# About SE - Training

Systems Engineering and Project Management are core engineering disciplines used to enable the delivery of complex projects within schedule and cost expectations.

Delivering complex projects demands cross-functional engineering disciplines such as Systems Engineering, Project Management, Safety Engineering, Product Development and Design Thinking.

SE-Training has been founded to offer specifically tailored solutions that support the drive, ambition and success in providing innovate and high quality products and services.

There are a high number of engineering organisations based across Europe with diverse needs; SE-Training addresses these unique needs through expert project coaching, process development, enterprise organisational design & training courses provided by expert engineering professionals and academics.

# Presenter

## Niels Malotaux

Project and Organizational 'Quality on Time' Coach

Helping projects and organizations to quickly become

- More effective - doing the right things better
- More efficient - doing the right things better in less time
- Predictable - delivering as needed

Getting projects back on track (project rescue)

Embedded Systems architect (electronics/firmware)

Project Types: Electronic Products, Firmware, Software, Space, Railway, Telecom, Industrial Control, Parking System

**Result Management**

# The Evolutionary Approach for delivering Quality on Time

## The Right Results at the Right Time

**Niels Malotaux**

ChatGPT:

Maximizing Quality and Efficiency: Achieving On-Time Delivery



# Quality on Time

The ultimate goal of what we do for our salary

- Delivering the **Right Results at the Right Time**, wasting as little time as possible (=efficiently)
- Providing the customer with:
  - What they need
  - At the time they need it
  - To be satisfied
  - To be more successful than they were without it
- Constrained by:
  - What the customer can afford
  - What we mutually beneficially and satisfactorily can deliver
  - In a reasonable period of time



# Quality on Time

- Do your projects deliver
  the Right Results at the Right Time?

- Right Results?
  - How do you know?
- Right Time?
  - Really?

- Any incentive to improve ?

# How to be on time ?

## Deceptive options

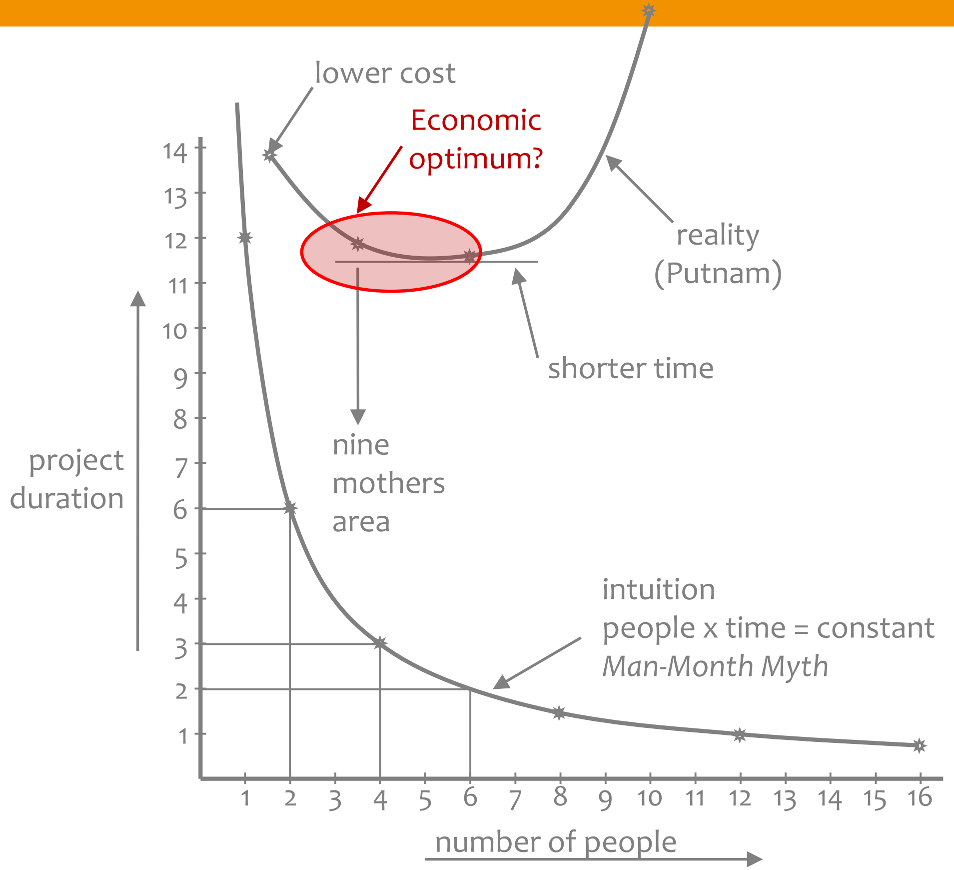
- Hoping for the best (fatalistic)
- Going for it (macho)
- Working overtime (fooling ourselves)

- 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

Intuition often guides us into the wrong direction

# How to be on time ?

## Adding people




Brooks' Law (1975)

Adding people to a late project makes it later

# Saving Time

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 not needed

## Efficiency in how we do it

- Doing things differently
- The Product

  Using the 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 to do things better, and overcoming bad tendencies

## Efficiency in when we do it

- At the right time
- In the right order

## Time Boxing

- Much more efficient than Feature Boxing!

