

Softwarearchitektur für Dummies

DAS INHALTS- VERZEICHNIS

» Hier geht's
direkt
zum Buch

Auf einen Blick

| | |
|--|------------|
| Einleitung | 23 |
| Teil I: Überblick | 29 |
| Kapitel 1: Wie wir Software-Systeme bauen | 31 |
| Kapitel 2: Das Mindset des Architekten..... | 41 |
| Teil II: Elemente von Architekturen | 53 |
| Kapitel 3: Das hab ich extra vergessen – Abstraktion..... | 55 |
| Kapitel 4: Wenn Rechner Gesprächig werden – Netzwerke..... | 65 |
| Kapitel 5: Zu viel zu tun für einen allein – Nebenläufigkeit..... | 89 |
| Kapitel 6: Vom Notizblock bis zum Aktenschrank – Datenhaltung..... | 113 |
| Teil III: Klassische Patterns und Stile | 145 |
| Kapitel 7: Wer macht was – Grundlegende Modularisierungsansätze..... | 147 |
| Kapitel 8: Ich hätt' noch eine kleine Bitte – Erweiterbarkeit..... | 171 |
| Kapitel 9: Rechnen auf dem Schreibtisch – Aufbau lokaler Anwendungen..... | 185 |
| Kapitel 10: Steckdosen und Verbindungen – Netzwerkanwendungen..... | 207 |
| Kapitel 11: Alle Hände voll zu tun – wenn viele Dinge gleichzeitig passieren..... | 225 |
| Kapitel 12: Der neue Ölboom – Analysen auf Daten..... | 241 |
| Teil IV: Architekturen für die Cloud | 259 |
| Kapitel 13: Das erledige ich schnell für Sie – Services..... | 261 |
| Kapitel 14: Hab ich dir doch gesagt – Messages..... | 299 |
| Kapitel 15: Zusammenwachsen – Enterprise-Integration-Patterns..... | 323 |
| Kapitel 16: Auf den Punkt fit – Reactivity..... | 341 |
| Kapitel 17: Das weiß ich schon längst – Verteilte Datenhaltung..... | 373 |
| Teil V: Top-Ten | 405 |
| Kapitel 18: Zehn Meilensteine des Software-Engineerings..... | 407 |
| Kapitel 19: Zehn einflussreiche Ideen..... | 415 |
| Kapitel 20: Zehn Hypes..... | 425 |
| Literaturverzeichnis | 435 |
| Abbildungsverzeichnis | 441 |
| Stichwortverzeichnis | 445 |

Inhaltsverzeichnis

| | |
|--|-----------|
| Einleitung | 23 |
| Über dieses Buch..... | 23 |
| Konventionen in diesem Buch..... | 24 |
| Was Sie nicht lesen müssen..... | 24 |
| Törichte Annahmen über die Leser..... | 25 |
| Wie dieses Buch aufgebaut ist..... | 25 |
| Teil I: Überblick..... | 25 |
| Teil II: Elemente von Architekturen..... | 26 |
| Teil III: Klassische Patterns und Stile..... | 26 |
| Teil IV: Architekturen für die Cloud..... | 26 |
| Teil V: Top-Ten..... | 26 |
| Symbole, die in diesem Buch verwendet werden..... | 26 |
| Wie es weitergeht..... | 27 |
| | |
| TEIL I | |
| ÜBERBLICK | 29 |
| | |
| Kapitel 1 | |
| Wie wir Software-Systeme bauen | 31 |
| Aufgaben und Ziele von Architektur..... | 31 |
| So wollen wir arbeiten – funktionale Anforderungen..... | 32 |
| Richtig rechnen reicht nicht – nicht-funktionale Anforderungen..... | 32 |
| Viele Köche ...– Stakeholder..... | 33 |
| Wer macht was wann wo? – Software-Strukturen..... | 35 |
| Von Silberkugeln und Sonnenseiten – Forces und Trade-offs..... | 36 |
| Luftschlösser sind nutzlos – Architektur, Design und Implementierung.... | 36 |
| Die Rolle des Architekten..... | 37 |
| Ich bin dann mal weg – die traditionelle Rolle..... | 37 |
| Ich begleite euch bis zum Ziel – die moderne Rolle..... | 38 |
| Wo Sie mehr lesen können..... | 39 |
| | |
| Kapitel 2 | |
| Das Mindset des Architekten | 41 |
| Software im Unternehmen..... | 42 |
| IT/Business-Alignment..... | 42 |
| IT als Profit-Center..... | 43 |
| Die Kosten im Blick..... | 43 |
| Hinterfragen für Fortschritt..... | 43 |
| Gesunde Paranoia..... | 44 |
| Gesunde Skepsis..... | 44 |
| Pragmatismus..... | 45 |
| Das Not-Invented-Here-Syndrom..... | 46 |

