Architektur von Informationssystemen

Hochschule für angewandte Wissenschaften

Sommersemester 2016

Nils Löwe / nils@loewe.io / @NilsLoewe

Über mich

- Jahrgang 1982
- Dipl. Ing (FH) - Technische Informatik *FH Wedel*
- MSc. Systems Engineering *Fernuni Hagen*
- Vollzeit-Softwareentwickler seit 2006
- Selbstständig seit 2011

Hinter den Kulissen

Friederike Löwe

- BEng. - Informations- und Elektrotechnik *HAW Hamburg*
- Vollzeit-Softwareentwicklerin seit 2011
- iSAQB - Certified Software Architect
- Aktuell in Elternzeit

Über Sie

- Bachelor 5. Fachsemester: **Angewandte Informatik**
- Software Engineering I & II
- Programmiermethodik I & II
- BWL I & II
- Programmiersprachen: Java, Erlang, Ruby
- ...

Ziele dieser Vorlesung

- Was ist Softwarearchitektur und wozu braucht man sie?
- Was muss ein Softwarearchitekt / eine Softwarearchitektin können?
- Wie konzipiert man eine große Anwendung?
- Wie werden Architekturen entworfen, dokumentiert und bewertet?

Keine Ziele dieser Vorlesung

- Der alleingültige Masterplan für **die** Architektur
- Details von .NET, JEE, Rails, ... beschreiben
- Eine weitere UML-Vorlesung

I graduated in Computer Science in the early 2000s.

When I took a Databases class, NoSQL didn't exist.
When I took a Computer Graphics class, OpenGL didn't support shaders.
When I took a Computer Security class, no one knew about botnets yet.
When I took an Artificial Intelligence class, deep learning didn't exist.
When I took a Programming Languages class, reactive programming wasn't a "thing".
When I took a Distributed Systems class, there was no Big Data or cloud computing.
When I took an Operating Systems class, hypervisors didn't exist (in PCs at least).
When I took a Networking class, there was no wifi in my laptop or internet in my phone.

Learn the fundamentals. The rest will change anyway.

Hisham @hisham_hm · 13. Dez. 2015
I felt like saying this.

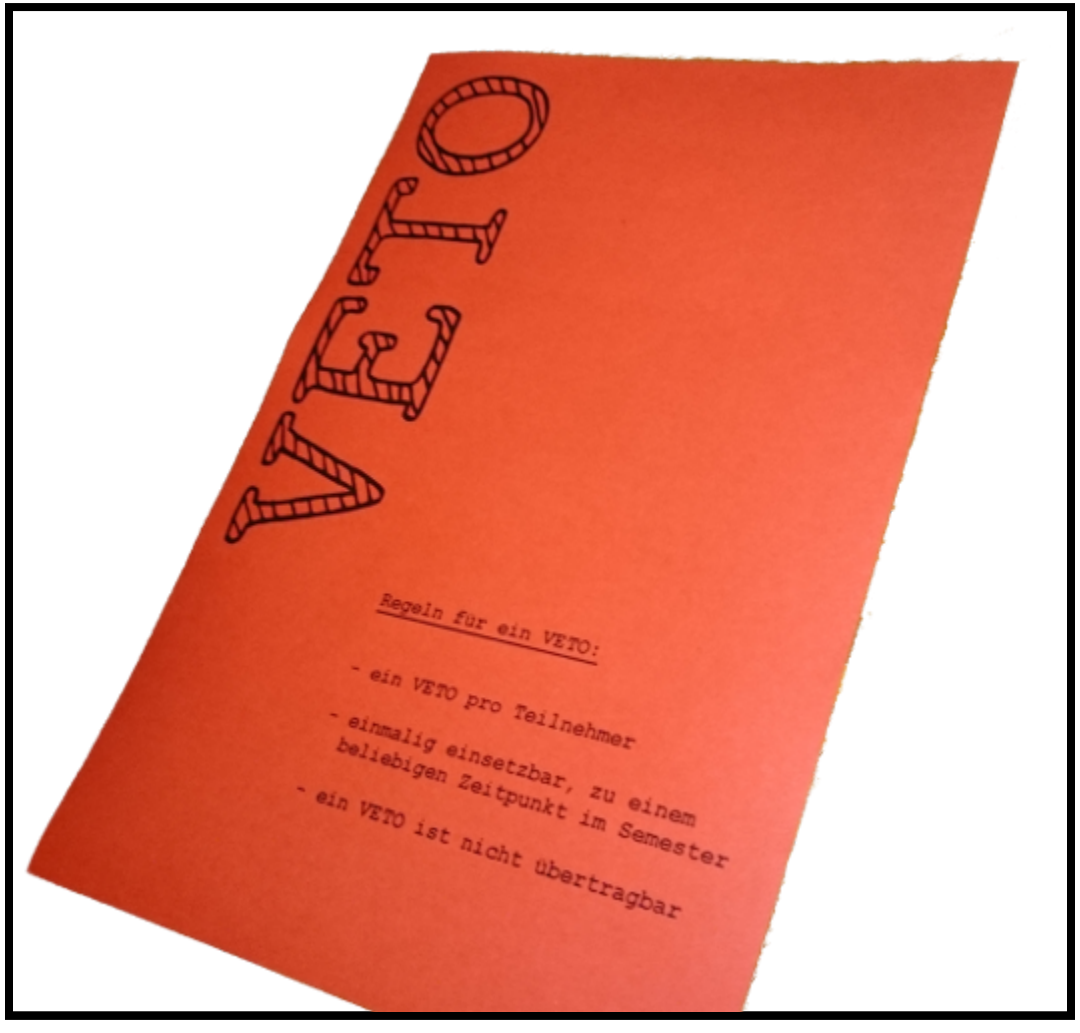
4,7 Tsd. 3,9 Tsd.

