software pilots
TRIFORK.

Er vi komplette idioter?

Jesper Boeg og Ole Friis Østergaard

March 23. and 24., 2011

Agenda

- Agile Reminder
- The economic perspective
- TDD
- You get what you measure!
- Right perception of time
- Transparency
- Pair Programming
- Trust
- Fail fast
- Force of habit
- Summary

HOW MANY BELIEVE AGILE IS THE RIGHT WAY TO GO?

HOW MANY ARE DOING AGILE DEVELOPMENT?

So naturally we are going to:

1. Satisfy the Customer through Working Software
2. Deliver Early and Often
3. Create and Embrace Change
4. Focus on Quality
5. Create Transparency through Visualization
6. Endorse Sustainable Pace
7. Bring People Closer Together
8. Trust in People and Decentralize Authority
9. Improve Continuously

We *know* XP “best practices”!

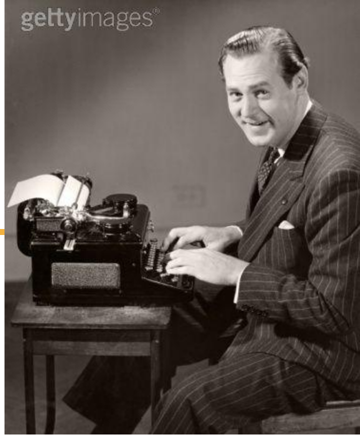
- Talk to the customer
- Practice pair programming
- Do TDD
- Review your code
- Clean up your code

WHY?

Because it makes economic sense

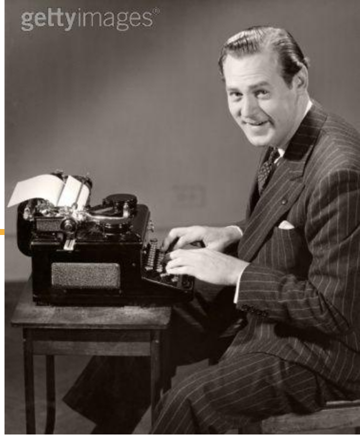
1. Satisfy the Customer through Working Software
2. Deliver Early and Often
3. Create and Embrace Change
4. Focus on Quality
5. Create Transparency through Visualization
6. Endorse Sustainable Pace
7. Bring People Closer Together
8. Trust in People and Decentralize Authority
9. Improve Continuously

TDD (done right)



- You’re constantly reminded of the whereabouts of your bad code
- Non-decoupled code is hard to test – use this knowledge to split up into more methods / classes / services
- Small bugs are quickly ironed out
- By naming your tests, you’re forced to explain what your code does and why

TDD (done wrong)



- If you don’t realise that the cause of troublesome tests is bad code, writing tests can be very frustrating
- Integration tests (which actually has nothing to do with TDD) are slow and fragile
- If just *one* team member doesn’t care about the tests, everybody else suffers

BUT ARE WE REALLY DOING IT?

YOU GET WHAT YOU MEASURE!

TRIFORK.

You Get What You Measure 1/2

- Don’t kid yourself!
- When you measure story points you get story points!
- You don’t get:
  - – Functional Quality
  - – Maintainability
  - – Long term focus
  - – Sustainable pace

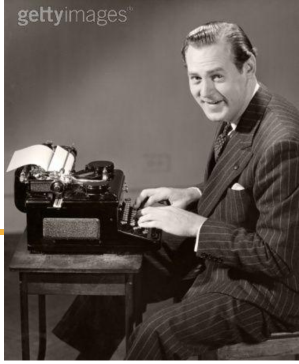
You Get What You Measure 2/2

- When your plan becomes your success criteria what you get is your ability to follow a plan
- You don’t get:
  - A product that fits your customer’s need
  - To see change as a business opportunity
  - Continuous improvement
  - Better economics by deferring decisions
  - Better economics by limiting WIP
  - Light weight change management procedures
  - Transparency, because it is simply too hurtful

PERCEPTION OF TIME

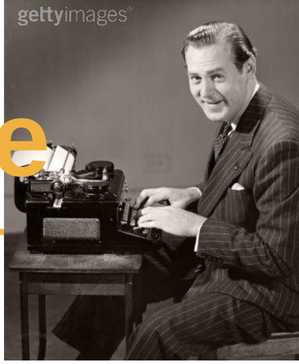
THE RIGHT PERCEPTION OF TIME WITH THE RIGHT TOOLS

Problems with SCRUM



- We always have small deadlines that we strive to make
- Just after a sprint, it’s easy to dismiss lacking code reviews, clean-ups, etc.
- “Clean Code” is a luxury we can only afford when it doesn’t get in the way of the sprint estimates.

