

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

ED

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

ONE IDEA
PER
STICKY
NOTE ✓

≠

- IDEA 1
- IDEA 2
- IDEA 3

X

FEW
WORDS ✓

≠

AVOID TOO MUCH
WRITING ON ONE
SINGLE NOTE AS
IT TAKES A WHILE
FOR PEOPLE TO
READ THE WHOLE
THING X

EASY
TO
READ ✓

≠

It doesn't
help to
write in
cursive
OR IN PEN X

CAPITAL
LETTERS

≠

Lowercase
letters

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 bug 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?”

DOS



Team

Sabotage/ fail

- Implementing more bugs

check state of system

Disable Tests/ fake results

- Single source of knowledge
- no specification

TEST IN
P.DOS
ON FRIDAY

"Letting go" of good people

- no func. rec.
- "downani"

IGNORING

- Not telling others

currently doing

test just what works

→ "downani"

- Ignoring test
- single source of \uparrow
- testing, but not \uparrow succeeding
- DOD \boxtimes

deploy and hope

deploy, in \uparrow actually
(not over ppl)

- not my problem

check how that affects user's
usage time

- Priority, lowest
- trade off → effort \uparrow vs benefit \downarrow

how to stop/improve

- clear process rules
- having responsible ppl. for ~~everything~~ ALL problems/ FIXER

- taking action
- being responsible
- doing it "now": priority
- TDD
- clear DOD

write it down
prior to doing it

give testing more importance

Measure test coverage
threshold as requirement

- if pain → fix it!
- transparency

Quality
Automation
Too LATE

Test each thing separately, never integrate

Delete all the unit test

Delete tests from repo
let go for other team

we will develop directly in PROD

Make change directly in production

Do everything in PROD

Ask about it that the would new member

change Testparameter, Data

LAB
by not update the code
Only get version in later on day of release

Write untestable code

No information for the testes

Happy in production
~~Unit test~~
No testability for coding (code first)

Let customer test

intestable code

Do not update the client specs

not update the test cases

Test again PROD

Create new test environment similar to prod

Keep testability in mind / Write my own test TDD

Consider not for more quality tests

Plan time to test before giving Timeline to customer

Plan time testing

Take time for Testing and writing Testcases

AUTOMATION FOR TESTING

Set up testing framework

No code change manually by other people

Automate environments / Infra as Code

Align between SW-logs and Test logs

Ensure that environment for testing

Ensure the test cases are updated

PRODUCTION TESTS

release code
without tests
writing stupid
unit tests

Not having enough
machines

bring enough
unit tests
before refactor
code

production deployment
with no test
Produce code with
some tiny data / code

No release

Just talk
about it
but do not
do it...

GIVE THE
Questions into
the team

**SPEAK
UP IN
TEAM**

Do NOTHING

No FAST ANSWER

RESERVE TIME
4 AUTOMATION

Take away all
machines &
Material

NOT ENOUGH
TIME FOR TESTING
=> DO NOTHING

REUSE

Methodology is important
in testing. The team
is not implementing
writing Tests

Deploy even if the
pipeline fails and
fix tests later +1

Start writing more
relevant tests and
gradually have a
healthy pipeline

No definition of done (DOD)
No acceptance criteria
No pull request & review

Deploying code without
on the running

1. Don't start testing - US
2. Add to backlog as a story
3. Write
4. Review
5. Talk to customer

**GOOD
QUALITY
OF USER
STORIES**

No CI/CD pipeline

It's time consuming
to bring a machine
to test state
~ 1/2 day

"test preparation"
=> 1-Click

Erste Hilfe bei Elektrounfällen

Berger des Verunfallten

1. Beurteilung
2. Alarmieren
3. Atmung kontrol
4. Massnahmen
5. Beatmung
6. Defibrillator (AED)

100% Coverage
- solution of
- ...

Somebody is going
long the way

Index reviewed

... with 100%
... and ...
... on ...

...
...
...

Awareness
↳ training /
Know How
on all levels

...
...
...

integrate quality
to the pipeline

Prioritize features/
time

...