Organisation

- Unterlagen: ai2016.nils-loewe.de
- Klausur am Semesterende **ohne** Hilfsmittel

Praktikum

- Bilden Sie Teams aus 3 Personen
- Das erste Praktikum findet in KW 13 statt (nach Ostern)
- Während der Praktikumstermine präsentieren die Teams ihre Ergebnisse in ca. 20-25 Minuten pro Team. Eine Bearbeitung der Aufgaben findet dort nicht statt!
- Die Vorbereitungszeit für ein Praktikum beträgt idr. drei Wochen. Nutzen sie diese Zeit ruhig :)
- Aufgaben: ai2016.nils-loewe.de


VETO
Regeln für ein VETO:
- ein VETO pro Teilnehmer
- einmalig einsetzbar, zu einem beliebigen Zeitpunkt im Semester
- ein VETO ist nicht übertragbar

Literatur

Effektive Software-Architekturen - Ein praktischer Leitfaden *(Starke)*

Knigge für Software-Architekturen *(Hruschka, Starke)*

Software Architecture in Practice *(Bass, Clements, Kazman)*

Pattern-Oriented Software Architecture *(Buschmann et.al)*

Systemarchitekturen für verteilte Anwendungen *(Dunkel et.al)*

Software Architecture and Design Illuminated *(Qian, Fu)*

Handbuch der Software-Architektur *(Reussner, Hasselbring)*

Software Architecture Foundations, Theory and Practice *(Taylor, Medvidovic, Dashofy)*

Software-Architekturen dokumentieren und kommunizieren *(Zörner)*

Themen der Vorlesung

Was ist Softwarearchitektur?

Geschichte und Trends

Sichten auf Architekturen

Qualiät und andere nichtfunktionale Anforderungen

Architekturmuster

Dokumentation von Architekturen

Technologien und Frameworks

Was ist Softwarearchitektur?

- Motivation & Abgrenzung
- Architektur vs. Entwurf
- Architektur vs. Softwareengineering
- Aufgaben und Rollen von Architekten

Geschichte und Trends

- Großrechner mit Thin-clients
- Personalcomputer
- Web / Smartphones / Tablets -> Browser als OS
- Internet of Things

Sichten auf Architekturen

- Was sind Sichten?
- Statische vs. Dynamische Sichten
- UML als Anwendung der Sichten

Qualiät und andere nichtfunktionale Anforderungen

Was ist

- eine "gute" Architektur?
- Skalierbarkeit?
- Performance?
- Sicherheit
- Wartbarkeit?
- Testbarkeit?

Architekturmuster

- Was sind Architekturmuster?
- Architekturmuster vs. Designpatterns?
- Vorstellung gängiger Architekturmuster

Dokumentation von Architekturen

- Warum dokumentieren?
- Wie dokumentieren?
- Für wen dokumentieren?
- Best practices?
- Tools zur Dokumentation
- Wie lese ich Architekturdoku?

Technologien und Frameworks

- Was sind Architketurframeworks?
- Vorteile/Nachteile von Frameworks?
- Beispiele für Frameworks: *Ruby on Rails*, *.NET*, *Twitter Bootstrap*
- Kriterien für die Auswahl eines Frameworks

Was ist Softwarearchitektur?

Geschichte und Trends

Sichten auf Architekturen

Qualiät und andere nichtfunktionale Anforderungen

Architekturmuster

Dokumentation von Architekturen

Technologien und Frameworks

Was ist Architektur?

- Der Begriff stammt aus dem Mittelalter
- Oberbegriff für Baustil / Baukunst
- Ziel: Ordnung und Generalisierung struktureller Beziehungen in Produkten des Bauwesens
- Ziel: Erfahrung und Wissen verallgemeinern.

Architektur zur Verteidigung

Architektur zur Verteidigung

Architektur um Macht zu zeigen

Architektur um Macht zu zeigen

Architektur um Verkehr zu lenken

Architektur um mobil zu sein

Definitionen für Softwarearchitektur

Die Architektur eines Softwaresystems ist die Menge der Haupt-Designentscheidungen über das System. *(Taylor)*

(“A software system’s architecture is the set of principal design decisions about the system.”)

Die Software-Architektur ist die grundlegende Organisation eines Systems, dargestellt durch dessen Komponenten, deren Beziehungen zueinander und zur Umgebung sowie die Prinzipien, die den Entwurf und die Evolution des Systems bestimmen. *(Reussner)*

Software Architecture = { what, how, why } *(Perry and Wolf)*

The software architecture of deployed software is determined by those aspects that are the hardest to change. *(Chris Verhoef)*

Noch mehr Definitionen von Softwarearchitektur auf
http://sei.cmu.edu/architecture/start/glossary/community.cfm

- Aber was heißt das denn praktisch?

