

1 Einleitung

- Was ist Continuous Integration? Und was bringt sie Ihnen?
- Wer ist Hudson und was kann er für Sie tun?
- Wer sollte dieses Buch lesen? Und wie?

1.1 Kennen Sie die »Integrationshölle«?

»Oh ja!«, sehe ich den einen oder anderen Leser leidvoll nicken. Falls Ihnen der Begriff nicht geläufig sein sollte – die Geschichte beginnt meistens so:

*Softwareintegration
im Fegefeuer*

Sie arbeiten in einem jener »führenden Softwarehäuser« an der Entwicklung des Flaggschiffproduktes mit. Am kommenden Tag soll Ihre Abteilung die längst überfällige neue Version Ihrer Software ausliefern – vom Vertrieb seit Monaten verlangt und vom Marketing noch länger beworben. Ihr Chef hat nachleuchtende Silikonarmbändchen mit dem Aufdruck »*Failure is not an option*« verteilen lassen.

Die Entwickler werfen also eilig auf dem Abteilungsserver ihre Codeänderungen ab, die sie in den letzten Wochen auf ihren Rechnern erbrütet haben. Doch schon bald wird klar: Die Integration wird auch dieses Mal wieder direkt im Fegefeuer stattfinden!

Erst nach vielen Korrekturen kompiliert das Projekt. Die (spärlich) vorhandenen Tests schlagen fehl, manchmal aber auch nicht. Adrenalin trifft auf Testosteron. Es wird laut in der Abteilung: »Bei mir läuft's aber« – »Ich habe dort nichts geändert!« – »Ohne vernünftiges Refactoring sage ich gar nichts mehr!«

Am späten Abend laufen die Tests endlich durch. Kein Wunder, denn störende Programmteile wurden beherzt auskommentiert. Man muss ja auch etwas für das nächste Service Pack übrig lassen, oder nicht?

Kurz vor Mitternacht übernimmt »der Meister« das Kommando. Als langjähriger Mitarbeiter mit Kopfmonopol kennt er als Einziger die geheimen Schritte, die notwendig sind, um eine Distribution zu

erstellen – und ja, nur er hat die notwendigen Werkzeuge dazu auf seinem Rechner installiert. Abblende nach Schwarz.

Am nächsten Tag wird die Software tatsächlich ausgeliefert. Das Team hat Bauchschmerzen, denn keiner weiß wirklich genau, welche Änderungen in dieses Release eingeflossen sind. Die alten Hasen haben schlauerweise schon mal vierzehn Tage Urlaub eingereicht. Die jüngeren Kollegen stellen sich unter den Türsturz und halten die Luft an. Hoffentlich nicht zu lange, denn das Marketing bewirbt bereits die nächste Version: »*the next big thing*«...

Neue Hoffnung

Während Sie also im Wartezimmer Ihres Betriebspsychiaters über die Ereignisse der letzten Tage nachsinnen, erleben Sie einen wilden Wunschtraum: Wäre es nicht fantastisch, wenn ...

- jemand *rund um die Uhr* alle Codeänderungen an Ihrem Produkt im Auge behielte?
- Jemand, der das komplette Produkt *selbstständig integrieren* könnte, bis hin zur Distribution? Am besten nach jeder Änderung? Ohne gelangweilt, genervt oder nachlässig zu werden?
- Jemand, der Ihr Produkt *gründlich testen* und bei neuen Problemen sofort *Alarm schlagen* würde?
- Jemand, der den gesamten *zeitlichen Verlauf* Ihres Projekts kennen würde, von der ersten Codezeile an? Der Ihnen mit gebrauchsfertigen *Visualisierungen* und *Berichten* helfen würde, Trends und wiederkehrende Fehlermuster aufzuspüren?

Mit anderen Worten: Wäre es nicht schön, einen persönlichen Butler zu haben, der sich kontinuierlich um die Integration Ihres Produktes kümmert?

Die gute Nachricht: Mit dem Continuous-Integration-System Hudson steht Ihnen genau ein solcher »Build-Butler« zur Verfügung. Er kann innerhalb von Minuten seine Dienstwohnung auf Ihrem Server beziehen, kommt mit den allerbesten Referenzen und arbeitet für Sie kostenlos. Interessiert?