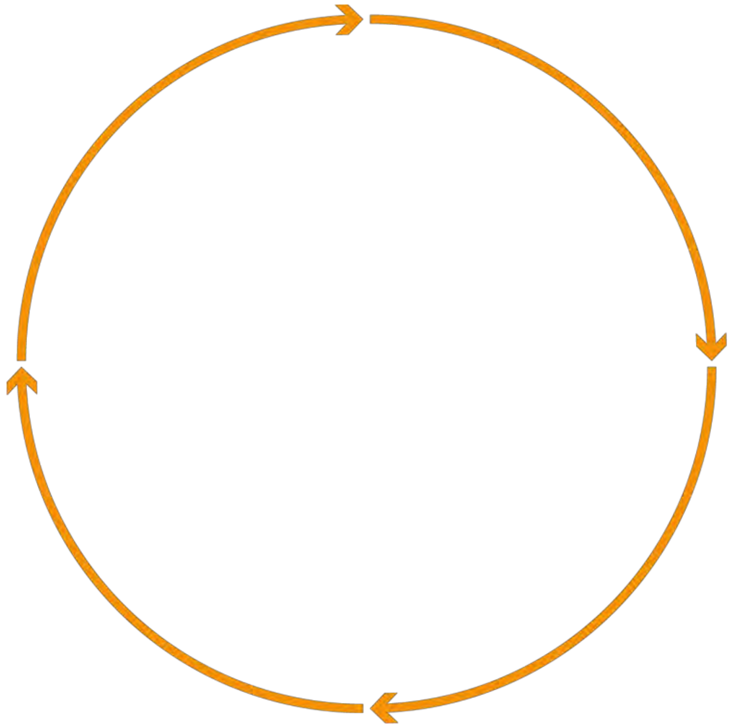
**Continuous elimination of waste**

# How to feed Evolution

**Plan – Do – Check – Act:** The Powerful Ingredient for Success

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

## Plan
- What to achieve
- How to achieve it



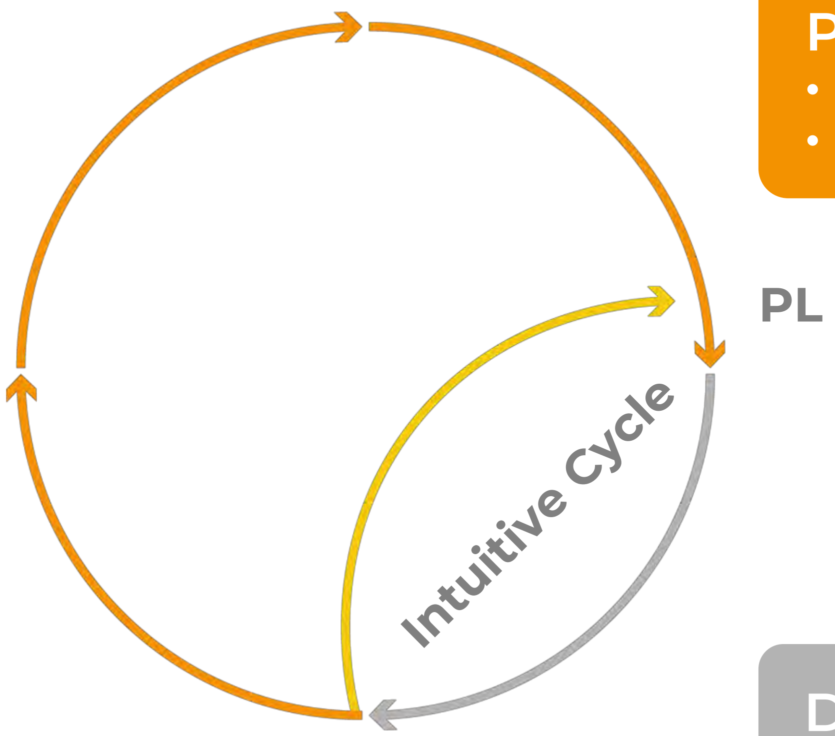
## Check
- Result according to plan?
- Achieved according to plan?

## Do
- Carry out the plan

# How to feed Evolution

**PL – Do :** The intuitive cycle

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




## Plan
- What to achieve
- How to achieve it

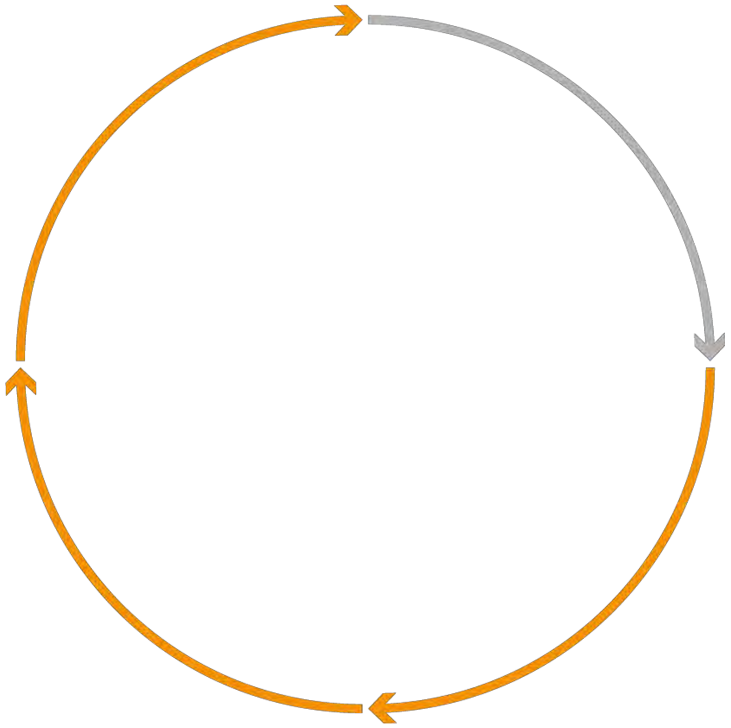
## Check
- Result according to plan?
- Achieved according to plan?

