lead Time to Changes
- Mean Time to Recovery
- Change Failure

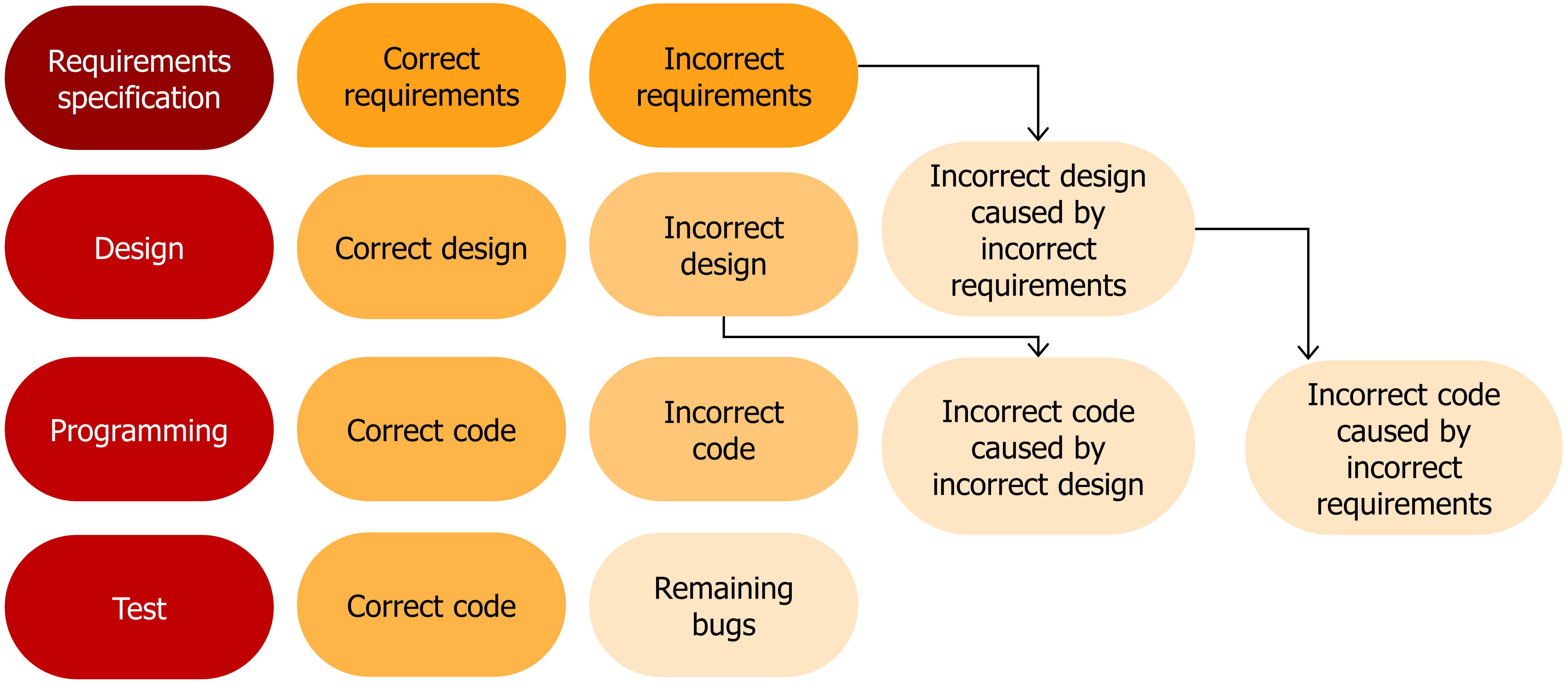

THE SCIENCE OF LEAN SOFTWARE AND DEVOPS
ACCELERATE
Building and Scaling High Performing Technology Organizations
SHINGO INSTITUTE PUBLICATION AWARD
Nicole Forsgren, PhD
Jez Humble, and Gene Kim
with forewords by Martin Fowler and Courtney Kissler
and a case study contributed by Steve Bell and Karen Whitley Bell

Waterfall vs. Agile

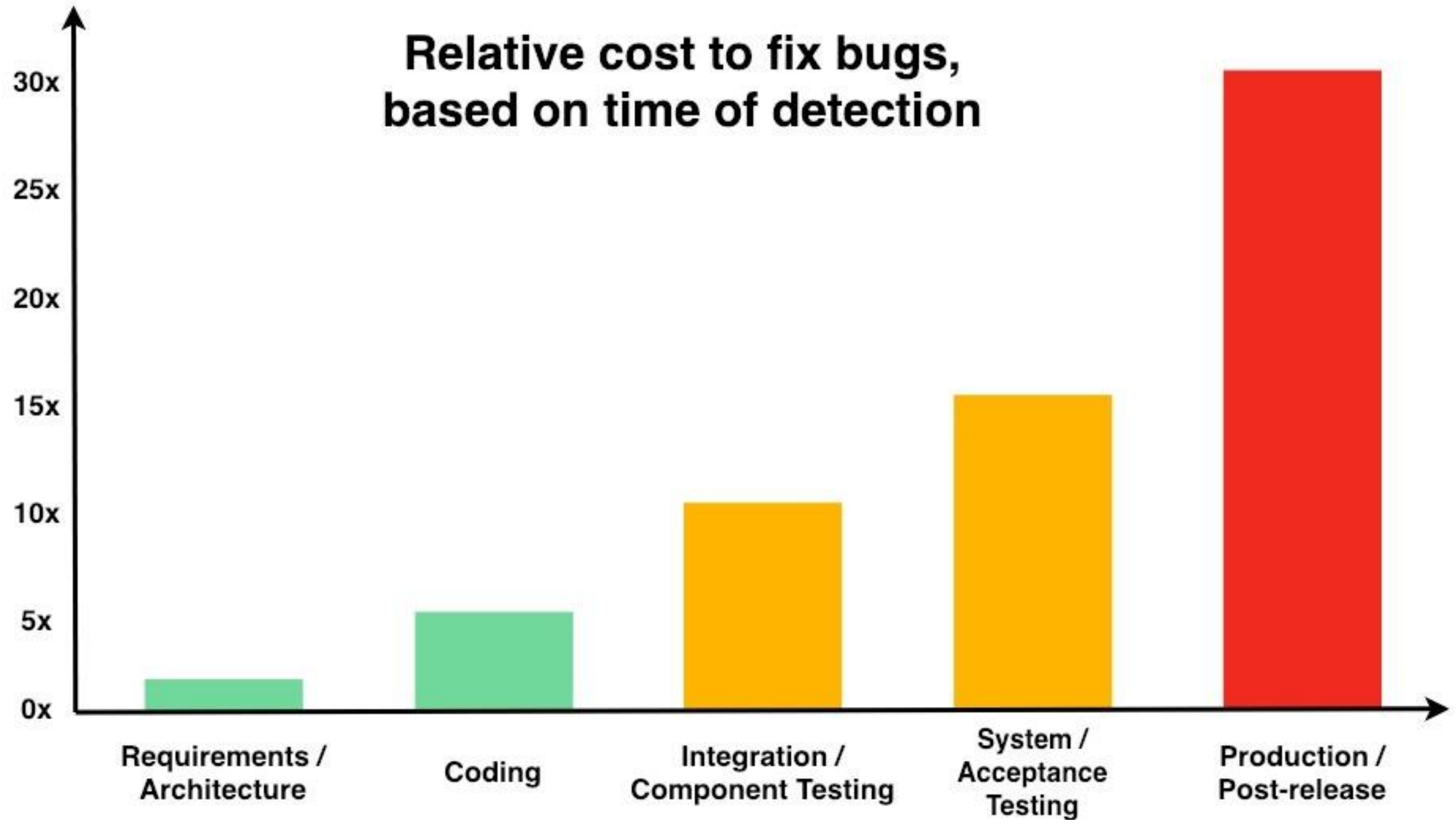




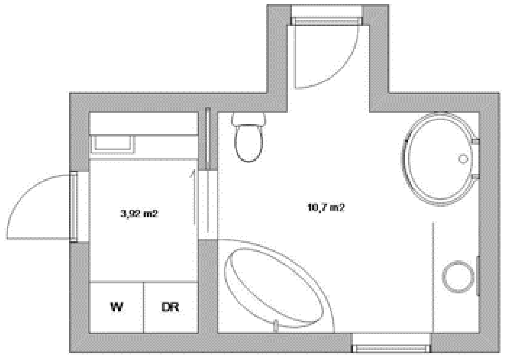
Where do bugs occur?

