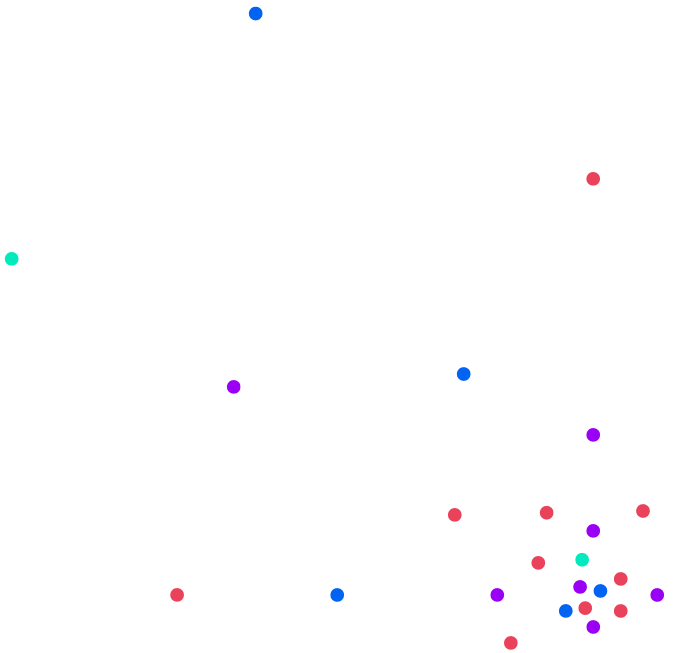
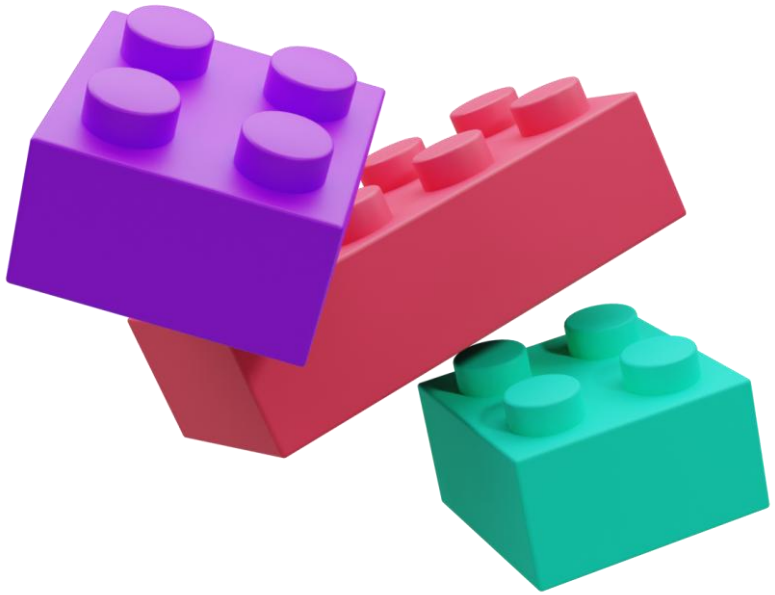
Wie kommt Software-Architektur ins Product-Backlog?

von Nils Kliesing



Agenda

- Software-Architektur
  - Was?
  - Wer?
- Anforderungen
- Umsetzung im Backlog



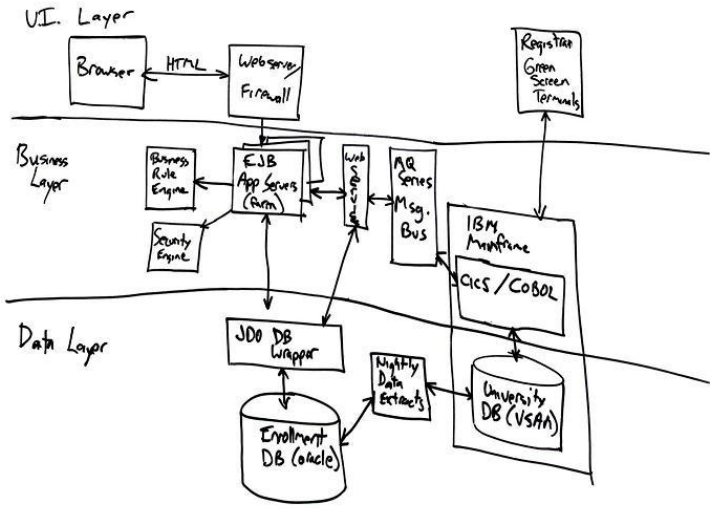
Kurz zu mir

Nils Kliesing

- 9 Jahre in Scrum Entwicklungsteams
- 2,5 Jahre Product Owner
- Zertifizierter Software-Architekt (iSAQB)

Was ist Software-Architektur?