12 Inhaltsverzeichnis

| | |
|---|----|
| Umgang mit Technologie..... | 47 |
| Die Technik fest im Blick | 47 |
| Neugier auf Technologie..... | 47 |
| Technologie löst keine konzeptuellen Probleme | 48 |
| Es menschelt | 49 |
| Stakeholder..... | 49 |
| Akzeptanz..... | 50 |
| Teambuilding..... | 50 |
| Fehlerkultur | 51 |

TEIL II ELEMENTE VON ARCHITEKTUREN..... 53

Kapitel 3 Das hab ich extra vergessen – Abstraktion..... 55

| | |
|--|----|
| Warum wir abstrahieren | 56 |
| Beherrschung von Komplexität | 56 |
| Decoupling | 57 |
| Wiederverwendung | 57 |
| Erweiterbarkeit..... | 58 |
| Wie wir abstrahieren..... | 59 |
| Abstraktionen als mentale Modelle..... | 59 |
| Abstraktionen als Strukturen..... | 59 |
| Abstraktionen als Sichtweisen..... | 60 |
| Abstraktionen als Entscheidungen..... | 60 |
| Abstraktion durch Commonality and Variability..... | 61 |
| Abstraktionen als Illusionen..... | 61 |
| Abstraktionen als Need-to-know..... | 62 |
| Abstraktion als Storytelling | 62 |

Kapitel 4 Wenn Rechner gesprächig werden – Netzwerke..... 65

| | |
|--|----|
| Anforderungen an Netzwerke..... | 66 |
| Aufbau und Leistung von Netzwerken..... | 67 |
| Pakete und Schichten..... | 67 |
| Die Rolle des Betriebssystems..... | 70 |
| Näher hingeschaut: TCP/IP..... | 71 |
| Schicht 1: Link Layer..... | 72 |
| Schicht 2: Network Layer..... | 74 |
| Schicht 3: Transport Layer..... | 79 |
| Ein eigenes kleines Netz..... | 84 |
| DHCP | 84 |
| NAT | 84 |
| Firewalls | 85 |
| VPN..... | 85 |
| Folgerungen für die Software-Architektur | 86 |
| Wo Sie mehr lesen können..... | 87 |

Kapitel 5**Zu viel zu tun für einen allein – Nebenläufigkeit..... 89**

| | |
|--|-----|
| Technische Grundlagen..... | 90 |
| Die Hardware-Ebene: CPUs, Cores, Caches | 90 |
| Die Happens-Before-Ordnung..... | 93 |
| Prozesse und Threads..... | 93 |
| Virtueller und physischer Speicher | 94 |
| Scheduling..... | 97 |
| Folgerungen für die Architektur..... | 99 |
| Synchronisation zwischen Threads..... | 99 |
| Race-Conditions | 99 |
| Join | 101 |
| Mutual Exclusion Locks | 101 |
| Deadlocks und Locking-Strategien | 103 |
| Condition Variables..... | 104 |
| Blocking-Queues | 107 |
| Asynchrone Operationen: Callbacks, Futures and Promises..... | 108 |
| Thread-Organisation..... | 109 |
| Thread-Pools..... | 109 |
| Das Leader-Followers-Pattern | 110 |
| Das Active-Object-Pattern..... | 111 |
| Wo Sie mehr lesen können..... | 112 |