## Do
- Do something

# How to feed Evolution

**Plan – Do – Check – Act:** The Powerful Ingredient for Success

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



### Plan
- What to achieve
- How to achieve it

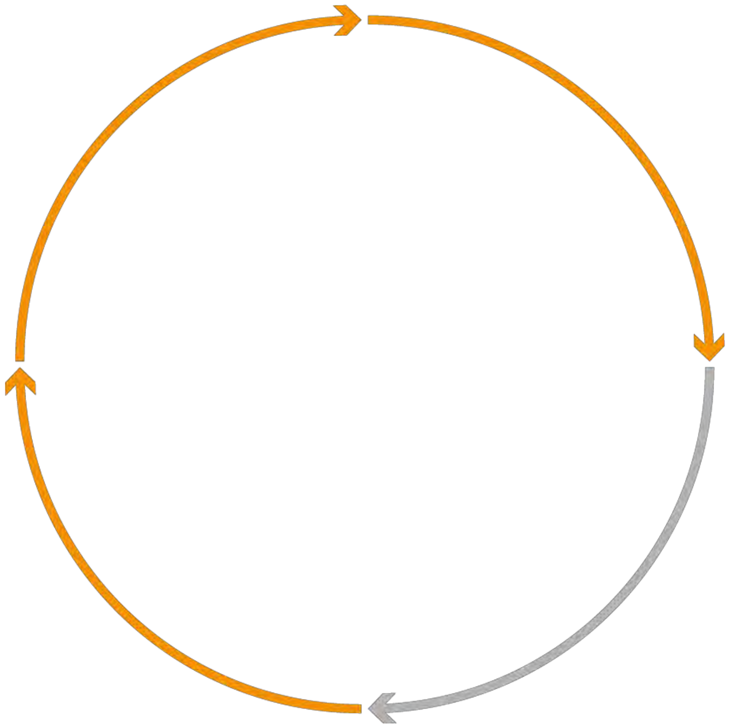
### Check
- Result according to plan?
- Achieved according to plan?

