Software Practice Advancement
Bedfordshire, 16-19 March 2008

Tutorial

Niels Malotaux

And you thought you could estimate ?

N R Malotaux - Consultancy
The Netherlands
+31-30-2288868
+31-30-2288869
niels@malotaux.nl
www.malotaux.nl

SPA2008 - 16-19 March 2008

Niels Malotaux

And you thought you could estimate ?

Niels Malotaux

Niels Malotaux is an independent Project Coach specializing in optimizing project performance. He has over 30 years experience in designing hardware and software systems, at Delft University, in the Dutch Army, at Philips Electronics and 20 years leading his own systems design company. Since 1998 he devotes his expertise to helping projects to deliver Quality On Time: delivering what the customer needs, when he needs it, to enable customer success. To this effect, Niels developed an approach for effectively teaching Evolutionary Project Management (Evo) Methods, Requirements Engineering, and Review and Inspection techniques. Since 2001, he taught and coached some 80 projects in 20+ organizations in the Netherlands, Belgium, Ireland, India, Japan, Romania and the US, which led to a wealth of experience in which approaches work better and which work less well in practice. He is a frequent speaker at conferences, see www.malotaux.nl/nrm/Conf .

Niels puts development teams on the Quality On Time track and coaches them to stay there and deliver their quality software or systems on time, without overtime, without the need for excuses. Practical methods are developed, used, taught and continually optimized for:

- Evolutionary Project Management (Evo)
- Requirements Engineering and Management
- Reviews and Inspections.

Within a few weeks of turning a development project into an Evo project, the team has control and can tell the customer when the required features will all be done, or which features will be done at a certain date. Niels enjoys greatly the moments of enlightenment experienced by his clients when they find out that they can do it, that they are really in control, for the first time in their lives.

N R Malotaux
Consultancy

Niels Malotaux
project coach

Result Management

Bongerdlaan 53
3723 VB Bilthoven
The Netherlands
tel +31-30-228 88 68
fax +31-30-228 88 69
mob +31-6-5575 3604
niels@malotaux.nl
www.malotaux.nl

And you thought you could estimate ?

Let's find out !

Niels Malotaux

N R Malotaux Consultancy

+31-30-228 88 68 niels@malotaux.nl www.malotaux.nl

1

The Right Result On Time

- Do you regularly deliver the Right Result On Time?
- Why not?
- Is this normal?
- Can we do something about it?

- What is the Right Result?
- What is On Time?

2

Is it difficult to be on time?

- **Did anyone miss a plane?**
- **What did you feel?**
- **Why did it happen?**
- **Did it happen again?**

3

The problem

- **Many projects don't deliver the right Results**
- **Many projects deliver late**

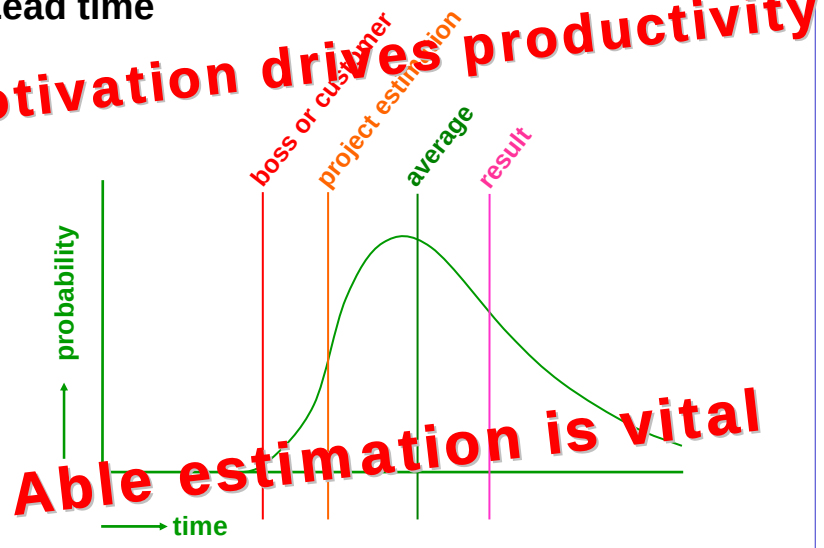
or, more positively:

- **I want my project to be more successful**
- **In shorter time**

4

Lead time

Motivation drives productivity



Able estimation is vital

5

How about *your* projects?

- **Are *you* any better than *they*?**

6

Estimation Exercise

Are you an optimistic or a realistic estimator?

Let's find out *!*

Project:
Multiplying two numbers of 4 figures

How many seconds would you need to complete this Project?

7

Is this what you did?

8

Defect rate

- **Before test ?**
- **After test ?**

9

Alternative Design (*how* to solve the requirement)

10

Another alternative design

11

What was the real requirement?

Assumptions, asssumptions ...

Better assume that many assumptions are wrong.

Check !