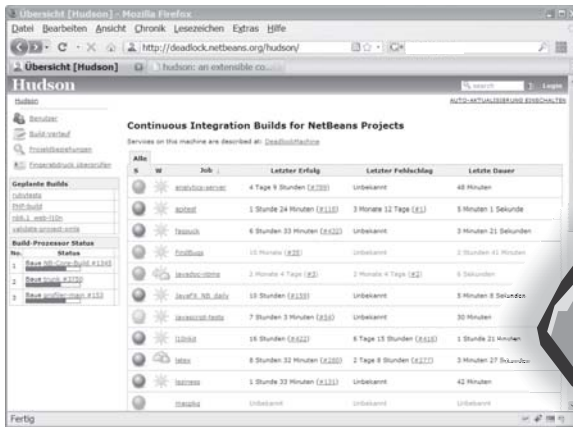


Abb. 1-1

Hudson – Ihr Build-Butler

1.2 Warum Continuous Integration (CI)?

Continuous Integration ist ein hervorragendes Werkzeug, um in Softwareentwicklungsprojekten Risiken zu minimieren und Qualität zu steigern. Wie das?

*Risiken minimieren,
Qualität steigern*

Viele Entwicklungsprojekte geraten in Gefahr, weil die Integration, also die »Endmontage« aller Programmkomponenten, erst im allerletzten Moment angegangen wird. Leider ist dies der denkbar schlechteste Zeitpunkt für unliebsame Überraschungen. Die eingangs geschilderte »Integrationshöhle« ist in vielen Softwareprojekten traurige Realität. CI propagiert häufigere Integrationen als den Ausweg aus dieser Misere. Nur: Wie soll ein Team gleich mehrmals am Tag integrieren, wenn schon die bisherige Frequenz »Einmal im Quartal« kaum zu bewältigen war?

Unverzichtbare Grundlage dafür ist die vollständige Automatisierung des Integrationsprozesses, der auch Tests, Dokumentation, Bereitstellung usw. eines Produktes umfassen kann. Viele Teams setzen dazu bereits Werkzeuge wie Ant, Maven oder Eigenentwicklungen ein. Im zweiten Schritt sorgt dann ein CI-Server für die regelmäßige Ausführung dieses Integrationsprozesses, etwa nach jeder Codeänderung oder einmal pro Nacht (*nightly build*).

Häufige Integration vermeidet nicht nur das Durchleiden der »Integrationshöhle« (oft verschlimmernd gefolgt von ihrem hässlichen Cousin »Big Bang Testing«). Sie bietet sogar noch eine ganze Reihe weiterer, wichtiger Vorteile:

Vorteile von CI

- *Der Integrationsaufwand sinkt*, weil bei frühzeitiger Warnung kleinere Korrekturen ausreichen, um ein auseinanderlaufendes Projekt wieder auf gemeinsamen Kurs zu bringen. Umfangreiche Nacharbeiten unter hohem Zeitdruck vor dem Fertigstellungstermin entfallen.
- *Die Fehlersuche wird vereinfacht*, weil häufigere Integrationen geringere Veränderungen zur vorausgegangenen Integration bedeutet. Das ist wünschenswert, denn je weniger geändert wird, desto weniger Stellen kommen als Fehlerquelle in Frage und müssen untersucht werden.
- *Die Teammoral steigt*, weil häufigere Integrationen auch mehr Rückmeldungen an die Entwickler bedeuten. Erfolgreiche Änderungen und neue Funktionen werden zeitnah vom CI-System positiv bestätigt. Stellen Sie sich einfach einen Kollegen vor, der Ihnen nach jeder Änderung wenige Minuten später anerkennend auf die Schulter klopft. Zum anderen lässt sich – wie durch ein Fangnetz gesichert – mutiger entwickeln, beherzter refaktorisieren und gefahrlos Neues ausprobieren.
- *Manuelle Routineaufgaben* entfallen durch Automatisierung. Die Erzeugung lauffähiger Zwischenstände und interner Demoversionen («Kann man schon was sehen?») ist somit kein lästiger Zeitfresser mehr, sondern läuft unaufgeregt nebenher. Pannen beim Integrieren und Bereitstellen eines Produktes werden vermieden, weil dieser Ablauf nicht mehr alle paar Monate, sondern mehrmals am Tag durchgeführt wird.

Was hält Sie also noch davon ab, diese Vorteile Ihrem Team zugänglich zu machen? Sie benötigen lediglich eine motivierte Person, welche die CI-Einführung in die Hand nimmt (das sind Sie), technisches und organisatorisches Know-how (dieses Buch) und ein gutes CI-System (nächster Absatz).

