Community Experience Distilled

Continuous Delivery and DevOps : A Quickstart Guide

Continuous delivery and DevOps explained

Paul Swartout

[PACKT] PUBLISHING

Table of Contents

Continuous Delivery and DevOps: A Quickstart Guide

Continuous Delivery and DevOps: A Quickstart Guide

First published: October 2012

Production Reference: 1181012

Livery Place

35 Livery Street

Birmingham B3 2PB, UK

ISBN 978-1-84969-368-4

www.packtpub.com

Cover Image by Abhishek Pandey (<abhishek.pandey1210@gmail.com>)

Credits

Author

Paul Swartout

Reviewers

Daniel Albu

Jacob Kottackal Ninan

Leonardo Risuleo

Veturi JV Subramanyeswari

Acquisition Editor

Wilson D'Souza

Commissioning Editor

Meeta Rajani

Technical Editor

Kirti Pujari

Copy Editor

Laxmi Subramanian

Project Coordinator

Michelle Quadros

Proofreader

Aaron Nash

Indexer

Hemangini Bari

Graphics

Aditi Gajjar

Valentina D'Silva

Production Coordinator

Prachali Bhiwandkar

Cover Work

Prachali Bhiwandkar

About the Author

Paul Swartout has spent over 20 years working in IT. Starting out as a Junior Developer within a small software house, Paul has filled a number of roles over the years including Software Engineer, System Administrator, Project Manager, Program Manager, Operations Manager, and Software Development Manager. He has worked across a number of different industries and sectors—from supply chain, through manufacturing, education, and retail to entertainment—and within organizations of various sizes from start-ups to multi-national corporates.

Paul is passionate about software and how it is delivered. Since first encountering "agile" almost a decade ago he has been committed to the adoption and implementation of agile techniques and approaches to improve efficiency and output for software development teams

Until very recently Paul headed up the team responsible for delivering continuous delivery solutions into the Nokia Entertainment business. Paul and his team spent the best part of a year changing the default ways of working and driving the adoption of CD and DevOps as the de facto mode of delivery for Nokia Entertainment products.

Paul lives in a seaside town in the southwest of the UK with his wife, daughters, and two sma yapping things.

Paul is a software development manager at Nokia and is based within the Nokia entertainme team in Bristol in the UK. The entertainment team is responsible for designing, building, and running the entertainment services and solutions for Nokia customers around the globe. These products include Nokia Music, Nokia Reading, and Nokia TV.

Acknowledgement

Firstly I would like to say a big thank you to my darling wife Jane, who has had to put up with a husband who for the past few months has done little more than spend every spare momen staring at a computer screen typing lots of things—which eventually turned into this book.

Thank you to my daughter Harriet for her proof reading skills—I knew that education would come in handy one day.

Next is my good friend and colleague John Clapham whose level headed approach and consistent vision helped to make the implementation of CD and DevOps within Nokia entertainment the success it was. Without that success there would be little to write about.

I would also like to thank Deon Fourie—one of the best bosses out there—and my CD team for their faith and commitment to delivering the CD and DevOps vision. Without you this body of work would be rather dry and many pages lighter.

A big thank you to John Fisher for allowing me to include his transition curve within the book and to Patrick Debois for his support and never ending enthusiasm and drive to bring DevOp to the masses.

About the Reviewers

Daniel Albu is a freelance Interactive Developer who specializes in Flash Platform and HTML5 based solutions, with more than 10 years of experience in production and deploymer of rich media websites, games, and applications.

Daniel has unique expertise in the development and delivery of Flash Platform-based technologies and HTML5 solutions.

Moreover, Daniel provides worldwide remote training and consultancy services around Flash Platform and HTML5 based solutions for companies and individuals alike.

Visit Daniel's website: http://www.danielalbu.com

Read his blog: http://www.flashdeveloper.co

Or follow him on Twitter: `@danielalbu`

Jacob Kottackal Ninan is a Project Manager with a Bachelor of Engineering and MBA degree. He has worked globally for companies like International Air Transport Association (IATA) and eBay.