### Do
- Carry out the plan

# How to feed Evolution

**Plan – Do – Check – Act:** The Powerful Ingredient for Success

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



### Plan
- What to achieve
- How to achieve it

### Check
- Result according to plan?
- Achieved according to plan?

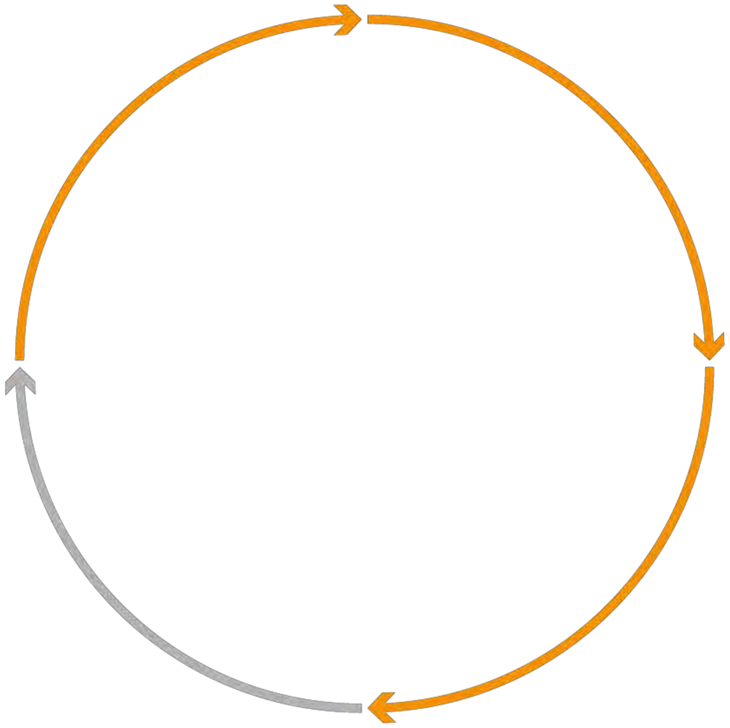
### Do
- Carry out the plan

# How to feed Evolution

**Plan – Do – Check – Act:** The Powerful Ingredient for Success

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

### Plan
- What to achieve
- How to achieve it



### Check
- Result according to plan?
- Achieved according to plan?

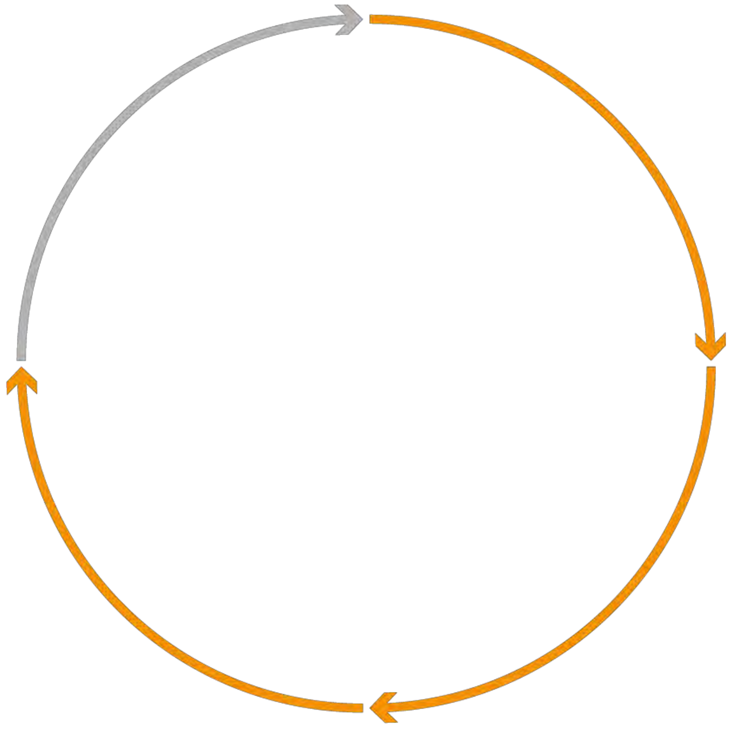
### Do
- Carry out the plan

# How to feed Evolution

**Plan – Do – Check – Act:** The Powerful Ingredient for Success

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

### Plan
- What to achieve
- How to achieve it



### Check
- Result according to plan?
- Achieved according to plan?

### Do
- Carry out the plan

# How to feed Evolution

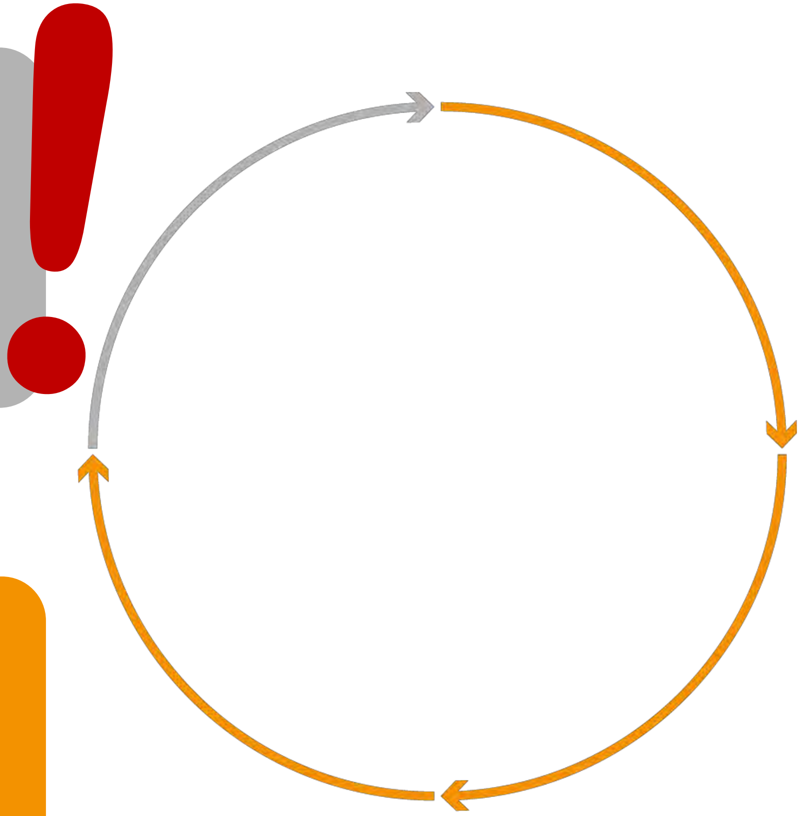
**Plan – Do – Check – Act:** The Powerful Ingredient for Success

## Plan
- What to achieve
- How to achieve it

## Do
- Carry out the plan

## Check
- Result according to plan?
- Achieved according to plan?

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



# Evolutionary Project Management (Evo)

## Plan-Do-Check-Act on every level

- **Zero Defects** — Attitude
  - Prevention costs less than repair
- **Business Case** — Why?
  - *Why* are we going to improve *what*?
- **Requirements Engineering** — What? How much? Are we done?
  - *What* are we going to improve, and *what not*?
  - *How much* will we improve? - Quantification
- **Architecture and Design** — How?
  - Selecting the optimum compromise for the conflicting requirements
- **Early Review and Inspection** — Check as early as possible
  - Measuring quality while doing, learning to prevent doing the wrong things

Quality

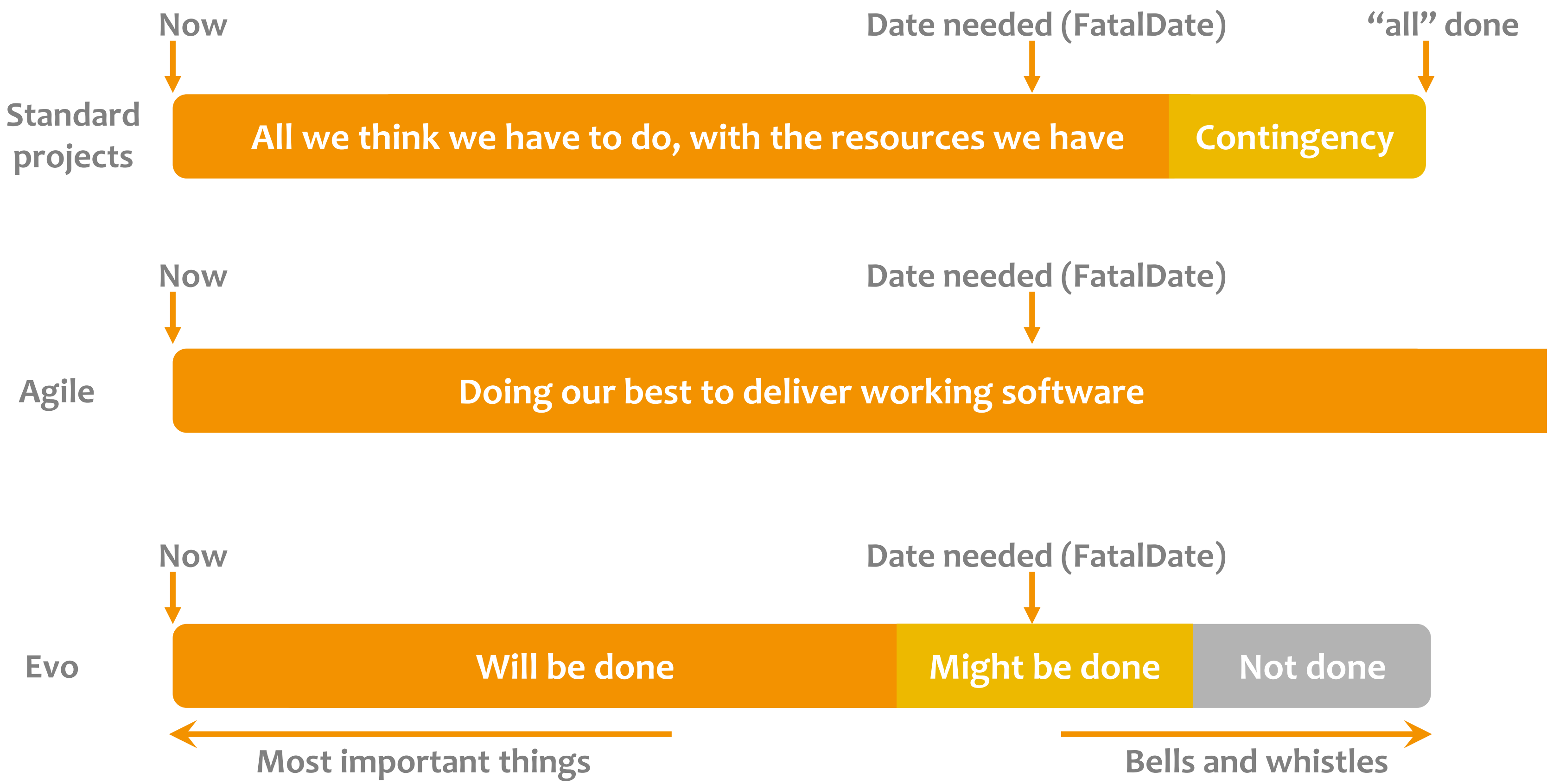
- **Weekly Task Cycle** — Efficiency of what we do
  - Short-Term Planning
  - Optimising Estimation
  - Promising what we can achieve
  - Living up to our promises
- **Bi-Weekly Delivery Cycle** — Effectiveness of what we do
  - Optimising the requirements and checking assumptions
  - Soliciting feedback by delivering real results to eagerly waiting stakeholders
- **Timeline** — **What will happen and *what will we do about it?***
  - Getting and keeping control of time: predicting the future
  - Feeding program/portfolio/resource management