12

Elements in the exercise

- **Estimation, optimistic / realistic**
- **Interrupts**
- **Test, test strategy**
- **Defect-rate**
- **Design**
- **Requirements**
- **Assumptions**

13

Human Behavior

- **Systems are conceived, designed, implemented, maintained, used, and tolerated *(or not)* *by* people**
- **People react quite predictably**
- **However, often differently from what we intuitively think**

- **Most project process approaches (as well as developers) *ignore* human behavior, incorrectly *assume* behavior, or decide how people *should* behave (ha ha)**
- **To succeed in projects, we must study and adapt to *real* behavior rather than *assumed* behavior**

14

Discipline

- Control of wrong inclinations
- Even if we *know* how it *should* be done …
  (if nobody is watching …)
- Discipline is very difficult
- Romans 7:19
  - For the good that I would I do not …

→ We must help each other (watching over the shoulder)
→ Rapid success helps
→ Making mistakes helps

15

Intuition

- Makes you react on every situation
- Intuition is fed by experience
- It is free, we always carry it with us
- Sometimes intuition is simply wrong
- In many cases the head knows, the heart not
- Coaching is about redirecting intuition

16

Communication

- **Traffic accident: witnesses tell *their* truth**
- **Same words, different *concepts***
- **Human brains contain rather fuzzy concepts**
- **Try to explain to a colleague**
- **Writing it down is explaining it to paper**
- **If it's written it can be discussed and changed**
- **Vocal communication evaporates immediately**
- **E-mail communication evaporates in a few days**

17

Perception

- **Quick, acute, and intuitive cognition (M-W)**
- **What people say and what they do is not always equal**
- **The head knows, but the heart decides**
- **Hidden emotions are often the drivers of behavior**
- **Customers who said they wanted lots of different ice cream flavors from which to choose,
still tended to buy those that were fundamentally vanilla**

- **So, trying to find out what the real value to the customer is, can show many paradoxes**
- **Better not simply believe what they say: check!**

18

Murphy's Law

- Whatever can go wrong, will go wrong
- Should we accept fate?

Murphy's Law for Engineers:

- Whatever can go wrong, will go wrong …

Therefore:

- We should actively check all possibilities that can go wrong and make sure that they cannot happen

19

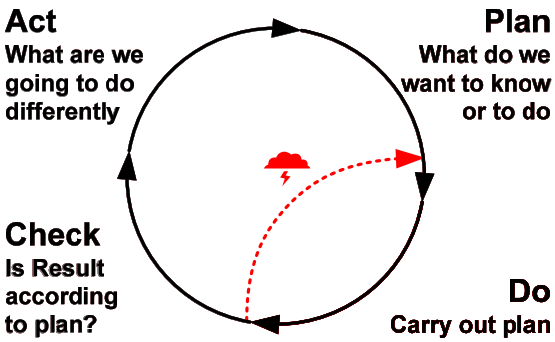
Preflection, foresight, prevention

- Only if we *change* our way of working, the result may be different
- Hindsight is easy, but reactive
- Foresight is less easy, but proactive
- Reflection is for hindsight and learning
- *Preflection* is for foresight and prevention
- Only with *prevention* we can save precious time
- Wasn't this used in the Deming or Plan-Do-Check-Act cycle?

20

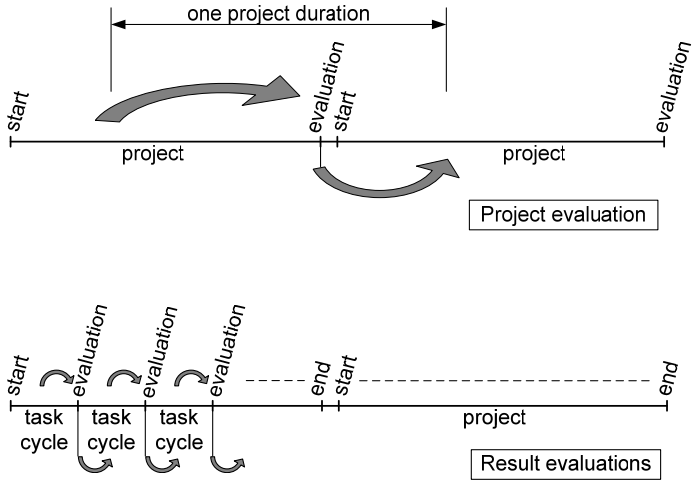
The essential ingredient: the PDCA cycle

(Deming cycle)



