Why Testing Is Fundamentally Different In DevOps Than Classic Projects

Peter Gfader

beyond-agility.com

Why Testing Is Fundamentally Different In DevOps Than Classic Projects

Meta Workshop

Peter Gfader

beyond-agility.com

Workshop based on

Growing Agile: A Coach's Guide to Agile Testing
https://leanpub.com/AgileTesting/
Samantha Laing and Karen Greaves

AGENDA

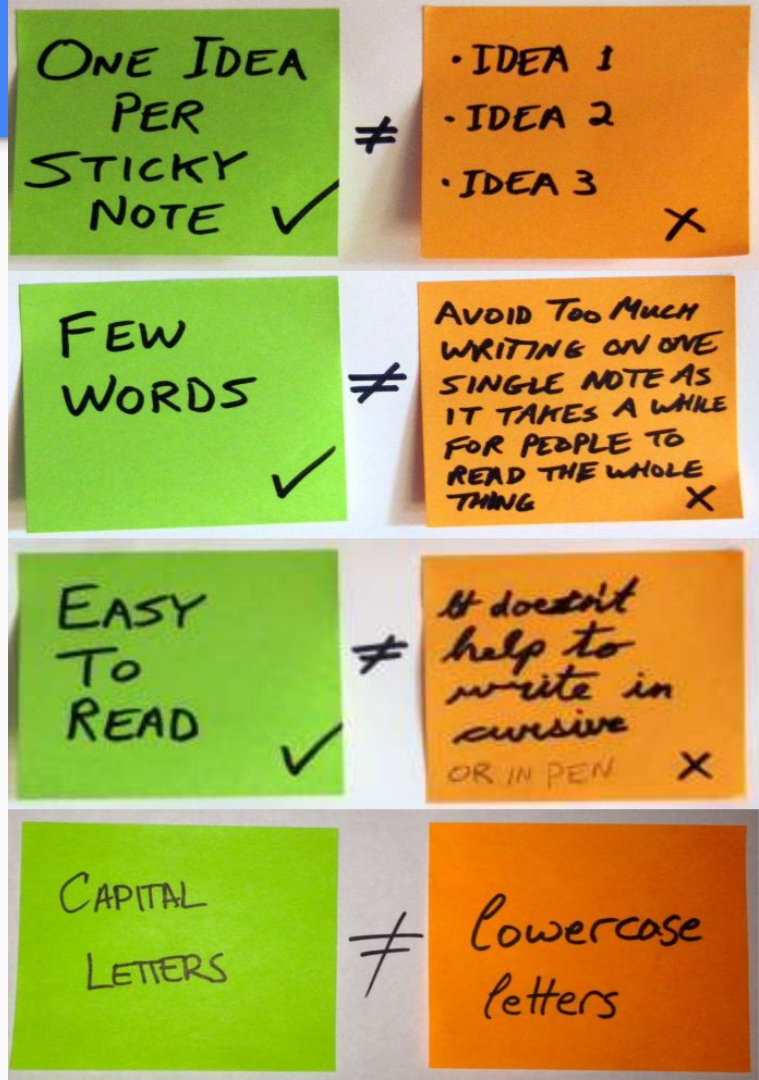
1. WARM UP
2. EXCHANGE PRACTICES
3. TRADITIONAL VS AGILE
4. COLLECT PRACTICES
5. TAKE AWAYS

TRIZ

Warm Up - Hand Up **If True**

1. Testing is always behind
2. Automation is even further behind that
3. Testers can’t work until development is done
4. There is pressure at the end of a sprint
5. There is blame around bugs “It’s his/her fault” etc.
6. The DevOps Team is unhappy

Optimize for Reading



http://agilitrix.com/2013/10/how-to-go-fast-with-sticky-notes/

1 Post It - for yourself

What is your pain with testing?