On Time

Evolutionary Project Planning

# Time Line

## How do we know that we get what is needed when it’s needed?

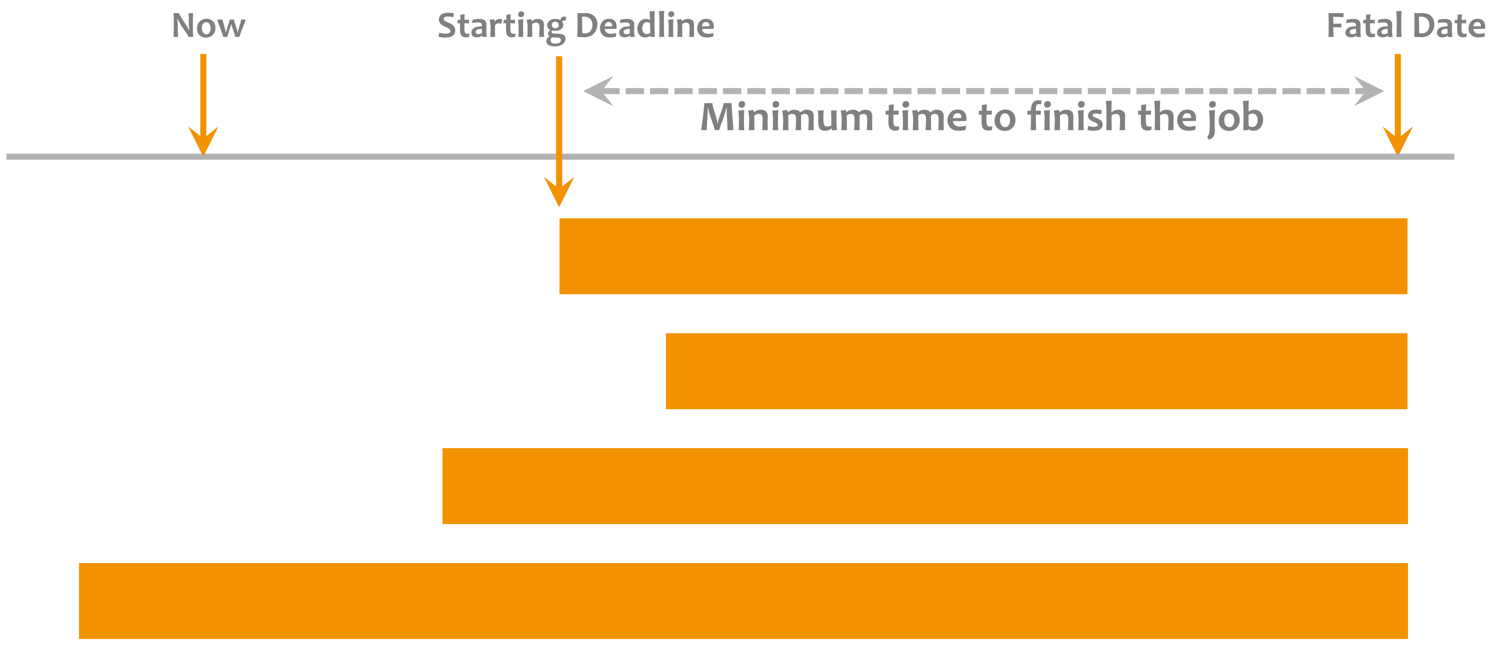



Better 80% 100% done,
than 100% 80% done

Let it be the most
important 80%

# Starting Deadlines

## More important to focus on than final deadlines




### Starting Deadline

Last day to start,

to make the finish on time

Everyday we start later,

we will end later

# Starting Deadline

- Buying trains from the catalogue, but some changes
- Cannot change everything: limited set of focus areas
- Example:

  Lifting train for maintenance

  - Supplier - lift
  - Maintenance - cable

- How much time left ?