- Definition 1/3*: Summe der wichtigen Entscheidungen!
  - Schwierig oder teuer rückgängig zu machen
  - Aufteilung in Unter-Module
  - Beziehungen zwischen Modulen
  - Technologien
    - Protokolle
    - Programmiersprachen
    - Frameworks
  - Prinzipien, Muster und Best Practices die Lösungsansätze zu bekannten Herausforderungen vorgeben

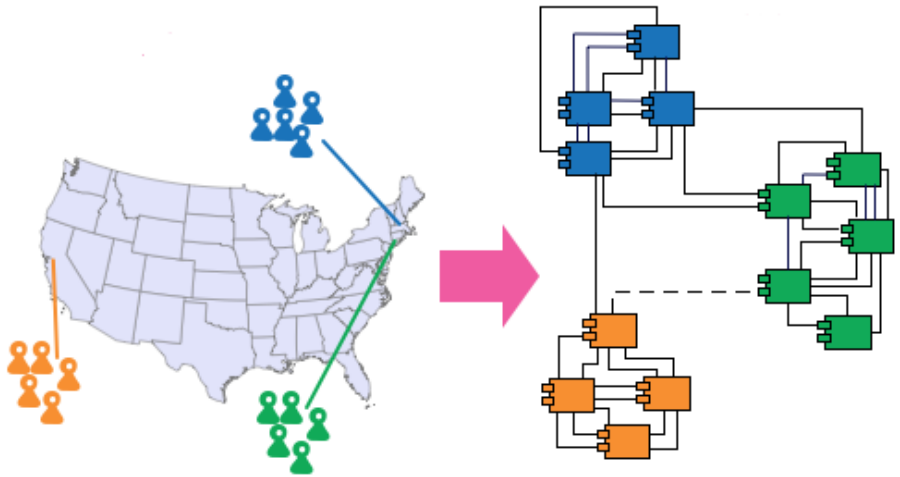


Bildquelle: https://agilemodeling.com/artifacts/freeform.htm

* Von vielen Definitionen wurden drei für diesen Vortrag ausgewählt

Was ist Software-Architektur?

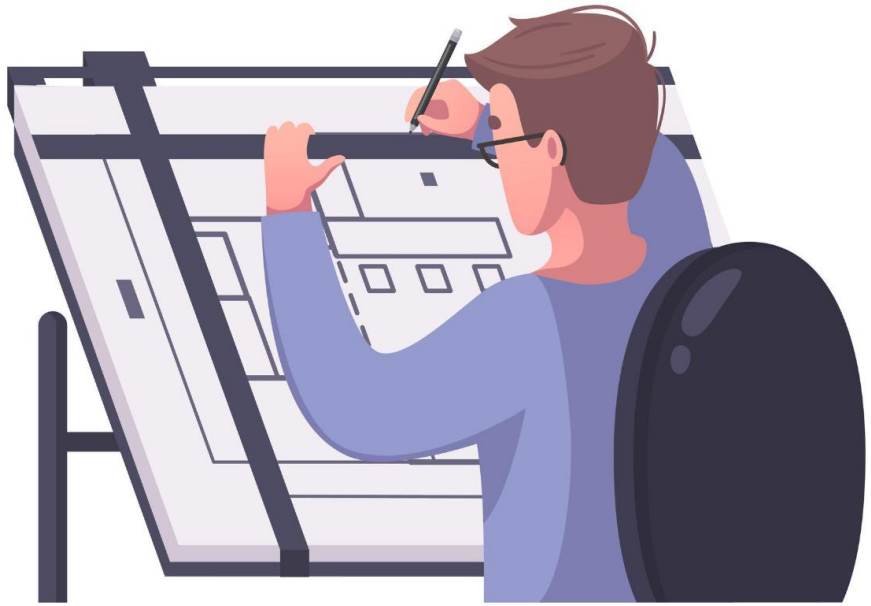
- Auch organisatorische Entscheidungen
  - z.B. Welches Team baut welches Modul
  - Beeinflusst die technische Umsetzung



Bildquelle: https://martinfowler.com/bliki/ConwaysLaw.html

Was ist Software-Architektur?

- Definition 2/3: Das, was der Software-Architekt macht!
  - Frage:
    Hat euer Team einen Software-Architekten?