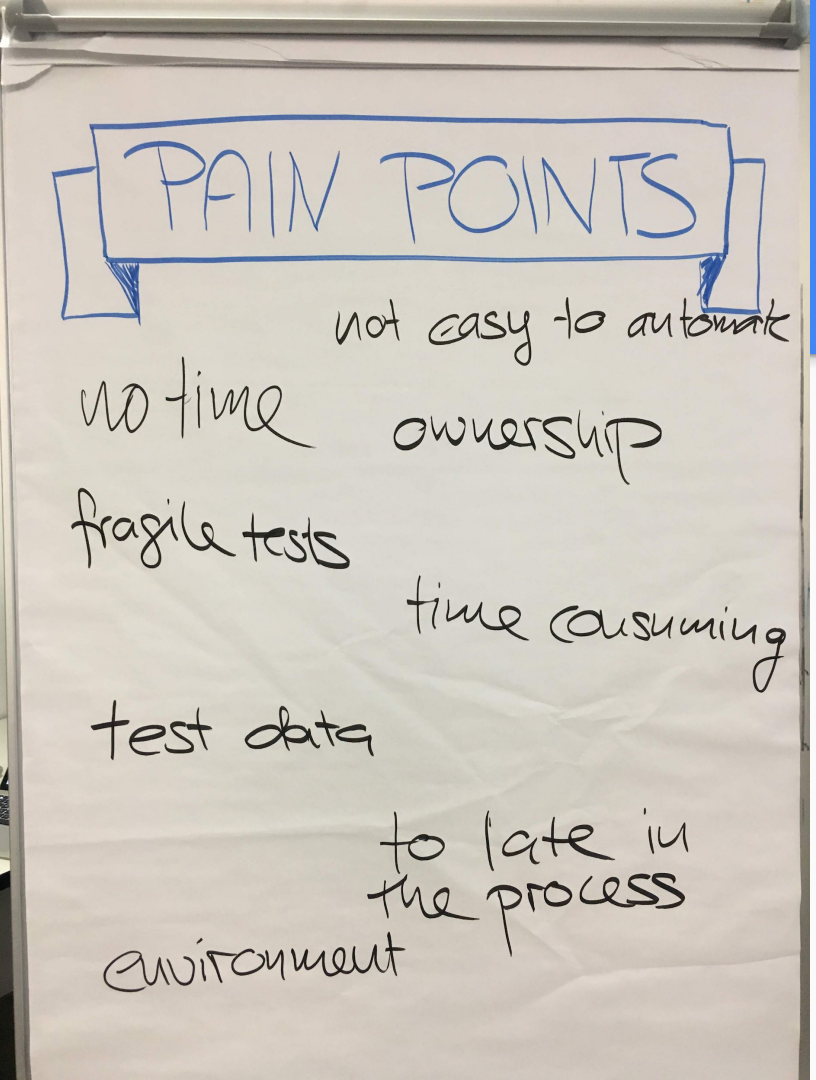
In Pairs… in 4 (Notice similarities, differences, develop further)

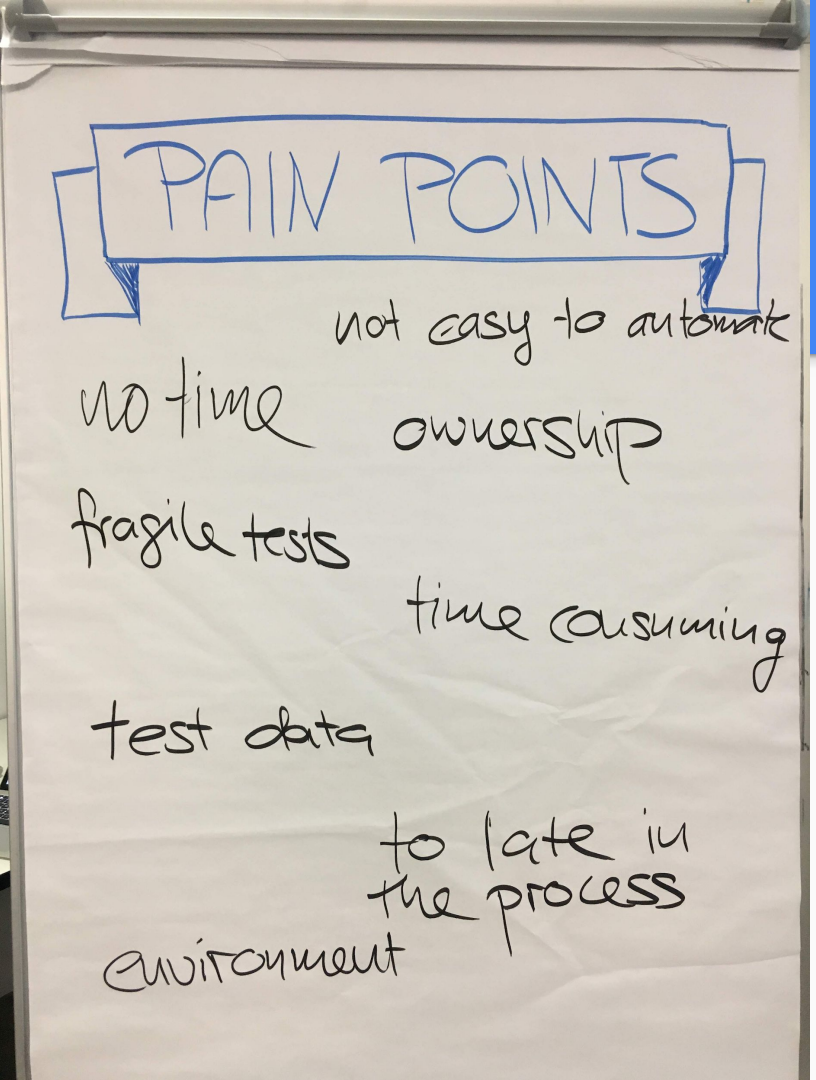
What is your pain with testing?