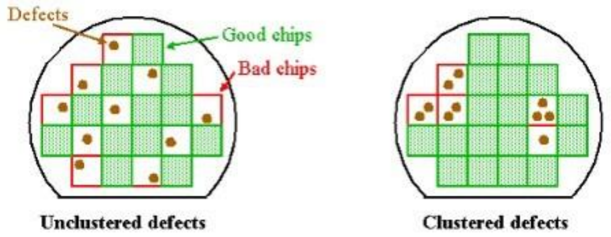

Requirements specification
Correct requirements
Incorrect requirements
Design
Correct design
Incorrect design
Incorrect design caused by incorrect requirements
Programming
Correct code
Incorrect code
Incorrect code caused by incorrect design
Incorrect code caused by incorrect requirements
Test
Correct code
Remaining bugs

Cost of bugs


Relative cost to fix bugs,
based on time of detection
30x
25x
20x
15x
10x
5x
0x
Requirements / Architecture
Coding
Integration / Component Testing
System / Acceptance Testing
Production / Post-release

7 test principles





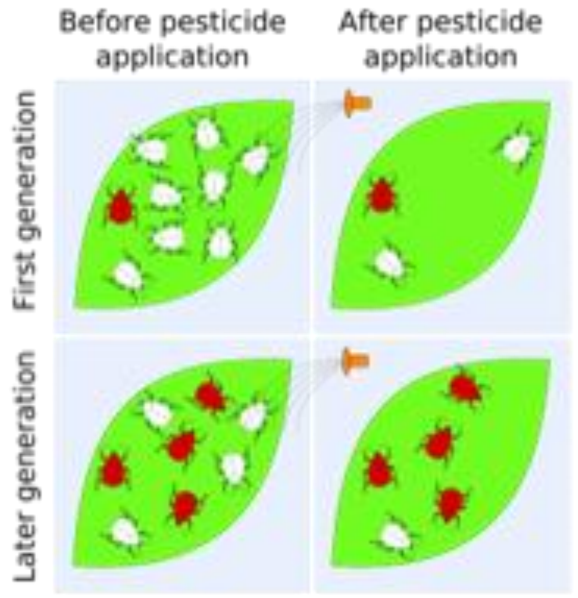
1 Testing shows the presence of defects, not their absence

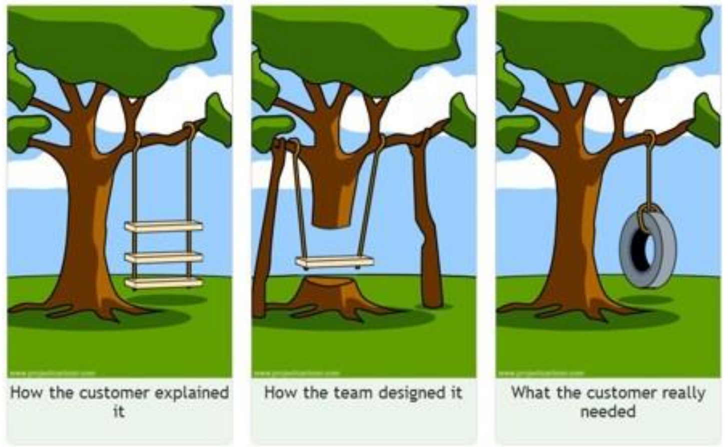
2 Exhaustive testing is impossible

3 Early testing saves time and money

4 Defect cluster together

7 test principles




How the customer explained it
How the team designed it
What the customer really needed

5

«Pesticid-paradox»