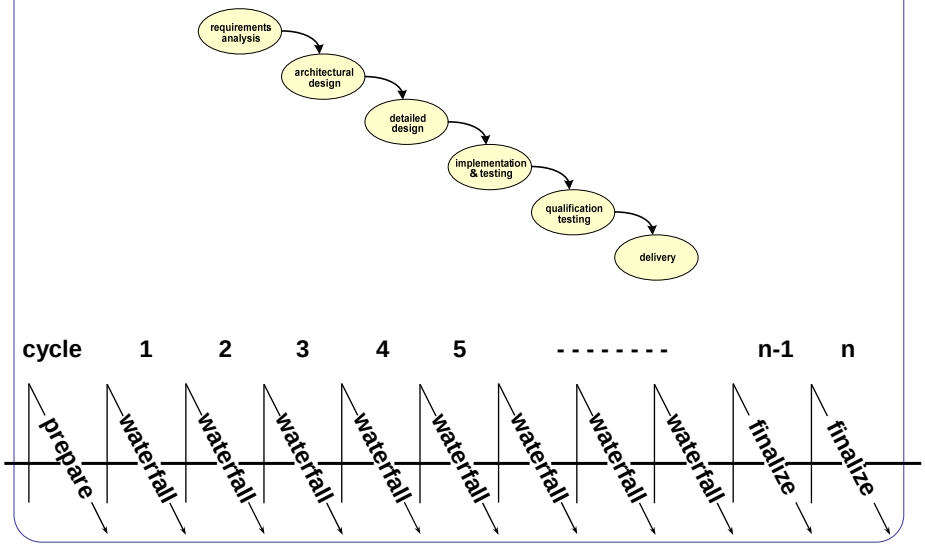
21

Project evaluations



22

Using many waterfalls of growing functionality



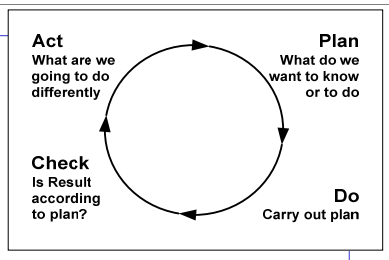
23

Project Goal

- 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
  - what the customer can afford
  - what we mutually beneficially and satisfactorily can deliver
  - in a reasonable period of time

24

Knowledge
how to achieve the goal



- Using very short Plan-Do-Check-Act cycles
- Constantly selecting the most important things to do

doing the right things

then we can

- Most quickly learn what the real requirements are
- Learn how to most effectively and efficiently realize these requirements

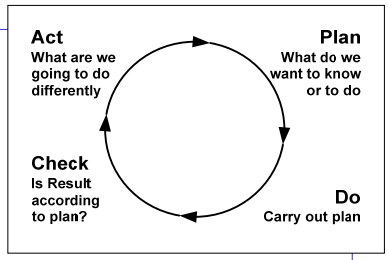
and we can

- Spot problems quicker, allowing more time to do something about them

doing the right things right

25

Evo



- Evo (short for Evolutionary...) uses PDCA consistently
- Applying the PDCA-cycle actively, deliberately, rapidly and frequently, for *Product*, *Project* and *Process*, based on ROI
- Combining Planning, Requirements- and Risk-Management into *Result Management*
- We know we are not perfect, but the customer should never find out
- Evo is about delivering Real Stuff to Real Stakeholders doing Real Things *"Nothing beats the Real Thing"*

26

Evo elements

- 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
- Agile Review & Inspection
  - Measuring the quality while we are doing, to prevent doing the wrong things

Zero Defects Attitude

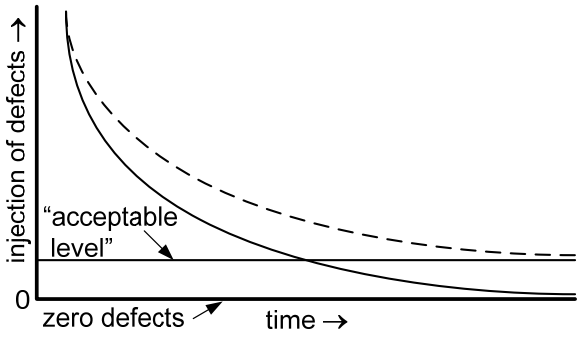
Evo planning

- Weekly TaskCycle
  - Short term planning
  - Optimizing estimation
  - Promising what you can achieve
  - Living up to your promises
- Bi-weekly DeliveryCycle
  - Optimizing the requirements and checking the assumptions
  - Soliciting feedback by delivering Real Results to appropriate and *eagerly waiting* Stakeholders
- TimeLine
  - Getting and keeping control of Time

27

Zero Defects

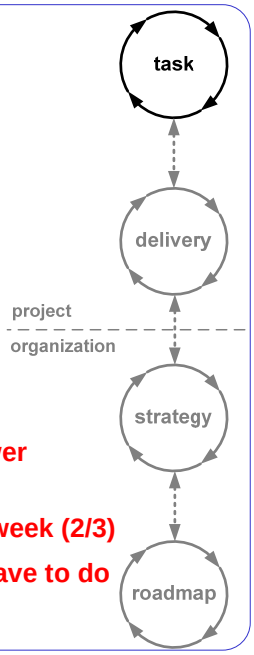
- Zero Defects is an asymptote



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

28

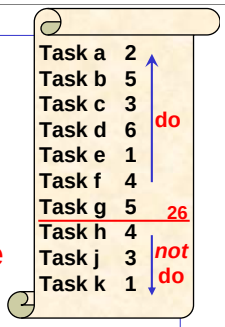
Cycles in Evo: Weekly TaskCycle



- Are we *doing*
  the *right things*,
  in the *right order,*
  to the right *level of detail for now*
- Optimizing estimation, planning and tracking abilities to better predict the future
- Select highest priority tasks, never do any lower priority tasks, never do undefined tasks
- There are only about 26 plannable hours in a week (2/3)
- In the remaining time: do whatever else you have to do
- Tasks are always done, 100% done

