Predictable Projects

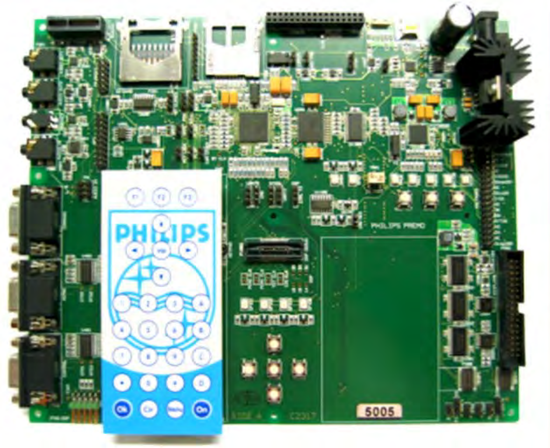
Using Evolutionary Project Management to get the Right Results at the Right Time

Niels Malotaux

N R Malotaux Consultancy

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

Niels Malotaux



- Project Coach
- Helping projects and organizations very quickly to become
  - More effective – doing the right things better
  - More efficient – doing the right things better in less time
  - Predictable – delivering as predicted
- Getting projects back on track

Quality on Time

Requirements with Planguage

ref Tom Gilb

Definition:

RQ27: Speed of Luggage Handling at Airport

Specific

Scale: Time between <arrival of airplane> and first luggage on belt

Measurable

Meter: <measure arrival of airplane>, <measure arrival of first luggage on belt>, calculate difference

Benchmarks (Playing Field):

Past: 2 min [minimum, 2009], 8 min [average, 2009], 83 min [max, 2009]

Current: < 4 min [competitor y, Jan 2010] ← <who said this?>, <Survey Feb2010>

Attainable

Record: 57 sec [competitor x, Jan 2010]

Wish: < 2 min [2011Q3] ← CEO, 19 Feb 2010, <document ...>

Requirements:

Time

Realizable

Must: < 10 min [99%, Q4] ← SLA

Must: < 15 min [100%, Q4, Schiphol] ← SLA

Goal: < 15 min [99%, Q2], < 10 min [99%, Q3], < 5 min [99%, Q4] ← marketing

Ultimate Goal of a Project

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
  - 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

What and How

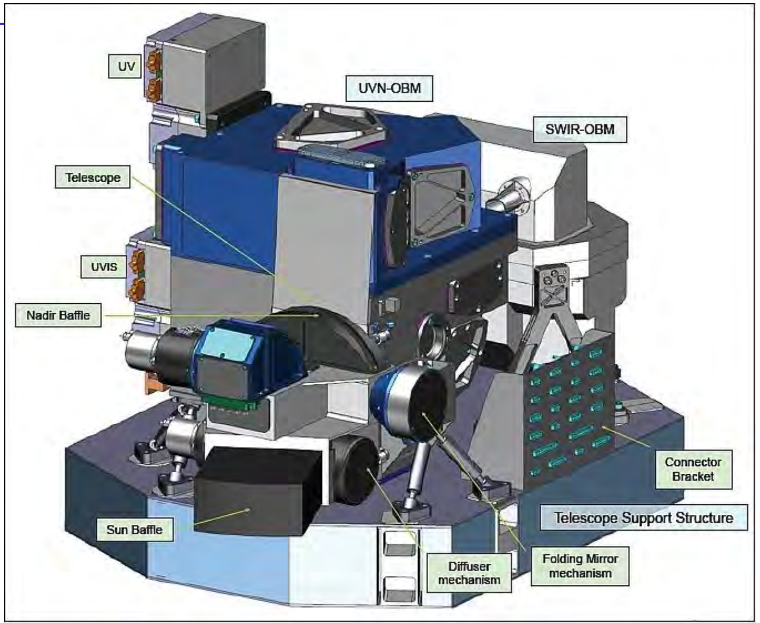
- Carry out an Evo delivery cycle
- Measure values delivered
- Measure costs incurred

retrospectively

- For *whom* ← stakeholders
- *What* to carry out ← requirements, prioritizing
- *How* to carry out
  - Effectively
    producing desired results : having an intended effect
  - Efficiently
    producing desired results *without wasting* (materials, time, energy)
  - Continuous improvement (at no extra burden: it should save !)

