Der Software Lifecycle

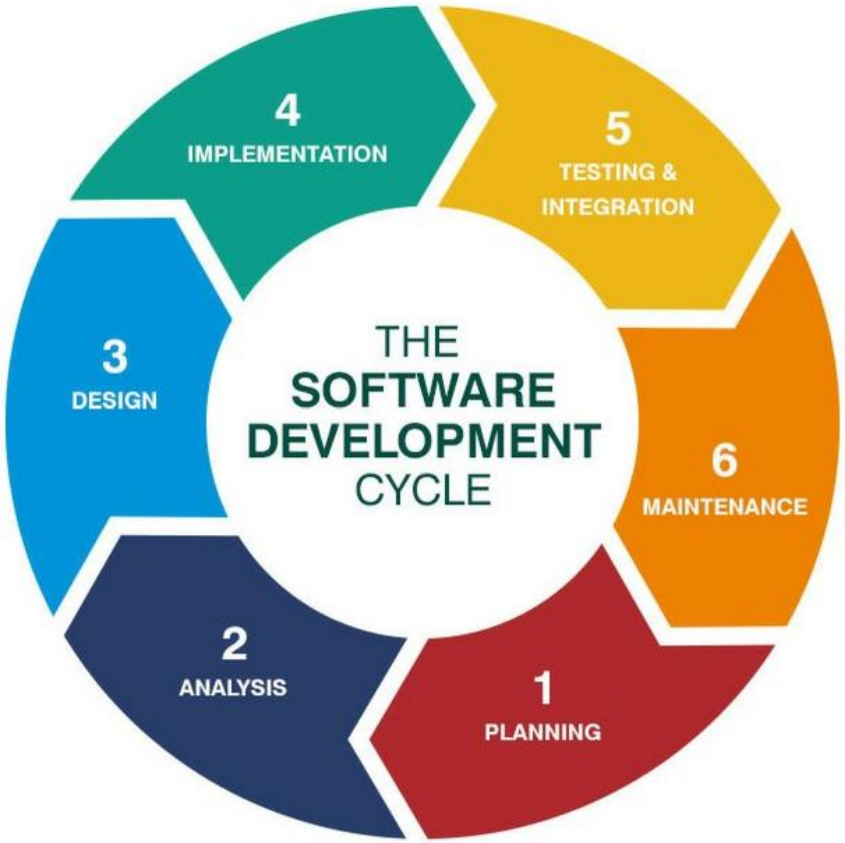
Teil 2

In welchen Phasen des Projekts spielen Komponenten eine Rolle?

- Eigentlich überall
  - von „Wie teile ich meine Komponenten überhaupt ein?“
  - …
  - bis „Wie warte ich meine Komponenten?“
- Nicht zu früh in die Tasten greifen → Planung ist wichtig
- Eine andere Definition von Software-Architektur:
  - *“Software architecture is those decisions which are both important and **hard to change**. […] Said another way, software architecture is those decisions which, if made poorly, will make a project either **succeed or fail**, in a needlessly expensive way.”*, Martin Fowler
- Daher Betrachtung in allen Phasen eines Software-Development-Life-Cycle (SDLC)

Was ist ein Software-Development-Life-Cycle?

