MICRO CONSULT

# Software-Entwicklung neu gedacht

# Teil 3: Software-Architektur

**Welche Ziele in einem Software-Projekt werden maßgeblich davon beeinflusst, ob es sich um ein Multicore-Design mit Safety- und Security-Aspekten handelt?**

Die drei klassischen Ziele in einem Software-Projekt sind die **Qualitätsanforderungen an das Projekt,** die **Kosten für Entwicklung** und **Wartung** sowie die **verfügbare Entwicklungszeit** (Time to Market).

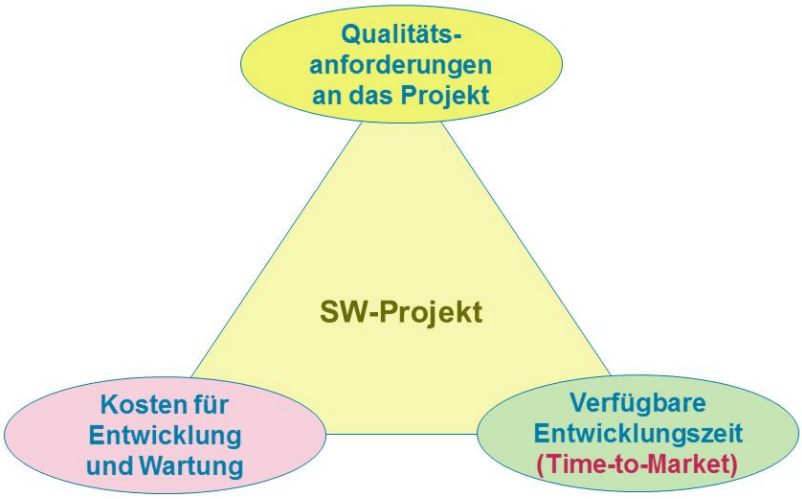


*Bild 1: Die drei klassischen Ziele für das Software-Projekt*

Die Software-Architektur bildet den Bauplan mit dem dazugehörigen Verhalten der Software. Sie beschreibt die grundlegenden Komponenten und deren Zusammenspiel innerhalb eines Softwaresystems. Als Teil eines Software-Entwurfes repräsentiert sie den ersten Entscheidungsschritt für das Software-Design. Das Zusammenstellen von Software-Requirements ist gleichzeitig Input für den Entwurf der Software-Architektur, die als Basis für das Design und die Software-Implementierung dient.

Dabei fällt der Rolle des Software-Architekten die Aufgabe der Entwicklung der Software-Architektur zu. Als wesentliche Grundlagen dafür dienen die Software-Qualitätskriterien. Diese werden auch als nichtfunktionale Eigenschaften bezeichnet und nach ISO/IEC 9126/25010 als Qualitätsanforderungen an das Software-Projekt definiert:

**Änderbarkeit bzw. Wartbarkeit**
- Analysierbarkeit, Konformität, Modifizierbarkeit, Stabilität und Testbarkeit der Software

**Benutzbarkeit**
- Attraktivität, Bedienbarkeit, Erlernbarkeit, Konformität und Verständlichkeit

**Effizienz**
- Konformität: erfüllt Software-Normen und Vereinbarungen zur Effizienz
- Zeitverhalten: angemessene Verarbeitungs- und Antwortzeiten
- Verbrauchsverhalten: Ressourcenverbrauch, CPU-Last

©MicroConsult 11-21

**Funktionalität**

- Angemessenheit: Die Funktionen erfüllen spezifizierte Aufgaben.
- Sicherheit: Das System kann unberechtigte Zugriffe erkennen und verhindern.
- Interoperabilität: Die Software kann mit vorgegebenen Systemen korrekt zusammenarbeiten.
- Konformität: Die Software hält Standards, Vorschriften und gesetzliche Bestimmungen ein.
- Ordnungsmäßigkeit: Die Software erfüllt anwendungsspezifische Normen bzw. Vorschriften und gesetzliche Bestimmungen.
- Richtigkeit: Das System liefert richtige bzw. vereinbarte Ergebnisse.

**Übertragbarkeit**

- Anpassbarkeit, Austauschbarkeit, Installierbarkeit, Koexistenz und Konformität der Software

### Die maßgeblichen Aufgaben der Software-Architektur

Die Software-Architektur stellt die Basis für die Arbeit der Software-Entwickler und Software-Tester dar. Sie schafft ein grundlegendes Verständnis des Software-Systems, der Abhängigkeiten der Software-Komponenten voneinander und der Strukturen und ist ein maßgeblicher Teil der Qualitätssicherung. Sie konserviert das erworbene Wissen in der Architekturdokumentation.

