siliconANGLE

7 Lessons Learned in the DevOps Trenches

http://siliconangle.com/blog/2012/03/07/7-lessons-learned-from-the-devops-trenches/

March 2012
By Klint Finley

Yesterday our partners at Wikibon hosted a teleconference on DevOps as part of its Peer Incite series. You can watch the video here. During the conversation J. Wolfgang Goerlich, information systems and security manager at a midwestern financial services firm, explains how he turned his company into a DevOps shop and answered questions from the community about what DevOps is and how to implement it.

Goerlich joined the systems security team the firm in 2005 as a bridge between the software development team, the investment development and the operations team. In 2008 he took over the network operations team. They were going through a radical virtualization project, and Goerlich knew it was time to take a different approach to managing IT. Goerlich started looking at the principles of agile development and how they could be applied to the operations world. Goerlich's team moved away from doing quarterly installs to a more agile system where changes happened every day without affecting business services. All of this was done on the operations side. It wasn't until 2010 that the development team at the firm asked Goerlich if could help them apply agile to development. Goerlich ended up merging both operations and development under one team with one budget, a process that took a full year. The result is what Goerlich calls "one team, one system."

Goerlich says he didn't model the process on any other company. The concept of "DevOps" by that name was just getting started as Goerlich was revamping the firm's departments, so although he kept an eye on the emerging DevOps movement he was largely making it up as he went along.

Under Goerlich's management the team managed to reduce its number of physical servers from 140 down to 43. Goerlich has also started the process of cutting down the total number of applications deployed by dividing them into three categories: 1) Business intelligence 2) SharePoint and 3) Not my problem/software-as-a-service. Goerlich says that although he's testing applications in the public cloud (specifically on Microsoft Azure), he's not using the cloud for anything in production yet. He has a broad concept of SaaS that includes on-premise software managed by third parties.

By consolidating the teams and reducing the number of servers and software that are in play, Goerlich was also able to cut the size of the IT team down to nine full time employees from its previous 26-28 employees (depending on the number of consultants working at any given time).

Here are some lessons Goerlich has learned along the way:

1. What Is DevOps?

First, let's define our terms. Goerlich describes DevOps as a system for taking best of development and the best of operations and making something that's better than the sum of its parts. At the core of it, it's about making infrastructure more like code in such a way that you can actually "version" your infrastructure. But virtualization isn't the key thing, he says: automation is. Virtualization is just a way to support automation.

This leads to the ability to do agile deployments and make changes quicker. Goerlich says that when he was hired in 2005, he was often introduced as an "agent of chaos" because he kept wanting to change things. He says that for the operations side, change is scary but necessary. Old school systems admins always have servers they're afraid to reboot and software that they don't want to let go of. And they don't like running patches because they're afraid of breaking things. The old change control model involves making lots of big changes all at once instead of making lots of small changes constantly. Continuous integration is about being willing to change things, and figuring out how to do it without affecting the end users' ability to get their work done.

2. How to Get Started with DevOps

Goerlich has separate advice for developers and operations staff trying to implement DevOps. He suggests that developers who want to get started with DevOps start by reaching out to ops and getting themselves on the call list so they can see what sort of issues ops has to deal with every day. If you're in infrastructure, he suggests reaching out to development and getting in on some build meetings and learn about how that process works and the pressures they are under.

In terms of selling the whole thing to management, he says you've got to be able to demonstrate the business value of DevOps. You need to tie it to a business initiative and show cost savings. He says his team established a good track record of meeting deadlines and getting projects done faster that previously thought possible and that brought management support.

3. What You Need to Measure

Metrics are an important part of measuring success, not just for making the case for DevOps to management but for making sure that you really are doing it right. Goerlich suggests three specific areas to measure:

1. Implemented changes: How frequently do you change? You don't have to be making changes every 11 seconds like Amazon.com does, Goerlich says (he says his team pushes about 47 a month). But you do want to keep track of how many changes you're pushing out so that you can prove that you're doing more.

2. Successful changes vs. failed changes How many of these changes had to be rolled back? Track this so that you can prove that you're making fewer mistakes.

3. Help desk tickets related to changes vs. help desk tickets not related to changes. Are you causing more problems for end users or fewer? Track this to prove that you're not only doing more, but you're doing it better and helping users get their work done.

4. One Team Means One Tool

Goerlich says that although automation tools like Puppet and Chef are great, the "one team, one system" approach means viewing infrastructure through a single pane of glass. So his team uses Microsoft System Center Suite to manage and automate its servers. Microsoft Hyper-V is their hypervisor solution. Goerlich says this will enable them to manage everything run on-premise as well as anything they run on Azure with just one system. That will make moving applications to the cloud, if and when they decide to do so, easier.

5. Keep Your Employees Smart

We've already covered Goerlich's role in the anti-stupid movement. Goerlich says his team spends 20% of their time on training so that they can not only meet current requirements of the company but be ready for the technological changes coming down the pike.

6. What a DevOps Superstar looks like

Goerlich says three things make a DevOps superstar:

1) Knowing your own area inside out.

2) Not limiting yourself to your own area. If you're a developer, learn about infrastructure and security. If you're in ops, learn about development.

3) Taking responsibility for your input. Goerlich says as a manager he doesn't want to be responsible for managing timesheets, forms and making sure employees are putting in enough hours. He wants to measure an employees output. It's up to an employee to make sure that they put in enough time and effort to get the output required.

7. Vendors and Consultants Can Be a Bottleneck for DevOps

Goerlich says that vendors and consultants tend to have a hit and run mentality that's incompatible with DevOps. They just ship code and it's up for the customers to support it. When something's developed in-house under the DevOps model the developers have skin in the game because they have to support it. He notes that when you call a vendor for support you always know you've got a serious issue when you hear that they're the getting developers involved. Vendors only get developers involved in a very high escalation. He suggests that vendors can become more like DevOps shops by getting developers more involved with tech support so that they "feel more of the pain customers feel."

How Does DevOps Scale?

When asked whether DevOps can scale beyond his small team, Goerlich says he thinks it can because it's been done elsewhere, such as at Amazon.com. He says that having only a single umbrella like he does now might not work if the company grew, in which case he'd have to start diving the team up depending on what they're working on – such as Web apps, line of business apps, etc. He says the big thing that would change is that he wouldn't be able to provide as much career development guidance to his employees as he does now if the team grew.