Beware of the pesticide paradox

6

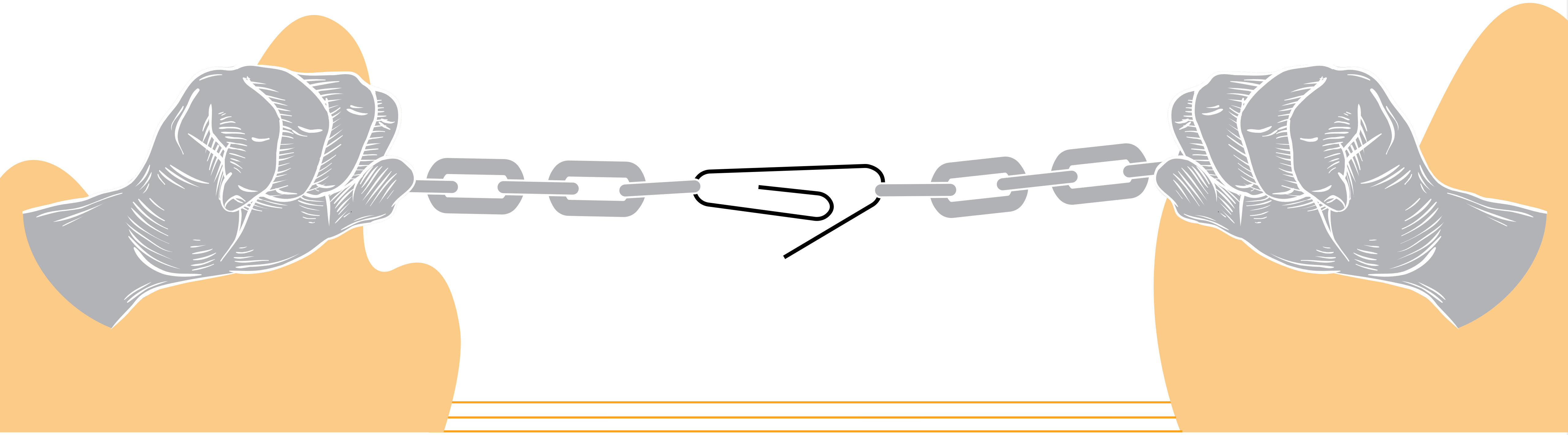
Testing is context dependent

7

Absence of errors is a fallacy

Quality

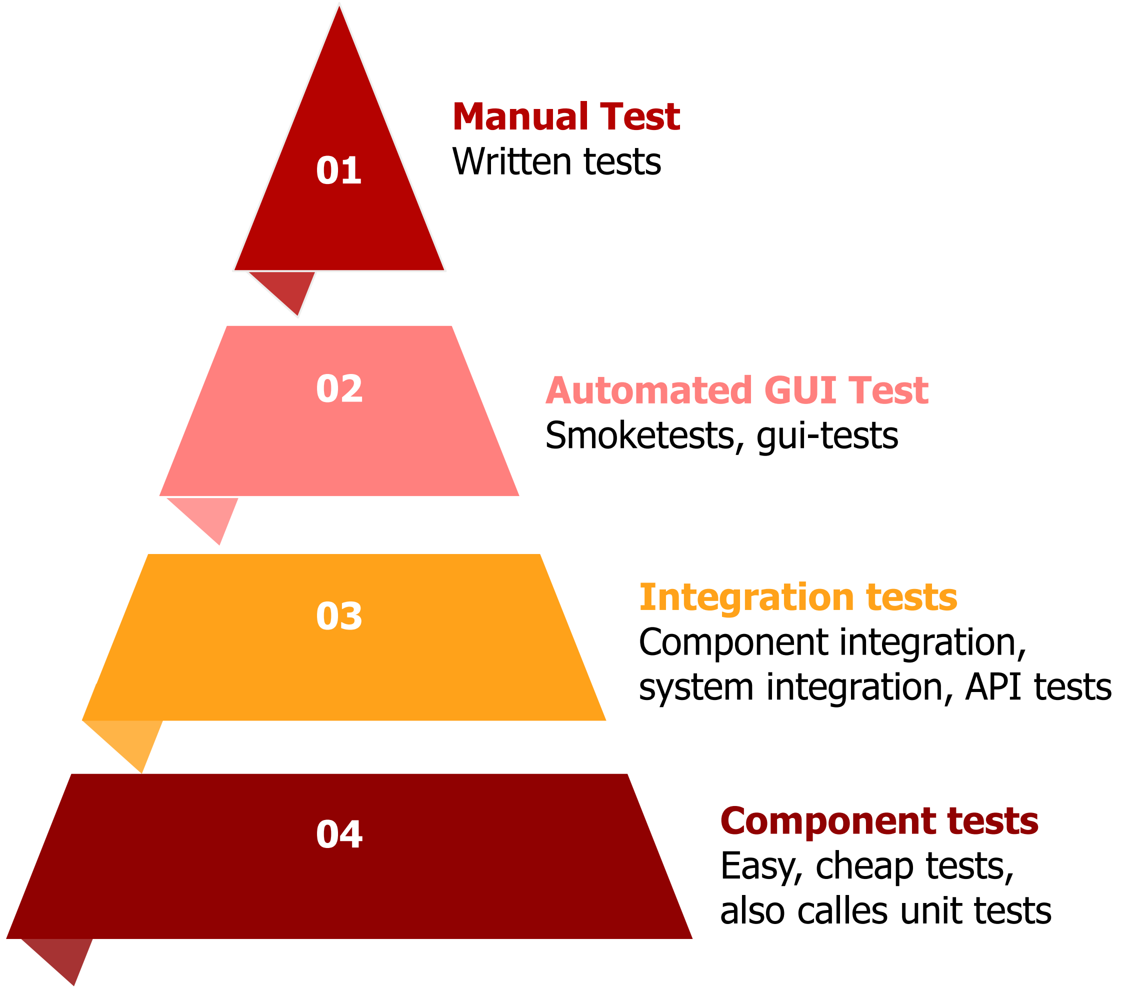
A chain is only as strong as its weakest link



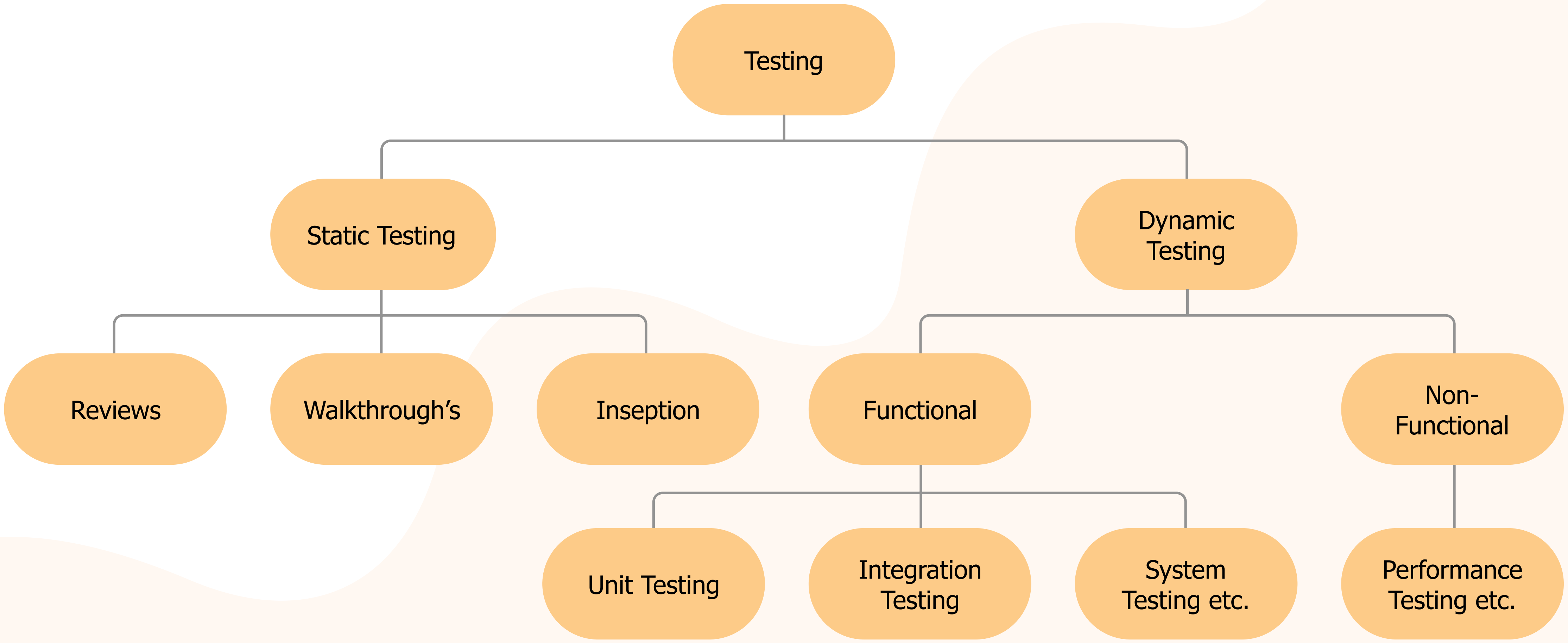
Break – 15 min

Testing pyramid

- Component test tests components, classes etc.
- Integration test tests integrations between components and systems
- Api-test tests that services do what they are supposed to
    - E.g. rest/soap-requests
- GUI-test tests that buttons do what is expected
    - E.g selenium, cypress
- The more the merrier
- All levels can be done by developers



