Help !
We have a QA Problem !

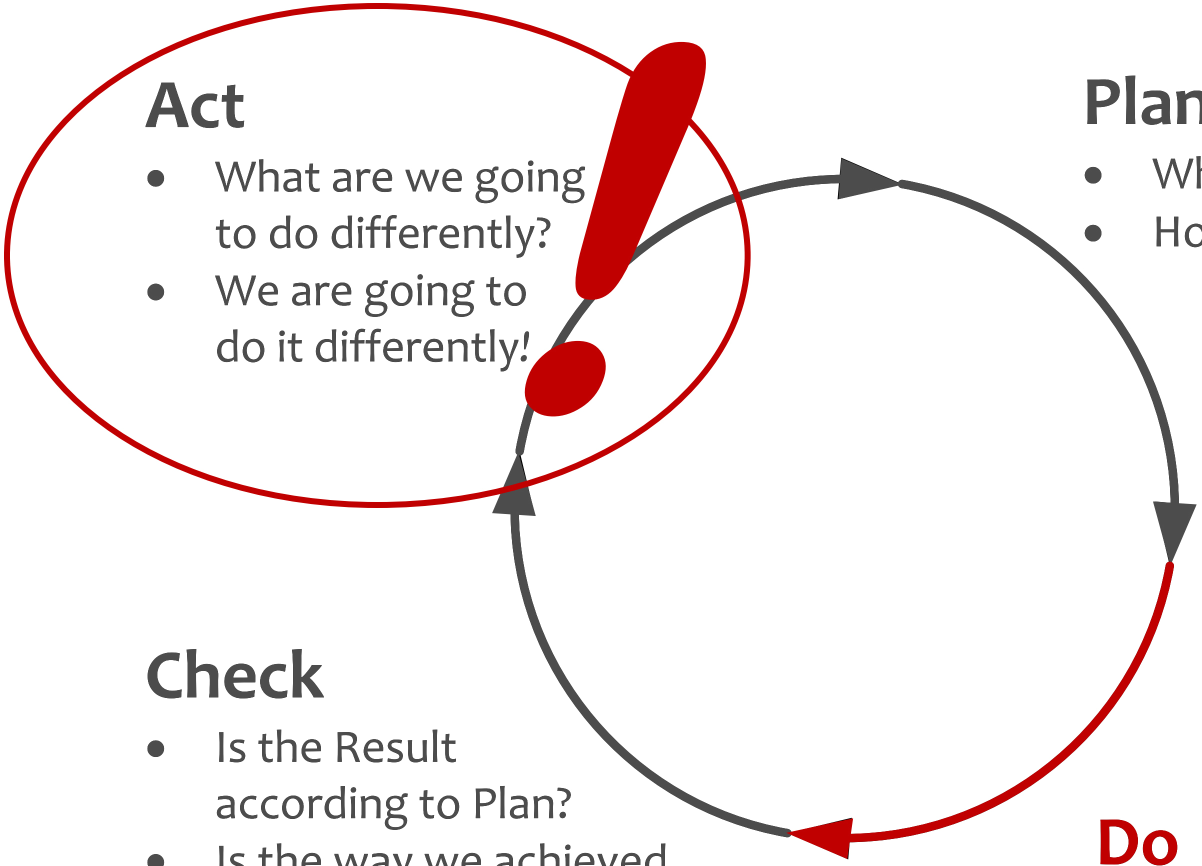
Niels Malotaux

We have a QA problem !

- Large stockpile of modules to test
  (hardware, firmware, software)
- You shall do Full Regression Tests
- Full Regression Tests take about 15 days each
- Too few testers ("Should we hire more testers ?")
- Senior Tester paralyzed
- Can we do something about this?

The essential ingredient: the PDCA Cycle

(Shewhart Cycle - Deming Cycle - Plan-Do-Study-Act Cycle - Kaizen)



Act

- What are we going to do differently?
- We are going to do it differently!