1.3 Warum Hudson?

Der Markt bietet inzwischen zahlreiche CI-Systeme an. Eine Vergleichsmatrix mit den dreißig bekanntesten Vertretern finden Sie unter [ThoughtWorks10]. Spricht man mit Entwicklern in Firmen und auf Konferenzen, lässt sich jedoch eine Konzentration auf wenige Anbieter beobachten. Vor allem Hudson erfreut sich hier großer Beliebtheit. Warum entscheiden sich Unternehmen wie eBay, Yahoo, Oracle, SAP oder die Allianz für Hudson?

Vorneweg: Es liegt *nicht* an der kostenlosen Verfügbarkeit. Selbstverständlich macht dieser Umstand den Einstieg leichter. Aber als alleinige Erklärung für Hudsons Erfolg greift dies zu kurz, denn:

- Es existieren zahlreiche kostenlose Alternativen. Zudem bieten inzwischen viele kommerzielle Hersteller eingeschränkte »Einstiegversionen« ebenfalls kostenfrei an.
- Selbst kostenpflichtige CI-Systeme dürften sich bereits nach wenigen Monaten amortisiert haben, etwa durch Produktivitätsgewinne.

Viele Anwender stufen stattdessen Hudsons einzigartige Kombination folgender Vorteile als ungleich wichtiger ein:

Vorteile von Hudson

- *Schnelle Resultate:*
Hudson ist zügig installiert und eingerichtet. In Kapitel 5 werden Sie eine produktionsstaugliche Instanz in 60 Sekunden aufsetzen. Sind Ihre Build-Prozesse schon mit Ant oder Maven automatisiert, können Sie in vielen Fällen bereits nach wenigen Minuten von CI mit Hudson profitieren.
- *Unkomplizierter Betrieb:*
Die meiste Zeit arbeitet Hudson, wie man es von einem erstklassigen Butler erwartet: diskret wachsam im Hintergrund, genügsam im Ressourcenverbrauch und geschliffen in den Umgangsformen. Die webbasierte Benutzeroberfläche lässt auch Gelegenheitsadministratoren die wichtigsten Handgriffe ohne Handbuch erledigen.
- *Plugin-Architektur:*
Trotz etablierter »best practices« und aller Standardisierung werden Sie in jedem Team leicht unterschiedliche Werkzeugketten vorfinden. Dank seiner Plugin-Architektur ist Hudson ein wahres »Infrastruktur-Chamäleon«. Es unterstützt unterschiedlichste Versionsverwaltungen, Build-Werkzeuge, Compiler, Testframeworks, Nachrichtenkanäle usw. Über 200 Plugins stehen bereits zur Verfügung, und wöchentlich kommen neue hinzu. Da diese Plugins ebenfalls unter der liberalen MIT-Open-Source-Lizenz vorliegen, sind sie exzellente Ausgangspunkte für Eigenentwicklungen, etwa um haus eigene, proprietäre Systeme anzubinden.
- *Große, hilfsbereite Anwender- und Entwicklergemeinde:*
Mit über 180 Comittern (15 an der Kernanwendung, die weiteren schreiben Plugins), dürfte Hudson eines der CI-Systeme mit der breitesten Entwicklerbasis sein. Bei einer strategischen Entscheidung – vor allem für den Einsatz eines Open-Source-Produkts – ist dies ein nicht unerheblicher Aspekt. Rund 18.000 ausgetauschte

Nachrichten auf Hudsons Mailinglisten im Jahr 2009 unterstreichen zusätzlich die Vitalität des Projekts.

Der Autor dieses Buches ist aktiver Evangelist und Committer im Hudson-Projekt. Sie dürfen also eine engagierte Darstellung der Vorzüge dieses Produktes erwarten. Trotzdem mag Hudson nicht für alle Teams optimale CI-Server sein. Insbesondere die kommerziellen Werkzeuge locken mit einer engeren Verzahnung in Produkte desselben Herstellers (z. B. IntelliJ TeamCity mit IDEA, Atlassian Bamboo mit JIRA, Microsoft Team Foundation Server mit Visual Studio). Am Ende von Kapitel 5 werden wir daher auch auf gängige Alternativen zu Hudson eingehen.

Statt sich nun aber monatelang auf die Suche nach dem idealen CI-System zu begeben, ist viel wichtiger, überhaupt erst einmal häufigere Integrationen einzuführen und eine gelebte CI-Kultur aufzubauen. Ob Hudson dabei nur übergangsweise als Türöffner fungiert oder zum ständigen Begleiter wird, ist nachrangig.

