The Road to **DevOps** ROI

Pradeep_Prabhu, 8 Aug 2013 CPOL

DevOp ROI relies on combining best practices with the right tools. Tooling enables practices, and vice versa. The CloudMunch platform makes it happen.

Editorial Note

This article is in the Product Showcase section for our sponsors at CodeProject. These articles are intended to provide you with information on products and services that we consider useful and of value to developers.

Understanding the interplay between best practices and the tool chain

DevOps is gaining traction as a software development methodology. Though it may not be the right fit for every development organization, the idea of collaboration between developers (Dev) and IT operations (Ops), as well as testers, solves a lot of problems inherent in organizations that must release code in rapid cycles. It's pretty intuitive. Who wouldn't want better coordination and tighter, more productive connections between the people who develop code and those responsible for testing and making it work in the real world?

Making **DevOps** pay off can be a little tricky, however. Return on the investment (ROI) in **DevOps** derives a combination of faster rollout of quality code to customers, increased productivity in both Dev and Ops teams, as well as potential labor cost savings across the entire process. To realize these outcomes, it is necessary to keep quite a few moving parts running in a coherent structure. In our experience, getting to success with **DevOps** involves mastering the methodology itself with best practices as well as the tool chain that is used for its implementation.

The need to align practices with the tool chain makes **DevOps** different from other IT paradigms, where the methodology and tools can be distinct. An organization can pursue cloud computing, big data or event-driven architecture, to name a few popular trends, and get away with selecting any number of tools to get the job done. **DevOps** is different. In **DevOps**, practices and tooling are deeply intertwined. The unique nature of software (it either works or it doesn't) and the fragile connections between distributed stakeholders put pressure on tools to make practices work.

For the last several months, we have been working with **DevOps** teams at Fortune 500 companies as well as those at lean start-ups. The challenges of implementing Continuous Delivery in these demanding environments have shaped our thinking and influenced the functionality we offer in our full stack CloudMunch **DevOps** platform. Channeling Marshall MacLuhan, we have come to see that *the tooling is the practice* and *the practice is the tooling,* so to speak. The following aspects **DevOps** show the value of this fusion between best practices and tool chain:

Making early, instant feedback available in a single pane of glass for all roles

Figure 1: DevOps dashboard for cross team data viewing

DevOps is based on collaboration between developers and operations teams. This represents a cultural shift for larger teams, where it has been historically difficult to get everyone to share goals and metrics. When it works, though, people get more productive and agile. Developers, testers, product managers and system engineers all deliver great results and see improvement in the qualities of their work lives. Collaboration becomes the norm of the work culture. The only way to ensure that every developer, tester, and system engineer is a high performer is to make sure they get early and instant feedback. When a developer pushes code into the repository, he or she should get immediate feedback on that it. A product manager needs to be able to create a sandbox to show the new feature to business stakeholders to get feedback on day one. System engineers have to be able to simulate new features in a production environment while the code is still in development.

Bringing early, instant feedback practice to life involves tooling. *The tooling is the practice*, in three senses: There has to be ongoing intelligence of the code in development. Operational intelligence and an automated dashboard for analytics are also essential. Development and operational tools should be configured to generate metrics. For example, a Continuous Integration tool could record when a build occurred, how many tests ran, how long the tests ran, whether the build was successful, how many tests were successful, which files are frequently changing and so on. This data can then be analyzed and displayed in automated dashboards. Operational intelligence is an aspect of application and infrastructure monitoring. It provides a mechanism feedback across all functions and roles in a single dashboard so that everyone can see it and act upon it as needed. Analytics and an automated dashboard provide real-time insight to the entire team across roles so that they can get early and instant feedback on all aspects of their respective tasks. If done right, the result is a productive, efficient **DevOps** team.

One-Click Access to the Entire Application Lifecycle

With **DevOps**, collaboration needs to be seamless across Dev and Ops teams. It has to function smoothly across the entire application delivery value chain. Good **DevOps** starts with setting up automated processes and systems for Integrated Configuration Management, Continuous Integration, Automated Testing, Deployment Planning, Infrastructure Provisioning, Continuous Deployment, Integrated Change Management and Monitoring. These practices are inextricably linked to tooling. *The practice is the tooling.*

It essential to integrate all these tools in a full stack **DevOps** platform so that everyone shares common tooling and won't need to hand off artifacts between teams and risk creating gaps. It means testing and deploying application code and infrastructure code in the same pipeline. Productivity and quality problems inevitably arise when **DevOps** teams create a patchwork of mixed toolsets. Things may work well at first, but as scale and complexity grows, the tool chain starts to break and ceases to provide the visibility and metrics the team needs to make better decisions. Making things worse, scale and complexity typically trend up with the success of the Continuous Delivery model.