Architektur besteht aus Strukturen

- die Komponenten (Bausteine), aus denen ein System besteht
- die wesentlichen (extern sichtbaren) Eigenschaften dieser Komponenten
- die Beziehungen der Komponenten untereinander

Architektur beschreibt eine Lösung im Sinne eines Bauplans

(die Architektur eines Gebäudes besteht aus einer Sammlung von Plänen - nicht aus Steinen und Zement)

Erst die *Implementierung* macht aus den Komponenten und Schnittstellen der Architektur ein reales System.

Architektur basiert auf Entwurfsentscheidungen

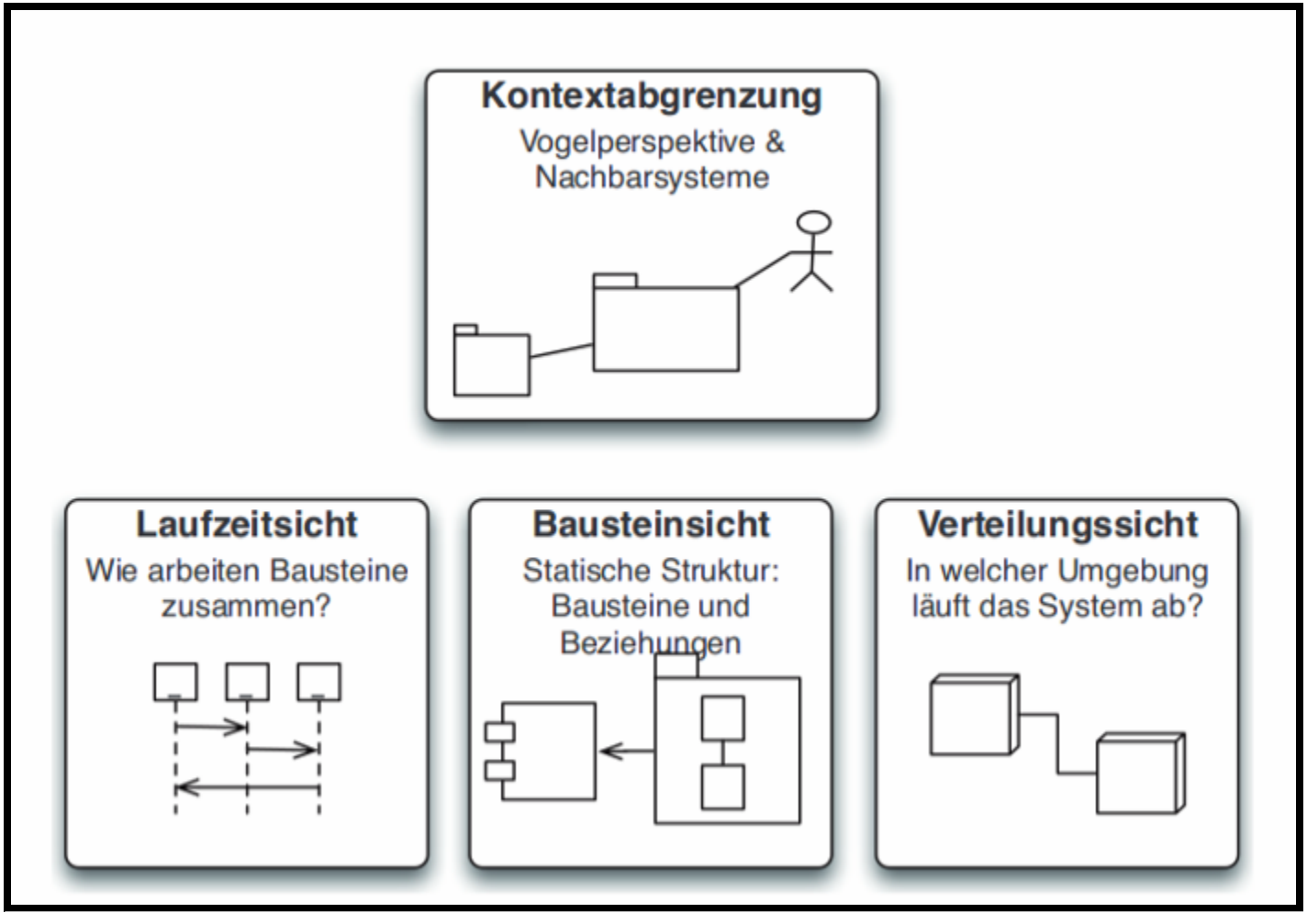
- Entscheidungen zum Entwurf der Komponenten
- Entscheidung für eine bestimmte Technologie
  Die Konsequenz vieler Entscheidungen können Architekten erst sehr viel später beurteilen!

