Inhalt

Einleitung . . . . . **XIII**

Vorwort . . . . . **XXI**

Stellen Sie sich eine Welt vor, in der Dev und Ops zu DevOps werden: Eine Einführung in das DevOps-Handbuch . . . . . **XXIII**

Teil I: Die Drei Wege **1**

1 Agile, Continuous Delivery und die Drei Wege **5**
Die Wertkette in der Herstellung 5
Die Technologie-Wertkette 6
Die Drei Wege: Die Prinzipien als Basis von DevOps 9
Zusammenfassung 11

2 Der Erste Weg: Die Prinzipien des Flow **13**
Wir machen unsere Arbeit sichtbar 13
Die Work in Process (WIP) beschränken 15
Die Batchgrößen reduzieren 17
Die Anzahl der Übergaben reduzieren 19
Unsere Flaschenhälse kontinuierlich identifizieren und reduzieren 20
Schwierigkeiten und Verschwendung in der Wertkette eliminieren 22
Zusammenfassung 24

3 Der Zweite Weg: Die Prinzipien des Feedbacks **25**
Mit komplexen Systemen sicher arbeiten 25
Probleme bei ihrem Auftreten erkennen 27

Gene Kim / Jez Humble / Patrick Debois / John Willis, Das DevOps-Handbuch, O'Reilly, ISBN 978-3-96009-047-2

Probleme umschwärmen und lösen, um neues Wissen zu schaffen . . . 28
Die Qualität näher zur Quelle bringen . . . 30
Für folgende Arbeitsstationen optimieren . . . 32
Zusammenfassung . . . 33

4 Der Dritte Weg: Die Prinzipien des kontinuierlichen Lernens und Experimentierens . . . 35
Firmenweites Lernen und eine Sicherheitskultur ermöglichen . . . 36
Institutionalisieren Sie die Verbesserungen bei der täglichen Arbeit . . . 38
Lokale Entdeckungen in globale Verbesserungen umwandeln . . . 40
Resilienzmuster in unsere tägliche Arbeit einbringen . . . 41
Führungskräfte unterstützen eine Kultur des Lernens . . . 42
Zusammenfassung . . . 44
Zusammenfassung Teil I . . . 44

Teil II: Wie man beginnt . . . 45

5 Die Auswahl der ersten Wertkette . . . 47
Greenfield- vs. Brownfield-Services . . . 50
Systems of Record und Systems of Engagement berücksichtigen . . . 53
Beginnen Sie mit den wohlgesonnensten und innovativsten Gruppen . . . 54
DevOps in unserer Firma verbreiten . . . 55
Zusammenfassung . . . 56

6 Die Arbeit in der Wertkette verstehen, sie sichtbar machen und verbreiten . . . 57
Die Teams identifizieren, die unsere Wertkette unterstützen . . . 59
Eine Value Stream Map erstellen, um die Arbeit sichtbar zu machen . . . 59
Ein dediziertes Transformationsteam aufstellen . . . 62
Tools einsetzen, um das gewünschte Verhalten zu unterstützen . . . 70
Zusammenfassung . . . 71

7 Organisation und Architektur anhand von Conways Gesetz entwerfen . . . 73
Organisatorische Archetypen . . . 76
Probleme, die häufig durch eine zu starke Funktionsorientierung verursacht werden (»Auf Kosten optimiert«) . . . 77

Marktorientierte Teams anstreben (»Auf Geschwindigkeit optimieren«) 78
Funktionsorientierung nutzen 79
Tests, Operations und Sicherheit als Aufgaben für jedermann an jedem Tag 80
Jedes Teammitglied zu einem Generalisten machen 82
Nicht Projekte finanzieren, sondern Services und Produkte 84
Teamgrenzen anhand von Conways Gesetz festlegen 85
Lose gekoppelte Architektur nutzen, damit Entwickler produktiv und sicher arbeiten können 85
Zusammenfassung 90

8 Operations in die tägliche Entwicklungsarbeit integrieren 91
Mit gemeinsam genutzten Services die Produktivität der Entwickler erhöhen 93
Ops-Techniker in unsere Serviceteams integrieren 95
Jedem Serviceteam eine Ops-Liaison verschaffen 96
Ops in die Dev-Rituale integrieren 97
Zusammenfassung 100
Zusammenfassung Teil II 101