Kapitel 6**Vom Notizblock bis zum Aktenschrank – Datenhaltung..... 113**

| | |
|--|-----|
| Die Rolle von Daten im Software-Projekt..... | 114 |
| Informationsaustausch zwischen Nutzern..... | 114 |
| Präzise erfasste Geschäftsvorfälle | 116 |
| Persistierter Anwendungszustand | 118 |
| Langfristig wertvolle, missionskritische Informationen..... | 119 |
| Die Grundlage von allem – das Dateisystem..... | 120 |
| Abstraktion Datei..... | 121 |
| Betriebssystem-Puffer als Herausforderung..... | 121 |
| Klassische Lösungen für klassische Probleme – relationale Datenbanken..... | 122 |
| Die großen Strukturen im Blick..... | 123 |
| Gleichförmige Verarbeitung – Tabellen | 124 |
| Konsistenzbedingungen – Constraints | 126 |
| Identifikation – Surrogate Keys..... | 127 |
| Redundanzvermeidung und Verweise – Foreign Keys..... | 128 |
| Verbindungen zwischen Datensätzen – Joins und Indexes..... | 130 |
| Geschachtelte Datenstrukturen – JSON-Support..... | 132 |
| Übergreifende Auswertungen – SQL-Abfragen..... | 133 |
| Daten-Evolution – Änderungen an Tabellen | 135 |
| Konsistente Datenänderungen – Procedures und Triggers | 137 |
| Gleichzeitige komplexe Datenzugriffe – Transaktionen..... | 138 |
| Verteilte Transaktionen mit Two-Phase Commit | 141 |

| | |
|---------------------------------------|-----|
| Folgerungen für die Architektur | 143 |
| Wo Sie mehr lesen können | 144 |

TEIL III
KLASSISCHE PATTERNS UND STILE **145**

Kapitel 7
Wer macht was – Grundlegende
Modularisierungsansätze **147**

| | |
|--|-----|
| Mehr als ein Buzzword: Decoupling | 148 |
| Coupling als Hindernis für Änderungen | 148 |
| Coupling als geteiltes Wissen | 150 |
| Weitere Formen und Bedeutungen von Coupling | 152 |
| Nicht trennen, was zusammengehört – Cohesion | 152 |
| Coupling, Cohesion und Abstraktion | 154 |
| Strategien für Decoupling | 155 |
| Folgerungen für den Software-Architekten | 157 |
| Decoupling mit Design-by-Contract | 157 |
| Zustand und Assertions | 158 |
| Die Methodenschnittstelle | 159 |
| Information Hiding mit Design-by-Contract | 161 |
| Design-Entscheidungen | 162 |
| Die Systemgrenze | 162 |
| Schichten/Layers | 163 |
| Grundstruktur | 163 |
| Decoupling mit Layers | 165 |
| Relaxed Layers | 167 |
| Design-Entscheidungen | 168 |
| Wo Sie mehr lesen können | 169 |

Kapitel 8
Ich hätt' noch eine kleine Bitte – Erweiterbarkeit **171**

| | |
|--|-----|
| Erweiterbarkeit auf der Design-Ebene | 172 |
| Objekte statt Werte | 172 |
| Cohesion als Schlüssel | 173 |
| Das Strategy-Design-Pattern | 174 |
| Frameworks | 175 |
| Grundstruktur | 175 |
| Varianten | 177 |
| Auswahl von Frameworks | 178 |
| Das Interceptor-Pattern | 180 |
| Das Microkernel-Pattern/Plugin-Architekturen | 181 |
| Folgerungen für die Architektur | 183 |
| Wo Sie mehr lesen können | 184 |

Kapitel 9**Rechnen auf dem Schreibtisch – Aufbau lokaler**

| | |
|--|------------|
| Anwendungen | 185 |
| Herausforderungen von Benutzerschnittstellen | 186 |
| Das Model-View-Controller-Pattern | 188 |
| Grundstruktur des MVC | 188 |
| Die Observer-Beziehung zwischen View und Model | 190 |
| Die Document-View-Variante | 191 |
| Das Wichtigste liegt innen – das Model | 193 |
| Aufgaben des Models | 193 |
| Der Schutz des Models | 194 |
| Entwicklung des Models | 196 |
| Ups, das wollte ich nicht – Undo/Redo | 196 |
| Undo/Redo mit dem Command-Pattern | 196 |
| Organisation mit dem Command-Processor-Pattern | 198 |
| Schichten beim MVC | 199 |
| Model-View-Separation | 199 |
| Design- und Architektur-Entscheidungen | 200 |
| Das Model-View-ViewModel-Pattern | 202 |
| Aufgaben des ViewModels | 203 |
| Folgerungen für die Architektur | 204 |
| Wo Sie mehr lesen können | 205 |