Static and dynamic testing



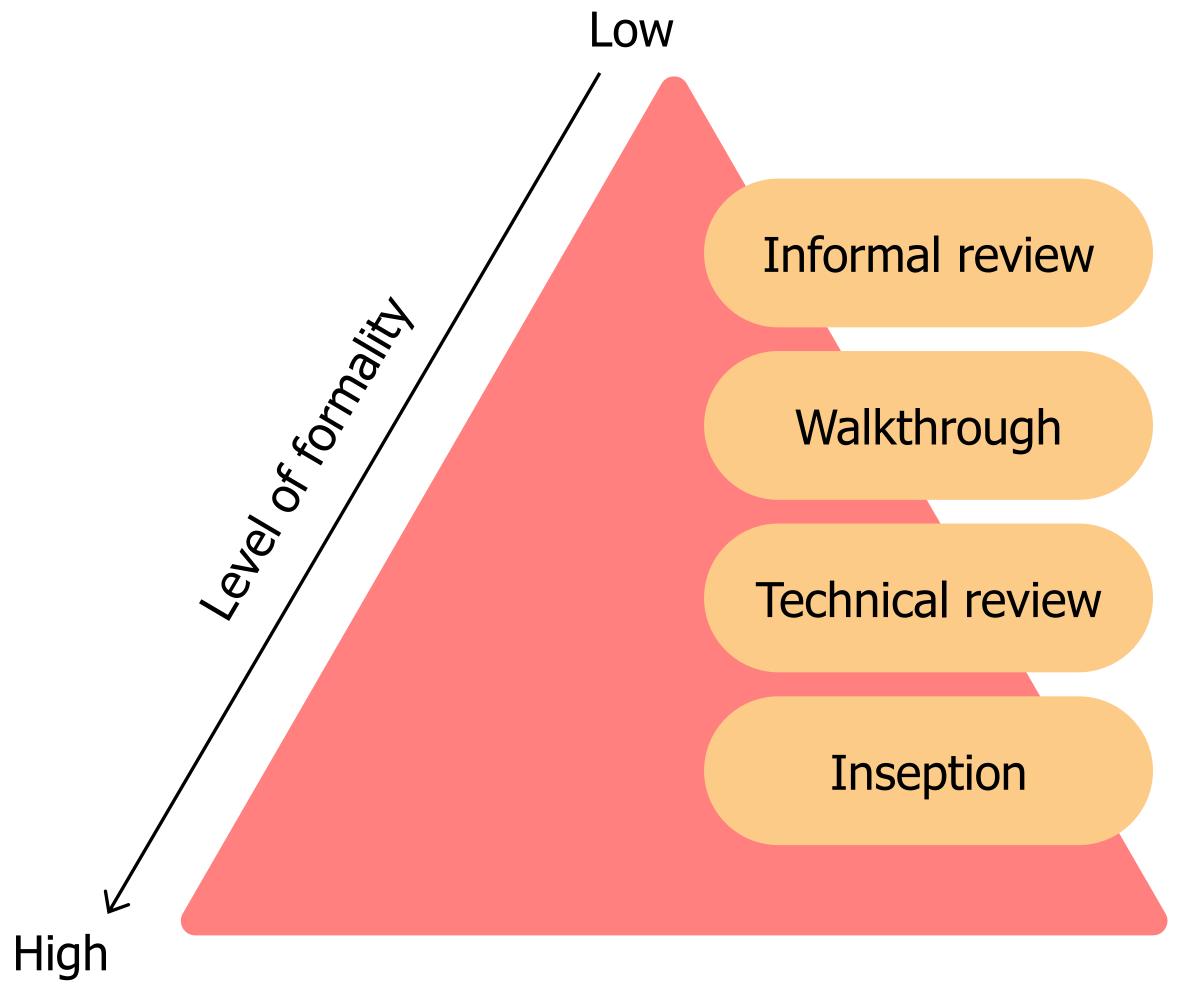
Static testing – requirements specification

1 Informal review
- «Coffee-talks»

2 Walkthrough
- Review of documents

3 Technical review
- Review by specialists

4 Inseption
- Often called formal review
- Mest thorough



Dynamic testing - development

BLACK BOX

Specification-/Experience based

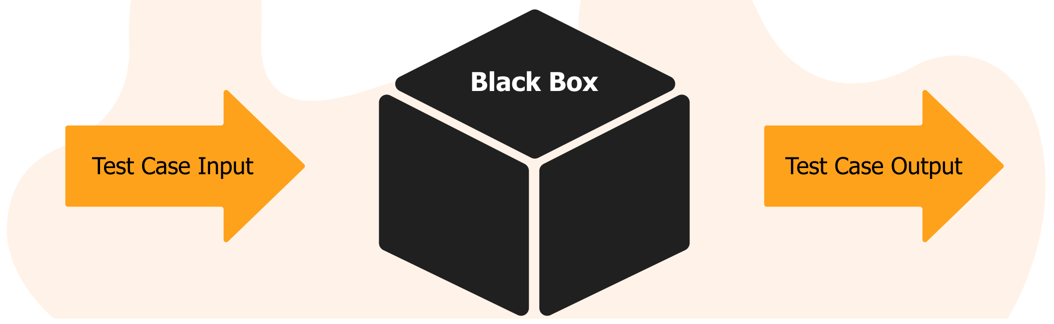
- What does the system do?
- Functional test
- Functionality
- Ease of use

WHITE BOX

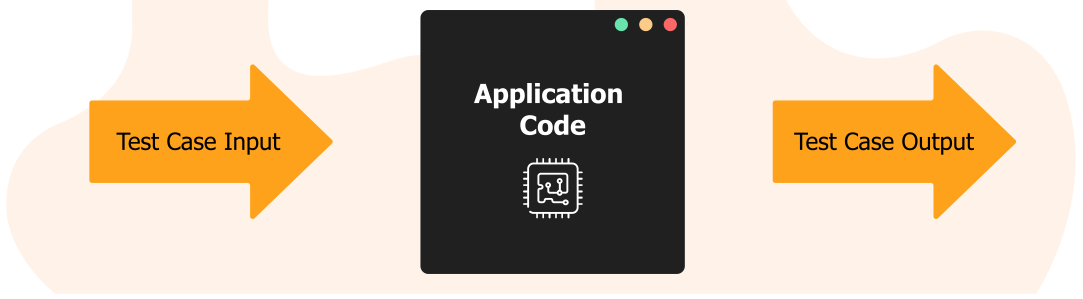
Structure based

- How the system is built
- Technical test
- Development / programming
- Structure
- Components

Black Box Testing



White Box Testing



Equivalence partitioning

Technique to reduce amount of test cases

How will you classify the fruit?

- Apple
- Grape
- Pear
- Strawberry
- Melon
- Oranges
- Blueberry
- Clementines
- Peach
- Pineapple

