

# Das Manifest agiler Architekturarbeit

Stefan Toth

Software Architecture Forum – 16.06.2026, München



In der Eile verlieren wir am Flughafen „Eli“ – DAS Kuscheltier meines Sohnes...

40 onL  
40onL Apartments  
to Stefan  
Today at 09:41

Dear Stefan,

I'm sorry to inform you that we have renovation trouble in the apartment you booked. As a consequence, we have to cancel your booking with us. We are very sorry for any inconvenience.

Das Taxi verspätet sich und ist dann nicht der bestellte Kombi. Unser Gepäck passt nicht in den Kofferraum...

Unser Kindersitz ist vom Flug beschädigt. Wir sind müde aber müssen reklamieren.

Unser Urlaub in  
**KAPSTADT**

SÜDAFRIKA

UNSERE EIGENEN  
KINDERSITZE MIT

- ✓ Für den Flug zugelassene
- ✓ Sitze für beide Kinder
- ✓ CARES-Gürtel für das Flugzeug nutzen
- ✓ Am Gate mit Transporttaschen aufgeben

WETTERBERICHT

☀️	28° / 18°
☀️	30° / 19°
☀️	28° / 17°
☀️	27° / 16°
☀️	26° / 15°

Zwiebellook, Sonnencreme & bequeme Schuhe!

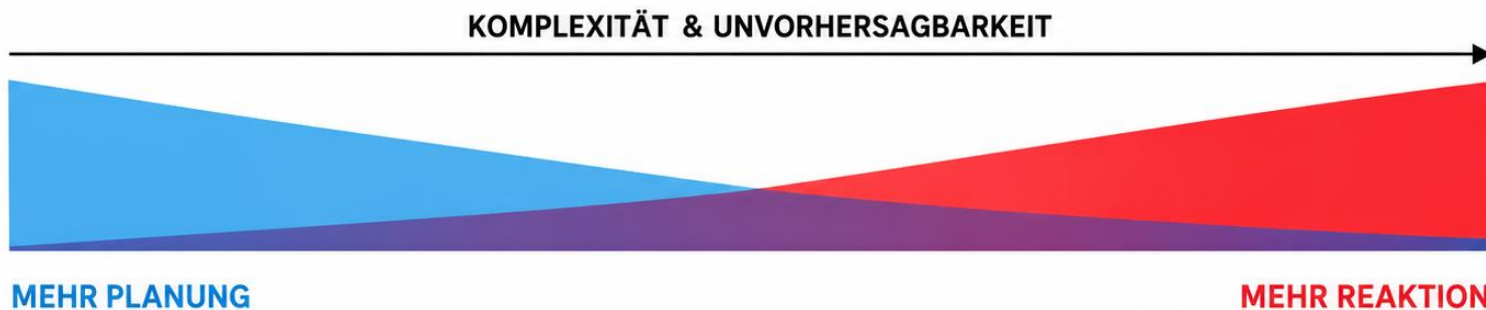
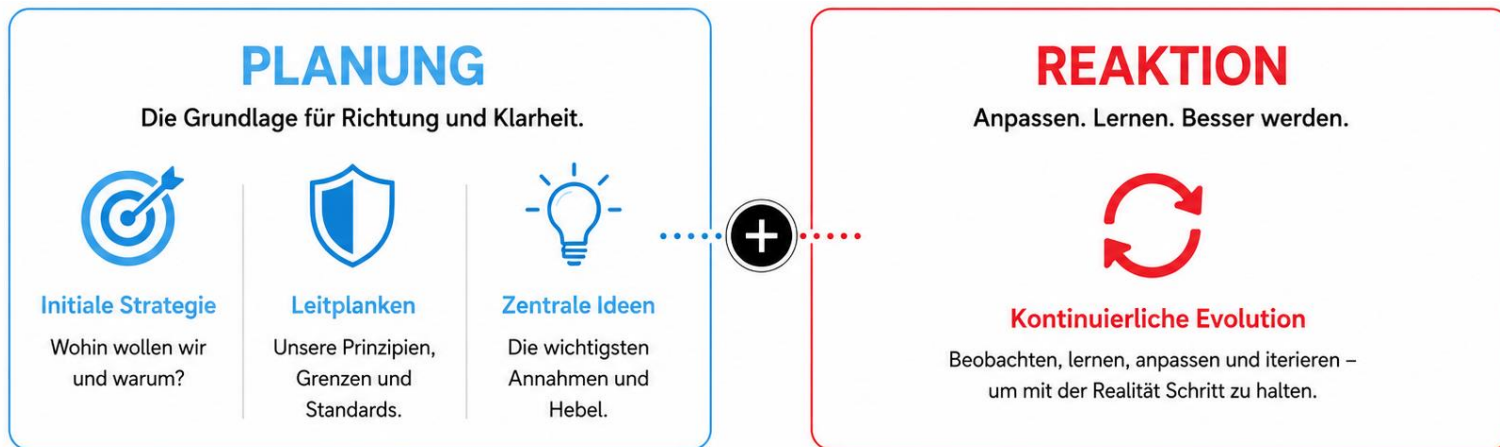
“EVERYONE  
HAS A PLAN  
UNTIL THEY GET  
**PUNCHED**  
IN THE MOUTH”

---

– MIKE TYSON



# Jedes Unterfangen hat...





# Über Software- Architektur





# Sichtweise von 1990

**architecture.** The organizational structure of a system or component. *See also:* **component;** **module;** **subprogram;** **routine.**

Approved September 28, 1990  
**IEEE Standards Board**



# Was passiert in den 1990ern?



## 1. Sprachen & Entwicklungswerkzeuge

- Python (1991)
- Visual Basic (1991)
- Java (1995)
- JavaScript (1995)
- PHP (1995)
- Ruby (1995)
- Delphi (1995)



## 2. Plattformen & Systeme

- Windows NT (1993)
- Linux (1991)
- Solaris (1992)
- Windows 95 (1995)
- Windows NT 4.0 (1996)



## 3. Web-, Server- & Enterprise-Technologien

- Apache HTTP Server 1.0 (1995)
- IIS 1.0 (1995)
- CSS1 (1996)
- HTML 4.0 (1997)
- JDBC (1997)
- Java Servlets (1997)
- RMI (1997)
- CSS2 (1998)
- XML 1.0 (1998)
- EJB 1.0 (1998)
- JSP 1.0 (1999)
- J2EE (1999)



## 4. Architektur- & Verteilungskonzepte

### Klare Technologie-/Standardeinführungen

- CORBA 1.0/1.1 (1991)
- COM (1993)
- DCOM (1996)

### Prägend bzw. stark verbreitet in den 1990ern

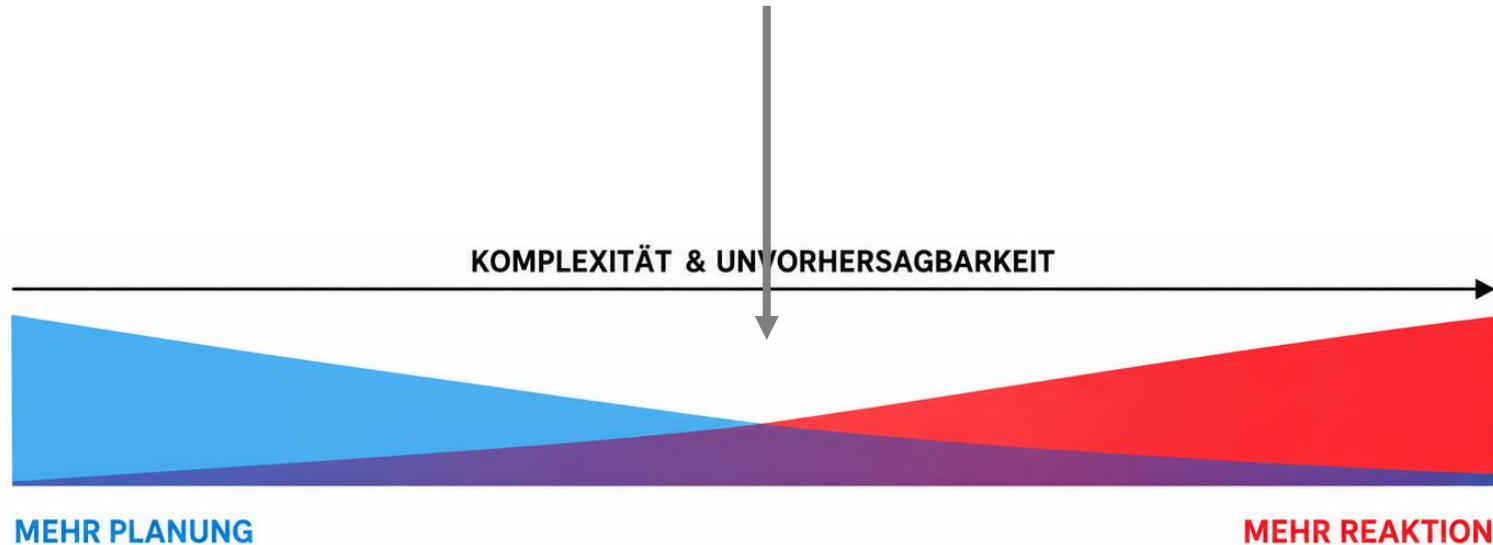
- Client-Server
- 3-Tier
- N-Tier
- Component-Based Architecture
- Distributed Systems
- Middleware
- Messaging
- Thin Client



# Sichtweise der 2000er Jahre

## IEEE Std 1471-2000

**3.5 architecture:** The fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution.





# CLOUD NATIVE LANDSCAPE

Application Definition and Image Build	Database	Continuous Integration & Delivery	Streaming & Messaging
	Scheduling & Orchestration	Service Proxy	Remote Procedure Call
API Gateway	Service Mesh	Coordination & Service Discovery	
Cloud Native Storage	Container Runtime	Cloud Native Network	
	Security & Compliance	Automation & Configuration	Container Registry
Observability	Chaos Engineering	Feature Flagging	

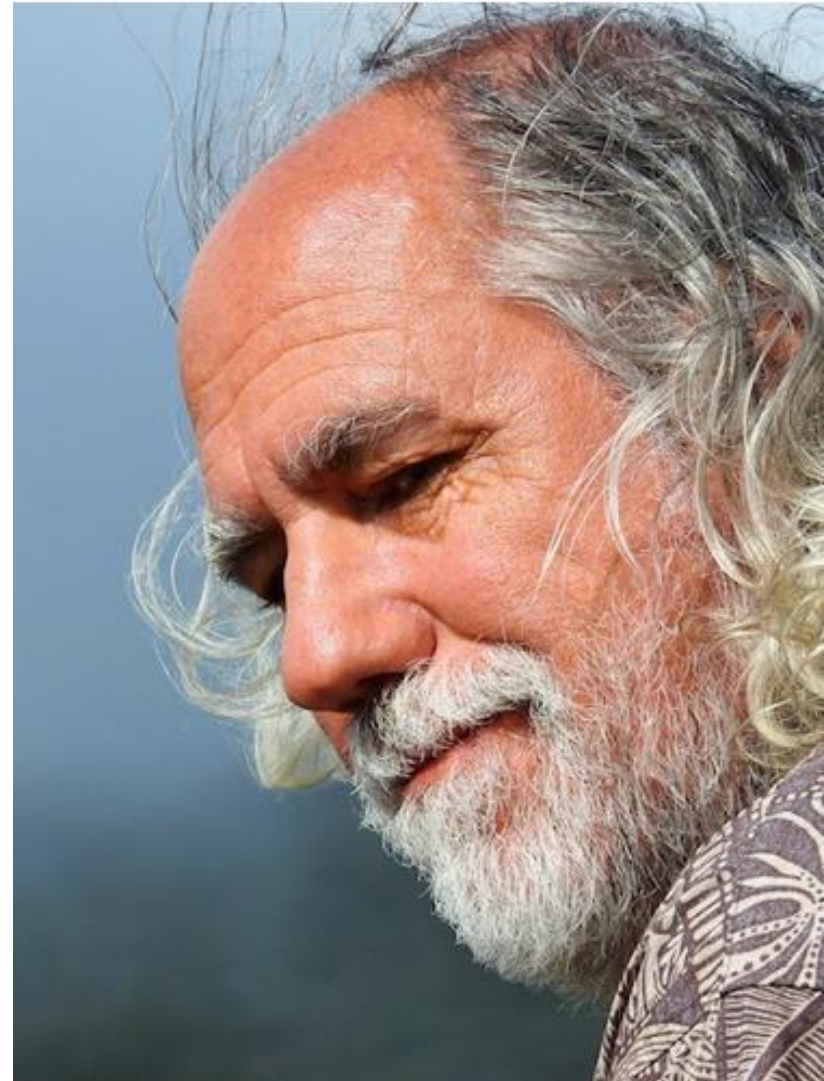
# Architektur

---

“The journey between vision and ultimate executable system is **complex**.

[...] myriad decisions, some [...] advance progress while others represent dead ends or trigger points for scrap and rework.”

*(Grady Booch)*



# Architektur

---

“The life of a software architect is a long and rapid succession of suboptimal design decisions taken partly in the dark.”

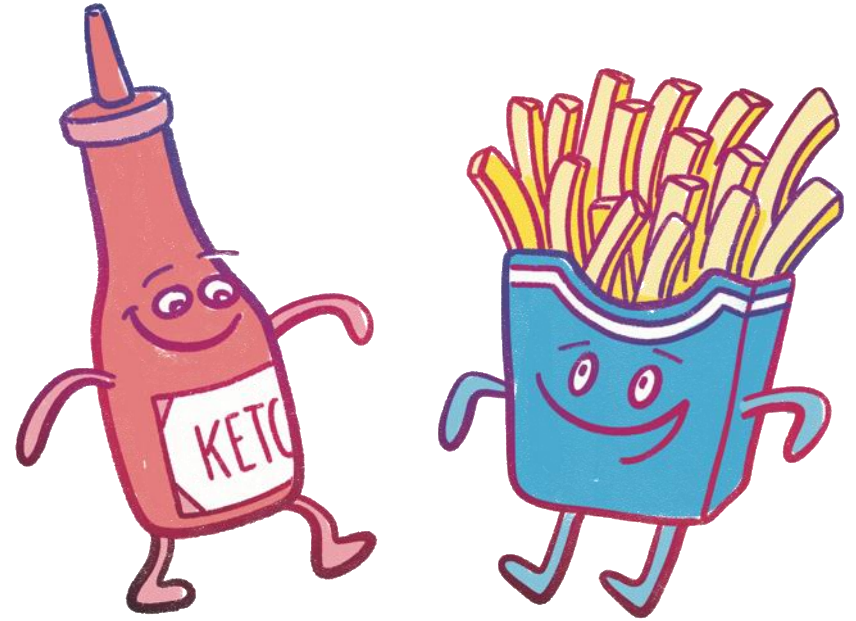
*(Philippe Kruchten)*





# Architektur und Agilität

A Perfect Match?