- Besteht aus mehreren Phasen



SDLC Phasen

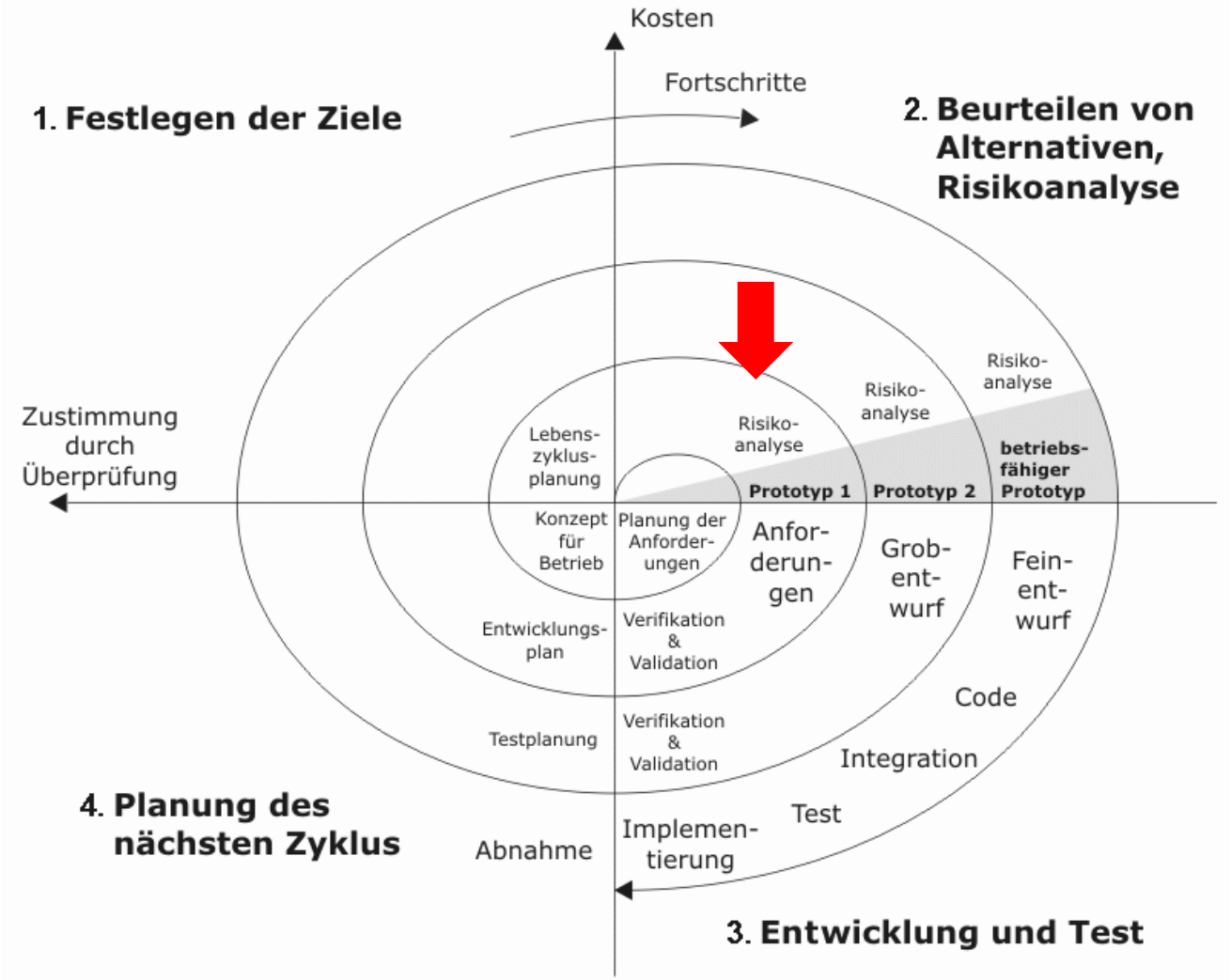
- **Planning**: Develops a project management plan and other planning documents. Provides the basis for acquiring the resources needed to achieve a solution.
- **Requirement analysis**: Analyzes user needs and develops user requirements. Create a detailed functional requirements document.
- **Design**: Transforms detailed requirements into complete, detailed Systems Design Document. Focuses on how to deliver the requirements document.

Quelle: Wikipedia

SDLC Phasen cont.