1.4 Warum dieses Buch?

Seit 2008 halte ich regelmäßig Vorträge zu Hudson auf Fachkonferenzen, bei Firmen und in Java User Groups. Ebenso regelmäßig kommt im Anschluss an diese Vorträge die Frage nach einer »richtigen Dokumentation« für Hudson auf.

Das sollte Sie stutzig machen: Seit wann wollen Anwender Handbücher lesen? Ausgerechnet für eine Software, die sich intuitive Bedienung und Benutzerfreundlichkeit auf die Fahnen geschrieben hat? Selbstverständlich werden die wichtigsten Funktionen und Plugins auf der Hudson-Homepage¹ erklärt. Es gibt hilfreiche Blog-Einträge im Internet und zwei vitale Mailinglisten für Anwender und Entwickler. Und wer es ganz genau wissen möchte, kann jederzeit den Quelltext inspizieren². Warum also ein Hudson-Buch?

Viele Anwender hatten das Zusammenpuzzeln von Informationshäppchen aus dem Internet leid und wünschten sich nicht nur Beschreibungen zu einzelnen Hudson-Funktionen, sondern auch konzeptionelles Hintergrundwissen zur CI sowie Erfahrungsberichte aus erfolgreichen CI-Einführungen. Erstaunlicherweise ist das Buchangebot für CI-Titel sehr überschaubar, z. B. [Duval07]. Andere Werke stellen das Thema vor, können aber aus Platzgründen nur sehr bedingt in

1. <http://hudson-ci.org>

2. Sie müssen dazu den Quelltext nicht herunterladen. Unter <http://fisheye.hudson-ci.org> können Sie diesen auch bequem im Webbrowser einsehen.

die Tiefe gehen [Hüttermann10, Larman10, Smart08, Hüttermann07, Clark04]. Titel explizit zu Hudson sucht man komplett vergeblich. Selbst auf dem internationalen Buchmarkt existieren momentan nur Veröffentlichungen in der Entstehungsphase, etwa das lesenswerte, aber bisher nur in Englisch verfügbare Open-Source-Buch von J.F. Smart [Smart10].

Umso mehr freute mich die Anfrage des dpunkt.verlages im Sommer 2009, »ob denn der deutsche Buchmarkt für ein Hudson-Buch reif wäre?«. Aus dieser Anfrage entwickelte sich das Ergebnis, das Sie nun in den Händen halten. Es soll Sie in folgenden Aspekten unterstützen:

- *CI-Grundlagen:*
Eine umfassende Einführung in die Konzepte, Vorteile und Herausforderungen kontinuierlicher Integration. Durch die produktneutrale Darstellung bleibt dieser Teil unabhängig von ständig neu erscheinenden Systemen und Versionen relevant.
- *Hudson für Einsteiger:*
Ein schneller Start mit Hudson, von der Installation über die Konfiguration bis hin zur Anwendung im Alltag
- *Hudson für Fortgeschrittene:*
Anregungen zu Themen wie Build-Zeit-Optimierung, verteiltem Bauen oder der Entwicklung eigener Plugins
- *CI einführen:*
Praxistipps zur erfolgreichen CI-Einführung in Ihrer Organisation, mit Fallstudien aus unterschiedlichen Unternehmensgrößen

Darüber hinaus soll dieses Buch nicht nur Ihren Kopf, sondern auch Ihren Bauch ansprechen und Ihnen Lust auf kontinuierliches Integrieren machen. Wie oft in Softwaretechnik stellt auch bei der CI der Mensch einen maßgeblichen Erfolgsfaktor dar. Wissen und Werkzeuge sind nutzlos, wenn CI nicht wirklich *gelebt* wird. In diesem Sinne wünsche ich Ihnen nicht nur großen Wissenszuwachs bei der Lektüre, sondern im Idealfall auch gute Unterhaltung.

1.5 Wer dieses Buch lesen sollte

Dieses Buch wurde primär für Softwareentwickler und -architekten konzipiert, dürfte aber auch IT-Projektleitern und Informatikstudierenden nützen:

Was bringt Ihnen dieses Buch?

- *Entwickler und Architekten:*
In diesem Buch lernen Sie Werkzeuge und Verfahren kennen, die Sie von lästigen Routineaufgaben entlasten und Ihnen mehr Zeit

für die kreativen und kniffligen Aufgaben der Softwareerstellung lassen. Sie werden sich mit Hudson ein zuverlässiges Frühwarnsystem einrichten, das Sie Probleme beheben lässt – längst bevor sie bei den Kollegen der Qualitätssicherung, Ihren Chefs oder Kunden aufschlagen.