prespectively

Earth Observation Satellite

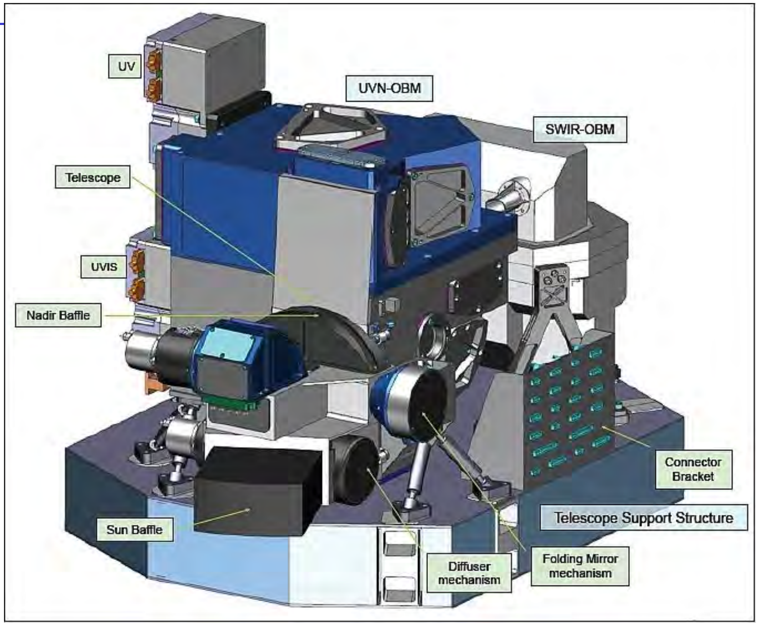


- Very experienced Systems Engineers
- Using quantified requirements routinely
- They don't know exactly where they'll end up
- 10 year project

Summary of requirements for the ozone products:

- Requirements for tropospheric O3
  - Ground-pixel size : 20 × 20 km2 (threshold); 5 × 5 km2 (target)
  - Uncertainty in column : altitude-dependent
  - Coverage : global
  - Frequency of observation :
    daily (threshold); multiple observations per day (target)
- Requirements for stratospheric O3
  - Ground-pixel size : 40 × 40 km2 (threshold); 20 × 20 km2 (target)
  - Uncertainty in column : altitude-dependent
  - Coverage : global
  - Frequency of observation :
    daily (threshold); multiple observations per day (target)
- Requirements for total O3
  - Ground-pixel size : 10 × 10 km2 (threshold); 5 × 5 km2 (target)
  - Uncertainty in column : 2%
  - Coverage : global
  - Frequency of observation :
    daily (threshold); multiple observations per day (target)

Earth Observation Satellite



- Only problem: They missed all deadlines
- Now: They haven't missed any deadline for a year

First Do and then Think, or First Think and then Do ?

