DevOps for Government

How DevOps Applies to Government Programs

December 2012

By Erik Levy, Chief Executive Officer, Chief Architect, Elluminates Software

Executive Summary

Commercial and Government organizations have adopted agile development practices to increase the ability to respond to market pressures and bring features into code bases quickly. However, operational practices have mainly focused on reliability and risk reduction and have not adapted to the increased speed of the development process. DevOps aims to bridge the gaps that have been occurring between the Development and Operations environments by looking at them as a single unified process instead of separate silos. This paper discusses how to apply this commercial approach to Government programs.

About Elluminates Software

We bring the best of commercial technology and techniques to infrastructure engineering. And because IT infrastructure needs to be built to last but also adapt, we look significantly ahead of today's needs so that our solutions can last. Which means that by combining the best of what is happening in the commercial space with the lifespan and mission of our customers' efforts, we can create unique scalable solutions that are a blend of both.

DevOps Responds To Business Challenges

One of the current trends in large-scale commercial organizations is the collaboration and integration of Development and Operations. Organizations such as Netflix, Google, and others have spoken publicly about how their processes have been influenced not only from the adoption of agile development practices but also the move to cloud based operations that has increased deployment provisioning and automation. The coined term, DevOps, signifies that combination. This trend is new enough that many Government organizations have yet to adopt it, and we believe it has a place for consideration in Government delivery.

What advantages do we see with DevOps from current operational models in the Government managed service and private/public cloud space?

The following challenges, covered from current literature on commercial DevOps[1], exist for current Government operational models as much as they do for commercial organizations:

Low communication between Development and Operations – Because Development and Operations are typically run by different parts of an organization or team, communication tends to go through separate chains of command. When problems arise, it can be difficult to coordinate between the two groups since those deploying production releases may rarely interact with those developing the software and systems.

Low coupling of motivation between Development and Operations – Development and Operations teams have different motivations. Development is paid to make changes that create the future capabilities for a program. The Operations team is there to ensure stability for the program so that the business mission can happen that day. Each team is needed for the program to be successful, but their competing motivations can make working together challenging.

Different tools used between Development and Operations – Another challenge is that Development and Operations teams use different tools in their day-to-day jobs. These tools may include integrated development environments (IDEs), source code control, configuration management, ticketing systems and deployment scripts for development, and staging and production environments. While some tools may be usable by both teams, such as configuration management, many of them are used uniquely by one group or the other and in different infrastructures with different IP addresses and system and network configurations. These disconnects produce deployment problems repeatedly when a hand off from Development to Operations occur.

Different processes used between Development and Operations – Development teams generally follow a software development or systems lifecycle that takes them from initial concepts through design and implementation to testing. Sometimes it's repeated in an iterative fashion, and sometimes it's a long cycle that ends with a production release in just one go. Operations teams tend to focus on ensuring that the existing deployments don't fail to serve the business mission for users and treat new deployments carefully so that they can fall back to previously known versions if something goes wrong. The lifecycle processes for Operations are focused on minimizing risks to user delivery, where the lifecycle for Development is focused on providing improvements through change for users and the business.

Common need to have successful deployments but different environments – Development and Operations teams end up working with the same specifications, components, and systems in one form or another but rarely are they one-to-one environmental matches. As such, when a release hands off from Development teams to Operations, the fact that Development might have had all local development or a degraded staging environment throws the production level deployment by Operations into chaos when something goes wrong. Deployment scripts created and used by Development teams don't work in the production environment because variables in the state of the environment don't match (e.g., IP addresses, naming conventions, database locations, etc).

But despite these common Government challenges, the success of a Government initiative depends on both Development, that is focused on making change happen, and Operations, which is focused on stability, to work in unison.

DevOps Approach for Government

The premise of DevOps is that aligned interests produce natural business outcomes that match the business mission through shared business objectives. Instead of seeing Development and Operations as separate activities that have their own lines of management, ownership and objectives, DevOps states Development and Operations are part of one unified business activity, regardless of organizational structure or business lines.