Boundary value analysis

Designing tests to validate values on the limits

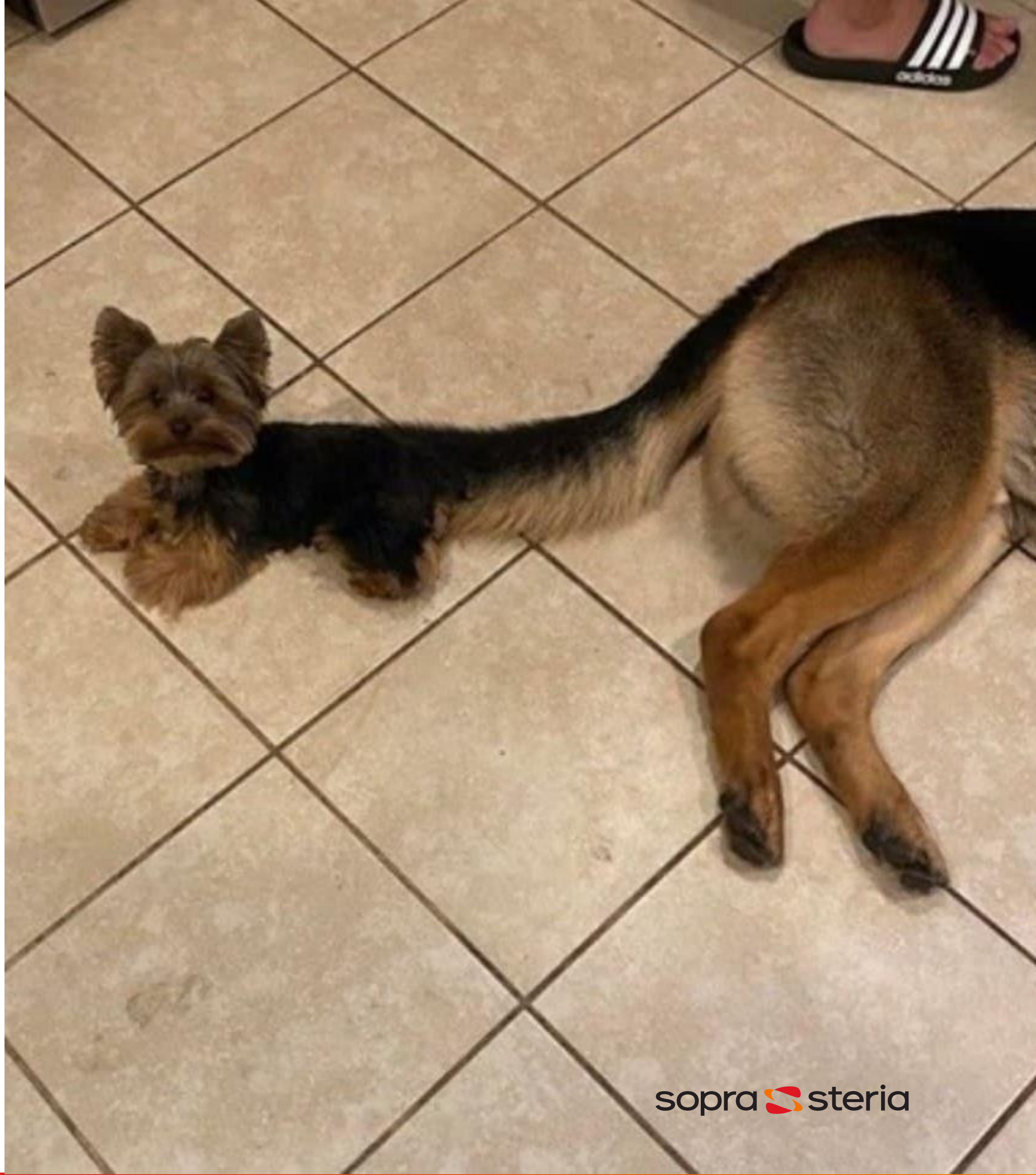
Many bugs can «hide»

Bugs because of misunderstandings

- From or starting from

Valid and invalid boundary values

- Values on the boundary of an equivalence class



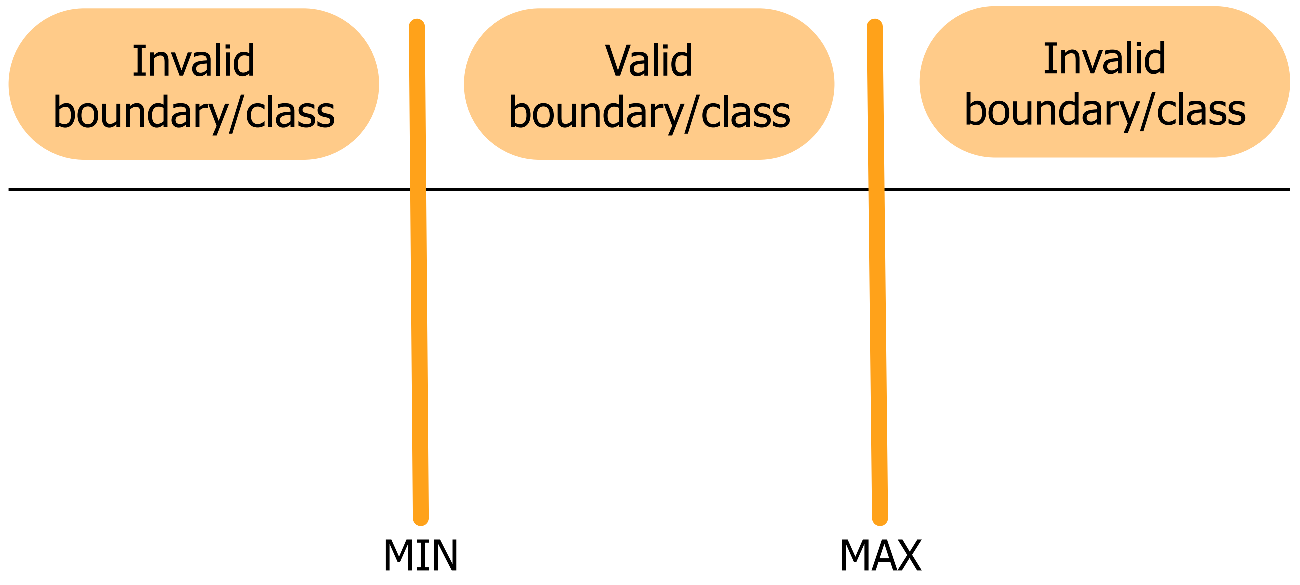
How to define equivalence classes and boundary values?

Systemdata is divided into categories

Equivalence class is an amount of data where all elements is treated equally. It can be identified by specification

Boundary values are values at the edge of equivalence classes

If tests on boundary value fails, we can check whether equivalence class also fails, or if it is only the boundary value – both techniques should be used

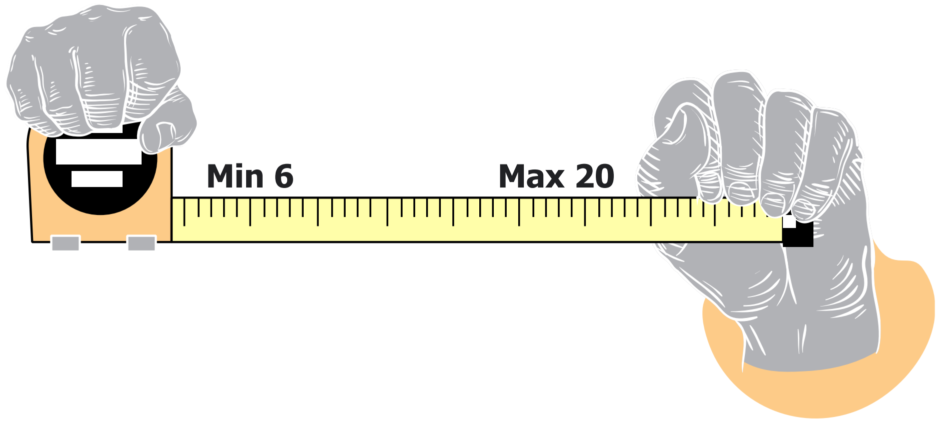


Exercises

Username

The field for usernames has a legal length of 6 characters minimum, and 20 characters maximum.

What are the valid and invalid equivalence classes?
What are the valid and invalid boundary values?



Young talents

Young Talents Community at UIO is for everyone under the age of 30. Alcoholic beverage can only be served to those who are 20 or older.

What are the valid and invalid equivalence classes?
What are the valid and invalid boundary values?

