Software engineering in industrial practice

Module 10: Large-Scale Project Management

Christian Schmitz

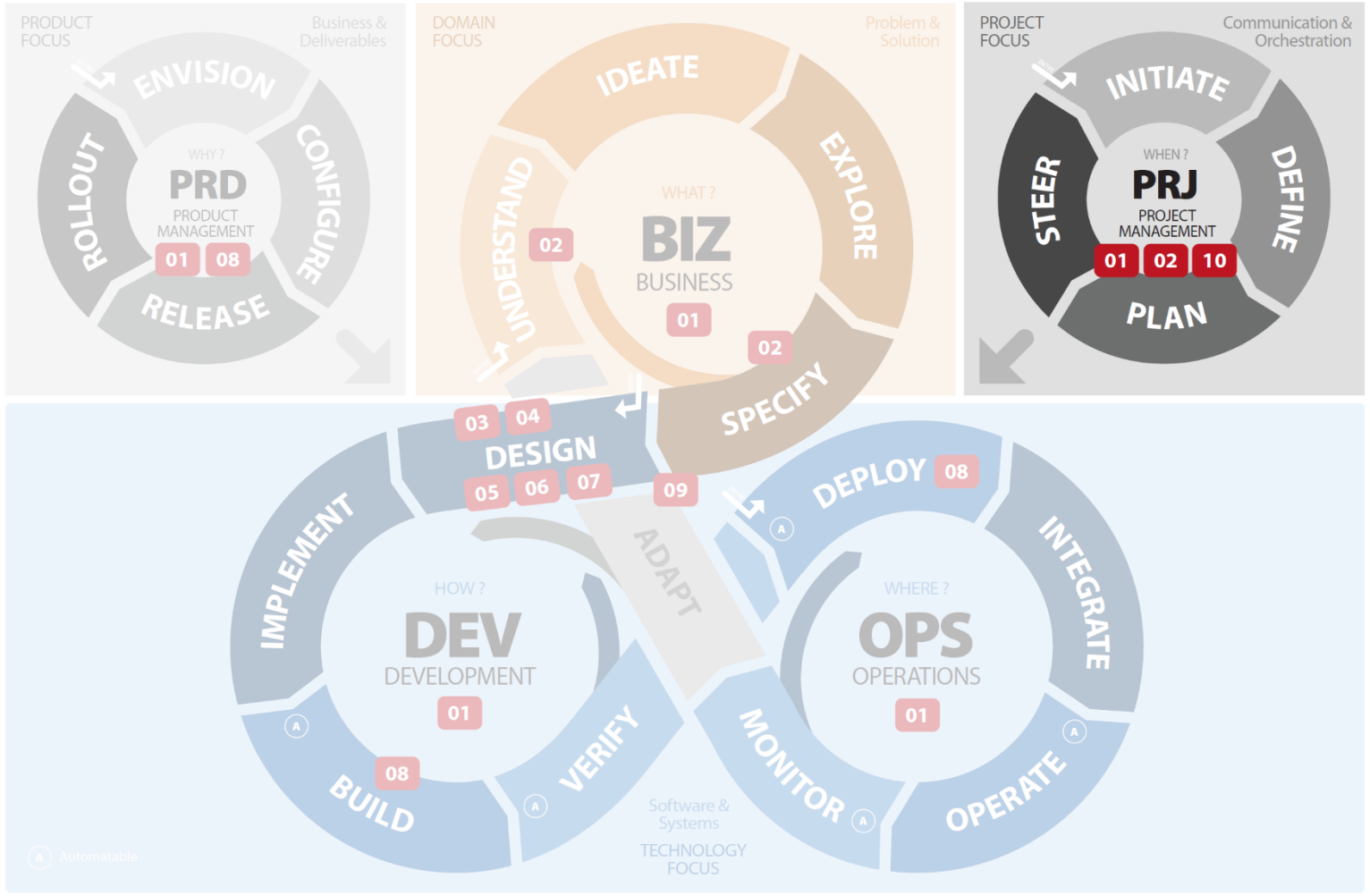
Speaker

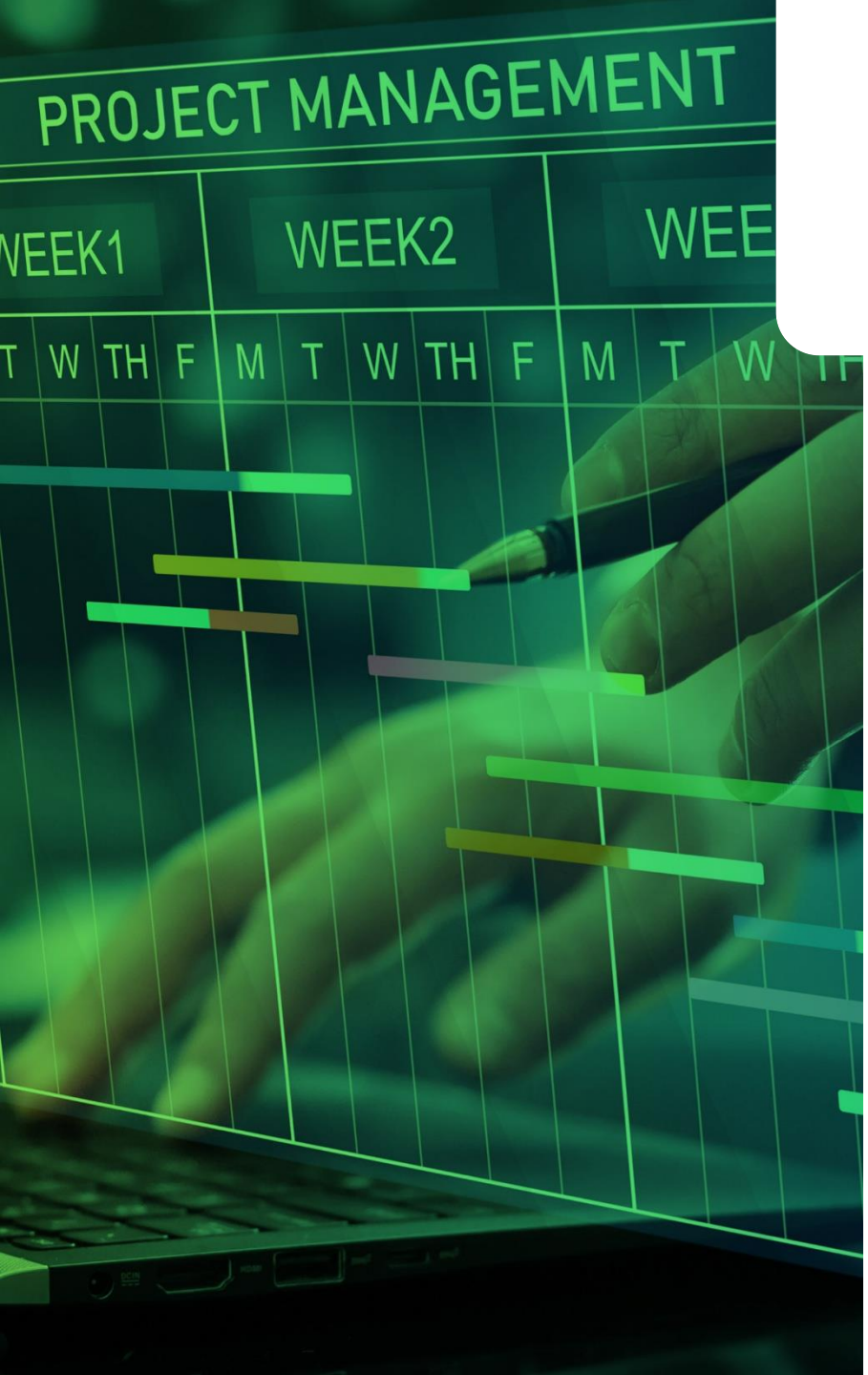
Christian Schmitz
Division Manager

Dieter Ebhart
Projectmanager

Would you like to stay in **touch**?
Feel free to contact me on **XING** or **LinkedIn**, keyword: Lecture SEIP

Software Engineering Workflow &
Lecture Software Engineering in Industrial Practice (SEIP)





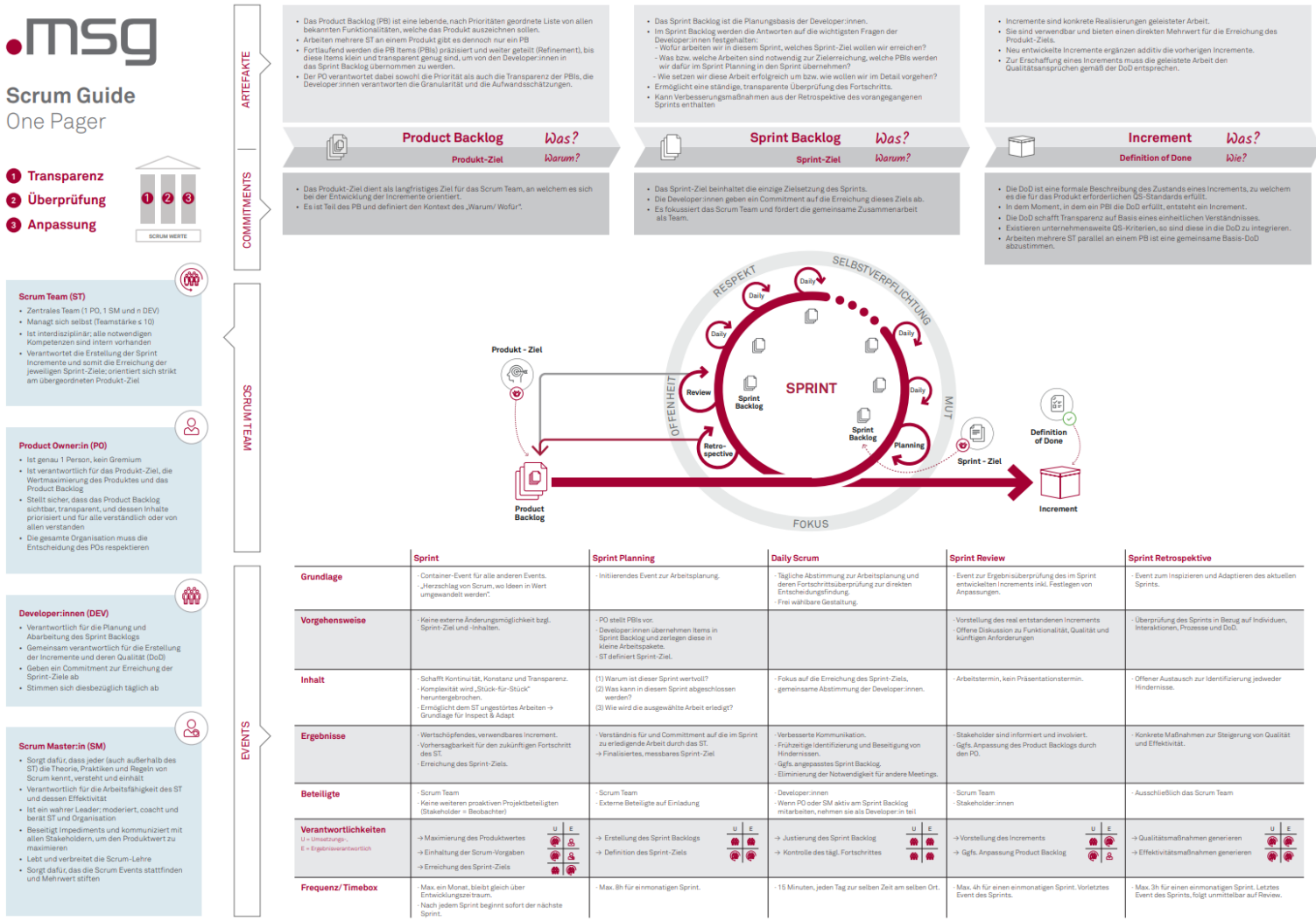
Large-Scale Project Management
Planned process

15:10 - 15:30	Enterprise Agile Frameworks
15:30 - 15:50	Economic efficiency of IT projects
15:50 – 16:00	Effort estimation
16:00 – 16:30	Project Controlling