Architektur bildet den Übergang von der Analyse zur Realisierung

- Analysephase → Architektur → technische Realisierung
- Fachdomäne → Architektur → Umsetzung in Software

Architektur besteht aus verschiedenen *Sichten*

- jede Sicht dokumentiert einzelne Aspekte des Gesamtsystems
- jede Sicht ist für bestimmte Stakeholder nützlich



Bildquelle: Starke / "Effektive Softwarearchitekturen" (5. Auflage)

Architektur schafft Verständlichkeit

- komplexe Anforderungen → geordnete Strukturen
- angemessene und problembezogene Dokumentation

Management: Anforderungen erfüllbar / erfüllt?

neue Mitarbeiter: Systemstruktur kennen lernen

Wartungsteams: betroffene Bestandteile leichter finden und Folgen von Änderungen abschätzen

Systembetreiber: Welche Software-Komponenten laufen auf welchen physischen Systemen ab?

Architektur ist der Rahmen für flexible Systeme

- stellt Flexibilität und Erweiterbarkeit sicher → *"framework for change"* (Tom DeMarco)

Architektur ist Abstraktion

- Essenzielle Aufgabe von Architekten: Weglassen von nicht benötigten Informationen
- Informationen werden bewusst gefiltert um die Darstellung lesbar und verständlich zu halten

Architektur schafft Qualität

Die Qualität eines Systems bezeichnet die Summe seiner nicht-funktionale Eigenschaften:

- Performance
- Verständlichkeit
- Flexibilität
- ...

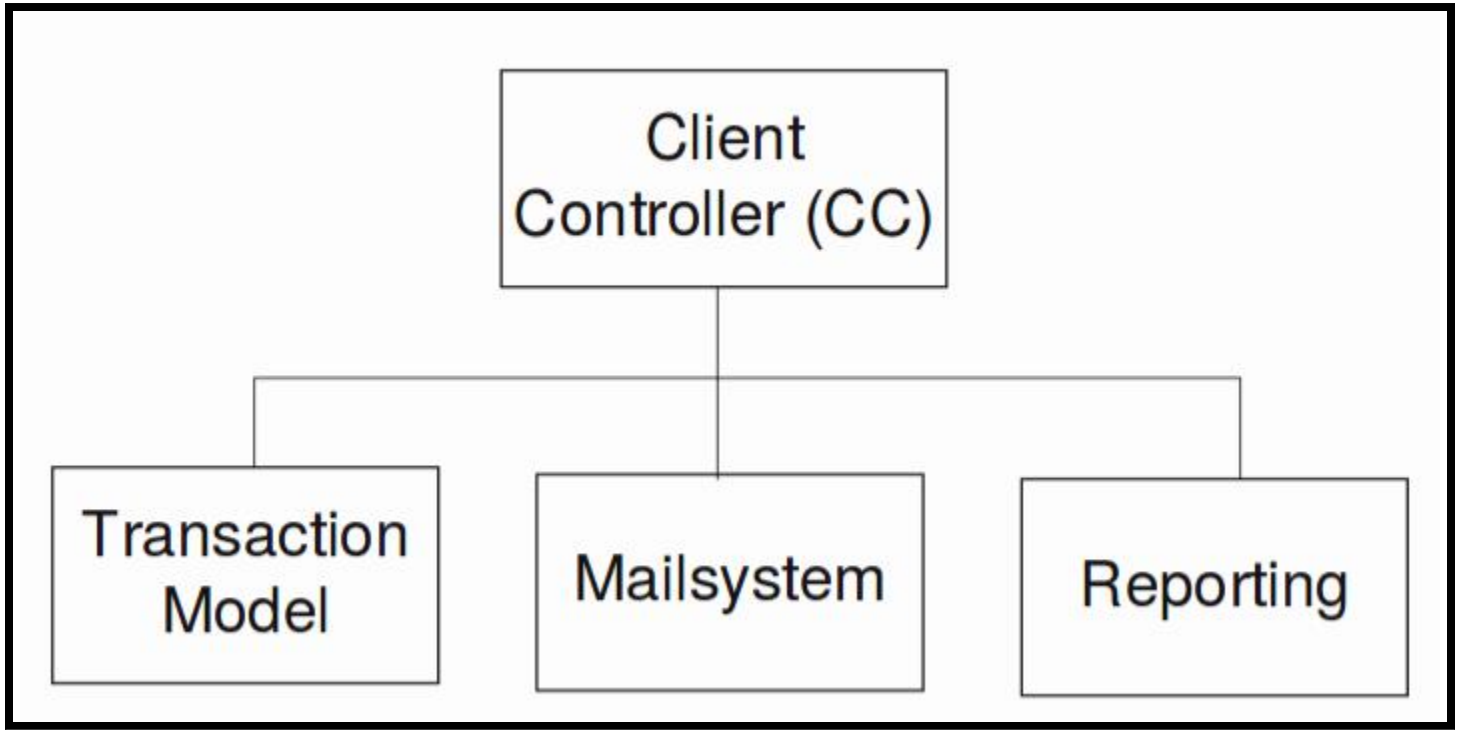
Das sind meistens die schwierigen Anforderungen!

Architektur vs. Entwurf/Design?

- die Grenze ist fließend
- Design (oder Entwurf) bezeichnet den Prozess der Erstellung der Architektur

→ Gehen Sie mit diesen Begriffen pragmatisch um und suchen Sie nicht nach einer "formalen" Definition.

Was Softwarearchitektur *nicht* ist...