Experience based testing

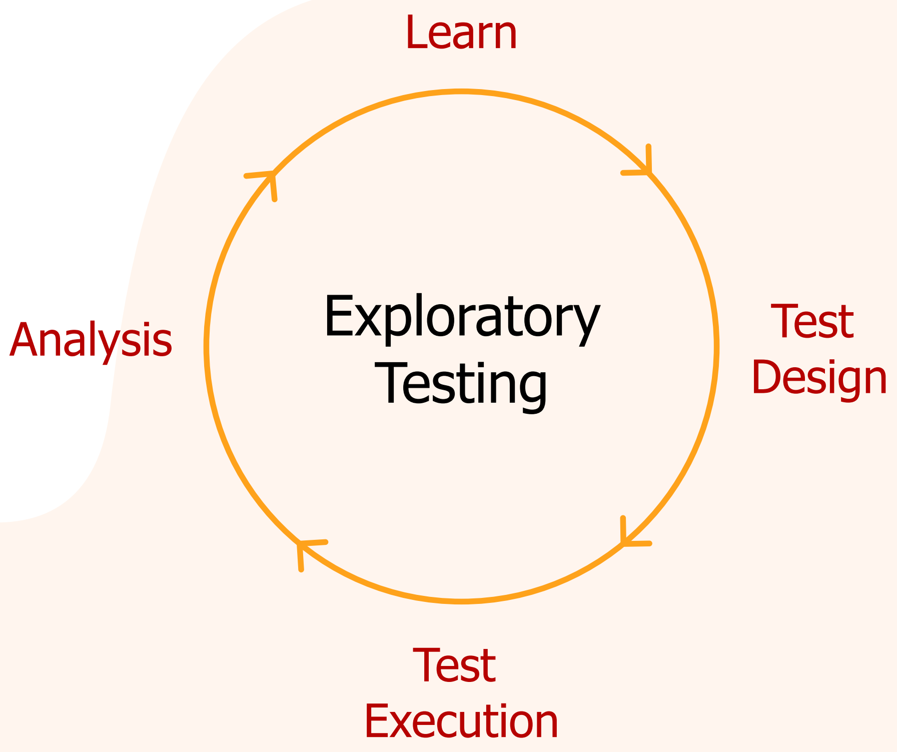
Based on expectation, instinct, and the testers experience.

Informal test technique in combination
with more formal techniques.

Hard to measure test coverage.

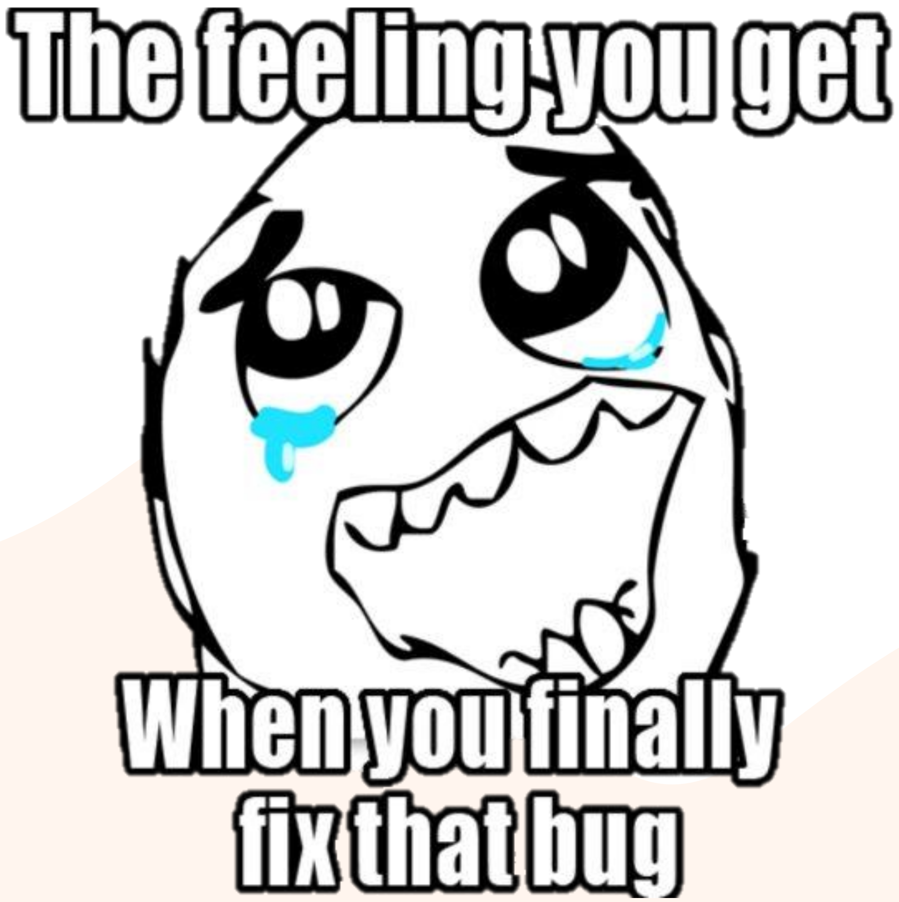
Techniques

- Error guessing
  - List of usual bugs
- Exploratory testing
  - Non-premade tests
  - Can be tested session-based
- Checklist-based
  - Based om domain knowledge and experience
  - Can be used as a guideline



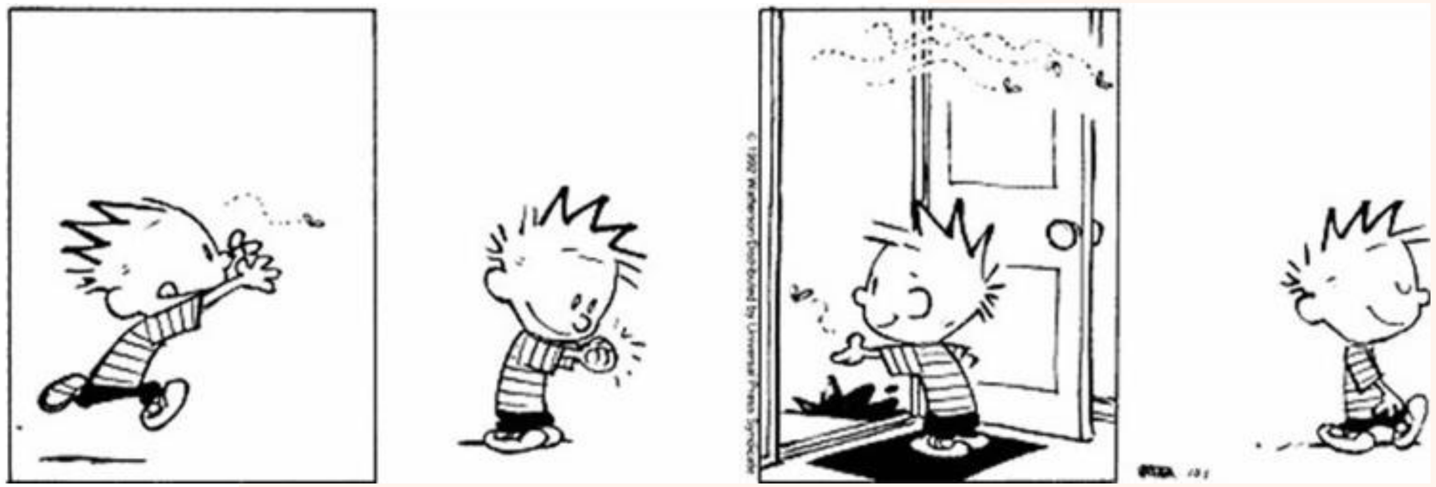
Retest/ confirmation test

Testing a bug that you found in the first test of the issue



Regression:

”when you fix one bug, you introduce several newer bugs.”



Security testing

Often done by specialized testers

Difference between:

- Authentication – checks that the user has access to the system – correct username and password.
- Authorization - is the user supposed to have acces to this part of the system?