■ *IT-Projektleiter:*

Softwareentwicklung steckt voller Überraschungen. Ihr Releaseprozess sollte keine davon sein. Sie werden Werkzeuge und Verfahren kennenlernen, die Ihnen helfen, Risiken zu minimieren und Qualität zu steigern. Sie werden CI erfolgreich in Ihrer Organisation einführen, weil Sie mit diesem Buch auf bewährten Vorgehensweisen und dem bereits bezahlten Lehrgeld anderer Teams aufbauen können.

■ *Informatikstudierende:*

In diesem Buch lernen Sie Werkzeuge und Verfahren kennen, deren Wert zwar im Prinzip unbestritten, aber längst noch nicht Standard in allen Entwicklungsteams ist. Ob als Praktikant oder baldiger Berufseinsteiger: Mit Ihrem CI-Wissen haben Sie eine Trumpfkarte im Ärmel, die Ihnen auch als »Junior« den Respekt Ihrer Kollegen einbringen kann. Hudsons geringer Ressourcenbedarf und seine hohe Anpassungsfähigkeit erlauben problemlos eine Einführung »nur mal als Experiment«, ohne gleich eine bestehende Infrastruktur komplett auf den Kopf stellen zu müssen.

Benötigtes Vorwissen

Idealerweise haben Sie bereits Erfahrung in Softwareprojekten sammeln können und die Höhen und Tiefen einer Produktentwicklung miterlebt. Die meisten Beispiele in diesem Buch verwenden als Technologiestapel Java, Maven und Subversion. Sie benötigen lediglich Grundkenntnisse in diesen Technologien, um dem vorliegenden Buch folgen zu können. Trotzdem sei Ihnen an dieser Stelle eine hervorragende Einführung in das Konfigurationsmanagement ans Herz gelegt [Popp09]. Der optimale Einsatz eines Versionsmanagementsystems und die Erstellung von Build-Skripten mit Ant oder Maven sind Gebiete, die problemlos den Rahmen dieses vorliegenden Werkes sprengen würden.

*Windows, Linux oder
Mac OS X?*

Hudson ist in Java implementiert und daher auf vielen Plattformen verfügbar, darunter auch Windows, Linux, Mac OS X. Rund zwei Drittel aller Hudson-Server werden unter Linux betrieben. Auf den Entwicklerarbeitsplätzen dominiert hingegen meist Windows. Die Beispiele in diesem Buch sind daher aus der Perspektive eines Entwicklers mit einem Windows-System gewählt. Durch Javas Plattformunabhängigkeit ist dies aber nur an wenigen Stellen von Bedeutung. Sie werden im Buch auf diese wichtigen Unterschiede jeweils hingewiesen werden. Auch softwareentwickelnde Mac-OS-X-Anwender dürften keine Probleme haben, sich an den Windows-Beispielen zu orientieren.

1.6 Wie man dieses Buch lesen sollte

Dieses Buch ist so strukturiert, dass Sie es linear von vorne bis nach hinten lesen können und dabei von den Grundlagen zu immer fortgeschritteneren Themen geführt werden.

Sollten Sie jedoch wenig Zeit haben, möchte ich Ihnen untenstehende Lesepfade mit passenden Abkürzungen empfehlen (Abb. 1–2). Die ausgelassenen Kapitel können Sie dann später, je nach persönlichem Bedarf, in Angriff nehmen.

Wenn Sie wenig Zeit haben...