Plan

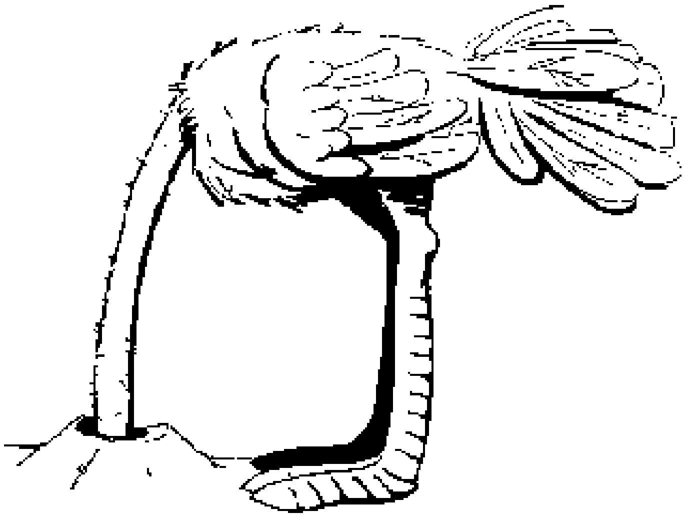
- What to achieve
- How to achieve it

Check

- Is the Result according to Plan?
- Is the way we achieved the Result according to Plan?

Do

Carry out the Plan



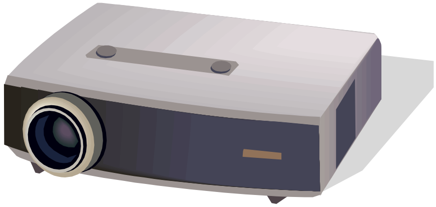
Instead of complaining about a problem …

(Stuck in the Check-phase)

Let’s do something about it !

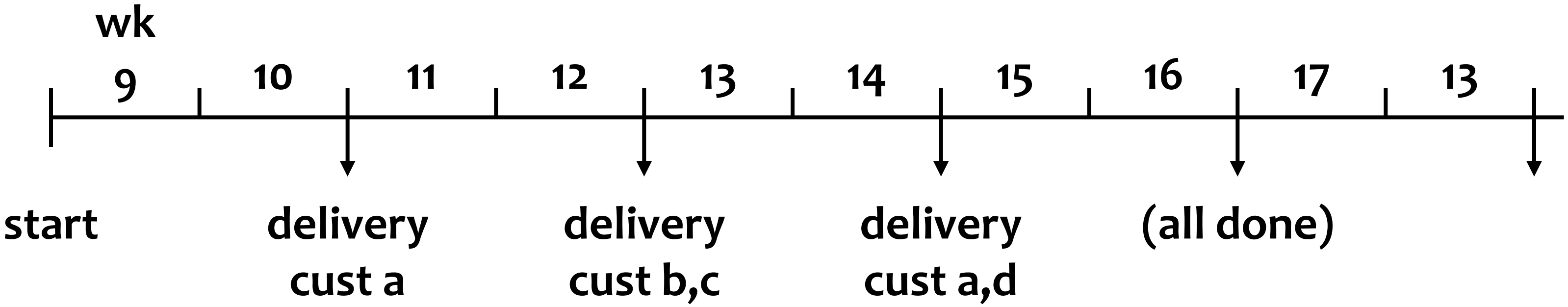
(Moving to the Act-phase)

Objectifying and quantifying the problem is a first step to the solution



Line	Activity	Estim	Alter native	Junior tester	Devel opers	Customer	Will be done (now=22Feb)
1	Package 1	17	2	17	4	HT	
2	Package 2	8	5		10	Chrt	
3	Package 3	14	7	5	4	BMC	
4	Package 4 (wait for feedback)	11				McC?	
5	Package 5	9	3		5	Ast	
6	Package 6	17	3	10	10	?	
7	Package 7	4	1		3	Cli	
8	Package 8.1	216	1			Sev	
9	Package 8.2	1	1			?	
10	Package 8.3	1	1			Chrt	24 Feb
11	Package 8.4	1	1			Chrt	
12	Package 8.5	1.1	1.1			Yet	28 Feb
13	Package 8.6	3	3			Yet	24 Mar
14	Package 8.7	0.1	0.1			Cli	After 8.5 OK
15	Package 8.8	18	18			Ast	
	totals	106	47	32	36		

TimeLine



wk

9 10 11 12 13 14 15 16 17 13

start

delivery
cust a

delivery
cust b,c

delivery
cust a,d

(all done)

Selecting the priority order of customers to be served

- “We’ll have a solution at that date … Will you be ready for it ?”
  Another customer could be more eagerly waiting
- Most promising customers

Result

- Tester empowered
- Done in 9 weeks
- So called “Full Regression Testing” was redesigned
- Customers systematically happy and amazed
- Kept up with development ever since
- Increased revenue

Recently:

- Tester promoted to product manager
- Still coaching successors how to plan

Universal Project Goal

Quality on Time

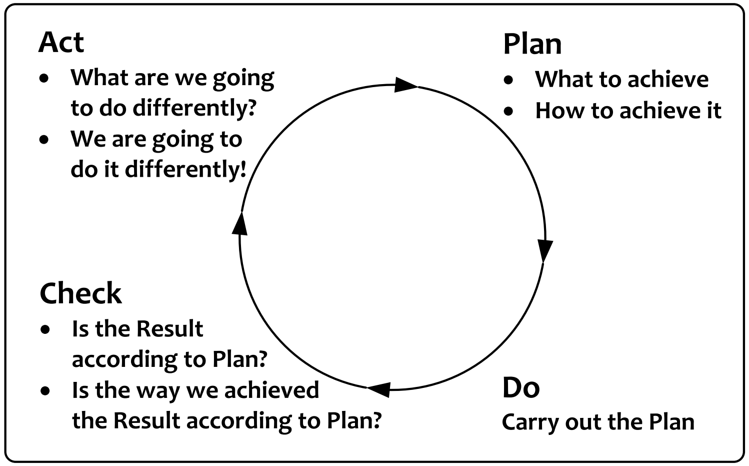
- Delivering the Right Result at the Right Time, wasting as little time as possible (= efficiently)

- Providing the customer with
  - what he needs
  - at the time he needs it
  - to be satisfied
  - and 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

Who is the 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.
- Developers are the customer
- Testers help developers to become perfect
- Testing is a project to run alongside and synchronized to the development project
- Therefore, it must be organised like any other project

Evo



Act
- What are we going to do differently?
- We are going to do it differently!

Plan
- What to achieve
- How to achieve it

Check
- Is the Result according to Plan?
- Is the way we achieved the Result according to Plan?

Do
Carry out the Plan

- Evo (short for Evolutionary...) uses PDCA consistently
- Applying the PDCA-cycle
  actively, deliberately, rapidly and frequently,
  for *Product*, *Project* and *Process*, based on RoI and highest *value*
- Combining Planning, Requirements- and Risk-Management into *Result Management*
- We know we are not perfect, but the customer shouldn't find out
- Evo is about delivering Real Stuff to Real Stakeholders doing Real Things

"Nothing beats the Real Thing"

- Projects seriously applying Evo, routinely conclude successfully on time, or earlier

Evolutionary Project Management (Evo)

Zero Defects Attitude

- Plan-Do-Check-Act
  - The powerful ingredient for success
- Business Case
  - *Why* we are going to improve *what*
- Requirements Engineering
  - *What* we are going to improve *and what not*
  - *How much* we will improve: quantification
- Architecture and Design
  - Selecting the optimum compromise for the conflicting requirements
- Early Review & Inspection
  - Measuring quality while doing, learning to prevent doing the wrong things

Right product

Evo Project Planning

- Weekly TaskCycle
  - Short term planning
  - Optimizing estimation
  - Promising what we can achieve
  - Living up to our promises
- Bi-weekly DeliveryCycle
  - Optimizing the requirements and checking the assumptions
  - Soliciting feedback by delivering Real Results to *eagerly waiting* Stakeholders
- TimeLine
  - Getting and keeping control of Time: Predicting the future
  - Feeding program/portfolio/resource management

Right time

The aim of Testing

- Being done as soon as the development is done
- Well, almost

- Excuses, excuses, excuses
  - The developers are always late
    (Evo developers live up to their promises)
  - The developers don't take us seriously
    (Evo developers ask testers for help)
  - The developers don't inject enough defects
    (now testing becomes a real challenge)
- Helping development to be successful