Teil III: Der Erste Weg: Die technischen Praktiken des Flow 103

9 Die Grundlagen für unsere Deployment-Pipeline legen 105
Dev-, Test- und Produktivumgebungen auf Anforderung erstellen können 107
Ein Single Repository of Truth für das gesamte System schaffen 109
Infrastruktur einfacher neu bauen, statt zu reparieren 111
Die Definition des Entwicklungs-»Done« so anpassen, dass der Code in produktivähnlichen Umgebungen läuft 113
Zusammenfassung 114

10 Schnelles und zuverlässiges automatisiertes Testen ermöglichen 117
Continuous Build, Test und Integration von Code und Umgebungen 120
Eine schnelle und zuverlässige automatisierte Validierungstest-Suite aufbauen 123
Unsere Andon-Cord ziehen, wenn die Deployment-Pipeline unterbrochen wird 133
Zusammenfassung 135

Gene Kim / Jez Humble / Patrick Debois / John Willis, Das DevOps-Handbuch, O'Reilly, ISBN 978-3-96009-047-2

11 Continuous Integration ermöglichen und umsetzen . . . 137
Entwicklung in kleinen Batchgrößen und was passiert, wenn wir zu selten Code in den Trunk committen . . . 140
Trunk-basierte Entwicklungspraktiken übernehmen . . . 142
Zusammenfassung . . . 145

12 Releases automatisieren und ihr Risiko reduzieren . . . 147
Unseren Deployment-Prozess automatisieren . . . 149
Deployments von Releases entkoppeln . . . 158
Übersicht über Continuous Delivery und Continuous Deployment in der Praxis . . . 169
Zusammenfassung . . . 171

13 Eine Architektur für risikoarme Releases . . . 173
Architektonische Prototypen: Monolith vs. Microservices . . . 176
Mit dem Strangler-Application-Muster die Firmenarchitektur sicher weiterentwickeln . . . 179
Zusammenfassung . . . 183
Zusammenfassung Teil III . . . 184

Teil IV: Der Zweite Weg: Die technischen Praktiken des Feedbacks . . . 185

14 Telemetriedaten erzeugen, um Probleme zu erkennen und zu beheben . . . 187
Unsere zentrale Telemetrie-Infrastruktur aufbauen . . . 190
Anwendungs-Logging-Telemetriedaten erzeugen, die in der Produktivumgebung helfen . . . 193
Mit Telemetriedaten beim Lösen von Problemen helfen . . . 195
Produktivmetriken als Teil der täglichen Arbeit ermöglichen . . . 196
Self-Service-Zugriff auf Telemetriedaten und Information Radiators schaffen . . . 198
Telemetrielücken finden und füllen . . . 201
Zusammenfassung . . . 206

15 Telemetriedaten analysieren, um Probleme besser vorherzusehen und Ziele zu erreichen . . . 207
Mit Mittelwert und Standardabweichung potenzielle Probleme erkennen . . . 208
Unerwünschte Ergebnisse instrumentieren und davor warnen . . . 210
Probleme, die entstehen, wenn die Telemetriedaten keine Gauß-Verteilung aufweisen . . . 211
Techniken zur Anomalie-Erkennung einsetzen . . . 215
Zusammenfassung . . . 219

16 Feedback ermöglichen, sodass Entwicklung und Operations Code sicher deployen können 221
Deployments mit Telemetriedaten sicherer machen 223
Dev nimmt am Bereitschaftsdienst von Ops teil 225
Entwickler ermutigen, ihre Arbeit in der Wertkette zu beobachten . . 226
Entwickler ihren Produktivservice initial selbst betreuen lassen 228
Zusammenfassung 233

17 Hypothesengetriebene Entwicklung und A/B-Tests in die tägliche Arbeit integrieren 235
Eine kurze Geschichte des A/B-Testens 237
A/B-Tests in unser Feature-Testing integrieren 238
A/B-Tests in unser Release einbinden 239
A/B-Tests in unsere Feature-Planung integrieren 240
Zusammenfassung 242

18 Review- und Koordinationsprozesse schaffen, um die Qualität der aktuellen Arbeit zu verbessern 243
Die Gefahren von Change-Approval-Prozessen 245
Potenzielle Gefahren von »zu sehr kontrollierten Änderungen« 246
Koordination und Planung von Änderungen ermöglichen 248
Peer Review für Änderungen einführen 249
Potenzielle Gefahren durch mehr manuelle Tests und Änderungssperren 252
Durch Pair Programming all unsere Änderungen verbessern 253
Bürokratische Prozesse unerschrocken stoppen 257
Zusammenfassung 259
Zusammenfassung Teil IV 260

Teil V: Der Dritte Weg: Die technischen Praktiken des fortlaufenden Lernens und Experimentierens 261