Large-Scale Project Management

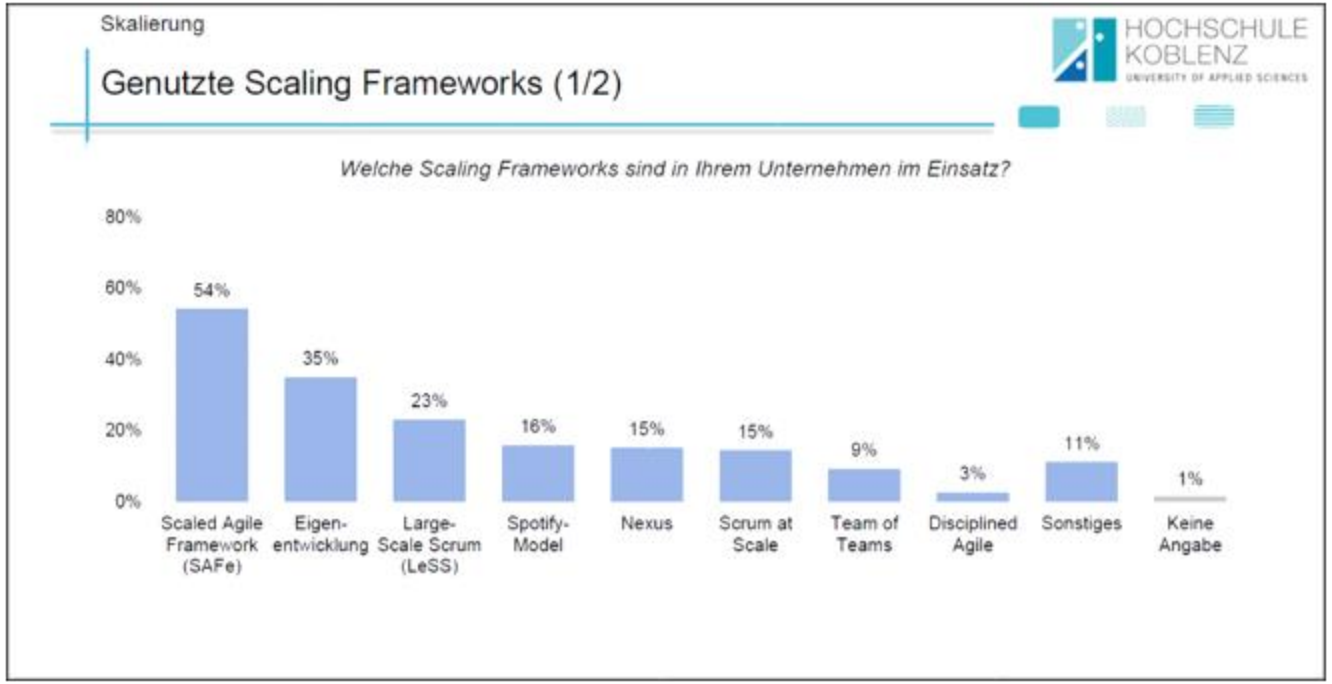
Enterprise Agile Frameworks

SCRUM at a glance

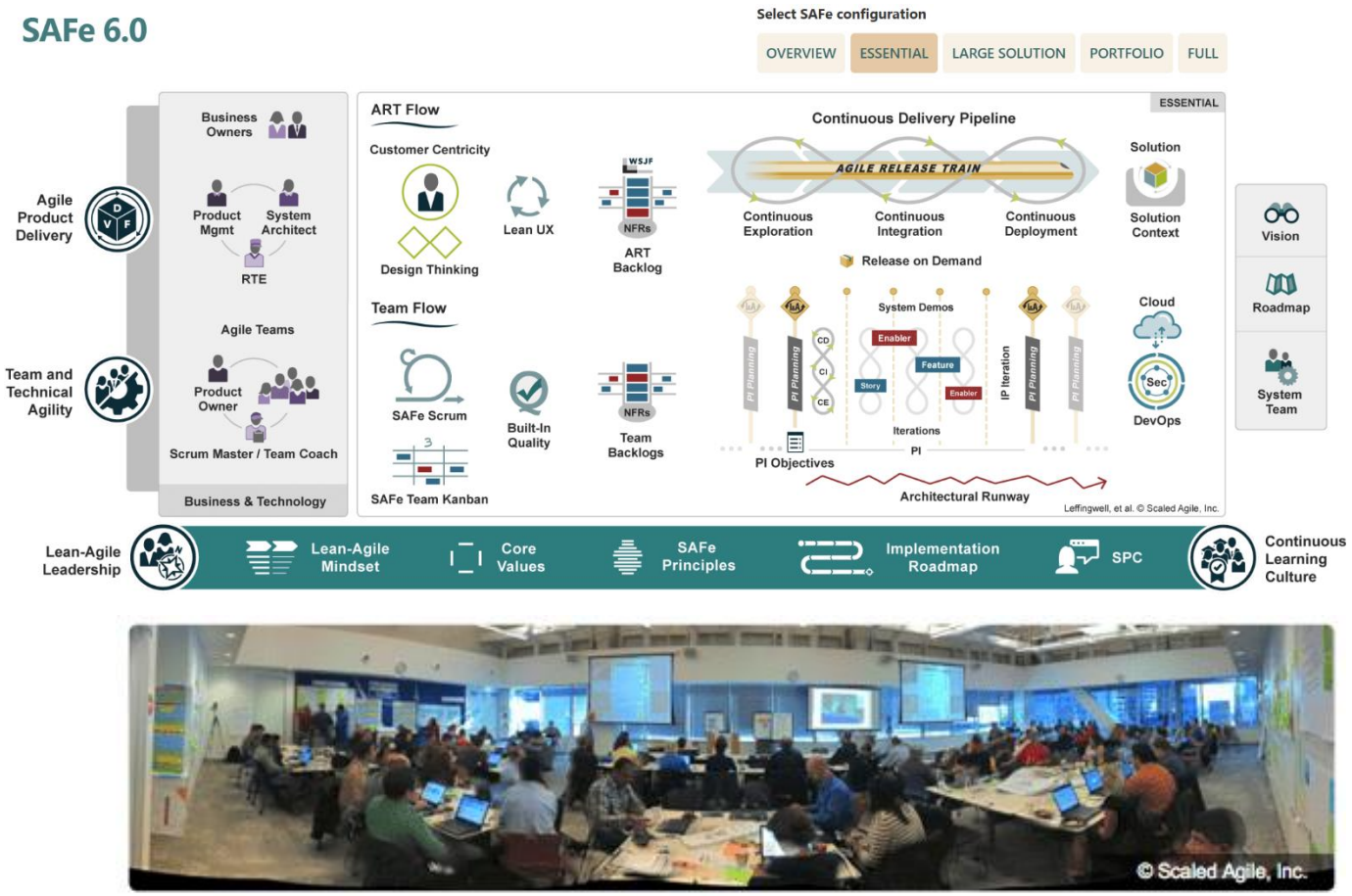


Overview of different enterprise agile frameworks

- SAFe and LeSS are the predominantly used enterprise agile frameworks
- Both frameworks are widely used
- Both frameworks have been implemented successfully or could be implemented incorrectly



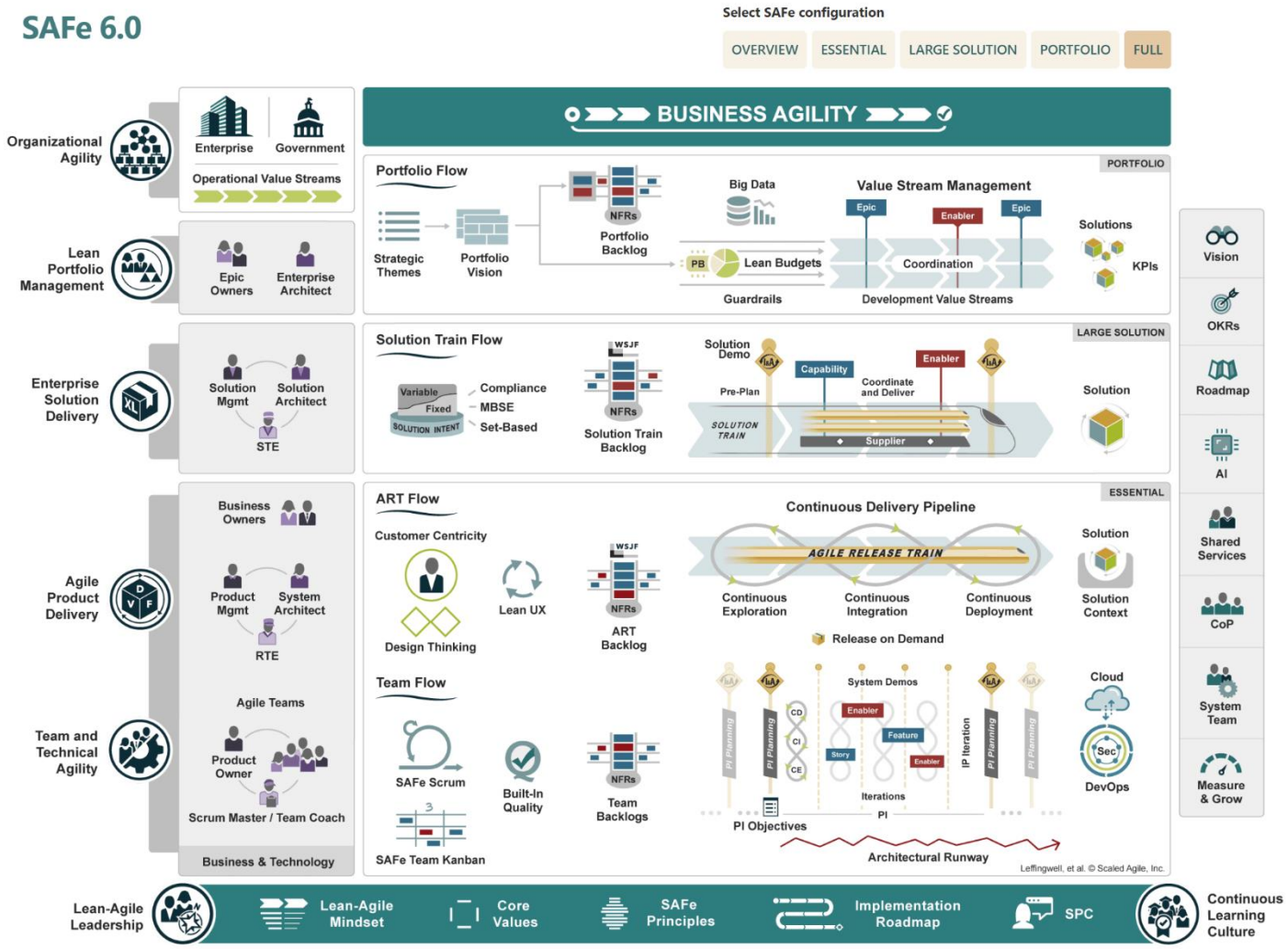
SAFe at a glance (1/2)



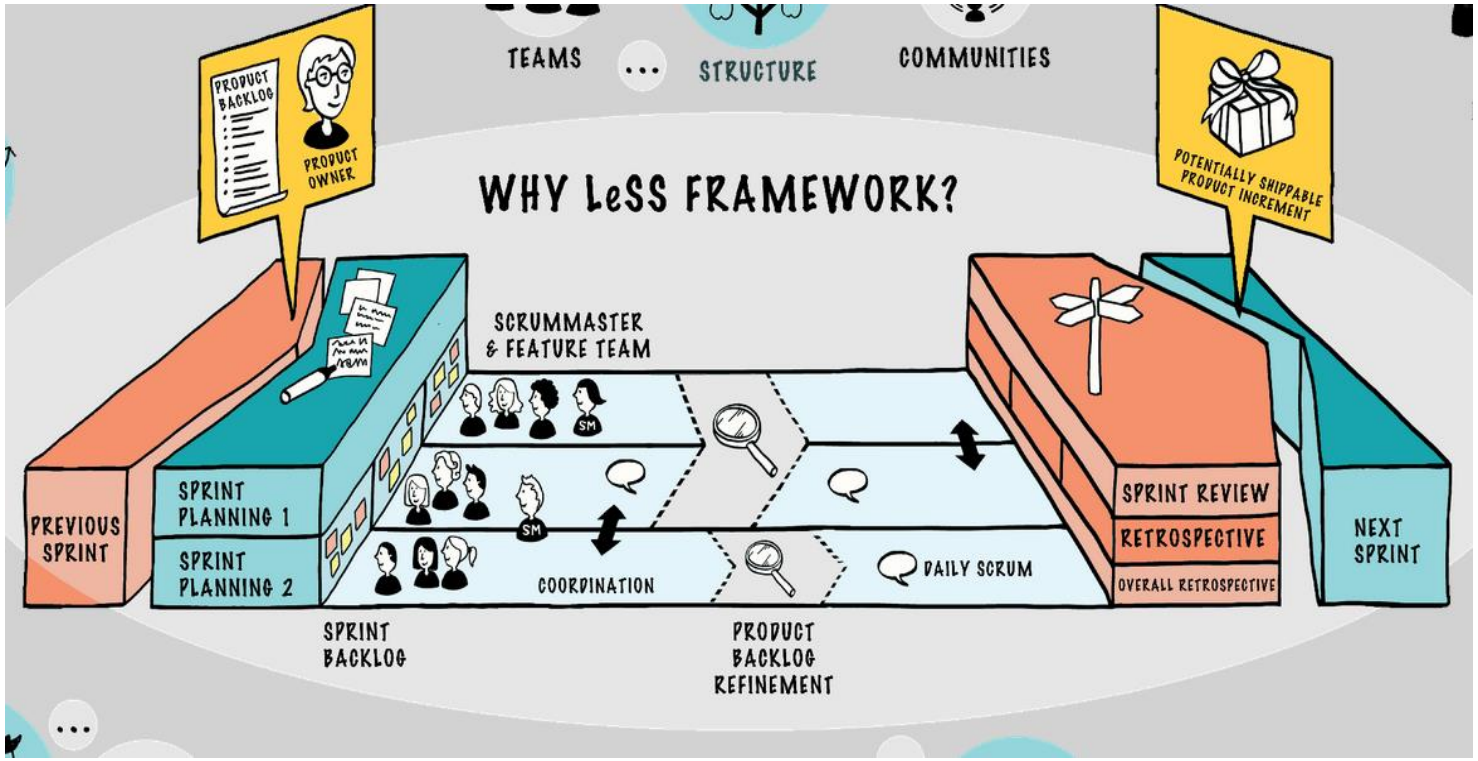
Summary

- Several roles (and maybe hierarchies)
- Big room meetings with multiple teams
  - PI-Planning,
  - Inspect & Adapt (Review, Retro)
- Team Meetings (with one team only)
  - Refinement
  - Iteration Planning Review, Retro
- Same Cadence with 5 Iterations and with Exploration, Integration, Deployment in an Iteration and PI
- Overall Consistent Approach on more levels e.g., Design Thinking or Portfolio Management (see next image)

SAFe at a glance (2/2)



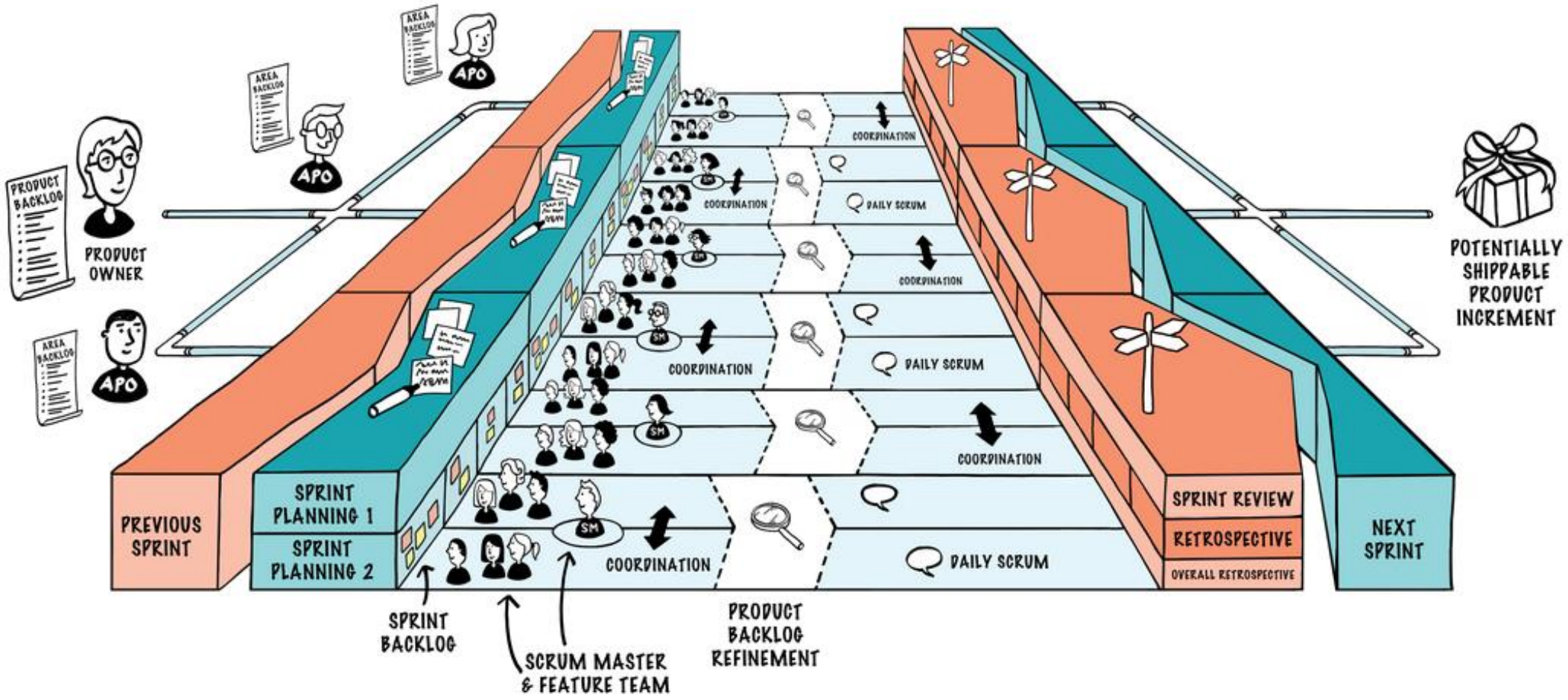
LeSS & LeSS Huge at a glance (1/2)



Summary

- LeSS does not talk much about hierarchies or roles
- Big room meetings with multiple team
  - Planning 1
  - Refinement
  - Retro jumpsuit
  - Review
- Team meetings (with one team only)
  - Planning
  - Refinement
  - Retro
- Focus on technical Software Development (not Portfolio)

LeSS & LeSS Huge at a glance (2/2)



Comparison chart of selected enterprise agile frameworks

Kosten / Aufwand

Auf der Suche nach dem besten Agile Framework: SAFe, LeSS oder ...