# Was macht „agil“ aus?

## ANNAHMEN



Es liegt keine  
Vollinformation vor



Was wir wissen,  
wird sich ändern



Wir lernen  
stetig hinzu



Es gibt keine perfekte  
Lösung – nur gültige  
Lösungen

## KERNIDEEN



Kleine  
Entwicklungsschritte



Gute Kommunikation  
und Transparenz  
sind zentral



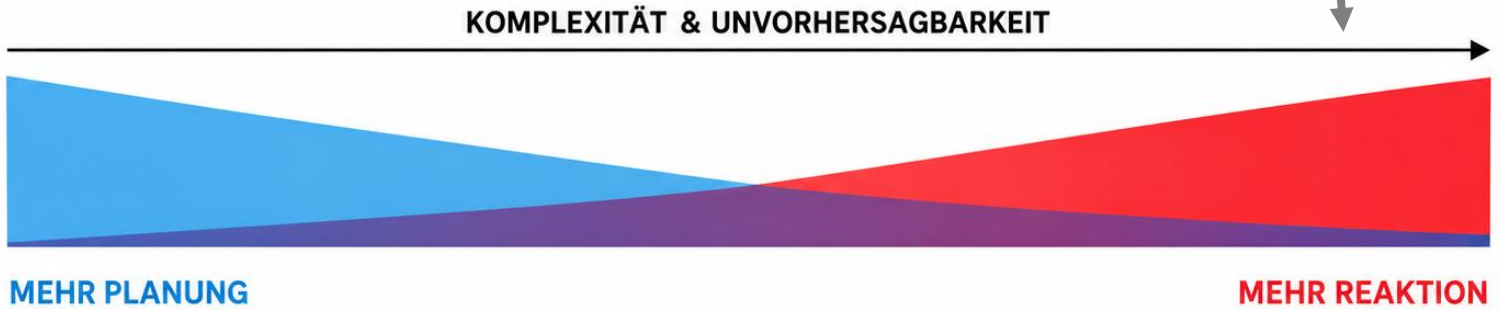
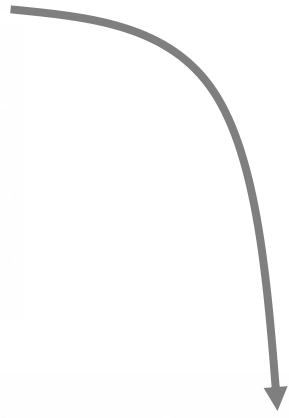
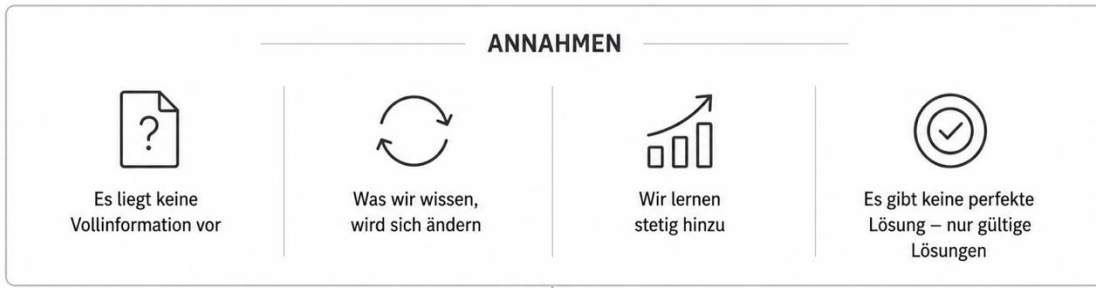
Risiken früh  
angehen

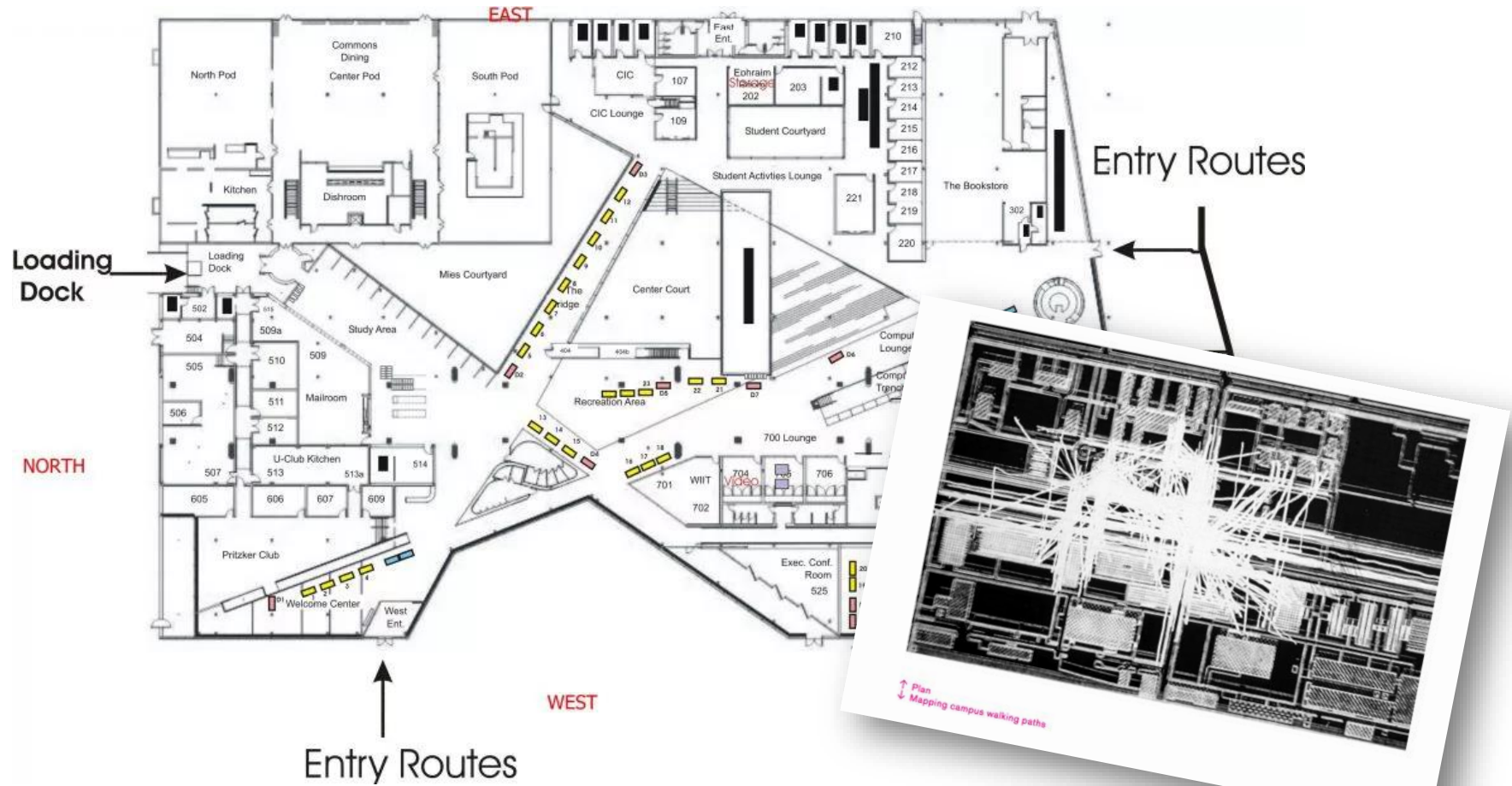


Stetiges, reales  
Feedback ist wichtig



Fehler und Probleme  
sind Lernchancen

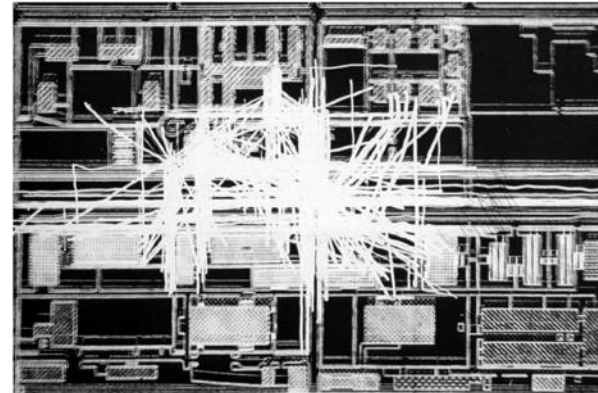




McCormick Tribune Campus Centers auf der Illinois Tech Universität

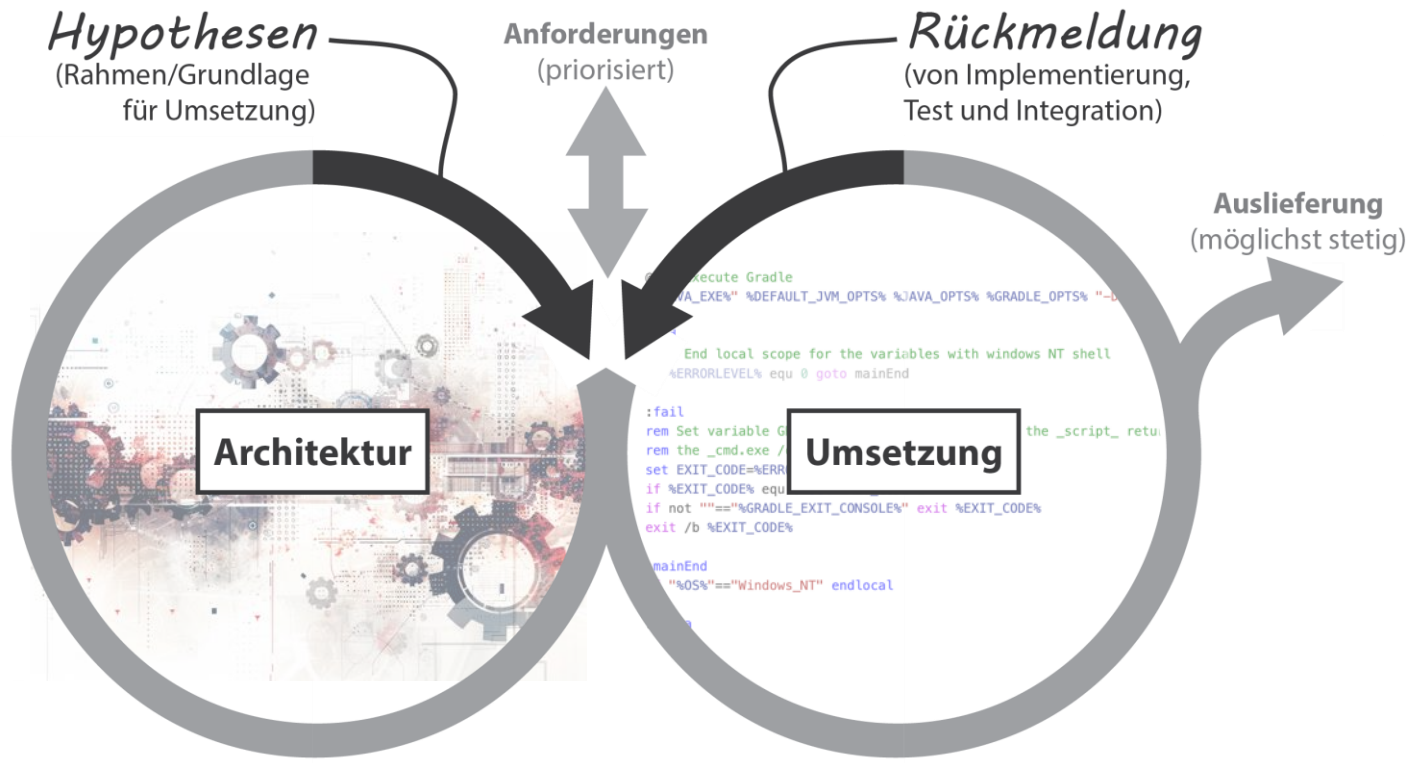
# Bottom-Up...

- **Ziele** von Nutzerinnen und Designer sind **homogen**
- Es gibt **Offenheit** aber auch einen **starken Rahmen**  
(Stundenpläne, Gebäudezwecke, ...)
- Was wenn der Kontext anders wäre?