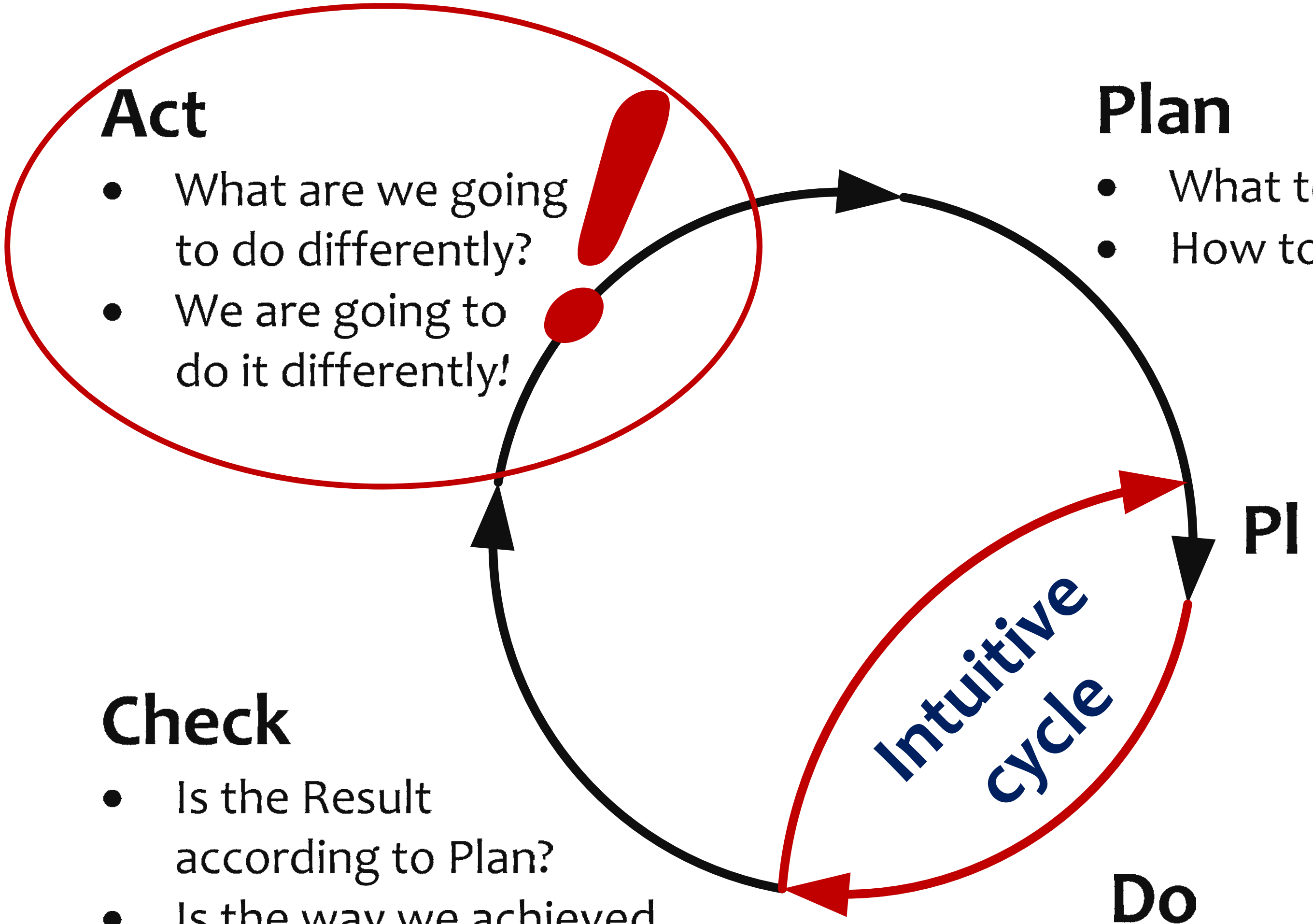
Insanity is doing the same things over and over again and hoping the outcome to be different *(let alone better)*

Albert Einstein 1879-1955, Benjamin Franklin 1706-1790, it seems Franklin was first

- 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 → retrospective
  - Preflection is for foresight and prevention → prespective
- Only with prevention we can save precious time
- This is used in the Deming/Plan-Do-Check-Act cycle

