

---

# Inhaltsübersicht

	<b>Einleitung</b>	<b>xiii</b>
<b>Teil I</b>	<b>Architekturgrundlagen</b>	<b>1</b>
1	Microservices	3
2	Mikro- und Makro-Architektur	17
3	Self-contained System (SCS)	37
4	Migration	47
<b>Teil II</b>	<b>Technologie-Stacks</b>	<b>59</b>
5	Docker-Einführung	61
6	Technische Mikro-Architektur	79
7	Konzept: Frontend-Integration	95
8	Rezept: Links und clientseitige Integration	107
9	Rezept: serverseitige Integration mit Edge Side Includes (ESI)	119
10	Konzept: Asynchrone Microservices	131
11	Rezept: Messaging und Kafka	145
12	Rezept: Asynchrone Kommunikation mit Atom und REST	165
13	Konzept: Synchrone Microservices	181
14	Rezept: REST mit dem Netflix-Stack	189
15	Rezept: REST mit Consul und Apache httpd	207

---

16	<b>Konzept: Microservices-Plattformen</b>	221
17	<b>Rezept: Docker-Container mit Kubernetes</b>	227
18	<b>Rezept: PaaS mit Cloud Foundry</b>	241

---

### **Teil III    Betrieb** **253**

---

19	<b>Konzept: Betrieb</b>	255
20	<b>Rezept: Monitoring mit Prometheus</b>	263
21	<b>Rezept: Log-Analyse mit dem Elastic Stack</b>	277
22	<b>Rezept: Tracing mit Zipkin</b>	287
23	<b>Und nun?</b>	293

### **Anhang**

---

<b>A</b>	<b>Installation der Umgebung</b>	297
<b>B</b>	<b>Maven-Kommandos</b>	299
<b>C</b>	<b>Docker- und Docker-Compose-Kommandos</b>	301
	<b>Index</b>	305

# Inhaltsverzeichnis

	<b>Einleitung</b>	<b>xiii</b>
<b>Teil I</b>	<b>Architekturgrundlagen</b>	<b>1</b>
<b>1</b>	<b>Microservices</b>	<b>3</b>
1.1	Microservices: Definition .....	3
1.2	Gründe für Microservices .....	5
1.3	Herausforderungen .....	12
1.4	Independent-Systems-Architecture-Prinzipien (ISA) .....	13
1.5	Bedingungen .....	13
1.6	Prinzipien .....	13
1.7	Bewertung .....	14
1.8	Variationen .....	15
1.9	Fazit .....	16
<b>2</b>	<b>Mikro- und Makro-Architektur</b>	<b>17</b>
2.1	Bounded Context und Strategic Design .....	18
2.2	Technische Mikro- und Makro-Architektur .....	24
2.3	Betrieb: Mikro- oder Makro-Architektur .....	28
2.4	Mikro-Architektur bevorzugen! .....	31
2.5	Organisatorische Aspekte .....	32
2.6	Variationen .....	34
2.7	Fazit .....	35

---

<b>3</b>	<b>Self-contained System (SCS)</b>	<b>37</b>
3.1	Gründe für den Begriff Self-contained Systems	37
3.2	Self-contained Systems: Definition	38
3.3	Ein Beispiel	41
3.4	SCS und Microservices	42
3.5	Herausforderungen	43
3.6	Variationen	45
3.7	Fazit	46
<b>4</b>	<b>Migration</b>	<b>47</b>
4.1	Gründe für eine Migration	47
4.2	Typische Migrationsstrategie	48
4.3	Alternative Strategien	53
4.4	Build, Betrieb und Organisation	54
4.5	Variationen	56
4.6	Fazit	58
<b>Teil II</b>	<b>Technologie-Stacks</b>	<b>59</b>
<b>5</b>	<b>Docker-Einführung</b>	<b>61</b>
5.1	Docker für Microservices: Gründe	62
5.2	Docker-Grundlagen	63
5.3	Docker-Installation und Docker-Kommandos	67
5.4	Docker-Hosts mit Docker Machine installieren	67
5.5	Dockerfiles	69
5.6	Docker Compose	72
5.7	Variationen	75
5.8	Fazit	77
<b>6</b>	<b>Technische Mikro-Architektur</b>	<b>79</b>
6.1	Anforderungen	79
6.2	Reactive	81
6.3	Spring Boot	84
6.4	Go	89
6.5	Variationen	92
6.6	Fazit	93

---

<b>7</b>	<b>Konzept: Frontend-Integration</b>	<b>95</b>
7.1	Frontend: Monolith oder modular? .....	95
7.2	Optionen .....	98
7.3	Resource-oriented Client Architecture (ROCA) .....	99
7.4	Herausforderungen .....	102
7.5	Vorteile .....	103
7.6	Variationen .....	104
7.7	Fazit .....	104
<b>8</b>	<b>Rezept: Links und clientseitige Integration</b>	<b>107</b>
8.1	Überblick .....	107
8.2	Beispiel .....	113
8.3	Rezept-Variationen .....	115
8.4	Experimente .....	116
8.5	Fazit .....	117
<b>9</b>	<b>Rezept: serverseitige Integration mit Edge Side Includes (ESI)</b>	<b>119</b>
9.1	ESI: Konzepte .....	119
9.2	Beispiel .....	120
9.3	Varnish .....	122
9.4	Rezept-Variationen .....	127
9.5	Experimente .....	129
9.6	Fazit .....	130
<b>10</b>	<b>Konzept: Asynchrone Microservices</b>	<b>131</b>
10.1	Definition .....	131
10.2	Events .....	134
10.3	Herausforderungen .....	137
10.4	Vorteile .....	142
10.5	Variationen .....	142
10.6	Fazit .....	143
<b>11</b>	<b>Rezept: Messaging und Kafka</b>	<b>145</b>
11.1	Message-oriented Middleware (MOM) .....	145
11.2	Die Architektur von Kafka .....	146
11.3	Events mit Kafka .....	152
11.4	Beispiel .....	153