19 Lernen ermöglichen und Erfahrungen in die tägliche Arbeit einbringen . . . 263
Eine Just Culture und eine Kultur des Lernens umsetzen 265
Post-Mortem-Meetings ohne Schuldzuweisung nach dem Eintreten von Unfällen 266
Unsere Post-Mortem-Analysen so weit wie möglich verbreiten 269
Die Toleranz gegenüber Vorfällen weiter absenken, um immer schwächere Fehlersignale zu erkennen 271
Fehler neu definieren und das Eingehen kalkulierter Risiken unterstützen 272

Gene Kim / Jez Humble / Patrick Debois / John Willis, Das DevOps-Handbuch, O'Reilly, ISBN 978-3-96009-047-2

Fehler in die Produktivumgebung einbringen, um Resilienz und Lernen zu ermöglichen 273
Game Days einführen, um Zwischenfälle zu üben 275
Zusammenfassung 277

20 Lokale Erkenntnisse in globale Verbesserungen verwandeln 279
Mit Chatrooms und Chat Bots firmenweites Wissen automatisieren und einfangen 279
Standardisierte Prozesse zur Wiederverwendung in Software automatisieren 281
Ein einzelnes, gemeinsam genutztes Quellcode-Repository für die gesamte Firma nutzen 282
Wissen durch automatisierte Tests als Dokumentation und Community of Practice verbreiten 285
Durch kodifizierte nicht funktionale Anforderungen für Operations designen 285
Wiederverwendbare Operations User Stories in die Entwicklung integrieren 286
Sicherstellen, dass technologische Entscheidungen dabei helfen, die Firmenziele zu erreichen 287
Zusammenfassung 290

21 Zeit für das firmenweite Lernen und für Verbesserungen reservieren 291
Rituale institutionalisieren, um technische Schulden zu bezahlen . . . 293
Lehren und Lernen für jeden ermöglichen 295
Teilen Sie Ihre auf DevOps-Konferenzen Ihre Erfahrungen 296
Internes Consulting und Coaches zum Verbreiten von Praktiken aufbauen 299
Zusammenfassung 301
Zusammenfassung Teil V 301

Teil VI: Die technischen Praktiken zum Integrieren von Information Security, Änderungsmanagement und Compliance 303

22 Information Security als Aufgabe für jeden Mitarbeiter an jedem Tag 305
Infosec in die Präsentationen der Entwicklungsiterationen einbinden 306
Infosec in das Fehler-Tracking und in Post-Mortem-Analysen einbinden 307
Präventive Sicherheitsmaßnahmen in die gemeinsam genutzten Quellcode-Repositories und Services integrieren 308

Infosec in unsere Deployment-Pipeline einbinden 309
Für die Sicherheit der Anwendung sorgen 310
Die Sicherheit unserer Software-Lieferkette sicherstellen 315
Die Sicherheit der Umgebung gewährleisten 317
Infosec in die Produktiv-Telemetriedaten integrieren 319
Sicherheits-Telemetriedaten in unseren Anwendungen erstellen 320
Sicherheits-Telemetriedaten in unserer Umgebung erstellen 320
Unsere Deployment-Pipeline schützen 323
Zusammenfassung 324

23 Die Deployment-Pipeline schützen 325
Sicherheit und Compliance in den Prozess zur Änderungsgenehmigung integrieren 325
Den Großteil unserer Änderungen mit geringem Risiko als Standardänderungen neu kategorisieren 327
Was zu tun ist, wenn Änderungen als normale Änderungen eingestuft werden 328
Das Vertrauen in die Segregation of Duty reduzieren 331
Für Dokumentation und Belege für Auditoren und Compliance Officer sorgen 334
Zusammenfassung 337
Zusammenfassung Teil VI 338

24 Ein Aufruf zum Handeln 339

Anhänge: Zusatzmaterial 343

A Die Konvergenz von DevOps 345

B Theory of Constraints und zentrale chronische Konflikte 349

C Tabellarische Form der Abwärtsspirale 351

D Die Gefahren von Übergaben und Queues 353

E Mythen der Arbeitssicherheit 357

F Die Toyota-Andon-Cord 359

G COTS-Software 361

H Post-Mortem-Meetings 363

Gene Kim / Jez Humble / Patrick Debois / John Willis, Das DevOps-Handbuch, O'Reilly, ISBN 978-3-96009-047-2

I Die Simian Army 367
J Transparent Uptime 369
K Weitere Ressourcen 371
L Danksagungen 375
Index 379