The essential 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

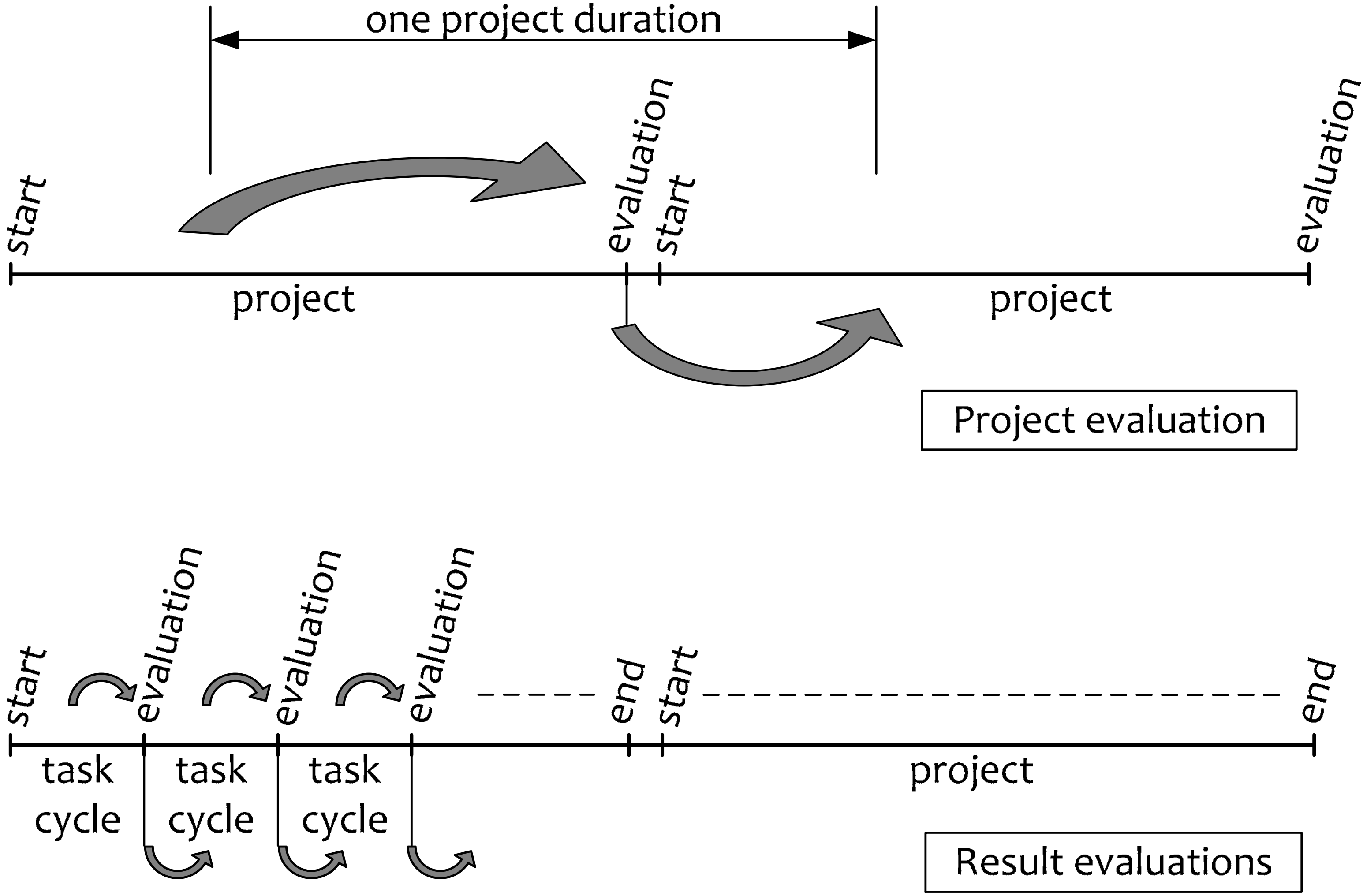
Check

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

Do

Carry out the Plan

Project evaluations - Post Mortem - Retrospectives



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* for *whom*
- 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

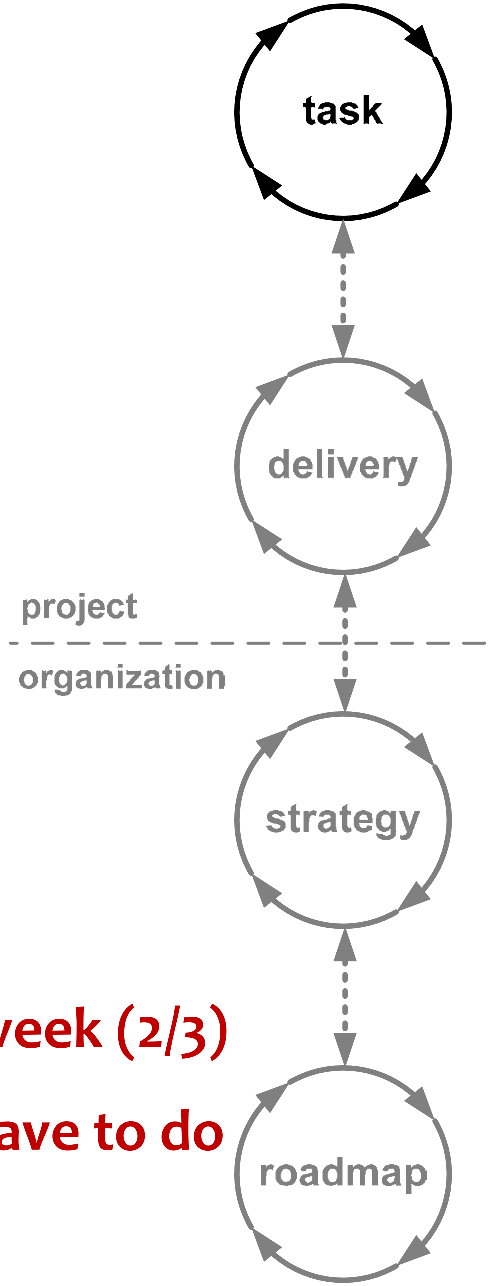
Efficiency of what we do

Effectiveness of what we do

What will happen and what will we do about it ?