One of the key aspects of DevOps for Government is to allow for programs to respond to environmental and citizen-driven changes in a rapid and agile fashion. Another is to make those changes in a cost-effective and risk-managed manner.

For Development teams that are familiar with agile development practices, where stakeholders, shareholders and Development produce rapid changes through backlogs driven by business needs, DevOps will seem familiar. However, we have also seen that having half of the unified team moving quickly using agile methodologies while Operations moves at a different and risk-averse manner ultimately causes bottlenecks for actual users and the program at large.

To break the increasing pace of Development, but waterfall pace for Operations dilemma, DevOps aims to have change occur at the program level where Operations can be kept in synchronization with the pace of change coming out of the development process.

The major items cited for DevOps for commercial organizations, but which are updated for Government programs, include:

1. **Staff Measurement –** If the Development and Operations teams report to different managers or program lines, which in large programs is common, each group's performance can only be measured separately. By instituting an end-to-end development-to-operations measurement process to determine success or failure, performance can be measured with all parties as an integrated unit. One of the cited techniques, adjusted for Government, is to take a top-down approach and institute Key Performance Indicators (KPIs) that are baselined, tracked and understandable to all stakeholders who make decisions for the program against the business objectives.

2. **Staff Incentives –** Once the measurements (e.g., KPIs) are understood and agreed upon, the program should start to measure end-to-end and focus on how the Development and Operations teams play a part of those measurements so that success for the program is a unified process. Incentives can then become part of the review process of the individual contributors, which should be an input to the organization's compensation processes. In addition, incentives for a program, such as promotions and leadership roles, can be adjusted to now include those that help the Government from end-to-end versus just the individual stars in their silos.

3. **Program Level Unified Processes and Tools –** The key for a successful DevOps approach is to have a unified end-to-end view of the development-to-operations lifecycle. That can mean using a unified process like agile in both Development and Operations or each group having their own process that plugs together in a way that allows the two groups to work in concert versus just interfacing when one group needs something from the other. The approach can be different for small teams versus large teams as well as if the teams are

geographically dispersed versus localized. The key is that the overall development-to-operations cycle is one concept.

In addition, by choosing the right tools, the program can increase the unification of the DevOps lifecycle process. For example, by unifying the source-control systems so all artifacts, components, and versions are captured across the program; confusion on the release packages can be reduced. Having architects model the system and capture the policies, manifests, and versions communicated across the program is another methodology to reduce disconnects between groups.

4. **Program Level Development-Build-Integrate-Deployment Tooling –** While there has been a push for some years for automated build tools to handle regression level issues in testing, it also allows for the removal of manual dependency management (which in complex systems can be hugely complicated where different program products can have different version dependencies on the same base software in one axis and in another axis having different environmental configurations (e.g., dev, stage, production). In addition, automation can accomplish system building, provisioning (e.g., Chef), updating and in case of failures, rollback to previously known good versions of the product. By moving the Development-Build-Integrate-Deployment process across teams, you remove chances for manual errors and provide a basis to have across team visibility. As an extended example, for Netflix[2], their NoOps approach, which is currently considered an advanced end state of DevOps, allows Development teams to deploy directly to production and bring in a smaller Operations team as part of the Development team to increase operational automation of the production environment.

Summary

DevOps, in the context of Government programs, offers the ability to increase the overall responsiveness of programs to changing Government needs, as well as the ability to scale programs gracefully to handle increasing system complexity. Unlike agile development methodologies used in Government software efforts that focus primarily on increasing the speed of change, DevOps is about integrating Development and Operations into a unified delivery lifecycle that cuts across a program to meet its mission at scale.

[1] Literature includes http://dev2ops.org/2010/02/what-is-devops/
[2] Literature includes http://perfcap.blogspot.com/2012/03/ops-devops-and-noops-at-netflix.html