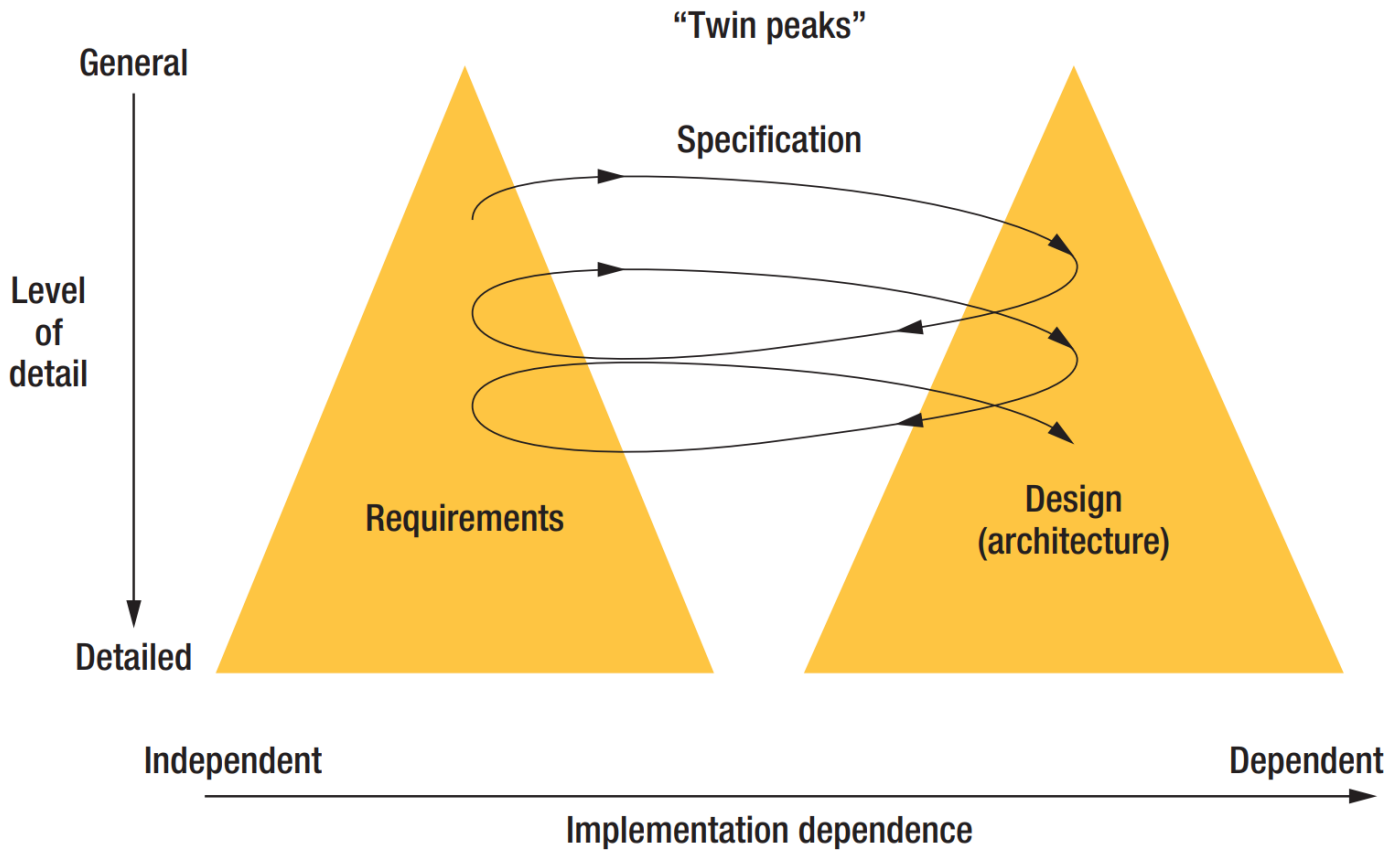
- **Development / Implementation**: Convert a design into a complete information system includes acquiring and installing system environment, creating and testing databases, preparing test case procedures, preparing test files, coding, compiling, refining programs, performing test readiness review and procurement activities.
- **Testing**: Demonstrates that develop system conforms to requirements as specific specified in the functional requirements document. Conducted by quality assurance staff and users. Produces test analysis reports.
- **Maintenance**: Describes tasks to operate and maintain information systems in a production environment. Includes post implementation and in-process reviews.

Nicht zwangsläufig Wasserfall … Iterationen!



Quelle: https://de.wikipedia.org/wiki/Spiralmodell

Wann eine Software-Architektur planen?

