# Summary of Lecture 16.10.2019


http://englishskills.se

… Very condensed summary of the 16.10.2019 lecture

https://de.fotolia.com

# Summary 16.10.2019

«Software Everywhere» … and what is the message?

## Success Stories

Software generates **Business Value**

⇒ Products, Services, Quality of Life, …

http://smallbusinessbc.ca

## Failure Stories

Software creates **Risks**

⇒ Accidents, failures, malfunctions, …

http://carpng.com

# Summary 16.10.2019

Requirements Specifications

**Dependability/Trustworthiness** is a consequence of good quality properties

**Functional** Properties

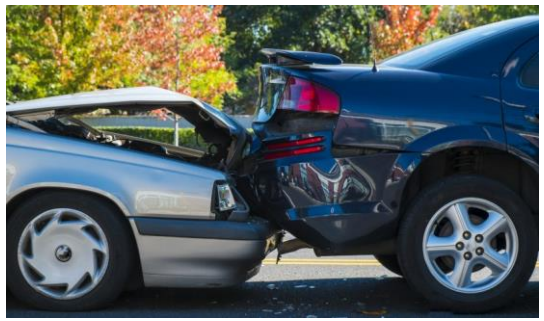
**Non-Functional** Properties

**Quality** Properties

https://www.usabilityblog.de

**Security**

**Safety**



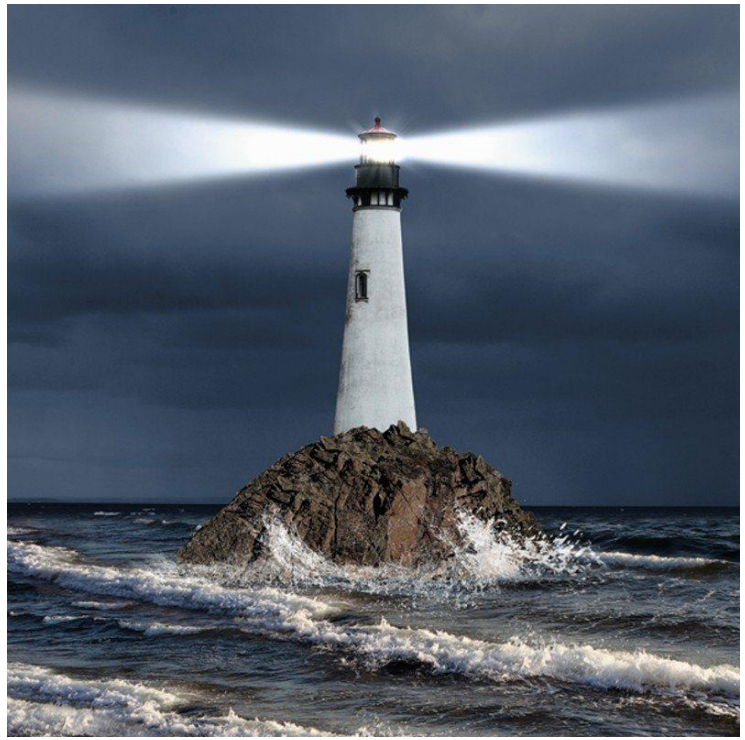

**Availability**

**... etc.**

Business-/Applications Functionality

# Summary 16.10.2019

Future-Proof Software-Systems

https://www.otto.de

*Our objective is:*

to build, evolve, and maintain

long-lived, mission-critical IT-systems

with a strong dependability,

an easy changeability

and a high business value.