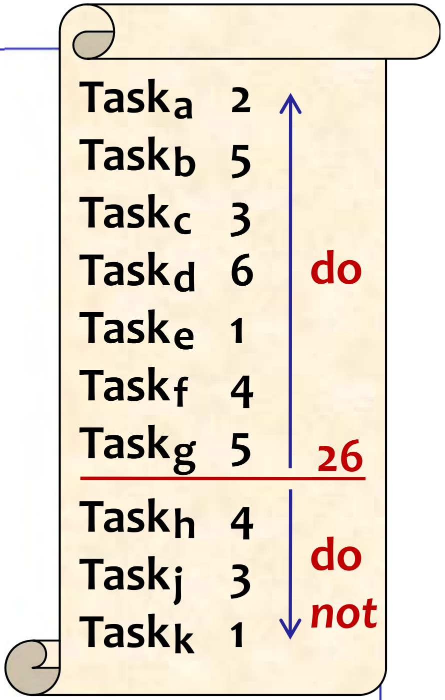
Weekly TaskCycle

- *What* are we going to do and *why*
- 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



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 *exactly* (default 26 hr per week)
- What can, and are we going to do
- What are we *not* going to do
- ***Not producing waste !***



Task$_a$	2	
Task$_b$	5	
Task$_c$	3	
Task$_d$	6	do
Task$_e$	1	
Task$_f$	4	
Task$_g$	5	26
Task$_h$	4	
Task$_j$	3	do
Task$_k$	1	*not*

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

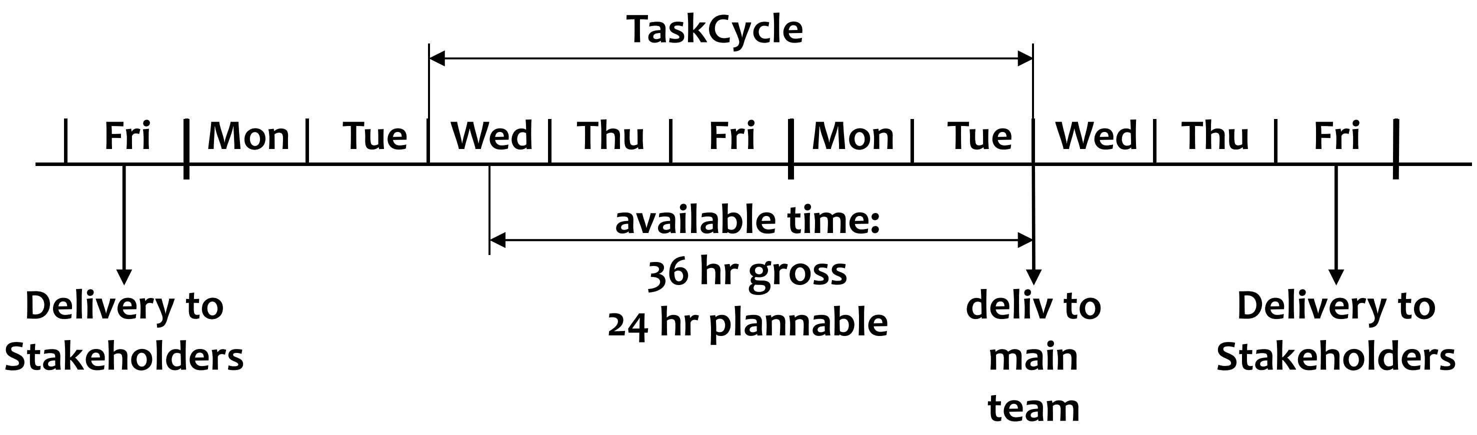
Planning: Retrospection *and* Prespection

- **Retrospection: Analysis of last week**
  - All work done ?
    - If real and estimated time significantly different: analyze to learn
  - Not all work done ?
    - Why ?
      - 3 hr planned, 3 hr spent, task not done: complexity estimation issue
      - 3 hr planned, 3 hr not spent, task not done: time management issue
- **Prespection: Planning of next week**
  - How much time available
  - Most important things to do
  - How much fits in the available time ?
  - How much time is *needed*; *may* we spend; *will* we spend (timebox)
  - What will we have done by the end of the cycle ?
  - What do we now already know that won't have been done ?

Weekly 3-Step Procedure

- Individual preparation
  - Conclude current tasks
  - What to do next
  - Estimations
  - How much time available
- Modulation with / coaching by Project Management
  - Status
  - Priority check
  - Feasibility
  - Commitment and decision
- Synchronization with group (team meeting)
  - Formal confirmation
  - Concurrency
  - Learning
  - Helping
  - Socializing

