

Inhaltsverzeichnis

1	Einleitung	1
1.1	Softwarearchitektur	1
1.2	Langlebigkeit	3
1.3	Technische Schulden	4
1.3.1	»Programmieren kann jeder!«	8
1.3.2	Komplexität und Größe	9
1.3.3	Die Architekturerosion steigt unbemerkt	11
1.3.4	Für Qualität bezahlen wir nicht extra!	13
1.3.5	Arten von technischen Schulden	14
1.4	Was ich mir alles anschauen durfte	15
1.5	Wer sollte dieses Buch lesen?	16
1.6	Wegweiser durch das Buch	16
2	Aufspüren von technischen Schulden	19
2.1	Begriffsbildung für Bausteine	19
2.2	Soll- und Ist-Architektur	21
2.3	Verbesserung am lebenden System	25
2.4	False Positives und generierter Code	43
2.5	Spickzettel zum Sotographen	45
3	Architektur in Programmiersprachen	47
3.1	Java-Systeme	47
3.2	C#-Systeme	52
3.3	C++-Systeme	54
3.4	ABAP-Systeme	55
3.5	PHP-Systeme	57

4	Architekturanalyse und -verbesserung	59
4.1	Entwickler und Architektur	59
4.2	Architekturarbeit ist eine Holschuld	60
4.3	Live-Workshop zur Architekturverbesserung	61
4.4	Der Umgang mit den Vätern und Müttern	63
4.5	Modularity Maturity Index (MMI)	64
4.6	Technische Schulden im Lebenszyklus	66
5	Kognitive Psychologie und Architekturprinzipien	69
5.1	Modularität	70
5.1.1	Chunking	70
5.1.2	Übertragung auf Entwurfsprinzipien	72
5.1.2.1	Einheiten	73
5.1.2.2	Schnittstellen	75
5.1.2.3	Kopplung	76
5.2	Musterkonsistenz	77
5.2.1	Aufbau von Schemata	78
5.2.2	Übertragung auf Entwurfsprinzipien	80
5.3	Hierarchisierung	84
5.3.1	Bildung von Hierarchien	84
5.3.2	Übertragung auf Entwurfsprinzipien	86
5.4	Zyklen = misslungene Modularität + Muster	88
5.5	Konsequenzen für die Architekturanalyse	89
6	Architekturstile gegen technische Schulden	91
6.1	Regeln von Architekturstilen	91
6.2	Trennung von fachlichen und technischen Bausteinen	92
6.3	Schichtenarchitekturen	94
6.3.1	Technische Schichtung	95
6.3.2	Fachliche Schichtung	96
6.3.3	Infrastrukturschicht	98
6.3.4	Integration von fachlichen Schichten	100
6.4	Microservices und Domain-Driven Design	101
6.5	Mustersprachen	105
6.5.1	WAM-Mustersprache	107
6.5.2	DDD-Mustersprache	109
6.5.3	Typische Framework-Muster	111
6.6	Langlebigkeit und Architekturstile	112

7	Muster in Softwarearchitekturen	113
7.1	Abbildung der Soll-Architektur auf die Ist-Architektur	113
7.2	Die ideale Struktur: fachlich oder technisch?	116
7.3	Schnittstellen von Bausteinen	121
7.4	Interfaces – das architektonische Allheilmittel?	126
7.4.1	Die Basistherapie	126
7.4.2	Die Nebenwirkungen	128
7.4.3	Feldstudien am lebenden Patienten	131
7.4.4	Der Kampf mit dem Monolithen	134
7.5	Der Wunsch nach Microservices	136
8	Mustersprachen – der architektonische Schatz!	139
8.1	Die Schatzsuche	139
8.2	Die Ausgrabungsarbeiten	141
8.3	Aus der Schatztruhe	143
8.4	Den Goldanteil bestimmen	147
8.5	Jahresringe	148
8.6	Unklare Muster führen zu Zyklen	149
9	Chaos in Schichten – der tägliche Schmerz	153
9.1	Bewertung des Durcheinanders	155
9.1.1	Ausmaß der Unordnung	156
9.1.1.1	Architekturstile und Zyklen	158
9.1.1.2	Programmzeilen in Zyklen	159
9.1.1.3	Dependency Injection und Zyklen	161
9.1.2	Umfang und Verflochtenheit	161
9.1.3	Reichweite in der Architektur	164
9.2	Das große Wirrwarr	168
9.2.1	Der Schwarze-Loch-Effekt	170
9.2.2	Der Befreiungsschlag	172
9.2.3	Technische Schichtung als Waffe	174
9.2.4	Mustersprache als Leuchtturm	176
9.3	Uneven Modules	179
10	Modularität schärfen	183
10.1	Kohäsion von Bausteinen	184
10.2	Größen von Bausteinen	188
10.3	Größen von Klassen	188
10.4	Größe und Komplexität von Methoden	194

10.5 Lose Kopplung 197

10.6 Kopplung und Größe von Klassen 203

10.7 Wie modular sind Sie? 205

11 Geschichten aus der Praxis 207

11.1 Das Java-System Alpha 208

11.2 Das C#-System Gamma 216

11.3 Das C++-System Beta 224

11.4 Das Java-System Delta 233

11.5 Das Java-System Epsilon mit C#-Satelliten 240

 11.5.1 Java-Epsilon 240

 11.5.2 C#-Epsilon 1 248

 11.5.3 C#-Epsilon 2 251

11.6 Das ABAP-System Lambda 256

12 Fazit: der Weg zu langlebigen Architekturen 263

Anhang

A Analysewerkzeuge 269

A.1 Lattix 271

A.2 Sonargraph Architect 272

A.3 Sotograph und SotoArc 274

A.4 Structure101 275

Literatur 279

Index 287