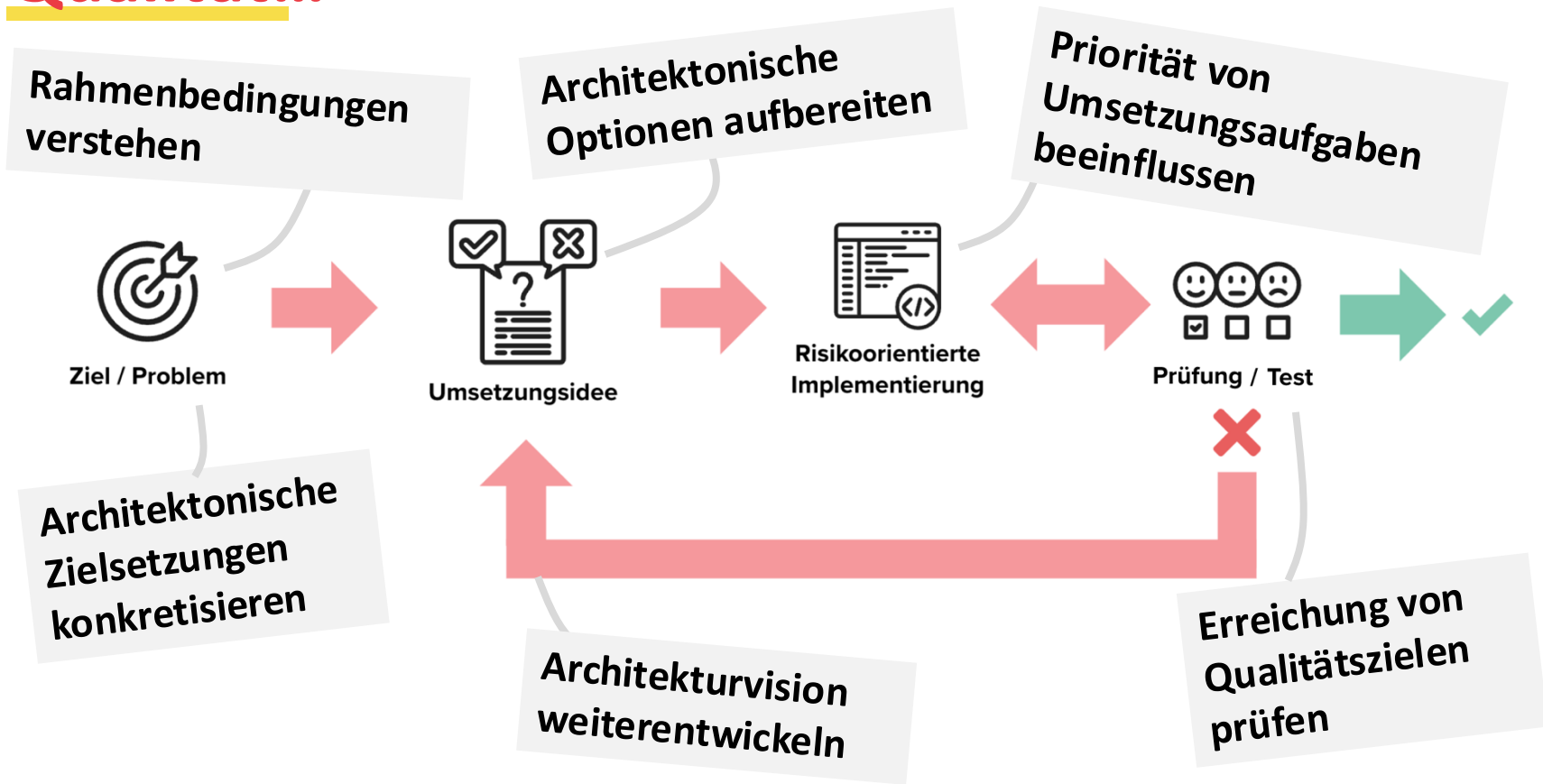




# Architektur als Entwicklungsmodus



# Zielorientierung, Flow, ausreichend Qualität...



# 03.

## Agile Software- Architektur



# Das Manifest agiler Architekturarbeit

---

**Kontinuierliche Verbesserung**



**Initialer Perfektion**

**Zielorientierung**



**Standardlösungen**

**Kollaboration**



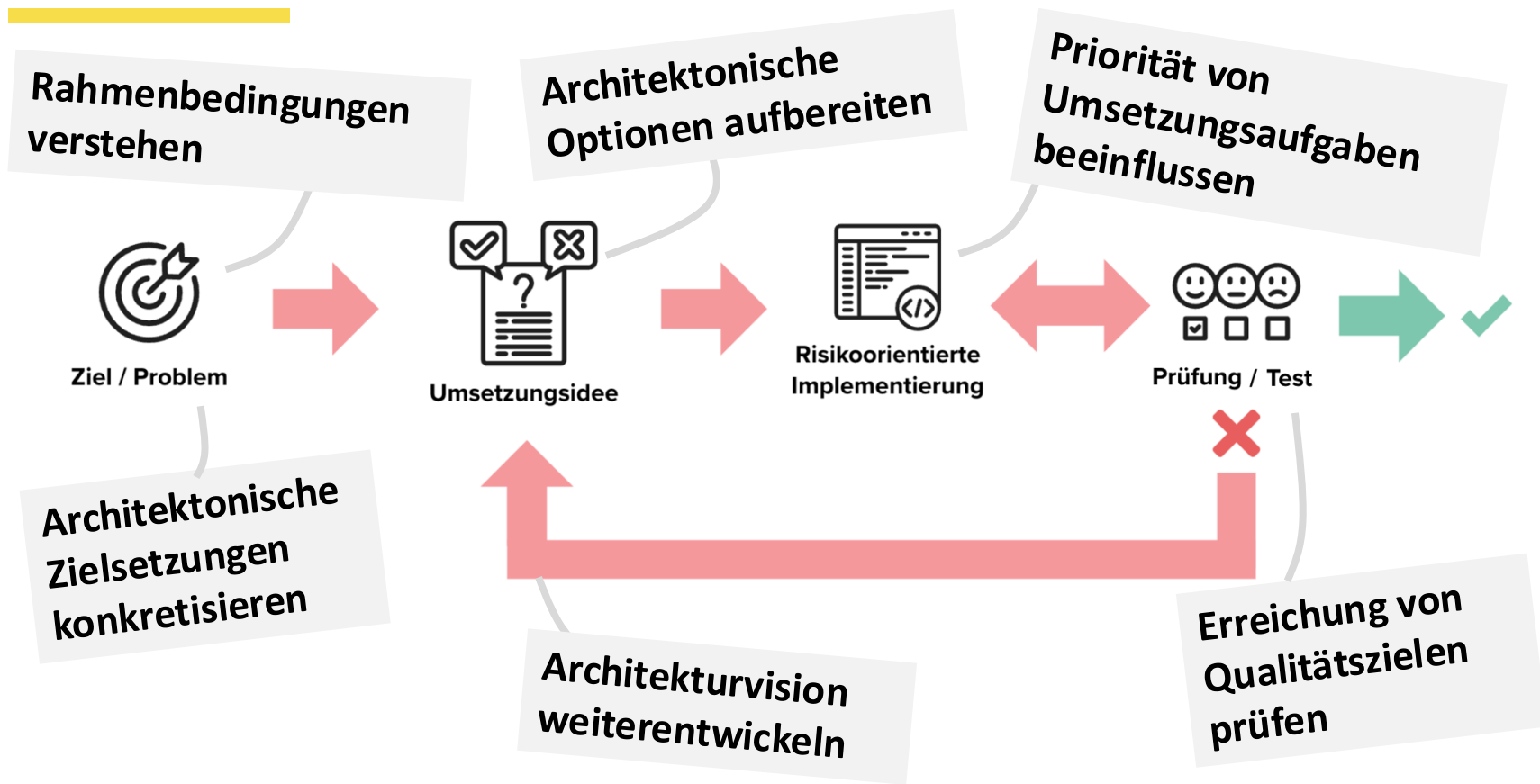
**Isolierter Spezialisierung**

**Breite Verantwortung**

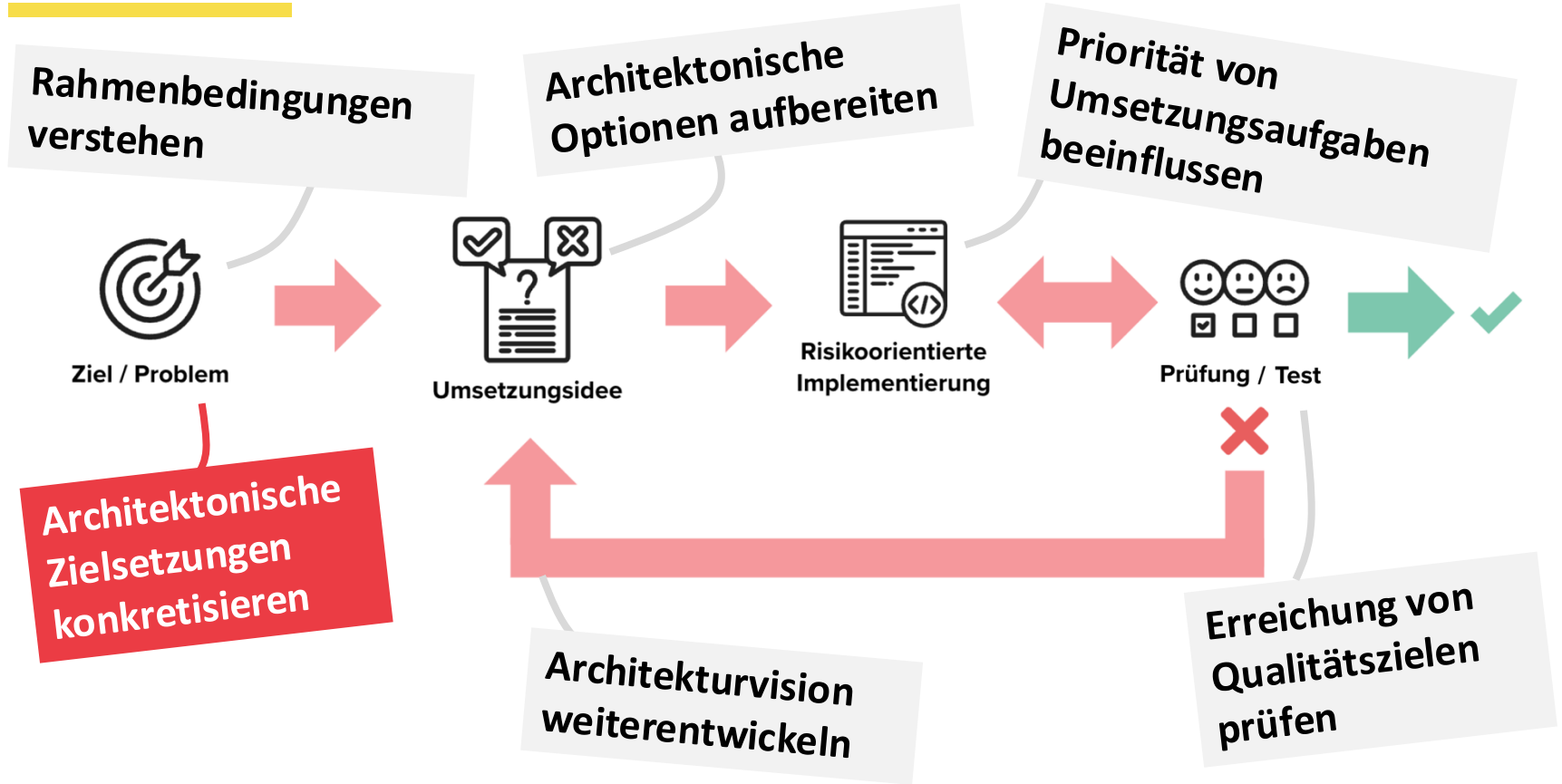


**Zentralisierung**

# Zielorientierung, Flow, ausreichend Qualität...



# Zielorientierung, Flow, ausreichend Qualität...



# Qualitätsmerkmale

Begriffe  
nach  
ISO 25010



## Funktionale Eignung (Functional Suitability)

Sind die berechneten Ergebnisse genau genug / exakt, ist die Funktionalität angemessen? ...



## Zuverlässigkeit (Reliability)

Ist das System verfügbar, tolerant gegenüber Fehlern, nach Abstürzen schnell wieder hergestellt? ...



## Benutzbarkeit (Usability)

Ist die Software intuitiv zu bedienen, leicht zu erlernen, attraktiv? ...



## Effizienz (Performance)

Antwortet die Software schnell, hat sie einen hohen Durchsatz, einen geringen Ressourcenverbrauch? ...



## Sicherheit (Security)

Ist das System sicher vor Angriffen? Sind Daten und Funktion vor unberechtigtem Zugriff geschützt? ...



## Portabilität (Portability)

Ist die Software leicht auf andere Zielumgebungen (z.B. anderes OS) übertragbar? ...



## Kompatibilität (Compatibility)

Ist die Software konform zu Standards, arbeitet sie gut mit anderen zusammen? ...



## Wartbarkeit (Maintainability)

Ist die Software leicht zu ändern, erweitern, testen, verstehen? Lassen sich Teile wiederverwenden? ...

# Konkretisierung in Qualitätsszenarien

ID

z. B. W0x

Zugeordnetes Qualitätsziel

z. B. Wartbarkeit

Szenariobeschreibung (2-3 Sätze)



