Softwarearchitektur – Grundlagen und Praxis

Tragfähige Strukturen für veränderbare Softwaresysteme entwickeln

01 INTRO

Softwarearchitektur bestimmt wesentliche Strukturen und Abhängigkeiten eines Systems und beeinflusst damit Eigenschaften wie Wartbarkeit, Skalierbarkeit, Testbarkeit und Betriebsfähigkeit. Architektur entsteht dabei nicht nur in Diagrammen, sondern manifestiert sich letztlich im Code.

Der Kurs vermittelt grundlegende Architekturprinzipien und deren praktische Anwendung. Die Teilnehmer analysieren Anforderungen und Qualitätsziele, entwickeln Systemstrukturen, bewerten Alternativen und dokumentieren wesentliche Architekturentscheidungen.

DAUER

3 Tage

FORMAT

Präsenz · Live Online · Inhouse

VORKENNTNISSE

Praktische Erfahrung in der Softwareentwicklung

02 TRAININGSPROGRAMM

Inhalte

Was ist Softwarearchitektur?

- Architekturbegriff
- Struktur
- Komponenten
- Beziehungen
- Entscheidungen
- Constraints
- Architektur versus Design

Anforderungen und Qualitätsziele

- funktionale Anforderungen
- Qualitätsanforderungen
- Wartbarkeit
- Performance
- Skalierbarkeit
- Verfügbarkeit
- Security
- Trade-offs

Zerlegung von Systemen

- Verantwortlichkeiten
- Kohäsion
- Kopplung
- Module
- Komponenten
- Schnittstellen
- Information Hiding

Architekturstile

- Layered Architecture
- Modular Monolith
- Hexagonal Architecture
- Ports and Adapters
- Microservices
- Event-driven Architecture
- Auswahlkriterien

Abhängigkeiten gestalten

- Dependency Inversion
- Dependency Direction
- Schnittstellen
- Adapter
- externe Systeme
- technische Infrastruktur
- Testbarkeit

Daten und Persistenz

- Datenhoheit
- Datenbanken
- Repository
- Transaktionen
- Konsistenz
- gemeinsame Daten
- Kopplung durch Daten

Verteilte Systeme einordnen

- Netzwerkgrenzen
- Latenz
- Fehler
- Synchronität
- Asynchronität
- Messaging
- Distributed Monolith

Architektur und Testbarkeit

- Komponenten testen
- Schnittstellen
- Test Doubles
- Integrationstests
- Observability
- Architekturtests
- Testbarkeit als Qualitätsziel

Architekturentscheidungen

- Alternativen
- Trade-offs
- Risiken
- Architecture Decision Records
- Annahmen
- Konsequenzen
- Entscheidungen revidieren

Architektur dokumentieren

- unterschiedliche Sichten
- Kontext
- Container
- Komponenten
- C4-Modell
- Diagramme
- Dokumentation aktuell halten

Architektur bewerten

- Szenarien
- Qualitätsziele
- Risiken
- Reviews
- technische Schulden
- Fitness Functions
- evolutionäre Architektur

Architektur-Workshop

- Anforderungen analysieren
- Qualitätsziele priorisieren
- System zerlegen
- Alternativen entwickeln
- Entscheidungen treffen
- Architektur visualisieren
- Trade-offs diskutieren

03 DURCHFÜHRUNG

Rahmen und Organisation

Dauer	3 Tage
Durchführung	Präsenz oder Live Online, vorzugsweise als Inhouse-Schulung
Schwerpunkt	Inhouse-Schulungen für Unternehmen und Teams
Sprache	Deutsch, englische Fachbegriffe und Dokumentation
Unterlagen	Kursunterlagen, Beispiele und Übungsaufgaben
Technik	Eigener Rechner für die praktischen Übungen

> **Inhouse-Schulungen**
> Ein eigenes System oder Vorhaben kann als Grundlage für den Architektur-Workshop dienen.

04 ZIELGRUPPE

Für wen ist der Kurs gedacht?

Erfahrene Entwickler, angehende Softwarearchitekten, technische Leads und Quality Engineers.

> **Voraussetzungen**
> Praktische Erfahrung in Softwareentwicklung und grundlegendes Verständnis moderner Softwaresysteme.

05 LERNZIELE

Was Sie nach dem Kurs können

Die Teilnehmer können Anforderungen in Architekturentscheidungen übersetzen, Systemgrenzen und Abhängigkeiten gestalten, Alternativen anhand von Qualitätszielen bewerten und Architektur nachvollziehbar dokumentieren.

06 FAQ

Häufige Fragen

Geht es hauptsächlich um Microservices?

Nein. Microservices sind lediglich eine mögliche Architekturform und werden gegen einfachere Alternativen abgewogen.

Muss Architektur vor der Entwicklung vollständig feststehen?

Nein. Architektur kann und sollte sich kontrolliert weiterentwickeln. Wesentliche Entscheidungen und Risiken müssen dennoch bewusst behandelt werden.

Werden Architekturdiagramme erstellt?

Ja. Diagramme dienen dabei als Kommunikationsmittel und nicht als Selbstzweck.

UC IT Service · Ulrich Cuber · kontakt@uc-it.de
https://www.uc-it.de/coaching/softwarearchitektur-grundlagen/