WHAT IS A DEVOPS TOOLCHAIN?

The idea of a toolchain isn't new. It's not incredibly abstract, either, like most things in DevOps. A toolchain is simply a digital set of tools that aid in a specific objective. When it comes to DevOps, the toolchain is a combination of the most effective tools for developing, delivering and maintaining software according to agile principles.

The following article explores the DevOps toolchain, giving a brief introduction to what it is and how to use it. If you're looking for the best combination of tools for your DevOps enterprise, you've come to the right place. We'll cover tools, too, so you have everything you need to make the best decision about DevOps toolchains. Let's get started:

(This article is part of our DevOps Guide. Use the right-hand menu to navigate.)

Introduction to the DevOps Toolchain

In DevOps, key principles are continuous integration and continuous delivery. The DevOps toolchain assists businesses in achieving the promise of DevOps by maintaining a healthy software development pipeline. Toolchains help team members complete and simplify more complex tasks of the development process.

If DevOps is the new industry standard, implementing a DevOps toolchain is the next iteration of it. Toolchains can be created, orchestrated and stored in the cloud using some key aspects of a DevOps approach:

- Automation
- Integration

- Self-service
- Collaboration

While creating a toolchain is a good way to flex your DevOps muscles, it also takes time resources and maintenance. As such, it is good to do the planning in advance to ensure overhead requirements can be met. Organizations also must be prepared to promote adoption of the toolchain with developers, who may be set in their ways. Otherwise, the whole effort could be for naught.

If you've decided you have the necessity, resources, and grit to get started on a toolset, here's some insight on how to build one:

How to Build a DevOps Toolchain

One aim of the DevOps toolchain is to be homogenous with the standardization of the DevOps framework. In that way, a toolchain integrates seamlessly with other DevOps tools and processes and helps developers follow the same consistent flow every time. Standardization and consistency is an important function of the DevOps toolchain, in a framework that is sometimes otherwise abstract and left for subjective interpretation. A toolchain quite literally shapes the processes of how you apply DevOps, so using the right one is important.

There are two ways to build a toolchain:

- In-the-Box; and
- Custom

An in-the-box toolchain is a solution that has been created by someone else and provides standard offerings that you can select from to tailor to your unique needs. Using a pre-orchestrated set of tools allows for greater standardization and integration with less manpower to do it.

If an in-the-box solution isn't for you, the other option is to opt to customize a toolset yourself. When you do this, you select the discrete tools you need for your toolchain and carefully orchestrate them to work together in your DevOps pipeline.

If you choose this approach, you can avoid being locked into tools or vendors, but it can be more budget prohibitive. You'll also need to ensure you have the resources to man this process, and standardization isn't as much of a given.

A Healthy Toolchain

If you spend enough time poking around the web, looking for answers, you'll see sometimes a DevOps toolchain is referred to as "healthy," as is the accompanying pipeline it supports. Creating a healthy toolchain allows you to produce the best software most quickly. That's because a healthy toolchain is one that supports a pipeline poised for continuous integration and continuous delivery -- a healthy pipeline, if you will. A note on a healthy pipeline is that its one that uses automation and integration to move down a logical flow to completion. Below are some tools and categories that make a pipeline healthy.

Must-Have Tools for Your Toolchain

To have a healthy toolchain, it needs to meet the needs of your development pipeline. Savvy developers must include tools for each of the following:

- **Planning and Collaboration:** These are the tools that help with sprint planning, offer transparency for stakeholders and opportunities for collaboration. Such things like project management tools, like Asana, could fall into this bucket as well as communication tools like Slack.
- **Source Control Tools:** These foundational tools help developers control source code across all assets and properties. A tool like this centralizes where and how code is stored and managed. Subversion would be one example of this.
- **Tracking and Escalating Issues:** These straightforward tools help developers catalog and track issues to resolution with increased responsiveness. For reference, a tool of this nature is Jira.
- **Continuous Integration:** These are largely collaboration tools that offer dashboards for stakeholders and developers, providing transparency and opportunities to work together throughout the CI process. *Think* Bamboo.
- **Configuration Management:** Configuration management tools do just that. They offer a centralized avenue for standardizing configurations across assets. Puppet and Chef are two household configuration management brands.
- **Artifact Repositories:** Artifact repository tools store all of the bulk binary artifact files that rarely, if ever, need to be altered or changed. Nexus offers a tool like this.
- **Monitoring:** Monitoring tools offer important information about how a piece of software is functioning and if there are any threats or security issues. A popular monitoring tool is Sensu.
- **Testing and Automation:** With automating testing tools developers can quickly validate code. These include QTP and TestComplete.
- **Deployment:** Finally, deployment tools, like IBM uDeploy, help developers achieve continuous delivery by assisting with the speed and quality of frequent deployments.

3 Uses for a DevOps Toolchain

You may have figured out that the DevOps toolchain is capable of a lot of things, but it does have some typical uses. Below, we are going to explore what they are:

Faster Time to Innovation

By standardizing a pipeline where software is always being deployed, a DevOps toolchain helps enterprise businesses innovate better and faster. These tools assure agile and rapid delivery of software products with monitoring, automated testing, and diagnosis capabilities. Businesses who innovate more quickly can ensure they keep an edge on the competition.

Fine-Tuning Incident Control

Incident management can be a problem for even the most agile fast-moving teams. Using a DevOps toolchain helps developers work from a place of incident control. Utilizing a mix of automation and good, old-fashioned collaborative efforts teams respond faster and more effectively to incidents with tools designed to simplify this process.

Quality Assurance

One of the more emphasized uses for a DevOps toolchain is to resolve software defects quickly and

accurately to assure quality releases. With automated notifications, everyone can get on the same page faster to major problems with software. This helps two-fold: offering a boost to resolution speed and end-user satisfaction.

Understanding the DevOps Toolchain

A key component of DevOps is creating an integrated environment where multiple cloud tools and in-house resources can work, together with automation, to produce quality software releases quickly in a way that is repeatable again and again. Now that the DevOps toolchain is out of the bag, there's no ignoring it as an essential component to the overall DevOps process. That's because the toolchain works in step with the goals of DevOps, and the same principles can be applied to building a toolchain as they are to developing a piece of software. That's to say that installing a toolchain requires collaboration, automation, self-service, and integration.

There are a number of tools and sets to choose from, but it's important you cover the right ground to offer the most homogenous pipeline. This means selecting tools in categories like planning and collaboration, configuration management and deployment. Other tools might be less obvious like binary artifact repositories. However, these tools are still important to the overall picture, so it's recommended you take inventory of your needs before you start this process.

Alas, selecting the right DevOps toolchain could be as easy as finding the right in-the-box solution or as complex as customizing your own. Before you get started on your DevOps toolchain, consult with the experts at BMC, today.