Bildquelle: https://www.freepik.com/free-vector/male-architect-drawing-house-plan-cartoon_17232771.htm

Wer gestaltet Software-Architektur?

Kein expliziter Architekt

Architektur-Champions

Architecture Owner

„klassischer“ Architekt

Herausforderungen

- Großes Projekt
- Komplexe Domäne
- Geografisch Verteilt
- Neue Technologien
- Neues Team
- Fehlende Lösungsansätze
- Unerfahrenes Team
- Wenig Commitment
- Hierarchische Unternehmensstruktur
- Widersprüchliche Anforderungen

Grafiken durch KI generiert

Was ist Software-Architektur?

- Definition 3/3: Ermöglicht die Entwicklung des richtigen Produktes für die gegebenen Anforderungen

Z·I·E·L

WENN DU DAS ZIEL NICHT KENNST,
IST KEIN WEG DER RICHTIGE.

Bildquelle:
https://www.germanposters.de/
motivation-ziel--berg-.html

Anforderungen

Funktionale Anforderungen
aka Features

- Eigene Schätzung auswählen
- Anzeigen, wer schon abgestimmt hat

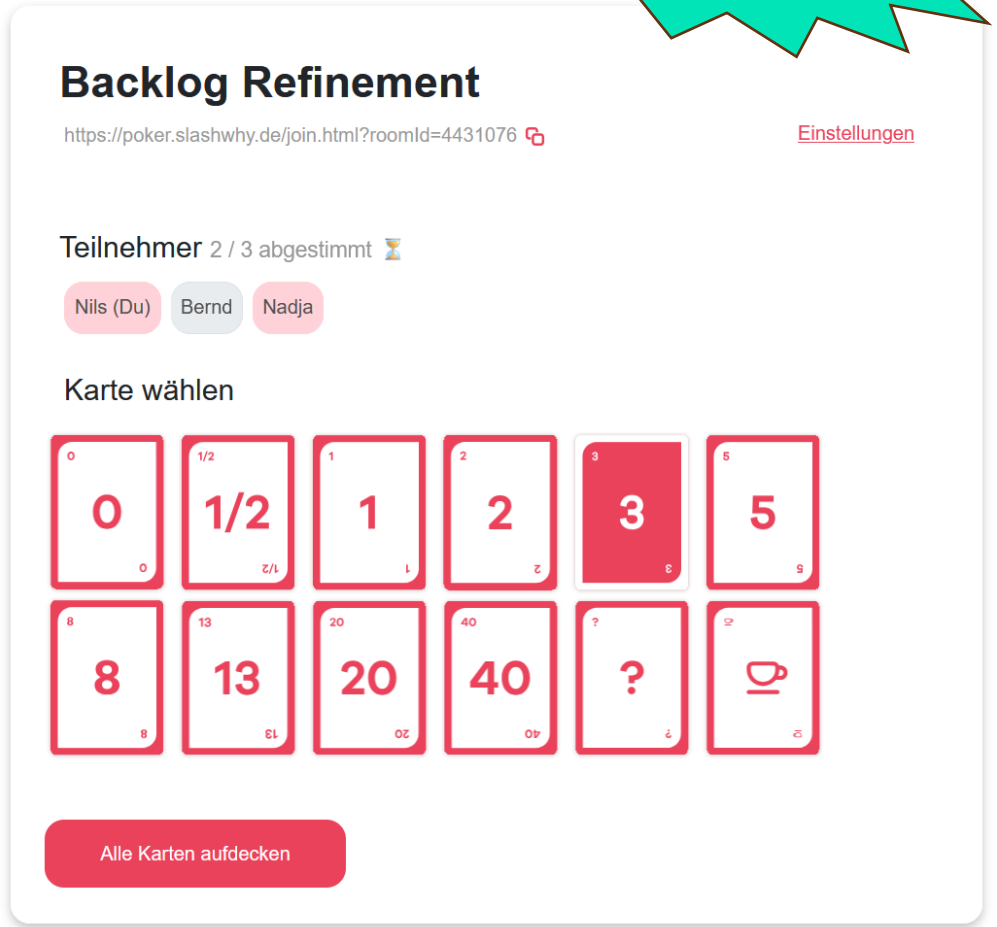
Randbedingungen
bei Entwicklung oder Betrieb