  Supplier people already working on the final design

- What still to do? Does that fit the available time ?

  Talk to our maintenance, talk to supplier, decision, agreement

- Why waste your time ?
- What is Plan B ?

# Evolutionary Project Management (Evo)

Plan-Do-Check-Act on every level

- **Zero Defects** — Attitude
  - Prevention costs less than repair
- **Business Case** — Why?
  - *Why* are we going to improve *what*?
- **Requirements Engineering** — What? How much? Are we done?
  - *What* are we going to improve, and *what not*?
  - *How much* will we improve? - Quantification
- **Architecture and Design** — How?
  - Selecting the optimum compromise for the conflicting requirements
- **Early Review and Inspection** — Check as early as possible
  - Measuring quality while doing, learning to prevent doing the wrong things.

Quality

- **Weekly Task Cycle** — Efficiency of what we do
  - Short-Term Planning
  - Optimising Estimation
  - Promising what we can achieve
  - Living up to our promises
- **Bi-Weekly Delivery Cycle** — Effectiveness of what we do
  - Optimising the requirements and checking assumptions
  - Soliciting feedback by delivering real results to eagerly waiting stakeholders
- **Timeline** — **What will happen and *what will we do about it?***
  - Getting and keeping control of time: predicting the future
  - Feeding program/portfolio/resource management

On Time

Evolutionary Project Planning

# Weekly TaskCycle

## What are we going to do, what not, and why

### Removing waste *before* time spent

- Are we going to do the right things?
  - In the right order
  - To the right level of detail for now
- Optimising estimation, planning, and tracking abilities
  - To better predict the future

- Selecting the highest priority tasks
  - Never any lower priority tasks
  - No undefined tasks
- There are only about 26 plannable hours in a week (default 2/3)
  - In the remaining time, we do whatever else we have to do
- Tasks are always done, 100% done

# Weekly plan

## What are we going to do, what not, and why

### Weekly Plan

- What should we have achieved by the end of the week
- How much time do we have available
- 2/3 of available time is net plannable time
- What is most important to do, to achieve successfully
- Estimating effort needed to do these things
- Which most important things fit the available time
  - Default 26 hours per week
- What can, and are we going to do
- What are we ***not*** going to do

| Task | Hrs |
|---|---|
| Task a | 2 |
| Task b | 5 |
| Task c | 3 |
| Task d | 6 |
| Task e | 1 |
| Task f | 4 |
| Task g | 5 |
| Task h | 4 |
| Task j | 3 |
| Task k | 1 |

do

26

do not

| 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 |
| | | mmmmmm | 5 | | | for proj x |
| | | nnnnnnnn | 2 | | | for proj x |
| | | pppppppp | 3 | | | for proj y |
| | | qqqqqqqq | 12 | | | for proj y |
| | | rrrrrrrrrrrr | 6 | | | for proj y |
| | | ssssssssss | 4 | | | for proj y |
| | | ttttttttttt | 4 | | | for proj y |
| | | | 40 | | | |

TaskCycle Analysis
(retrospective)

learning

TaskCycle Planning
(presepective)

# Weekly Planning

## Optimising Time Spent on Planning

### Individual Preparation

- Conclude current tasks
- What to do next
- Estimates
- How much time is available

### Modulation / coaching 1-on-1

- Status - Learning
  - Previous tasks done, completely done, no need to think about it anymore?
- Priority Check
  - Are the new tasks really the most important things?
  - Feasibility
  - Will it be done by the end of the week?
- Commitment and Decision

### Synchronisation with group (team meeting)

- Synchronisation
- Concurrency
  - Do we have to synchronize?
- Formal Confirmation
  - This is what we plan to do
- Learning
- Helping
- Socialising

# New Oscilloscope Platform

- 4 teams of 10 people, 8 more people in Bangalore
- Introduced first in one team
- Other teams followed after a few weeks
- One team lagged because fear of ‘micro-management’

Heard at 1-on-1:

- Wow! Even if we would drop all you suggested,
  the 1-on-1’s will be kept, because so powerful:
  - We used to do something,
    afterwards finding out it wasn’t what it should be
  - Now we find out before, allowing us to do more right-the-first-time

# Results

## One year later

### Product manager:

- Schedule accuracy for this platform development was **50% better than the program average** over the last 5 years
(as measured by program schedule overrun)
- This product was the **fastest time-to-market** with the **highest quality** at introduction of any platform in our group in more than 10 years
- The team also won a prestigious **Team Award** as part of the company's Technical Excellence recognition program

www.malotaux.eu/doc.php?id=19 chapter 4.7.1, page 70

# Evolutionary Project Management (Evo)

## Plan-Do-Check-Act on every level

- **Zero Defects** — Attitude
  - Prevention costs less than repair
- **Business Case** — Why?
  - *Why* are we going to improve *what*?
- **Requirements Engineering** — What? How much? Are we done?
  - *What* are we going to improve, and *what not*?
  - *How much* will we improve? - Quantification
- **Architecture and Design** — How?
  - Selecting the optimum compromise for the conflicting requirements
- **Early Review and Inspection** — Check as early as possible
  - Measuring quality while doing, learning to prevent doing the wrong things.

