



Oxford
23 April 2009

Niels Malotaux

Controlling Project Risk *by Design*

version as presented

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

Controlling Project Risk by Design

Niels Malotaux

Niels Malotaux is an independent Project Coach specializing in optimizing project performance. He has over 35 years experience in designing electronic 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 over 100 projects in 25+ organizations in the Netherlands, Belgium, China, Germany, India, Ireland, Israel, Japan, Romania, South Africa and the US, which led to a wealth of experience in which approaches work better and which work less in the practice of real projects.

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	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
<i>Result Management</i>	

Controlling Project Risk *by Design*

Niels Malotaux

N R Malotaux
Consultancy

+31-30-228 88 68

niels@malotaux.nl

www.malotaux.nl

1

Niels Malotaux

Project Coach

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

Result Management

- Researching problems in projects
- Finding ways to fundamentally overcoming these problems
- Ploughing back into projects
- Tuning of the results (because theory isn't practice)

2

Booklets:

www.malotaux.nl/nrm/pdf/MxEvo.pdf - www.malotaux.nl/nrm/pdf/Booklet2.pdf
www.malotaux.nl/nrm/pdf/EvoQA.pdf - www.malotaux.nl/nrm/pdf/EvoRisk.pdf
www.malotaux.nl/nrm/pdf/TimeLineISo9.pdf - www.malotaux.nl/nrm/pdf/HumanBehavior.pdf

Risk Definition

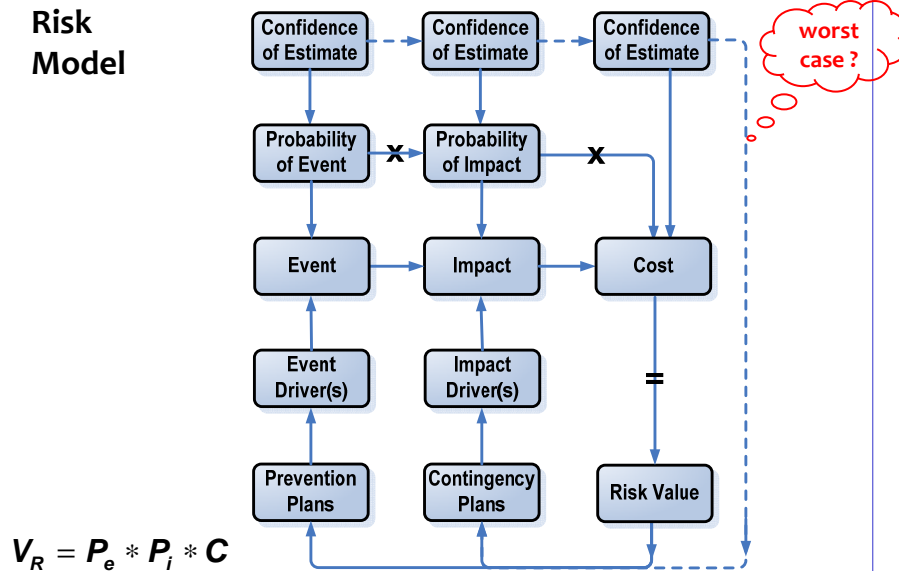
**An uncertain event or condition that,
 if it occurs,
 has a negative effect
 on a project's objectives**

(PMBOK)