- Durch Azubi, fertigstellen im ersten Lehrjahr
- Ohne komplexe Frameworks

Qualitätsziele

- Intuitiv und minimalistisch
- Für Teamzusammen-setzungen bei /y optimiert

Beispiel Planning Poker
https://poker.slashwhy.de

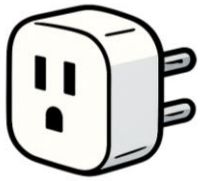


Qualitätsziele für mein Produkt

- Basierend auf Qualitätsmerkmalen aus ISO 25010:2023

Funktionale Eignung

Performance



Kompatibilität

Usability

Zuverlässigkeit

Security

Wartbarkeit

Flexibilität

Safety

- 3-5 festlegen und priorisieren
- Anschließend für das Produkt konkretisieren

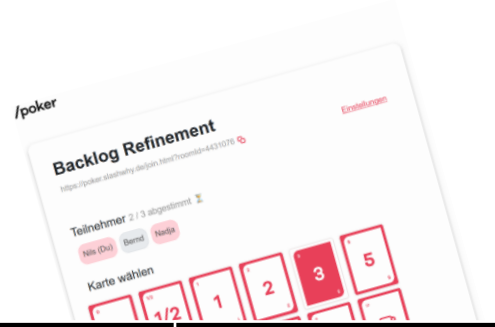
Grafiken durch KI generiert

Qualitätsziele finden und konkretisieren mit LASR

- **L**ightweight **A**pproach for **S**oftware **R**eviews*
- 4-stündiger Workshop mit wenigen Beteiligten
  - Remote oder vor Ort möglich
- Top 3-5 Qualitätsziele des Produktes identifizieren
  - Aus Diskussionen lernen, was die Qualitätsmerkmale konkret für das Produkt bedeuten
- Risiken identifizieren und Lücken in der Architektur ableiten
  - Startpunkt für Verbesserung des Produktes
- In slashwhy-Projekten unterstützen wir gerne mit diesem Workshop

* Mehr Details und Bildquelle unter: https://www.lasr-reviews.org/de/

Qualitätsszenarien für das Planning Poker



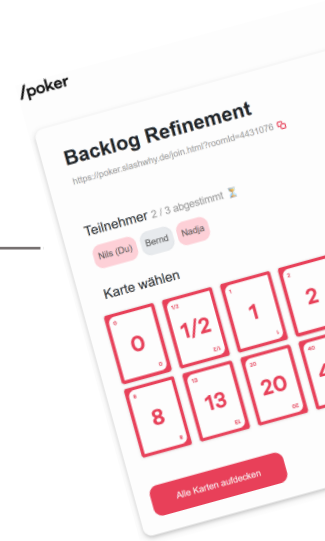
Szenario ID	Quelle	Ereignis	Produkt-Modul	Reaktion	Metrik	Qualitäts-merkmal
QS1	Ein Teammitglied	wählt seine Schätzung	/poker-UI	In den UIs aller Mitschätzenden ist sichtbar dass er geschätzt hat	Innerhalb von 3 Sekunden	Performance
QS2	Ein neues Teammitglied	nimmt erstmalig an einer /poker-Schätzrunde teil	/poker-UI	Es ist ihm problemlos möglich seine erste Schätzung abzugeben	Innerhalb einer Minute	Usability
QS3	Ein Teammitglied außerhalb des /y-Netzes	greift zu	/poker-UI	Die UI funktioniert wie gewohnt	Immer erreichbar übers Internet	Zuverlässigkeit

Umsetzung der Szenarien im Backlog

Fall 1:
Szenario bezieht sich direkt auf ein Feature

→ Qualitatives Akzeptanzkriterium bei User Story des Features ergänzen

QS1: max. 3 Sekunden Verzögerung
→ Zum Feature „Anzeigen wer schon abgestimmt hat“



Fall 2:
Szenario betrifft mehrere Module (Cross-Cutting Concern)

→ Einhaltung der Architektur-Vorgaben werden in der Definition of Done ergänzt und bei allen Code-Reviews überprüft
(z.B. Prinzipien, oder Projekt-interne Best-Practices)

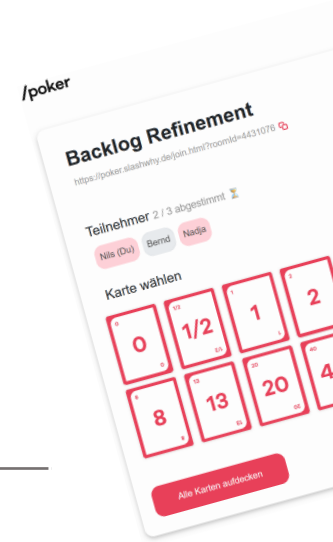
QS2: Die UI ist so intuitiv, dass sich neue Nutzer innerhalb einer Minute zurechtfinden
→ DoD: UX/UI Styleguide wird eingehalten