SzenarioArt

- Verwendung/Usability
- Änderung/Wachstum
- Fehler/Katastrophen/Stress

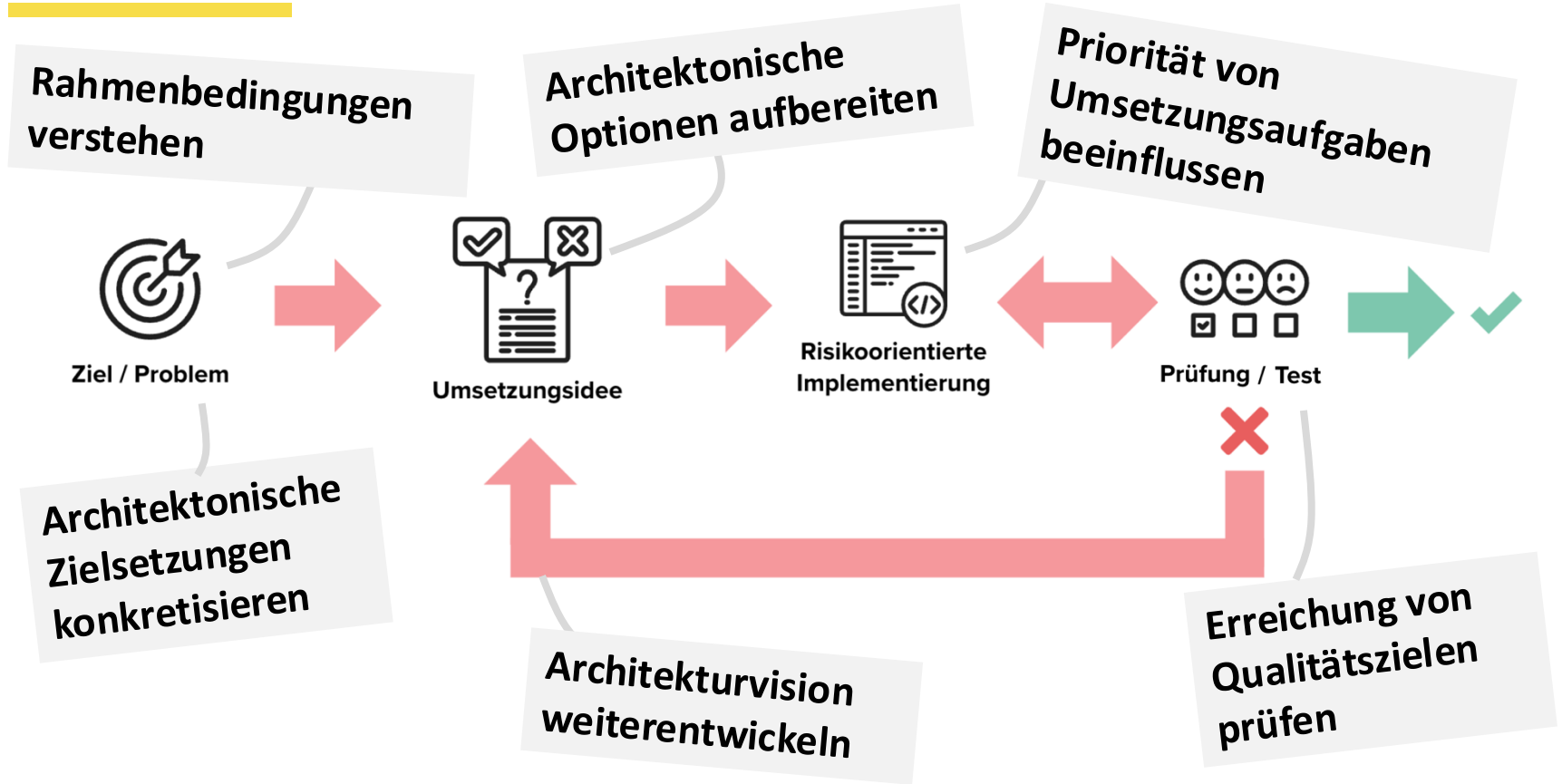
SzenarioKategorie

- Akzeptanzkriterium
- Qualitätsgeschichte
- Prinzipienlücke

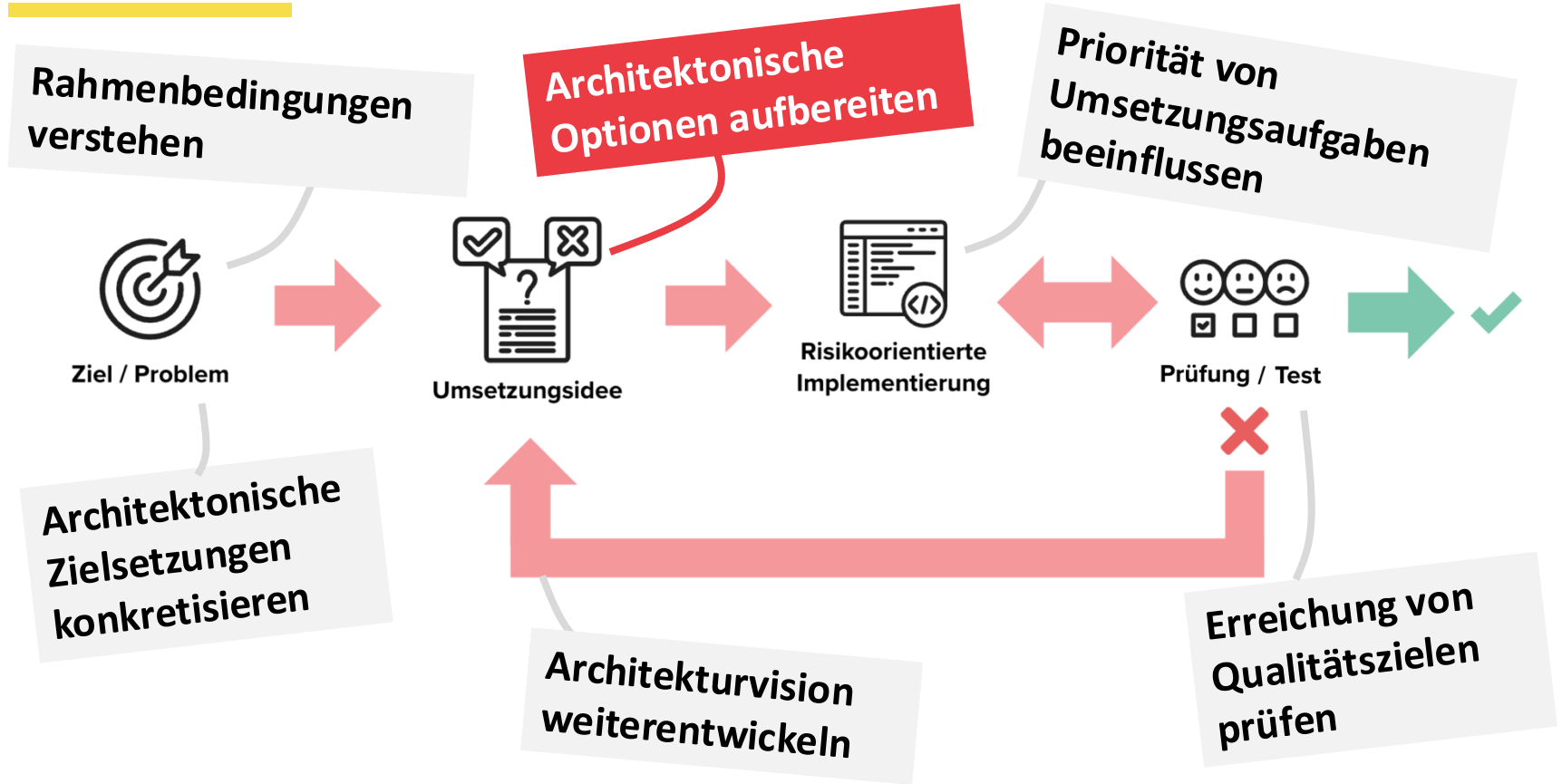
Aufwandsschätzung

- S
- M
- L
- XL

# Zielorientierung, Flow, ausreichend Qualität...



# Zielorientierung, Flow, ausreichend Qualität...

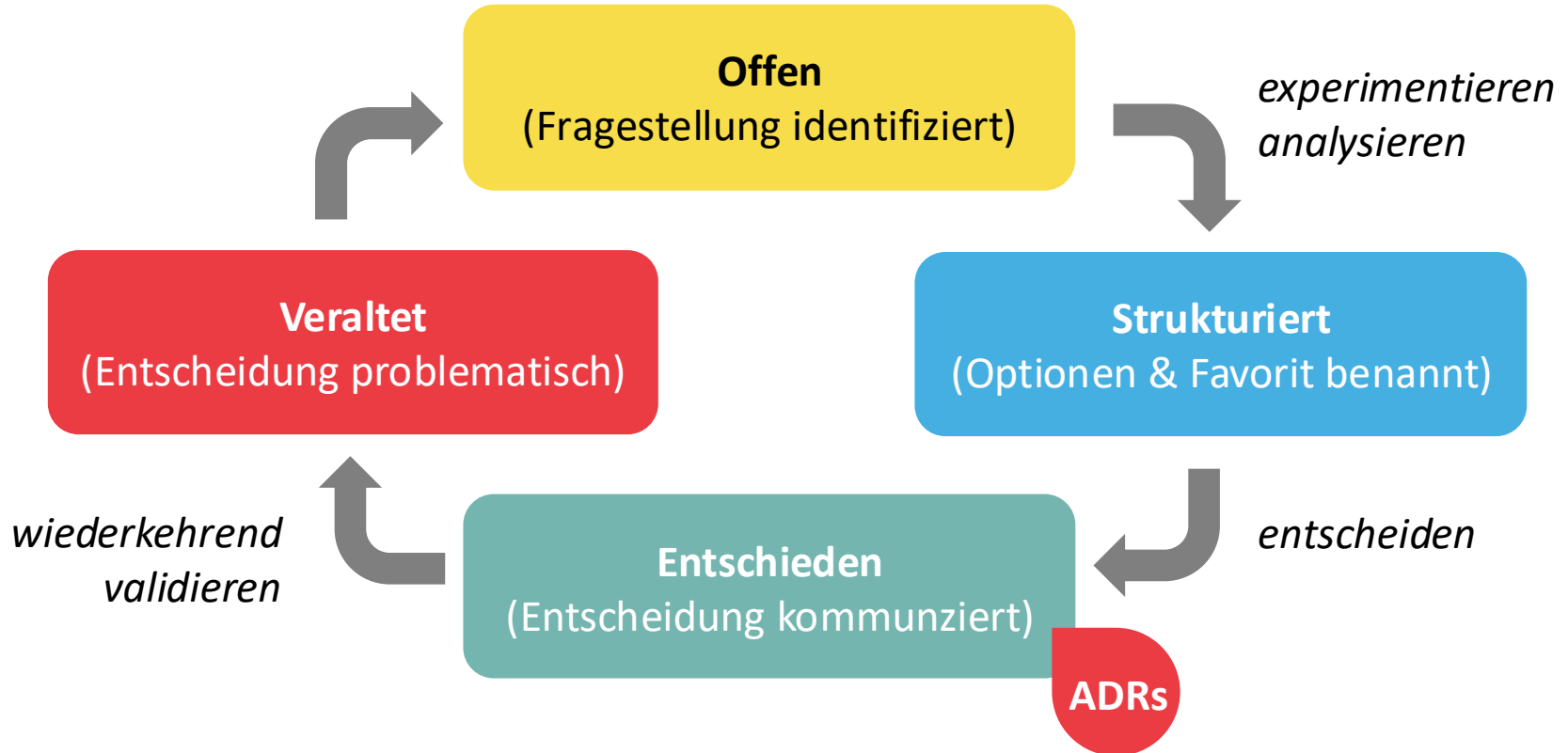




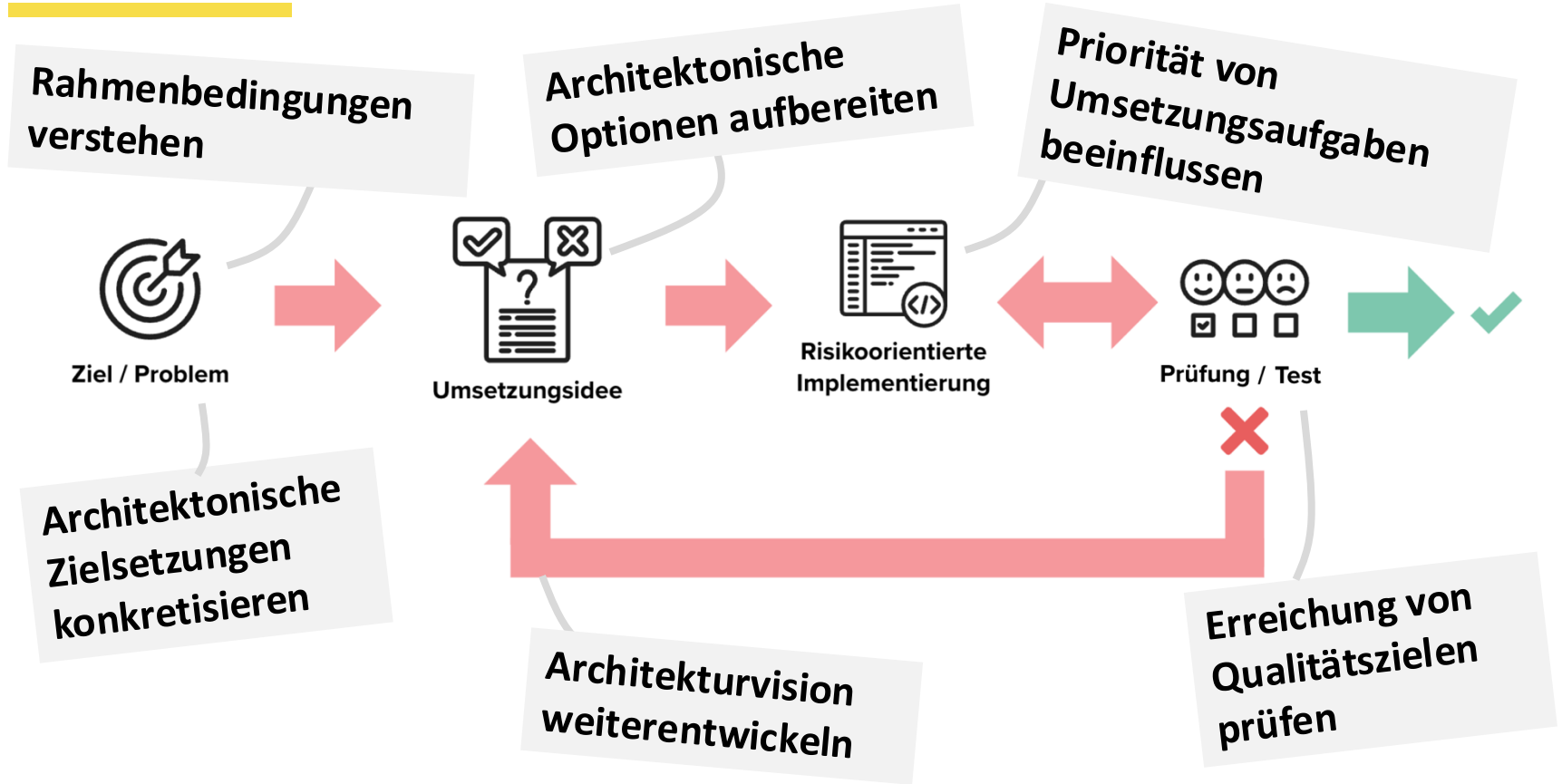
# Architekturentscheidungen - ADRs



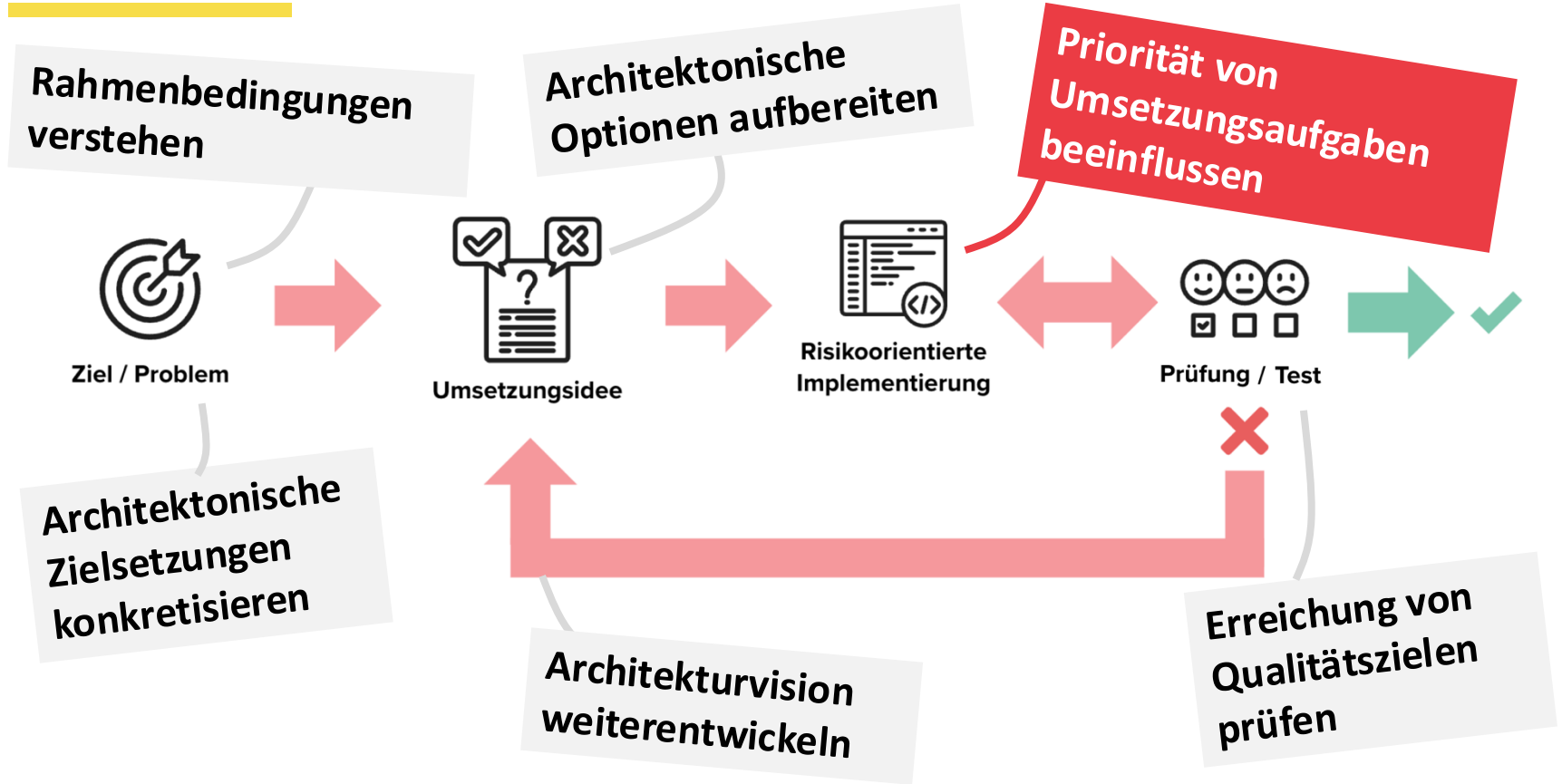
# Architektonische Fragestellungen



# Zielorientierung, Flow, ausreichend Qualität...



# Zielorientierung, Flow, ausreichend Qualität...



# Walking Skeleton



„A **Walking Skeleton** is a tiny implementation of the system that performs a small end-to-end function. It need **not** use the **final** architecture, but it should **link together** the **main architectural components**. The architecture and the functionality can then evolve in parallel.“