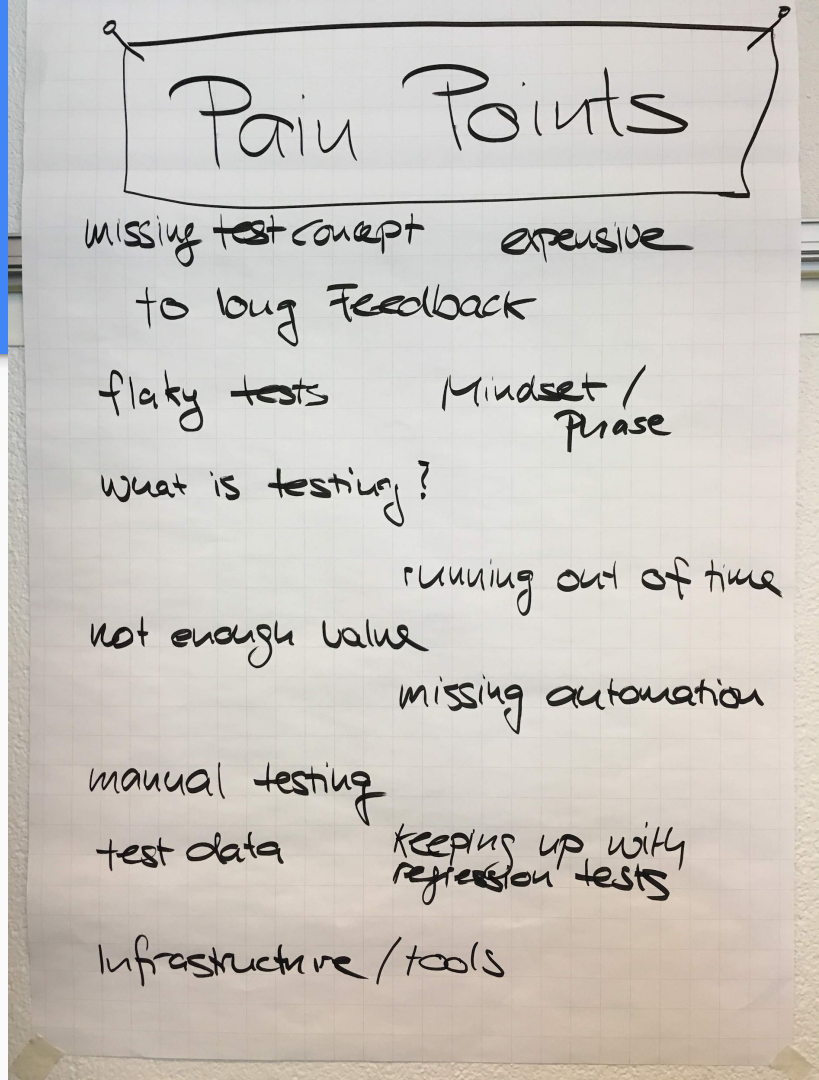
Build up on your input

What is your pain with testing?

What is 1 thing that stood out in your conversation?


PAIN POINTS
not easy to automate
no time
ownership
fragile tests
time consuming
test data
to late in the process
environment


PAIN POINTS
not easy to automate
no time
ownership
fragile tests
time consuming
test data
to late in the process
environment


Pain Points
missing test concept
expensive
to long Feedback
flaky tests
Mindset / Phase
what is testing?
running out of time
not enough value
missing automation
manual testing
test data
keeping up with regression tests
Infrastructure / tools

