New Relic®

eBook

Kickstarting DevOps

How to get everyone on board and successfully make the transition

 | US +888-643-8776 | www.newrelic.com | www.twitter.com/newrelic | blog.newrelic.com

Table of Contents

New Relic.

Introduction

So you're ready to plunge headlong into DevOps. Right on! You've done your homework, and you realize that in order to keep up with the rapid pace of business today, you need to implement DevOps practices at your organization. Yet despite the obvious benefits, such as faster deployments and fewer failures, the rest of your organization may not be so enthusiastic. This eBook will help you familiarize yourself with the common barriers to DevOps adoption so that you can come up with ways to win over the skeptics—and get your organization's DevOps movement rolling.

CHAPTER 1

Challenges to Implementing DevOps

Challenges to Implementing DevOps

Many people fear change and, without a doubt, DevOps represents change. However, much of the uncertainty and hesitation that are attached to the notion of DevOps can be lessened simply by educating individual stakeholders on how DevOps will affect them, its pluses as well as minuses, and most importantly, their role in shaping your organization's approach to DevOps.

You Say 'to-may-to', I Say 'to-mah-to'

Bringing developers and operations together often highlights the differences in the way each group views the world—and each other. Red Hat found out the hard way:

> *We discovered early (but not right away) that even between our few team members, the same terms often had different meaning. In one grooming session, two team members—one sys admin, one infosec wonk—spent 10 minutes talking past each other because to one of them, the term "full stack" included the load balancers; to the other, it didn't.*[1]

Language differences are just one way the two groups tend to have different reactions to DevOps.

Convincing Developers? Easy. Operations? Not So Much.

Developers are often the easiest converts to DevOps for several reasons. First, they're more accustomed to change than other disciplines. Unless they graduated from engineering school yesterday, most software engineers are coding today in a language that they learned in the workplace, not a university classroom. In addition, the buzz about DevOps often emphasizes the quality of life and job satisfaction benefits to developers, including faster provisioning, and thus, greater room for creativity and innovation. And let's face it: Developers just like new toys, and DevOps comes with lots of them.

Operations professionals react differently. Many fear that with DevOps, developers will dump untested, failure-prone software into production, destabilizing the system. While there's ample evidence that the reverse is usually true, keeping the system running is the number one imperative for operations, so DevOps can be a harder sell for them. The trick here is to help Operations personnel understand that faster deployment cycles, with significantly fewer changes per cycle, means that problems are much easier to identify and rollbacks should be painless.

[1] Bill Montgomery, "Red Hat IT's DevOps Journey: harder than we thought," Red Hat developer blog, http://developerblog.redhat.com/2014/03/05/rh-devops-journey-harder/#more-411284, March 5, 2014.

Another concern for operations is that their function will go away, a fear that is exacerbated by the loose talk about NoOps, which is "the concept that an IT environment can become so automated and abstracted from the underlying infrastructure that there is no need for a dedicated team to manage software in-house."[2]

> *The last thing many application developers want to do is have a sit-down with the ops guys…NoOps means that application developers will never have to speak with an operations professional again. NoOps will achieve this nirvana, by using cloud Infrastructure-as-a-Service and Platform-as-a-Service to get the resources they need when they need them.*[3]

To be fair, the NoOps talk has mostly died out in the last two years, but even so, many operations professionals are apprehensive about being assimilated into development, which they suspect is one step on the path to being eliminated altogether.

Everybody Gets into the Act

The impact of DevOps on people in the organization comes in a number of areas. It requires new skills; for example, system administrators may have to develop new tools, a skill they may not have learned or used often. The pace of DevOps can be challenging, especially for developers who work in the more linear world of waterfall development. Line managers may resist the formation of cross-functional teams based on concerns about the difficulty of managing in such an environment—in fact, this objection is a major inhibitor to DevOps adoption in larger enterprises with hierarchical management systems. Executives fret that they will not be able to attract and retain talent with the new skills and abilities necessary for DevOps and worry about undermining salary guidelines to compete for scarce software professionals.

Have You Thought About…?