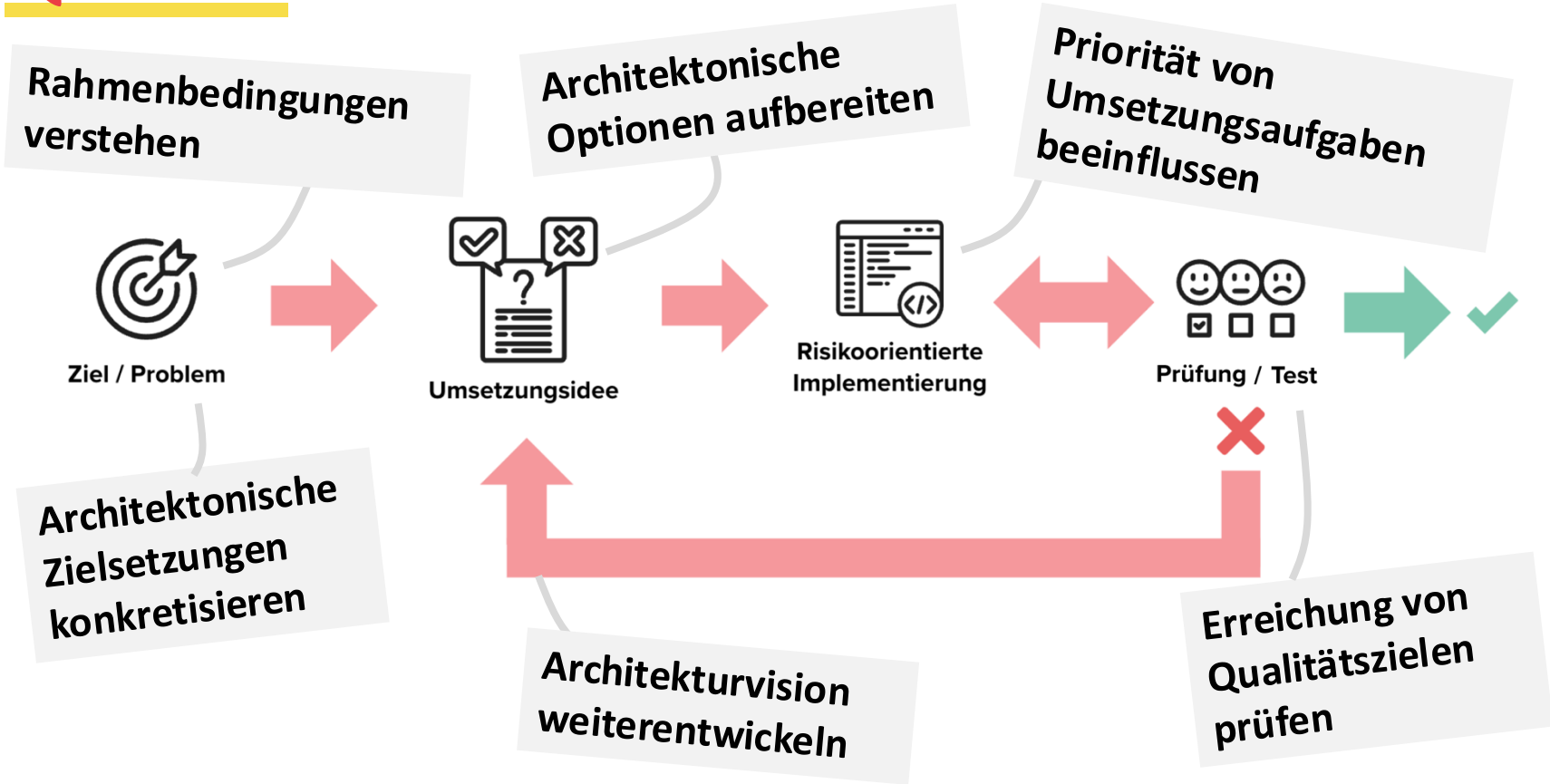
- Alistair Cockburn

**Offen**  
(Fragestellung identifiziert)

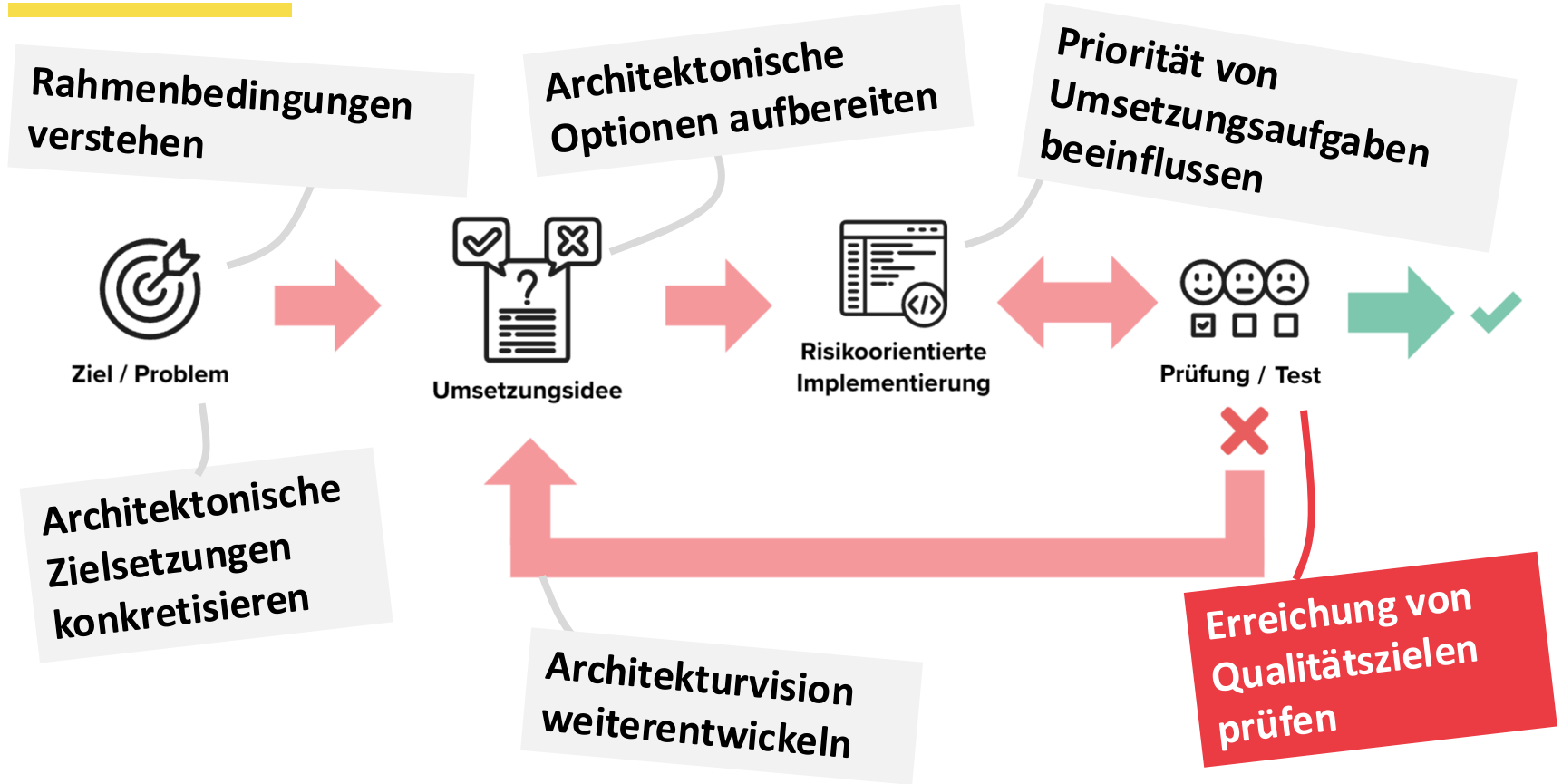
**Strukturiert**  
(Optionen & Favorit  
benannt)



# Zielorientierung, Flow, ausreichend Qualität...



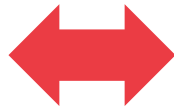
# Zielorientierung, Flow, ausreichend Qualität...





# Elemente von Architekturarbeit

- Effizienz
- Sicherheit
- Zuverlässigkeit
- Skalierbarkeit
- Wartbarkeit
- Portierbarkeit
- ...



„Qualitätsmerkmale“

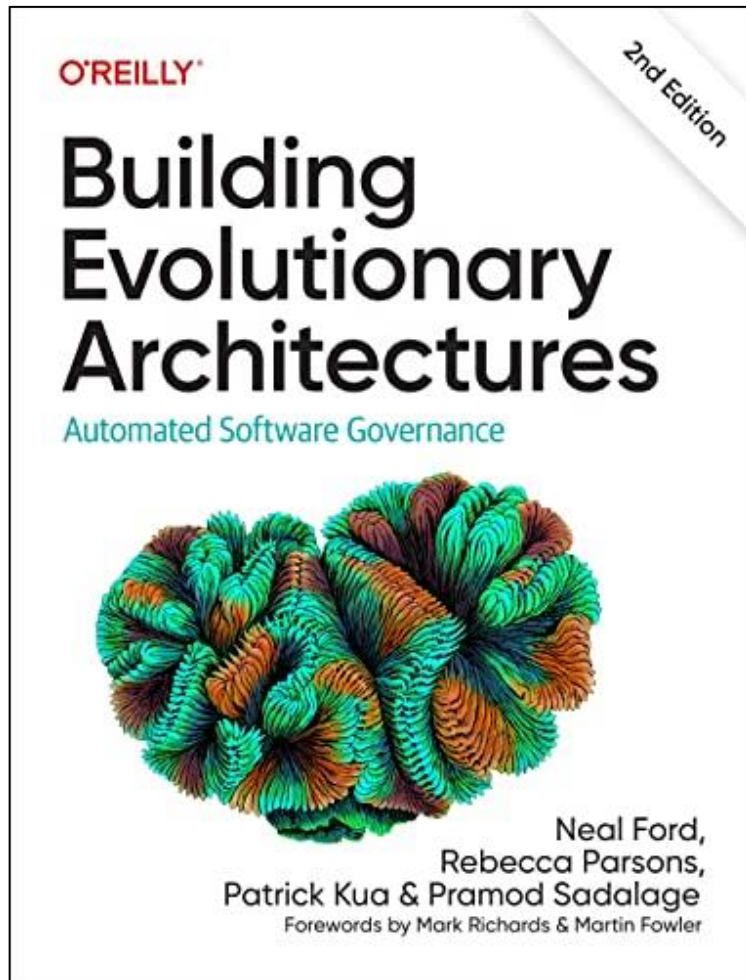
- Strukturierung
- Muster, Stile
- Framework, Libs
- Protokolle
- Basistechnologien
- Umgebungen
- ...



„Entscheidungen“



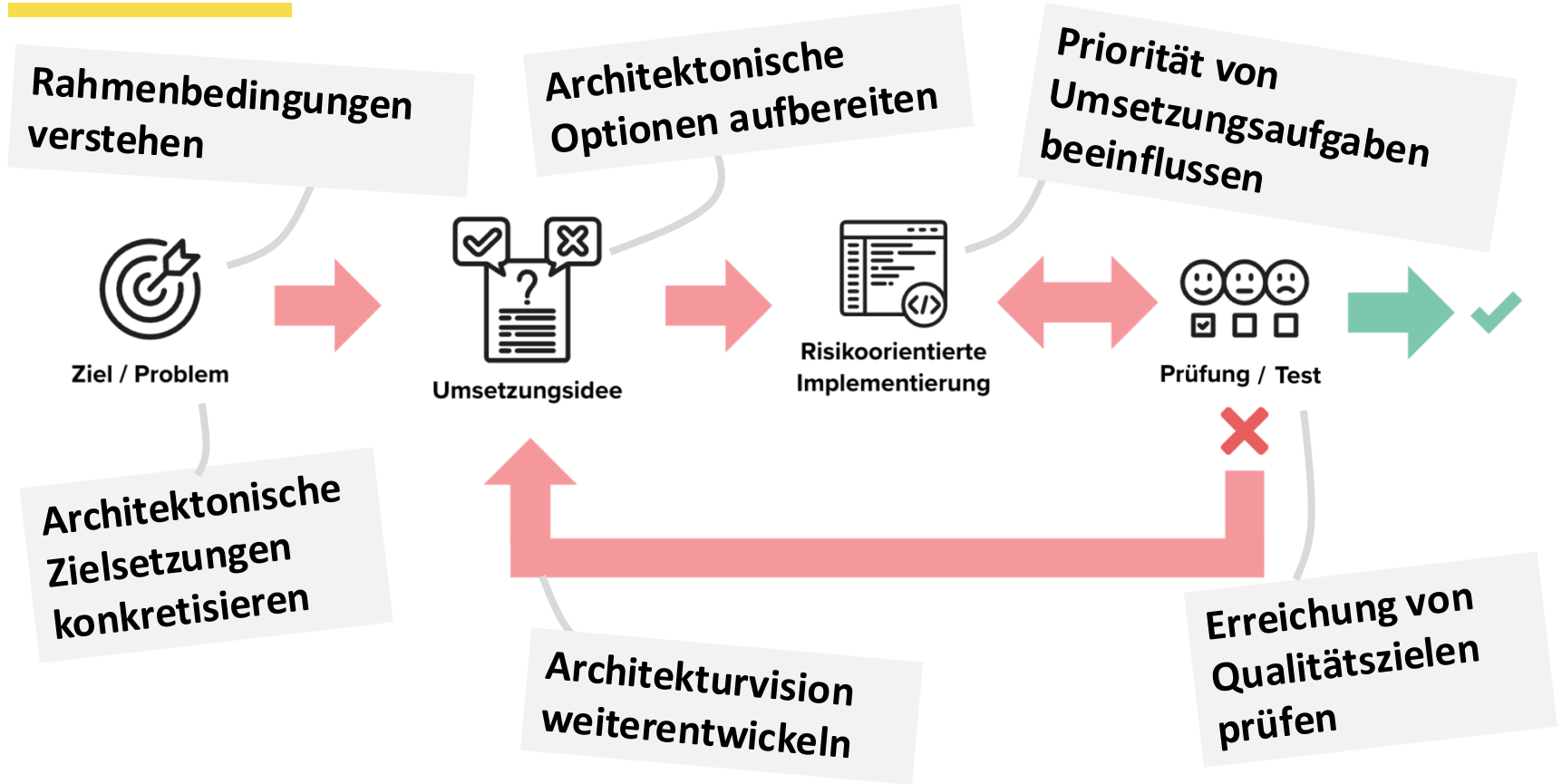
Feedback!  
?



