Making the Transition to DevOps

docker

Table of Contents

Overview

Driven by the need to improve product quality and accelerate innovation, the DevOps methodology has grown in popularity amongst software teams over the last few years. But this transition is not easy. There isn't a single technology tool or platform that allows you to simply turn DevOps on or off. DevOps requires a cultural change to occur within the organization; in today's world, development and operations teams are both physically separate and each has their own processes. The critical part of the transformation is breaking down the traditional barriers that exist between Dev and Ops.

Taking a step back and in the spirit of learning from those who traveled the DevOps journey, we have gathered three experts from Docker, Chef and Microsoft to discuss seven questions on DevOps:

1. What does DevOps mean and why is it seen as a better way of doing things?
2. Where is a good place for teams to start their DevOps initiative?
3. Where have you seen organizations stumble in their DevOps journey?
4. What are the tough questions teams should ask themselves before launching a DevOps initiative?
5. Share a DevOps success story and why it was a success.
6. When thinking about large technology shifts, how can DevOps enable these dramatic transitions?
7. How are your respective companies participating in the DevOps transition?

Contributor Profiles

Matt Bentley

Matt Bentley is a Solutions Engineer at Docker and has been using Docker since late 2013. He has been working with teams to develop a DevOps philosophy for years as a consultant and technical advisor.

@matthewbentley
github.com/mbentley

Jessica DeVita

Jessica DeVita is a Solutions Architect at Chef Software. Prior to Chef she was an IT Pro Technical Evangelist at Microsoft. She helps teams adopt automation capabilities with Chef and Powershell/DSC and address the cultural components of DevOps initiatives.

@UberGeekGirl

Volker Will

As Chief Technical Evangelist, Volker Will leads the DevOps Technical Evangelism team at Microsoft. His team works with customers and partners around the world about DevOps. They work on sharing best practice based on customers and POCs. Work on joint activities with joint customers.

Highlights

Defining DevOps

There is no single definition of DevOps. It is a cultural shift involving changes in how people are organized and the processes they facilitate and is not defined by technology or tools. Depending on the organization they will likely define DevOps in relation to where they reside on that spectrum. The methodology emphasizes communication between developers and operations teams and operating within short, fast feedback loops. This enables companies to not only build fast but fail fast, and have the ability to quickly remediate their mistakes. In the end, this is what will lead to greater efficiency and better software - the core benefit of DevOps.

Embarking on your DevOps journey

There is no simple guide for "Doing DevOps." It starts with the breaking down of traditional silos, improving communication between development and operations teams and creating new structures and processes. This communication allows the two teams to better understand each other's requirements and work more closely together versus having a single hand off point. A good place to begin is to take a small project, and apply a new method to the development, test, deploy and remediate process to retrain the team's muscles instead of trying to convert your entire company over to DevOps in one night. In this process you will tweak and adjust how teams are organized and how feedback loops are handled to find the right balance for your business.

Common stumbling blocks

The single most critical blocker is when organizations fail to move away from a siloed environment. This fundamentally defeats the purpose of DevOps altogether. Also, DevOps is not just about technology tools. The tools help, but you need to make sure the people and process changes are happening in conjunction and give yourself time to assess the knowledge base and expertise of your team, operational readiness and experiment with different tools. This will more accurately determine the tools in which you will need. A lack of assessment of the current situation can create roadblocks in the transformation.

Ask the tough questions

The most important question to ask yourself is "do you know how to measure success within your organization?" If the answer is no, then their is some prerequisite work that companies should do before even considering making the transition. Understanding where your organization stands now is also important. Ask questions like "How does your software lifecycle look today?" "What are we doing for automation today?" "What tools are we currently using. Is there cross functional teamwork going on already?" These are the questions that companies can't forget to ask themselves.

Views and Comments from our Contributors

1. What does DevOps mean, why is it seen as a better way to do things?

Volker: There is no one definition for DevOps. It has to do with improving the flow of software architect from ideation to end users and back. It involves many different things. It's as much a people thing as it is about the tools they use that enables improved collaboration between groups. It has to do with automation along to the software development cycle. Developers are agile for a longer time than operations. Many of the ops teams have some catchup to do.