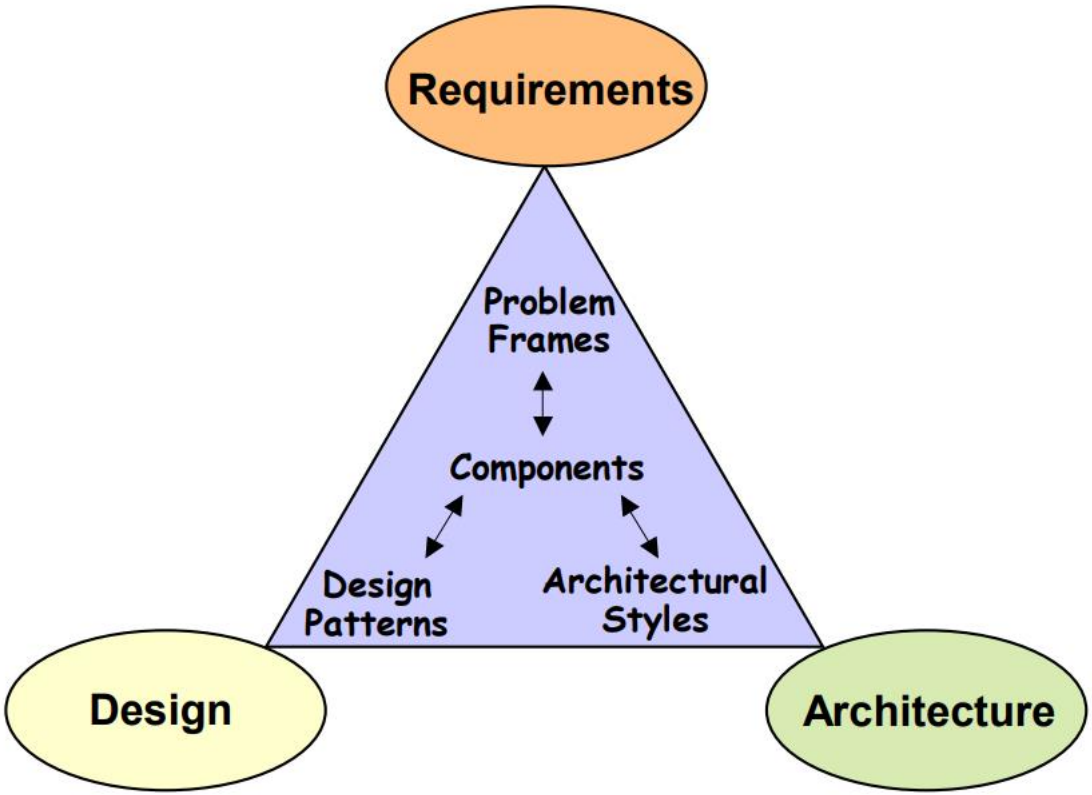


Cleland-Huang, J., Hanmer, R. S., Supakkul, S., & Mirakhorli, M. (2013). The twin peaks of requirements and architecture. IEEE Software, 30(2), 24-29.

Cont.

- Nicht alle Requirements (Anforderungen) sind von Anfang an bekannt
  - Stakeholder (Endbenutzer) haben neue Anforderungen, wenn sie die erste Version der Software testen / evaluieren → I'll Know It When I See It (IKIWISI)
- Somit macht es auch keinen Sinn, die Architektur am Projektbeginn in den Stein zu meißeln
- Qualitäts-Attribute (e.g. Performance, Security) können wiederrum e.g. die funktionale Anforderungen wieder beeinflussen / verfeinern
- Wechselwirkung zwischen Architektur-Möglichkeiten / -Einschränkungen und den Anforderungen der Stakeholder
- **Hauptreiber einer Softwarearchitektur (Bausteine sind u.a. Komponenten) sind nicht-funktionale Anforderungen**

Wie werden Komponenten geformt?



Nuseibeh, B. (2001). Weaving the software development process between requirements and architecture. From Software Requirements to Architectures (STRAW'01).

Cont.

- Was kann im Laufe der Zeit passieren? Praxisbeispiele:
  - Interfaces können sich durch ändernde Anforderungen ändern
  - Neue Services / Komponenten können durch neue Anforderungen entstehen
  - Es kann zu Komponenten „Splits“ kommen
  - Sich ändernde nicht-funktionale Anforderungen können ein Re-Design erfordern
  - Schnittstellendefinitionen (welche verwendet werden müssen) vom Dachkonzern können sich ändern (Auslöser: Anforderungen)
  - Stakeholder wünschen sich doch eine Smartphone oder Web-App
  - Daten aus Fremdsystemen müssen angezeigt werden (eventuell komplexe Aggregationen, Berechnungen erforderlich)
  - …

Ziel der Vorlesung

- Komponenten sind der essentiellste Baustein der Architektur
- In welchen Projektphasen wird die Architektur / Komponenten in welcher Form beeinflusst?