- 0% probability is not a risk
- 100% probability is an issue or a problem

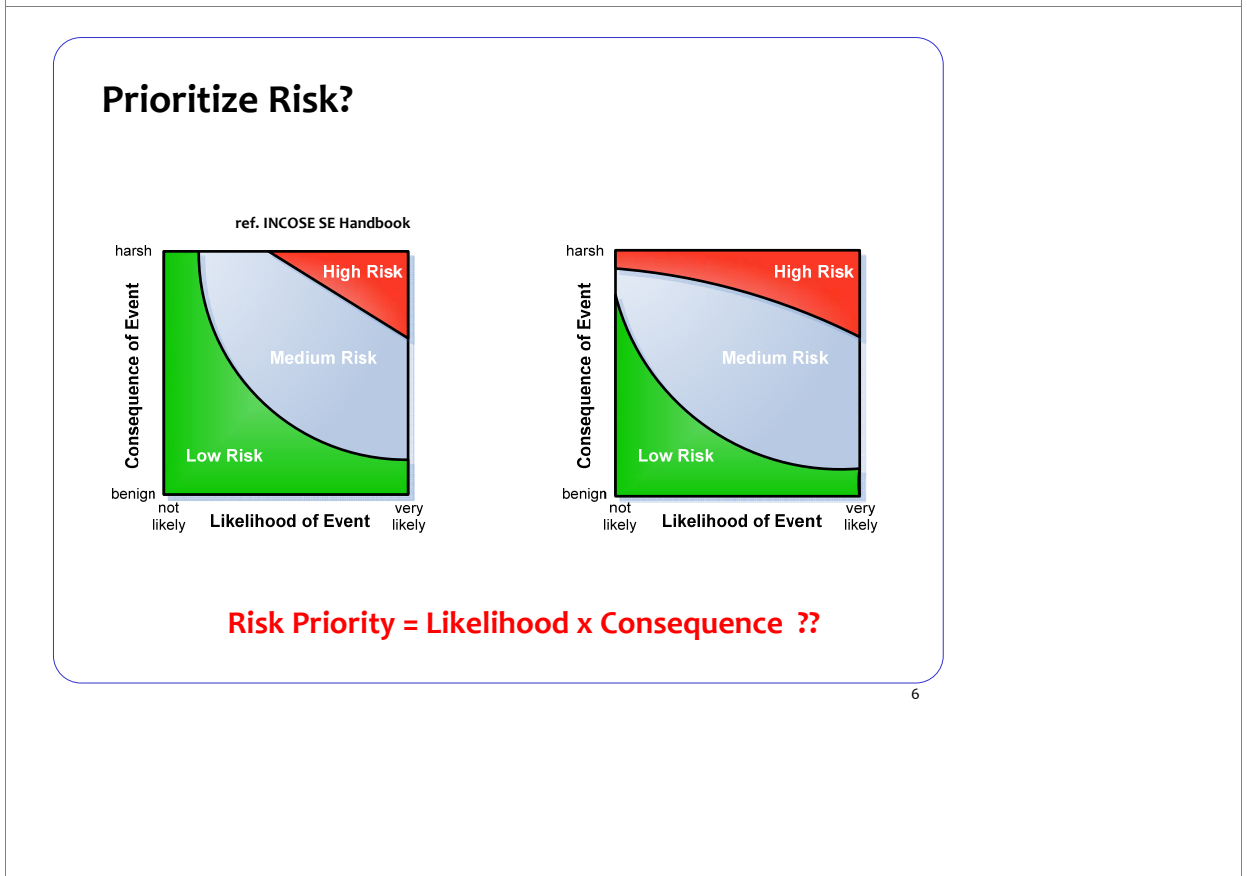
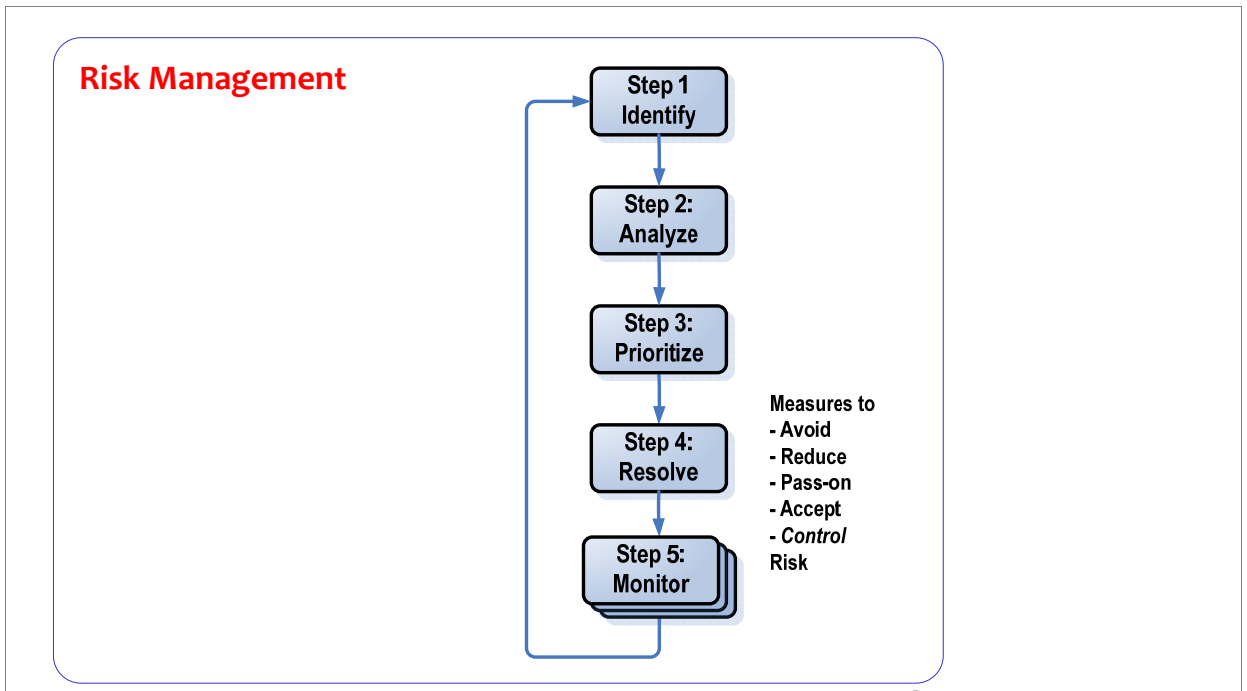
3

Risk Model



4

Controlling Project Risk by Design



Mathematical Risk Management can be risky

		Item 233 (Program constraint)							WB tasks	
		Staff	Budget	Facilities	Type of contract	Restrictions or Dependencies	Customer	Subcontractors	WB	WB number
Item	Develop Project Charter								120	1
	Define scope	I=8, pr=5, R=20	I=7, pr=4, R=20			I=8, pr=5, R=20	I=7, pr=4, R=20		22	3
	Develop Business Plan	I=8, pr=5, R=20				I=8, pr=5, R=20			47	6
	Develop Communications Plan	I=8, pr=5, R=20				I=8, pr=5, R=20			21	3
	Develop Risk Plan	I=8, pr=5, R=20							22	3
	Develop Change Control Plan								5	6
	Develop Quality Plan					I=8, pr=5, R=20			22	20
	Develop Resource Plan								5	22
	Develop Cost Plan		I=8, pr=4, R=20						22	6
	Develop Organization Plan								5	22
	WB1	Develop Project Schedule							5	22
		Conduct Kickoff meeting	I=8, pr=5, R=20			I=8, pr=5, R=6			41	6
		Weekly Status Meeting							5	22
		Monthly Technical Meeting							5	22
		Project closing meeting					I=8, pr=5, R=6		5	44
	Standards	I=8, pr=5, R=20				I=8, pr=5, R=20	I=8, pr=4, R=20	22	2	
	Program Closure					I=8, pr=5, R=20	I=8, pr=4, R=20	22	4	
	WB	216	60	6	22	26	226	26		
	Risk events	1	6	2	6	2	6			

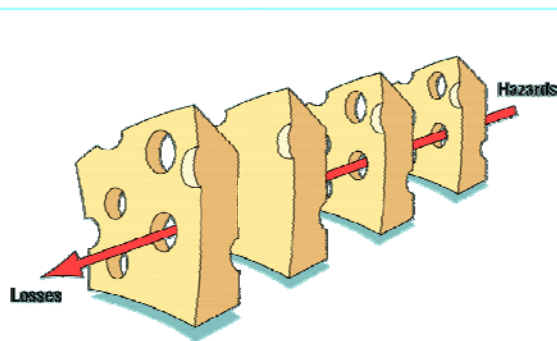
ref
 Carlo Rafele,
 David Hillson,
 Sabrina Grimaldi

Exhibit 3 - Matrix RBM for a software development with a cardinal scale approach

7

Swiss Cheese model

ref James Reason

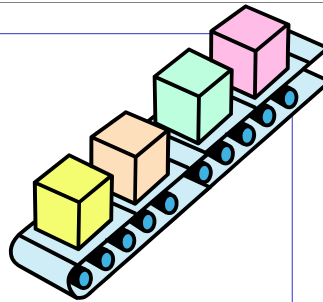


Can we add some cheese from Holland?