Jessica: If you ask that question to 100 people you will get a hundred answers. People are recognize how complex their environments are, and they will go out to find tools that will help them solve this complexity, as if the holy grail begins with the tools. But, when you start to move fast with these software tools, things break a lot faster. This is where the culture stuff comes up. People are looking for a more humane, better way to manage systems of doing thing.

Matt: Enabling operations with tooling. Long gone are the days of throwing code over the wall and hoping it works. A lot of it is communication between developers and operations teams. That seems to be the most important piece of it all. Just getting everyone talking and reading from the same playcard really makes it an easy way to learn and fail fast. The ability to recover quickly and takes the blows is important.

2. Where is a good place for folks to start when they begin a DevOps initiative?

Jessica: There are no simple 5 steps to get there. It is a journey, and people can support it by working in small batches, making their work visible, and committing to a blameless culture that responds to failure with learning. We've seen success with organizations that invest in building an internal DevOps community in the organization so they can start to understand what other teams are working on. Doing a 2-day workshop can a be a catalyst for real change.

Matt: Communications is a real big place to get started. Getting everyone educated. Developers need to be able to tell the ops team the big things that are coming. Ops will then be well aware that this is coming. Visibility is key here - much more visibility into what's going on in a day to day basis. Enabling the process through tooling lets everyone see what's going on through the entire lifecycle. Communication is critically important.

Volker: Communications across the so called siloes is something that is extremely important. In particular, as one of the promises of DevOps is being faster more efficient, more agile. You don't want to boil the ocean though. A good place is to start small, take a project. Don't try to convince your mainframe team to convert all their projects into an agile and DevOps style. That's most likely not going to work. Addressing the current state across the stage is important.

3. Where have you seen organizations stumble in their DevOps journey?

Matt: Volker mentioned siloes. And siloes is a great way to explain what most orgs are trying to transition out of. That's one of the biggest problems. Companies get comfortable with siloes. Companies may be using DevOps practices, but they are doing them within siloes - and that doesn't really help anyone. Forgetting that DevOps really is a proactive paradigm and not just a job title. They assume that they are on the same page, but in fact they are not. And the breakdown in communication seems to be the biggest cause of organizations to fail with DevOps.

Volker: One of the things that we see in talking to customers and where they fail is in enforcing DevOps without really understanding what DevOps, in the essence, is about. It's not just about automation. It's not just about cool tools. You need automation, and you definitely need the tools that support your teams in accomplishing their goals, but you don't start with selecting the proper tools. You have to experiment. You have to see what kind of people do you have on your team? What kind of project are you looking at? This is something where many organizations have challenges. If you start your DevOps journey with some of the very obvious practices like infrastructure, you need to access where you are. The people on your team, what kind of knowledge do they have? Are they used to communicating? Or are they living in what we just called Siloes. These kinds of things are very important. So, in my opinion, a lack of assessment of the current situation, and a careful selection of the project that you want to go after, applying some of the DevOps practices, is where we see a lot of organizations, not fail, but stumble. Its very well described in the question. And stumbling in the DevOps way is not a bad thing. You should stumble, you should make mistakes, and you should definitely learn from these mistakes, and pick up again try a different approach, learn. Have an iterative process And if the team leads or upper management says "you do DevOps now. Figure out how it works." Figuring out how it works is a good thing. But demanding to do DevOps is not necessarily a good approach. And in many cases what we see is that companies start with a bottoms up approach. But even that can't be successful without proper buy in from all teams, all organizations, and the entire hierarchy involved.

In my opinion, a lack of assessment of the current situation, and a careful selection of the project that you want to go after, applying some of the DevOps practices, is where we see a lot of organizations, not fail, but stumble. Its very well described in the question. And stumbling in the DevOps way is not a bad thing.

