How can I accelerate my DevOps initiative?

Everyone has a DevOps initiative – everyone. No matter what size our client or prospective customer the term “DevOps” comes into the conversation whenever we ask someone to describe current initiatives and challenges. Most are trying transform an existing organization to take advantage of DevOps practices such as continuous delivery and deployment automation.

The challenge is most acutely felt at the largest organizations. Organizations where DevOps is usually seen as a chance to accelerate the delivery of software from monthly or quarterly releases to weekly or daily releases. These large organizations usually have existing development groups and large departments dedicated to supporting applications that have existed for a decade or more, and the path to DevOps usually involves retraining existing staff or selecting a small group of projects for a proof of concept.

As time to market and release frequency are quickly becoming essential to keeping up with the competition. IT management is under pressure to accelerate the acceleration. Some companies have made the transition to DevOps successfully, but many more are wondering how they can play “catch up” quickly.

These are some general tips and tricks from our experience helping guide the largest organizations to a successful transition to DevOps.

1. Make gradual improvements.

DevOps isn’t an all-or-nothing proposition. You might miss this nuance if you talk to people selling you DevOps services or products. Some DevOps proponents can be very impatient with existing processes and will often suggest throwing out everything and starting from scratch because they have been trained to think about software from the perspective of a small startup. When every project is new and you don’t

have to worry about billion-dollar risks to an existing business you can start a revolution.

When you are responsible for an entire enterprise there's a good chance you can't just wake up one day and tell a few thousand developers to "DevOps." You need to create a multi-year plan to move parts of the organization over to DevOps practices, and you should target gradual improvements. If you are just starting your DevOps initiative, take time to analyze your projects and create a multi-year phased approach with metrics and milestones.

2. Target active projects first.

Don't tell every project to start planning for developer-driven production deployments in the next month. Some of your development teams are supporting legacy systems that rarely, if ever, change. It makes no sense to spin up a DevOps initiative for a project that isn't under active development.

If you use a tool like Plutora across all of your projects you'll be able to identify the most frequently released projects. Target these projects first because they will see the most benefit from process improvement.

3. Aim directly at existing inefficiency.

Using Plutora you can analyze your existing release and deployment processes and identify the largest opportunities for process improvement. Using Plutora's comprehensive view of release schedules and environment dependencies you can identify the projects that are consistently causing environment conflicts and release delays.

Some projects are so central to the organization that entire departments end up having to wait for release events to make any progress. It is these "mega-projects" that tend to introduce inefficiency into the organization. These are the behemoth projects that need to be released more frequently or that need to be split up into smaller, more independent projects that can benefit from DevOps practices such as continuous delivery and deployment automation. Start with the toughest projects.

4. Don’t hire DevOps, invest in your people

Hiring top DevOps talent might be unrealistic as DevOps-related skills are some of the most sought after skills in the market. Instead of trying to hire your way to agility you should consider investing in the people you have. Focus on desired skills and create a plan to train your existing developers. Find a team of champions who “get it,” and ask them to schedule regular sessions to encourage existing teams to understand what it means for developers to take more responsibility for operations. It will be more cost effective to send your high potential staff to agile and DevOps training, and you’ll have the added benefit of training people who are already familiar with the projects and systems they will be supporting. If you hire your way into DevOps it’ll take months for you to acclimate new talent to the way your organization works.

5. Adopt Popular DevOps Tools

In a crowded market of DevOps tools and vendors you’ll be getting advice from just about everyone on what DevOps tools to adopt. Should you use Chef or Puppet? Should you use Atlassian tools for builds or Jenkins? What about build tools and logging frameworks? You’ll be faced with a thousand big decisions and a thousand little decisions, and every decision will be attached to strong opinions from your technical staff.

Do yourself a favor and adopt popular DevOps automation tools. If you are looking for a continuous deployment tool don’t just choose the option your existing vendor offers because it is already included in your support contract. Use tools that have a demonstrated track record of success in the market and look toward open source as a source of innovation. Tools like Git and Jenkins come with millions of users ready to answer questions online and more resources in the form of books than you’ll get with any unpopular and highly proprietary solution. Beware of DevOps snake oil salesmen.

6. Understand the Goal: Define DevOps

The definition of DevOps is very subjective with some people defining it as developer-driven operations and others defining it as “more cooperation between developers and system administrators.” These definitions are nebulous, and often when you ask for clarification you’ll be asked to read one of several popular “business novels.” There’s no right or wrong definition of DevOps because, in reality, DevOps is defined differently by every organization adopting a set of a la carte best practices.

Bring your developers and managers together and explore what people mean when they say “DevOps.” Identify the goals of each of the participants and also make sure to reach out across departmental boundaries. Make sure that your system administrators and operations professionals have a shared definition of DevOps with your developers.

7. Define success. Establish KPIs and measure them.

How do you know you are moving in the right direction without a good set of key performance indicators? Make sure your teams understand what success looks like and measure KPIs such as a release cadence, software quality, time required for deployments, and other key metrics related to your release process. DevOps will increase agility overall, but it will affect your release management process more than anything else.

Use Plutora as a platform to define your KPIs and measure them across all projects even those that you are not migrating to DevOps practices. Plutora will give you the raw data for release time and time dedicated to release management. You can use Plutora’s release-focused analytics to support decisions about where your DevOps initiative dollars are best spent.