Bildquelle: Starke / "Effektive Softwarearchitekturen" (5. Auflage)

Wenn Ihnen jemand eine Architektur-Darstellung vorlegt, stellen Sie folgende Fragen (die in einer guten Dokumentation immer beantwortet sind):

- Welche Verantwortlichkeiten (*responsibilities*) hat jees der Kästchen und Verbindungslinien im Diagram?
- Für jede Verbindungslinie: Warum existiert sie und welche Semantik oder Bedeutung hat sie?
- Was wird zu welchem Zeitpunkt auf welche Weise über diese Verbindungen transportiert?

Die Aufgaben von Softwarearchitekten

"Das Leben von Software-Architekten besteht aus einer langen und schnellen Abfolge suboptimaler Entwurfsentscheidungen, die meist im Dunkel getroffen werden."

(Phillipe Kruchten)

Architekten konstruieren und entwerfen

- Komponenten: Verantwortlichkeiten definieren
- Schnittstellen: "Verträge" beschreiben, auf deren Basis die Komponenten miteinander arbeiten (*design by contract*)
- Strukturen: Komponenten + Zusammenspiel → statische und dynamische Strukturen

Architekten entscheiden

"...schnelle Folge suboptimaler Designentscheidungen"

- Welche Bausteine?
- Welche Schnittstellen?
- Welche Abläufe?
- Welche technischen Frameworks?
- Selbst implementieren, kaufen oder ein Mittelweg davon?
- Welches Teilteam entwickelt welche Komponenten?
- Wie sollen die Bausteine der Architektur heißen? (Aussagekräftige Namen sind wirklich, wirklich wichtig!)

"...Entwurfsentscheidungen, die meist im Dunkel getroffen werden"

- teilweise zeigt sich erst Monate oder Jahre (!) später, ob eine Architekturentscheidung vernünfigt, angemessen oder sinnvoll war
- oft haben Architekten mit Frameworks, Betriebssystemen oder sonstigen Dingen zu tun, deren genaues Verhalten sie gar nicht kennen können

→ Dabei hilft iteratives Vorgehen erheblich weiter!

→ Entscheidungen großer Tragweite sollten Sie angemessen dokumentieren!

Architekten garantieren die Erfüllung von Anforderungen

- Machbarkeit von Anforderungen sicherstellen (z.B. durch Protoypen)
- dafür sorgen, dass die Anforderungen auch erfüllt werden (gilt für funktionale und nicht-funktionale Anforderungen und Randbedingungen)
- einen angemessenen Kostenrahmen einhalten

Architekten beraten

- Management und Auftraggeber bei der Projektplanung und -Organisation
- Auftraggeber und Analyseteams zu Machbarkeit, Kosten/Nutzen, Auswirkungen von Anforderungen, Realisierung und Betrieb
- Projektleiter bei der Organisaton und Steuerung des Implementierungsteams
- Projektleiter beim Managment (technischer) Risiken
- das Implementierungsteam bei der Umsetzung
- Hardware-Architekten und Systembetreiber hinsichtlich Hardware-Anforderungen
- die Qualitätssicherung über Kritikalität und Testbarkeit von Systembestandteilen

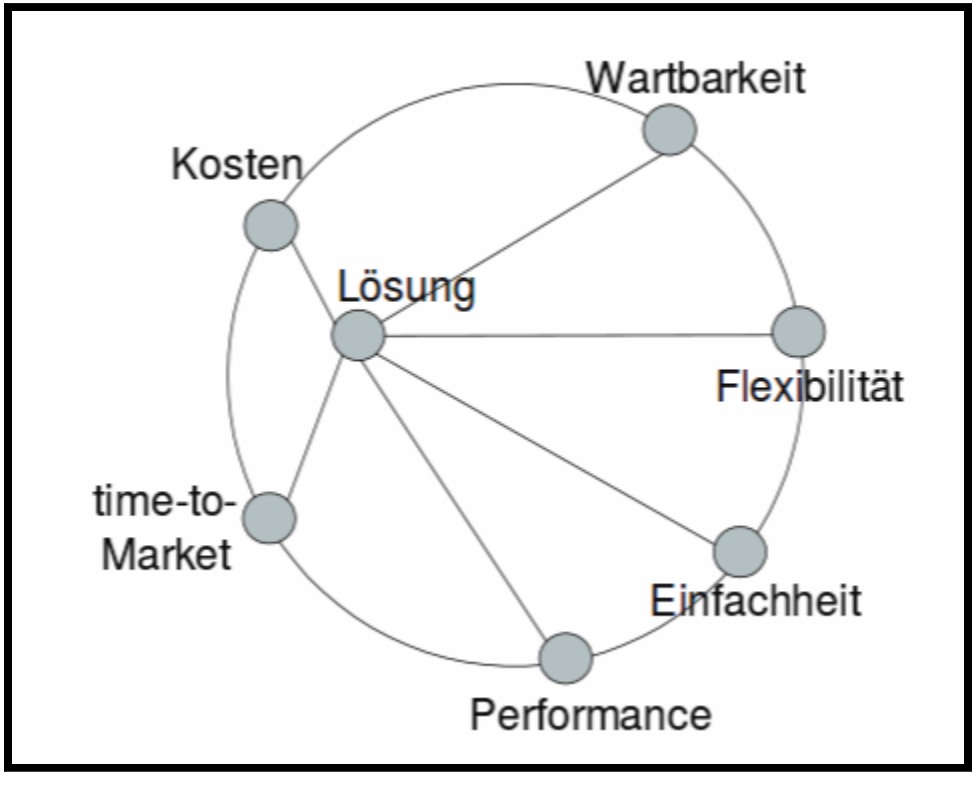
Architekten dokumentieren - angemessen

- an den Bedürfnissen der Adressaten orientieren
- pragmatisch arbeiten (manchmal reicht eine Skizze auf einem alten Umschlag)

Die Projekte sollen agil, flexibel und kurzfristig wandlungsfähig bleiben!