You should stumble, you should make mistakes, you should learn from these mistakes, and pick up again try a different approach - learn. Have an iterative process Figure out how it works." Figuring out how it works is a good thing. But demanding to do DevOps is not necessarily a good approach. And in many cases what we see is that companies start with a bottoms up approach. But even that can't be successful without proper buy in from all teams, all organizations, and the entire hierarchy involved.

Jessica: Prepare for failure. It is about recovering from issues and outages, and taking that learning into the next iteration. One stumbling block is this tendency towards local optimization - trying to make changes in one team - and that was never the constraint. I see a lot of organizations spending a lot of time and money, but at the wrong part of the system. I think about entering work around a product, or a particular project, and involving lots of different people and teams in different areas of expertise. Then, select a 6 or 8 week project where you can rapidly make these experiments to see if you are onto something.

4. What are the tough questions that teams should ask themselves when thinking of going DevOps?

Volker: This goes back to the previous question. Did you select the proper project? Do you have management buy-in? Do you have the right process in place? What are the right processes for you, and for your team? What's your current state of DevOps? If you have started DevOps, or you want to go into. How does your software lifecycle look today? Do you have pockets of automation in place? Are you working across the teams? Do you talk to your Ops people. Do you Ops people have influence in how applications are instrumented? These are things that are important. Even, "which are the proper products?" is another later question that you want to ask yourself. It all goes back to whether you chose the right project and do you have proper buy-in. So there's tons of questions, and they are almost all equally important. And there are some guidance from Docker, Microsoft and Chef as to how to approach this, and how to go about exploring what you need to know and what you need to "skill" your teams on when you embark on the journey to DevOps.

Jessica: Do you know how to measure things in your organization? What I mean by that is, do you have good data on what your systems are like now? How long does it take you to stand up a production environment? Is it days? Is it weeks? If you can gather data right now on where you are, it will help you because when you finish an iteration, now you can go and see, "did my metrics change the way I thought that they would?"

Matt: Jessica made a great point about being able to measure success. If you don't know where you are starting from then what exactly is "success?" One of the big things people don't even consider is, do you have time to make change? Sometimes changes can take a long time. Sometimes you may not be successful in implementing the changes that you are making. And do you have buy-in from everyone? If folks don't have time then you may have to push things back. Companies don't always ask this questions right out of the box.

5. Can you tell us a DevOps success story and why/how it worked for them?

Jessica: We worked with an organization and our first meeting noticed everyone separated as far as possible from one another on different sides of the table. The interesting thing is that after a few days, they were pair programming - really working together on their use case. I think this was positively impacted by effective facilitation of the learning space. These folks, who all invested significant time in this, came together and were willing to learn something new. We saw a really different level of energy in the group.

Matt: One thing that stood out to me was when I was working with a small company that creates educational software, where a lot of tooling had been put in place to enable them to work remote or local. They designed a lot of their communication process to be remote. They were working on taking a very large monolithic application, and splitting it to different pieces. They started by improving their communication. They had daily standups and they made sure to maintain that open communication between groups. They created processes that made it easy to roll back changes when necessary. They got comfortable breaking things, and in fact one of the best things that they did was accept that fact that they were going to tear down and stand up their environment on a daily basis. So they knew everything worked, and they knew that they could recover in case of a catastrophic failure. It was great to see.

Volker: One thing that we do in our (Microsoft) evangelism work is we run a number of events where we get developers and operations people get together for a few days and start explaining to them what they might gain by working together and how they can apply certain DevOps practices to their work. What has been the biggest success is that in many cases, for the first time, the developers start understanding their Operations people. And finally they start working together, finding out that there are better results in many cases, and to Matt's point, they suddenly find out that if they do things properly they can set up and tear down their new environments in not time. Just by talking to people that have different expertise, and use the expertise for the greater good of a project.

So, what often happens is they then ask us. How we do things at Microsoft. We are a large company. So we have many teams, and as expected not all teams are at the same level of maturity. We have put together an eBook where we outline how we transition teams within our company, through learning and then apply what DevOps means. And through applying DevOps practices.