Encryption

- Everything from password etc should be encrypted

Firewall

- Blocks unauthorized traffic

IDS (Intrusion Detection System)

- discovers security violations

XSS - Cross site scripting

- Can you insert scripts in the site and/or fields?

Results from security testing should be given on need-to-know-basis

Validation vs verification

Verification and validation is the process of checking that the software meets specification and requirements that fulfill the purpose.

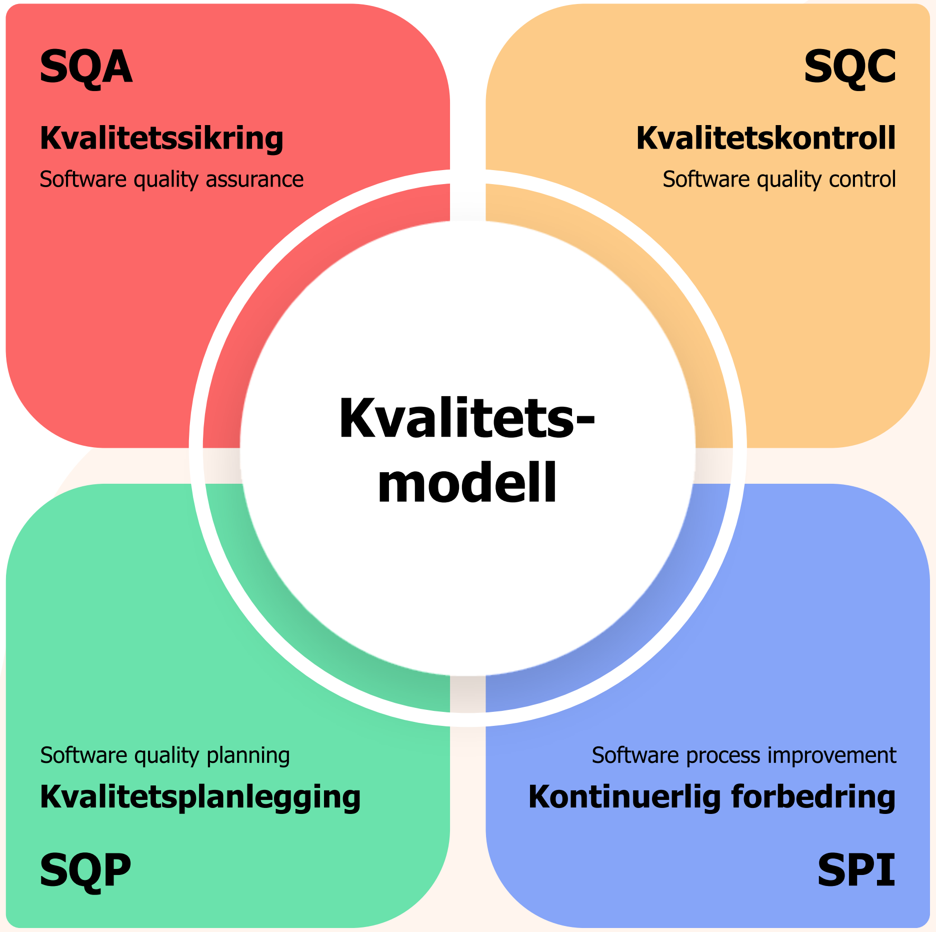
Verification

- Process for checking that documents, design, code and software is build according to requirements
- Static testing

Validation

- Mechanism for testing and validate that the software actually fullfill user needs
- Dynamic testing

Building Quality



sopra steria

DevOps

A way to close the gap between development and operations

Tby utilizing tooling, Continous Integration and delivery the responsibility to maintain a system is shared with the development team.

«You build it, you run it!»

It’s been a thing for over a decade

The Phoenix Project
A Novel About IT, DevOps, and Helping Your Business Win
REVISED WITH NEW RESOURCE GUIDE
Gene Kim, Kevin Behr, and George Spafford

Knowledge – about what?

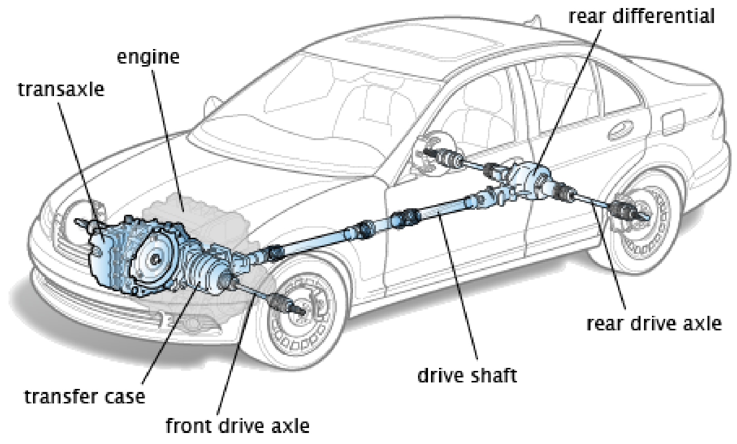
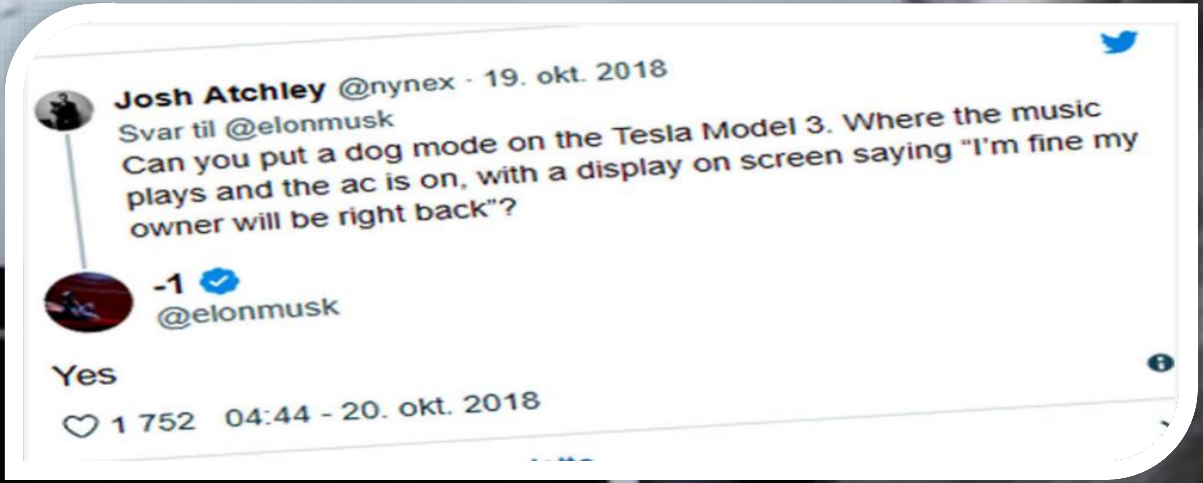


Image courtesy of ClearMechanic.com

Docker
Kubernetes
Kafka
Java
Maven
Azure
React.js
.NET Core
Nuget
Webpack
Vue.js
DevOps
AWS
SSL
Cert
MQ
Rider
Intellij
JUnit
Selenium
Cypress
Linux
OpenShift
HTTPS
XUnit
AI
IOT
CasperJS
Visual Studio
Jenkins
Arkitektur
Testautomatisering
Artefakter
NUnit
Ytelse
Testdata
Tjenestevirtualisering
Testplanlegging
Versjonering
Monitorering
Ikke-funksjonelle tester


Josh Atchley @nynex · 19. okt. 2018
Svar til @elonmusk
Can you put a dog mode on the Tesla Model 3. Where the music plays and the ac is on, with a display on screen saying "I'm fine my owner will be right back"?