TRIZ - 3 Steps

Every act of creation is first an act of destruction.
– Pablo Picasso

http://www.liberatingstructures.com/6-making-space-with-triz/

First alone, then in your group. **6min**

1. How can we reliably create **not done** products?

--> Go wild

First alone, then in your group. **6min**

2. Is there anything we are doing that resembles in any shape or form these activities?

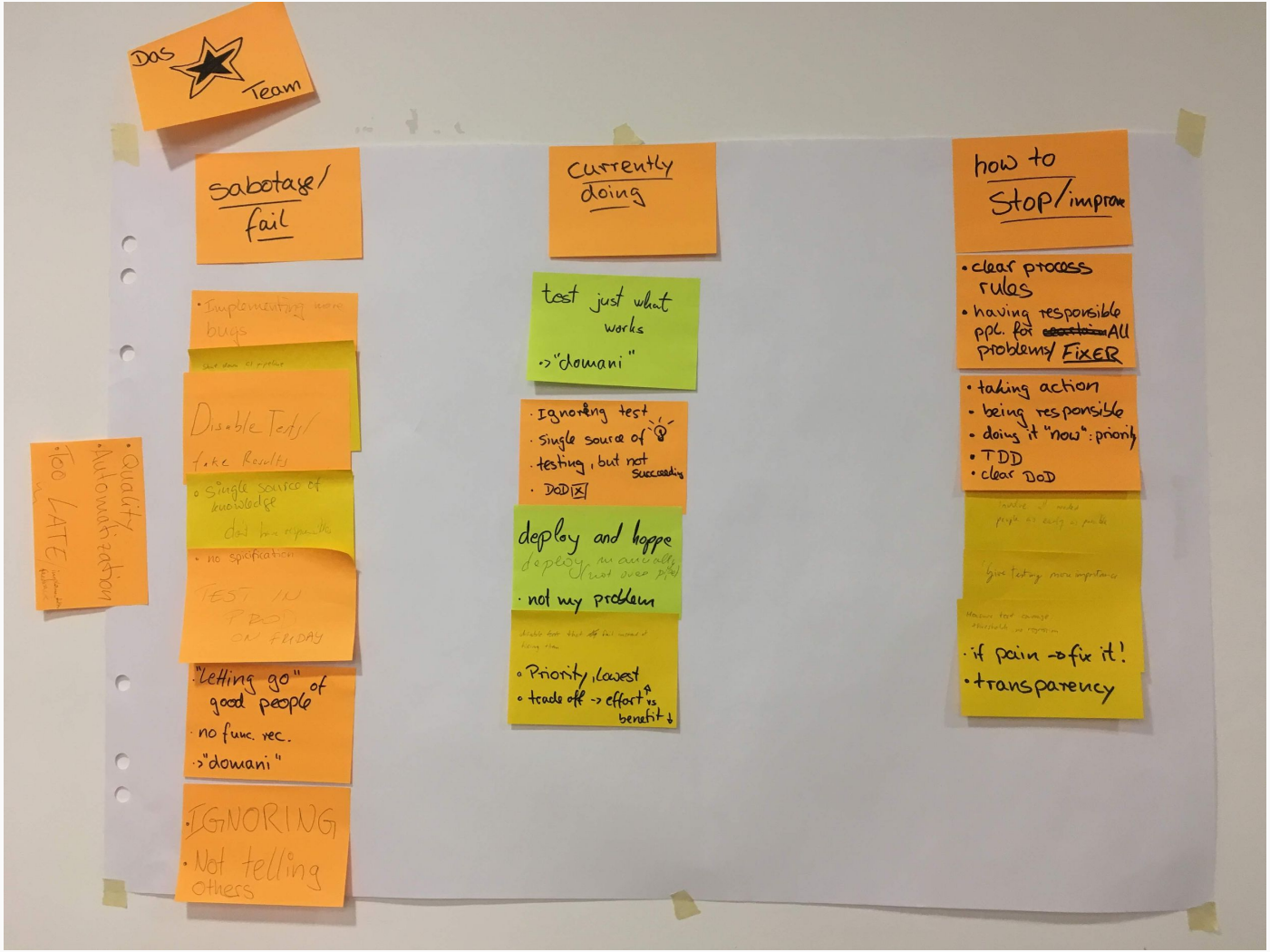
First alone, then in your group. **6min**

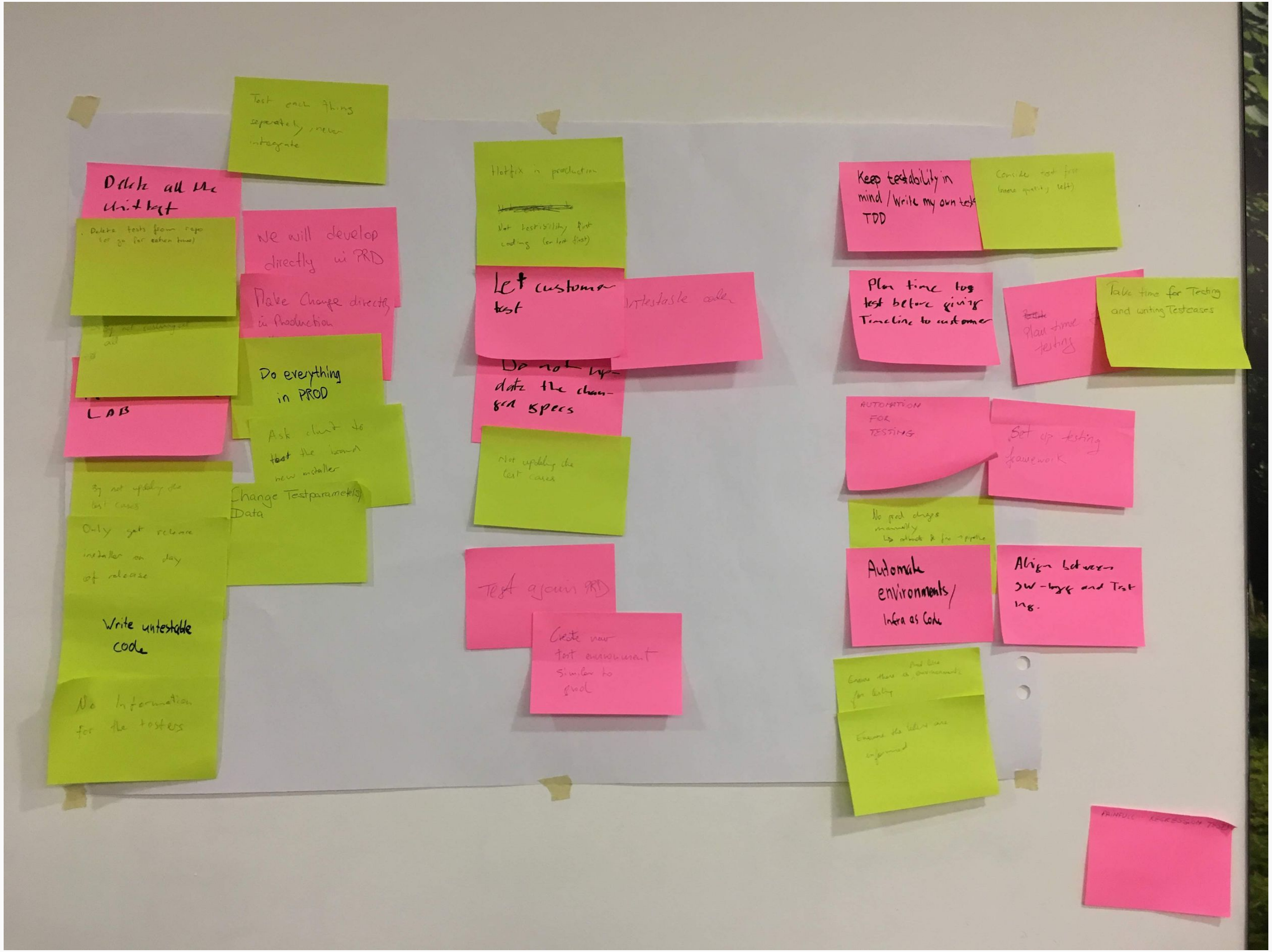
3. How am I going to stop it?