Figure 2: An integrated toolchain ensures seamless handoff

Multiple deployments every day are not uncommon as new features are released. As **DevOps** flourishes, the business gets used to more innovation from IT and the cycle grows more intense. This is known as having a rapid innovation culture. It arises when **DevOps** best practices and the tool chain are one. Having a fully integrated **DevOps** tool chain with one-click access to the entire application lifecycle has been proven to help teams scale effortlessly.

Develop and run seamlessly

Applications have to be designed to morph with business needs. This best practice is the essence of **DevOps**, one that is realized through the tool chain. A company may start with an application running in a public cloud but soon needs to move it behind the firewall or have the back end moved to a big data platform and so forth. When a **DevOps** team starts developing applications, it usually has a target environment in mind. However, the best practice is to ensure that there is flexibility in both development and runtime environments so the application can change as the business evolves. For instance, the team might use a public PaaS today but tomorrow it will need to shift to an internal private cloud.

Tooling the **DevOps** team for flexibility involves elastic provisioning of systems for Contiguous Integration, automated testing, deployment and production environments. *The tooling is the practice*. It is imperative to have the flexibility to develop/test/run in any environment seamlessly.

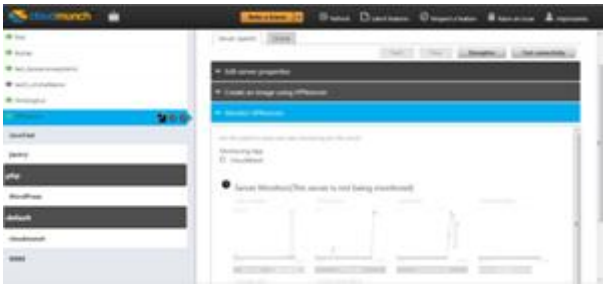

Figure 3: Enable provisioning to anywhere

The only way to do this is to provision the infrastructure and tool chain anywhere and carry the application stack across these environments. Projects and profiles can be tracked and versioned so that previously configured systems can be re-created with a single click to easily test new code, revert to previous configurations, and compare system versions. A fully searchable repository must be available to facilitate standardization and reuse.

DevOps ROI: Making the Tooling and the Practice One and the Same

The **DevOps** platform is the connection between the tool chain, best practices and **DevOps** ROI. In the case of CloudMunch, we enable the **DevOps** team to plug a large number of different development tools, PaaS platforms and production environments into the core platform. Centralized as a platform but multi-faceted as the team requires, CloudMunch gives each participant access to, and the ability to give, instant feedback on any aspect of the process. Integrated across Dev and Ops teams, CloudMunch provides that sought-after one-click access to the entire application lifecycle. CloudMunch is also built for the flexible movement of applications between diverse production environments. All three of these practices enabled through tooling drive efficiency and productivity, which in turn leads to the kind of rapid deployment and low cost **DevOps** that the methodology promises.

To learn more about DevOps ROI, read the full whitepaper.

License

Share

About the Author

Pradeep_Prabhu

United States

Pradeep Prabhu is co-founder and CEO of CloudMunch, a **DevOps** management platform provider. Previously Pradeep was Vice President and Head of SaaS (Software-as-a-Service) for Infosys, where in his 18-year career he was instrumental in growing the tech company to a multi-billion dollar giant.

You may also be interested in...

- Active Directory Webservices for DevOps
- An Introduction to DevOps
- DevOps needs DevSec
- SAPrefs - Netscape-like Preferences Dialog
- DevOps in 2015 – Beyond Basic Metrics
- Generate and add keyword variations using AdWords API

Comments and Discussions

You must Sign In to use this message board.

Search Comments Go

First Prev Next

Error Getting Whitepaper

Mr. Iman Mayes 4-Nov-14 7:41

Hi,

I am interested in getting the whitepaper you link to in your article regarding the ROI, but the form returns a server error. Any chance of fixing this?

Thanks,

Iman

Sign In · View Thread · Permalink

Refresh 1

General News Suggestion Question Bug Answer Joke Praise Rant Admin