8

Controlling Project Risk by Design

Controlling Risk by design



- **Every (software) project is unique**
(otherwise it's production)

however

- **A lot is always the same:**
 - Every project is done by people
 - No project is very much unique
 - There are many similarities (with known risks)
 - So, a lot is predictable
 - We know Requirements will change (don't know which)
 - Engineers control risks by design (= engineering)

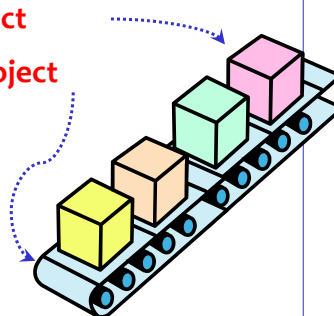
9

Known risks are hardly risks

- **Most of the real risks are in the product**
- **Most of the known risks are in the project**

$$V_{Risk} = P_{event} * P_{impact} * C \quad \begin{array}{l} P_{event} = 1 \\ P_{impact} \rightarrow 0 \end{array}$$

- **We don't only design the product,**
- **We should also design the project**
- **If we control 80% of the risks by design**
- **We have more time to handle the 20% real risks**



10

Universal 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** (win - win)
 - what the customer can afford
 - what we mutually beneficially and satisfactorily can deliver
 - in a reasonable period of time

11

Defect and Risk

If a Defect is:

a cause of a problem experienced by a stakeholder of the system, ultimately by the customer

- Not satisfying the Goal is a defect
- Being late is a defect
- Being over budget is a defect

Then a Risk is:

an event that may cause a defect

12

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**

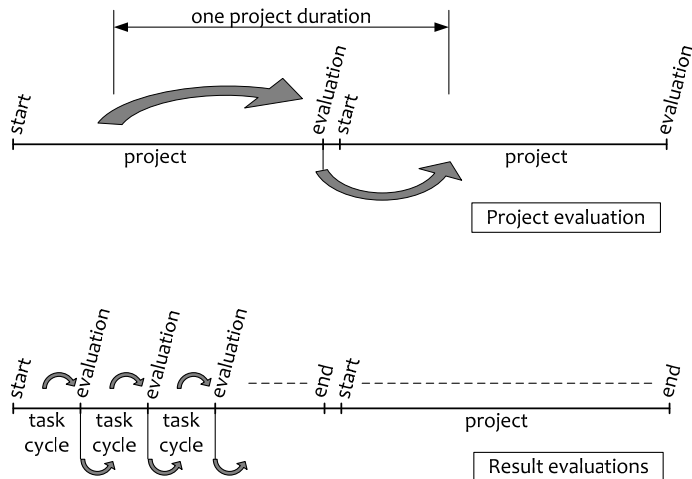
13

**If your previous project was late,
your current project will probably be late**

**If we don't learn from history,
we are doomed to repeat it**

14

Project evaluations



15

Preflection, foresight, prevention

**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**
- **Preflection is for foresight and prevention**

Only with prevention we can save precious time

This is used in the Deming or Plan-Do-Check-Act cycle

16

The essential ingredient: the PDCA cycle

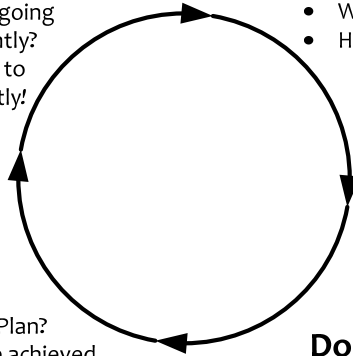
(Deming cycle)

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

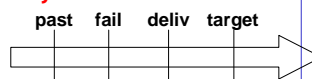
Carry out the Plan

17

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 and highest value**
- **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”
- **Controlling risk by design**
- **Projects seriously applying Evo, routinely conclude successfully on time, or earlier**



18

Booklets:

www.malotaux.nl/nrm/pdf/MxEvo.pdf - www.malotaux.nl/nrm/pdf/Booklet2.pdf
www.malotaux.nl/nrm/pdf/EvoQA.pdf - www.malotaux.nl/nrm/pdf/EvoRisk.pdf
www.malotaux.nl/nrm/pdf/TimeLineISo9.pdf - www.malotaux.nl/nrm/pdf/HumanBehavior.pdf

- **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

Evolutionary Project Management (Evo)

Zero
Defects
Attitude

Right product

- **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

Evo Project Planning

Right time

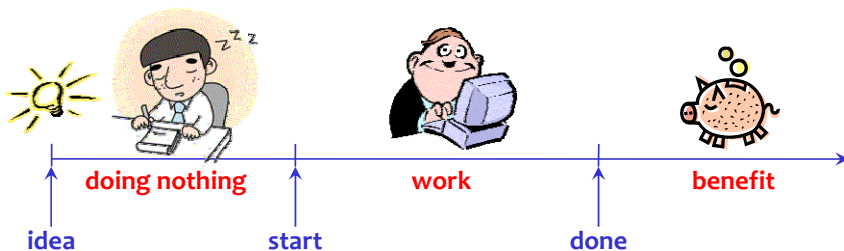
19

Business Case

- **Why to improve**
- **Drives the decision making processes**
- **To continually align the Projects progress to the business objectives**
- **Stakeholders**
- **Expected Return on Investment (ROI)**
 - + Benefit of doing
 - Cost of doing
 - Cost of doing nothing
 - Cost of being late
- **Total LifeCycle**

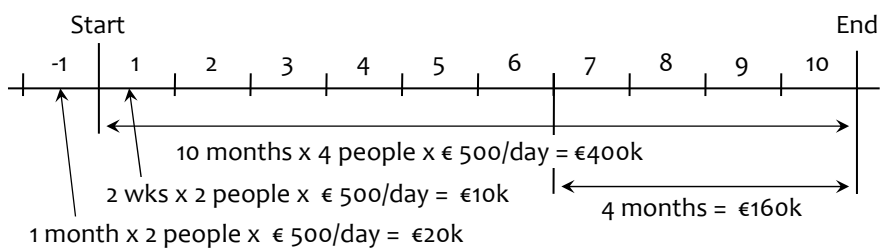
20

What's the Cost of Doing Nothing ?