29

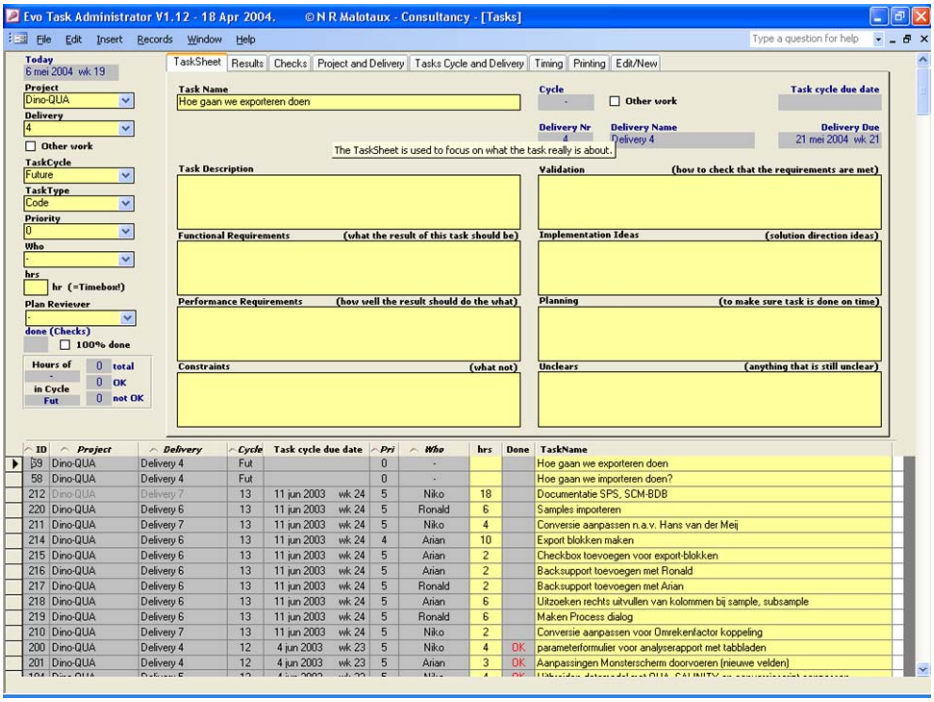
Every week we plan



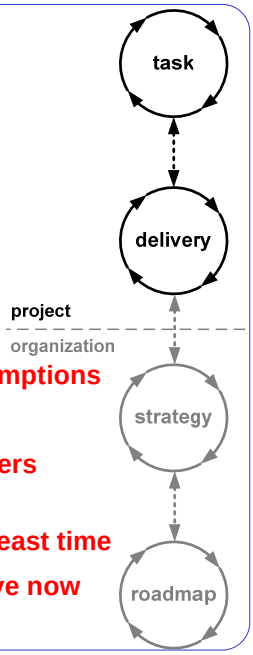
- How much time do we have available
- 2/3 of available time is net plannable time
- What is most important to do
- Estimate effort needed to do these things
- Which most important things fit in the net available time (default 26 hr)
- What can, and are we going to do
- What are we *not* going to do

- 2/3 is default start value
- This value works well in development projects

30



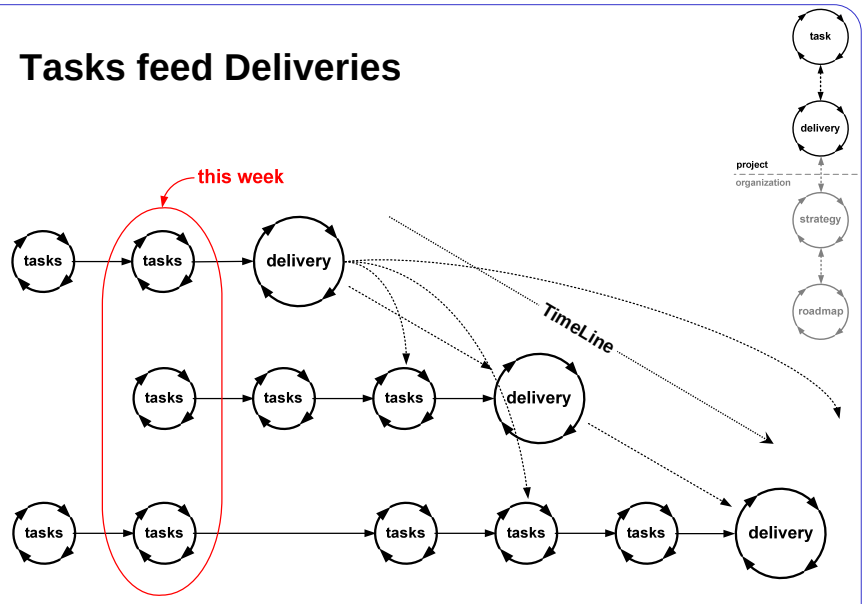
Cycles in Evo: DeliveryCycle



- Are we delivering the *right things*, in the *right order* to the right *level of detail for now*
- Optimizing requirements and checking assumptions
- What will generate the optimum feedback
- We deliver only to *eagerly waiting* stakeholders
- Delivering the juiciest, most important stakeholder values that can be made in the least time
- What will make Stakeholders more productive now
- Not more than 2 weeks