DevOps changes software workflows and that has implications as well. For one thing, legacy linear management tools are often not suited for DevOps, requiring substantial capital investment in new tools (although many organizations are turning to open source and home-grown tools as a way to mitigate the expense). The intertwined workflows are harder to visualize and document, making regulatory compliance more difficult and time-consuming. Some security professionals worry that security is being given short shrift:

> *When you ask DevOps people if they factor in security from the get-go, the response is pretty much: "Um, nope." That's a problem.*[4]

Security and other teams do not need to be left out in the cold. A good DevOps implementation will ensure that all teams' needs are met. But to get there, you need to make sure that everyone adjusts their traditional workflows and views to be more fluid and based on quicker iteration cycles.

Meet the New Boss

Perhaps the least appreciated barrier to DevOps—particularly for traditional organizations—is the power shift that it implies. In such organizations, IT is frequently a fiefdom, holding a great deal of power within the corporate

2 Margaret Rouse, "NoOps," TechTarget, http://searchcloudapplications.techtarget.com/definition/noops, May 2012.
3 Mike Gualtieri, "I Don't Want DevOps, I Want NoOps,"Forrester blog post, http://blogs.forrester.com/mike_gualtieri/11-02-07-i_dont_want_devops_i_want_noops, February 2011.
4 Barb Darrow, "Does devops leave security out in the cold?" Gigaom, http://gigaom.com/2014/03/12/does-devops-leave-security-out-in-the-cold/, March 2014.

structure. However, the wind direction is changing, in large part because more and more companies see themselves as software-based companies. Guess who rules in that world:

> *"Developers are king, deal with it," said Barton George, Dell's director of cloud development programs... IT alone is no longer a difference maker. Keeping your site up and running 24/7 is just "table stakes..." DevOps—and the rise of the developer class—poses a direct challenge to IT's way of doing things, and the new world order is going to be a bitter pill for many IT traditionalists.*[5]

[5] Fredric Paul, "Meet the 18.5 million people who rule your world," http://www.networkworld.com/community/blog/meet-185-million-people-who-rule-your-world, December 19, 2013.

CHAPTER 2

Ready, Set...Wait... Are You *Sure* You're Ready?

Ready, Set...Wait... Are You *Sure* You're Ready?

Let's say you're an IT manager who wants to introduce DevOps into your organization. You've seen how DevOps is transforming IT in a wide range of companies and helping IT get new products to market faster, and you want to do the same in your company. So what do you do now? You can start by following these four steps.

Step 1: Align with Business Goals

You're probably chomping at the bit to dive into the technology, but before you do, put on your business hat and go talk to your business counterparts. How many? As many as you can. The more clearly you understand the needs and wants of the business side, the better you can leverage DevOps to meet those needs. It's better if you don't even mention DevOps, because they may not know what it is or may have preconceived notions about what DevOps is. So as you start these conversations, try asking questions along these lines:

- How do you measure success?
- What are your goals and objectives for the next year?
- Which of those goals are most at risk?
- What is holding you back from blowing the doors off?

Step 2: Characterize Your Existing Environment

As obvious as it may seem to define the starting point, many organizations simply plunge right in—and then have no way to adequately assess their progress. Invest the time to characterize your current state as thoroughly as possible. Use discovery tools and processes to develop a clear inventory of all devices in the environment. Document the configurations of every configurable component in the infrastructure. Record any information that might be required to reconstruct the starting point. You will never actually do so, but capturing all the knowledge required to return to the starting point means that you have all the information necessary to monitor the changes as your DevOps initiative progresses. This capability will become invaluable for identifying drift from your known, trusted state.

Step 3: Get Buy-in From All Stakeholders

Much of the DevOps discussion focuses on development and operations, but the implications of DevOps extend thorough the organization. Therefore it is vital to clearly define the processes—and their owners—who will be impacted and include them in the planning stage. Establish open channels of communication. Be transparent by keeping everyone in the

loop as the project progresses—particularly when you hit snags. By securing the buy-in of every stakeholder, you increase the probability of success.