http://www.gettyimages.ch

# Summary 16.10.2019

Lecture Dates:

| # | Date | Time | Location |
|---|---|---|---|
| **1** | Wed., 16. Oct. 2019 √ | 09:20 – 10:50 and 11:10 – 12:40 | APB/E006 |
| **2** | Wed., 30. Oct. 2019 | 09:20 – 10:50 and 11:10 – 12:40 | APB/E006 |
| **3** | Wed., 13. Nov. 2019 | 09:20 – 10:50 and 11:10 – 12:40 | APB/E006 |
| **4** | Wed., 27. Nov. 2019 | 09:20 – 10:50 and 11:10 – 12:40 | APB/E006 |
| **5** | Wed., 11. Dec. 2019 | 09:20 – 10:50 and 11:10 – 12:40 | APB/E006 |
| | *Christmas Holidays* | - | |
| **6** | Wed., 15. Jan. 2020 | 09:20 – 10:50 and 11:10 – 12:40 | APB/E006 |
| **7** | Wed., 29. Jan. 2020 | 09:20 – 10:50 and 11:10 – 12:40 | APB/E006 |
| **8** | 6. & 7. Feb. 2020 | FPSS Exams/Püfungen | tbd. |

Summary 16.10.2019

# Oral Exam

Participants can receive a grade via an **oral exam**
**(3 credits ECTS)**

Examination dates: 6. & 7.2020 February (+ more)


http://ipc.sze.hu

Deutsch

Examination Language
(**Your** Choice)

English

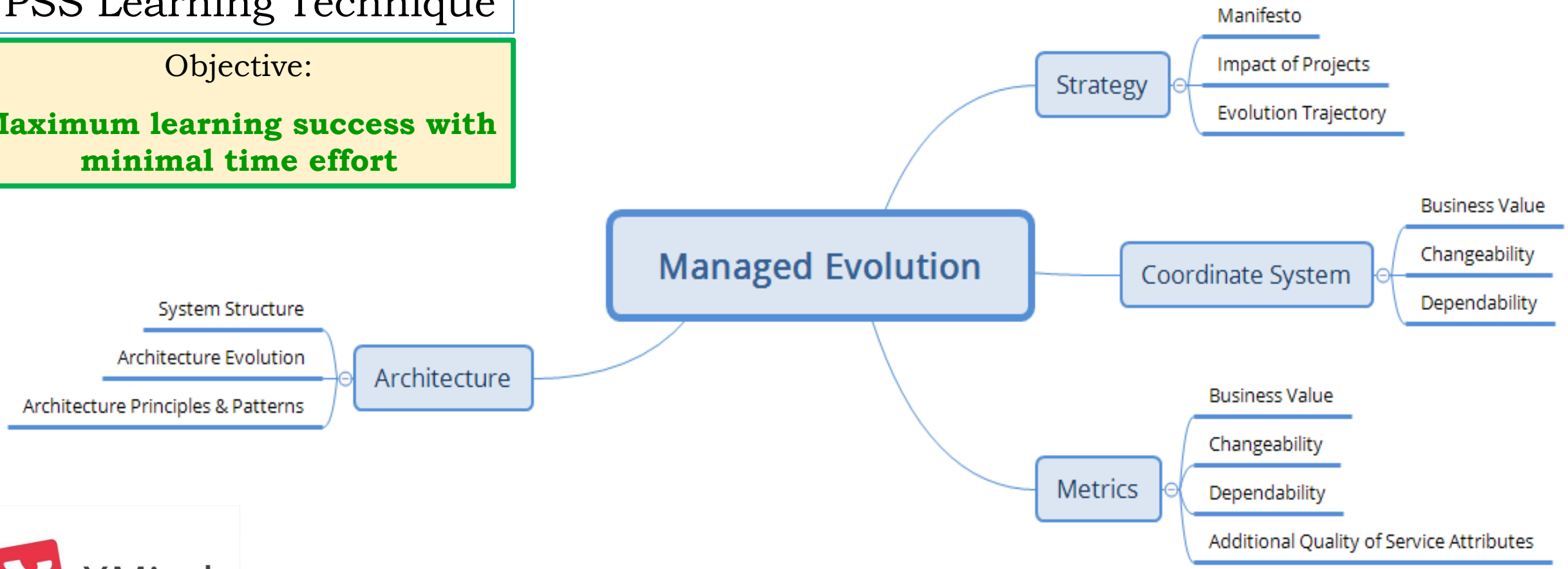
# Summary 16.10.2019

**Hierarchical Representation:**
«Mind Map»

## FPSS Learning Technique

Objective:

**Maximum learning success with minimal time effort**



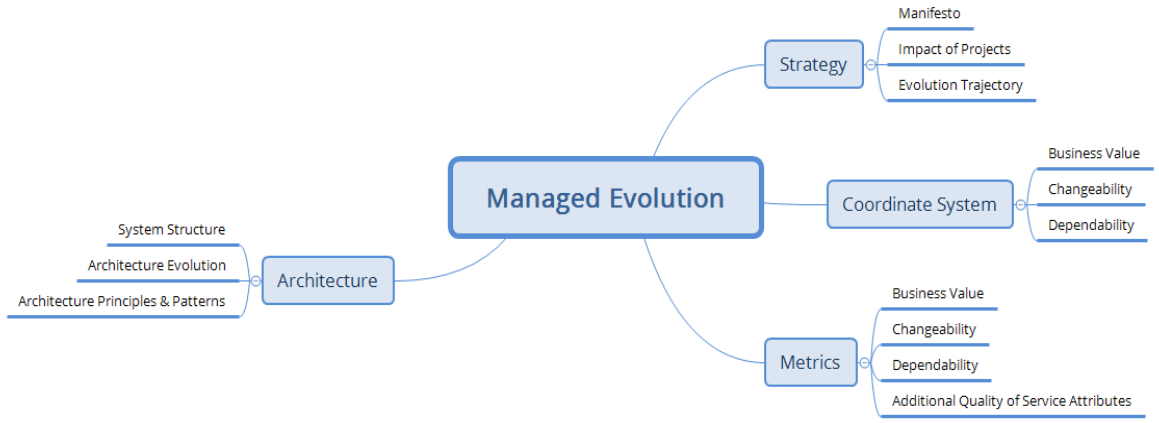

XMind

http://www.xmind.net

Summary 16.10.2019

**Step 2:**
Representation as «learning cards»

## Front: Concept Hierarchy




## Back: Notes

- Negative: Technical debt, architecture erosion, business + market pressure
- Business value, changeability and dependability are continuously improved
- Managed Evolution coordinate system: ME evolution channel
- Tracking through metrics (BV, THM, DevC, size)
- Dependability = survival / Changeability = adaptability to new requirements (THM, DevC)
- Architecture = Key success factor
- Business ↔ IT interests/conflict

# Summary 16.10.2019

FPSS Definition


**Managed Evolution Strategy**

**Architecture Principles**

A *future-proof software-system* is a structure

that enables the management

of complexity, change and uncertainty

with the least effort, with acceptable risk, and with specified quality properties

**Changeability**

**Dependability**

Domain-specific
**Quality Properties**

# Summary 16.10.2019

**Business Value** = Reason for the Creation of the Software

**Changeability** = Competitive Advantage (DevC, TtM)

**Dependability** = Assures Survival in today's World

Domain-specific **Quality Properties** = «Fit for Function»

**FPSS**

# Summary 16.10.2019

**Metrics**

Example:

http://www.eco-way.ch/?p=10846

## Business Value

Net Present Value: **NPV**

## Changeability

averaged

$$\text{Changeability} = \frac{\Sigma \text{TtM}_i * \Sigma \text{DevC}_i}{(\Sigma \text{Size}_i)^2}$$

How much does it **cost** and **how long** does it take to produce a certain **functionality**

