Examples
how to move towards
Zero Defects

Niels Malotaux:
"In my experience the 'zero defects' attitude results in 50% less defects almost overnight."

join Q&A at Slido.com with #test2018

niels@malotaux.nl

www.malotaux.nl/conferences
Do we deliver Zero Defect software?

- Who is a tester?
- Do the requirements specify a certain number of defects?
- Do you check that the required number has been produced?

In your projects

- How much time is spent putting defects in?
- How much time is spent trying to find and fix them?
- Do you sometimes get repeated issues?
- How much time is spent on defect prevention?

Software development process:

- 1st phase is developing phase
- 2nd phase is de-bugging phase

Better quality costs less
What is a defect?

A defect is the cause of a problem experienced by any of the stakeholders while relying on our results.
What is Zero Defects

• Zero Defects is an asymptote

• When Philip Crosby started with Zero Defects in 1961, errors dropped by 40% almost immediately

• AQL > Zero means that the organization has settled on a level of incompetence

• Causing a hassle other people have to live with
Crosby (1926-2001) - Absolutes of Quality

- Conformance to requirements
- Obtained through prevention
- Performance standard is zero defects
- Measured by the price of non-conformance (PONC)

Philip Crosby, 1970
Ultimate Goal of a What We Do

Delivering the Right Result at the Right Time, wasting as little time as possible (= efficiently)

Providing the customer with
- what he needs
- at the time he needs it
- to be satisfied
- to be more successful than he was without it

Constrained by (win - win)
- what the customer can afford
- what we mutually beneficially and satisfactorily can deliver
- in a reasonable period of time
Root Cause Analysis to feed prevention

- Is Root Cause Analysis routinely performed – every time?
- What is the Root Cause of a defect?

- **Cause:**
The error that caused the defect

- **Root Cause:**
What caused us to make the error that caused the defect

- Without proper Root Cause Analysis, we’re doomed to repeat the same errors
We're QA: What has this to do with us?

- What is the goal of QA in a software development project?
Who is the (main) customer of Testing and QA?

- Deming:
  - Quality comes not from testing, but from *improvement of the development process*
  - Testing does not improve quality, nor guarantee quality
  - It’s too late
  - The quality, good or bad, is already in the product
  - You cannot test quality into a product

- Who is the main customer of Testing and QA?
- What do we have to deliver to these customers?
  - What are they waiting for?
- Testers and QA are consultants to development

Deming (1900-1993)
Some Examples

We’re not perfect, but the customer shouldn’t find out
Design techniques

- Design
- Review
- Code
- Review

Iterate as needed

- Test (no questions, no issues)
- If issue in test: no Band-Aid: start all over again:
  Review: What’s wrong with the design?
- Reconstruct the design (if the design description is lacking)
- What happens if you ask "Can I see the DesignLog?"

Cleanroom
In the pub

James:
Niels, this is Louise
Louise, this is Niels, who taught me about DesignLogging
Tell what happened

Louise:
We had only 7 days to finish some software
We were working hard, coding, testing, coding, testing
James said we should stop coding and go back to the design
"We don't have time!" - "We've only 7 days!"
James insisted
We designed, found the problem, corrected it, cleaned up the mess
Done in less than 7 days
Thank you!
What James told me afterwards

- I gave the design to two colleagues for review
- Louise corrected some minor issues
- It went into a ‘final’ review, with another colleague
- Based in his expertise, the solution was completely reworked
- Actually, two features were delivered and deployed
  - The one that was design and code reviewed had no issues after deployment
  - The other one was the source of quite some defects
- In summary, this success has proved instrumental in buy-in for DesignLogs which are now embedded in the development process
There are many ways to represent a design

- Only few are useful
- Don't waste reviewer's time
Useful design?
Choose the appropriate design

47 pages documentation condensed into one page
How could it look like?
What is better than reviewing code?

• Do you ever review software?
• What do you review?

• What is better than reviewing code?
  • May I review the design first?
Case: Scrum Sprint Planning

• What is the measure of success for the coming sprint?
• “What a strange question! We're Agile, so we deliver working software. Don't you know?”
• Note: Users are not waiting for software: they just need improved performance of what they’re doing
• How about a requirement for 'Demo': No Questions – No Issues
• How's that possible !?!
• They actually succeeded!
Demo ??

• Give the delivery to the stakeholders
• Zip your mouth
• Keep your hands handcuffed on your back
• and o-b-s-e-r-v-e what happens
• Seeing what the stakeholders actually do provides real feedback
• Then we can ‘talk business’ with the stakeholders
• Is this what you do?
The ‘Demo’

Concurrent database record update
Delivery Strategy Suggestions (Requirements)

• What we deliver will be used by the appropriate users immediately, within one week not making them less efficient than before

• If a delivery isn’t used immediately, we analyse and close the gap so that it will start being used (otherwise we don’t get feedback)

• The proof of the pudding is when it’s eaten and found tasty, by them, not by us

• The users determine success and whether they want to pay (we don’t have to tell them, but it should be our attitude)
Case: How much legwork is being done in your project?

- Requirements/specifications were trashed out with product management
- Technical analysis was done and
- Detail design for the first delivery

At the first delivery:
- James: How is the delivery? (quality versus expectation)
- Adrian: It's exactly as expected, which is absolutely unprecedented for a first delivery
  The initial legwork has really paid off
Some techniques shown

• Design
• Drawings
• DesignLog
• Review
• No Questions – No Issues

A Zero Defects attitude makes an immediate difference
Basic approach

- Design the requirement
- Review
- Design implementation
- Review
- Implement (code ?)
- Review
- Test doesn’t find issues (because they’re not there)

Iterate fast, as needed
What’s in it for QA?

- Did we see much testing in the previous?
- Testing shouldn’t find anything (because there should be no issues)
- Did you ever find similar issues as you found before?
  - First time: Developers ‘fault’
  - Second time: Testers ‘fault’

Develop  -->  Test  -->  Repair

What we often see

Develop  -->  Check  -->  Act

What we should expect

- QA to help developers to produce less and less defects
Do we deliver Zero Defect software?

- How many defects are acceptable?
- Do the requirements specify a certain number of defects?
- Do you check that the required number has been produced?

In your projects
- How much time is spent putting defects in?
- How much time is spent trying to find and fix them?
- Do you sometimes get repeated issues?
- How much time is spent on defect prevention?
- Could you use “No Questions – No Issues”? 

Better quality costs less
Approaching Zero Defects is Absolutely Possible

If in doubt, let's talk about it

Niels Malotaux

niels@malotaux.nl

www.malotaux.nl/conferences