Kategorie	Kriterien	LeSS	Nexus	Spotify	Scrum @ Scale	SAFe
Allgemeine Info	Von wem und wann?	Craig Larman, Bass Vodde, 2005	Ken Schwaber, 2015	Henrik Kniberg, Anders Ivasson, 2012	Jeff Sutherland, 2018	Dean Leffingwell, 2011
	Dokumentation	LeSS.works	scrum.org (Nexus Guide)	labs.spotify.com	scrumalliance.org	scaledagileframework.com
	Basis (Scrum/ Kanban, ...)	Scrum	Scrum	offen für alle agilen Methoden	Scrum	offen für alle agilen Methoden
Referenz / Einsatz	Referenzkunden	Telekommunikation, BMW, UPS, > 30	keine bekannt	Spotify, ING, Deutsche Telekom, Rewe Digital, > 4	SAP, Intel, Comcast, Allianz	Hoch / viele >60
	Teamgröße	LeSS: bis ca. 8 Teams; LeSS Huge: ab ca. 8 Teams	3 bis 9 Teams	Squad < 8, ab 4 Squads	Team SoS: 2-5 ; SoSoS 6-25 Mtarbeiter SoS <45; SoSoS <225	ab 7 Teams
Einsatz	Unterstützt standortüber-greifendes Arbeiten	anwendbar Co-located	anwendbar	nicht anwendbar (Huddle Rooms)	anwendbar	anwendbar aber nicht empfohlen (Big Room Planning)
Bewertung	Verbindlichkeit/ Strukturierungsgrad	niedrig bis mittel	mittel	mittel bis hoch	hoch	hoch
	Abhängigkeiten zwischen den Teams	LeSS: möglich; LeSS Huge: möglichst keine Abhängigkeiten	möglichst keine Abhängigkeiten (oder minimieren)	möglichst gering	offen => eigene Governance möglich	viele Abhängigkeiten möglich => wird gesteuert
	Artefakte	Inkrement, 1 Produkt Backlog, n Sprint Backlog, m Area Back-logs, 1 DoD	Nexus Sprint Backlog, Product Backlog, Increment	Squad, Backlog, Roadmaps	Scrum Backlog, SoS Impedimemnt Backlog, Release Plan	Program Board, PI Objectives
	Events und Zeremonien	Overall Retro, großes und kleines Refinement, Planning 1 (which), Planning 2 (how), Daily (öfter mal woanders teilneh-men), Communities	Nexus Sprint Planning, Nexus Sprint Retro, Nexus Sprint Review, Nexus Daily Scrum	Scrum Kanban Meetings, Hackdays, Quartalsstudie, Chapter Meeting, Gilde Treffen	Scrum (Daily, ..); Scaled Daily Scrum, Exec. Meta Scrum, PO Team Release Planning	PI Planning, SoS, Syst. Demo; (epic, feature, story) Dependancy Map Enabler
Einführung	Kosten / Lizenz / Schulung	agiles Mindset	gering	abhängig von aktuellem kulturellen Umfeld	Lizenz: durchschnittlich; Change: hoch; Training: niedrig	mittel bis hoch
	Was muss „ontop" getan/ entschieden werden?	Mindset organisatorische Änderungen	Agiles Mindset	ggf. Agile / Spotify Mindset Schulung	Change Mgmt: Transformation, Setup; FW: Fehlende Vorgaben in SoS entwickeln	Implementation Roadmap durchlaufen: v.a. Schulung für alle, LACE Team bevoll-mächtigen
	Organisationskultur Ist-Bild => Wunschbild	–	–	Agiles Mindset, willing to fail, failure culture	Startup	eher geeignet für traditionelle Unternehmen, für alle offen
	Risiken und Hindernisse	–	–	kein Blick auf das Product Backlog als Ganzes durch zu viel Autonomie	Change Aufwand hoch, Freiheit => Chaos?	kann fehlinterpretiert werden und über bestehende Organisation übergestülpt werden ohne viel zu ändern
Besonderheiten	Besonderheiten	LeSS Test, Area Product Owner, Travelers	Integration Team	Autonomie Squads, Huddle- rooms, Fehler-Kultur, Gemeinschaft von Hierarchie, ständige Motivation, langfristige Ziele bei Squads	ganze Organisation, methodische Governance im SoS, nicht FW ; leichtgewichtig	Selfassessment (Essential SAFe, Lean Enterprise, Dev Ops Health,...), Imple-mentation Map, geeignet für (Weiter-) Entwicklung von Programmen (mehrere Produkte)
	Rollen (extra)	–	Integration Team	Agile Coach	Chief Product Owner	Release Train Engineer, Pro- duct Manager, SPC, System Architect, Business Owner, System Team

Strukturierungsgrad / Verbindlichkeit

Large-Scale Project Management

Economic efficiency of IT projects

Software engineering in industrial practice
Project management: economic efficiency

1. Basics and definitions of terms
2. Economic efficiency of IT projects
3. Literature

Software engineering in industrial practice
Project management: economic efficiency

1 **Basics and definitions of terms**

2 Economic efficiency of IT projects

3 Literature

IT efficiency

IT investments

- IT plans and IT projects often represent major investments in the company.
- The often non-transparent benefits are offset by high costs.
- Limited IT budgets require an objective prioritization of these investments.

IT organization

- Profitability is a key issue for every company.
- The IT organization is seen as a cost driver.
- In contrast to "traditional" corporate functions, IT used to be seen as a "black box" for decision-makers, but since the rise of digital business models, the benefits of IT have become clearer.

Objective

- Measurable, complete and sustainable criteria as a basis for business decisions
- Ongoing monitoring of target achievement

$$\text{Economic efficiency through IT} = \frac{\text{Benefits through IT}}{\text{IT costs}}$$

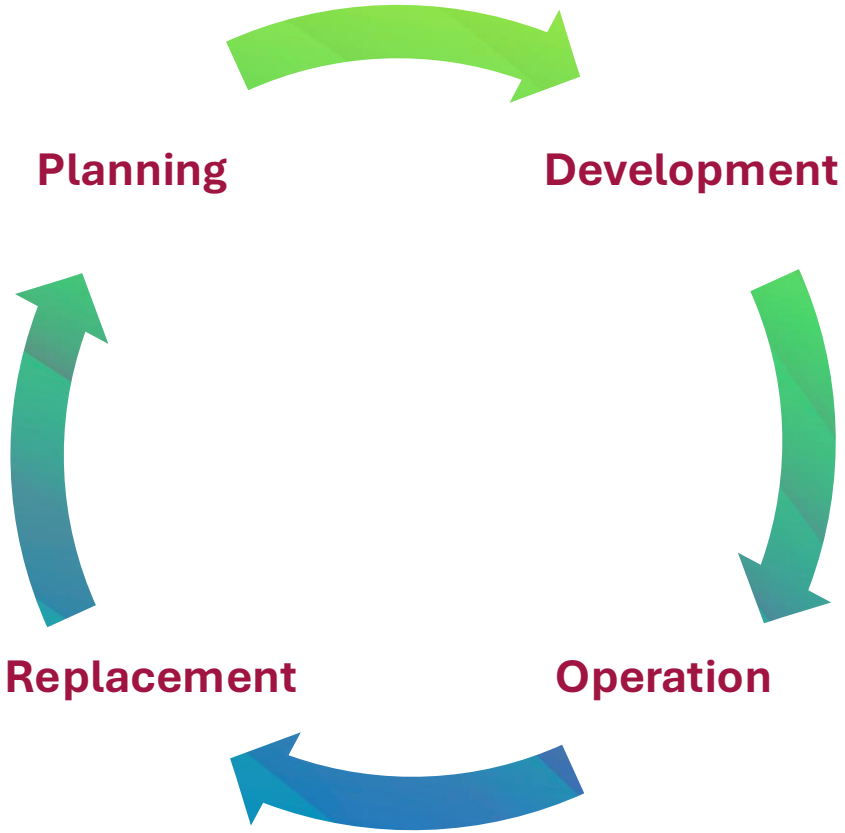
Software engineering in industrial practice
Project management: economic efficiency

1 Basics and definitions of terms

2 **Economic efficiency of IT projects**

3 Literature

The phases of an IT project



Neglected or unconsidered cost drivers

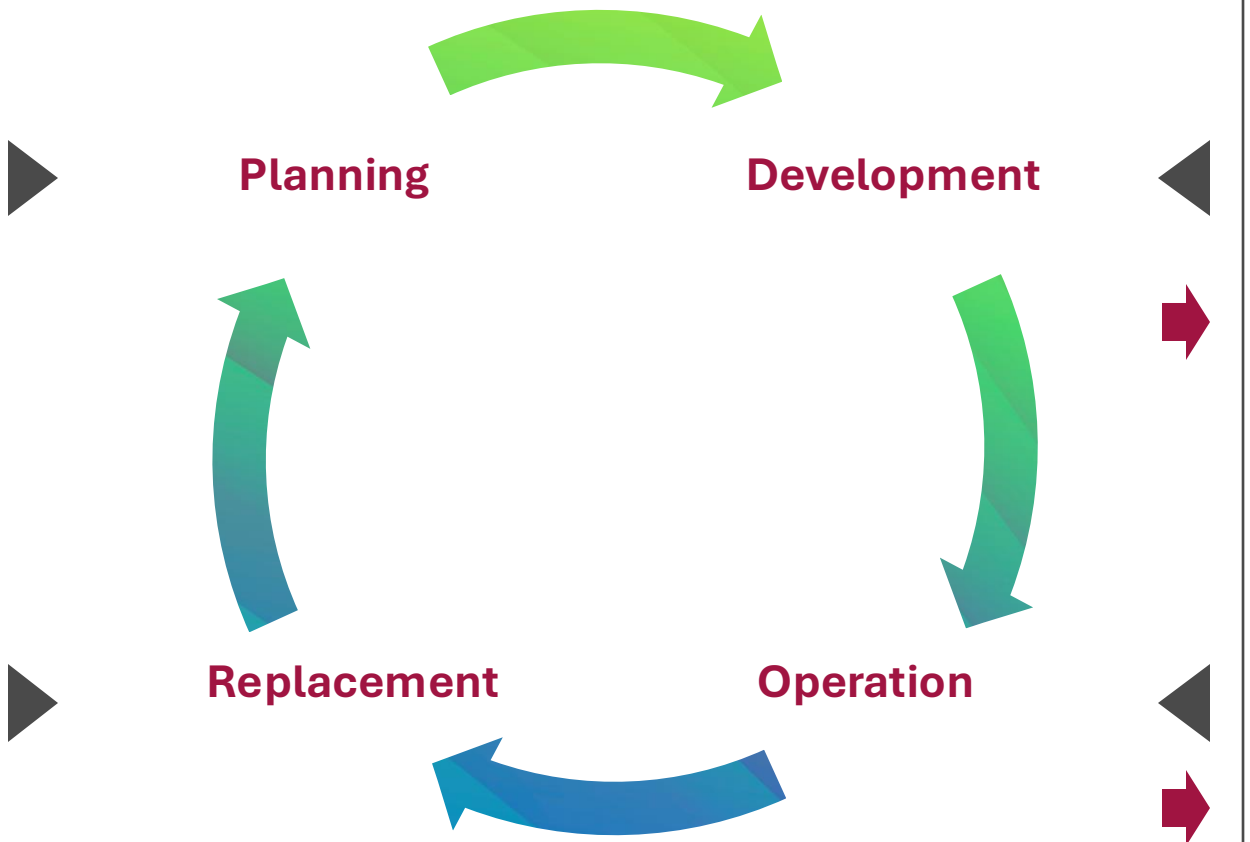
Planning

- Feasibility study / business case
- Rough concept / requirements analysis
- Make or buy decision
- Technical concept / IT concept, design

Frequently neglected cost drivers

Replacement

- **Waste disposal**
- **Data backup**
- **Data migration**



Development / Introduction

- Implementation / Customizing
- Test / Integration
- Hardware, Software
- SLA creation
- Commissioning / Roll-out
- **Organizational changes**

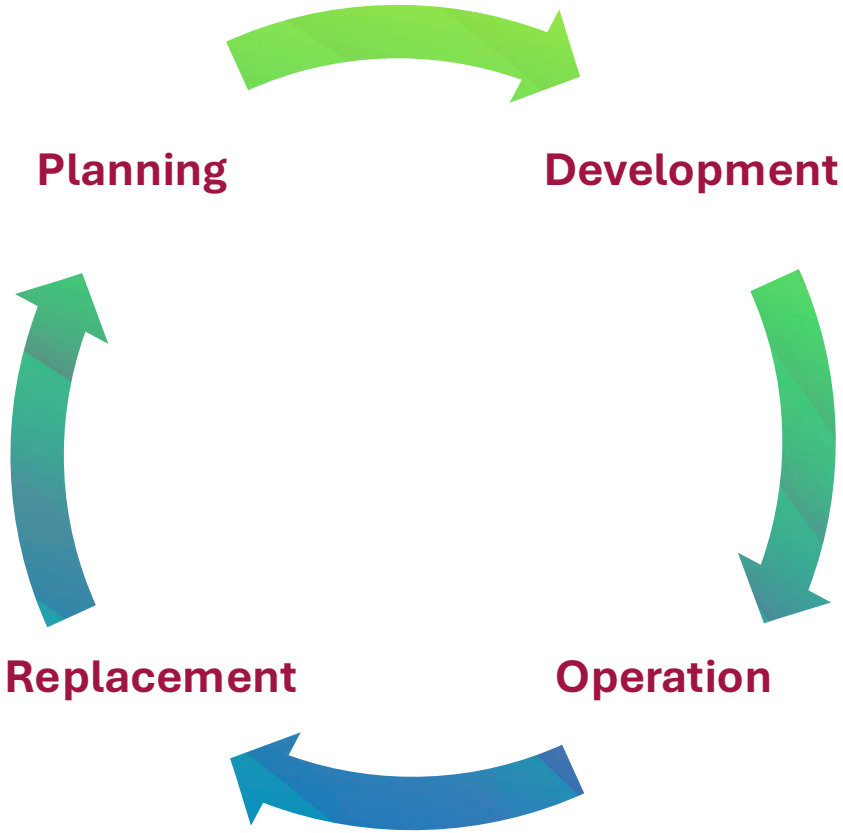
Operation

- Adjustments
- Maintenance and care
- Operating resources
- **Further development**
- **Release change**