Unit: (days*k€)/#UCP2

## Dependability

- **General Resilience**
- **Domain-specific dependability properties**

## Domain-specific Quality Properties

→ **Additional properties: Usability, Resources, Energy, ...**

# Summary 16.10.2019

Program Module

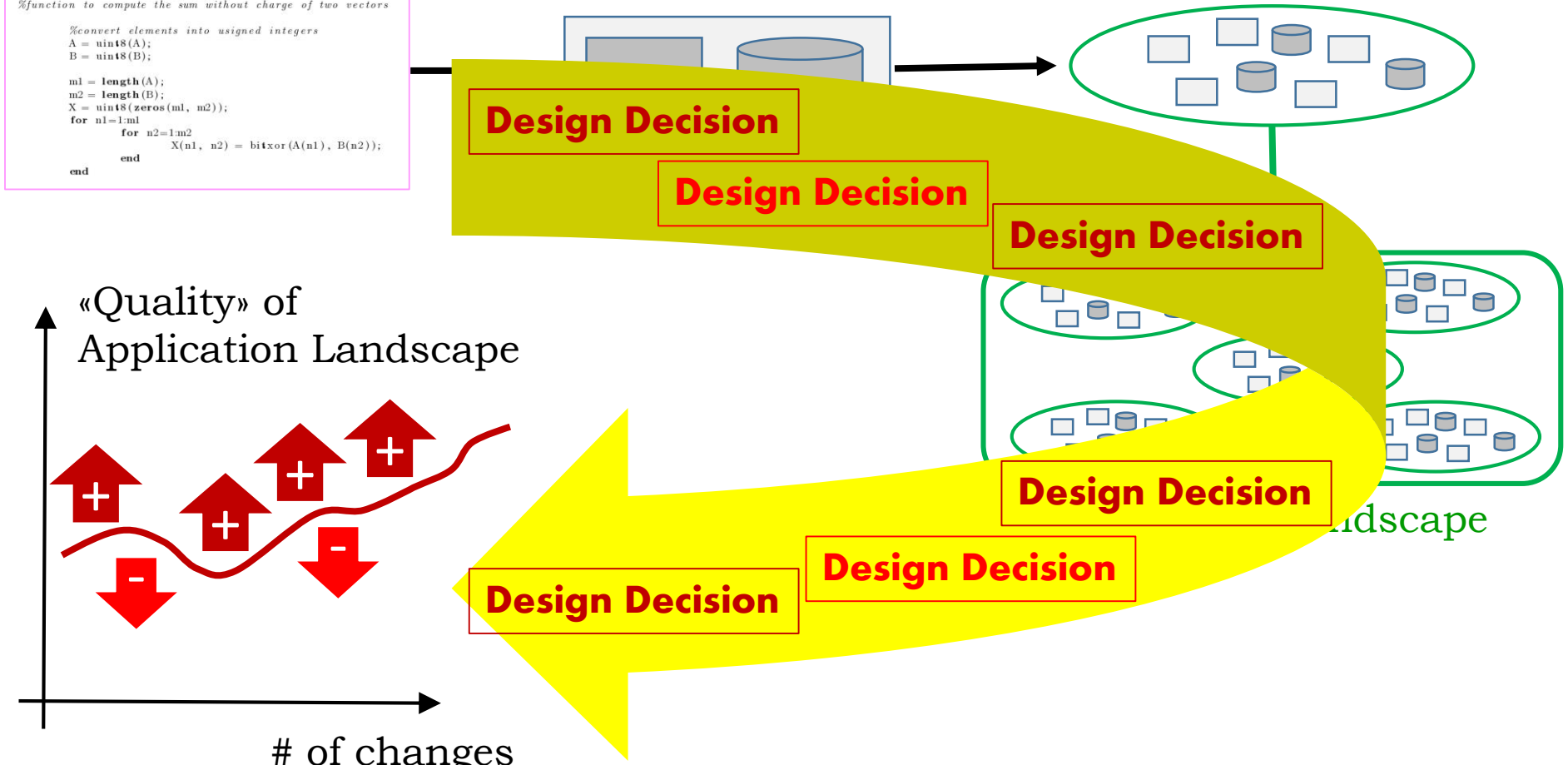
Component

Application

```
The next code will be directly imported from a file:
function X = BitXorMatrix(A,B)
%function to compute the sum without charge of two vectors

        %convert elements into usigned integers
        A = uint8(A);
        B = uint8(B);

        m1 = length(A);
        m2 = length(B);
        X = uint8(zeros(m1, m2));
        for n1=1:m1
                for n2=1:m2
                        X(n1, n2) = bitxor(A(n1), B(n2));
                end
        end
```

