DevOps for IT Pros: The Key to Addressing Bi-Modal IT

by Curtis Preston, Senior Analyst

DevOps can deliver a strong return on investment (ROI) for those environments choosing to invest in it. Environments adopting a DevOps mentality move from merely supporting operations to truly enabling better, more efficient operations. IT moves from merely changing the oil and swapping tires to designing an engine that uses less oil and tires that need fewer changes.

IT professionals of all types should get a basic understanding of the DevOps phenomenon, enabling them to decide whether this approach is right for their organization. If it is, they can make an informed explanation to their CIO and CFO exactly how it will, or will not, benefit their companies. Even organizations that are not ready to move toward a full DevOps adoption will benefit by understanding it, and incorporating some of the concepts into their existing operations

What is DevOps

DevOps is a methodology that creates a much tighter relationship between the development team and the operations team *with the goal of faster software product delivery*. The DevOps methodology applies to delivering new software, as well as supporting frequent updates to existing apps that include both bug fixes and feature requests. It's origin stems from the disconnect, real or perceived, that often exists between those who develop applications for the company (i.e. the *Dev* in DevOps) and those who support the operations that enable those applications to run (i.e. the *Ops* in DevOps).

In a DevOps environment, the development team must have a solid understanding of the operational requirements that their code places on IT infrastructure. At the same time, the operations team must have an understanding of the code running on that infrastructure. By thinking like developers, they gain a greater understanding of what it's like to develop code. Together these two teams can enable smoother running applications on a solidly running infrastructure. This collaboration results in better applications they can produce and fix quickly.

The focus on releasing and enhancing code quicker is directly tied to the immense financial benefit it can have for a company. It allows companies to release new products faster, giving them a competitive advantage. Quickly fixing any issues that come up with the code, as well as adding feature requests along the way, makes for happier customers. This increases customer traction, which has an appreciable effect on the company's bottom line. Quick code fixes are also cheaper than long, drawn-out ones, so that doesn't hurt the bottom line either.

Agile Development

Constantly updating code creates the need for developers to be very *agile*. The focus is on creating the best possible product at this moment, and then repeatedly making it better and better with a continual focus on the customer. *Agile development* environments welcome changing requirements – even late in the process – because they represent the wishes of the customer and will make the product better. This fast-moving, frequently changing operation places significant demands on the infrastructure that supports the effort.

Agile developers need to be able to easily stand up an environment that easily replicates the environment that the code is living in now or will live in upon its release. They also need to easily adjust that environment if the demands of the code create new demands on the infrastructure. Perhaps they need more processing power, more memory, more storage, or different storage. They need to be able to adjust as developers and without having to make a phone call or create an IT support ticket just to spin up a new VM. This is the challenge that DevOps addresses.

Operations

Before DevOps, an IT person would create a host, attach a LUN or NFS volume to it, perhaps install some database or web software, etc. There is little to no automation in this process. It always involved one or more operations personnel doing these tasks at the behest of a development person. This way of doing things is a time-tested way of ensuring the security, stability, uptime and performance of IT, but was never focused on speed and agility. Unfortunately, modern developers are used to the agile development process and expect IT to respond accordingly. When they don't, the ubiquity and ease of access to the public cloud leads to the very quick development of shadow IT, which is never good idea.

The operations team's should be to support agile development and production by giving developers the tools that make their new programs possible. Automating each of the steps of creating a development or production environment enables the development team to do what they need to do with a push of a button or execution of a simple command. This is referred to as *infrastructure-as-code*, in that the operations team is now responsible for creating code that automates the creation and destruction of infrastructure.

In the old days, a new version of Apache might be available, for example, and someone might request its installation. Today's operation person tests the new version and then creates the code so that this new version can be selected as a valid configuration for the developers. They simply need to choose OS version X, Apache version Y, and mySQL version Z and the code infrastructure defines the infrastructure on the fly, without any help from the operations team. If something goes wrong, the operations team is concerned primarily over why its design of the infrastructure code did not work. Just like a developer wants his-her customer to have a predictable product experience, the operations team wants the development team to have a consistent experience when they call on the infrastructure to do something for them.

There are a few fundamental "must-haves" for the infrastructure in order to deliver the infrastructure-as-code concept. It requires a data management and orchestration layer that links storage and data with compute and gives organizations the modernization services they need: more automation, user self-service with proper security controls (role based access) and the ability to drive things both through a UI interface as well as through code via APIs. Therefore adoption of the DevOps model frequently leads IT teams to pursue significant changes to the infrastructure. Orchestration software that integrates with existing infrastructure and delivers the required infrastructure as code functionality is often an attractive alternative in that it can move the organization toward the DevOps methodology without a significant "rip-and-replace" effort to swap out the existing infrastructure.

StorageSwiss Take

DevOps is an exciting way for IT organizations to demonstrate solid ROI to their senior management. Releasing code sooner and updating and fixing it faster gives them a competitive advantage, makes for happier customers and reduces cost. To support this methodology, the development team needs to be able to quickly stand up and stand down environments for development, test, and operations. The operations team makes this possible by creating infrastructure-as-code, or code that makes infrastructure appear and disappear at the behest of the development team. Very often, the changes required to adopt DevOps necessitate significant investment in new infrastructure. Innovative vendors like Catalogic Software are allowing customers to move toward DevOps without the infrastructure upheaval by delivering data management orchestration software that enables infrastructure as code leveraging existing infrastructure. It is clear that DevOps is a force that will remain a key part of IT for the foreseeable future. Whether pursuing a full-scale DevOps adoption, or simply seeking incremental improvements to existing operations, IT professionals will be wise to understand the DevOps methodology and consider how its concepts can help within the environments that they are responsible for.