Cost allocation for IT projects

60-80 %

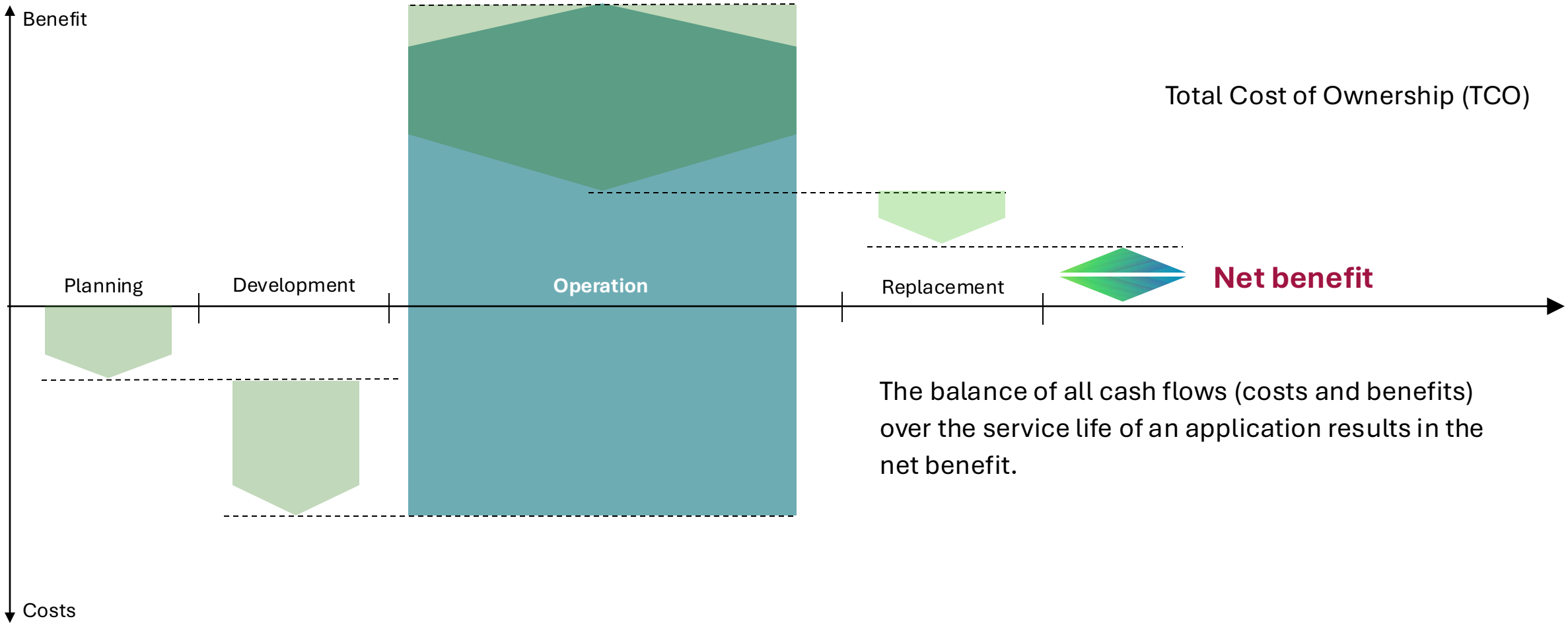
of the costs are incurred during the operation of the application.



20-40 %

of the costs for a typical IT solution are attributable solely to the creation of the application.

Costs and benefits over the software life cycle



The balance of all cash flows (costs and benefits) over the service life of an application results in the net benefit.

Software engineering in industrial practice
Project management: economic efficiency

1. Basics and definitions of terms
2. Economic efficiency of IT projects
3. **Literature**

Literature

- Bernotat J., Stein J., “10 Tipps & Tricks zum Business Case “, GPM-Magazin PMAktuell, 2/2007, S. 43-47
- Stein J, “Mit dem Business Case Wirtschaftlichkeit von Projekten nachweisen – der Business Case sichert den Erfolg von IT-Projekten”, GI/ACM-Regionalgruppe, Karlsruhe, 25.09.2007
- Brugger R., “Der IT Business Case“, Springer, 1. Aufl., 2005

Large-Scale Project Management

Cost estimation and calculation of major projects

Software engineering in industrial practice
Project management: Effort estimation

1. Basics and definitions of terms
2. Bottom-up estimate (expert estimate)
3. Top-down estimation (use case points)
4. Literature

Software engineering in industrial practice
Project management: Effort estimation

1. **Basics and definitions of terms**
2. Bottom-up estimate (expert estimate)
3. Top-down estimation (use case points)
4. Literature

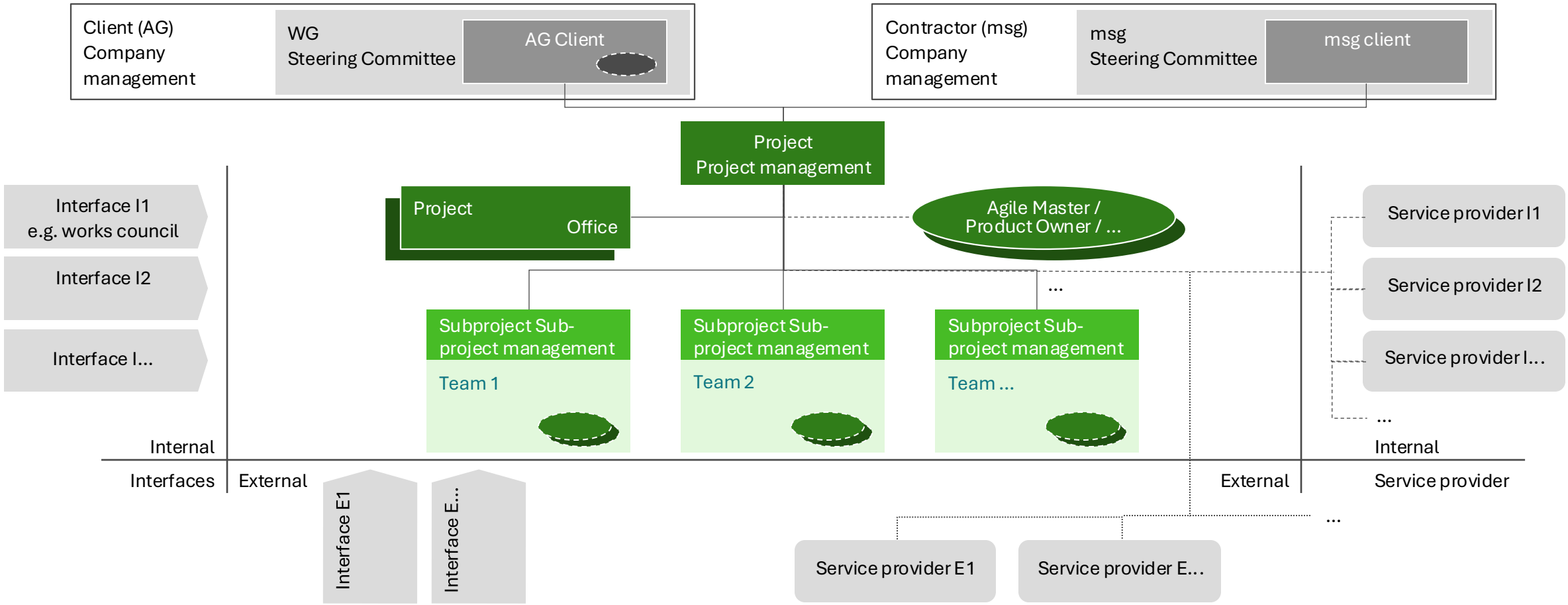
When projects become "big"!
What is the difference between a project and a major project?

- Project and major projects have the same definition
  - A project is a "project that is essentially characterized by the uniqueness of the conditions as a whole, e.g. objectives, time, financial, personnel or other limitations, project-specific organization.[1]
- Projects and major projects differ "only" in the characteristics of some criteria, usually the high quantity of objectives and/or delivery results (specification) and the scope in terms of time and costs.

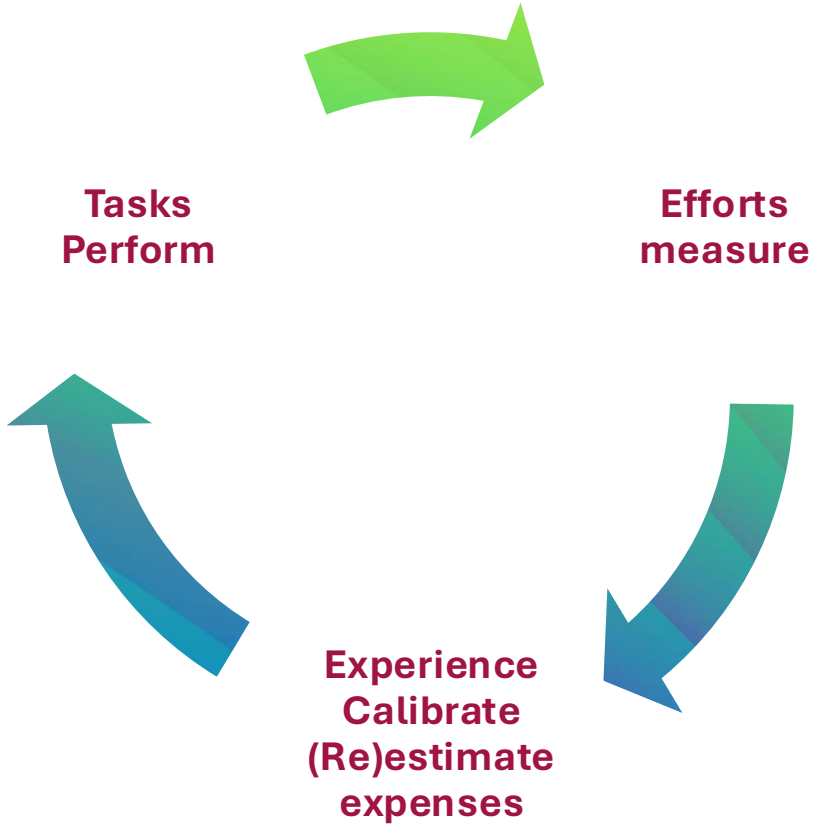
Project (IPMA Checklist)	Major project
Unique and innovative	"Unique", there are fewer comparable projects
Limited in time	Long term (> 2 years)
Interdisciplinary cooperation	And many disciplines
Defined responsibility for results	Project management, possibly co-project management
Complexity	Very high complexity
Target	Targets extensive, unstable
Limited resources	High quantity of resources

1. Definitions: GPM (E-Book) | PM3 | DIN 69901-5 (DIN, 2009c)

Internal and external interfaces and service providers must be considered in the same way as dual clients and other (agile) roles!

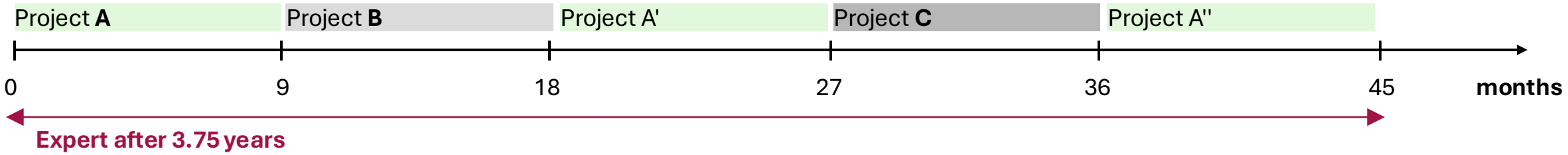


Cost estimates are always based on practical experience and intuition

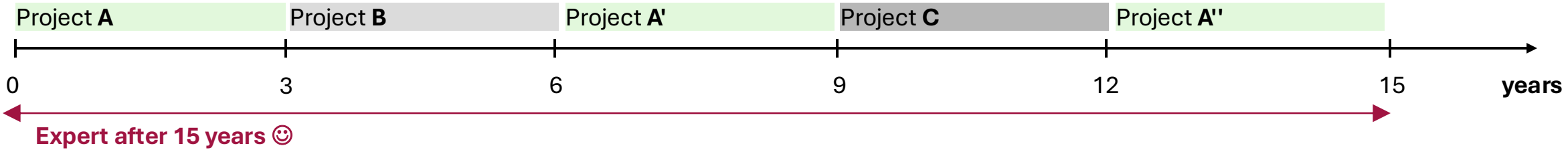


The limits of intuition are reached in large-scale projects

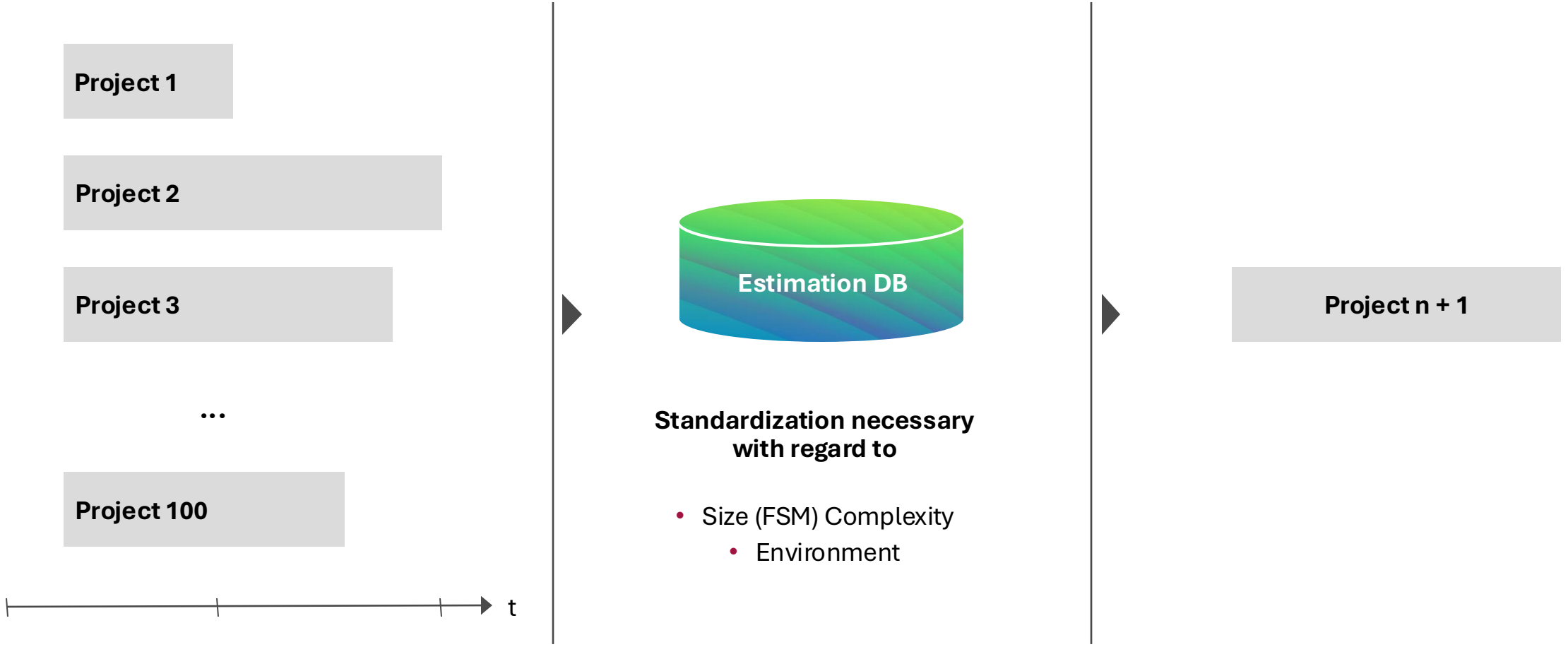
- Expert estimates are based on the experience of experts:
  Each element of the parts list is individually estimated by the expert
- Expert: At least 3 x a comparable task/project carried out by yourself
- Assumption: **a typical (small) project** takes 9 months:



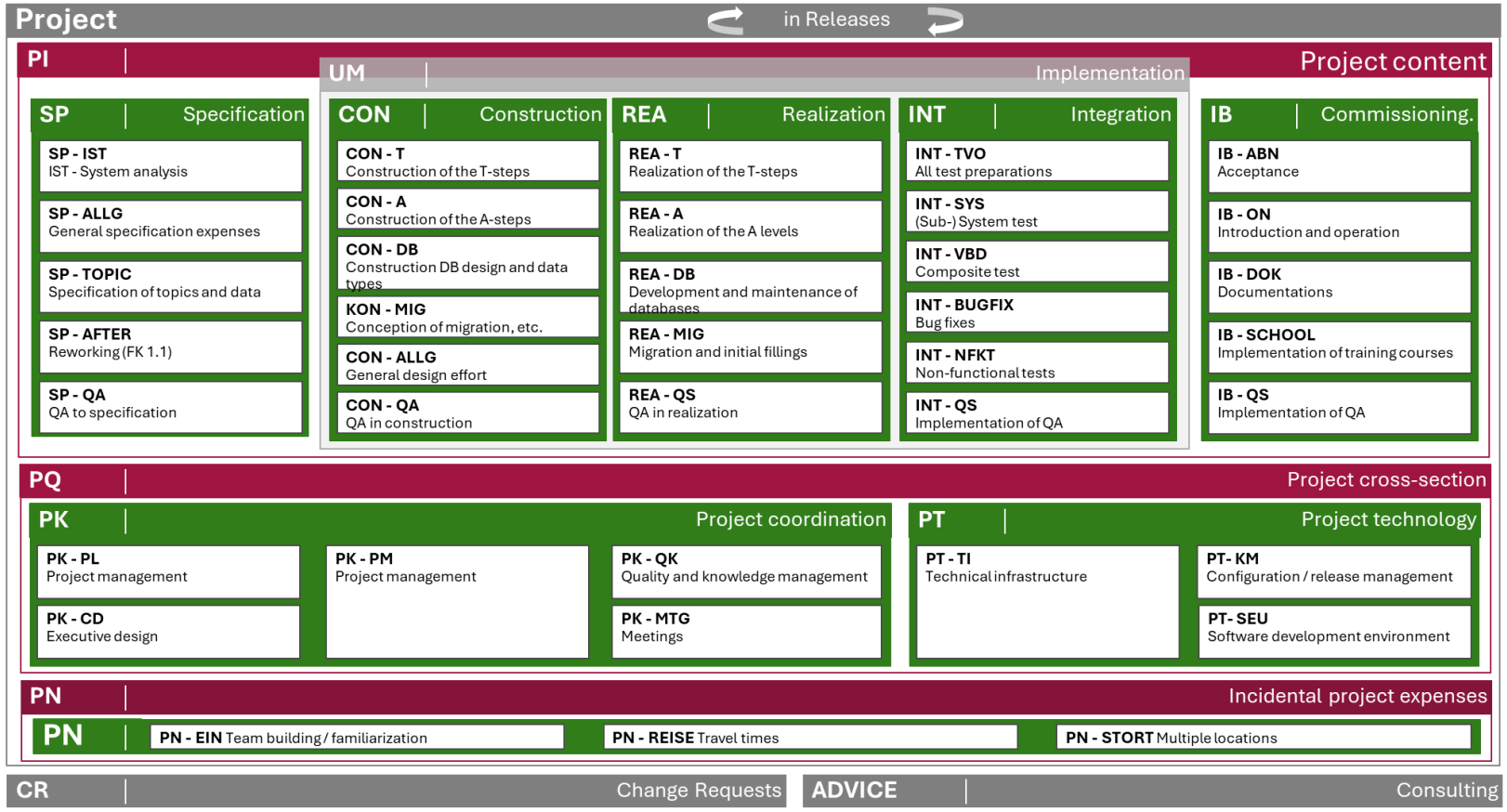
- Assumption: a **major project** or **program** lasts 3 years:



Estimation databases with FSM (Functional Size Measurement) overcome the limits of intuition in large projects



The effort model1 structures project activities according to task categories
→ All activities in a project can be clearly assigned!



E0 | Level 0

Used to collect key figures and evaluations

E1 | Level 1

Structures the task categories from level 2 according to subject areas

E2 | Level 2

Defines the task categories of a project. All activities in a project fall into one of these task categories.

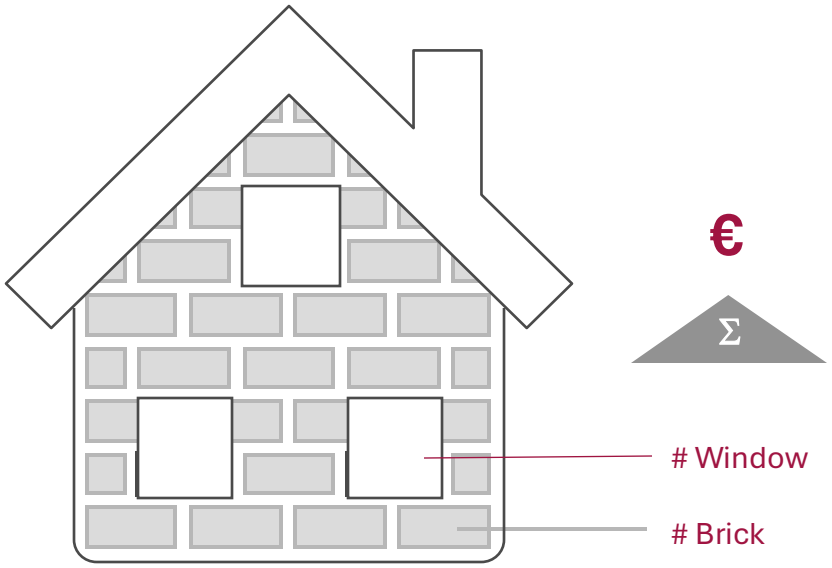
UM | Implementation

Serves to group the subject areas belonging to the implementation

1. Source: Dissertation "Use Case Points 3.0" by Dr. Stephan Frohnhoff, University of Paderborn, 2009

We distinguish between bottom-up and top-down budget methods

Bottom-Up



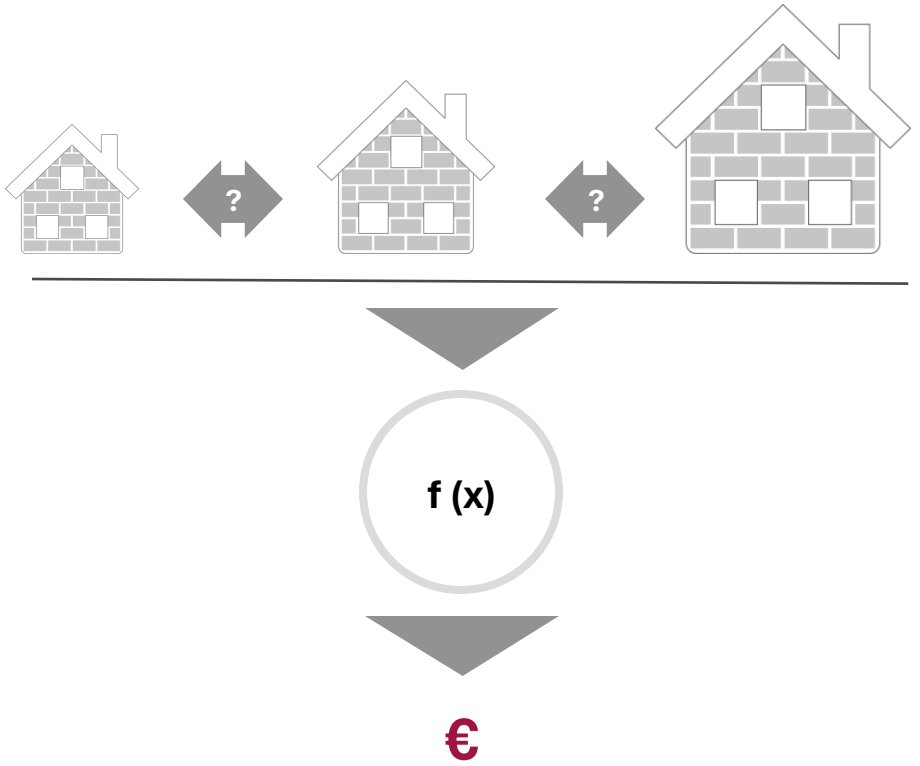
Specification fication

1

Conversion implemen-tation

2

Top-Down



Software engineering in industrial practice
Project management: Effort estimation

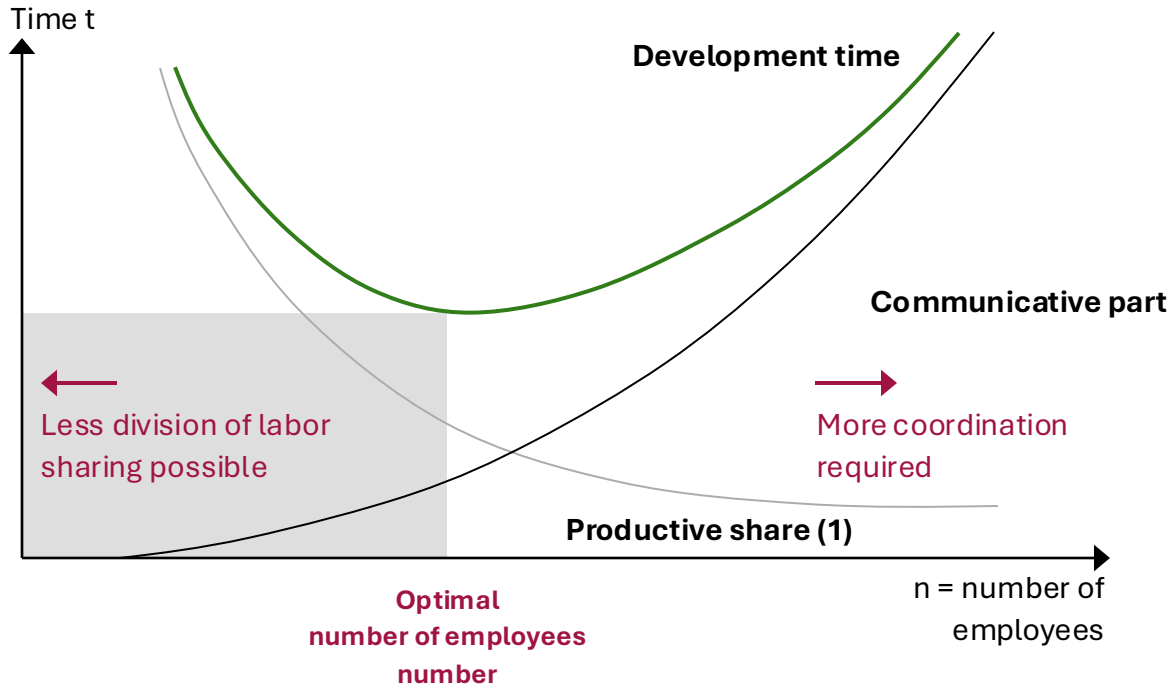
1. Basics and definitions of terms
2. **Bottom-up estimate (expert estimate)**
3. Top-down estimation (use case points)
4. Literature

The key figures of the expense model are used to check the plausibility of an estimate

Kennzahlenplausibilisierung						
Kennzahlen		Schätzung	Erfahrungswerte aus Aufw.-Modell (~1σ-Bereich) von	bis	Median	Kommentar
SP	/ PI	0%	8%	28%	19%	Detailspezifikation fertig bzw Rest nach Aufwand bis v1.0
KON	/ UM	13%	9%	25%	17%	
REA	/ UM	53%	35%	65%	52%	
INT	/ UM	34%	17%	40%	32%	
INT-BUGFIX	/ UM	8%	5%	19%	13%	
PK	/ PI	45%	15%	40%	28%	
PK-PL	/ PI	16%	6%	18%	12%	PL & Entwicklungsleitung, daher mehr als normal
PK-PM	/ PI	3%	2%	6%	4%	
PK-CD	/ PI	12%	4%	12%	8%	
PT	/ PI	17%	3%	10%	6%	DevOps sind mit eingeplant (zusätzliche Resource)
PN-EIN	/ PI	1%	2%	7%	4%	
QS	/ PI	0%	3%	8%	5%	wurde nicht separat geschätzt, sollte aber abgedeckt sein

- We define a key figure as any quotient from two task categories of the expense model. This means that each key figure has a clear meaning in terms of content, which is a prerequisite for the company-wide use of key figures.
- The key figure plausibility can be plausilized,
  - all expense categories were comprehensively covered in the estimate.
  - the distribution of the workload appears to make sense based on experience from previous projects.
- The empirical values for percentage distribution only serve as guidelines, as each project is individual. However, major deviations should at least be questioned and justified.

As an initial guide to team size and project duration is Brook's rule of thumb



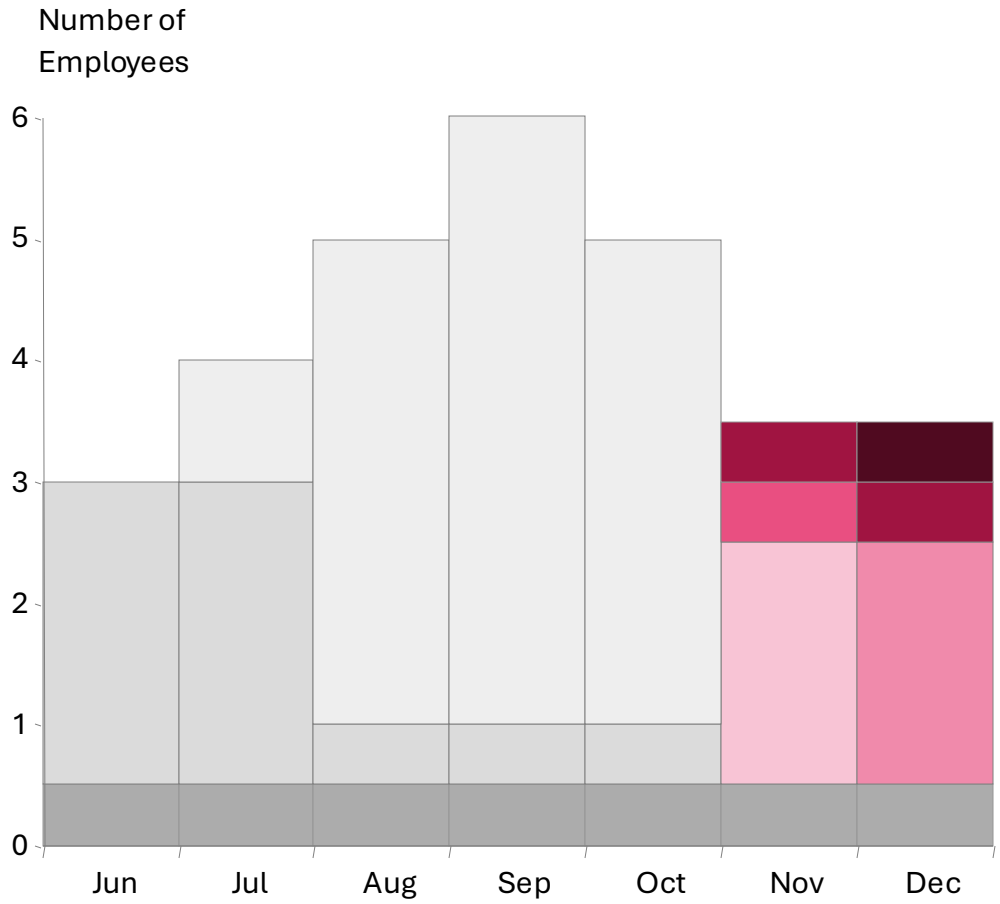
Optimal team size ~ $\sqrt{\text{Expenditure in PM}}$

"The man-month as a measure of the amount of work is a dangerous and misleading myth. The term wants to make us believe that agents and months are interchangeable factors"

Fred Brooks in "The myth of the man-month"

The cost estimate is checked for plausibility using an employee checked for plausibility

- Outline the project schedule with estimated duration and team size
- Calculate area, here: 30 time months (ZtM)
- 1 ZtM = 0.8 BM due to public holidays, training, illness, meetings, etc.
- Here the conversion from ZtM to PM results in: 30 * 0.8 = 24 PM
- Does this match the cost estimate?