21

The Cost of Time



- We can save 4 months by investing €200k → "That's too much !"
 - It's a nicer solution - Let's do 2 weeks more research on the benefits
 - What are the expected revenues when all is done? → €16M/yr (1.3M/mnd)
 - So 2 weeks extra doesn't cost €10k, but rather €16M/24 = €670k
 - And saving 4 months brings €16M/3 = €5M extra
- Invest that €200k NOW and don't waste time !

22

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 (± 20) 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**

23

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?**

24

Basic Types of Requirements

- **Functional** *binary*
 - Determine the scope of the project:
 - What are we working on
- **Quality / performance*** *scalar*
 - To enhance the performance of the selected functions
- **Constraints** *binary / scalar*
 - What should we not do, be aware of, be limited by



* Better not use '*non-functional*' requirements !

25

Design is always a compromise

The Requirements are always conflicting

example:

- **Performance** 
- **Budget (time, money)** 

26

Design Process

- Collect obvious design(s)
- Generate one not obvious design
- Compare the relative ROI of the designs
- Select the best compromise
- Describe the selected design

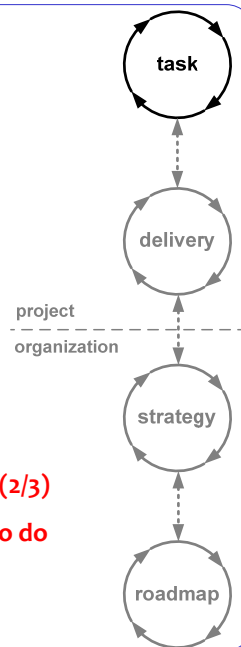
- How much time do you spend designing?

- Books:
 - Ralph L. Keeney: Value Focused Thinking
 - Gerd Gigerenzer: Simple Heuristics That Make Us Smart

27

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



28

Controlling Project Risk by Design

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 per week)
- 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

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

29

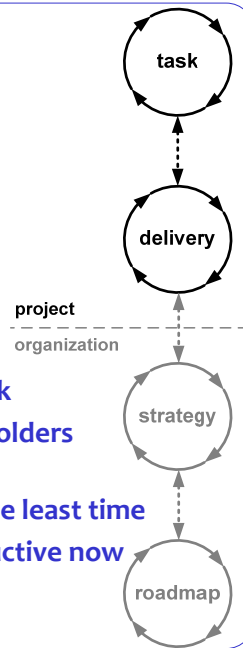
Weekly 3-Step Procedure

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

30

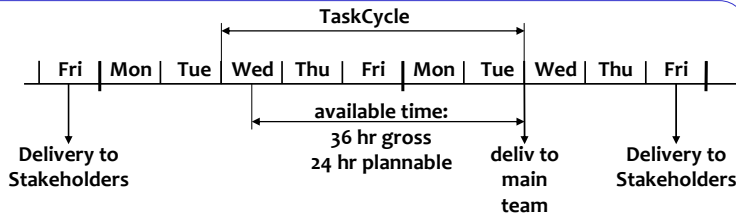
DeliveryCycle

- **Are we delivering the right things, in the right order to the right level of detail for now**
- **Optimizing requirements and checking assumptions**
 - a. **What will generate the optimum feedback**
 - b. **We deliver only to eagerly waiting stakeholders**
 - c. **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**



31

Designing a Delivery



Serge (ProjLead)		Gregory		Gregory (later)	
MbWA	3	Draft design	0	Draft design	0
Planning nxt wk	3	Finish design	0	Finish design	0
Work for deliv	4	Work for deliv	3	...	
-	6	-	1		
-	2	-	2	Repair deliv	
-	1	-	2	...	
-	5	-	3		
Total	24	-	5		
		-	6		
		XMLa	1	XMLa	3
		XMLb	1	XMLb	3
		Total	24	...	

Zero Defects Attitude

32

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
- So, we already work more efficiently

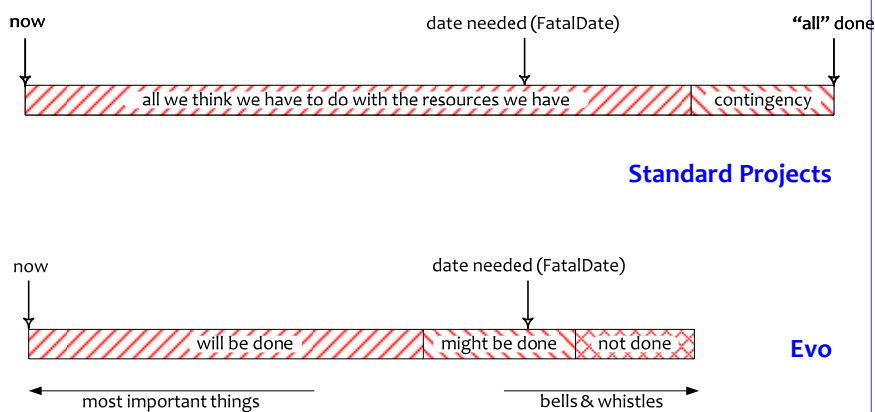
but ...