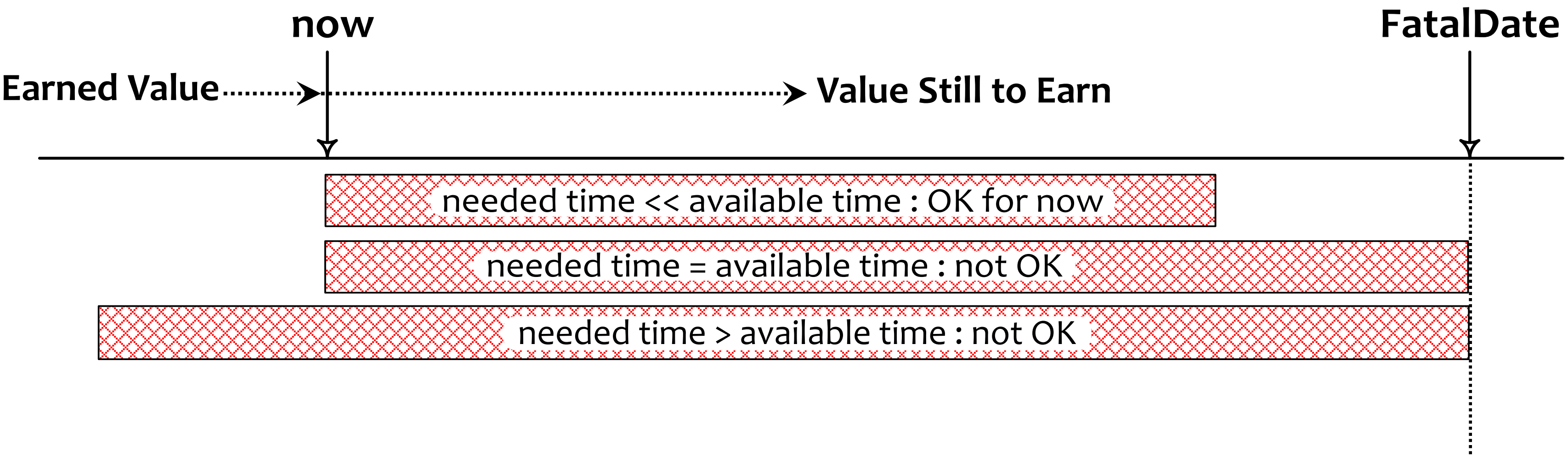
TimeLine

- Cutting the work into chunks
- Estimating
- Adding up (averages the uncertainties !)
- Usually doesn’t fit in the available time
- Find strategies to solve the dilemma
- Select ‘optimum’ strategy
- Predict what will happen when
- Learn and repeat every week, keeping predictions up-to-date
- Allow Portfolio Management to manage rather than to game

TimeLine: Predicting *what* will be done *when*

Line	Activity	Estim	Spent	Still to spend	Ratio real/es	Calibr factor	Calibr still to	Date done
1	Activity 1	2	2	0	1.0			
2	Activity 2	5	5	1	1.2	1.0	1	30 Mar 2009
3	Activity 3	1	3	0	3.0			
4	Activity 4	2	3	2	2.5	1.0	2	1 Apr 2009
5	Activity 5	5	4	1	1.0	1.0	1	2 Apr 2009
6	Activity 6	3				1.4	4.2	9 Apr 2009
7	Activity 7	1				1.4	1.4	10 Apr 2009
8	Activity 8	3				1.4	4.2	16 Apr 2009
↓	↓							
16	Activity 16	4				1.4	5.6	2 Jun 2009
17	Activity 17	5				1.4	7.0	11 Jun 2009
18	Activity 18	7				1.4	9.8	25 Jun 2009

What do we do if we see we won’t make it on time ?


now
FatalDate
Earned Value
Value Still to Earn
needed time << available time : OK for now
needed time = available time : not OK
needed time > available time : not OK