-1
@elonmusk

Yes

1 752 04:44 - 20. okt. 2018

sopra steria

DevOps – how to test?

Goal:

Minimize functional testing and make development process as agile as possible.

Testers:

Advisors and test developers.

Focus:

Technical tests and automated tests.
Roll-forward in stead of roll-back.
Domain knowledge.
High level of code- and test-coverage.
Monitoring processes in mandatory.
Requires continous dialogue with operations and users.

DevOps – how to test?

Different terms

Continous testing:

- Dependent on tools, team, individuals and services
- Runs automated tests as part of the pipeline for quick feedback on release-candidate

Continous deployment:

- Dependent on product owners and developers
- Makes the code always deployable

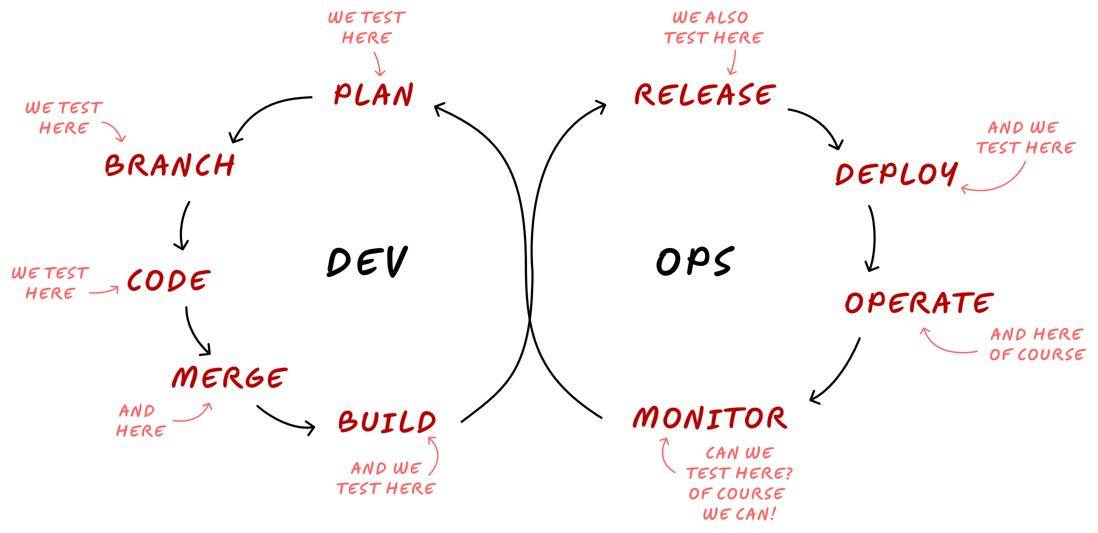
Continous integration:

- Tool-driven
- Code from different developers are often integrated – several times a day

Continous delivery:

- Tools and team-driven
- Release of any approved build to production

Devops and testing



AGILE VS. DEVOPS

AGILE	DEVOPS
Test as early and as often as possible	Test continuously
Automate testing as much as possible	Automate almost everything
Continuous integration and testing is a step forward	Continuous integration and testing is mandatory
Potentially shippable code at the end of a sprint	Potentially shippable code following every integration

sopra steria

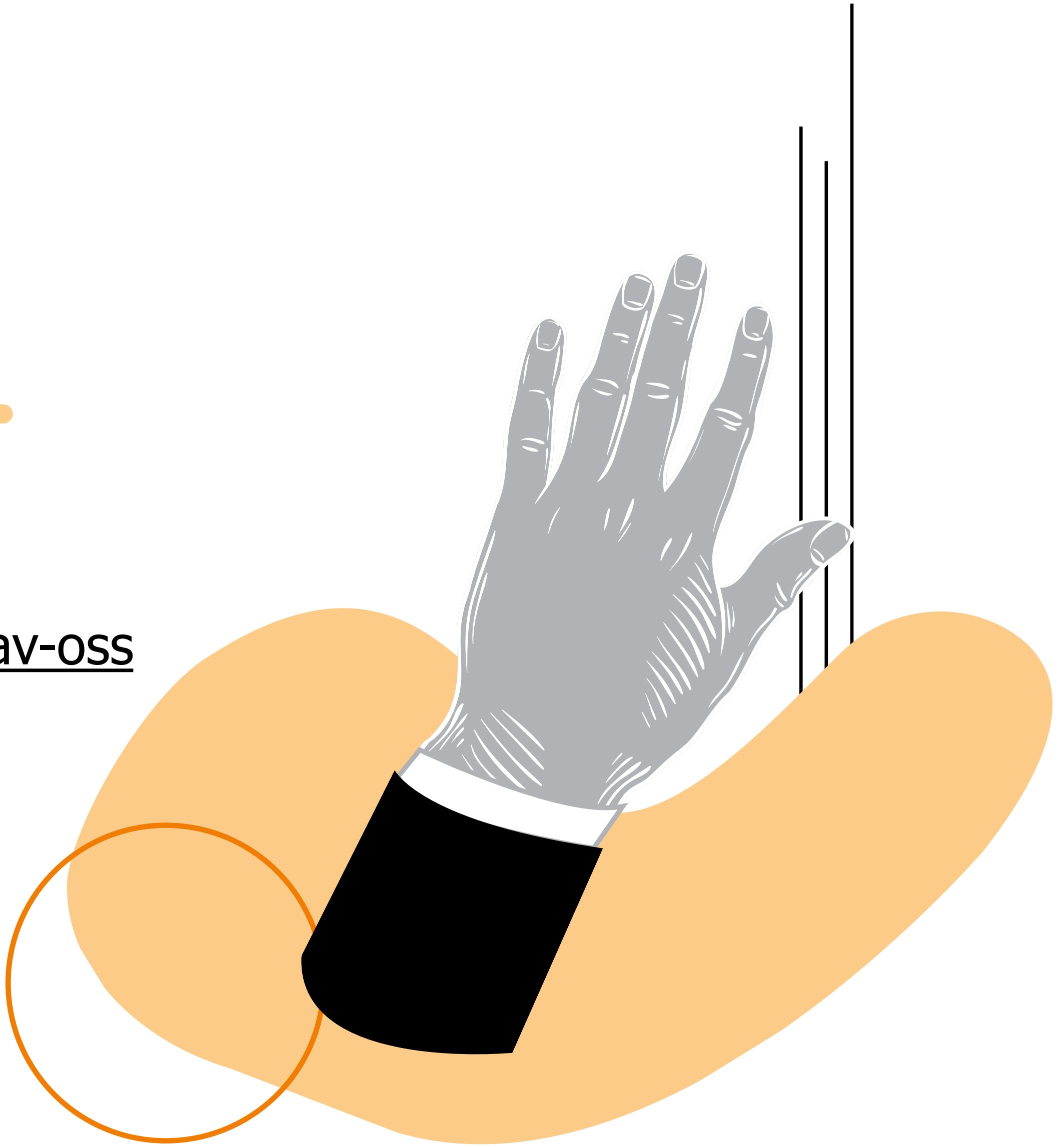
Remember!

- Communication is key
- Specifications are challenging
- Insight is crucial
- There won’t be enough time – get your priorities straight is key
- There will be uncertainties, talk about them loudly
- Stuff will fail – learn from it!

Questions?

Dagen@ifi September 15th

https://www.soprasteria.no/bli-en-av-oss



Thank you!