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

33

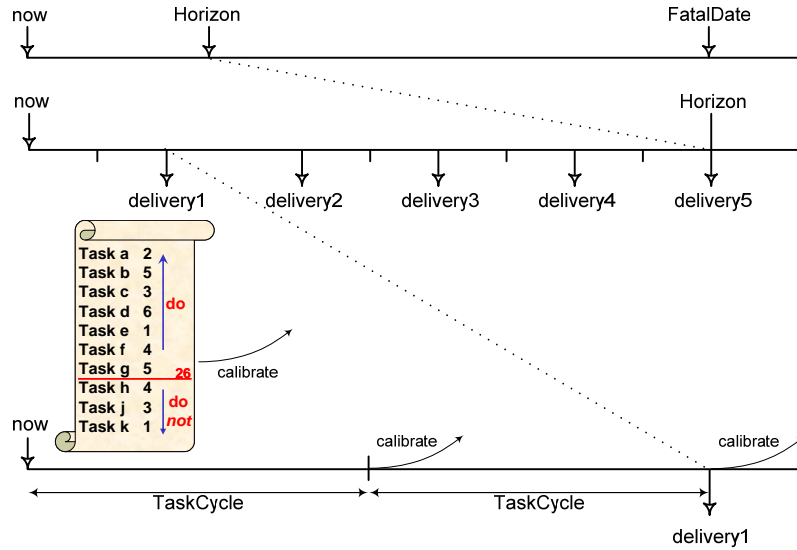
TimeLine

What the customer wants, he cannot afford



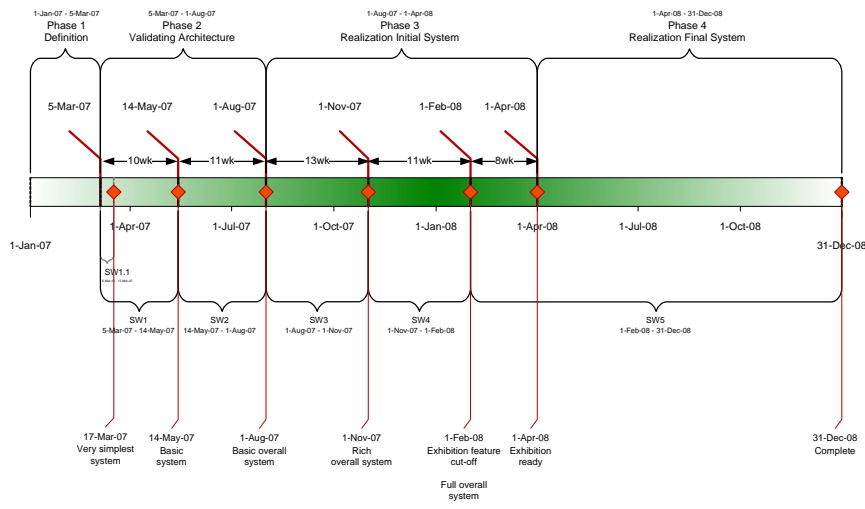
34

Result to Tasks and back



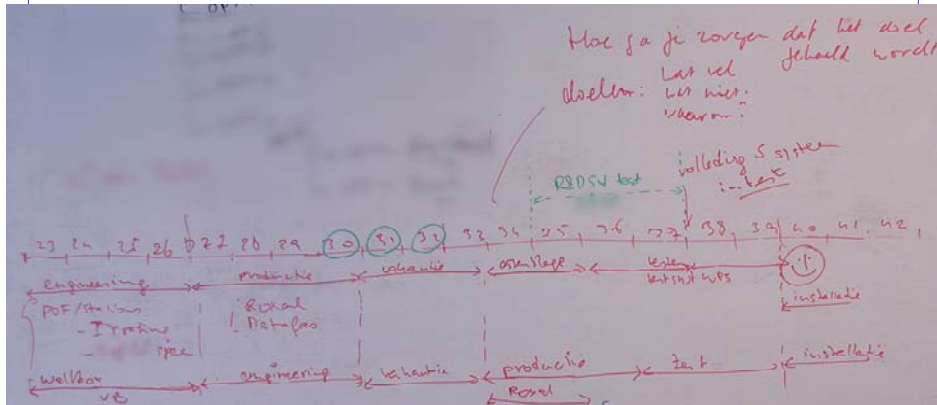
35

TimeLine example



36

Whiteboard TimeLine Planning



37

Activity	Estimate	Real
Act1	Ae1	Ar1
Act2	Ae2	Ar2
Act3	Ae3	Ar3
Act4	Ae4	Ar4
Act5	Ae5	Ar5
Act6	Ae6	Ar6
Act7	Ae7	Ar7
Act8	Ae8	Ar8
Act9	Ae9	Ar9
Act10	Ae10	Ar10
Act11	Ae11	
Act12	Ae12	
Act13	Ae13	
Act14	Ae14	
Act15	Ae15	
Act16	Ae16	
Act17	Ae17	
Act18	Ae18	
Act19	Ae19	
Act20	Ae20	
Act21	Ae21	
...	...	
Act...	Ae...	

Calibration

Calibration Factor

$$\frac{\sum_{now-1}^{now-n} Ar}{\sum_{now-1}^{now-n} Ae} = \frac{1}{n} \sum_{now-1}^{now-n} \frac{Ar}{Ae}$$

Value Still To Earn

Calibration Factor * $\sum_{now}^{then} Ae$

ratio $\Sigma Ar / \Sigma Ae$ in the past

now

predicted Value Still To Earn in the future

then

then2

38

Booklets:

- www.malotaux.nl/nrm/pdf/MxEvo.pdf - www.malotaux.nl/nrm/pdf/Booklet2.pdf
- www.malotaux.nl/nrm/pdf/EvoQA.pdf - www.malotaux.nl/nrm/pdf/EvoRisk.pdf
- www.malotaux.nl/nrm/pdf/TimeLineISo9.pdf - www.malotaux.nl/nrm/pdf/HumanBehavior.pdf

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	3.5	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