Jacob is passionate about technology. He also takes up consulting assignments in management and business.

Jacob was a contributor to the 5th edition of the Project Management Book of Knowledge (PMBOK) for the Project Management Institute, USA.

He was also a member and Independent Consultant at Technical Working Group, World Resources Institute / WBCSD Scope 3 Emissions Protocol Development contributing to the GHG emission standards.

Jacob co- founded Kottackal Business Solutions Pvt. Ltd., which is behind the e-commerce hotel booking portal for the Indian market (www.luxotel.in).

He can be contacted at <jacobninan@gmail.com>

I would like to thank my father K. C. Ninan, mother Mariamma Ninan, and my wife Irina for their support.

Leonardo Risuleo is a designer and developer with several years experience in mobile, new media, and user experience (http://www.leonardorisuleo.info/). He's a highly dedicated professional and passionate about what he does. He started back in 2003 and during these years he worked on a variety of different mobile and embedded platforms for a number of

well-known brands. Leo designs, prototypes, and develops mobile applications, games, widgets, and websites.

Apart from being a Flash Platform enthusiast Leo also contributes to the Flash and mobile community as an author and blogger, and he's co-founder of the Italian Mobile & Devices Adobe User Group. From 2008 to 2010 Leo had the honor to be the Forum Nokia Champion a recognition and reward program for top mobile developers worldwide.

In 2010 he formally founded Small Screen Design (http://www.smallscreendesign.com/), a design and development studio focused on mobile design and user experience.

Veturi JV Subramanyeswari is currently working as a Solution Architect at a well-known software consulting MNC in India. Post joining this company she served a few Indian MNCs, many start ups, and R&D sectors in various roles such as Programmer, Tech Lead, Researc Assistant, Architect, and so on. She has more than eight years of experience working with web technologies covering media and entertainment, publishing, healthcare, enterprise architecture, manufacturing, public sector, defense communication, gaming, and so on. She also a well-known speaker who delivers talks on Drupal, Open Source, PHP, Women in Technology, and so on.

She has reviewed other tech books, some of which are as follows:

- *Drupal 7 Multi Sites Configuration*
- *Building Powerful and Robust Websites with Drupal 6*
- *Drupal 6 Module development*
- *PHP Team Development*
- *Drupal-6-site-blueprints*
- *Drupal 6 Attachment Views*
- *Drupal E-Commerce with Ubercart 2.x*
- *Drupal 7: First Look*
- *Twitter bootstrap*
- *Drupal SEO*

I would like to thank my family and friends who supported me in completing my reviews on time with good quality.

Support files, eBooks, discount offers and more

You might want to visit www.PacktPub.com for support files and downloads related to your book.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.PacktPub.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at <service@packtpub.com> for more details.

At www.PacktPub.com, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.

http://PacktLib.PacktPub.com

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can access, read and search across Packt's entire library of books.

Why Subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print and bookmark content
- On demand and accessible via web browser

Free Access for Packt account holders

If you have an account with Packt at www.PacktPub.com, you can use this to access PacktL today and view nine entirely free books. Simply use your login credentials for immediate access.

Preface

For a while now, there has been a buzz around the IT industry about something called continuous delivery and DevOps. Strictly speaking that should be "some things" as continuou delivery and DevOps are actually two separate things.

- Continuous delivery, as the name suggests, is a way of working whereby quality products, normally software assets, can be built, tested, and shipped in quick succession
- Continuous delivery is not a methodology you can use to speed up large releases which happen every few months
- DevOps is another way of working whereby developers and system operators work in harmony with little or no organizational barriers between them towards a common goa
- DevOps is not a way to get developers doing operational tasks so that you can get rid of the operations team and vice versa

Just like any other agile methodology, tool, or way of working you are not forced to use everything you read or hear concerning continuous delivery and DevOps. You simply need to understand what bits they are made up of and which of these bits would bring most value to you and your organization.