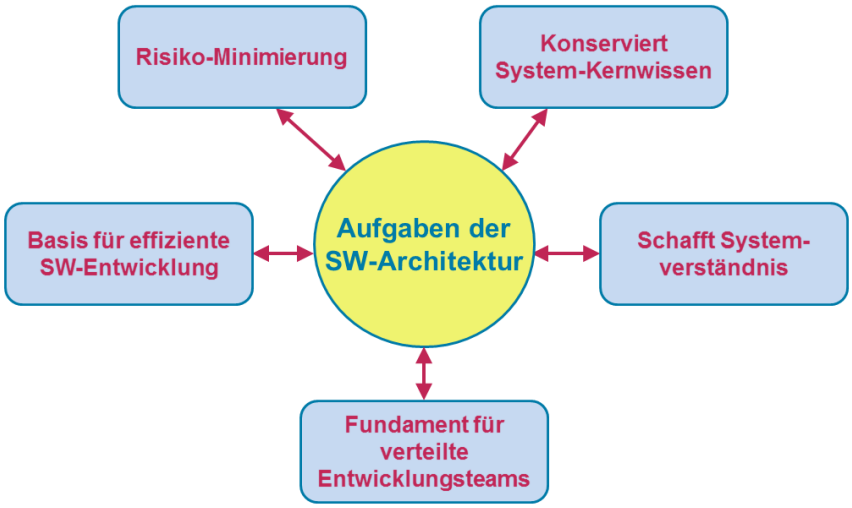


*Bild 2: Die Software-Architektur – fit für das Projekt*

In agilen Softwareprojekten wird die Software-Architektur evolutionär entwickelt, basierend unter anderem auf testgetriebenen Software-Entwicklungsmethoden. Zwei unterschiedliche Entwicklungsansätze herrschen dabei vor:

- **Funktionsarchitektur**: Hier wird das Software-System in Funktionen bzw. Features und deren Abhängigkeiten dargestellt.
- **Komponenten-Architektur**: Entwickelt einen Grobentwurf sowie mehrere Feinentwürfe, die eine feingranulare Struktur der Software enthalten.

Für die Darstellung der Software-Komponenten und deren Abhängigkeiten (Relationen) wird sehr häufig die UML (**U**nified **M**odeling **L**anguage) verwendet.

Mit der grafischen UML-Modellierungssprache lassen sich Konstruktionselemente einer Software-Architektur durch UML-Diagramme gut vereinfacht veranschaulichen. Die UML dient dabei der Übertragbarkeit, der Lesbarkeit, dem Verständnis sowie einer besseren Testbarkeit der Software-Ergebnisse und ist die Basis für eine vereinfachte Software-Wiederverwendung.

Als Basis für die Wiederverwendung können Referenz-Architekturen bzw. Referenz-Modelle und Architekturmuster entworfen werden.

Der Einsatz von Multicore-Architekturen und Anforderungen an unterschiedliche Safety-Ziele der Komponenten (Komponenten mit SIL1 – SIL3 bzw. ASIL-A bis ASIL-D) und Security-Anforderungen haben unmittelbaren Einfluss auf die Architektur eines Software-Projektes.


*Bild 3: Einfluss von Safety & Security auf das Software-Projekt*

Dies manifestiert sich in einer strengeren Unterteilung, der Abgrenzung und der Implementierung von Software-Strukturen. So bestehen z.B. für spezifische Safety-Levels Sicherheitsanforderungen, die Möglichkeit der Zugriffskontrolle von Softwaremodulen auf Daten in Speichern und Peripheriemodulen.

Dies muss auch in der Software-Implementierung (z.B. gesteuert durch **M**emory **M**anagement **U**nits, MMUs**,** bzw. **M**emory **P**rotection **U**nits, MPUs) durch Funktionalitäten der Fehlererkennung und durch die dazu notwendige Fehlerbehandlung garantiert werden.

## Fazit

Setzen wir Multicore-Systeme ein und wollen Systemsicherheit durch Safety- und Security-Anforderungen im Software-Projekt umsetzen, stellt die Software-Architektur einen entscheidenden Baustein des Projekterfolgs dar.

Die weiteren Teile dieser Beitragsreihe zum Thema **Embedded-Software-Entwicklung unter den Aspekten Multicore, Safety und Security** beschäftigen sich näher mit aktuellen Einflüssen und Herausforderungen, Software-Architektur und Software-Tests. Teil 1 beleuchtet, wie Multicore, Safety- und Security-Aspekte die Softwareprojekte von heute verändern. In Teil 2 erfahren Sie, wie sich mithilfe umfassender Software Requirements typische Fehler vermeiden lassen.

**Holen Sie sich das richtige Wissen zu Embedded Software-Entwicklung, Multicore und Safety & Security.** MicroConsult bietet Ihnen professionelle Trainings und Coachings rund um diese Themen an – im Live-Online- und im Präsenz-Format.

## Weiterführende Infos

MicroConsult Fachwissen: Multicore & Mikrocontroller

MicroConsult Training & Coaching: Multicore & Mikrocontroller

MicroConsult Fachwissen: Embedded SW-Entwicklung

MicroConsult Training & Coaching: Embedded SW-Entwicklung

MicroConsult Fachwissen: Safety & Security

MicroConsult Training & Coaching: Safety & Security

Alle MicroConsult Trainings & Coachings

## Autor

Ingo Pohle ist Mitgründer und Geschäftsführer der MicroConsult GmbH und international anerkannter Spezialist für Embedded-Lösungen, mit einem reichen Erfahrungsschatz rund um den Einsatz von Embedded-Mikrocontrollern, Bussystemen und RTOS.

©MicroConsult 11-21