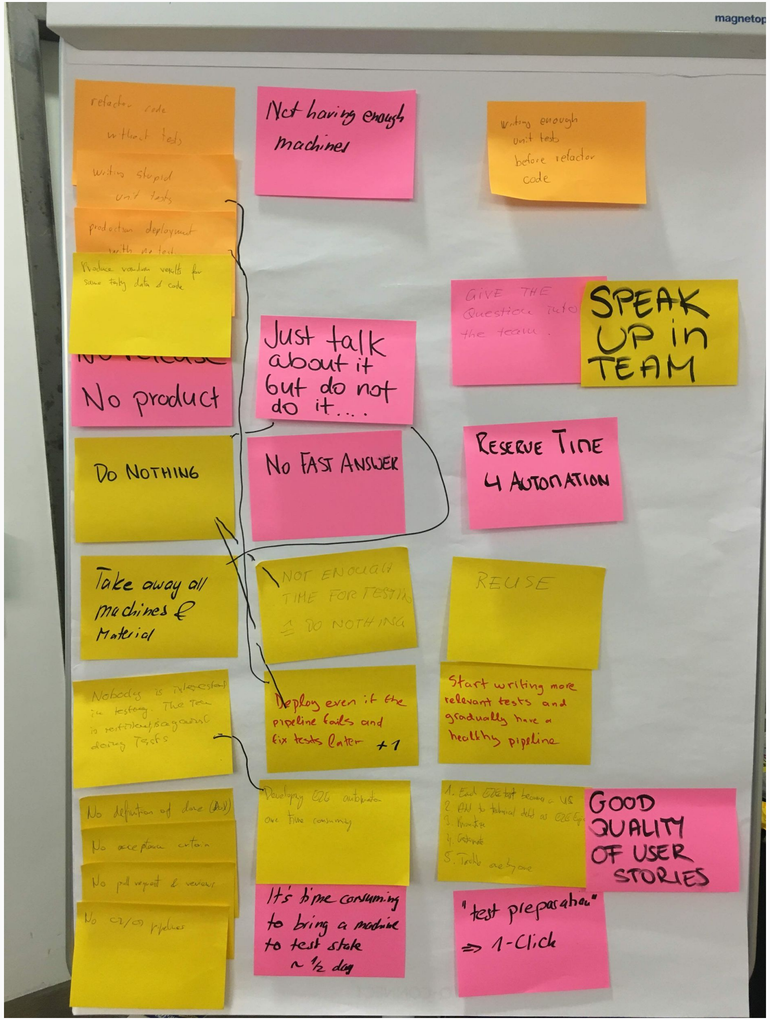
What is my first move?

“We will …”

“Who else needs to be included?”

