

Langlebige Software-Architekturen

Technische Schulden analysieren, begrenzen und abbauen

Bearbeitet von
Carola Lilienthal

1. Auflage 2015. Taschenbuch. XII, 276 S. Paperback
ISBN 978 3 86490 292 5
Format (B x L): 16,5 x 24 cm

[Weitere Fachgebiete > EDV, Informatik > Software Engineering > Modellierung, UML, SysML](#)

Zu [Leseprobe](#)

schnell und portofrei erhältlich bei


DIE FACHBUCHHANDLUNG

Die Online-Fachbuchhandlung beck-shop.de ist spezialisiert auf Fachbücher, insbesondere Recht, Steuern und Wirtschaft. Im Sortiment finden Sie alle Medien (Bücher, Zeitschriften, CDs, eBooks, etc.) aller Verlage. Ergänzt wird das Programm durch Services wie Neuerscheinungsdienst oder Zusammenstellungen von Büchern zu Sonderpreisen. Der Shop führt mehr als 8 Millionen Produkte.

Inhaltsverzeichnis

1	Einleitung	1
1.1	Softwarearchitektur	1
1.2	Langlebigkeit	3
1.3	Technische Schulden	4
	1.3.1 »Programmieren kann jeder!«	7
	1.3.2 Komplexität und Größe	8
	1.3.3 Die Architekturerosion steigt unbemerkt	10
	1.3.4 Für Qualität bezahlen wir nicht extra!	12
	1.3.5 Arten von technischen Schulden	13
1.4	Was ich mir alles anschauen durfte	14
1.5	Wer sollte dieses Buch lesen?	15
1.6	Wegweiser durch das Buch	15
2	Aufspüren von technischen Schulden	17
2.1	Begriffsbildung für Bausteine	17
2.2	Soll- und Ist-Architektur	19
2.3	Verbesserung am lebenden System	23
2.4	False Positives und generierter Code	41
2.5	Spickzettel zum Sotographen	43
3	Architektur in Programmiersprachen	45
3.1	Java-Systeme	45
3.2	C#-Systeme	49
3.3	C++-Systeme	51
3.4	ABAP-Systeme	52
3.5	PHP-Systeme	54

4	Architekturanalyse und -verbesserung	57
4.1	Entwickler und Architektur	57
4.2	Architekturarbeit ist eine Holschuld	58
4.3	Live-Workshop zur Architekturverbesserung	59
4.4	Der Umgang mit den Vätern und Müttern	61
4.5	Technische Schulden im Lebenszyklus	62
5	Kognitive Psychologie und Architekturprinzipien	65
5.1	Modularität	66
5.1.1	Chunking	66
5.1.2	Übertragung auf Entwurfsprinzipien	68
5.1.2.1	Einheiten	69
5.1.2.2	Schnittstellen	71
5.1.2.3	Kopplung	72
5.2	Musterkonsistenz	73
5.2.1	Aufbau von Schemata	74
5.2.2	Übertragung auf Entwurfsprinzipien	76
5.3	Hierarchisierung	79
5.3.1	Bildung von Hierarchien	79
5.3.2	Übertragung auf Entwurfsprinzipien	81
5.4	Zyklen = misslungene Modularität + Muster	83
5.5	Konsequenzen für die Architekturanalyse	84
6	Architekturstile gegen technische Schulden	87
6.1	Regeln von Architekturstilen	87
6.2	Trennung von fachlichen und technischen Bausteinen	88
6.3	Schichtenarchitekturen	90
6.3.1	Technische Schichtung	91
6.3.2	Fachliche Schichtung	92
6.3.3	Infrastrukturschicht	94
6.3.4	Integration von fachlichen Schichten	96
6.4	Microservices	97
6.5	Mustersprachen	99
6.5.1	WAM-Mustersprache	101
6.5.2	DDD-Mustersprache	103
6.5.3	Typische Framework-Muster	104
6.6	Langlebigkeit und Architekturstile	105

7	Muster in Softwarearchitekturen	107
7.1	Abbildung der Soll-Architektur auf die Ist-Architektur	107
7.2	Die ideale Struktur: fachlich oder technisch?	110
7.3	Schnittstellen von Bausteinen	115
7.4	Interfaces – das architektonische Allheilmittel?	120
	7.4.1 Die Basistherapie	120
	7.4.2 Die Nebenwirkungen	122
	7.4.3 Feldstudien am lebenden Patienten	125
7.5	Die Schatzsuche	128
7.6	Der Kampf mit dem Monolithen	129
8	Mustersprachen – der architektonische Schatz!	133
8.1	Die Ausgrabungsarbeiten	133
8.2	Aus der Schatztruhe	135
8.3	Den Goldanteil bestimmen	139
8.4	Jahresringe	141
9	Chaos in Schichten – der tägliche Schmerz	143
9.1	Bewertung des Durcheinanders	145
	9.1.1 Ausmaß der Unordnung	146
	9.1.1.1 Architekturstile und Zyklen	148
	9.1.1.2 Programmzeilen in Zyklen	149
	9.1.1.3 Dependency Injection und Zyklen	151
	9.1.2 Umfang und Verflochtenheit	151
	9.1.3 Reichweite in der Architektur	154
9.2	Das große Wirrwarr	158
	9.2.1 Der Schwarze-Loch-Effekt	160
	9.2.2 Der Befreiungsschlag	162
	9.2.3 Technische Schichtung als Waffe	164
	9.2.4 Mustersprache als Leuchtturm	166
9.3	Uneven Modules	170
10	Modularität schärfen	173
10.1	Kohäsion von Bausteinen	174
10.2	Größen von Bausteinen	178
10.3	Größen von Klassen	178
10.4	Größe und Komplexität von Methoden	184
10.5	Lose Kopplung	187
10.6	Kopplung und Größe von Klassen	193
10.7	Wie modular sind Sie?	195

11	Geschichten aus der Praxis	197
11.1	Das Java-System Alpha	197
11.2	Das C#-System Gamma	205
11.3	Das C++-System Beta	213
11.4	Das Java-System Delta	222
11.5	Das Java-System Epsilon mit C#-Satelliten	229
	11.5.1 Java-Epsilon	229
	11.5.2 C#-Epsilon 1	237
	11.5.3 C#-Epsilon 2	240
11.6	Das ABAP-System Lambda	245
12	Fazit: der Weg zu langlebigen Architekturen	251

Anhang

A	Analysewerkzeuge	257
A.1	Lattix	259
A.2	Sonargraph Architect	260
A.3	Sotograph und SotoArc	262
A.4	Structure101	263
	Literatur	267
	Index	275