From the number of employees and the total effort the project duration can be determined

In this example, the total expenditure of 104 PM was spread over 14 months:
Maximum 11 employees, average 8.9 employees or 7.4 PM, team structure and maximum team size are reasonable.

Expenditure [BM] (10/12)	4,2	6,5	7,3	8,6	9,4	9,4	9,4	9,4	9,4	8,5	8,5	8,1	3,9	1,7
Cumulative expense	4,2	11	18	27	36	45	55	64	74	82	91	99	103	104

Software engineering in industrial practice
Project management: Effort estimation

1. Basics and definitions of terms
2. Bottom-up estimate (expert estimate)
3. **Top-down estimation (use case points)**
4. Literature

Software engineering in industrial practice
Project management: Effort estimation

1. Basics and definitions of terms
2. Bottom-up estimate (expert estimate)
3. Top-down estimation (use case points)
4. **Literature**

Literature

- https://www.msg.group/images/msggroup/services/techrefresh/Dissertation_Use_Case_Points_3.0_Frohnhoff_msg.pdf → Search for "UCP"
- Balzert, H.: Textbook of Software Engineering, Volume 1, Software Development. Spektrum Akademischer Verlag, 2nd edition, 2000.
- Siedersleben, J.: "Softwaretechnik - Praxiswissen für Software-Ingenieure" 2nd revised and updated edition, Hanser Verlag, 2003.
- Frohnhoff, S.; Jung, V.; Engels, G.: "Use Case Points in der industriellen Praxis" In "Applied Software Measurement - Proceedings of the International Workshop on Software Metrics and DASMA Software Metrik Kongress", Abran, A. et al. Eds. Shaker Verlag, 2006, pp. 511-526
- Cockburn, A.: "Writing Effective Use Cases", Addison-Wesley, 2001.
- Smith, J.: "The Estimation of Effort Based on Use Cases", Rational Software, Cupertino, CA.TP-171, October 1999. http://whitepapers.zdnet.co.uk/0,39025945,60071904p-39000629q,00.htm

Large-Scale Project Management

Effort Estimation

Software engineering in industrial practice
Project management: Effort estimation

1. Basics and definitions of terms
2. Empirical estimation methods (expert estimate)
3. Parametric estimation methods (counting parameters that determine the effort)
4. Literature

Software engineering in industrial practice
Project management: Effort estimation

1. **Basics and definitions of terms**
2. Empirical estimation methods (expert estimate)
3. Parametric estimation methods (counting parameters that determine the effort)
4. Literature

Why is effort estimation important?

As IT service provider, our customers expect us to make fixed price offers
The financial risk for the project has to be taken by the service provider

Dilemma

Estimate too low

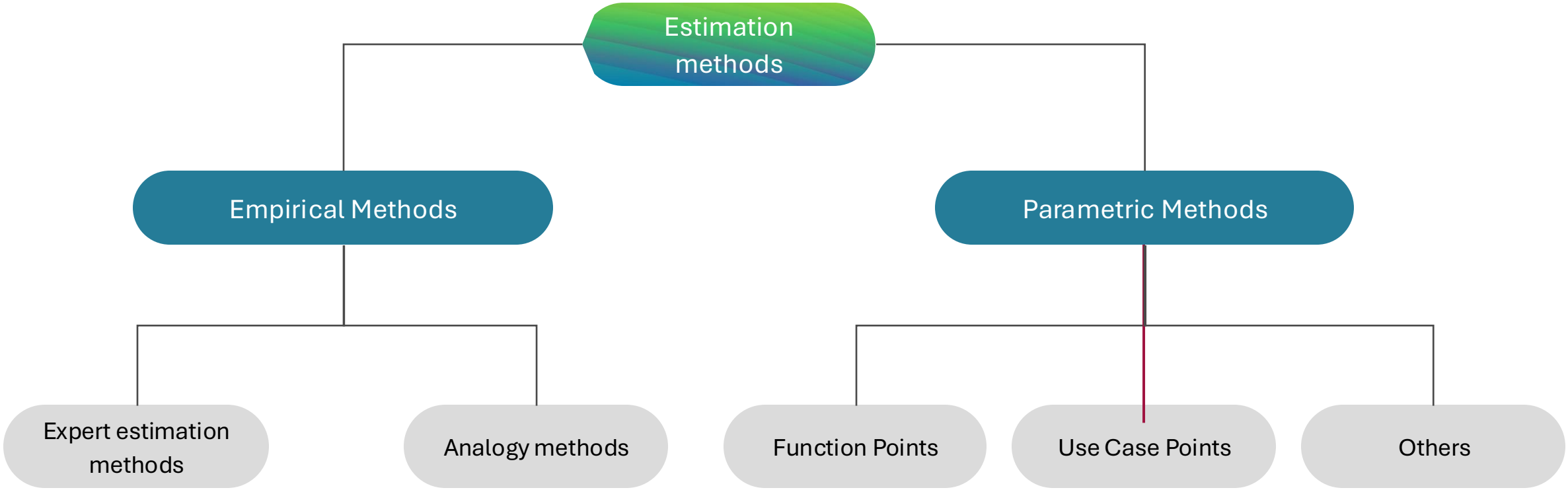
- Additional costs have to be taken by the service provider
- Investment opportunities are lost (work equipment, travel, training and development)
- Follow-up projects are at risk
- Overtime to meet deadlines

Estimate too high

- No longer competitive
- A competitor gets the contract
- The customer loses trust

Correct estimates are essential!

Estimation methods are divided into empirical and parametric methods



The estimation methods have different strengths and weaknesses.

Estimation method	principle	formality	Project-type	Domain knowledge
Expert estimate	data of completed projects, expert experience and intuition.	low / medium	**all**	high
Function Points	Counting and evaluating system size via interfaces, queries and data; subsequent formula-based conversion into effort	high	development	**medium**
Use Case Points	Counting and evaluating system size via use cases (scenarios, steps, interaction elements); subsequent formula-based conversion into effort	high	development	**medium**
Analogy method	Use of actual values of comparable projects/topics and evaluation of the differences, e.g. as large as 1.5 times project/topic XYZ	medium	**all**	high

Formality (How well defined ist the method):
low – no or only few specifications
medium – additional defined procedure
high – specifications and procedure are well defined

Project type:
Development: most of the functionality has to be newly developed
Enhancement: add functionality to an existing program
Maintenance: troubleshooting change existing functionality

All estimation methods require similar information

Estimation method	Comparison with other projects	Brakedown (Domain requirements)	Decomposition of technical design	Experience in other projects	Environment
Expert estimation methods	X	X	X	X	X
Function Points	X	X	X		X
Use Case Points	X	X	X		X
Analogy methods	X			X	X

Software engineering in industrial practice
Project management: Effort estimation

1. Basics and definitions of terms
2. **Empirical estimation methods (expert estimate)**
3. Parametric estimation methods (counting parameters that determine the effort)
4. Literature

Analogy methods – Principles

Basic idea

Estimate the effort by comparing with a similar project.
Prerequisite: The project can be broken down into parts or activities, and both the actual effort and a metric (LOC, number of tables, number of features, etc.) are known.

Procedure.

For each function, use case, or activity, a suitable size measure is determined (e.g., 50 features) and compared with the measure of the analogue project (e.g., 75 features & 130 PT actual effort). The effort is calculated by using the ratio (in this example, 0.67 * 130 PT = 87 PT; 0.67 = 50 features/75 features)

Characteristics

Accuracy	depends on the Quality of the analogy
Effort for the Estimation	high
Follower effects	doesn‘t exist
When to use	if the prerequisites are met

Examples for analogy methods are shown in Module 02 (Three Point Estimation & Estimation Sizes).

Estimation meeting – increase the accurancy by increasing the persons involved and consolidate the individual estimations in an meeting

Basic idea	the estimate is prepared by several experts. In a joint meeting, the estimate, the relevant assumptions and the premises are discussed and consolidated.
Procedure.	Each developer prepares his estimate independently. In the meeting, all individual estimation items are discussed, and a consolidated value is determined.

Characteristics

Accuracy	high (depends on the expert)
Effort for the Estimation	medium
Follower effects	exist
When to use	medium and large projects

Hint: The essential added value is the open discussion to find misunderstandings or gaps. After the meeting, all estimators have a common solution approach

Software engineering in industrial practice
Project management: Effort estimation

1. Basics and definitions of terms
2. Empirical estimation methods (expert estimate)
3. **Parametric estimation methods (counting parameters that determine the effort)**
4. Literature

Parametric methods – Principles

Basic idea

Estimate the effort by counting the **functional factors (unadjusted)** and rating the **project conditions (adjusted)** that influence the effort. Prerequesite: The project can be broken down into the factors used by the estimation method.

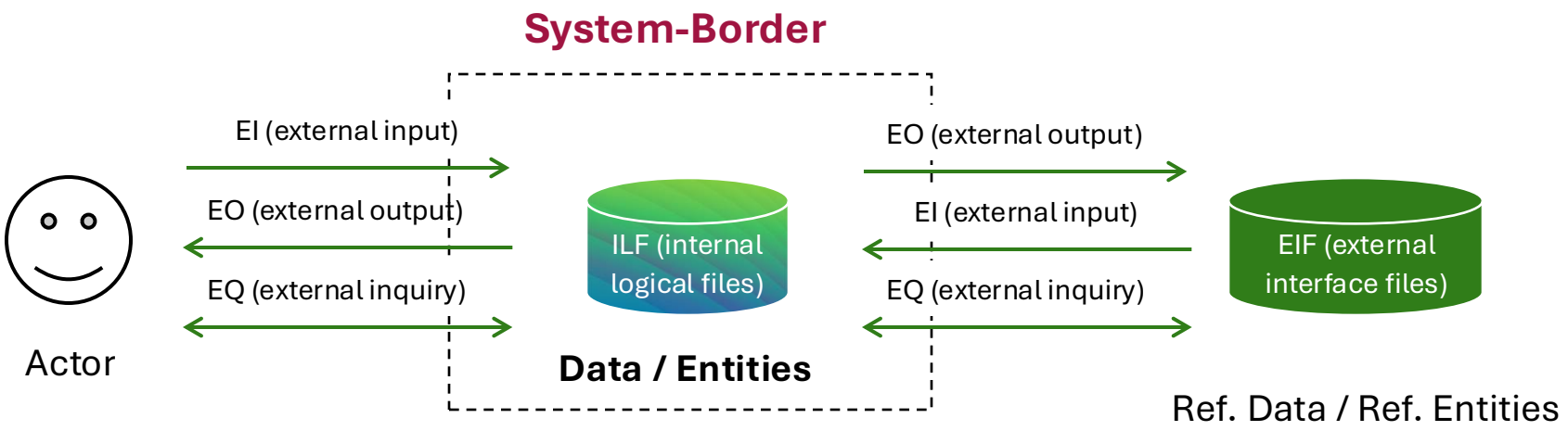
Procedure.

Each functional factor, defined by the estimation method, is counted and valued as in the methodology described. For the project conditions a questionnaire with a determined rating is proposed. The result is then converted into effort.

Characteristics

Accuracy	depends on the experience with the methodology and how good it fits the Requirements description
Effort for the Estimation	medium
Follower effects	doesn‘t exist
When to use	if the prerequisites are met

Functional factors of a Function Point estimation (Unadjusted Function Points)



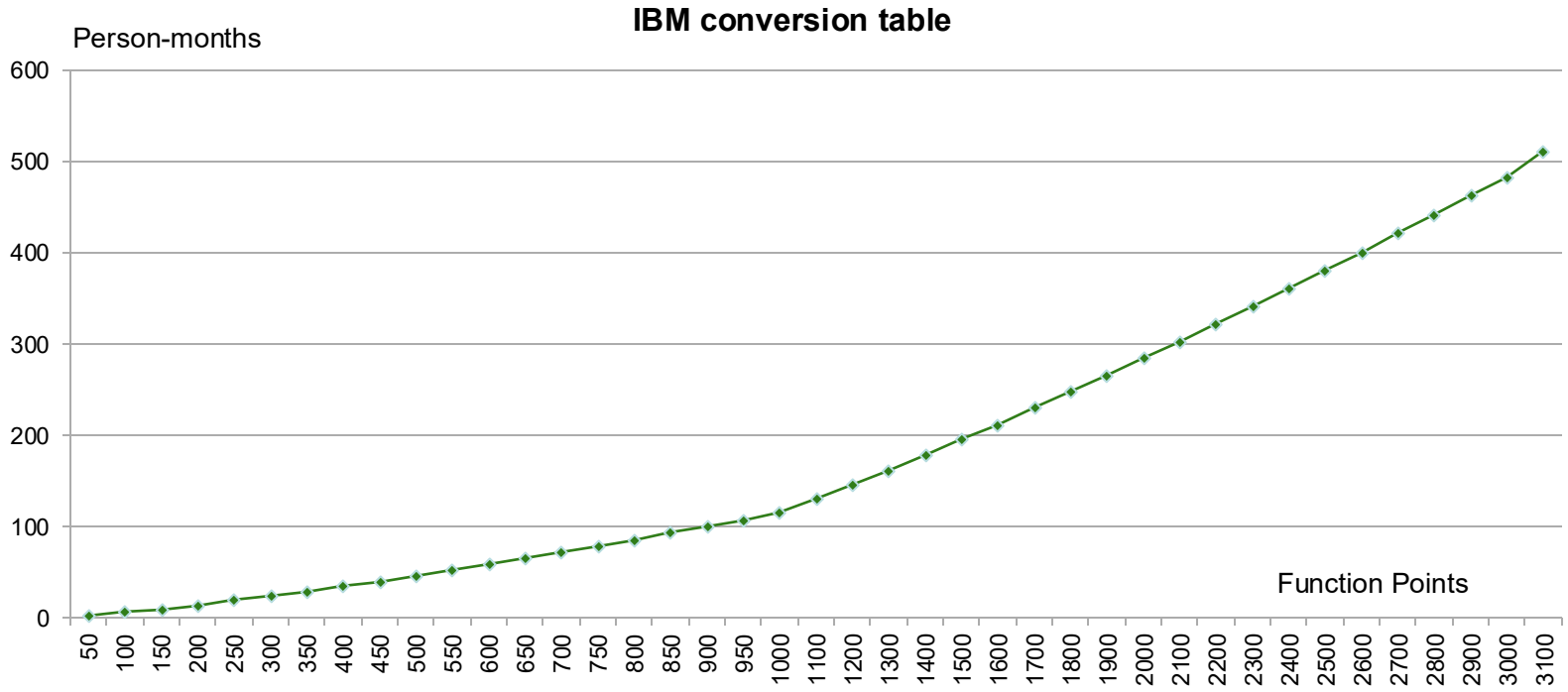
Input: EI (Inputs for the internal logical Files, ILF)
Output: EO (Output of calculated / transformed data)
Inquiries: EQ (Input: question, Output: data)

Conditional factors to calculate Adjusted Function Points

Characteristics	Description
Data Communications	How many communication facilities are there to aid in the transfer or exchange of information with the application or system?
Distributed Data Processing	How are distributed data and processing functions handled?
Performance	Did the user require response time or throughput?
Heavily Used Configuration	How heavily used is the current hardware platform where the application will be executed?
Transaction Rate	How frequently are transactions executed daily, weekly, monthly, etc.?
On-Line Data Entry	What percentage of the information is entered online?
End-user Efficiency	Was the application designed for end-user efficiency?
Online Update	How many ILFs are updated by online transaction?
Complex Processing	Does the application have extensive logical or mathematical processing?
Reusability	Was the application developed to meet one or many users needs?
Installation Ease	How difficult is conversion and installation?
Operational Ease	How effective and/or automated are start-up, back-up, and recovery procedures?
Multiple Sites	Was the application specifically designed, developed, and supported to be installed at multiple sites for multiple organizations?
Facilitate Change	Was the application specifically designed, developed, and supported to facilitate change?