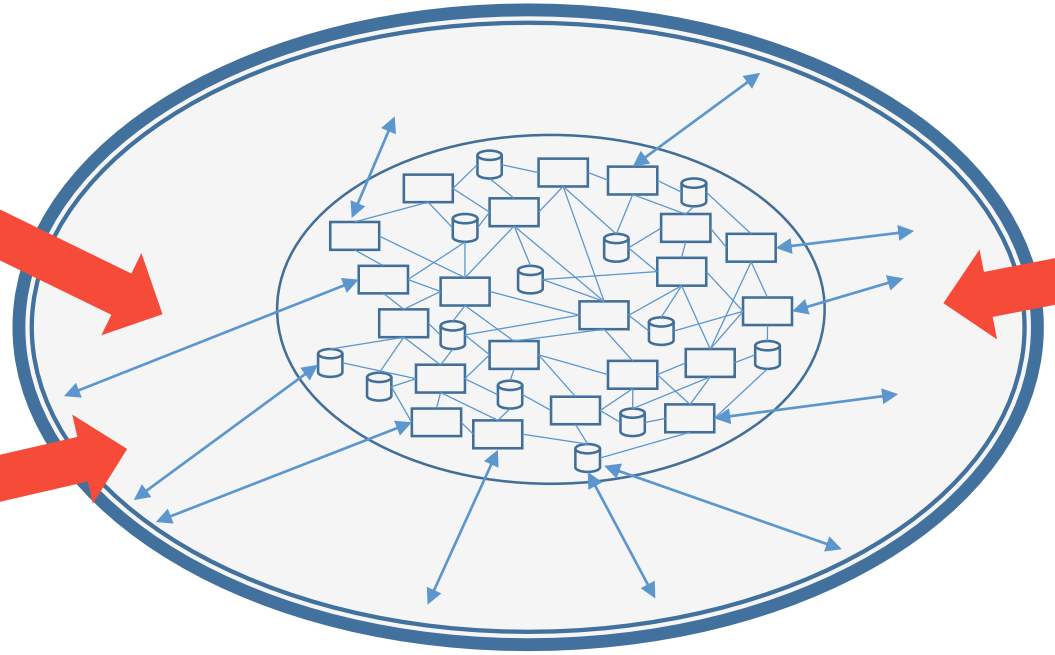



The «quality» of the application landscape is the sum of all design & implementation decisions

# Summary 16.10.2019

## Three Devils of Systems Engineering

Change



Complexity

Uncertainty

… you cannot fight complexity, change and uncertainty
⇒ **You can only manage it !**

… by using principles, methods, strategies, and processes for **future-proof software-systems**

# Summary 16.10.2019


Activity: Steering the development & evolution
→ **Strategy**

Parts of the system and their relationssships
→ **Architecture**

A *future-proof software-system* is a structure that enables the management of complexity, change and uncertainty with the least effort, with acceptable risk, and with specified quality properties

Acceptable probability for undesired effects and consequences
→ **Dependability**

Assuring the desired non-functional properties
→ **„Fit for Purpose“**

Best value for the parameters ‘money‘ and ‘time-to-market‘
→ **Changeability**

# Summary 16.10.2019


Managed Evolution Strategy

Architecture [Principle-Based Architecting]

Complexity

Change

Uncertainty


Future-Proof Software-System

Business Value

Changeability

Dependability

cont.