Kapitel 10**Steckdosen und Verbindungen –**

| | |
|--|------------|
| Netzwerkanwendungen | 207 |
| Programmieren mit Netzwerken | 207 |
| Anwendungsprotokoll | 208 |
| Sockets | 209 |
| Ein typischer Server-Aufbau | 212 |
| Threading-Strategien | 212 |
| Modularität und Erweiterbarkeit | 214 |
| Die 8 Fallstricke bei verteilten Systemen | 214 |
| Fallstrick 1: Das Netzwerk ist zuverlässig | 215 |
| Fallstrick 2: Die Latenz ist null | 215 |
| Fallstrick 3: Die Bandbreite ist unbeschränkt | 215 |
| Fallstrick 4: Das Netzwerk ist sicher | 216 |
| Fallstrick 5: Die Netzwerktopologie ändert sich nie | 216 |
| Fallstrick 6: Es gibt nur einen einzigen Administrator | 216 |
| Fallstrick 7: Der Datentransport kostet nichts | 217 |
| Fallstrick 8: Das Netzwerk ist homogen | 217 |
| Das Problem der zwei Generäle | 217 |
| Three-Tier-Architekturen | 219 |
| Data-Tier | 220 |
| Business-Tier | 221 |

| | |
|---|-----|
| Presentation-Tier | 221 |
| Gewinn und Herausforderungen bei Three-Tier-Architekturen | 223 |
| Herausforderungen durch die 8 Fallstricke | 224 |
| Wo Sie mehr lesen können..... | 224 |

Kapitel 11

Alle Hände voll zu tun – wenn viele Dinge gleichzeitig

passieren 225

| | |
|---|-----|
| Eventgetriebene I/O | 226 |
| Technische Grundlage: Non-Blocking I/O | 226 |
| Non-Blocking I/O in Java | 227 |
| Das Reactor-Pattern..... | 228 |
| Synchrone Verarbeitung und Nebenläufigkeit | 230 |
| Asynchrone I/O | 230 |
| Betriebssystem-Mechanismen | 231 |
| Java-Mechanismen..... | 232 |
| Das Proactor-Pattern | 233 |
| Komplexität durch Event-driven Architecture..... | 235 |
| Synchrone und blockierende Logik | 235 |
| Übergang zu eventgetriebener Anwendungslogik..... | 236 |
| Verständnis durch Zustandsmaschinen..... | 237 |
| Die Komplexität in der Gesamtschau | 238 |
| Wo Sie mehr lesen können..... | 239 |

Kapitel 12

Der neue Ölboom – Analysen auf Daten..... 241

| | |
|---|-----|
| Pipes and Filters..... | 242 |
| Filter-Netzwerke | 242 |
| Filter für Stream-Processing..... | 243 |
| Pipes..... | 244 |
| Synchronisation zwischen Filtern..... | 245 |
| Folgerungen für die Architektur..... | 245 |
| Business-Intelligence-Lösungen..... | 246 |
| Meier, können Sie mal rauskriegen ...- Analytical Queries | 247 |
| Benutzerführung..... | 248 |
| OLTP vs. OLAP | 249 |
| Archive für die Datenanalyse – Data-Warehouses | 250 |
| Das Wichtigste auf einen Blick – Datendesign für Analytics..... | 250 |
| Ein Blick unter die Haube – spaltenorientierte Speicherung..... | 253 |
| Ich habe da mal was vorbereitet – Data Cubes..... | 255 |
| Das Data-Warehouse füllen – Extract Transform Load..... | 256 |
| Wo Sie mehr lesen können..... | 257 |