Our twisted perception of time



- We don’t do pair programming, because it takes too long to discuss what to code
- We don’t do TDD because we then spend time on the test code
- We don’t learn new tools because they take time to learn
- We don’t chat with the customer because they might tell us we’re wrong

TRANSPARENCY… IS THAT SOME KIND OF EXOTIC ANIMAL?

Transparency is good

- Transparency surfaces problems quicker while they are still fixable
- Transparency makes alignment much easier because of a shared overview
- Transparency makes people work closer together

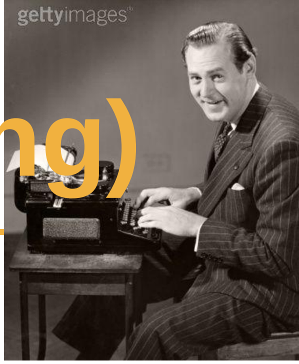
But Transparency also means

- You cannot hide your mistakes
- You have to accept reality
- You no longer get to believe in magic or wishful thinking
- You have to trust each other to focus on what is important for the project

Pair Programming (done right)

- Code review is built-in
- You’re more likely to refactor along the way, and push each other to perform your best
- Small misconceptions are found faster
- You don’t get stuck or run off on a tangent for days
- The team is not completely disfunctional the day the LDAP expert is away

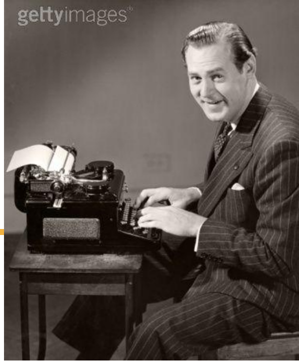
Pair Programming (done wrong)



- Experienced developers can get frustrated working with complete novices
- Different meeting times on the team minimizes potential pair-up time
- An odd number of team members...?
- Off-days, meetings, small interruptions...
- Lack of small breaks during the day can drain all of your energy

RIPPING YOUR HEART OUT

Personal issues



- For many, pair programming is too personal
- You suddenly have to explain what you do and why
- You cannot hide bad habits

But... Hey?

That’s how we get better!

TRUST… ISN'T THAT WISHFUL HIPPIE THINKING?

When you trust

- You can decentralize authority
- You can focus on the shared goal
- You don’t need wasteful control procedures
- You can collaborate instead of negotiate

Yeah, I Like the Idea of Trust But:

- This particular Customer, Client, Supplier, Developer will turn it against me
- This fixed price contract is strangling our economy
- I am not sure they would understand
- What if the project fails?

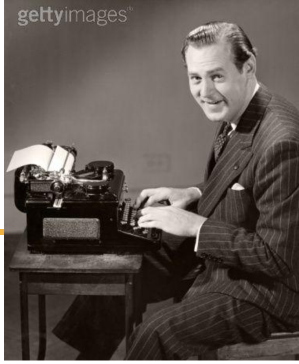
FAILURE IS NOT AN OPTION!

What Are We Afraid of?

- What happened to fail early?
- What happened to failure as learning opportunity?
- Will you be distrusted and ridiculed because you admit to not being perfect?
- Is it really a failure if you still have time to fix what went wrong?
- Do people really expect you to be a fortune teller?

THE POWER OF HABITS

About habits



- Too many of us keep the habits all the way from university/school
- Why don’t we change them?
- Do we feel a constant pressure forcing us to ”just” complete the next task before we pull ourselves together?
- How, then, are we going to move our profession forward?

But we have to change habits!

- If we cannot do it ourselves, we need help!
- Where’s the SCRUM master?
- How about ”coding dojos” once in a while?
- How about helping each others learning new tools and using each others across teams?

So sometimes we don't always

1. ~~Satisfy the Customer through Working Software~~
2. ~~Deliver Early and Often~~
3. ~~Create and Embrace Change~~
4. ~~Focus on Quality~~
5. ~~Create Transparency through Visualization~~
6. ~~Endorse Sustainable Pace~~
7. ~~Bring People Closer Together~~
8. ~~Trust in People and Decentralize Authority~~
9. ~~Improve Continuously~~

Could it be because deep down?

- We are afraid to Fail
- We are afraid to Trust
- We are afraid of Transparency
- We Measure the wrong things