6. When thinking about large technology shifts like on-prem to cloud, how can DevOps support those kinds of dramatic transitions?

Matt: Transitions are not always easy when you are used to traditional infrastructures. They can be scary to start out with. The streamlining of processes that DevOps allows you to accomplish helps with the ability to migrate to the cloud, or stand up new environments. Over time it becomes easier, then suddenly teams are doing it on a regular basis, and the transition may not actually be as different as you expected it to be.

Volker: I think that the promise of DevOps, becoming faster, more efficient and more effective in your software development cycle is very much supported by a flexible and agile cloud environment. The tooling that you see that is available from many partners and Microsoft, are founded in and solidly grounded in open source. We support the DevOps practices in order to enable dev and ops to find new and better ways to be better and more efficient in processes. For the better of the business. What I also think that the cloud offers in particular, or almost demands, is that developers and everyone else involved in development process, and the deployment process and feedback loops, need to talk. Because by enabling your applications and taking advantage of the agility of the cloud, you need to react faster to customer requests and demands in order to stay relevant in the market.

Jessica: People want to move data and compute wherever possible, from on-premises to cloud or re-architecting for PaaS. I think it comes down to repeatability. Being declarative about the process and making it repeatable. For me, this means using infrastructure in a way where you almost don't care where it is. If you walk to work and there is yellow tape around your building, you need to be able to redeploy from source code, and you should be able to do that with tremendous flexibility.

7. Each of you works at a technology vendor, how are your tools helping the DevOps transition?

Volker: With our cloud platform, with IaaS and PaaS we are aiming at proving an infra-as-a platform for our customers to build their software, the software they need for their business, and opening it up to the open source world has shown tremendous results for us, for our company, but also for our customers. The partnership engagement that we are driving, and evangelism that we are driving with Docker, Chef and many other partners show that Microsoft Azure, as a platform is a great platform for tooling, as well as business applications as a whole. In the tooling world, we enable developers to build their applications for a mobile first, cloud first world. And use cloud tech and agility to enable a DevOps transition as fast and efficiently as possible. the open source, and opening up the Microsoft ecosystem is important for us, and we see from the interactions we have with our customers, that it is very well accepted and demanded from them.

Jessica: Chef has built some amazing technology. We give you a framework, an automation platform so that you can describe your infrastructure in a way that can be version controlled, meet compliance needs and be able to provision that in the cloud or on premises. Chef streamlines the configuration and maintenance of a company's infrastructure and application code and our customers are using Chef to become fast, efficient, software driven organizations.

Matt: Docker has come in really strong with the developer experience. Often times we see organizations and groups just starting with a single developer acting as a champion. The developer started playing with Docker and would like others in the organization to start using it because people see benefits. It allows developers to no longer have to worry about some of the common frustrations of code working on one machine, but not working in test environments up on the CI system. We are giving a standard container based infrastructure that developers can begin from. Doing everything from enabling better CI workflows that brings back a point that Jessica mentioned earlier. The ability to have repeatable builds docker helps a lot with that. Building a CI system using Docker to perform builds in disposable containers so that you are always able to reproduce artifacts if need be. And not only that, but when you are building it locally, and you build it in your CI system, developers will have the exact same results from the same image that they are building from. Docker, creating a standard unit of work helps simplify this process. So getting from a developer out to production all you have to know is how to create a container and how to deploy it out to an environment, it certainly enables the operations team to be able to deploy containers. If you do it once, you have the ability to do it anywhere. It enables developers to build containers for your operations team, and with your operations team in mind,make it easier to reach production deployment. Being able to build, ship and run any applications, anywhere really helps the journey from on prem to cloud, and that's a common thing that I am seeing today. People see Docker as a way to transition to cloud-based environments.

Learn More

- Watch the webinar recording: docker.com/webinars
- Start your DevOps journey with Docker today: https://hub.docker.com/enterprise/trial/

www.docker.com