

Inhaltsverzeichnis

1	Einleitung	1
1.1	Softwarearchitektur als Disziplin im Software Engineering	2
1.2	iSAQB – International Software Architecture Qualification Board	4
1.3	Certified Professional for Software Architecture – Foundation Level	5
1.4	Zielsetzung des Buches	7
1.5	Voraussetzungen	7
1.6	Leitfaden für den Leser	8
1.7	Zielpublikum	9
1.8	Danksagungen	9
2	Grundlagen von Softwarearchitekturen	11
2.1	Einbettung in den iSAQB-Lehrplan	12
2.1.1	Was sollen die Teilnehmer können?	12
2.1.2	Was sollen die Teilnehmer verstehen?	13
2.1.3	Was sollen die Teilnehmer kennen?	13
2.2	Softwareintensive Systeme und Softwarearchitekturen	13
2.2.1	Was ist ein softwareintensives System?	14
2.2.2	Ausprägungen von softwareintensiven Systemen	15
2.2.3	Bedeutung der Softwarearchitektur für ein softwareintensives System	18
2.3	Grundlegende Konzepte von Softwarearchitekturen	20
2.3.1	Was ist eine Softwarearchitektur?	20
2.3.2	Bausteine, Schnittstellen und Konfigurationen	21
2.3.3	Konzepte der Beschreibung von Softwarearchitekturen	28
2.3.4	Architekturbeschreibung und Architekturebenen	31

2.3.5	Wechselwirkungen zwischen Softwarearchitektur und Umgebung	33
2.3.6	Qualität und Nutzen der Softwarearchitektur	35
2.4	Der Softwarearchitekturentwurf aus der Vogelperspektive	36
2.4.1	Ziele und Aufgaben des Softwarearchitekturentwurfs	37
2.4.2	Der Softwarearchitekturentwurf im Überblick	38
2.4.3	Wechselspiel der Tätigkeiten und Abstraktionsstufen im Entwurf	40
2.4.4	Aufgaben des Softwarearchitekten und Bezug zu anderen Rollen	43
3	Beschreibung und Kommunikation von Softwarearchitekturen	45
3.1	Einbettung in den iSAQB-Lehrplan	45
3.1.1	Was sollen die Teilnehmer können?	46
3.1.2	Was sollen die Teilnehmer verstehen?	46
3.1.3	Was sollen die Teilnehmer kennen?	46
3.2	Das CoCoME-Beispiel	46
3.2.1	Anwendungsfälle im CoCoME-System	47
3.2.2	Übersicht über den strukturellen Aufbau des CoCoME-Systems	48
3.3	Sichten und Schablonen	49
3.3.1	Bewährte Sichten nach iSAQB	49
3.3.2	UML-Diagramme als Notationsmittel in Sichtenbeschreibungen	51
3.3.3	Sichtenbeschreibung – Grobaufbau und Einführungsbeispiel	54
3.3.3.1	Grobaufbau – Schablonenartige Sichtenbeschreibung	54
3.3.3.2	Beispiel: Auszug aus einer Sichtenbeschreibung für eine Bausteinsicht	56
3.3.4	Kontextsicht oder Kontextabgrenzung	58
3.3.5	Bausteinsicht	62
3.3.6	Laufzeitsicht	65
3.3.7	Verteilungssicht bzw. Infrastruktursicht	69
3.3.8	Wechselwirkungen zwischen Architektursichten	73
3.3.9	Hierarchische Verfeinerung von Architektursichten	73