39

Project Risks

wrong	wrong
wrong	right
right	wrong
right	right

- Delivering the wrong things at the wrong time (being late!)
- Doing the wrong things for too long
- Promising more than we can do
- Not living up to our promises due to unrealistic optimism
- Not being able to predict what we can have done when
- Making too many mistakes because of fatigue or sloppiness
- Others causing us to fail
- Interrupts

40

Project Risks

(*Boehm, 1991)

- **Personnel shortfalls***
- **Unrealistic schedules and budgets***
- **Management impeding the workers**
- **Developing the wrong functions or properties***
- **Developing the wrong interface***
- **Developing user interfaces ignoring real human behaviour**
- **Gold plating, doing more than necessary***
- **Changing requirements***
- **Real-time performance shortfalls*** (no issue with real engineers)

41

Personnel Shortfalls

Boehm 1991

- **There are a certain number of people in the organization**
- **If we don't get the people we think we need, they are working on more profitable activities**
- **Using TimeLine, we inform management about the consequences**
- **This is not risk - it's choice**

42

Controlling Project Risk by Design

Unrealistic schedules and budgets

Boehm 1991

- How can we speak about realistic schedules if the requirements will change anyway?
- If the time/cost budgets are insufficient to get a profit, we shouldn't start or continue
- If management insist on unrealistic schedules (*Check*), they may need education (*Act*), or their aim is to fail
- People can quickly learn to change from optimistic to realistic estimators and thus live up to their promises
- We continuously update the TimeLine to predict what we will surely achieve, what not and what we may achieve
 - Using "Earned Value" to calibrate the "Value Still to Earn"

43

Continuing stream of Requirements changes

Boehm 1991

- Requirements do change because
 - We learn
 - They learn
 - The circumstances change
- If we would deliver according to obsoleted requirements, we don't create customer success
- We know that requirements will change, so we have to find out quickly which will change:
- Provoking requirements change as quickly as possible

44

Real time performance shortfalls

Boehm 1991

- This is why we have Performance Requirements
- We use engineering practices to make sure the system *will* work according to the requirements
- Real time predictability is easy, if you know how to do it
- If you don't know how to do it, it's not a Risk, it's an Issue (if it can go wrong, it will - Murphy)

45

Managers ignorance

- The product has to generate income
- If management impede the workers to produce the product in the most optimal way ...
- Management usually is not stupid
- But if we don't supply them the right facts ...

- The boss may mess up the Result, if he's the owner of the company
- All the others have the option to leave

46

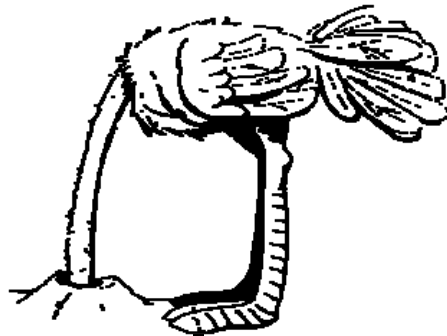
Worst risk

The worst risk is that we forgot an important issue

- It's within our control, but we didn't see it before it happened
- It's beyond our control, but we saw it too late and/or we didn't react appropriately
- The trick is to be ahead of any problem, before it occurs
- Don't ostrich: actively take your head out of the sand!
- If anybody complains, we're too late

If we control 80% of the risks *by design*, we have a lot more time to address the remaining 20%

47



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

48

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/Insp
Inspection pages
- www.malotaux.nl/Booklets
 - 1 Evolutionary Project Management Methods (2001)
 - 2 How Quality is Assured by Evolutionary Methods (2004)
 - 3 Optimizing the Contribution of Testing to Project Success (2005)
 - 3a Optimizing Quality Assurance for Better Results (2005)
 - 4 Controlling Project Risk by Design (2006)
 - 5 TimeLine: How to Get and Keep Control over Longer Periods of Time (2007)
 - 6 Human Behavior in Projects (2008)
 - 7 How to Achieve the Most Important Requirement (2009)
- www.malotaux.nl/nrm/Evo/ETAF.htm
Download the Evo Task Administrator (ETA) tool
(expects MSAccess 2000-2003)

49

Controlling Project Risk *by Design*

Niels Malotaux

N R Malotaux
Consultancy

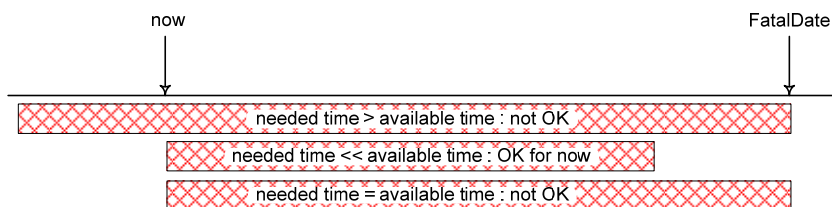
+31-30-228 88 68

niels@malotaux.nl

www.malotaux.nl

50

If it doesn't fit ... count backwards



51

Deceptive options

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

52

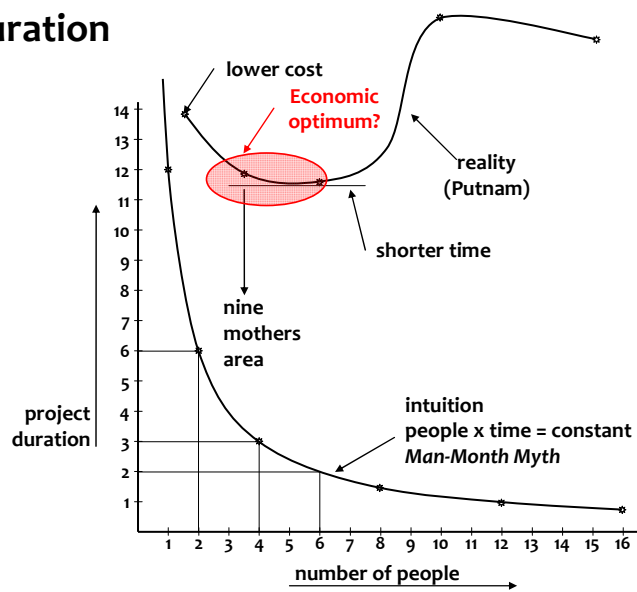
Adding people to a late project ...