Quality

On Time

- **Weekly Task Cycle** — Efficiency of what we do
  - Short-Term Planning
  - Optimising Estimation
  - Promising what we can achieve
  - Living up to our promises
- **Bi-Weekly Delivery Cycle** — Effectiveness of what we do
  - Optimising the requirements and checking assumptions
  - Soliciting feedback by delivering real results to eagerly waiting stakeholders
- **Timeline** — What will happen and *what will we do about it?*
  - Getting and keeping control of time: predicting the future
  - Feeding program/portfolio/resource management

Evolutionary Project Planning

# Quantified Requirements

## How to quantify

SMART

Specific

Measurable

### Definition:

| | |
|---|---|
| RQ27: | Speed of Luggage Handling at Airport |
| Scale: | Time between <arrival of airplane> and first luggage on belt |
| Meter: | <measure arrival of airplane>, <measure arrival of first luggage on belt>, calculate difference |

Attainable

### Benchmarks (Playing Field):

| | |
|---|---|
| Past: | 2 min [minimum, 2018], 8 min [average, 2018], 83 min [max, 20184] |
| Current: | < 4 min [competitor y, Jan 2018] ← <who said this?>, <Survey April 2018> |
| Record: | 57 sec [competitor x, Jan 2018] |
| Wish: | < 2 min [2022Q3, new system available] ← CEO, 19 Jan 2021, <document …> |

Realizable

### Requirements:

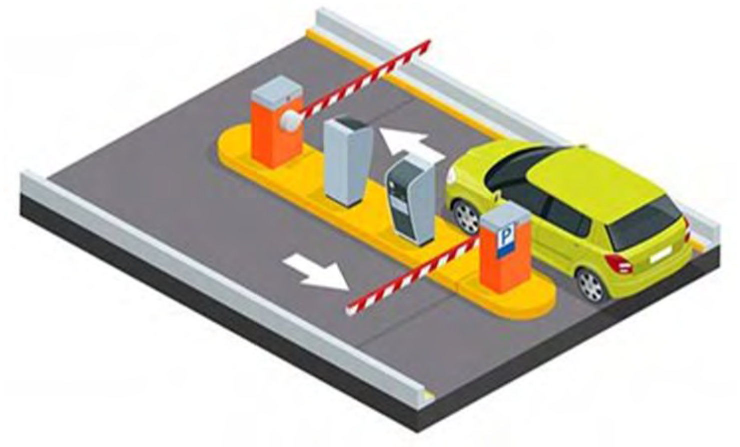
Time

Traceable

| | |
|---|---|
| Tolerable: | < 10 min [99%, Q4] ← SLA |
| Tolerable: | < 15 min [100%, Q4, Heathrow T4] ← SLA |
| Goal: | < 15 min [99%, Q2], < 10 min [99%, Q3], < 5 min [99%, Q4] ← marketing |

