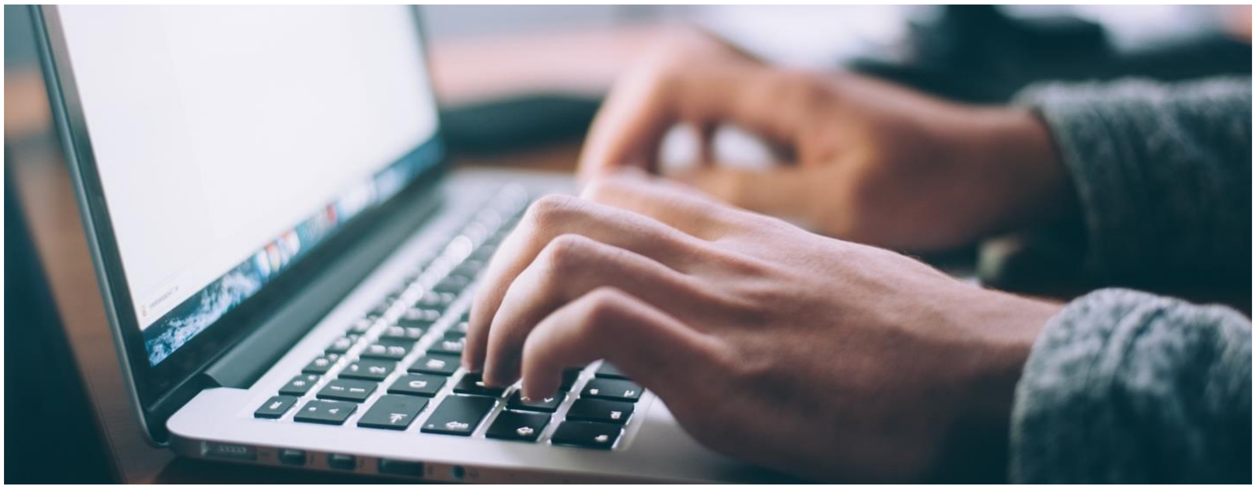
DevOps Drives Efficiency Across the Maturity Spectrum

Steve Curtis

devis®



1 Executive Summary

All enterprises are at some level of DevOps maturity. A highly mature organization is able to deploy applications more quickly, in one or more locations, while maintaining high output across activities and supporting repeatable tasks. Less mature organizations spend time executing a highly manual, error-prone process, resulting in costly rework or Service Level Agreement failure. Wherever an organization resides along this spectrum, Devis and DevOps are poised to improve their performance.

The more mature an organization is, the more likely its DevOps process incorporates automation. The benefits of automated DevOps include increased efficiency and accuracy, which translate into increased cost savings. This is in comparison to an immature organization following inconsistent and unrepeatable artisanal processes, producing inconsistent and non-maintainable applications. Automated DevOps also has the additional benefit of providing a completely auditable path from inception to production.

DevOps is as much about culture and practice as it is about tools. By applying the practices and technologies that comprise DevOps incrementally, an organization incrementally reduces risk and decreases time-to-deliver. Organizations are also able to deliver solutions at a higher velocity. ***In short, by embracing DevOps, less mature organizations can realize significant benefits, including cost savings, improved quality, and shorter delivery cycles.***

Devis helps organizations achieve benefits through easy-to-manage, incremental changes across the full spectrum, bringing organizations to higher levels of maturity at a completely customizable pace. We offer a dual-pronged approach in which we help an organization create a corporate DevOps model, but also provide development horsepower to implement and execute the model.

2 What is DevOps?

First and foremost, DevOps is a holistic approach to applications development that integrates development and operations. Its purpose is to increase an organization's velocity in evolving and enhancing products for users, while improving quality, reducing cost, and maximizing repeatability.

DevOps is a combination of software development and information technology (IT) operations and is comprised of multiple disciplines, including:

- Planning
- Agile development
- Automated testing
- Continuous integration (CI)
- Configuration management
- Continuous deployment (CD)
- Automated monitoring

Each one of these disciplines can be executed individually, but combining disciplines and their supporting staff is the goal of a DevOps process. Disciplines are important categorizations and each one has one or more tools to support its specific tasks and goals. The combination of these tools is called a *toolchain*. The key to integrating the individual disciplines and toolchains is a workflow tool that automates activities and reports to stakeholders in a timely and meaningful manner.