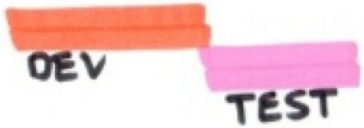

Das Team
sabotage/ fail
currently doing
how to Stop/improve
• clear process rules
• having responsible ppl. for ~~detection~~ All problems/ FIXER
• taking action
• being responsible
• doing it "now": prioritize
• TDD
• clear DoD
• transparency
test just what works
-> "domani"
• Ignoring test
• single source of knowledge
• testing, but not succeeding
• DoD [X]
deploy and hope
• not my problem
• Priority, Lowest
• trade off -> effort ↑ vs benefit ↓
• Implementing more bugs
Disable Tests/ fake Results
• single source of knowledge
• no specification
TEST IN PROD ON FRIDAY
"Letting go" of good people
• no func. rec.
-> "domani"
IGNORING
• Not telling others
• Quality
• Automatization
TOO LATE


Test each thing separately, never integrate
Delete all the unit test
Delete tests from repo
We will develop directly in PRD
Make Change directly in Production
Do everything in PROD
Change Testparameter Data
Only get release installer on day of release
Write untestable code
No Information for the testers
Hotfix in production
Let customer test
Not updating the test cases
Test again PRD
Create new test environment similar to prod
Keep testability in mind / Write my own tests TDD
Plan time for test before giving Timeline to customer
Plan time testing
Take time for Testing and writing Testcases
AUTOMATION FOR TESTING
Set up testing framework
Automate environments / Infra as Code
Align between
PAINFULL REGRESSION TEST

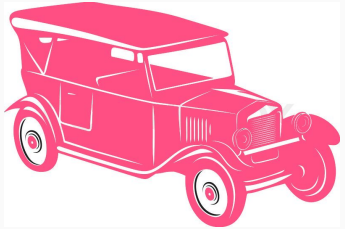

refactor code without tests
writing stupid unit tests
production deployment with no test
No product
Not having enough machines
Just talk about it but do not do it....
writing enough unit tests before refactor code
SPEAK UP in TEAM
Do NOTHING
No FAST ANSWER
RESERVE TIME 4 AUTOMATION
Take away all machines & Material
REUSE
Deploy even if the pipeline fails and fix tests later +1
Start writing more relevant tests and gradually have a healthy pipeline
GOOD QUALITY OF USER STORIES
It's time consuming to bring a machine to test state ~ 1/2 day
"test preparation" => 1-Click

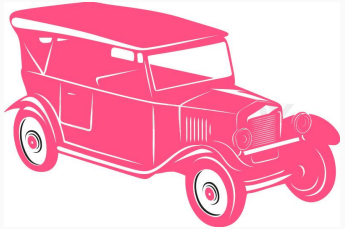

Erste Hilfe bei Elektrounfällen
Bergen des Verunfallten
1. Beurteilung
2. Alarmieren
3. Atmung kontrol
4. Massnahmen
5. Beatmung
6. Defibrillator (AED)
write tests which cannot fail (falsely report success)
V-model
Increase Feature Pressure
DEV/TEST
Gouvernance
Integrate quality to the pipeline
Awareness
Prioritize features/ time