Umsetzung der Szenarien im Backlog

Fall 3:
Szenario ist als eigenständiges Arbeitspaket bearbeitbar

→ Eigene Quality User Story, die abgeschlossen ist, wenn die Qualität hergestellt und getestet wurde

QS3: Außerhalb des /y-Netzes erreichbar
→ Story: Deployment und ggf. Firewall passend einrichten



Fall 4:
Es ist unklar, wie genau die Qualität erreicht werden kann

- Die bisherige Architektur hat eine „Lücke“

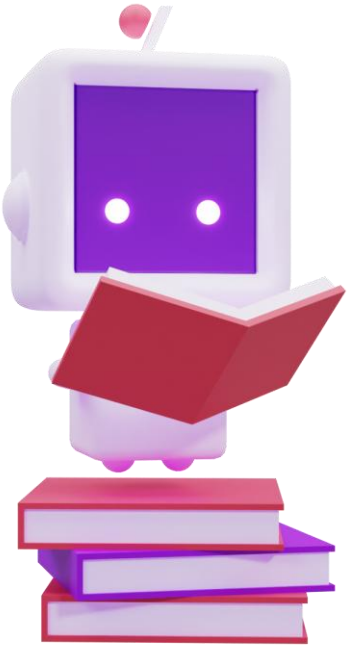
→ Eigene User Story oder eigenes Epic zum erweitern der Architektur:

- Anforderung beleuchten und Lösungsansätze abwägen
- Ergebnis sollte eines der Fälle 1-3 ermöglichen
- Ergebnis wird sichtbar dokumentiert
- Wenn eine Entscheidung getroffen wird, wird diese aussagekräftig festgehalten

Zusammengefasst

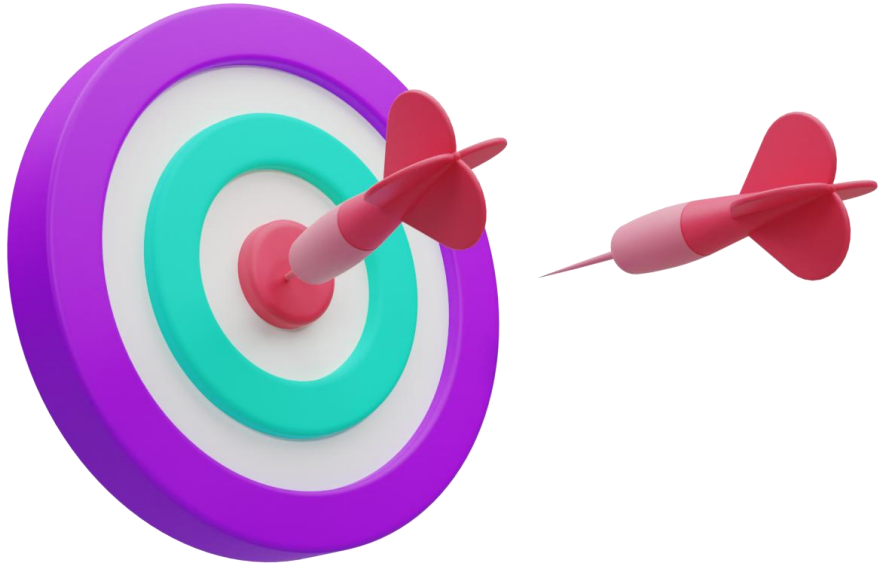
- Die Architektur-Verantwortung muss klar sein
  - einzelne oder mehrere Personen
  - Anforderungen und Entscheidungen dokumentieren und kommunizieren
  - erfahrene Entwickler von slashwhy unterstützen hier gerne

- Architektur braucht Anforderungen
  - vor allem auch nicht-funktionale, nicht nur Features, Features, Features
  - das Ermitteln der priorisierten Qualitätsziele kann slashwhy mit einem LASR-Workshop unterstützen



Zusammengefasst

- Architektur muss im Product Backlog auftauchen
  - durch Akzeptanzkriterien zu Features
  - als eigene User Storys
  - als Definitions of Done



P.S.

Die Backlog Priorisierung sollte auch auf Qualitätszielen basieren!

…sonst bekommt ihr euer Produkt, aber wie „bei Wish bestellt“

Leseempfehlung:
Vorgehensmuster für Software-Architektur
von Stefan Toth

Zeit für den Austausch

- Möchtet ihr etwas ergänzen?
- Gibt es Fragen, die offen geblieben sind?
- Konntet ihr etwas mitnehmen?