Effort: From FP to PM – Empirical Method

Conversion of Function Points into person-months



In practice Unadjusted Function Points are useful for:

- **Validating** an empirical estimation
- **Benchmarking**, e.g. development time / FP, Costs / FP

The methodology is further developed by the International Function Point User Group (IFPUG)

The Use Case Points (UCP) method is based directly on a use case-based specification and is very easy to apply

Use Case	Com-plexity		Use case weight
Manage order	high	→	5
Manage customer	simple	→	1
Manage product	medium	→	3
...	...		...

Σ Use case weights

Actuator	Type		Actuator weight
Master data	Neighboring system (API)	→	2
Business partner	Neighboring system (protocol)	→	2
Retailer	User interface	→	3
...	...		...

Σ Actuator weights

\+

M-factor x **T-factor** x **A-factor** **= Adjusted use case points**

Questionnaire with Cost factors

Questionnaire with Cost factors

Use Case Points

Productivity factor

Effort across all phases[1]

ABC Individual analysis

→ Calculation according to standard metric (simple, medium, complex)

⬟ Calculation according to proprietary metric

1) according to mapping to expenditure model

Factors to calculate the Adjusted Use Case Points

Management-Factors

- Business Architect Points: in a range from 1 (very good) to 5 (poor), Weight: 3.6
- Technical Architect range from 1 (very good) to 5 (poor), Weight: 3.6
- Team collaboration range 1 (very good) 3 (typical) 5 (poor), Weight: 4
- Team continuity range 1 (very good) 3 (typical) 5 (poor) , Weight: 2,7
- Quality of the specification range 1 (very good) 3 (typical) 5 (poor), Weight: 2
- Process overhead range 1 (optimal) 3 (typical) 6 (very complex) , Weight: 2.5
- Tight Deadlines range 1 (normal) 3 (tight) 5 (very tight) , Weight: 1
- Stability of requirements 1 (stable) 3 (typical) 5 (unstable) 6 (very unstable), Weight: 1.3
- Number of contact persons 1 (few) 3 (typical) 5 (many), , Weight: 1
- Integration into existing systems 1 (few) 3 (typical) 5 (many), Weight: 4.3
- Project maturity level 1 (follow-up project) 3 (new project, known environment), 3 (new project, unknown environment), Weight: 1

In practice unadjusted Use Case Points are useful for

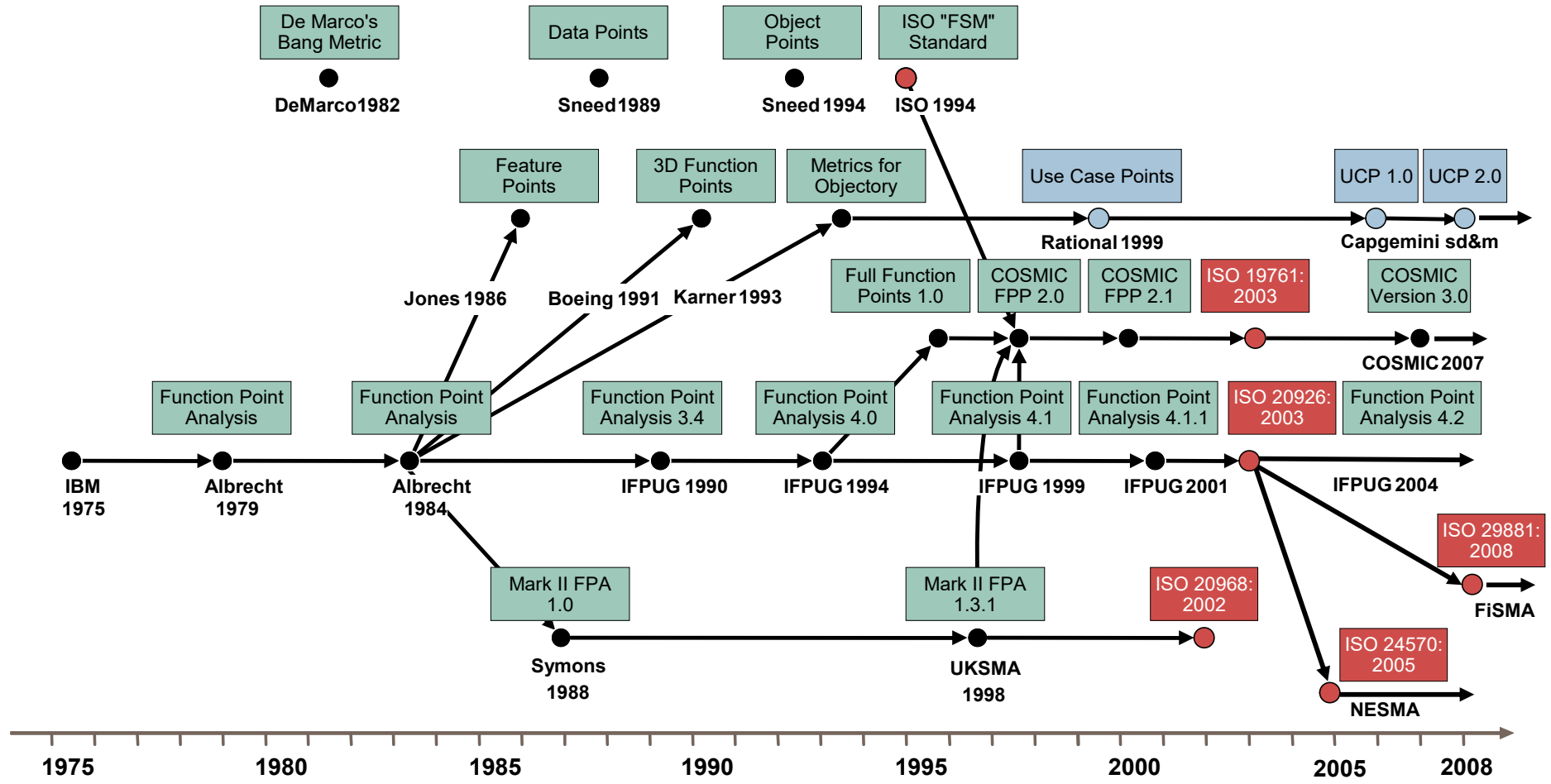
- **Validating** an empirical estimation,
- especially if the **Requirements are** documented as **Use Cases**.

The method is not widespread and is not further developed.

Technical-Factor Calculation

- GUI complexity from 1 (no GUI) to 5 (highly integrated), Weight: 5
- Adaptability and configurability from 1 (not needed) to 5 (highly adaptable and configurable), Weight: 5
- Installation and Deployment range from 1 (very good) to 5 (very complex requirements), Weight: 2
- Operational requirements range from 1 (very good) to 5 (very complex requirements), Weight: 2
- Distributed system range 1 (no requirements) 3 (typical) 5 (highly distributed System), Weight: 2
- Performance and load requirements range from 1 (no requirements) to 5 (very complex requirements), Weight: 4
- Availability requirements range from 1 (no requirements) to 5 (very complex requirements), Weight: 2
- Security requirements range from 1 (no requirements) to 5 (very complex requirements), Weight: 2
- Reuse requirements 1 (no requirements) 2 (few) 3 (limited) 5 (complex requirements, e.g. company wide framework), Weight: 4
- Quality and flexibility of the development environment 1 (very flexible) 3 (typical) 5 (limited flexibility), Weight: 6
- Quality of the technical architecture 2 (suitable) 3 (potentially not suitable) 4 (limited suitable) 5 (not suitable), Weight: 5
- Type of programming 3 (modern languages % tools) 3 (below 50% mainframe) 4 (between 50 to 75% mainframe) 5 (100% mainframe), Weight: 4

Development of functional size measurement



Source: Lother, M.; Dumke, R.: Points Metrics - Comparison and Analysis. in: Dumke et al (Eds.): Current Trends in Software Measurement - Proceedings of the 11th IWSM. Proceedings of the 11th IWSM, Montréal, Shaker Verlag. Aachen. pg: 228-267. 2001; supplemented by S. Frohnhoff, sd&m AG

Software engineering in industrial practice
Project management: Effort estimation

1. Basics and definitions of terms
2. Empirical estimation methods (expert estimate)
3. Parametric estimation methods (counting parameters that determine the effort)
4. **Literature**

Literature

Balzert: Lehrbuch der Softwaretechnik 3. Auflage, Hanser 2009

Tiemeyer: Handbuch IT-Projektmanagement, 3. Auflage, Hanser 2014

Sneed, Harry: diverse Veröffentlichungen

- Software in Zahlen, Hanser 2010
- Software Projektkalkulation, Hanser 2005
- Aufwandschätzung v. Software Reengineering (in Wirtschaftsinformatik 45/2003)

Poensgen: Function-Point-Analyse, 2. Auflage , dpunkt-Verlag 2012

IFPUG Function Point Counting Practices Manual, Release 4.3.1, 2010

IFPUG Function Point 4.3 – Quick Reference

www.ifpug.org – International Function Point User Group

www.defpag.org – Deutschsprachige Function-Point-Anwendergruppe

Large-Scale Project Management

Project Controlling

Software engineering in industrial practice
Project management: Projectcontrolling

1. Earned Value Method – Definitions of terms
2. Earned Value Method - Example
3. Milestone Trend Analysis: Definition of terms
4. Milestone Trend Analysis - Example
5. Literature

Software engineering in industrial practice
Project management: Projectcontrolling

1. **Earned Value Method – Definitions of terms**
2. Earned Value Method - Example
3. Milestone Trend Analysis: Definition of terms
4. Milestone Trend Analysis - Example
5. Literature

What does Projectcontrolling mean?

Project controlling serves to ensure the achievement of all project objectives through :

- actual data recording,
- Target vs. actual comparison,
- analysis of deviations,
- evaluation of deviations with possible correction suggestions,
- planning of measures, and
- control of the implementation of measures.

(according to DIN 69901-5 3.5461)

Project controlling is part of the discipline Project management

Main KPIs in Projectcontrolling

Effort: The working time required for a project.

Duration: measures the duration from project start to project completion.

Budget at completion (BAC): estimated total cost of a project based on the project plan.

Actual costs (AC): describes all actual costs incurred during the project.

Percent complete (PC): serves to represent the progress of a project or an individual task within a project.

Earned Value (EV): This value expresses the progress of a project relative to the budget.

Planned Value (PV): Planned costs for a defined reporting date (can be derived from the project-plan).

Schedule Performance Index (SPI): Measure of the temporal deviation of the performance achieved from the project-plan.

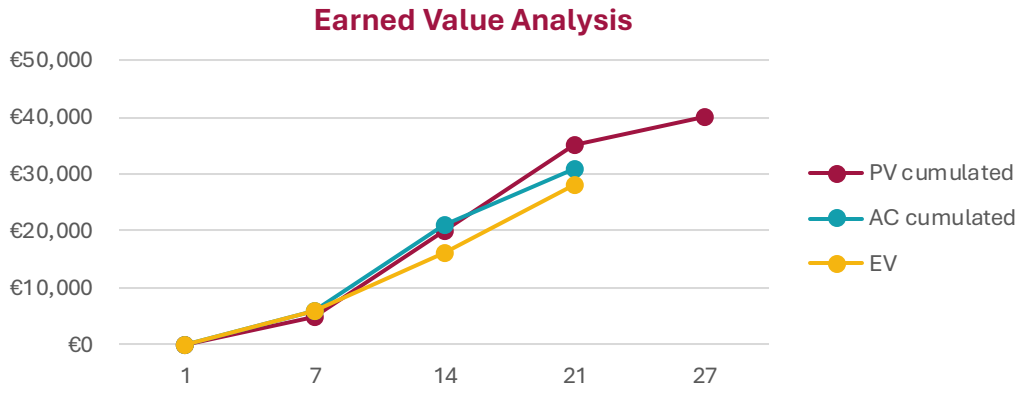
Cost Performance Index (CPI): Measure of the efficiency of previous performance.