This book will provide you with some background information into these two new kids on the block and how they can help you to optimize, streamline, and improve the way you work and ultimately how you ship quality software. Included within are some tricks and tips based upon real-world experiences, which may help you reduce the time and effort needed if you were to do it alone.

What this book covers

Chapter1, Evolution of a Software House introduces you to ACME systems and the evolution of the business, from fledgling start-up through the growing pains following acquisition by a global corporate, to get the best of both worlds.

Chapter 2, No Pain, No Gain introduces some of the introspective techniques that can be used to determine what the current pain points are within the software delivery process and try to understand where they stem from.

Chapter3, Plan of Attack gives you some pointers into how the success of implementing continuous delivery and DevOps can be defined and success measured.

Chapter 4, Tools and Technical Approaches will give you some options around the tooling tha can help with the implementation of continuous delivery and DevOps—right from basic engineering best practices to advanced, fully automated solutions.

Chapter 5, Culture and Behaviors highlights the importance of the *human* factors that must b taken into account.

Chapter 6, Hurdles to Look Out For will give the reader some useful tips and tricks for overcoming or avoiding the bumps in the road as they go through their evolution.

Chapter 7, Measuring Success and Remaining Successful focuses on the various key performance indicators and measures that can be used to monitor and communicate the relative success of continuous delivery and DevOps adoption. We also look at what to do when it's all working.

Appendix, *Some Useful Info* provides you with some more detailed information about the too referenced within the book and some useful contacts within the global continuous delivery an DevOps community.

This appendix is only available as a free download at http://www.packtpub.com/sites/default/files/down0loads/Some_Useful_Info.pdf.

What you need for this book

There are many tools mentioned in this book that will help you no end. These include technic tools, such as Jenkins, GIT, Sonar, and graphite and non-technical tools and techniques, suc as Scrum, Kanban, agile, and TDD.

You may have some of these (or similar) tools in place or are looking at implementing them, which will help. However, the only things you'll really need to enjoy and appreciate this book i the ability to read and an open mind.

Who this book is for

Whether you are a freelance software developer, a system administrator working within a corporate business, an IT project manager, or a CTO in a startup, you will have a common problem: regularly shipping quality software is painful. It needn't be.

This book is not focused on a specific demographic or specific type of person. If you've neve heard of continuous delivery or DevOps, this book should give you an insight into what all the fuss is about. If you have already set out to adopt continuous delivery and/or DevOps, then this book may help provide some useful tips and tricks. If you know everything there is to know about both/either of the subjects, then this book will help reaffirm your choices and may provide some additional things to chew over.

All in all the target audience is quite broad; it includes anyone who wants to understand how ship quality software regularly without the pain.

Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Note

Warnings or important notes appear in a box like this.

Tip

Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book–what you liked or may have disliked. Reader feedback is important for us to develop titles tha you really get the most out of.

To send us general feedback, simply send an e-mail to <feedback@packtpub.com>, and mention the book title through the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on www.packtpub.com/authors.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you t
get the most from your purchase.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do
happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—
we would be grateful if you would report this to us. By doing so, you can save other readers
from frustration and help us improve subsequent versions of this book. If you find any errata
please report them by visiting http://www.packtpub.com/support, selecting your book, clicking
on the **errata submission form** link, and entering the details of your errata. Once your errat
are verified, your submission will be accepted and the errata will be uploaded to our website,
or added to any list of existing errata, under the Errata section of that title.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt
we take the protection of our copyright and licenses very seriously. If you come across any
illegal copies of our works, in any form, on the Internet, please provide us with the location
address or website name immediately so that we can pursue a remedy.

Please contact us at <copyright@packtpub.com> with a link to the suspected pirated material

We appreciate your help in protecting our authors, and our ability to bring you valuable
content.

Questions

You can contact us at <questions@packtpub.com> if you are having a problem with any aspec
of the book, and we will do our best to address it.