...

dev/test
concurrently

Gouvernance

- ...
- ...

• ...
...
...

...
...
...

Do DD
(...)

Write tests which
cannot fail
(falsely report success)

V-model
...
...

Increase
Feature
Pressure



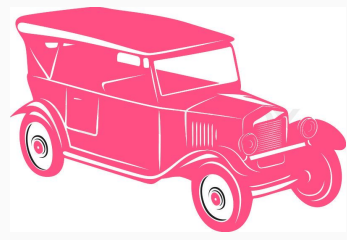
Traditional Testing

vs.

Testing in DevOps

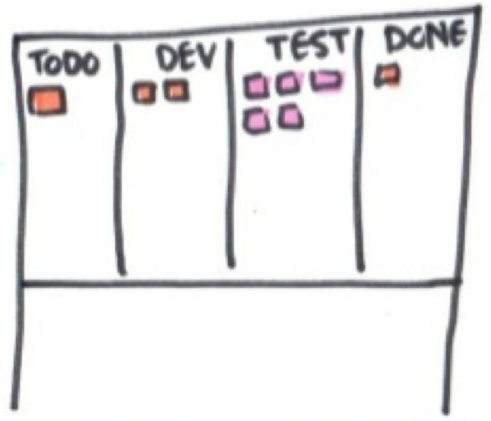
DEV

TEST



Testing is a phase

I have my own column.

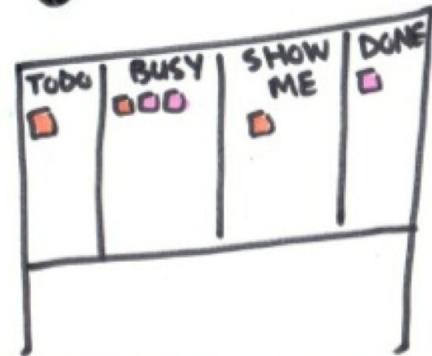





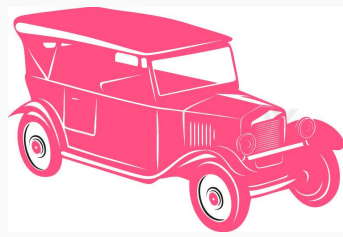
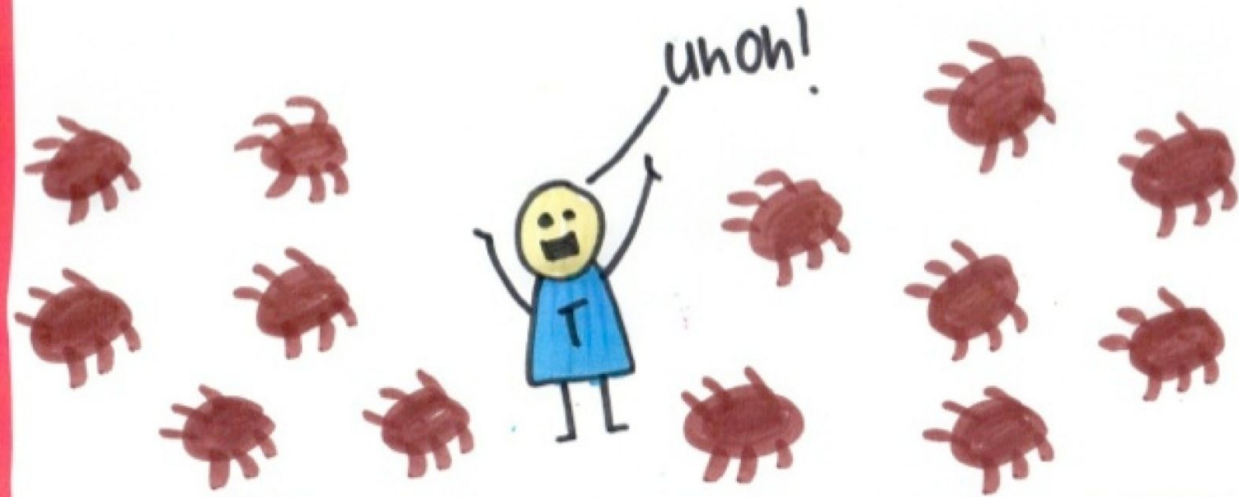
D D T D
T D T T
D T D D

Testing is an ACTIVITY!