Traditional Testing vs. Testing in DevOps

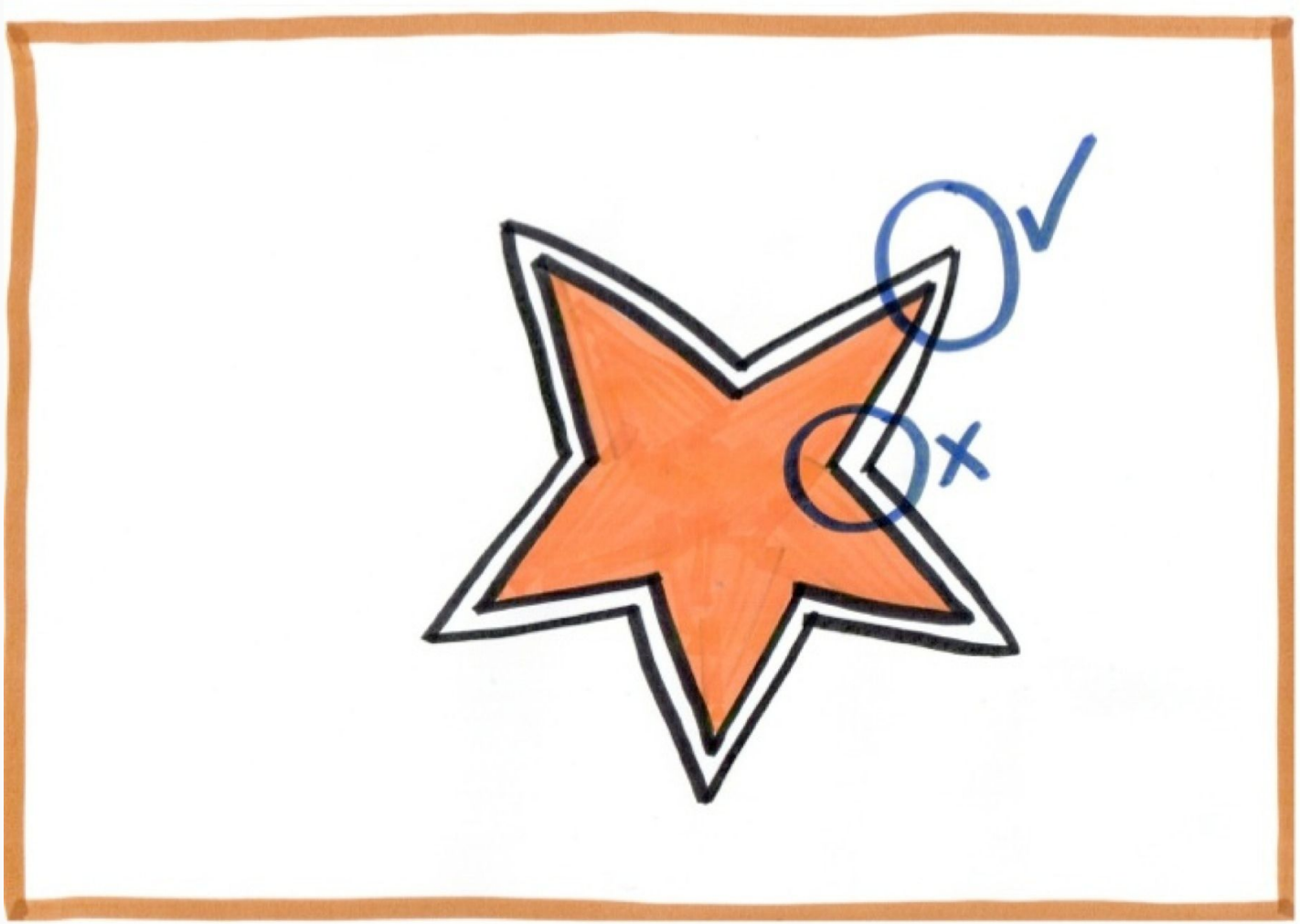


Testing is a phase

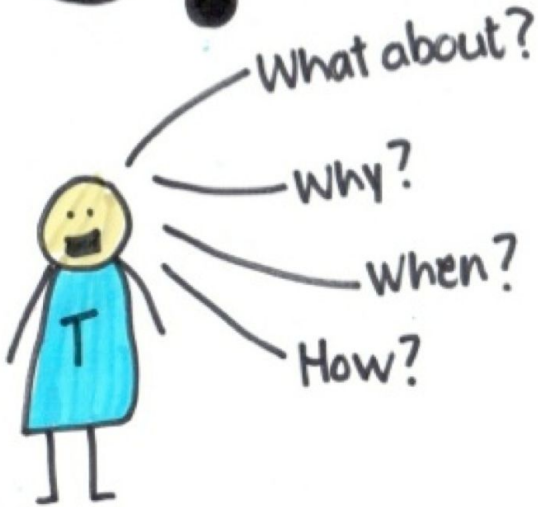


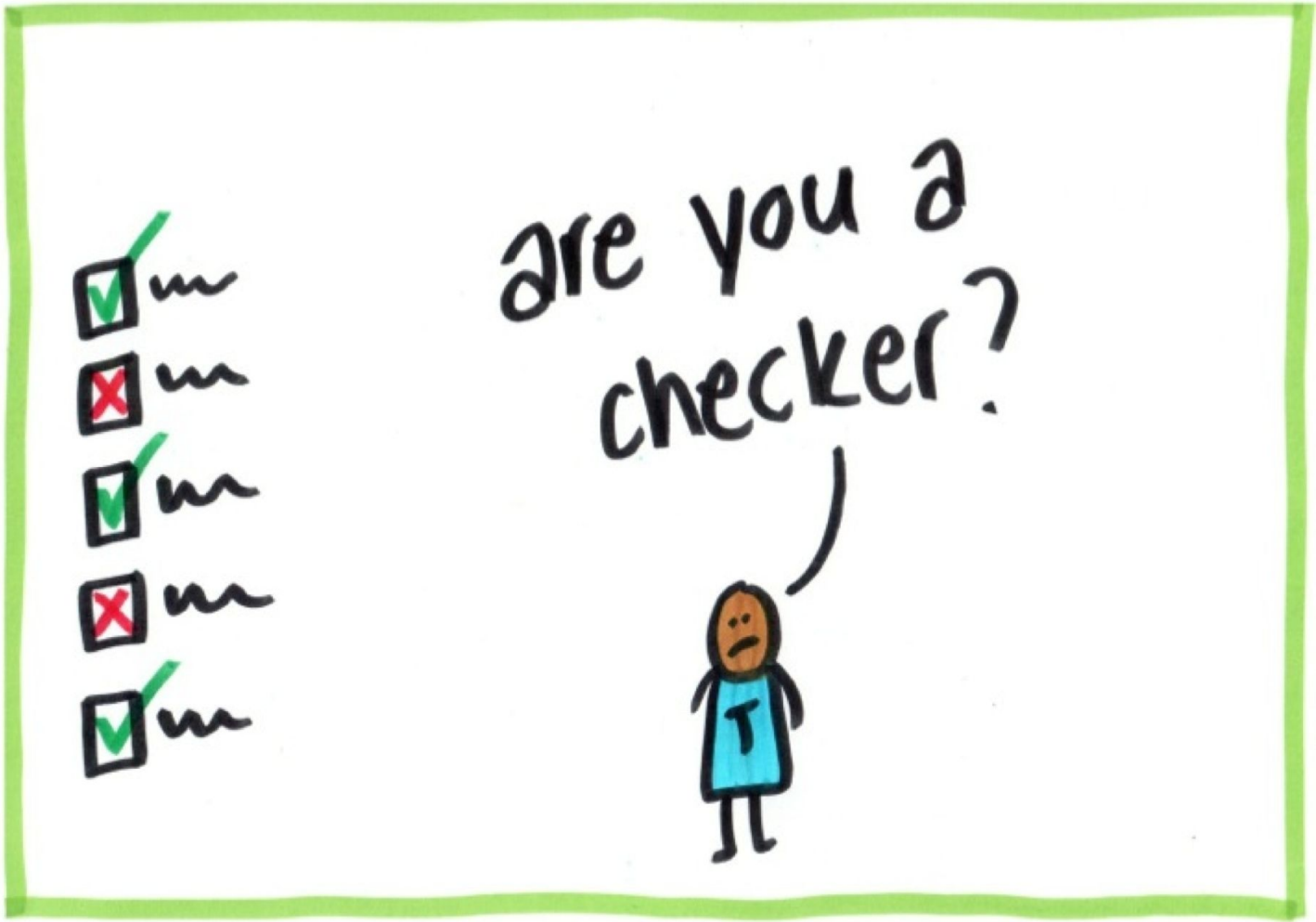


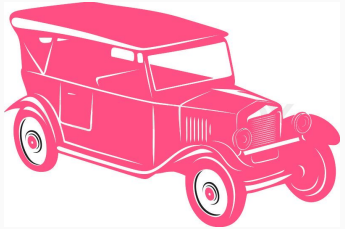
Get measured by bugs found.
Artificially introduced


✓
x

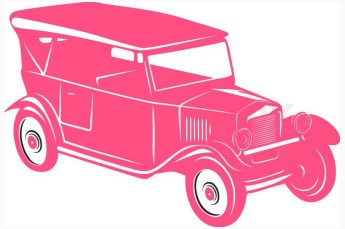
PREVENT BUGS!





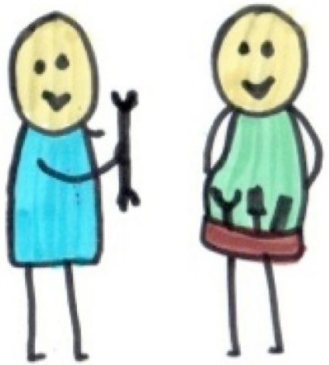


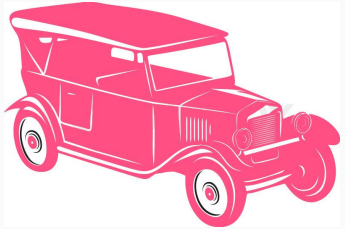

be a tester!
How can we test that?
How do you...
Well, first...
T
DESK OF USER



Breaking the system

Helping to build the BEST system





QA Team got removed since Quality was so high.

Whole team takes responsibility for quality

Posters
Gallery Walk
+ Russian Roulette?

Walk to poster that…

You feel comfortable with

First alone, then in your group. 5min

Discuss

Add practices that worked

Walk to poster that…

Confuses you

First alone, then in your group. 6min

Discuss

Add observations that helped

Walk to poster that…

What is worthy yet illusive?

Was ist wertvoll (bemerkenswert), aber illusorisch bei uns?

First alone, then in your group. 6min

Discuss

Add observations that helped

Close Up

Hands up - Who thinks ...

Is it possible to prevent bugs BEFORE you write code.

The whole team can be responsible for quality.

It is possible to delight the customer without pressure in the Sprint.

It’s possible to be part of a happy DevOps team.

In Pair. **2min**

What is your take away?

In Pair. **2min**

Finish this sentence

“If only they would …”

In Pair. **2min**

Finish this sentence

“If only I …”

In Pair. **2min**

What are next steps for you?

THE TESTING Manifesto

we value:

Testing throughout **OVER** testing at the end	Preventing bugs **OVER** finding bugs	Testing understanding **OVER** checking functionality	Building the best system **OVER** breaking the system	Team responsibility for quality **OVER** tester responsibility

www.GrowingAgile.co.za @growingAgile

What will you do different tomorrow?

Continue the conversation

@peitor

Grab the book

https://beyond-agility.com/books/