3.4	Technische oder querschnittliche Konzepte in Softwarearchitekturen	76
3.4.1	Technische bzw. querschnittliche Konzepte: Beispieldimensionen	76
3.4.2	Beispiel: Fehlerbehandlung	77
3.4.3	Beispiel: Sicherheit	78
3.5	Architektur und Implementierung	79
3.5.1	Beispiel Implementierung	80
3.6	Übliche Dokumenttypen für Softwarearchitekturen	81
3.6.1	Zentrale Architekturbeschreibung	81
3.6.2	Architekturüberblick	82
3.6.3	Dokumentübersicht	82
3.6.4	Übersichtspräsentation	82
3.6.5	»Architekturtapete«	82
3.6.6	Handbuch zur Dokumentation	83
3.6.7	Technische Informationen	83
3.6.8	Dokumentation von externen Schnittstellen	83
3.6.9	Template	83
3.7	Praxisregeln zur Dokumentation	84
3.7.1	Regel 1: »Schreiben aus der Sicht des Lesers«	84
3.7.2	Regel 2: »Unnötige Wiederholung vermeiden«	85
3.7.3	Regel 3: »Mehrdeutigkeit vermeiden«	85
3.7.4	Regel 4: »Standardisierte Organisationsstruktur bzw. Schablonen«	85
3.7.5	Regel 5: »Begründen Sie wesentliche Entscheidungen schriftlich«	85
3.7.6	Regel 6: »Überprüfung auf Gebrauchstauglichkeit«	86
3.7.7	Regel 7: »Übersichtliche Diagramme«	86
3.7.8	Regel 8: »Regelmäßige Aktualisierungen«	86
3.8	Beispiele weiterer Architektur-Frameworks	87
3.8.1	4+1-Framework	87
3.8.2	RM-ODP	88
3.8.3	SAGA	89

4	Entwurf von Softwarearchitekturen	91
4.1	Einbettung in den iSAQB-Lehrplan	92
4.1.1	Was sollen die Teilnehmer können?	92
4.1.2	Was sollen die Teilnehmer verstehen?	92
4.1.3	Was sollen die Teilnehmer kennen?	92
4.2	Überblick über das Vorgehen beim Architekturentwurf	92
4.3	Entwurfsprinzipien und Heuristiken	99
4.3.1	Top-down und Bottom-up	99
4.3.2	Hierarchische (De-)Komposition	100
4.3.2.1	Divide et impera	100
4.3.2.2	Prinzipien bei der Zerlegung	101
4.3.2.3	So-einfach-wie-möglich-Prinzip	101
4.3.2.4	Trennung von Verantwortlichkeiten	102
4.3.3	Schmale Schnittstellen und Information Hiding	102
4.3.3.1	Information Hiding	102
4.3.3.2	Verwendung von Schnittstellen	103
4.3.4	Regelmäßiges Refactoring und Redesign	103
4.4	Architekturzentrierte Entwicklungsansätze	104
4.4.1	Domain Driven Design	104
4.4.1.1	Fachmodelle als Basis	104
4.4.1.2	Systematische Verwaltung der Domänenobjekte	105
4.4.1.3	Strukturierung der Fachdomäne	106
4.4.2	MDA	106
4.4.3	Referenzarchitekturen	107
4.4.3.1	Generative Erzeugung von Systembausteinen	107
4.4.3.2	Aspektororientierung	108
4.4.3.3	Objektorientierung	109
4.4.3.4	Prozedurale Ansätze	110
4.5	Techniken für einen guten Entwurf	110
4.5.1	Ausgangssituation und Motivation: Degeneriertes Design	110
4.5.2	Lose Kopplung	111
4.5.3	Hohe Kohäsion	113
4.5.4	Offen-Geschlossen-Prinzip	114
4.5.5	Umkehr der Abhängigkeiten	114
4.5.6	Abtrennung von Schnittstellen	115
4.5.7	Zyklische Abhängigkeiten auflösen	116
4.5.8	Liskov'sches Substitutionsprinzip	116

4.6	Architekturmuster	117
4.6.1	Schichtenarchitektur	118
4.6.2	Modularisierung	119
4.6.3	Pipes and Filters	120
4.6.4	Blackboard	120
4.6.5	Model View Controller	121
4.6.6	Presentation Abstraction Control	122
4.6.7	Serviceorientierung	123
4.6.8	Mediator	124
4.6.9	Broker	125
4.7	Entwurfsmuster	126
4.7.1	Adapter	126
4.7.2	Observer	127
4.7.3	Decorator	128
4.7.4	Proxy	128
4.7.5	Fassade	129
4.7.6	Brücke	130
4.7.7	State	130
4.7.8	Dependency Injection	131
5	Softwarearchitekturen und Qualität	133
5.1	Einbettung in den iSAQB-Lehrplan	134
5.1.1	Was sollen die Teilnehmer können?	134
5.1.2	Was sollen die Teilnehmer verstehen?	135
5.1.3	Was sollen die Teilnehmer kennen?	135
5.2	Bewertung von Softwarearchitekturen	135
5.2.1	Qualitative Bewertung	135
5.2.1.1	DIN ISO/IEC 9126	136
5.2.1.2	Qualitätsmerkmale	136
5.2.1.3	Auswirkungen bestimmter Qualitätsmerkmale	137
5.2.1.4	Taktiken und Praktiken	138
5.2.1.5	Qualitätsmodelle	139
5.2.1.6	ATAM-Methode	140
5.2.2	Quantitative Bewertung	146
5.2.2.1	Überprüfung von Architekturregeln	147
5.2.2.2	Metriken	147
5.2.2.3	Zyklomatische Komplexität	148