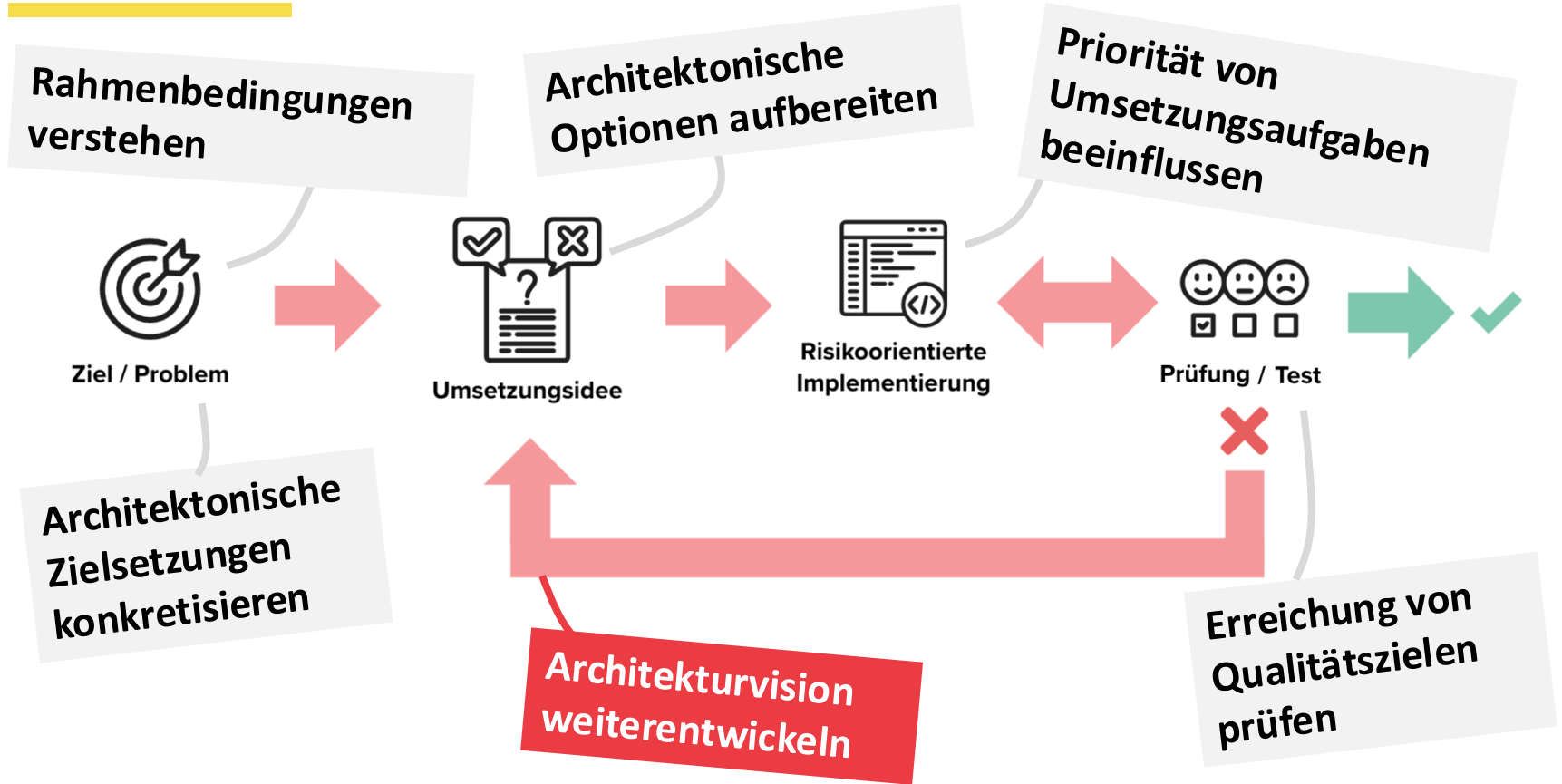
Eine **Fitness Function** misst objektiv, wie gut eine Lösung die an sie gesetzten Ziele erreicht.

- Building Evolutionary Architectures – Ford, Parsons, Kua, Sadalage, O'Reilly 2017

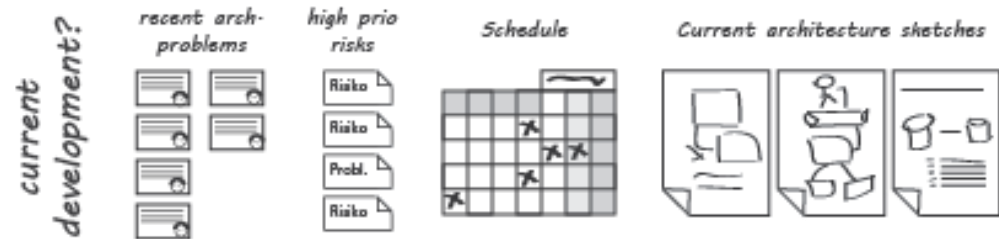
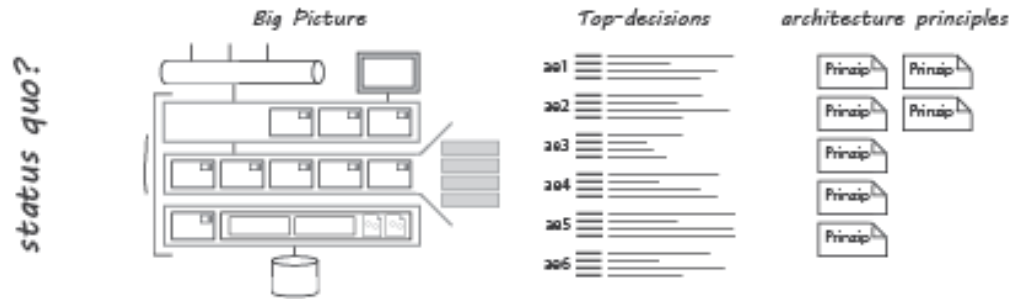
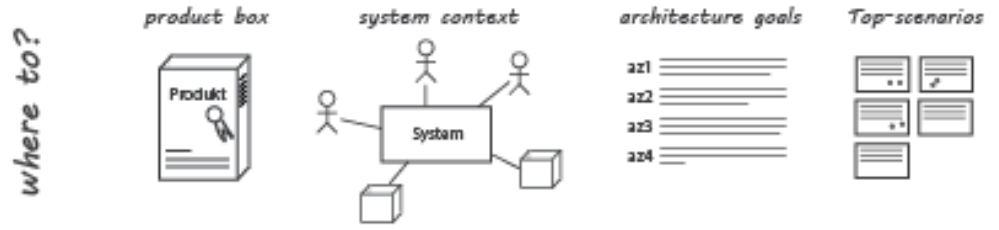
# Zielorientierung, Flow, ausreichend Qualität...



# Zielorientierung, Flow, ausreichend Qualität...



# Vision und Zustand der Architektur zeigen



- Knapper Überblick
- Änderungshäufigkeit: unten mehr
- Physisch oder digital
- Erhöht Transparenz
- Ermöglicht Post-It Sessions

# Vision and Goals

## Quality Goals

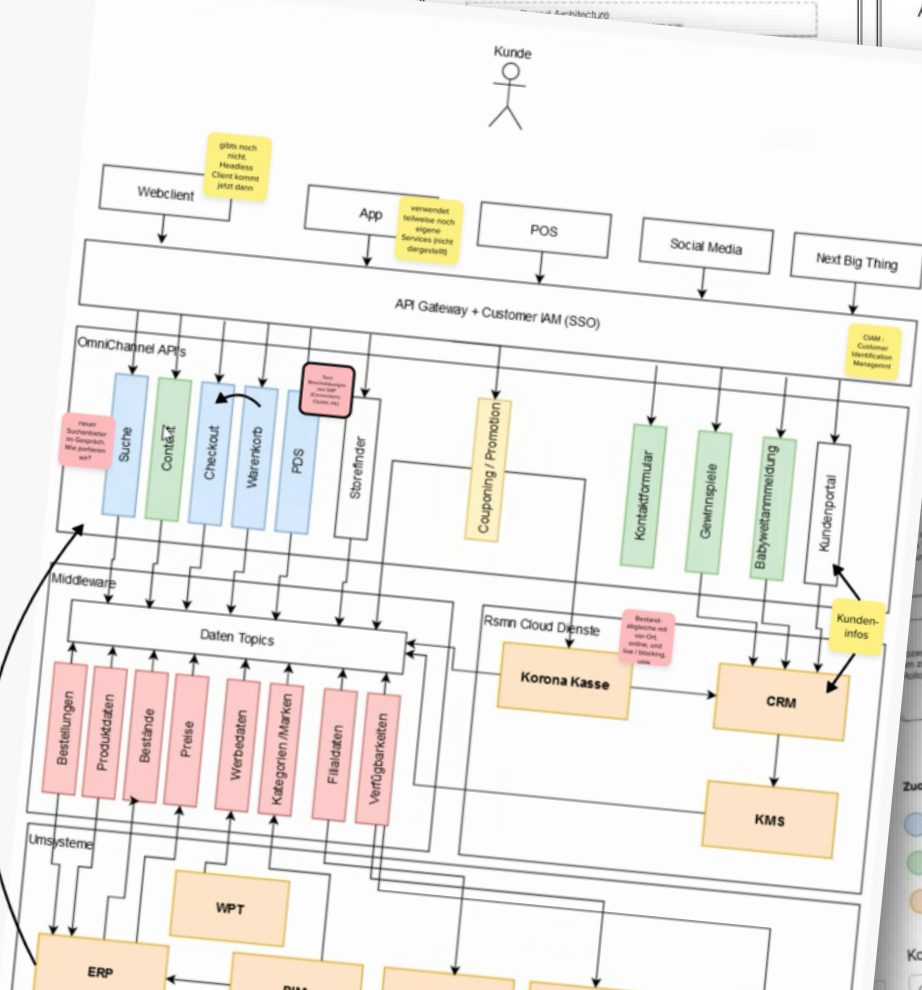
The quality goals are defined in conformance with ISO 2007. The foundation is the result of individual priorities from business requirements, requirements engineering, detailed representation, Project Management was given double IT

- 1. Maintainability**  
Sponsors: PM, DEV, ARCH  
Sub-Characteristics: Maintainability, Modifiability
- 2. Reliability**  
Sponsors: PM, DEV, ARCH  
Sub-Characteristics: Maintainability, Availability
- 3. Functional Correctness**  
Sponsors: PM, DEV, ARCH  
Sub-Characteristics: Data Quality, Content
- 4. Performance**  
Sponsors: PM, DEV, ARCH  
Sub-Characteristics: Time Behavior, Cost
- 5. Interoperability**  
Sponsors: DEV, ARCH  
Sub-Characteristics: Compatibility, Interoperability
- 6. Usability**  
Sub-Characteristics: Complexity, User Experience
- 7. Security**  
Sub-Characteristics: Confidentiality, Integrity
- 8. Portability**  
Sub-Characteristics: Adaptability, Portability

## System Components

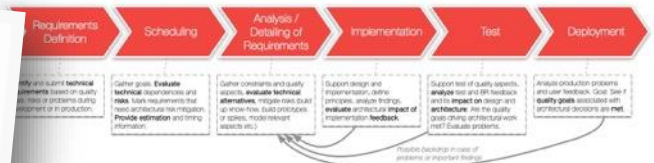
- Daten**
  - Bestandsupdate
  - Größe: Proportionale zur Oberfläche (z.B. CRM, etc.)
- Information**
  - Adress Doctor
  - Zerbus

# Technical Principles



# Process and Methodology

## Architecture and the development process



## Principles of Architectural practice

- 1.1 Early risk assessments:** Based on required quality goals, e.g. carried out as a 'Risky Check' of quality goals.
- 1.2 Requiring technical / conceptual design reviews:** That try to uncover inconsistencies, redundancies, areas that bear false hope, missing general solutions, special solutions or 'black solutions' that don't fit quality objectives or project constraints etc.
- 1.3 Inclusion of architects in the activities before pre-approval:** To identify architecturally relevant requirements early on, activities leading to the conceptual planning, prototyping or later implementation and influence the provision in high risk areas.
- 1.4 Establish an architecture board:** To make the status of identified architecture topics visible and to support their management.
- 2.1 Architectural needs are visible as requirement:** and influence functional requirements and sub-requirements.