Figure 1 illustrates the elements of and interactions in a DevOps environment. Note that personnel are focused in creativity-focused areas and the rest of the process flow falls to intelligent, purpose-built tools.

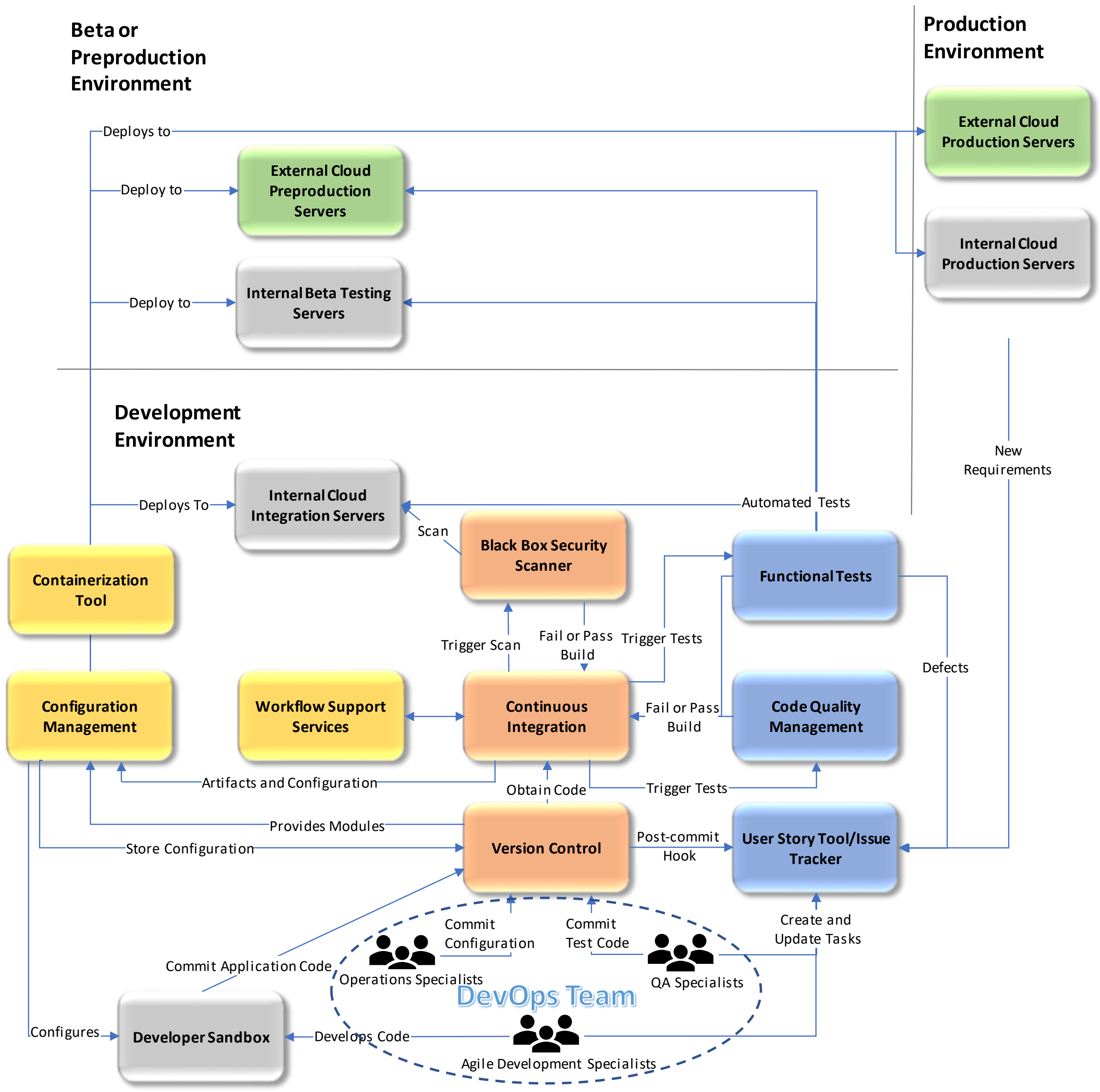


Figure 1 A Sample DevOps Tool Chain. Integrating the tools for each discipline maximizes the opportunity for automation and cost efficiency.