5.3	Prototyp und technischer Durchstich	149
5.3.1	Technischer Durchstich	149
5.3.2	Prototyp	149
5.3.2.1	Einsatz von Softwareprototypen	150
5.3.2.2	Arten von Softwareprototypen	150
5.4	Architekturanalyse	151
6	Werkzeuge für Softwarearchitekten	153
6.1	Einbettung in den iSAQB-Lehrplan	153
6.1.1	Was sollen die Teilnehmer können?	153
6.1.2	Was sollen die Teilnehmer verstehen?	153
6.1.3	Was sollen die Teilnehmer kennen?	153
6.2	Allgemeine Hinweise zu Werkzeugen	154
6.2.1	Kosten von Werkzeugen	154
6.2.2	Lizenzen und Lizenzbedingungen	154
6.3	Werkzeuge zum Anforderungsmanagement	155
6.3.1	Anforderungen und Entscheidungskriterien	155
6.3.2	Herausforderungen von Werkzeugen für das Anforderungsmanagement	156
6.3.3	Beispielhafte Vertreter	156
6.4	Werkzeuge zur Modellierung	156
6.4.1	Anforderungen und Entscheidungskriterien	156
6.4.2	Herausforderungen von Werkzeugen für die Modellierung	157
6.4.3	Beispielhafte Vertreter	157
6.5	Werkzeuge zur Generierung	157
6.5.1	Anforderungen und Entscheidungskriterien	158
6.5.2	Herausforderungen von Codegeneratoren	158
6.5.3	Beispielhafte Vertreter	158
6.6	Werkzeuge zur statischen Codeanalyse	158
6.6.1	Anforderungen und Entscheidungskriterien	159
6.6.2	Herausforderungen von Werkzeugen zur statischen Codeanalyse	159
6.6.3	Beispielhafte Vertreter	159

6.7	Werkzeuge zur dynamischen Analyse	160
6.7.1	Anforderungen und Entscheidungskriterien	160
6.7.2	Herausforderungen von Werkzeugen zur dynamischen Analyse	160
6.7.3	Beispielhafte Vertreter	160
6.8	Werkzeuge zum Build-Management	160
6.8.1	Anforderungen und Entscheidungskriterien	161
6.8.2	Herausforderungen von Werkzeugen zum Build-Management	161
6.8.3	Beispielhafte Vertreter	161
6.9	Werkzeuge zum Konfigurations- und Versionsmanagement	162
6.9.1	Anforderungen und Entscheidungskriterien	162
6.9.2	Herausforderungen von Werkzeugen zum Konfigurations- und Versionsmanagement	162
6.9.3	Beispielhafte Vertreter	163
6.10	Werkzeuge zum Codemanagement	163
6.10.1	Herausforderungen von Werkzeugen zum Codemanagement	163
6.10.2	Beispielhafte Vertreter	163
6.11	Werkzeuge zum Test	164
6.11.1	Anforderungen und Entscheidungskriterien	164
6.11.2	Herausforderungen von Testwerkzeugen	164
6.11.3	Beispielhafte Vertreter	164
6.12	Werkzeuge zur Dokumentation	165
6.12.1	Anforderungen und Entscheidungskriterien	165
6.12.2	Herausforderungen von Dokumentationswerkzeugen ...	165
6.12.3	Beispielhafte Vertreter	165

Anhang

A	Beispielfragen	169
A.1	Auszüge aus der Prüfungsordnung	169
A.2	Beispielfragen	170
B	Abkürzungsverzeichnis	171
C	Glossar	173
D	Literaturverzeichnis	185
	Index	191