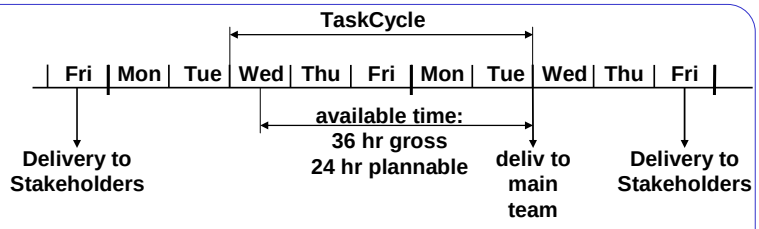
32

Tasks feed Deliveries



33

Designing a Delivery



Serge (ProjLead)	
MbWA	3
Planning nxt wk	3
Work for deliv	4
-	6
-	2
-	1
-	5
Total	24

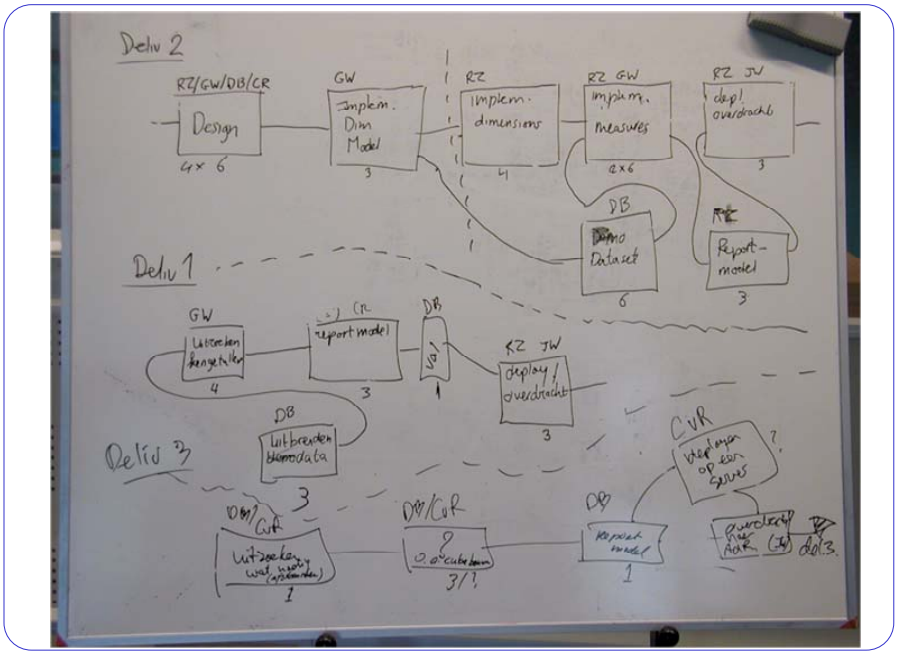
Gregory	
Draft design	6
Finish design	6
Work for deliv	3
-	1
-	2
-	2
-	3
-	5
-	6
XMLa	4
XMLb	4
Total	42

Gregory (later)	
Draft design	0
Finish design	0
...	
Repair deliv	0
...	

Zero Defects Attitude

Jerome	
XMLa	3
XMLb	3
...	

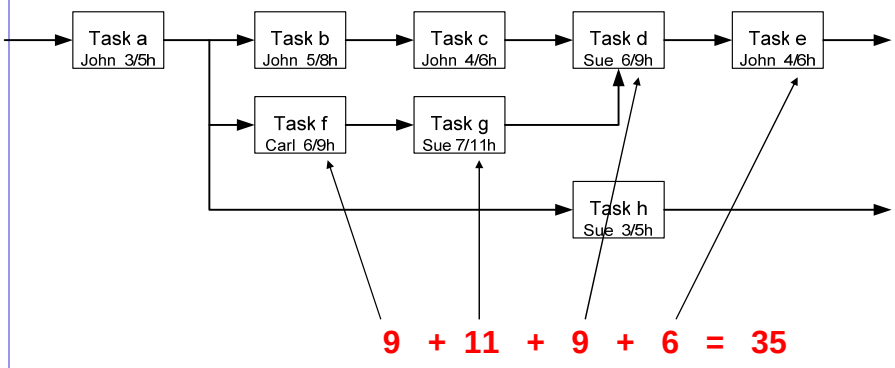
34



35

PERT (Project Evaluation Review Technique)

used for *Designing* a Delivery



36

Agile, but Still On Time

- **Organizing the work in very short cycles**
- **To make sure we are doing the right things**
- **And that we are doing it the right way**
- **So, we already work more efficiently**

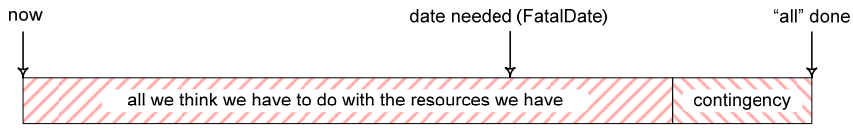
but ...