I'm part
of the
board!



Finding bugs...



Get measured by bugs found.
Artificially introduced







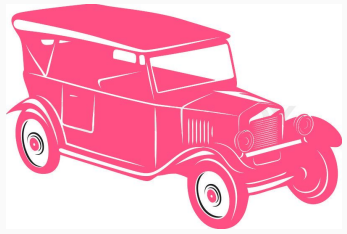
PREVENT BUGS!



- What about?
- Why?
- When?
- How?

- m
- m
- m
- m
- m

are you a
checker?



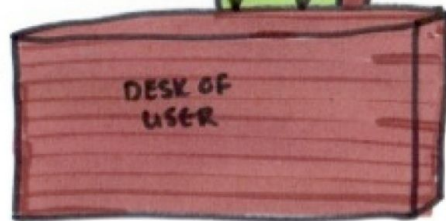


be a tester!

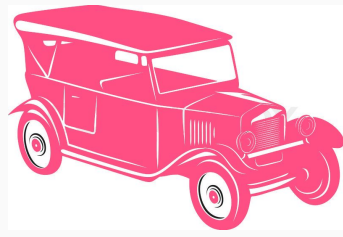
How can we test that?

How do you....

Well, first...



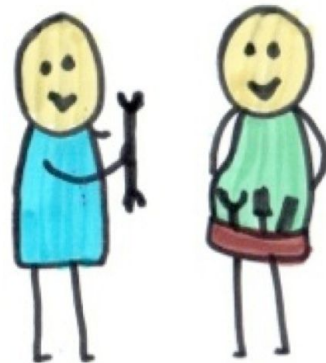
Breaking the system



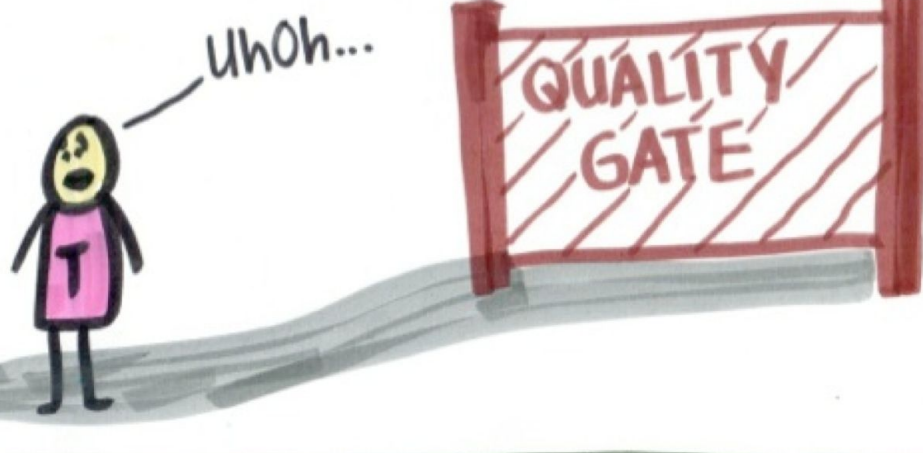
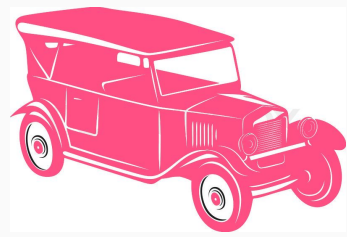
You build it, I break it. Not a bug.



Helping to build the
BEST system



Tester is responsible
for quality





Whole team takes
responsibility for quality



We're ready
to ship!

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

What will you do different tomorrow?

Continue the conversation



Grab the book

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