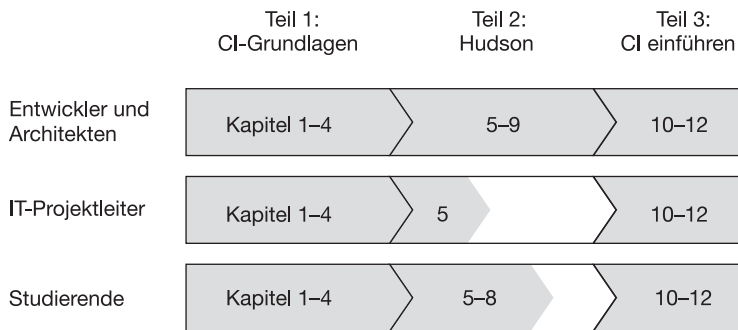


Abb. 1–2
Empfohlene Lesepfade für die Zielgruppen dieses Buches

- *Entwickler und Architekten:*
Beginnen Sie mit den CI-Grundlagen in Teil 1 und lernen Sie Hudson in Teil 2 komplett kennen. Überfliegen Sie abschließend Teil 3, um einen Eindruck von den organisatorischen Aspekten einer CI-Einführung zu bekommen.
- *IT-Projektleiter:*
Beginnen Sie mit den CI-Grundlagen in Teil 1 und lernen Sie Hudson in Teil 2 steckbriefartig kennen. Da Sie Hudson hauptsächlich als Beobachter verwenden werden, überspringen Sie die technischen Details. Die organisatorischen Aspekte in Teil 3 sind für Sie relevanter. Sie sollten diesen Teil daher komplett lesen.
- *Studierende:*
Beginnen Sie mit den CI-Grundlagen in Teil 1. Lernen Sie danach Hudson in Teil 2 detailliert kennen, sparen sich jedoch die ganz fortgeschrittenen Themen wie etwa Plugin-Entwicklung für später auf. Überfliegen Sie abschließend Teil 3, um einen Eindruck von den organisatorischen Aspekten einer CI-Einführung zu bekommen.

1.7 Konventionen

Deutsche und englische
Fachbegriffe

In diesem Buch finden Sie überwiegend deutsche Fachbegriffe. Sollte keine allgemein akzeptierte deutsche Übersetzung existieren oder der Bezug zur Dokumentation eines Werkzeuges durch die Übersetzung erschwert werden, wurden die englischen Begriffe belassen. Deutschen Übersetzungen ist bei der ersten Verwendung der englische Originalbegriff in Klammern (*brackets*) nachgestellt.

Typografie

Der *kursive Schriftschnitt* hebt wichtige Stellen hervor. Texte, die der Benutzeroberfläche Hudsons entnommen wurden (z. B. *Hudson* → *Hudson verwalten*), sind ebenfalls kursiv gestellt.

Ausschnitte aus Quelltexten oder der Kommandozeile sind in der Schriftart LetterGothic dargestellt. Passen Quelltexte oder Befehle nicht in eine Zeile, wurden Sie mithilfe des Zeichens ⇨ umbrochen:

Dieser Text sollte in einer langen, langen, langen ⇨
Zeile eingegeben werden.

1.8 Website zum Buch

Auf der Website zum Buch, www.ci-mit-hudson.de, finden Sie Fehlerkorrekturen, ein Literaturverzeichnis mit »anklickbaren« Links sowie die Quelltexte zur Plugin-Entwicklung aus Kapitel 9.

1.9 Danksagungen

Mein herzlicher Dank gilt zunächst meinem Lektor René Schönfeldt. Er ermunterte mich nicht nur erfolgreich zu diesem Buchprojekt, sondern verlor auch bei grotesken Terminverspätungen meinerseits nie die Fassung.

Großer Respekt gebührt außerdem den aufmerksamen und kritischen Lesern der Betaversionen dieses Buches: Ullrich Hafner, Michael Hüttermann, Jan Matèrne, Hans-Peter Seitz, Martin Stransfeldt, Jürgen Walter, Lorenz Wiest sowie den unbestechlichen anonymen Gutachtern! Mario Ernst danke ich für seine Recherche zum Thema »Blau/Grün« im japanischen Kulturkreis.

Am meisten stehe ich jedoch in der Schuld meiner Familie, allen voran meiner Frau Evelyn und meiner Tochter Floriane, die in den letzten Monaten viel zu oft auf Ehemann und Papa verzichten mussten. Ohne Euer Verständnis gäbe es dieses Buch nicht. Danke.

Simon Wiest

Gomaringen, im Oktober 2010

1.10 Zusammenfassung

Dieses Kapitel eröffnete Ihnen einen ersten Einblick in die Welt der CI: Sie haben erfahren, welche Probleme CI lösen möchte, und Sie haben Hudson als verbreitetes CI-System kennengelernt. Sie wissen, was Sie als Entwickler, IT-Projektleiter oder Studierender in diesem Buch erwartet, und kennen Ihren optimalen Lese pfad.

Im nächsten Kapitel definieren wir genauer, was Continuous Integration ist – und was nicht. Anschließend statten wir »unserem« Entwicklungsteam vom Beginn dieses Kapitels zwei Besuche ab, jeweils vor und nach der CI-Einführung. Ähnlichkeiten mit Ihrem Alltag sind nicht ausgeschlossen ...