- **How do we make sure the whole project is done on time?**

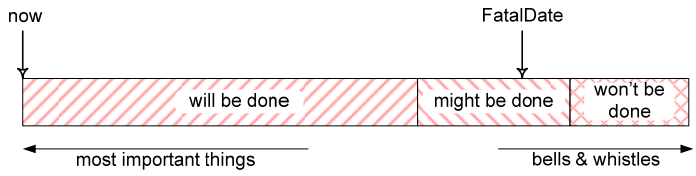
37

TimeLine

What the customer wants, he cannot afford



Standard Projects



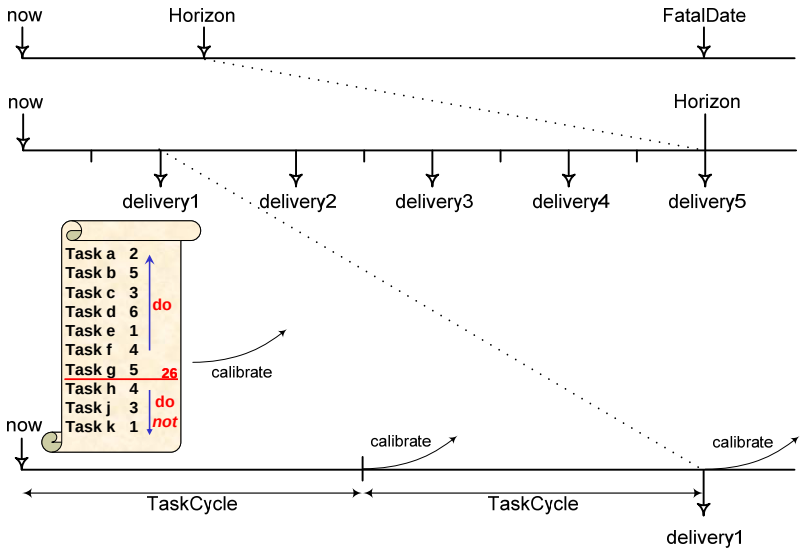
Evo

38

If it easily fits ...

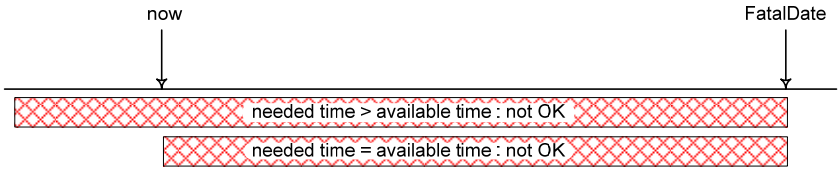
39

Result to Tasks and back



40

If it doesn't fit ...



41

Deceptive options

- **Hoping for the best** (fatalistic)
- **Going for it** (macho)
- **Working Overtime** (fooling ourself)
- **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

42

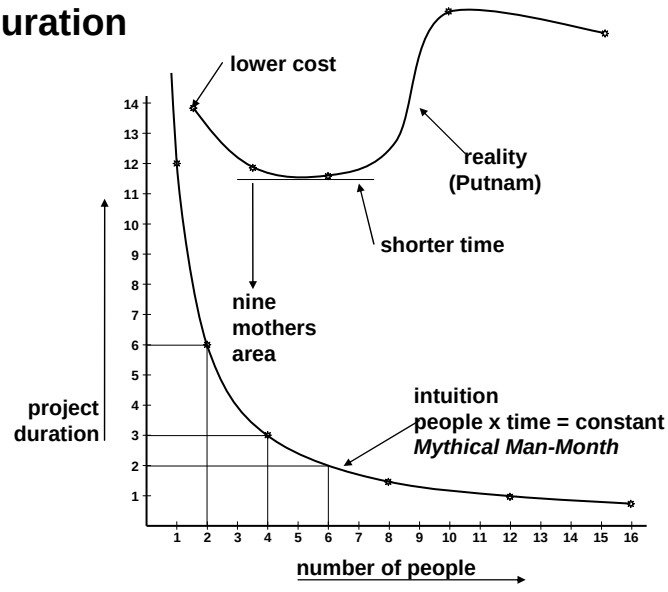
Adding people to a late project ...

makes it later

(Brooks' Law, 1975)