Designing a Delivery



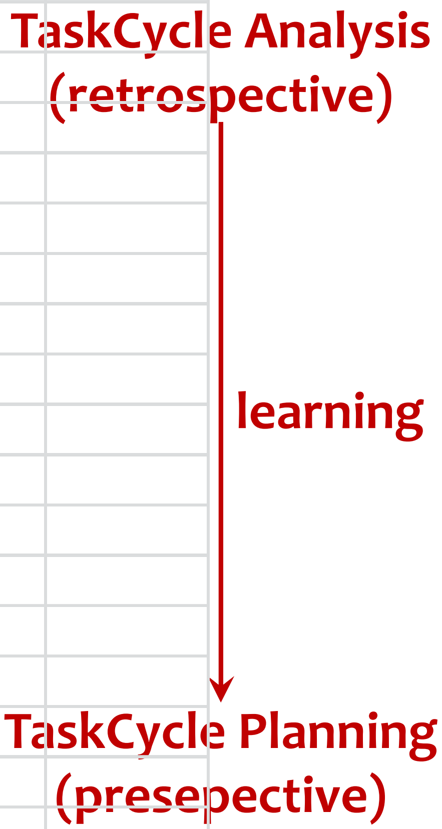
Serge (ProjLead)	
MbWA	3
Planning nxt wk	3
Work for deliv	4
-	6
-	2
-	1
-	5
Total	24

Gregory	
Draft design	0
Finish design	0
Work for deliv	3
-	1
-	2
-	2
-	3
-	5
-	6
XMLa	1
XMLb	1
Total	24

Gregory (later)	
Draft design	0
Finish design	0
…	

Jerome	
XMLa	3
XMLb	3
…	

cycle	who	task description	estim	real	done	issues		
3	John	*Net time available: 26*						
		aaaaaaaaa	3	3	yes			
		bbbbbbbb [Paul]	1					
		cccccccccc	5	13	yes			
		dddddddd	2					
		eeeeeeee	3	2				
		fffffffffff	2	1				
		ggggggggg	6	7	yes			
		hhhhhhhh	4					
			26	26				
4	John	*Net time available: 26*						
		jjjjjjjjjjjjjj	3			for proj x		
		~~kkkkkkkkk~~	1			for proj x		
		mmmmm	5			for proj x		
		~~nnnnnnnn~~	2			for proj x		
		~~pppppppp~~	3			for proj y		
		qqqqqqqq	12			for proj y		
		rrrrrrrrrrrr	6			for proj y		
		~~sssssssss~~	4			for proj y		
		~~ttttttttttt~~	4			for proj y		
			40 26					

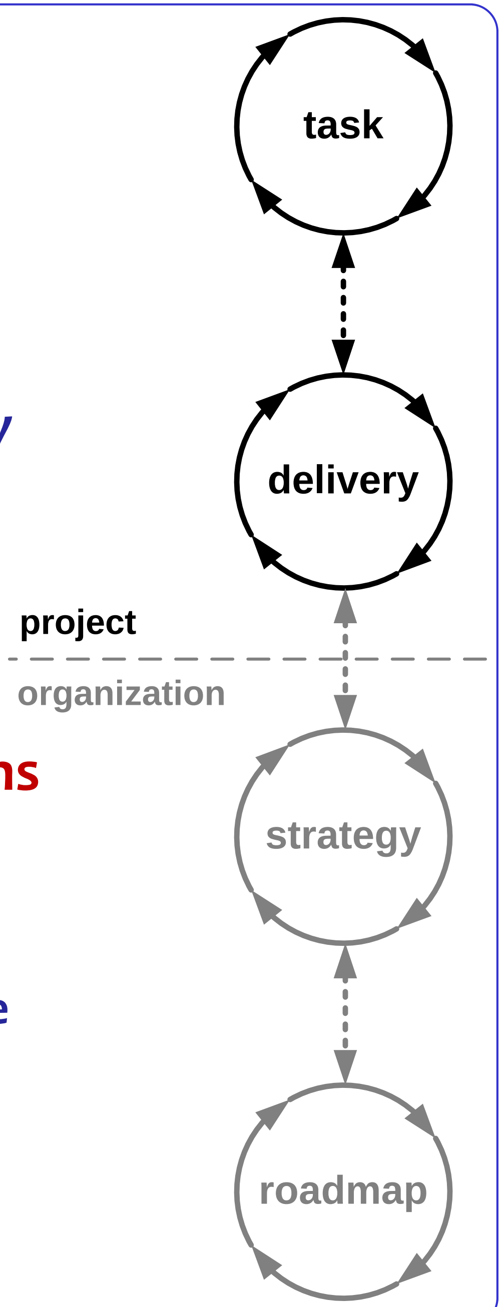


TaskCycle Analysis (retrospective)

learning

TaskCycle Planning (presepective)

DeliveryCycle



- *What* are we going to deliver to *whom* and *why*
- Are we *delivering* the right things,
  in the right order,
  to the right level of detail for now
- Optimizing requirements, checking assumptions
  1. What will generate the optimum feedback
  2. We deliver only to eagerly waiting stakeholders
  3. Delivering the juiciest, most important stakeholder values that can be made in the least time
  - What will make Stakeholders more productive now
- Making sure we understand what real value is
- Not more than 2 weeks

Agile, but will we be on time ?