Architekten sind Diplomaten und Akrobaten

Sie schließen Kompromisse zwischen widersprüchlichen oder kunkurrierenden Forderungen.



Bildquelle: Starke / "Effektive Softwarearchitekturen" (5. Auflage)

Architekten vereinfachen Strukturen

- leichter und günstiger realisierbar
- einfacher verständlich
- weniger fehleranfällig

"Die zuverlässigste, preiswerteste und robusteste Komponente eines Systems ist diejenige, die erst gar nicht realisert werden muss!"

(Gernot Starke)

Architekten kommunizieren

- Architekturentscheidungen für unterschiedliche Stakeholder angemessen aufbereiten
- Stakeholder von Entscheidungen überzeugen (präsentieren + vermarkten)
- Team coachen und unterrichten

Architekten bewerten die Güte der Architektur

An welchen Stellen des Systems sind nicht-funktionale Anforderungen (z.B. Performance) riskant oder kritisch?

→ Maßnahmen zur Optimierung oder Risikominderung ableiten

Architekten brauchen Mut

Aus Zeitgründen verfügen Architekten oft nicht über genügend Informationen, um optimale Entscheidungen zu treffen.

→ Mut für möglicherweise suboptimale Entscheidungen aufbringen - damit das Projekt weiter laufen kann...

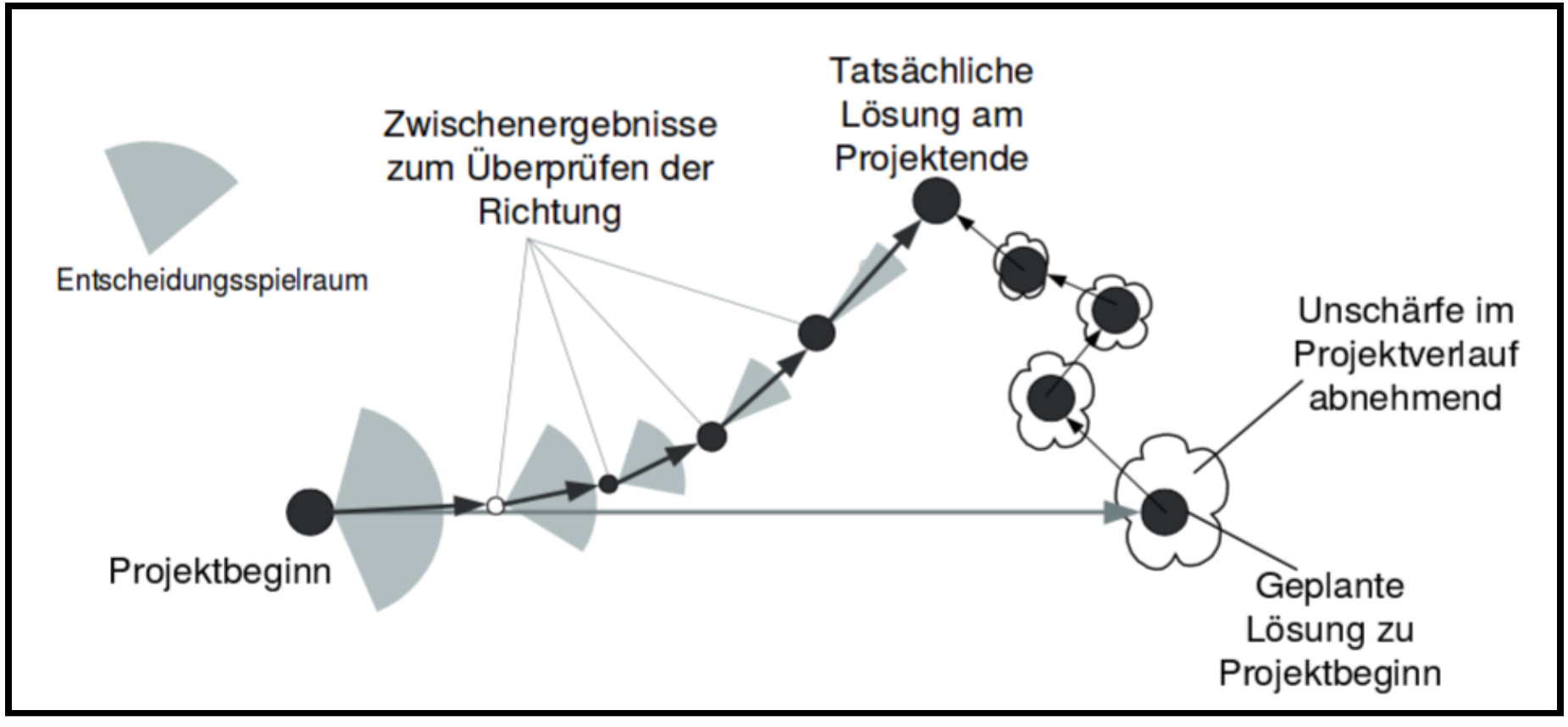
Damit ist nicht Waghalsigkeit gemeint! Bewusste Risikoabwägung, Prüfen von Alternativen und Beachtung der Konsequenzen gehört immer dazu!