| | |
|--|------------|
| TEIL IV | |
| ARCHITEKTUREN FÜR DIE CLOUD | 259 |
| Kapitel 13 | |
| Das erledige ich schnell für Sie – Services | 261 |
| Aus einem Stein gemeißelt – der Monolith | 262 |
| Kommunikation in einem Prozess | 262 |
| Zentrale Datenhaltung | 264 |
| Koordinierte Releases | 265 |
| Technologisch einheitliche Implementierung | 265 |
| Folgerungen für die Architektur | 266 |
| Microservices | 266 |
| Unabhängigkeit beim Deployment | 266 |
| Leichtgewichtige Protokolle und einfache, stabile APIs | 268 |
| Vertikale Funktionsteilung | 269 |
| Eigene Datenhaltung | 270 |
| Das beste Werkzeug für die Aufgabe | 271 |
| Skalierbarkeit | 272 |
| Die Größe eines Microservice | 273 |
| Folgerungen für die Architektur | 273 |
| Hexagonale Architekturen | 274 |
| Ports | 274 |
| Adapters | 275 |
| Primäre und sekundäre Ports | 275 |
| API-Design für Services | 276 |
| Information Hiding | 276 |
| Geschäftslogik und Domain-driven Design im Fokus | 276 |
| Consumer First – Client-orientierte Schnittstellen | 277 |
| Coarse-grained, non-chatty | 277 |
| Systemgrenze | 278 |
| Nachrichtenformate | 278 |
| Stateless | 278 |
| Idempotent | 279 |
| Versionierung | 280 |
| Remote Procedure Calls | 281 |
| REST APIs | 282 |
| GraphQL | 285 |
| Asynchrone Nachrichten und Websockets | 287 |
| Serverless Architectures | 287 |
| Stile der Zusammenarbeit | 288 |
| Orchestration | 288 |
| Process-driven | 288 |
| Choreography | 289 |
| Event-driven | 290 |
| Event-Sourcing | 291 |

| | |
|--|-----|
| Eventual Consistency mit Sagas | 292 |
| Compensating Transactions technisch | 293 |
| Compensating Transactions im Anwendungsbereich | 295 |
| Transaktionsplanung für Sagas..... | 297 |
| Folgerungen für die Architektur..... | 298 |
| Wo Sie mehr lesen können..... | 298 |

Kapitel 14

Hab ich dir doch gesagt – Messages..... 299

| | |
|---|-----|
| Ich vertraue Ihnen diesen Brief an – Message Brokers..... | 300 |
| Anatomie einer Nachricht..... | 300 |
| Channels | 301 |
| Point-to-Point-Nachrichten über Queues..... | 301 |
| Publish/Subscribe mit Topics..... | 302 |
| Zuverlässigkeit | 303 |
| Zustellgarantien..... | 304 |
| Transaktionalität..... | 307 |
| Channel-Design..... | 309 |
| Folgerungen für die Architektur..... | 310 |
| Grundlegende Muster für Nachrichten..... | 311 |
| Command..... | 311 |
| Event..... | 312 |
| Document..... | 313 |
| Request/Reply..... | 314 |
| Message Sequence | 316 |
| Fehlerbehandlung | 317 |
| Asynchrone Service-Kommunikation mit Messages..... | 318 |
| Ausfallsicherheit und Skalierung durch Nachrichten..... | 318 |
| Nachrichten als API..... | 319 |
| Versionierung von Nachrichten | 320 |
| Messages und Eventual Consistency..... | 321 |
| Eventgetriebene Kommunikation | 322 |
| Wo Sie mehr lesen können..... | 322 |

Kapitel 15

Zusammenwachsen – Enterprise-Integration-Patterns..... 323

| | |
|---|-----|
| Anwendungen verbinden..... | 324 |
| Formen der Nachrichtenübermittlung..... | 324 |
| Messaging Gateway und Endpoint..... | 325 |
| Service Activator | 326 |
| Message Translator..... | 327 |
| Canonical Data Model..... | 328 |
| Transactional Client..... | 329 |
| Abstraktion über den Empfänger..... | 330 |
| Publish-Subscribe-Channel..... | 330 |
| Message Router..... | 331 |

| | |
|-------------------------------|-----|
| Message Broker..... | 332 |
| Message Bus..... | 334 |
| Prozesse organisieren..... | 335 |
| Pipes and Filters..... | 335 |
| Routing Slip..... | 337 |
| Process-Manager..... | 338 |
| Wo Sie mehr lesen können..... | 339 |