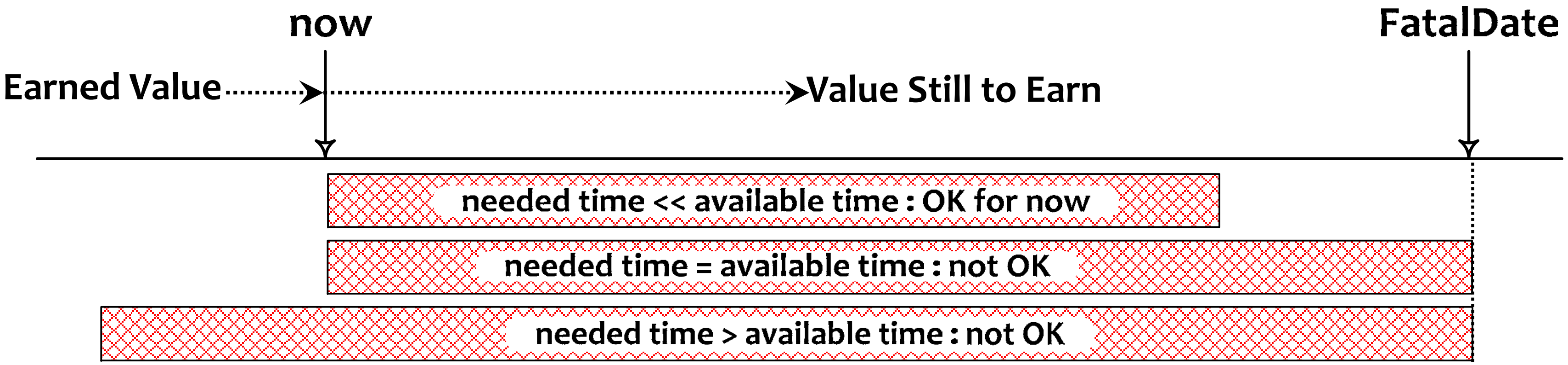
- Organizing the work in very short cycles
- Making sure we are doing the right things
- Doing the right things right
- Continuously optimizing (also what *not* to do)
- So, we already work more efficiently

but ...

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

TimeLine:

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

- **Value Still to Earn**

versus

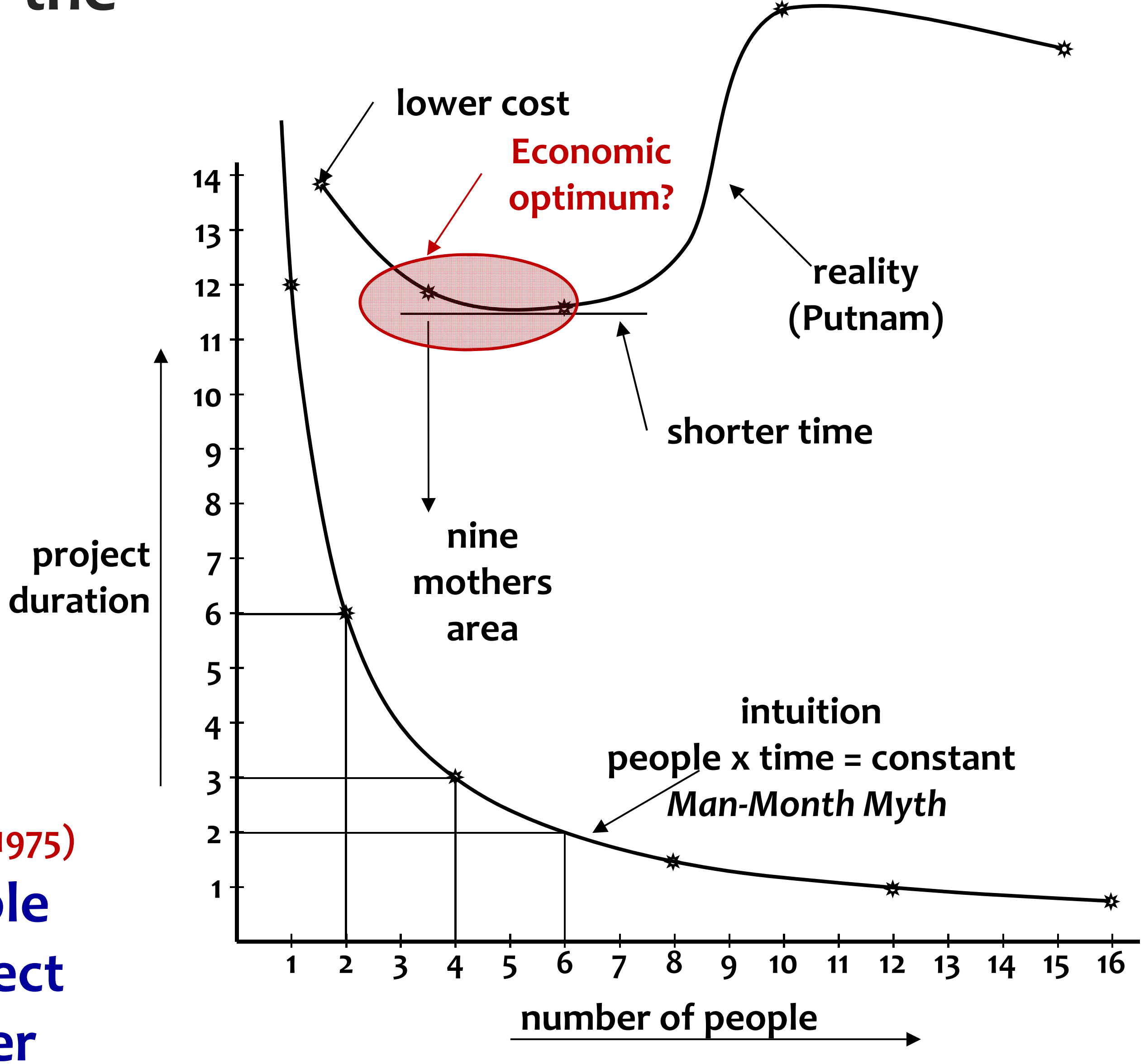
- **Time Still Available**

If the match is over, you cannot score a goal

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



Brooks' Law (1975)

Adding people to a late project makes it later

Saving time

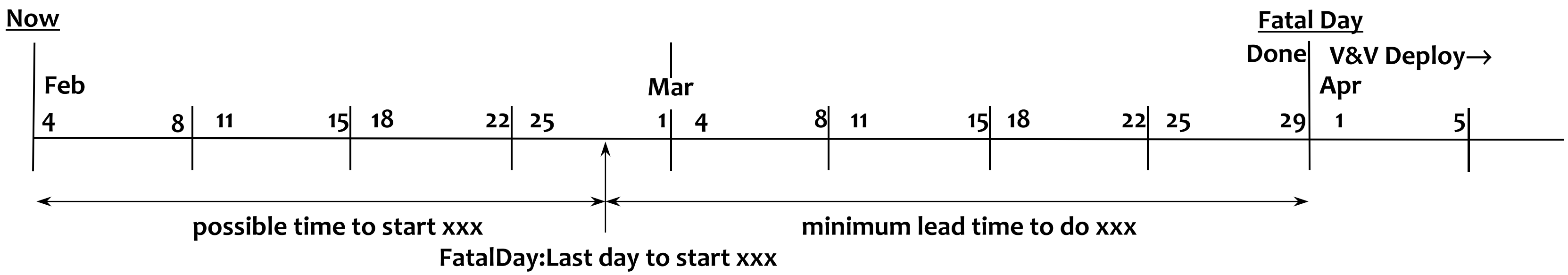
Continuous elimination of waste

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, in the right order**
- **TimeBoxing - much more efficient than FeatureBoxing**

Starting deadlines

- What will we have done when
- Last day of starting xxx not to need an excuse later


Now
Feb
4
8
11
15
18
22
25
Mar
1
4
8
11
15
18
22
25
29
Fatal Day
Done
V&V Deploy→
Apr
1
5
possible time to start xxx
minimum lead time to do xxx
FatalDay:Last day to start xxx

Last day of starting activity xxx not to need an excuse later

All this prespection takes too much time !

- It doesn’t
- It should save time, otherwise: don’t do it !
- It worked in many projects,
  statistically there is a good chance that it works for you

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 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/inspections

Inspection pages

Predictable Projects

Using Evolutionary Project Management to get the Right Results at the Right Time

Niels Malotaux

N R Malotaux
Consultancy

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