Inhalt

Vorwort ..... **XIII**
Danksagung XVII

Der Autor ..... **XIX**

1 Grundlagen **1**
1.1 Definition Architektur 1
1.1.1 Strukturierung in Komponenten – Modularisierung 2
1.1.2 Abläufe 3
1.1.3 Anforderungen 4
1.1.4 Technologien 5
1.1.5 Operative Systeme 5
1.2 Was Architektur definitiv NICHT ist 6
1.3 Organisation 7
1.4 Über Enterprise-Architektur (EA) 11
1.5 Evolution 13
1.5.1 Managed Evolution (Credit Suisse) 14
1.5.2 Aim42 15
1.5.3 Purer Pragmatismus 16
1.6 Dokumentation 16
1.6.1 Mikro-Architektur – Softwaredesign 16
1.6.2 Makro-Architektur 17
1.7 Digitale Transformation – Digitalisierung 17

2 Prinzipien des Software-Entwurfs **19**
2.1 Keep it Simple and Stupid (KISS) 19
2.2 Don't Repeat Yourself (DRY) 21
2.3 Information Hiding Principle 21
2.4 Open Closed Principle 24
2.5 Lose Kopplung 25
2.5.1 Code Reuse 26

2.5.2 Datenbankintegration - gemeinsames Datenmodell 26
2.5.3 Datenbankintegration - selbe Datenbank, unterschiedliche Datenmodelle 27
2.5.4 Synchroner Remote Procedure Call 28
2.5.5 Datenreplikation 28
2.5.6 Messaging 28
2.5.7 Composite-UI 29
2.6 Hohe Kohäsion 30
2.7 Separation Of Concerns 31
2.8 Hierarchischer Aufbau 34
2.9 Zusammenfassung 37

3 Mikro-Architektur - Softwaredesign 39
3.1 SOLID 40
3.1.1 Liskovsches Substitutionsprinzip 40
3.1.2 Interface Segregation Principle 42
3.1.3 Dependency Inversion Principle 43
3.2 Dependency Injection 44
3.3 Law of Demeter 45
3.4 Composition over Inheritance 46
3.5 Selbst-Dokumentation 46
3.6 Design by Contract 48
3.7 Design Pattern 50
3.7.1 Decorator und Delegate (Structural) 51
3.7.2 Adapter (Structural) 52
3.7.3 Facade (Structural) 53
3.7.4 Observer (Behavioral) 54
3.7.5 Simple Factory (Creational) 56
3.7.6 Factory Method (Creational) 57
3.7.7 Abstract Factory (Creational) 58
3.7.8 Builder (Creational) 59

4 Domänengetriebener Entwurf - Domain Driven Design (DDD) ... 63
4.1 Ubiquitous Language 63
4.2 Aufteilung in Subdomänen 64
4.3 Bounded Context 64
4.4 Integration 65
4.4.1 Das Problem mit dem Konformismus 66
4.5 Upstream/Downstream-Beziehungen 67
4.6 Context Map 68
4.7 Beispiel 68
4.8 Fazit 70

5 Enterprise Application Integration Pattern (EAIP) 73
5.1 Orchestrierung vs. Choreografie 74
5.2 Das Prinzip der Dumb Pipes and Smart Endpoints 74
5.3 Tooling 76
5.3.1 Message Bus 76
5.3.2 Message Broker 77
5.3.3 Enterprise Service Bus (ESB) 77
5.3.4 Business Process-Management-Systeme (BPMS) 79
5.3.5 API-Gateways 81
5.3.6 Service Discovery/Service Registration 82

6 Makro-Architektur 83
6.1 Antipattern 84
6.1.1 Maximierung des Reuse 84
6.1.2 Kanonisches Modell 85
6.1.3 Service Versioning 87
6.1.4 Zentraler Mediator - Enterprise Service Bus (ESB) 88
6.2 Empfohlene Pattern 91
6.2.1 Consumer Driven Contract Tests 91
6.2.2 Robustness Principle (Tolerant Reader) 91
6.2.3 Feature Toggles 92
6.2.4 Circuit Breaker 93
6.2.5 Bulkhead 94
6.2.6 Adapter 95
6.2.7 Backend for Frontend (BFF) 96
6.2.8 Saga 96
6.2.9 Pipes and Filters 97
6.2.10 Correlation IDs 98
6.2.11 Event Sourcing 98

7 Verteilte Systeme - Distributed Systems 101
7.1 Monolithen 102
7.1.1 Keine Continuous Delivery möglich!? 103
7.1.2 Automatische Erosion der Struktur!? 104
7.1.3 Monolithische Architekturen skalieren nicht!? 104
7.1.4 Es ist eine Frage von entweder/oder!? 107
7.2 Idempotenz 108
7.2.1 Idempotent Receiver Pattern 108
7.3 Representational State Transfer - REST 109
7.4 Konsistenz 112
7.4.1 Datenbankintegration (Konsistent) 114
7.4.2 Two Phase Commit (Konsistent) 114
7.4.3 Ein großer Datenservice (Konsistent) 114