# Nice Requirements

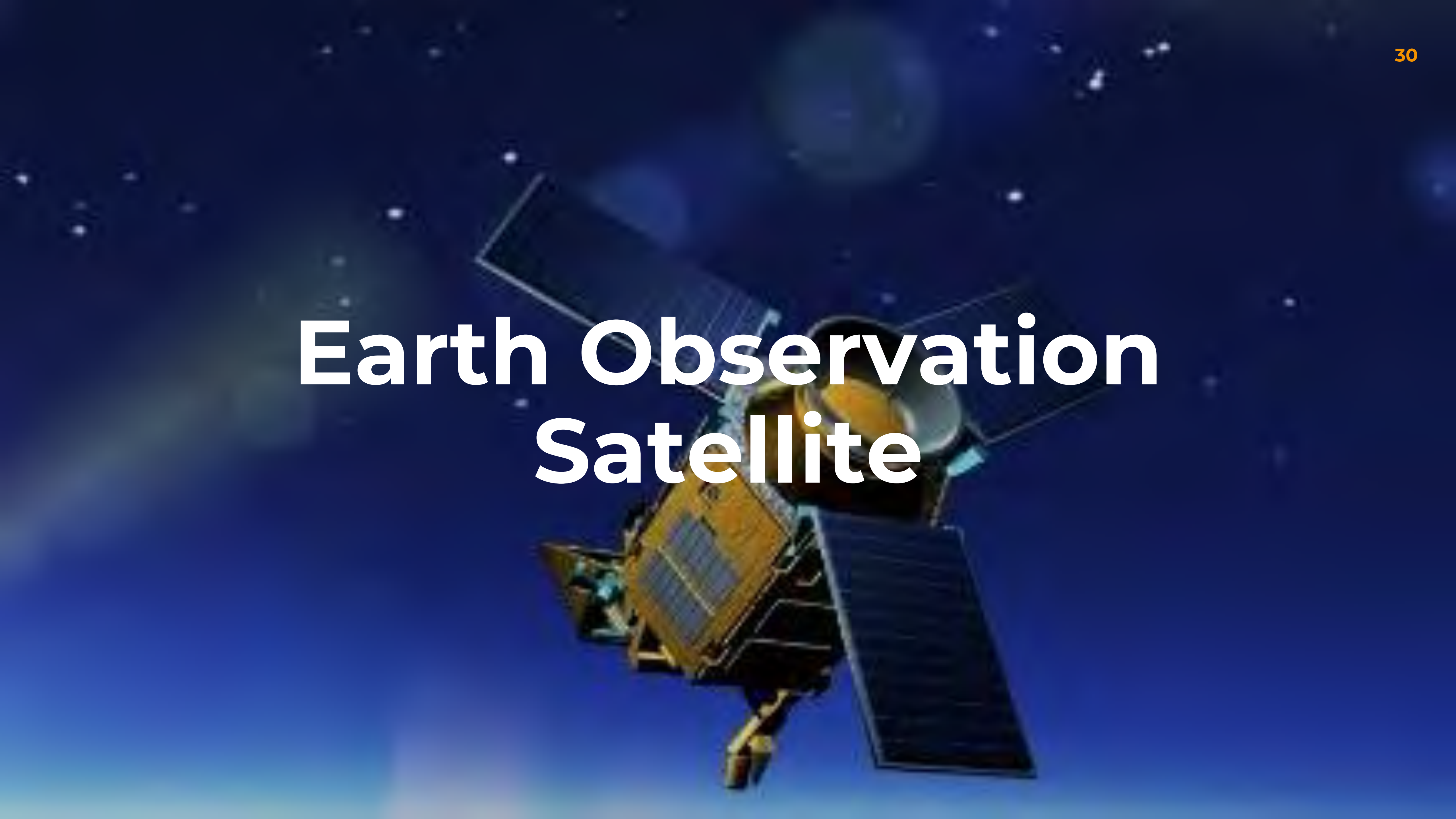
## Parking system



- Handle up to 400 cars per hour 9 sec per car
- Approval to enter: < 3 sec
- Uptime  99,95%
  downtime: 4.4 hr / yr
  @400 cars per hour → 1750 missed per year → deemed acceptable
- Response time < 150msec
- Max screen build up time < 500ms
- Life span 15 years

- Can you put a system at our office entrance ?
  - Took quite some weeks
  - Response time: 2 sec
  - Approval to enter: 7 sec
  - Can the architecture handle improving these up to required levels ?

# Earth Observation Satellite



# On Time

## Earth Observation Satellite

- Very experienced Systems Engineers
- One problem: They missed all deadlines
- Can you help us?
- **Taught them ‘Quality on Time’ Evo Planning**
- 9 weeks later: haven’t missed a deadline since
- 2.5 years later: delivered 1 day early
  (instead of expected 1 year late)
- Savings: at least 40 man-years (about €6million?)

# Awful Schedule Pressure !

## Quantifying the problem

### Problem - Solution

- Meeting with sub-contractors in three weeks
- 2 weeks to review documents
- “Impossible deadline”
- How many documents to review ?
- How much time per document ?
- How much time available ?
- Some suggestions…
- Result: well reviewed, great meeting, everyone satisfied

| | Doc 1 | Doc 2 | Doc 3 | Doc 4 | Doc 5 | Doc 6 | Doc 7 |
|---|---|---|---|---|---|---|---|
| John | x | | x | x | x | x | |
| Samuel | x | x | | x | | x | x |
| Paul | x | x | x | x | x | x | x |
| Michael | x | | | x | x | | |
| Marc | | | x | x | | x | x |

| Per Doc | | Hour |
|---|---|---|
| 4 Heavy | 15 | 60 |
| 3 Easy | 2 | 6 |
| | Total | 66 |
| Other Work | | 33 |
| | Total | 99 |

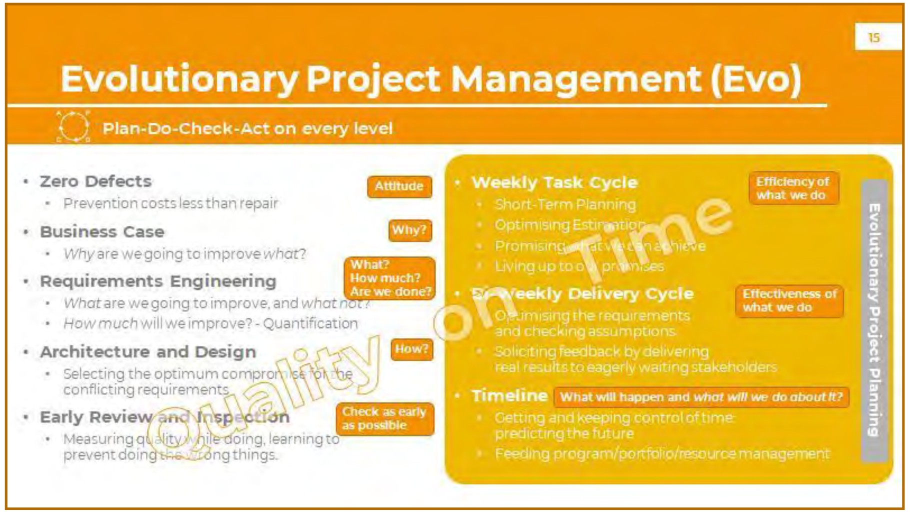
| | | |
|---|---|---|
| Available | 2x26 | 52 |

# From now on, will you deliver
# Quality on Time ??

The Right Results at the Right Time

**Niels Malotaux**

Quality on Time course
6 - 7 March 2023 - Zürich



# Classroom Course Highlights February 2023



| Course Name | Date | Location |
|---|---|---|
| Requirements - The good, the bad, the ugly | 20 - 22 February | Zürich |
| Practical MBSE & SysML | 20 - 22 February | Zürich |
| SE in a Nutshell (online) | 24 February | Online |

# Classroom Course Highlights Mach 2023



| Course Name | Date | Location |
|---|---|---|
| Quality on Time | 06 - 07 March | Zürich |
| Systems Architecting Intermediate Level | 13 - 15 March | Zürich |
| SE Foundations | 29 - 31 March | Munich |
| SE Management | 30 - 31 March | Munich |

# Classroom Course Highlights April 2023

| Course Name | Date | Location |
|---|---|---|
| Technical Problem Solving | 05 April | Zürich |
| INCOSE SEP Exam Preparation | 18 - 21 April | Zürich |
| SE in a Nutshell (online) | 28 April | Online |

** Early Bird discount of 10% on any of our 2-, 3-, 4- and 5-Day classroom courses if your register 6 weeks or more prior to course start date.*

To view the full schedule of upcoming courses, visit our website:
www.se-training.net

Follow us on LinkedIn for regular updates on free webinars!