---

11.5	Rezept-Variationen .....	162
11.6	Experimente .....	163
11.7	Fazit .....	164
<b>12</b>	<b>Rezept: Asynchrone Kommunikation mit Atom und REST</b>	<b>165</b>
12.1	Das Atom-Format .....	165
12.2	Beispiel .....	171
12.3	Rezept-Variationen .....	175
12.4	Experimente .....	177
12.5	Fazit .....	178
<b>13</b>	<b>Konzept: Synchrone Microservices</b>	<b>181</b>
13.1	Definition .....	181
13.2	Herausforderungen .....	184
13.3	Vorteile .....	187
13.4	Variationen .....	187
13.5	Fazit .....	188
<b>14</b>	<b>Rezept: REST mit dem Netflix-Stack</b>	<b>189</b>
14.1	Beispiel .....	189
14.2	Eureka: Service Discovery .....	191
14.3	Router: Zuul .....	195
14.4	Lastverteilung: Ribbon .....	197
14.5	Resilience: Hystrix .....	199
14.6	Rezept-Variationen .....	203
14.7	Experimente .....	204
14.8	Fazit .....	206
<b>15</b>	<b>Rezept: REST mit Consul und Apache httpd</b>	<b>207</b>
15.1	Beispiel .....	207
15.2	Service Discovery: Consul .....	209
15.3	Routing: Apache httpd .....	212
15.4	Consul Template .....	212
15.5	Consul und Spring Boot .....	214
15.6	DNS und Registrator .....	215
15.7	Rezept-Variationen .....	217

---

15.8	Experimente .....	217
15.9	Fazit .....	219
<b>16</b>	<b>Konzept: Microservices-Plattformen</b>	<b>221</b>
16.1	Definition .....	221
16.2	Variationen .....	224
16.3	Fazit .....	225
<b>17</b>	<b>Rezept: Docker-Container mit Kubernetes</b>	<b>227</b>
17.1	Kubernetes .....	227
17.2	Das Beispiel mit Kubernetes .....	229
17.3	Beispiel im Detail .....	232
17.4	Weitere Kubernetes-Features .....	234
17.5	Rezept-Variationen .....	236
17.6	Experimente .....	237
17.7	Fazit .....	239
<b>18</b>	<b>Rezept: PaaS mit Cloud Foundry</b>	<b>241</b>
18.1	PaaS: Definition .....	241
18.2	Cloud Foundry .....	244
18.3	Das Beispiel mit Cloud Foundry .....	245
18.4	Rezept-Variationen .....	249
18.5	Experimente .....	249
18.6	Serverless .....	250
18.7	Fazit .....	251
<b>Teil III</b>	<b>Betrieb</b>	<b>253</b>
<b>19</b>	<b>Konzept: Betrieb</b>	<b>255</b>
19.1	Warum Betrieb wichtig ist .....	255
19.2	Ansätze für den Betrieb von Microservices .....	258
19.3	Auswirkungen der behandelten Technologien .....	260
19.4	Fazit .....	261
<b>20</b>	<b>Rezept: Monitoring mit Prometheus</b>	<b>263</b>
20.1	Grundlagen .....	263
20.2	Metriken bei Microservices .....	265

---

20.3	Metriken mit Prometheus .....	267
20.4	Beispiel mit Prometheus .....	270
20.5	Rezept-Variationen .....	272
20.6	Experimente .....	273
20.7	Fazit .....	275
<b>21</b>	<b>Rezept: Log-Analyse mit dem Elastic Stack</b>	<b>277</b>
21.1	Grundlagen .....	277
21.2	Logging mit dem Elastic Stack .....	280
21.3	Beispiel .....	282
21.4	Rezept-Variationen .....	284
21.5	Experimente .....	284
21.6	Fazit .....	285
<b>22</b>	<b>Rezept: Tracing mit Zipkin</b>	<b>287</b>
22.1	Grundlagen .....	287
22.2	Tracing mit Zipkin .....	288
22.3	Beispiel .....	291
22.4	Rezept-Variationen .....	292
22.5	Fazit .....	292
<b>23</b>	<b>Und nun?</b>	<b>293</b>

## Anhang

---

<b>A</b>	<b>Installation der Umgebung</b>	<b>297</b>
<b>B</b>	<b>Maven-Kommandos</b>	<b>299</b>
<b>C</b>	<b>Docker- und Docker-Compose-Kommandos</b>	<b>301</b>
	<b>Index</b>	<b>305</b>