Step 4: Establish and Track Key Performance Indicators

Because DevOps fundamentally changes the way that IT does its job, having the right metrics is critical to evaluating the success of a DevOps initiative. Key performance indicators (KPIs) act as a "canary in the coal mine," providing an early warning when the system begins to degrade. An initial set of KPIs might include:[6]

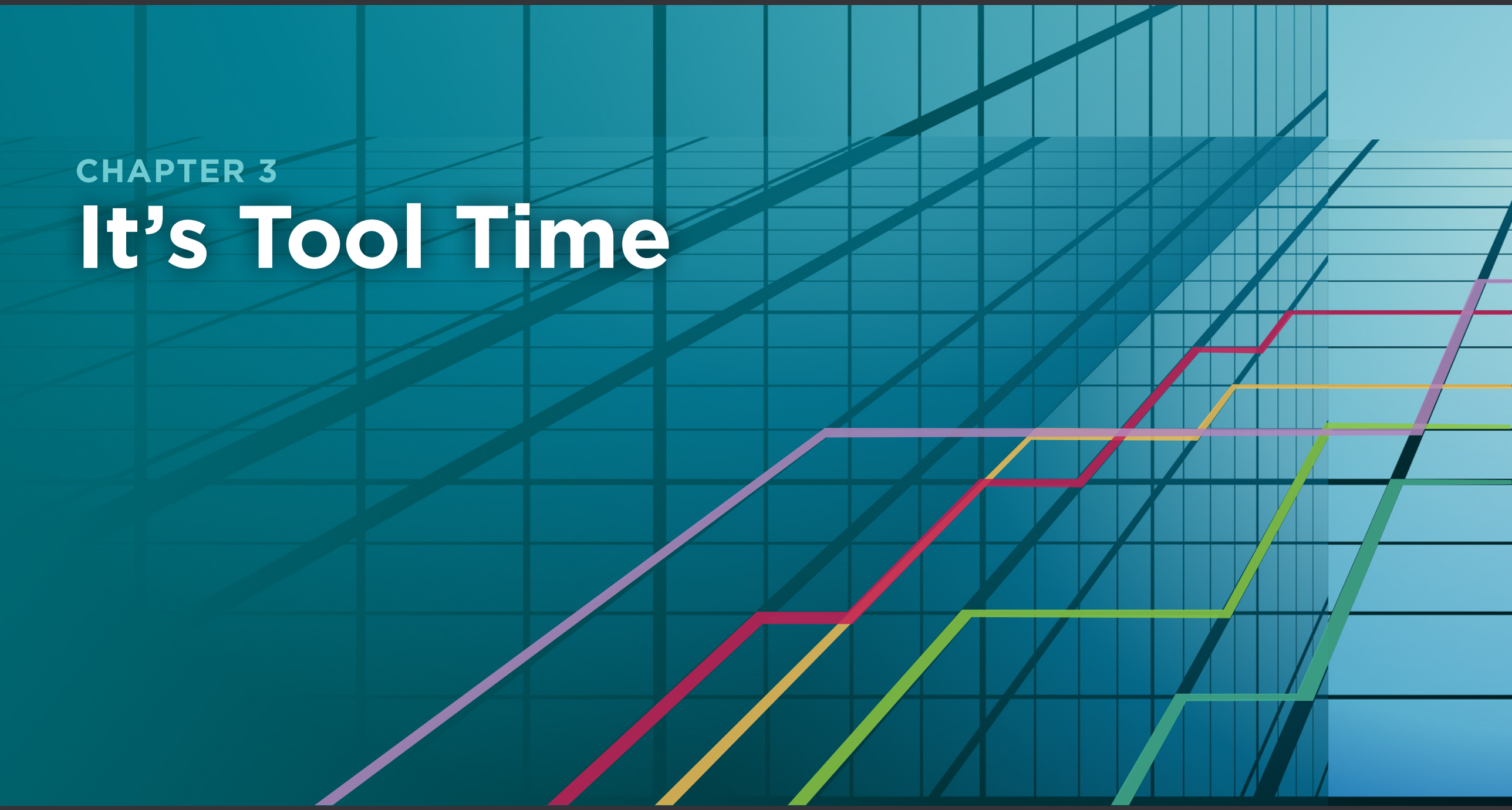
- **Deployment frequency:** Companies with DevOps cultures deploy much more frequently. The extreme case is Amazon, where new code is deployed on average 300 times per hour. While it's unlikely that you will achieve anything close to that number, or will even need to, accelerating releases by an order of magnitude is not uncommon.
- **Change lead time:** Making changes quickly is the very definition of agility. By shortening the change lead time dramatically, one survey finds that "agile organizations can make 8,000 changes before their slower competitors can vet and deploy a single change." Now that's competitive advantage!
- **Mean Time To Recover:** Every organization has failures, but DevOps companies recover in minutes, not hours. Having precise measurements of MTTR helps IT managers monitor the people, processes, and technology that enable rapid recovery and head off problems before they result in significant downtime.
- **Change fail rate:** Even with rapid recovery, it's better not to fail at all. A recent survey finds that the average DevOps group has at least 50% fewer failures from code changes. Any spike in failures allows IT managers to find and fix the organization problem that is driving the increase and ensure that change fail rates remain low.