7.4.4 Send at least once (Eventually Consistent) ... 115
7.4.5 Orchestrierung (Eventually Consistent) ... 117
7.4.6 Choreografie (Nicht automatisch Konsistent) ... 118
7.4.7 Event Sourcing und CQRS (Konsistent) ... 119
7.4.8 Zentraler Mediator Antipattern - Enterprise Service Bus (Eventually Consistent) ... 119

8 Service-orientierte Architektur (SOA) ... 121
8.1 Service-Antipattern ... 122
8.1.1 Service-Kategorien ... 122
8.1.2 Webservice vs. Service ... 123
8.1.3 API im Vordergrund ... 123
8.1.4 SOA 1.0 ... 123
8.2 Microservices ... 128
8.2.1 Monolith First ... 132
8.2.2 Hybride ... 132
8.3 Nanoservices ... 133
8.4 Modulare SOA - Right Sized Services - SOA 2.0 ... 134
8.5 Self Contained Systems (SCS) ... 135
8.6 Integration kommerzieller Systeme (Commercial off the Shelf - COTS) ... 136

9 Metriken ... 139
9.1 Unit-Test-Abdeckung und das Legacy-Code-Dilemma ... 140
9.2 Technische Schuld ... 141
9.3 Komplexität und Modulgröße ... 142
9.3.1 Semantische Komplexität ... 143
9.3.2 Strukturelle Komplexität ... 143
9.3.3 Verschachtelungskomplexität ... 144
9.4 Kohäsion ... 144
9.4.1 Relational Cohesion ... 144
9.4.2 Lack of Cohesion in Methods IV (LCOM4) ... 144
9.5 Component Rank ... 146
9.6 Software-Package-Metriken nach Robert C. Martin ... 146
9.6.1 Afferent Coupling (Ca) ... 147
9.6.2 Efferent Coupling (Ce) ... 147
9.6.3 Instability ... 148
9.7 Metriken nach John Lakos ... 148
9.7.1 Depends Upon und Used From ... 148
9.7.2 Cumulative Component Dependency (CCD) ... 149
9.7.3 Average Component Dependency (ACD) ... 149
9.7.4 Relative Average Component Dependency (RACD) ... 150
9.7.5 Normalized Cumulative Component Dependency (NCCD) ... 150

9.8 Relative Cyclicity 151
9.8.1 Azyklischer Monolith 152
9.9 Strukturkennzahlen und verteilte Systeme 153

10 Zusammenfassung 155
10.1 Die Frage nach dem „richtigen Schnitt" 155
10.1.1 Ausrichtung nach dem Kunden 155
10.1.2 Optimierung der Software 156
10.2 Geteilte Daten 157
10.3 Migration 158
10.3.1 Extraktion 158
10.3.2 APIs First 160
10.3.3 Aushöhlung 162
10.3.4 Ach wenn ich doch nur anfangen könnte! 163
10.3.5 Über die Migration der Mitarbeiter 164
10.4 Pitfalls 164
10.4.1 Es funktioniert 164
10.4.2 Enterprise- vs. Makro-Architektur 165
10.4.3 Scrum 165
10.4.4 Microservices 165
10.4.5 Vereinheitlichung 166
10.4.6 Keine klare Linie 166
10.4.7 Überbewertung des Themas Prozesse 167
10.4.8 Irrationalität 167
10.4.9 Big-Bang-Migration 170
10.4.10 Scope Creep 170
10.4.11 Elfenbeinturm 171
10.4.12 Ignorieren von Feedback zu Machbar- und Sinnhaftigkeit 171
10.4.13 Widersprüchliche Ziele 172
10.4.14 Management durch Kennzahlen 172
10.4.15 Falsche Anreize (Kobra-Effekt) 173
10.4.16 Cargo-Kult 173
10.5 War Story 174
10.6 Fazit 176

11 Umsetzung 179
11.1 Datenreplikation 179
11.1.1 Extract Transform Load (ETL) 179
11.1.2 Features der Datenbanken 179
11.1.3 Polling 179
11.1.4 Push Messaging 180
11.2 Composite UI 180
11.2.1 Partielle Integration im Web 181

11.2.2 Integration über ein Trägerportal 182
11.2.3 Vollständige Pages 183
11.2.4 Komplexe Integration 183
11.3 Design eines Moduls 184
11.4 Consumer Driven Contract Testing mit PACT 186
11.5 Modulares Design 186
11.5.1 Javascript 186
11.5.2 TypeScript 187
11.5.3 Java 188
11.5.4 Open Services Gateway initiative - OSGi 189
11.5.5 ArchUnit 190
11.5.6 Sonargraph 191

12 Glossar 195

13 Quellen 197

Index 201