Chapter 1. Evolution of a Software House

Before we get into the meat of what continuous delivery (CD) and DevOps are all about, I would like to introduce you to ACME systems. This fictional web-based software business started out—as these things sometimes do—in a back bedroom of one of the founders. The founders were visionaries with big ambitions, good ideas, dreams, and a little bit of cash.

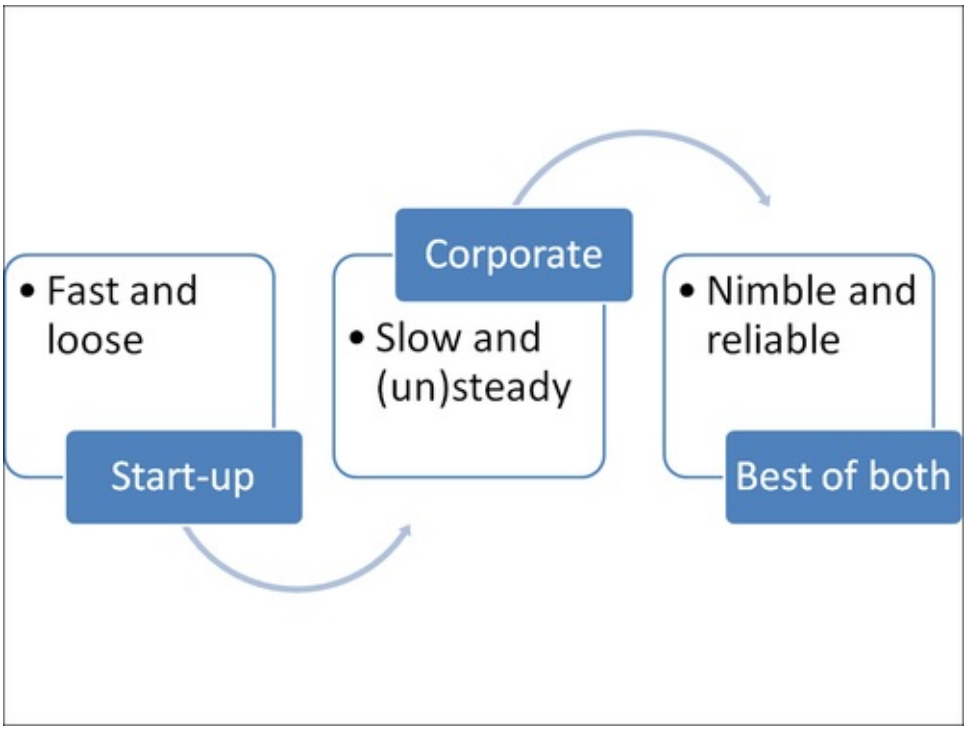
After a few years of hard work, determination, and much blood, sweat and tears, the dream the founders was realized. The business was recognized as a leader in their field and was acquired by a multinational corporate. This acquisition brought with it the funding and resources needed to allow the business to grow and expand to become a global player. That is all well and good, but the story doesn't end there. With corporate owners comes corporate responsibilities, corporate rules, corporate bureaucracy, and corporate processes.

The ACME systems team—now embellished with the corporate owner's name and now operating under corporate governance—started to find it increasingly difficult and painful to deliver quality software. They adopted and adhered to the parent company's processes to improve quality and reduce risk but this made the simple task of delivering software, laboriou and extremely complex.

They came to an evolutionary crossroad and had to make a decision—live with the corporate baggage that they had inherited, try and get back to the good old days and good old ways, which had reaped benefits previously, or try to take the best of both worlds.

It so happens that they made the right choice to implement CD and DevOps and did indeed reap the best of both worlds.

The following diagram gives an overview of the evolution of ACME systems:



Over the next few pages, we'll go through this evolution in more detail. As we do, you may recognize some familiar traits and challenges.

> **Note**
>
> The name ACME is purely fictional and based upon the ACME Corporation first used in *Road Runner Cartoons* in the 1950s—just in case you were wondering.