Software engineering in industrial practice
Project management: Projectcontrolling

1. Earned Value Method – Definitions of terms
2. **Earned Value Method - Example**
3. Milestone Trend Analysis: Definition of terms
4. Milestone Trend Analysis - Example
5. Literature

Example for an Earned Value Analysis

Report Date	1	7	14	21	27
PV	0€	5.000€	15.000€	15.000€	5.000€
PV cumulated	0€	5.000€	20.000€	35.000€	40.000€
AC	0€	6.000€	15.000€	10.000€	0€
AC cumulated	0€	6000€	21.000€	31.000€	
PC	0%	15%	40%	70%	

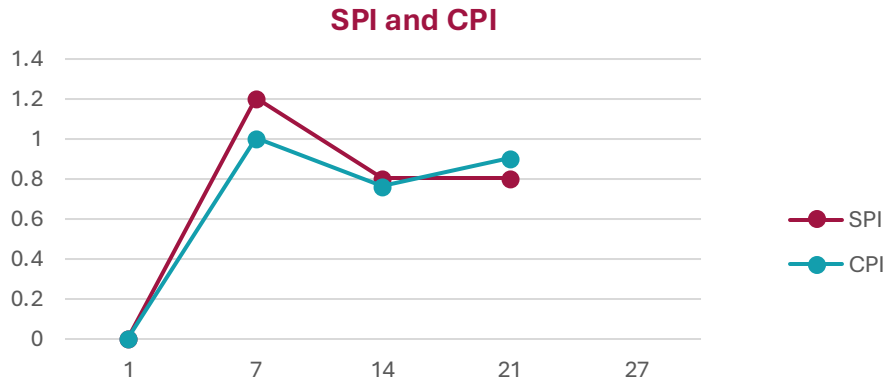


Calculation for reporting date 21

EV: BAC * PC = 40T€*70% = 28T€

SPI: EV / PV = 28T€/35T€ = 0,8

CPI: EV / AC = 28T€/31T€ = 0,9



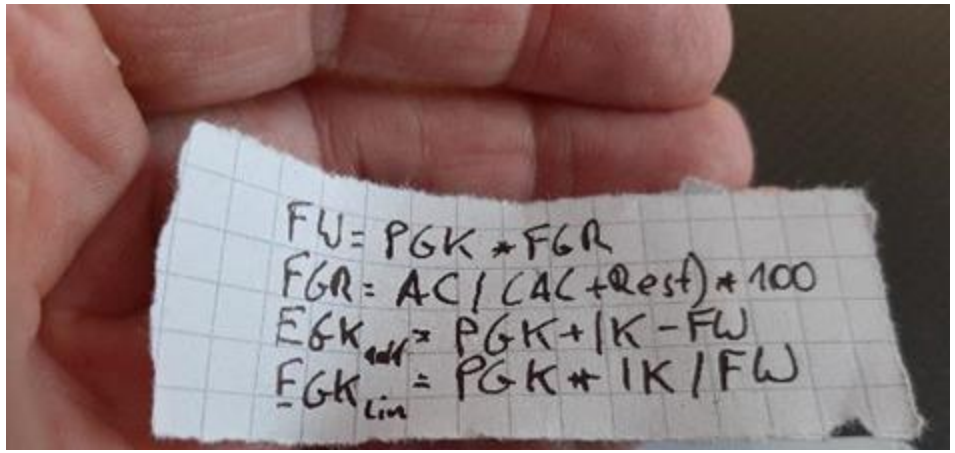
EVA Interpretation:

- AC > EV: project costs increased
- EV < PV: project is delayed

CPI & SPI Interpretation:

- CPI: Countermeasures are working, cost increase is now in the uncertainty range
- SPI: Delay stabilizes. Further countermeasures required.

How to get from effort to cost?



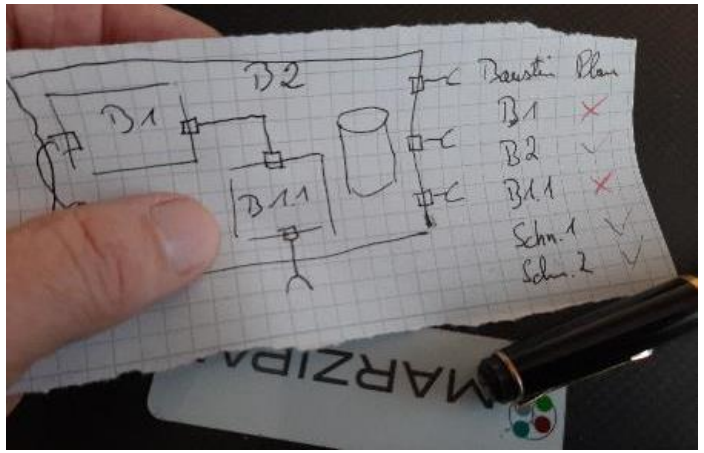
Simplification for determining costs per project day:

- Instead of the costs, the effort can be used (1 PT or 1h = 1€)
- Use of a fixed mixed cost rate (e.g. the mixed cost rate from the quotation).

\+ This makes it much easier to determine the EV.

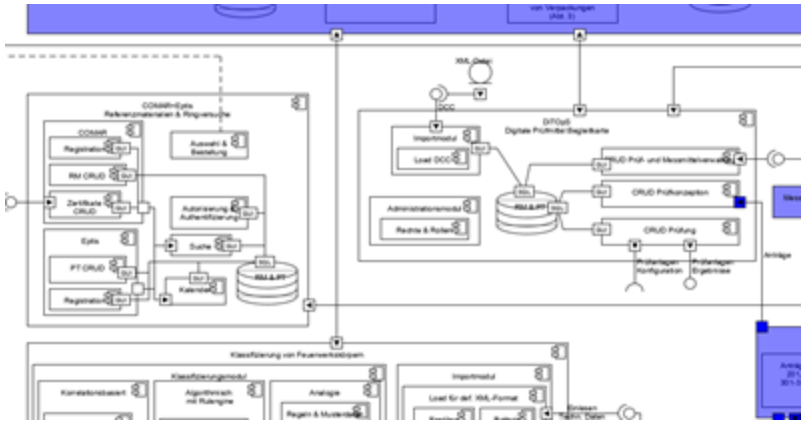
– Some Effects are no longer visible, such as the performance of tasks by cheaper or more expensive employees.

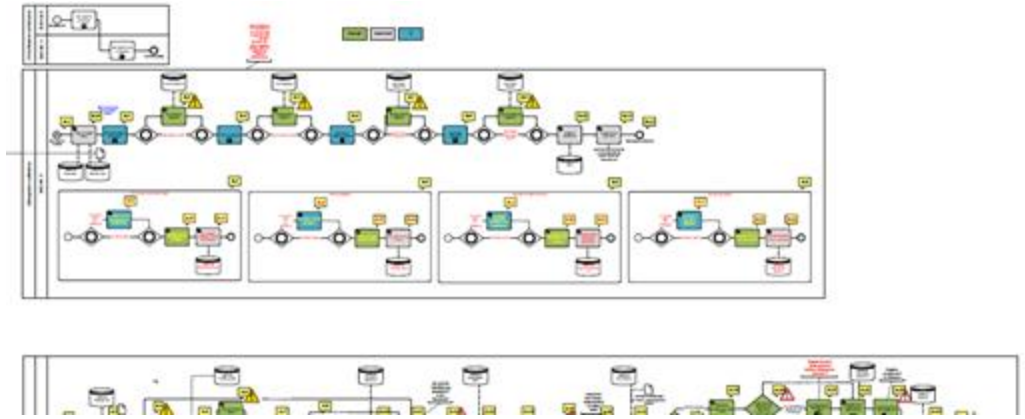
How to get to plan efforts?



Check the completeness of the backlog:

- Creating a rough building block diagram or a process diagram from the main topics in the specification
- Instead of estimates, use complexity levels or T-shirt sizes with traceable metrics.

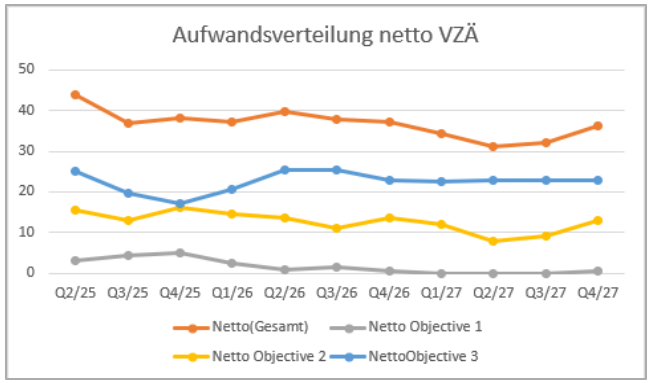




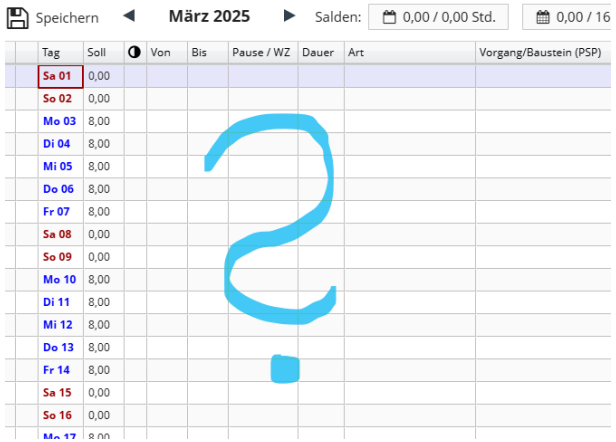
How to distribute effort across the timeline ?

Distribution of effort over the timeline, with unknown dependencies and unknown team

- equal distribution of effort over the project duration
- Project duration can be calculated using the CoCoMo formula:
  T = 2,5 * PM 0,35 (medium-sized projects)
  Hint: There are different formulas for easy, medium-sized and complex projects. In the operational praxis the formula for medium-sized projects is a good guess.
- Team size can be determined using Böhm's rule of thumb: sqrt(effort in PM)



What can be done if the actual costs are not recorded?



If the actual costs are not available, the following two approximation methods can be used:

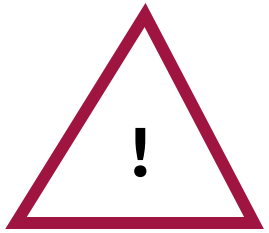
- Time-proportional, i.e. the number of FTE in the reporting period
- Proportional to progress, i.e. the difference between planned effort and remaining effort

---

\+ Both approximations help to determine whether the team velocity is sufficient for on-time completion.

– Effects such as insufficient team capacity or increased remaining effort are no longer visible.

When reality strikes …



- Only apply as few simplifications as possible at a time
- Replace simplifications as quickly as possible
- No subsequent changes in the controlling sheets

Gilberts Law:
when you take on a task, finding the best ways to achieve the desired result is always your responsibility

Software engineering in industrial practice
Project management: Projectcontrolling

1. Earned Value Method – Definitions of terms
2. Earned Value Method - Example
3. **Milestone Trend Analysis: Definition of terms**
4. Milestone Trend Analysis - Example
5. Literature

What are the main KPIs in Projectcontrolling?

Milestone Trend Analysis (MTA): The MTA shows the chronological progression of milestones in a project. It is a method to monitor project progress and to identify delays at an early stage.
Hint: Prerequisite for an MTA are clearly defined milestones with fixed deadlines and precisely defined results.

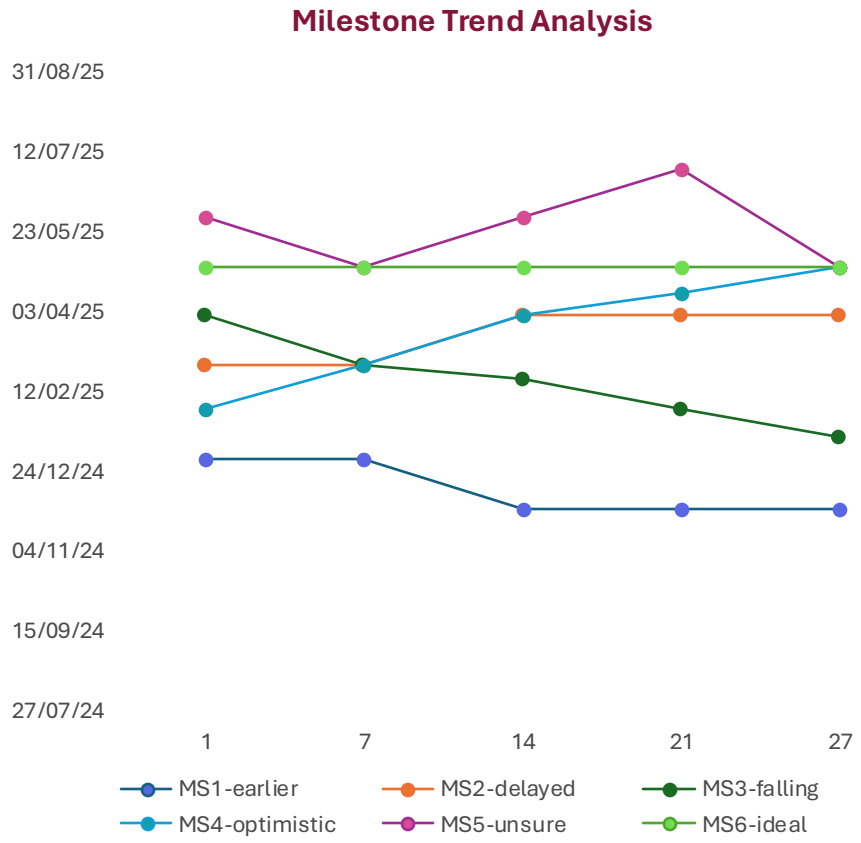
Milestone (MS): A milestone is an event of particular importance. A milestone consists of a unique identifier, a name, the results to achieve and a due-date.
Hint: The results must be defined so precisely that the achievement of the milestone can be clearly verified.

Software engineering in industrial practice
Project management: Projectcontrolling

1. Earned Value Method – Definitions of terms
2. Earned Value Method - Example
3. Milestone Trend Analysis: Definition of terms
4. **Milestone Trend Analysis - Example**
5. Literature

Example for an Milestone Trend Analysis

Report Date	1	7	14	21	27
MS1-earlier	01.01.25	01.01.25	30.11.24	30.11.24	30.11.24
MS2-delayed	01.03.25	01.03.25	01.04.25	01.04.25	01.04.25
MS3-falling	01.04.25	01.03.25	20.02.25	01.02.25	15.01.25
MS4-optimistic	01.02.25	01.03.25	01.04.25	15.04.25	01.05.25
MS5-unsure	01.06.25	01.05.25	01.06.25	01.07.25	01.05.25
MS6-ideal	01.05.25	01.05.25	01.05.25	01.05.25	01.05.25



MTA interpretation:

MS1-ealier: Milestone is reached earlier. Check whether follow-up tasks can be brought forward to keep the team busy.

MS2-delayed: The Milestone will be reached later. Milestone will be reached later. Dependent tasks need to be reviewed.

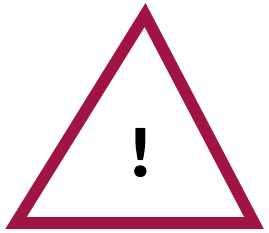
MS3-falling: The Milestone is reached earlier and earlier. Check whether planning has included too much buffer.

MS4-optimistic: consecutive delayed Milestone. Check weather the planning was to optimistic.

MS5-insecure: The zigzag pattern indicates a high degree of uncertainty regarding the deadlines. It is possible that problems have been identified, and countermeasures have been successfully implemented.

MS6-ideal: Milestone will be reached as planned.

When reality strikes …



- Very simple method, no tools or cheats required
- Easy health check for the project schedules
- The curve is important for the MTA interpretation

Parkinsons Law:
work expands to fill the time allotted for its completion

Software engineering in industrial practice
Project management: Projectcontrolling

1. Earned Value Method – Definitions of terms
2. Earned Value Method - Example
3. Milestone Trend Analysis: Definition of terms
4. Milestone Trend Analysis - Example
5. **Literature**

Literature

Thank you very much

Christian Schmitz
+49 (170) 9241329
christian.schmitz@msg.group

Dieter Ebhart
+49 (160) 7195378
dieter.ebhart@msg.group

msg for banking ag
Amelia-Mary-Earhart-Straße 14
60549 Frankfurt a. M.

+49 69 580 045 – 0

info-banking@msg.group
www.msgforbanking.de