## Zuordnung zu Architekturzielen (Farbcode)

- Offlinefähigkeit
- Benutzbarkeit
- Zuverlässigkeit
- Wertbarkeit
- Skalierbarkeit

## Komponentendiagramm

① C4 Diagramme zur Darstellung von: ... Diagramme – Ebene 3): Zerlegen Container in zusammenhängende Komponenten und setzen die

# 04.

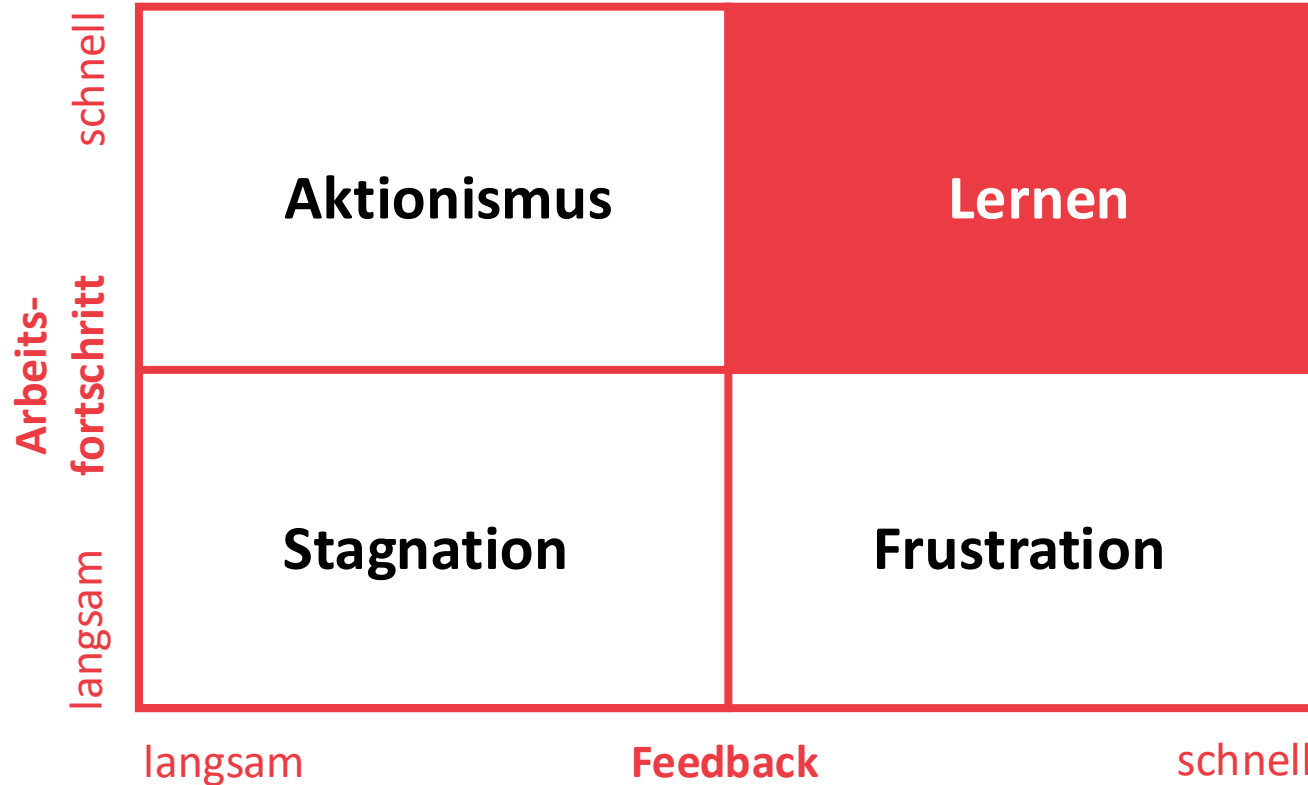
**Und AI !?11!!!**

AI! denkt irgendjemand auch mal an AI?



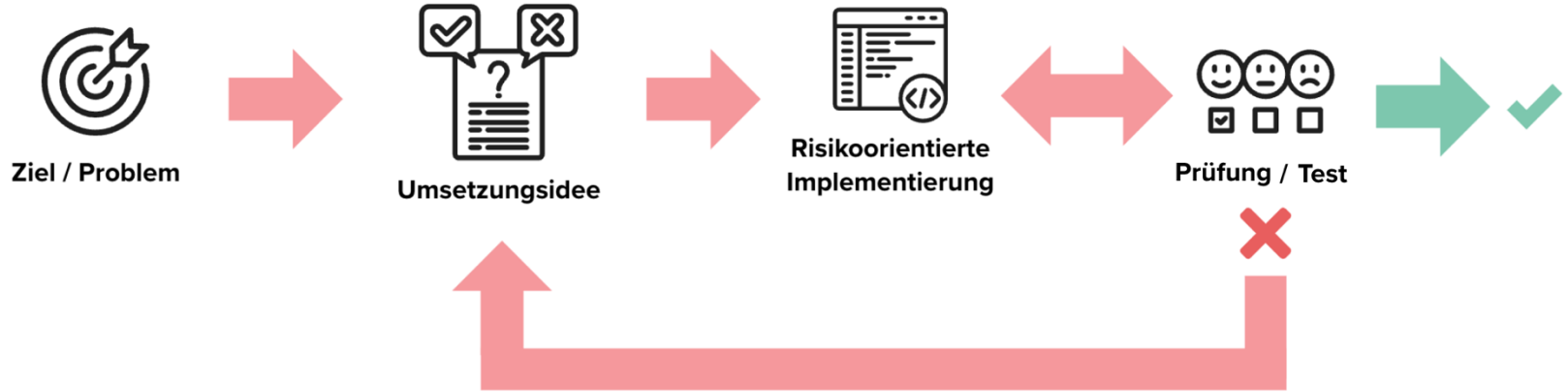


# Feedback & Lernen

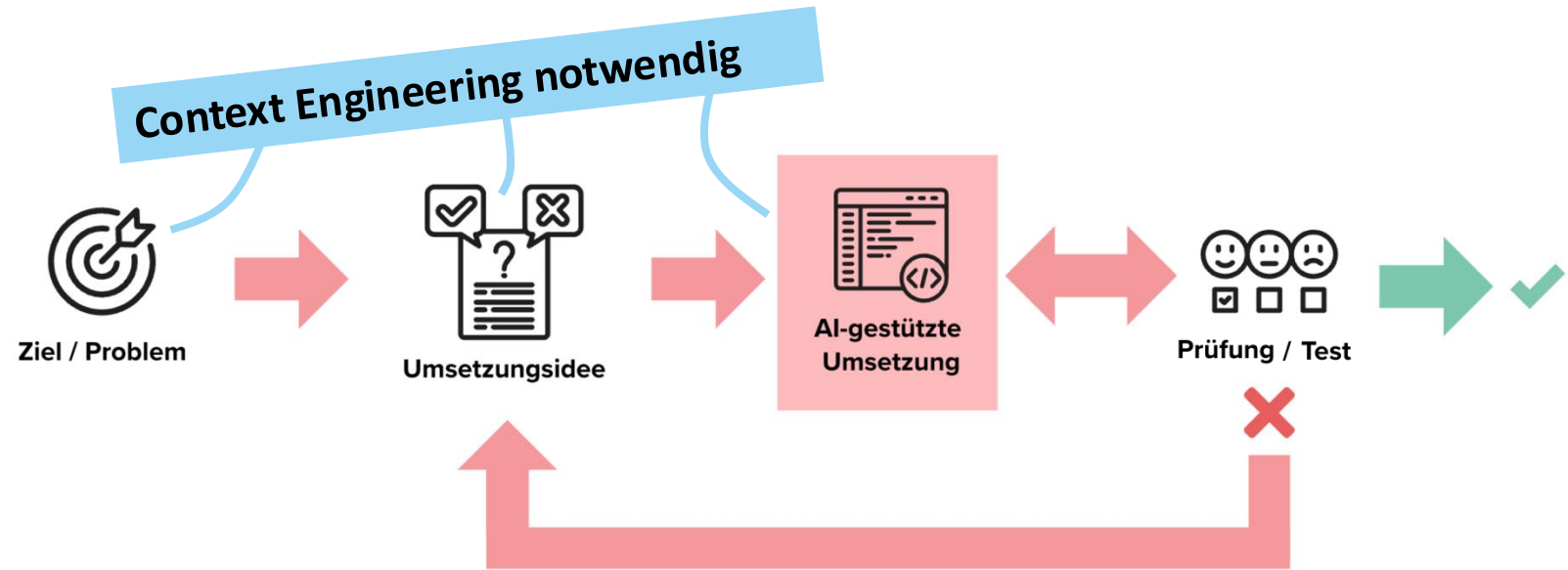




# Was ändert die Verwendung von LLMs?

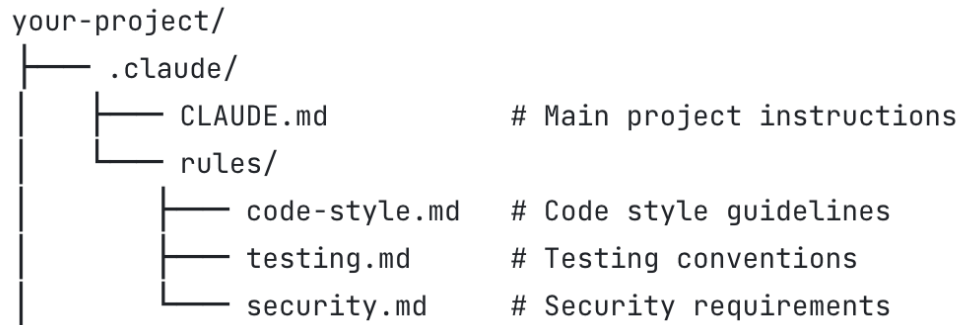


# Generative KI in der Entwicklung



# Context Engineering Optionen

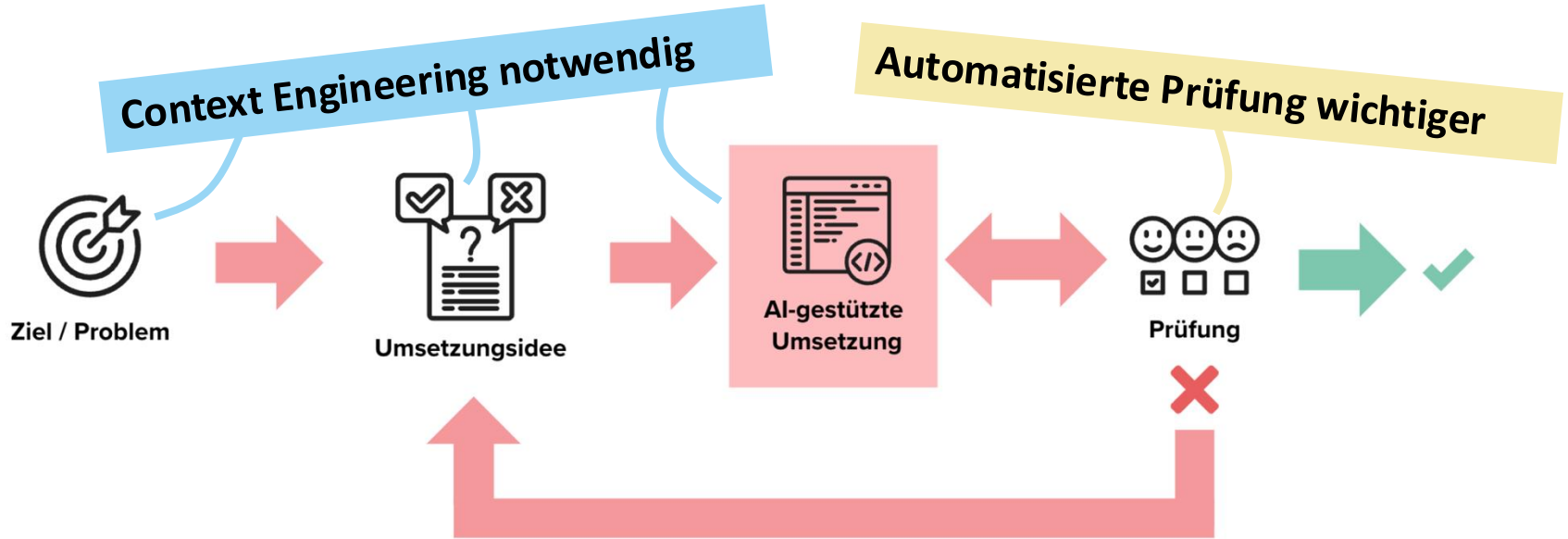
- Claude.md / Agents.md
- Regeln (z.b. als .md)
- System Prompt
- Tools (Fähigkeiten)
- MCP-Server
- Sub-Agenten / Modes (eigener Kontext)
- Skills
- Hooks
- Templates
- ...



```

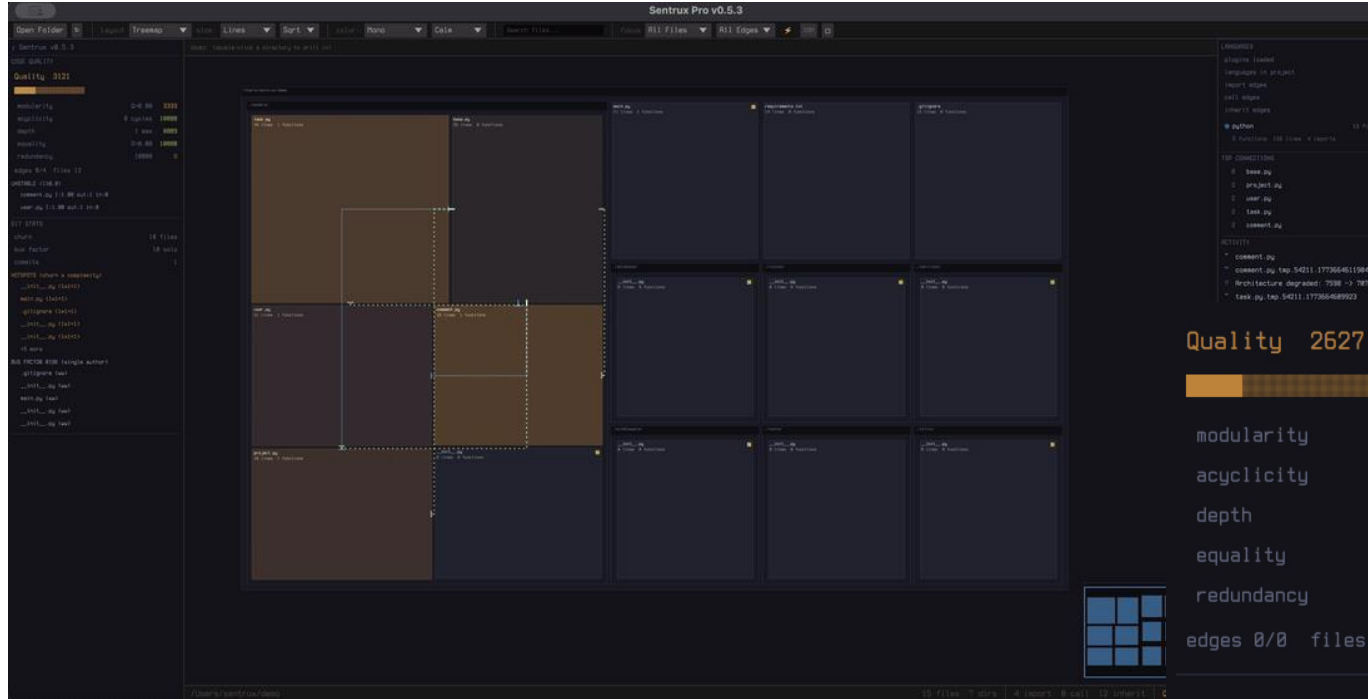
/context
├── Context Usage
│   └── claude-sonnet-4-20250514 • 116k/200k tokens (58%)
│       ├── System prompt: 2.8k tokens (1.4%)
│       ├── System tools: 11.6k tokens (5.8%)
│       ├── MCP tools: 17.0k tokens (8.5%)
│       ├── Memory files: 178 tokens (0.1%)
│       ├── Messages: 84.0k tokens (42.0%)
│       └── Free space: 84.4k (42.2%)
├── MCP tools • /mcp
│   └── mcp_applescript_execute_applescript_execute (applescript_execute)588 tokens
└── mcp_filesystem_read_file (filesystem): 475 tokens
    └── mcp_filesystem_read_text_file (filesystem): 556 tokens
  