Die Werkzeuge von Architekten

- Modelle (vereinfachte Abbildungen der Wirklichkeit)
- System-Dokumentationen (zur Kommunikation mit anderen Projektbeteiligten)
- Heuristiken (Erfahrungen, Regeln, Tipps)
- Muster (Vorlagen für elegante Lösungen zu spezifischen Entwurfsproblemen)
- Partitionierung (Problem in Teilprobleme zerlegen)
- Aggregation (Einzelteile zu Software-Systemen zusammensetzen)
- Iteration (zyklische Vorgehensweise und kurzfristiges Feedback)
- Compiler, Debugger, Prototypen (Implementierung unterstützen, Machbarkeit testen, technische Risiken prüfen)

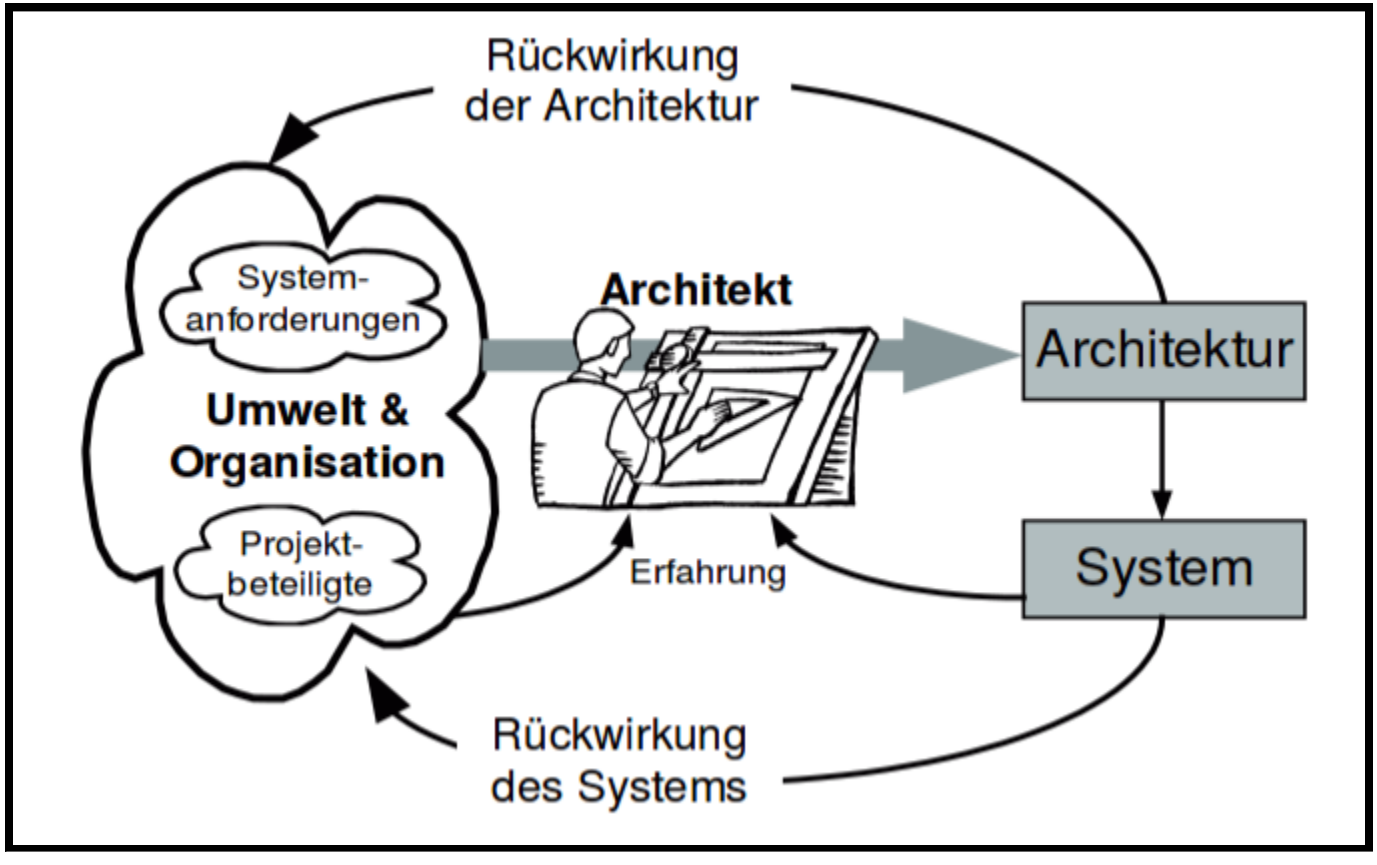
Wie entstehen Architekturen?