We'll start with the initial incarnation of the ACME systems business, which for want of a better name, will be called *ACME systems Version 1.0*.

ACME systems Version 1.0

Some of you have most probably worked for (or currently work for) a small software busines There are many such businesses scattered around the globe and they all have one thing in common: they need to move fast to survive and they need to entice and retain customers at all costs. They do this by delivering what the customer wants just before the customer needs it. Deliver too soon and you may have wasted money on building solutions that the customer decides they no longer need, as their priorities or simply their minds have changed. Deliver too late and someone else may well have taken your customer—and more importantly your revenue—away from you. The important keyword here is *deliver*.

As previously mentioned, ACME systems started out in humble beginnings; the founders had the big vision and could see a gap in the market for a web-based solution. They had entrepreneurial spirit and managed to obtain backing from a few parties who injected the life blood of all small businesses—cash.

They then went about sourcing some local, keen, and talented engineers and set about building the web-based solution that bridged the gap, which they had seen before anyone els could.

At first, it was slow going and hard work; lots of pre-sales prototypes needed to be built in a hurry—most of which never saw the light of day—some went straight into production. After many long days, long nights, long weeks, long weekends, and long months, things started to come together. The customer base started to grow and the orders started rolling in; so did th revenue. Soon the number of employees were in double figures and the founders had become directors.

This is all well and good but what has this got to do with CD or DevOps? Well everything really. The culture, default behaviors, and engineering practices of a small software house ar what would be classed as pretty good in terms of CD and DevOps. For example:

- There are next to no barriers between developers and the Operations teams—in fact, they are generally one and the same
- Developers normally have full access to the production environment and can closely monitor their software
- All areas of the business are focused on the same thing, that is, getting software into the production environment as quickly as possible and delighting customers
- Speed of delivery is of the essence
- When things break, everyone swarms around to help fix the problem
- The software evolves quickly and features are added in incremental chunks
- The ways of working are normally very agile

There is a reason for stating that the *culture, default behaviors, and engineering practices of small software house* would be classed as *pretty good* rather than *ideal*. This is because ther are many flaws in the way a small software house has to operate to stay alive:

- Corners will be cut to hit deadlines, which will compromise software design and elegance
- Engineering best practices will be compromised to hit deadlines
- Testing will not be in the forefront of the developer's mind and even if it were, there may not be enough time to work in a test-driven development way
- Source and version control is not used religiously
- With unrestricted access to the production environment, tweaks and changes can be made to the infrastructure
- Software releasing will be mainly manual and most of the time an afterthought of the overall system design
- At times a rough and ready prototype may well become production code without the opportunity for refactoring

To emphasize this, let's have a look at a selection of typical conversations between the Development and Operations teams and the management team of ACME systems.

Developer/Ops	**Management**
We would like to invest some time in implementing a source control system.	Is it free? We don't have time
We would like to invest time in developing a fully automated test suite.	Is it free? We don't have time
We would like to invest time in implementing automated server provisioning.	Is it free? We don't have time

The production environment has crashed.	You're not going home until it' fixed. I'll get the pizza!
This prototype is rough and ready and needs to be rewritten before we hand it over to our customers.	Don't worry, you'll get time to rewrite it.
The prototype we are now using in production keeps crashing.	You're not going home until it' fixed. I'll get the pizza!
We want to work in a test-driven development mode.	That will only slow things dow and we don't have the time.
I can manually hack the production server to improve performance and stability to overcome the issues we're having.	I fully trust your judgment on this, just get it done quickly.
The manual hack I did last week has caused the disks to fill up and the production server has crashed.	You're not going home until it' fixed. I'll get the pizza!

We'll now have a look at the software delivery process for ACME systems Version 1.0, which to be honest, shouldn't take too long.

Software delivery process flow Version 1.0

The following diagram gives an overview of the simple process used by ACME systems to deliver software. It's simple, elegant (in a rough and ready kind of way), and easy to communicate and understand.