makes it later

(Brooks' Law, 1975)

53

Project-duration



54

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, in stead of the solution we always used
 - **The project**
 - Doing the same in less time in stead 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** - doing things in the right order, at the right time
- **TimeBoxing** - much more efficient than FeatureBoxing

55

56

Effort and Lead Time

- Days estimation → lead time (calendar time)
- Hours estimation → effort

- Effort variations and lead time variations have different causes
- Treat them differently and keep them separate
 - Effort: complexity
 - Lead Time: time-management
 - (effort / lead-time ratio)

57

ID	Project	Delivery	Cycle	Task cycle due date	Pri	Wfw	lrs	Done	TaskName
59	Dino-QUA	Delivery 4	Fut		0	-			Hoe gaan we exporteren doen
58	Dino-QUA	Delivery 4	Fut		0	-			Hoe gaan we importeren doen?
212	Dino-QUA	Delivery 7	13	11 jun 2003 wk 24	5	Niko	18		Documentatie SPS, SCM-BDB
220	Dino-QUA	Delivery 6	13	11 jun 2003 wk 24	5	Ronald	6		Samplers importeren
211	Dino-QUA	Delivery 7	13	11 jun 2003 wk 24	5	Niko	4		Conversie aanpassen n.a.v. Hans van der Meij
214	Dino-QUA	Delivery 6	13	11 jun 2003 wk 24	4	Anian	10		Export blokken maken
215	Dino-QUA	Delivery 6	13	11 jun 2003 wk 24	5	Anian	2		Checkbox toevoegen voor export-blokken
216	Dino-QUA	Delivery 6	13	11 jun 2003 wk 24	5	Anian	2		Backsupport toevoegen met Ronald
217	Dino-QUA	Delivery 6	13	11 jun 2003 wk 24	5	Ronald	2		Backsupport toevoegen met Anian
218	Dino-QUA	Delivery 6	13	11 jun 2003 wk 24	5	Anian	6		Uitzoeken rechts invullen van kolommen bij sample, subsample
219	Dino-QUA	Delivery 6	13	11 jun 2003 wk 24	5	Ronald	6		Maken Process dialog
210	Dino-QUA	Delivery 7	13	11 jun 2003 wk 24	5	Niko	2		Conversie aanpassen voor Omrekenfactor koppeling
200	Dino-QUA	Delivery 4	12	4 jun 2003 wk 23	5	Niko	4	OK	parameterformulier voor analyse rapport met tabbladen
201	Dino-QUA	Delivery 4	12	4 jun 2003 wk 23	5	Anian	3	OK	Aanpassingen Monsterverscherm doorvoeren (nieuwe velden)

Booklets:

- www.malotaux.nl/nrm/pdf/MxEvo.pdf - www.malotaux.nl/nrm/pdf/Booklet2.pdf
- www.malotaux.nl/nrm/pdf/EvoQA.pdf - www.malotaux.nl/nrm/pdf/EvoRisk.pdf
- www.malotaux.nl/nrm/pdf/TimeLineISo9.pdf - www.malotaux.nl/nrm/pdf/HumanBehavior.pdf

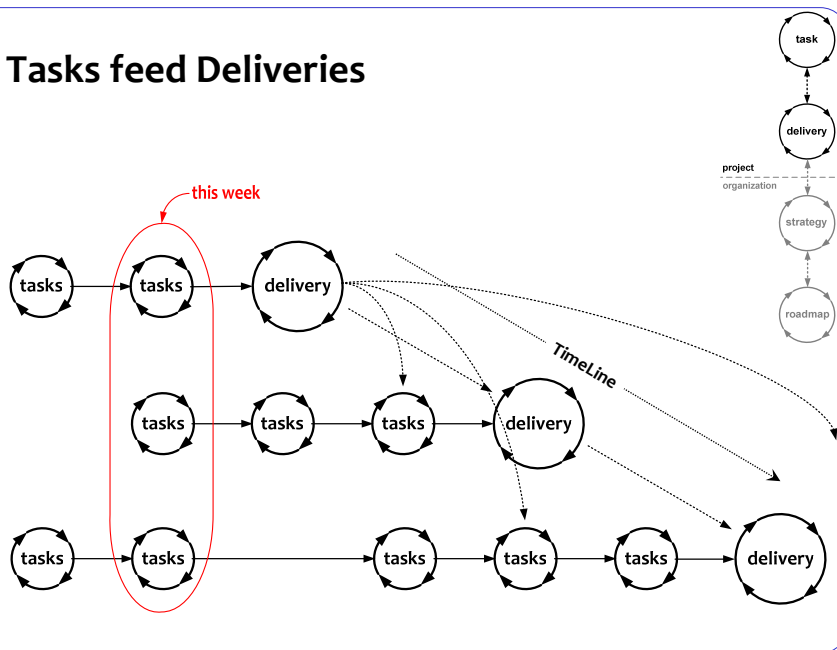
TaskCycle Exercise

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

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	not
Task k	1	do

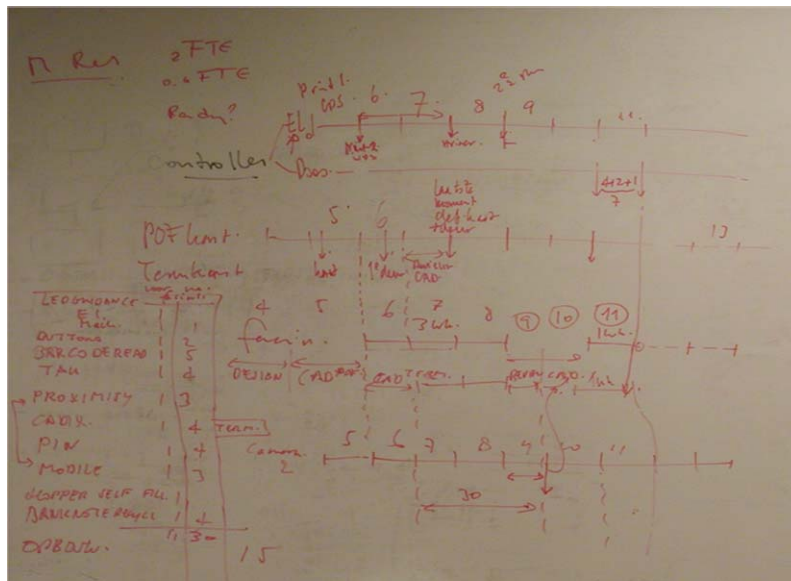
59

Tasks feed Deliveries



60

Whiteboard TimeLine Planning



61

Interrupts

- Boss comes in: "Can you paint my fence?"
- What do you do?