43

Project-duration



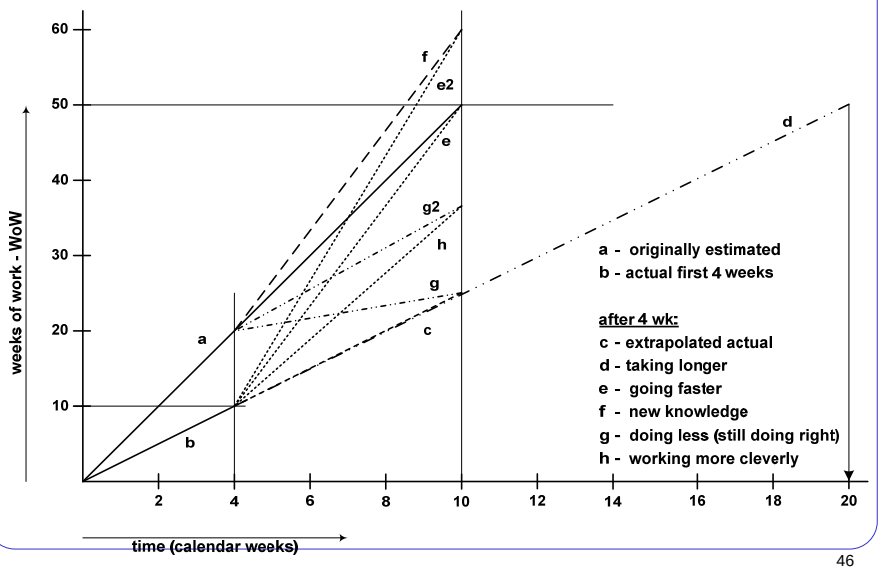
44

Saving time

- We don't have enough time
- We *can* save time, *without negatively affecting the Result !*
- Efficiency improvement in:
  - What (why, for whom): doing only what is needed, *not* doing things that later prove to be superfluous
    - Because people tend to do more than necessary
      especially if they don't exactly know what to do
    - Better 80% 100% done, than 100% 80% done
      let it be the most important 80%, the other part isn't used anyway
    - Don't let things happen; *control* how things happen
    - Is everything we think we have to do really needed?
    - Magic question: "Who is waiting for this?"
  - How: doing things differently
    - Should we do it as we always did it?
    - Can we do it differently?
    - First think, then do: Plan before Do, Design before Implement
    - Using Check and Act to improve
  - When: doing things at the right time, in the right order
- Using TimeBoxing
  - Much more efficient than FeatureBoxing

45

Accepting Fate?



46

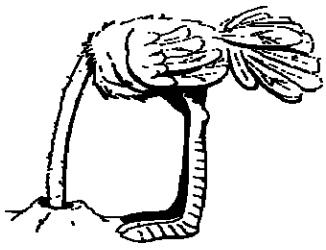
Are we doing all of this?

- **Your current project can be 30% (±10%) faster, with *better* results, almost *immediately***

Because of:

- **Better focus**
- **Better decisions**
- **Better planning**
- **Preventing wasting time**
- **Solving issues before they become problems**
- **Seeing problems earlier**
- **Ignoring "bad" politics**
- **Actively using PDCA all the time, on the Product, the Project *and* the Process**

47



The problems in projects are not the real problem, the real problem is that we don't do something about it

48

Stakeholders and Requirements

- A Stakeholder is anybody with a stake in what we are working on
- Customer, user, ... up to ourselves
- Every project has about 30 Stakeholders
- The set of Stakeholders doesn't change much

- *Requirements* are what the Stakeholders require

but for a project ...

- Requirements are the set of stakeholder needs that a project is planning to satisfy

49

No Stakeholder?

- No Stakeholder: no requirements
- No requirements: nothing to do
- No requirements: nothing to test
- If you find a requirement without a Stakeholder:
  - Either the requirement isn't a requirement
  - Or, you haven't determined the Stakeholder yet
- If you don't know the Stakeholder:
  - Who's going to pay you for your work?
  - How do you know that you are doing the right thing?
  - When are you ready?

50

Requirements with Planguage

ref Tom Gilb

Definition:

RQ27: Maximum Response Time

Scale: Seconds between <asking> for information and <appearance> of it.

Meter: Add a function to the software to measure the maximum response time value and the <range of values> per <working day>.

Benchmarks (Playing Field):

Past: 3 sec (our previous product)

Current: 0.6 sec [competitor y, product x, 2006] ← Marketing Survey Jan 2006

Record: 0.2 sec [competitor x, product y]

Wish: 0.2 sec [2008] ← customer's head of R&D, 19 Feb 2005, <document ...>

Note: Less than 0.2 sec is not noticed by the user, so there is no use in trying to be better than 0.2 sec

Requirements:

Must: 1 sec [99%] ← project-contract

Must: 1.5 sec [100%] ← project-contract

Goal: 0.5 sec ← project-contract

51

Adding performance

	V8.5	V9.0	
• **Usability.Productivity:**			
• Time to set up a typical specified report	65	20	min
• Time to generate a survey	120	0.25	min
• Time to grant access to report, distribute logins to end-users	80	5	min
• **Usability.Intuitiveness:**	265	25.25	min
• Time for medium experienced programmer to find out how to do ...	15	5	min
• **Capacity.RuntimeConcurrency**			
• Max number of concurrent users, click-rate 20 sec, response time < 0.5 sec	250	6000	users

after FIRM / Gilb 2005

52

Design Process

- **Requirements: *What* is needed**
- **Design: *how* we can realize what is needed**
  - **Collect obvious design(s)**
  - **Generate *one* not obvious design**
  - **Compare the relative ROI of the designs**
  - **Select the best compromise**
  - **Describe the selected design**
  - **Check the ROI of the result**
- **Books:**
  - **Ralph L. Keeyney: Value Focused Thinking**
  - **Gerd Gigerenzer: Simple Heuristics That Make Us Smart**