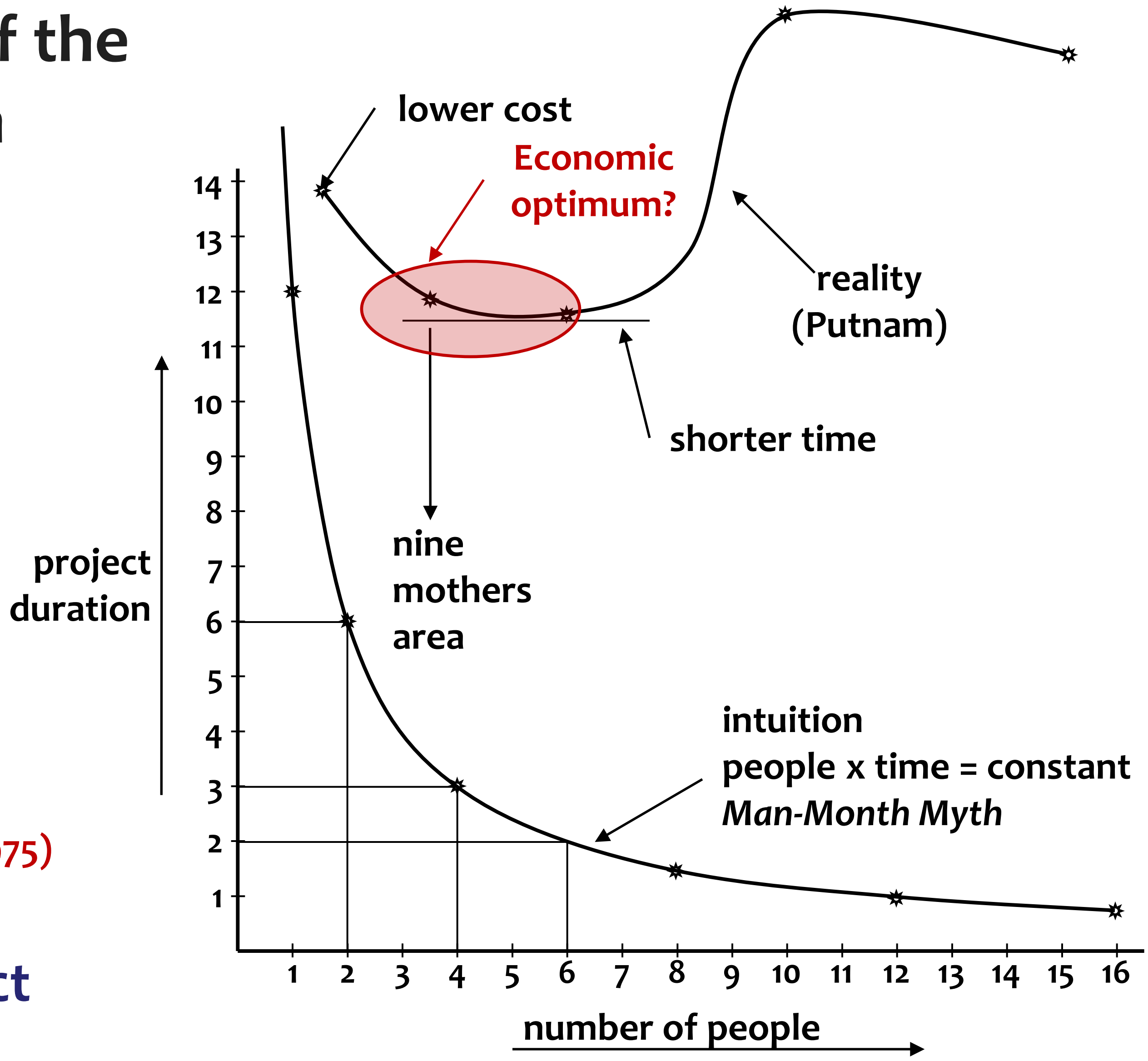
If it doesn’t fit ... count backwards

When the match is over, you can’t score a goal any more

Deceptive options

- Hoping for the best (fatalistic)
- Going for it (macho)
- Working Overtime (fooling ourselves)
- Moving the deadline
  - – Parkinson’s Law
    - Work expands to fill the time for its completion
  - – Student Syndrome
    - Starting as late as possible, only when the pressure of the FatalDate is really felt

The Myth of the Man-Month



lower cost

Economic optimum?

reality (Putnam)

shorter time

nine mothers area

intuition
people x time = constant
Man-Month Myth

Brooks' Law (1975)

Adding people to a late project *makes it later*

Saving time

We don’t have enough time, but we can save time *without negatively affecting the Result !*

- Efficiency in *what* (*why*, for *whom*) we do - doing the right things
  - *Not* doing what later proves to be superfluous
- Efficiency in how we do it - doing things differently
  - The product
    - Using proper and most efficient solution, instead of the solution we always used
  - The project
    - Doing the same in less time, instead of immediately doing it the way we always did
  - Continuous improvement and prevention processes
    - Constantly learning doing things better and overcoming bad tendencies
- Efficiency in when we do it - right time, right order
- TimeBoxing - much more efficient than FeatureBoxing

More

www.malotaux.nl/Booklets

1 **Evolutionary Project Management Methods (2001)**
Issues to solve, and first experience with the Evo Planning approach

2 **How Quality is Assured by Evolutionary Methods (2004)**
After a lot more experience: rather mature Evo Planning process

3 **Optimizing the Contribution of Testing to Project Success (2005)**
How Testing fits in

3a **Optimizing Quality Assurance for Better Results (2005)**
Same as Booklet 3, but for non-software projects

4 **Controlling Project Risk by Design (2006)**
How the Evo approach solves Risk by Design (by process)

5 **TimeLine: How to Get and Keep Control over Longer Periods of Time (2007)**
Replaced by Booklet 7, except for the step-by-step TimeLine procedure

6 **Human Behaviour in Projects (APCOSE 2008)**
Human Behavioural aspects of Projects

7 **Evolutionary Planning, or How to Achieve the Most Important Requirement (2008)**
Planning of longer periods of time, what to do if you don't have enough time

8 **Help ! We have a QA Problem ! (2009)**
Use of TimeLine technique: How we solved a 6 month backlog in 9 weeks

RS **Measurable Value with Agile (Ryan Shriver - 2009)**
Use of Evo Requirements and Prioritizing principles

www.malotaux.nl/nrm/Insp

Inspection pages

Help !
We have a QA Problem !

Niels Malotaux