- In case of interrupt, use interrupt procedure

62

Controlling Project Risk by Design

Interrupt Procedure "We shall work only on planned Tasks"

In case a new task suddenly appears in the middle of a Task Cycle (we call this an *Interrupt*) we follow this procedure:

1. Define the expected Results of the new Task properly
2. Estimate the time needed to perform the new Task, to the level of detail really needed
3. Go to your task planning tool (many projects use the ETA tool)
4. Decide which of the planned Tasks is/are going to be sacrificed (up to the number of hours needed for the new Task)
5. Weigh the priorities of the new Task against the Task(s) to be sacrificed
6. Decide which is more important
7. If the new Task is more important: replan accordingly
8. If the new Task is *not* more important, then do not replan and do not work on the new Task. Of course the new Task may be added to the Candidate Task List
9. Now we are still working on planned Tasks.

63

Active Synchronization

Somewhere around you, there is the bad world.

If you are waiting for a result outside your control, there are three possible cases:

1. You are sure they'll deliver Quality On Time
2. You are not sure
3. You are sure they'll not deliver Quality On Time
 - If you are not sure (case 2), better assume case 3
 - From other Evo projects you should expect case 1
 - Evo suppliers behave like case 1

In cases 2 and 3: **Actively Synchronize: Go there !**

1. Showing up increases your priority
2. You can resolve issues which otherwise would delay delivery
3. If they are really late, you'll know much earlier

64

Is Human Behavior a risk?

- **Human behavior is a risk for the success of the system**
 - When human behavior is incorrectly modeled in the system
 - Not because human users are wrong
- **Things that can go wrong**
 - Customers not knowing well to describe what they really need
 - Users not understanding how to use or operate the system
 - Users using the system in unexpected ways
 - Incorrect modeling of human transfer functions within the system: ignorance of designers of systems engineers
- **Actually, the humans aren't acting unpredictably**
 - Because it happens again and again

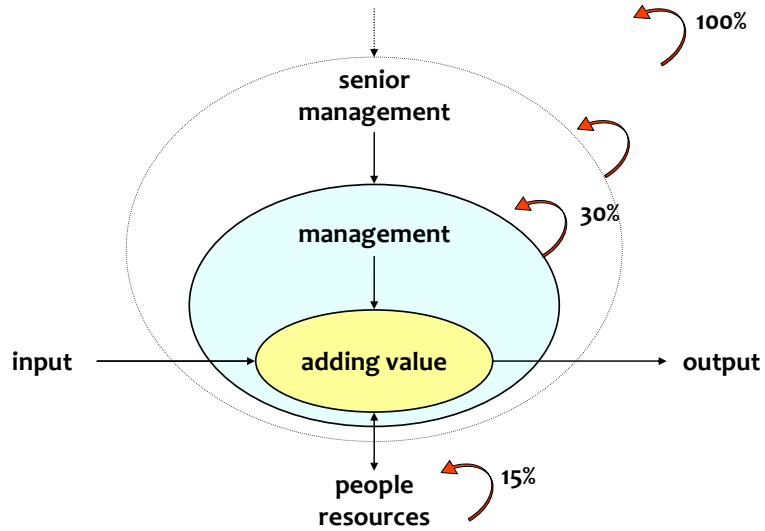
65

Human behavior

- **In the project**
 - Discipline
 - Intuition
 - Communication
 - Perception
- **In the system**
 - Users are *part of the system*
 - So we should model the user interaction correctly

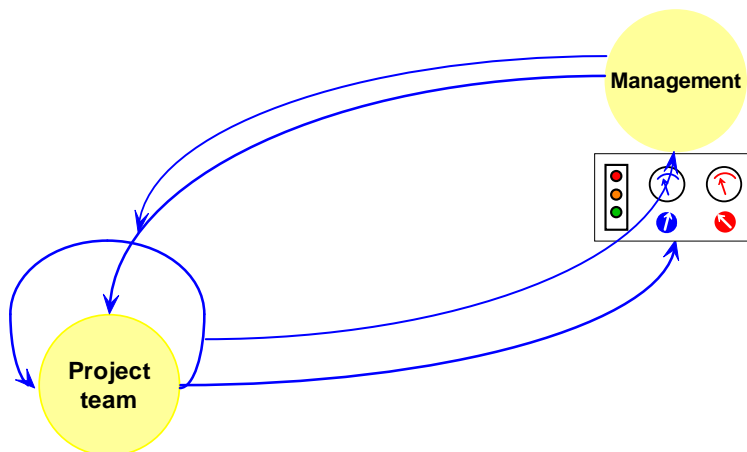
66

The managers task



67

Local loop principle



68

Product Risks



- **Development**
 - Requirements errors
 - Incorrect Assumptions
 - Design errors
 - Calculation errors
 - Implementation errors
- **Maintenance**
 - Incorrect or insufficient maintenance
- **Use**
 - Operator errors
 - User errors
 - Victims

All these risks are introduced by humans

69

Controlling Project Risk *by Design*

Niels Malotaux

N R Malotaux
Consultancy

+31-30-228 88 68

niels@malotaux.nl

www.malotaux.nl

70

ACCU2009 Conference - April 2009 - Oxford
Niels Malotaux
Controlling Project Risk by Design