```

# Generative KI in der Entwicklung





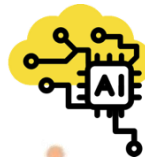
# Design Feedback-Loop



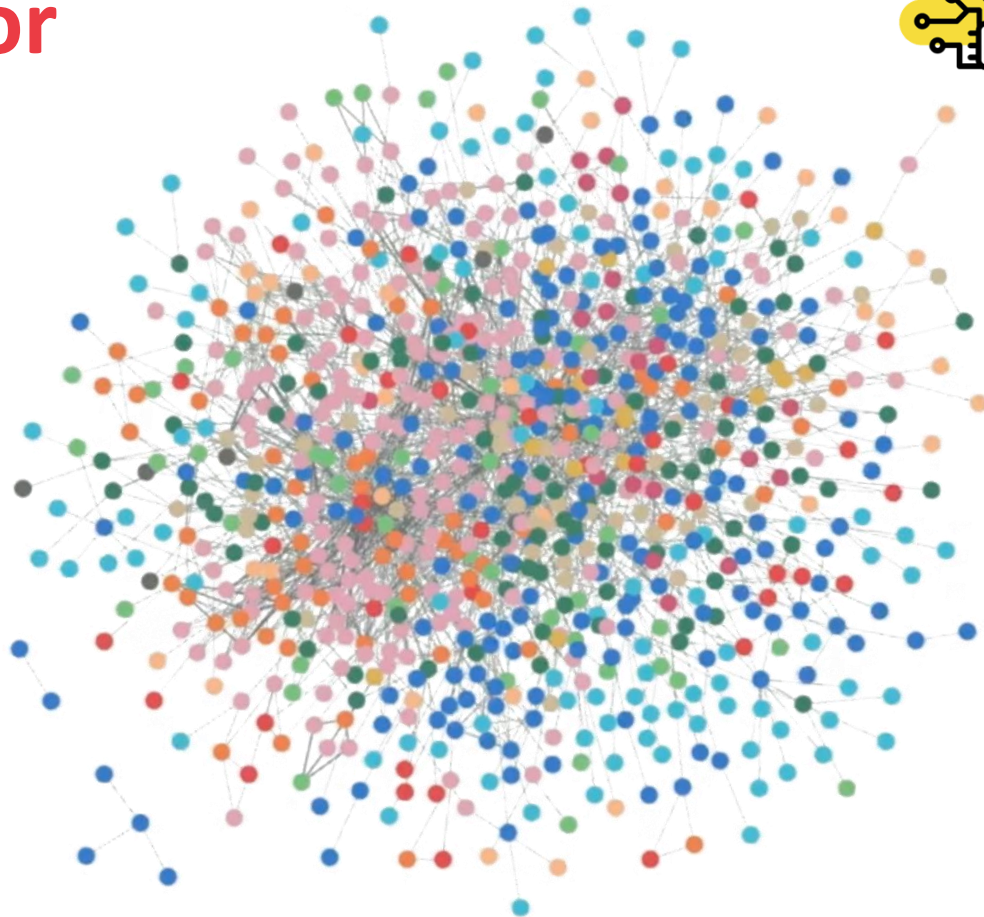
Quality 2627

modularity	Q=1.00	10000
acyclicity	0 cycles	10000
depth	0 max	10000
equality	G=0.08	1250
redundancy	10000	0
edges	0/0	files 0

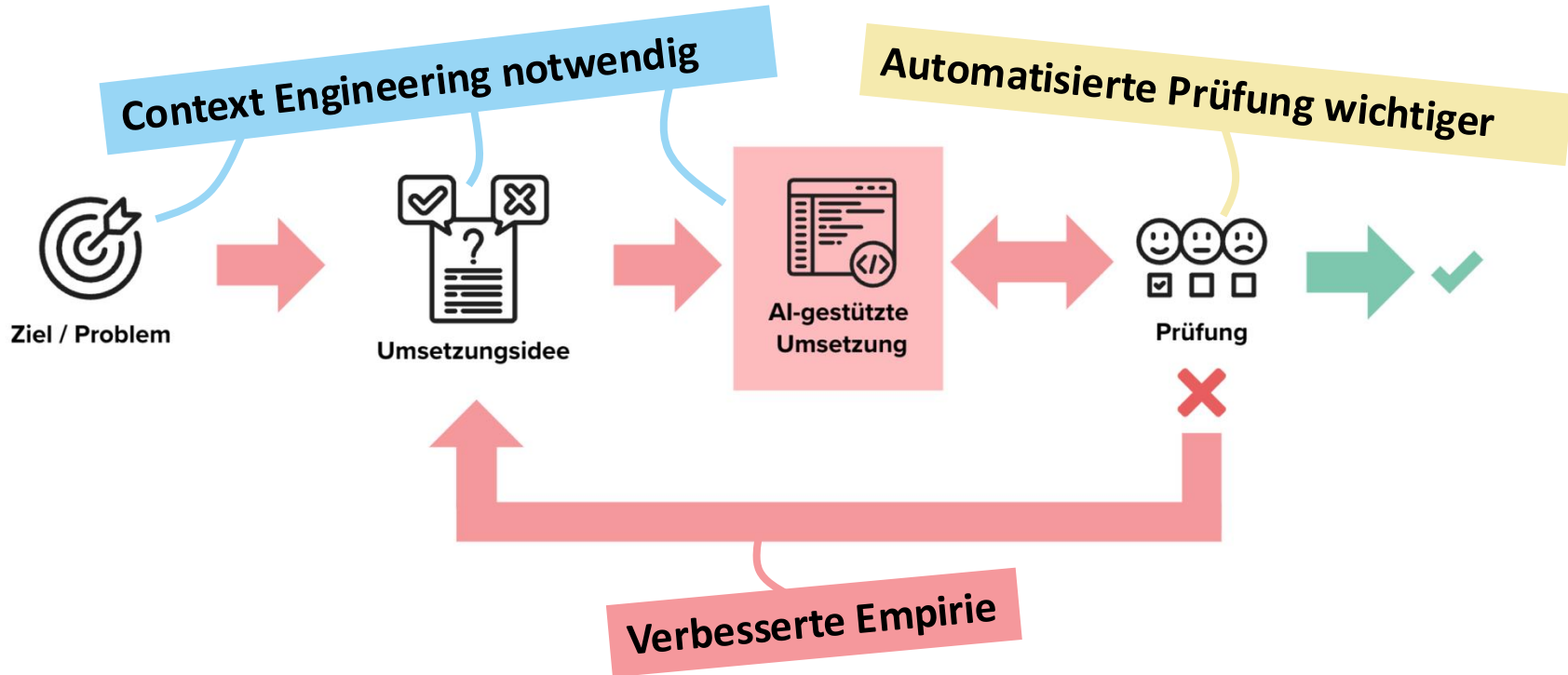
# Graph Constructor



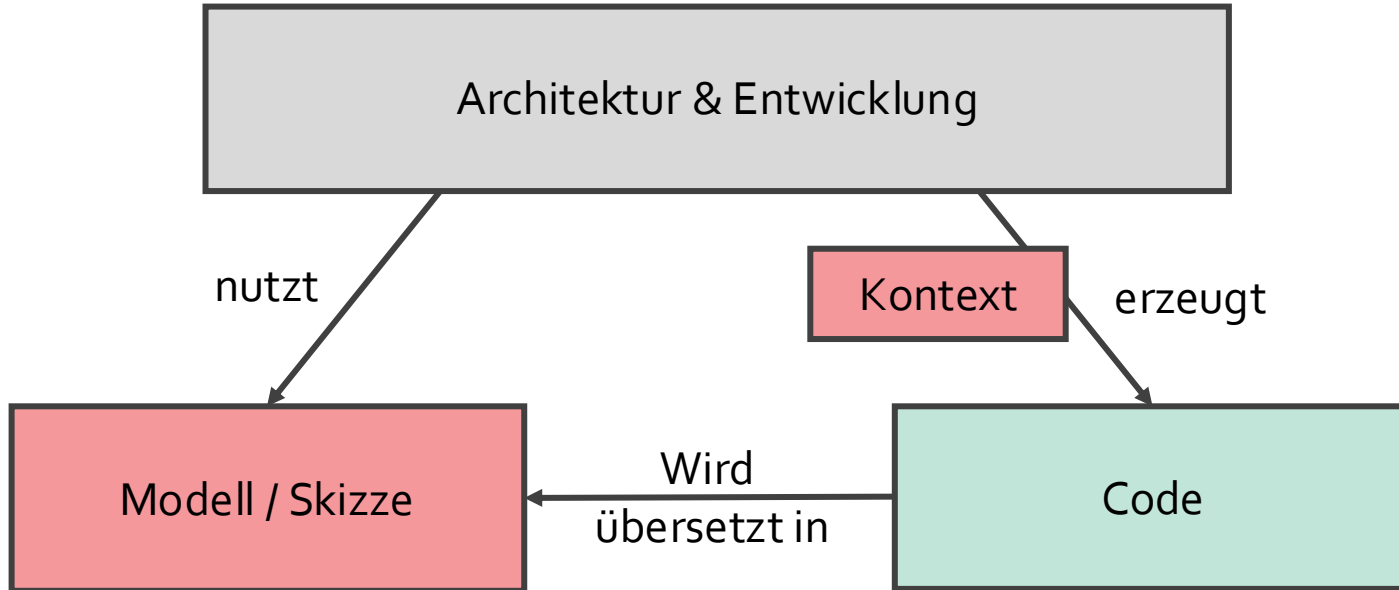
- Entitätsbeziehungen
- Methodenaufrufe
- Endpunkte
- Gemeinsame Änderung
- DB-Beziehungen
- Remote Methodenaufrufe
- Config-Dependencies
- Semantische Nähe
- ...



# Generative KI in der Entwicklung



# Agentic: Architektur und Entwicklung



# Aktuell als RC veröffentlicht: AGENTA

## ISAQB Advanced Level Curriculum AGENTA

### Status

CI - Releases and Main **passing** last commit last friday contributors 6 issues 8 open issues 36 closed

This is [copyrighted work](#).

### Content

This repository contains the ISAQB Advanced Level curriculum AGENTA in AsciiDoc.

### Released documents

The latest released HTML and PDF documents (German and English) are published via GitHub Pages at [public.isaqb.org/curriculum-agenta](https://public.isaqb.org/curriculum-agenta).

They are produced by CI on each release.

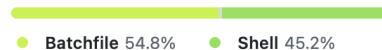
### How to build locally

Prerequisite: [Docker](#). Nothing else — no JDK, no Ruby.

### Contributors 7



### Languages



Generated from [isaqb-org/advanced-template](https://github.com/isaqb-org/advanced-template)

Architektur für Agentic Engineering Kontexte

<https://github.com/isaqb-org/curriculum-agenta>

# 05.

## Abschluss

Weitere Informationen...



# Folien als PDF zum Download

The screenshot shows the embarc.de website. The header includes the embarc logo and navigation links: Leistungen, Themen, Impulse, Aktuelles, Über uns. The main heading is 'Downloads und Medien' with the subtitle 'Unsere Folien, Videos, Artikel und Beiträge'. A yellow fuel can with a lightbulb icon is featured. Below this, there are filters for 'Vortragssfolien | Videos | Podcasts | Artikel | Alle' and a link to 'Hier gehts zu unseren Architekturspickern →'. The content area displays four PDF download cards:

- Alles Wichtige über Softwarearchitektur**  
Alles Wichtige über Softwarearchitektur in 45 Minuten  
Stefan Zörner  
pdf
- What can you expect from this job?**  
"As much architectural work as needed" beim Software Architecture ...  
Falk Sippach  
pdf
- Ein schlankes Review für Dein Software-System**  
Ein schlankes Review für Dein Software-System  
Stefan Toth, Stefan Zörner  
pdf
- Dynamische Entwicklungsorganisationen in Zeiten von Cloud**  
Dynamische Entwicklungsorganisationen in Zeiten von Cloud  
Kim Nena Duggen, Alexander Kaserbacher  
pdf



→ [embarc.de/download/](https://embarc.de/download/)

# 04/2025

## Vorgehensmuster für Softwarearchitektur

Kombinierbare Praktiken in Zeiten von Agile  
und Lean

**Stefan Toth**

4. Auflage  
Hanser Verlag



# AGILA – Agile Softwarearchitektur



▶ TRAININGS

▶ THEMEN

TRAINER:INNEN

▶ LEISTUNGEN

NEWS

KONTAKT

DE | EN

Home > Trainings > Agile Softwarearchitektur

## Agile Softwarearchitektur

Training iSAQB® CPQA®-Advanced AGILA – 3 Tage

Technik – Methodik 20 Kommunikation 10

### Vor-Ort-Termine

< Anderen Termin auswählen

**AGILA (Wien) - Stefan Toth -  
Deutsch**

14.-16. Dezember 2026

**Early Bird Ticket (AT)**

50% Rabatt ab der 2. Teilnehmerin (Anzahl im  
Warenkorb wählen).

<https://www.socreatory.com/de/trainings/agila>

# Feedback & Fragen?

Ich freue mich auf Fragen,  
Diskussionen, Pizza!