Architekturen entstehen in Zyklen und Iterationen



Bildquelle: Starke / "Effektive Softwarearchitekturen" (5. Auflage)

Conway's Law



Bildquelle: Starke / "Effektive Softwarearchitekturen" (5. Auflage)

Architekturen entstehen in kleinen Teams

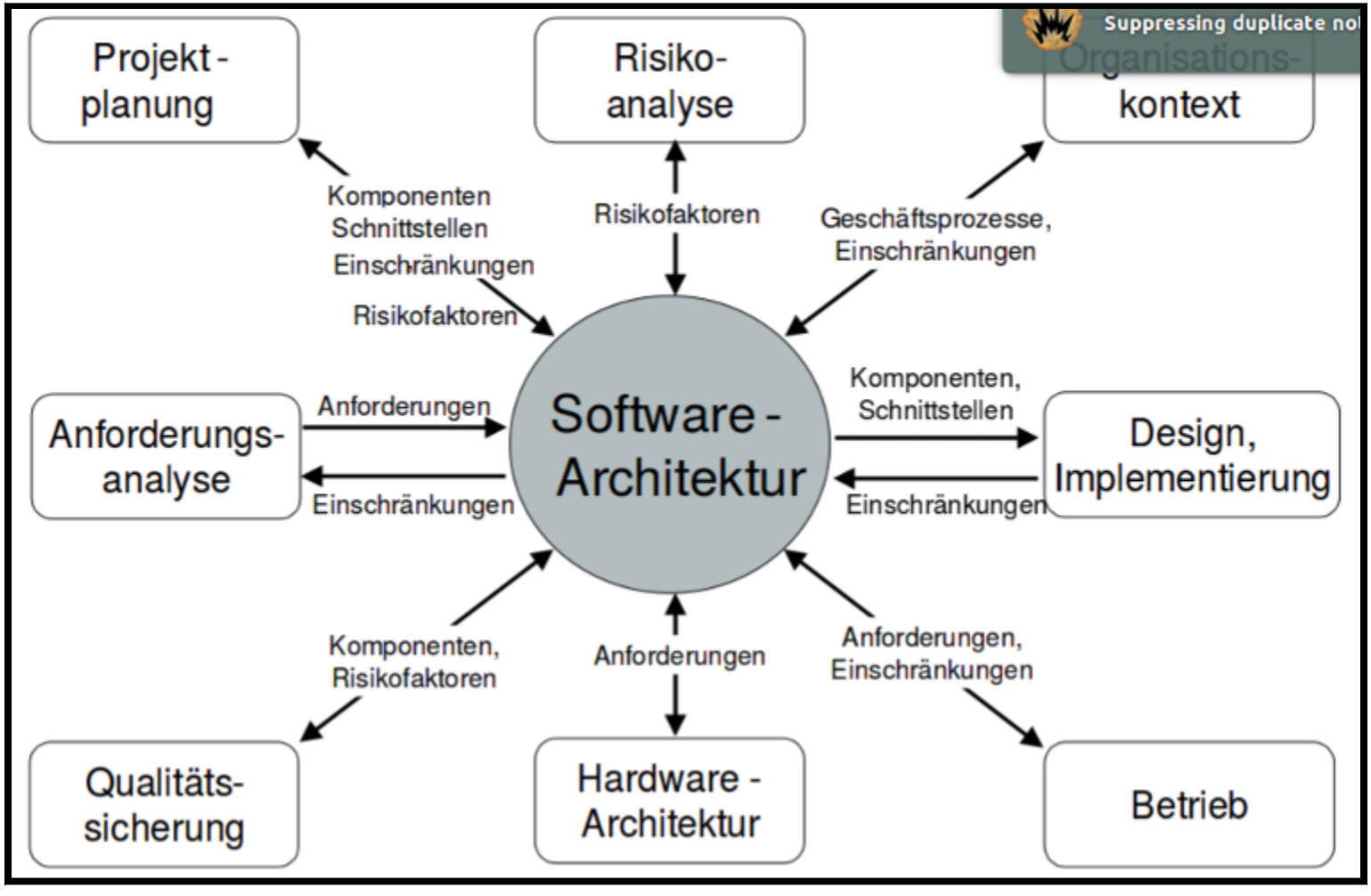
- gemeinsames Ziel: Die Anforderungen des Kunden erfüllen
- Erfahrung in Software-Engineering
- Erfahrung in der Fachdomäne

Wie Architekturen nicht entstehen sollten

- im Architekturkomitee, das jeden Donnerstag um 15 Uhr im großen Besprechungsraum tagt
- im Elfenbeinturm ohne Kunden, Auftraggeber, Projektleitung und Realisierungsteam
- ausschließlich auf bunten Marketing-Folien
- "Wir machen jetzt {Name-der-Technologie}!"

Ist *Softwarearchitekt/in* ein Status?

In welchem Kontext steht Architektur?



Bildquelle: Starke / "Effektive Softwarearchitekturen" (5. Auflage)

Was ist Softwarearchitektur?

Geschichte und Trends

Sichten auf Architekturen

Qualiät und andere nichtfunktionale Anforderungen

Architekturmuster

Dokumentation von Architekturen

Technologien und Frameworks

Fragen?

Unterlagen: ai2016.nils-loewe.de