Now that you have a better idea of how to get started, let's take a deeper dive into the specific tools you'll likely use in a DevOps environment.

[6] "2013 State of DevOps Report," Puppet Labs and IT Revolution Press, http://info.puppetlabs.com/2013-state-of-devops-report.

CHAPTER 3

It's Tool Time



It's Tool Time

Despite the assertions of more than a few tool vendors, there's really no such thing as a "DevOps tool." DevOps experts can get quite fired up on this topic:

> *As far as toolmakers, tool-consumers, and recruiters are all concerned, DevOps is the new black. I get frustrated with the "slap a DevOps on it and they will come" attitude we see prevailing...If it's a tool for scaling or automation or monitoring or cloud, it's now a "DevOps tool".*
> *—Sascha Bates, Consultant, Chef* [7]

Nevertheless, it's hard to overstate the importance of tools in a DevOps environment. The Puppet Labs survey cited earlier found that high-performing organizations share two common practices: Nearly nine in 10 use version control systems for infrastructure management, and eight in 10 use automated deployment tools.[8] If DevOps is in your future, then understanding the tools landscape is essential.

We'll be up front about this: There's no way that New Relic can advise you about the definitive set of tools for your DevOps environment (however, we can make a pretty strong case for using New Relic for software analytics.) Every enterprise is different, every IT group is different, every environment is different, and the tools are changing so fast that anything you read about tools is obsolete by the time you read it.

The goal of this section is simply to give you a head start in the tool arena by defining the basic categories and listing some of the favorite tools used by New Relic's engineering team. And if you find yourself wanting an even deeper discussion of a few specific tools, check out "The 5 Unsung Tools of DevOps," an eBook written by New Relic's site reliability engineer Jonathan Thurman.

Configuration Management

Configuration management—sometimes called infrastructure automation—refers to tracking and controlling changes to the software code base. When your app crashes or some other problem arises, configuration management helps determine what was changed and who changed it. It's an essential practice for establishing and keeping consistent product performance, especially when various developers and system administrators are working on the same code base, as is the case in DevOps environments. Some of the more commonly used configuration management tools include:

- **Ansible**
- **CFEngine**
- **Chef**
- **Puppet**
- **SaltStack**
- **Ubuntu Juju**

[7] Sascha Bates, "Shenanigans," http://blog.brattyredhead.com/, April 5, 2013.
[8] "2013 State of DevOps Report," op cit.

Test and Build Systems

Test and build systems automate a number of developer tasks such as compiling source code into binary executables, running texts, and creating documentation. Typical open source tools for building and testing code include:

- Ant
- Gradle
- Jenkins
- Maven

Application Deployment

Also known as release automation, application deployment tools are critical to continuous delivery of software—one of the key tenets of DevOps. The most popular standalone application deployment tool for Ruby-based Web applications is Capistrano, a remote server automation tool. Fabric and Jenkins are also popular tools for automating application deployment.

Monitoring Tools

There are two monitoring needs in DevOps. Application performance monitoring (APM) provides code-level visibility that enables quick identification of performance issues as well as rapid remediation. The more fully featured APM tools provide trending reports and alerts.

Server monitoring operates at the infrastructure level, allowing reliability engineers to track server health in cloud, physical, and hybrid environments. These tools show capacity, memory, and CPU status for each server so that problems can be addressed early—ideally, before they impact application performance.

New Relic offers a range of monitoring tools, which fall under our broader software analytics suite of products. Other popular monitoring tools include:

- Cacti
- Ganglia
- Graphite
- Nagios (and many offshoots)
- PaperDuty
- Sensu

According to Puppet Labs' 2014 State of DevOps report,[9] "teams that practice proactive monitoring are able to diagnose and solve problems faster, and have a high degree of accountability." So whichever tools you decide to use, just make sure you're monitoring proactively.