53

DesignLog (project level)

- **In computer, not loose notes, not in e-mails, not handwritten**
  - **Text**
  - **Graphics (drawings)**
  - **On subject order**
  - **Initially free-format**
  - **For all to see**
- **All concepts contemplated**
  - **Requirements**
  - **Assumptions**
  - **Questions**
  - **Available techniques**
  - **Calculations**
  - **Choices + argumentation:**
    - **If rejected: why?**
    - **If chosen: why?**
- **Rejected choices**
- **Final (current) choices**
- **Implementation**

54

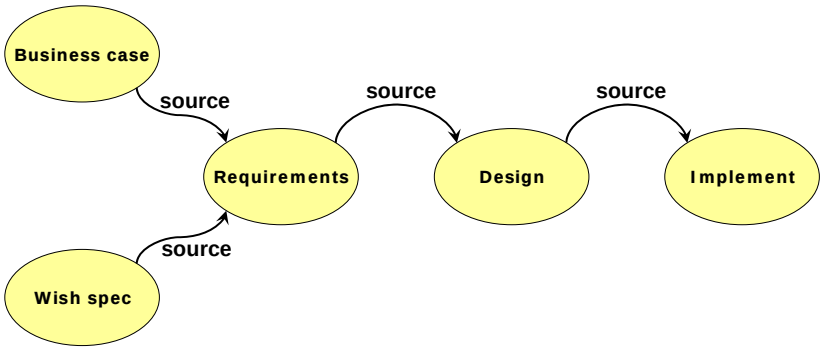
Reviews

- Don't wait until the document is "completely" done
- It probably isn't anyway

- No review without a source

55

Documents and Sources



56

Evo elements

- **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
- **Agile Review & Inspection**
  - Measuring the quality while we are doing, to prevent doing the wrong things

Zero Defects Attitude

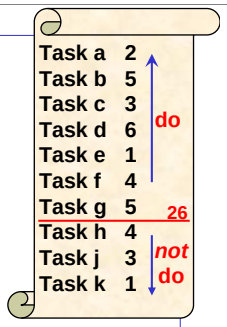
Evo planning

- **Weekly TaskCycle**
  - Short term planning
  - Optimizing estimation
  - Promising what you can achieve
  - Living up to your promises
- **Bi-weekly DeliveryCycle**
  - Optimizing the requirements and checking the assumptions
  - Soliciting feedback by delivering Real Results to appropriate and *eagerly waiting* Stakeholders
- **TimeLine**
  - Getting and keeping control of Time

57

Planning you next week

exercise



- How much time do you have available
- 2/3 of available time is net plannable time
- What is most important to do
- Estimate effort needed to do these things
- Which most important things fit in the net available time (default 26 hr)
- What can, and are you going to do
- What are you *not* going to do

58

Links

- **www.gilb.com**
  Tom Gilb's website: Evo guru
- **www.malotaux.nl**
  Niels' activities: Evo evangelist
- **www.malotaux.nl/nrm/Evo**
  Evo pages
- **www.malotaux.nl/nrm/pdf/MxEvo.pdf**
  Evolutionary Project Management Methods
  (issues and first - 2001 - experience)
- **www.malotaux.nl/nrm/pdf/Booklet2.pdf**
  How Quality is Assured by Evolutionary Methods
  (more recent - 2004 - practical implementation experience)
- **www.malotaux.nl/nrm/pdf/EvoTesting.pdf**
  Optimizing the Contribution of Testing to Project Success (2005)
- **www.malotaux.nl/nrm/pdf/EvoRisk.pdf**
  Controlling Project Risk *by Design* (2006)
- **www.malotaux.nl/nrm/pdf/TimeLine.pdf**
  TimeLine: How to get and keep control over longer periods of time (2007)
- **www.malotaux.nl/nrm/Evo/ETAF.htm**
  Download the Evo Task Administrator (ETA) tool
  (expects MSAccess 2000~2003)

59

Can you afford not to use Evo?

Niels Malotaux

N R Malotaux Consultancy

+31-30-228 88 68 niels@malotaux.nl www.malotaux.nl

60

SPA2008 - 16-19 March 2008

Niels Malotaux

And you thought you could estimate ?

TaskList exercise

Project: .....

Gross hours available this TaskCycle: .....

Net hours available (default 2/3 of gross hours): .....

Fill net ("plannable") hours exactly with the most important things to do, never ever plan less important things.

	Task description	hrs[1]	Delivery[2]	delivery due date	prio[3]	done[4]
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						

[1] hrs: Net number of hours needed to completely conclude this Task. Cut Tasks of more than 6 hrs in smaller pieces.

[2] Delivery: To *whom* are you delivering *what*, and *why*? Write down the name of the Delivery for which you do this task.

[3] prio: Normally, you don't need to prioritize, because at the end of the cycle, all Tasks will be done, completely done. However, if you expect an interrupt during the cycle, you can indicate which Tasks you think may be sacrificed first.

[4] done: Check when this task is 100% done. Only 100% done is done.

SPA2008 - 16-19 March 2008

Niels Malotaux

And you thought you could estimate ?