Kapitel 16

Auf den Punkt fit – Reactivity 341

| | |
|---|-----|
| Reactivity als Konzept..... | 342 |
| Die Sicht des Reactive Manifesto..... | 342 |
| Folgerungen für den Aufbau einzelner Komponenten..... | 344 |
| Reactive Streams..... | 345 |
| Datenströme als Abstraktion..... | 345 |
| Unterschiede zu reinen Datenströmen..... | 347 |
| Non-Blocking Backpressure..... | 348 |
| Zusammenarbeit zwischen Publisher und Subscriber..... | 349 |
| Einfache Publisher als Start der Datenströme..... | 351 |
| Publisher mit Zustand und Ressourcen..... | 352 |
| Publisher aus Event-Quellen..... | 355 |
| Klassische Operatoren für Folgen..... | 356 |
| Ein Gesamtergebnis bereitstellen..... | 358 |
| Teilaufgaben..... | 360 |
| Mehrere Subscriber für einen Strom..... | 362 |
| Fehlerbehandlung..... | 363 |
| Ablauflogik..... | 364 |
| Zeitliche Aspekte..... | 367 |
| Nebenläufigkeit..... | 368 |
| Umgang mit Backpressure..... | 369 |
| Überblick über weitere Mechanismen..... | 370 |
| Folgerungen für die Architektur..... | 371 |
| Wo Sie mehr lesen können..... | 372 |

Kapitel 17

Das weiß ich schon längst – Verteilte Datenhaltung 373

| | |
|--|-----|
| Die Herausforderung..... | 374 |
| Die Gesamtsituation..... | 375 |
| Nebenläufiges Schreiben und Lesen..... | 377 |
| Das CAP-Theorem..... | 378 |
| Schwächere Zusicherungen..... | 379 |
| Eventual Consistency..... | 380 |
| Read-your-writes Consistency..... | 380 |
| Monotonic Read Consistency..... | 381 |
| Monotonic Write Consistency..... | 381 |
| Causal Consistency..... | 381 |
| BASE..... | 382 |

| | |
|---|-----|
| Konflikterkennung und -behandlung..... | 382 |
| Master/Slave-Replication..... | 382 |
| Versionsvektoren zur Konflikterkennung..... | 384 |
| Last Write Wins..... | 385 |
| Siblings..... | 386 |
| Quorum..... | 386 |
| CRDTs..... | 387 |
| Folgerungen für die Architektur..... | 390 |
| Es muss nicht immer SQL sein – NoSQL-Datenbanken..... | 391 |
| Cluster-Betrieb und verteilte Datenhaltung..... | 391 |
| Key-Value-Datenbanken..... | 392 |
| Document-Databases..... | 394 |
| Weitere Arten von nicht-relationalen Datenbanken..... | 396 |
| Eigenschaft schemaless..... | 398 |
| Anwendungsorientierte Schemas..... | 399 |
| Transaktionen..... | 400 |
| Space-based Architecture..... | 401 |
| Die Grundidee..... | 401 |
| Data Pumps..... | 402 |
| Middleware..... | 403 |
| Folgerungen für die Architektur..... | 403 |
| Wo Sie mehr lesen können..... | 404 |

TEIL V**TOP-TEN..... 405****Kapitel 18****Zehn Meilensteine des Software-Engineerings..... 407**

| | |
|---|-----|
| Hochsprachen..... | 407 |
| Funktionale Programmierung..... | 408 |
| Softwaretechnik-Sprachen..... | 409 |
| Objektorientierte Sprachen..... | 409 |
| Garbage Collection..... | 410 |
| Typsysteeme..... | 410 |
| IDEs..... | 412 |
| Agile Software-Entwicklung..... | 412 |
| Free Software und Open-Source-Software..... | 413 |
| Cloud-Computing..... | 414 |

Kapitel 19**Zehn einflussreiche Ideen..... 415**

| | |
|-----------------------------------|-----|
| Software-Engineering..... | 415 |
| No Silver Bullet..... | 417 |
| Coupling und Cohesion..... | 417 |
| Separation of Concerns..... | 418 |
| Information Hiding..... | 419 |
| Responsibility-driven Design..... | 419 |

| | |
|----------------------------|-----|
| Verträge..... | 420 |
| Behavioural Subtyping..... | 420 |
| Inversion of Control..... | 421 |
| Refactoring..... | 422 |

Kapitel 20

Zehn Hypes 425

| | |
|--|-----|
| Objektorientierte Programmierung..... | 425 |
| Komponenten..... | 426 |
| Middleware..... | 427 |
| Model-driven Development und Architecture..... | 427 |
| DevOps..... | 428 |
| Reactivity..... | 429 |
| Serviceorientierte Architekturen..... | 430 |
| Microservices..... | 431 |
| Ökosysteme in der Cloud..... | 431 |
| Low-Code-Plattformen..... | 432 |

Literaturverzeichnis..... 435

Abbildungsverzeichnis 441

Stichwortverzeichnis..... 445