Open Source and DevOps

DevOps has its origins in open source. Indeed, for some commentators, the two are inextricably linked:

> *It was the open source movement that gave developers the idea that they were free to code, free to build applications and other software for their companies without needing prior approval. It was open source that crowned developers kingmakers. It was open source that gave developers the freedom to not only write code, but also manage it. It was open source that gave us DevOps.*[10]

[9] "2014 State of DevOps Report," Puppet Labs, http://puppetlabs.com/2014-devops-report, June 2014.

[10] Bill Montgomery, "Red Hat IT's DevOps Journey: harder than we thought," Red Hat developer blog, http://developerblog.redhat.com/2014/03/05/rh-devops-journey-harder/#more-411284, March 5, 2014.

At the same time, it's naïve to expect that commercial tool companies will simply sit this one out and let open source have the field. Traditional vendors such as CA, IBM, and HP offer software development suites that support DevOps with release automation, configuration management, and monitoring. Their selling proposition is that all the necessary tools are supported within a single integrated suite, a claim that is especially convincing for IT executives in large enterprises that have long-standing relationships with these vendors.

While it's seldom smart to bet against the heavyweights, indications are that the open-source world is up to the task of supporting large organizations going forward.

CHAPTER 4

How Do We Know We Did It Right?

How Do We Know We Did It Right?

The proof is in the pudding: You did it right if good things happen. Some organizations evaluate the impact of DevOps by looking at internal factors such as cost, efficiency, return on investment, and other traditional measures. Others monitor key performance indicators and pore over reports that calculate mean time to recover, change fail rate, and deployment frequency. However, there is a growing realization that the best approaches include business-oriented measures such as time to market, customer experience, and of course, revenue.

Fortunately, DevOps has been around long enough to allow early adopters to assess their gains based on measured impact to the business. And those assessments are very, very good. According to recent research, not only do IT organizations using DevOps perform better, they also, more importantly, see higher overall business performance.[11]

[11] Gene Kim, "Enterprise DevOps Adoption Isn't Mandatory – but Neither is Survival," CIO Journal, http://blogs.wsj.com/cio/2014/05/22/enterprise-devops-adoption-isnt-mandatory-but-neither-is-survival/, May 22, 2014.

Conclusion

There are no signs of the DevOps movement slowing down. Companies big and small are increasingly trying to emulate DevOps pioneers such as Facebook, Amazon, and Google in hopes of achieving benefits like greater infrastructure stability, faster deployments, and even improved security. However, in order to get there, you first need to overcome a number of obstacles and know how to get started.

One of the keys to DevOps success lies in tools, many of which are open source. Automating tasks such as version control, server configuration, testing, and even the deployment process itself pay huge benefits. Monitoring tools such as New Relic also play a major role in enabling DevOps teams to deploy faster and with greater confidence. To learn more about DevOps, visit: http://newrelic.com/devops.

About New Relic

New Relic is a software analytics company that makes sense of billions of data points about millions of applications in real time. New Relic's comprehensive SaaS-based solution provides one powerful interface for Web and native mobile applications and consolidates the performance monitoring data for any chosen technology in your environment. More than 250,000 active users employ our cloud solution, analyzing more than 200 billion data points across more than 3 million applications every day.

The DevOps movement is focused on helping dev and ops teams improve the odds of application success. New Relic provides the data and accountability application teams need to measure and monitor the impact of new features, prioritize fixes, ensure stability, and keep costs in check. As a DevOps-driven SaaS company, New Relic understands the specific challenges software teams are facing, and has built specific features with agile app delivery in mind.

New Relic, San Francisco HQ
188 Spear Street, Suite 1200
San Francisco, CA 94105

New Relic, Portland
111 SW 5th Avenue, Suite 2800
Portland, OR 97204

Tel: +1.888.643.8776
support@newrelic.com
www.newrelic.com

New Relic, Seattle
2101 4th Avenue, 19th Floor
Seattle, WA 98121

New Relic, Dublin
34-39 Nassau Street, 3rd Floor
Dublin 2, Ireland