3 Integrating DevOps into an Organization

Once an organization makes the decision to implement a DevOps process and receives management buy-in, the organization must analyze its structure and the software development and deployment life cycle; look for opportunities for collaboration, improvements, and efficiencies in process; and identify opportunities and tools for automation. Some initial analysis that must be undertaken includes:

- **Organizational structure and culture**—Are teams siloed? Is there any concept of joint ownership? Are there competing priorities? Are there teaming opportunities? Is there open communication? Should there be reorganization?
- **What work is performed and why**—What is the full lifecycle of any application or component? Is the current process a response to industry or regulatory requirements, or is it "how we have always done it"?
- **How is the work completed**—What processes and tools are used to accomplish each task?

DevOps requires an organization to establish shared understanding and ownership of the systems modernization process amongst its members. Coming to a common understanding requires involving staff from across the lifecycle: requirements stakeholders; designers; developers; testers; documentation; quality assurance; security and operations, production, and user support staff.

Knowing how and why work is completed is important because the organization will need to adapt or change these processes as they implement a DevOps methodology. For example, there will be tools put in place to do work that was previously performed by hand. One effect of this shift is that staff tasks will morph. Mundane tasks, such as metric collection, will be performed by tools, allowing staff to give more attention to the important, creative work, such as design, testing, and analysis of those metrics.

An organization can move as quickly or deliberately as it finds comfortable, implementing processes and tools across the full lifecycle simultaneously or focusing efforts on individual disciplines, one or two at a time. The organization will realize discernable, quantifiable improvements in their performance with either approach.

4 A Closer Look

The goal is to deliver high quality features to users in a timely, consistent, and sustainable manner. Increasingly mature DevOps environments seek to accomplish this goal with as little human intervention as possible to maximize consistency and quality while reducing cost and time-to-deliver.

4.1 Organizational Structure and Culture

In siloed organizations, system ownership passes from one team to the next. Each team "throws the code over the transom" as the system advances through the lifecycle. In this model, issues result in finger pointing and little system or process improvement is achieved beyond fixing the immediate problem. DevOps strives to combine development and operations staff into a single team (or even a single organization), one that takes full ownership of systems throughout the lifecycle.

A team approach that integrates Operations staff early provides the organization with a better understanding of the features that will require support as a system is deployed. Development staff benefit from knowing Operations' capabilities and limitations. Pairing developers and operators also builds trust and understanding across disciplines. As engineers work through the application lifecycle, they expand their range of skills and can support the system from development to test to operations.

The most mature teams also integrate security experts to achieve DevSecOps. Identifying and addressing security risks early in the process greatly improves system quality and reliability.

Creating joint discipline teams is a simple, inexpensive step less mature DevOps organizations can implement that pays big dividends in system quality, timeliness, and cost.

4.2 What Work is Performed and Why?

Typically, organizations lack a compressive view of the activities needed to support a system throughout the delivery pipeline. Most have Standard Operating Porcedures (or SOPs) or “tribal” knowledge of the steps and procedures needed to deploy a system in a respective environment. But many lack a full view of the chain of events and activities required to plan, code, build, test, release, deploy, operate, and monitor systems.

By exploring, documenting, and diagramming workflows and by understanding the full lifecycle of any application or component, an organization can begin to identify bottlenecks and/or redundant processes. As a result, individual processes can be steamlined, removed, confirmed as best practice, or verified to meet regulatory requirements. Organizations can also improve the interplay between processes and optimize their workflows. Finally, organizations can identify performance metrics by manually mapping and tracking the delivery pipeline. These metrics allow an organization to measure the effects of implemented changes to ensure continuous improvement. Each of these activities can provide value to organizations, regardless of DevOps maturity, but are a prerequisite for less mature organizations before implementing automation and creating toolchains.

4.3 How is the Work Completed?

Once an organization understands what processes are performed, it can examine how processes are performed by asking questions of themselves, such as:

- What tools are used to accomplish each task?
- Can these tools or processes be automated?
- Should they be replaced?
- Can they be standardized?

Through automation and standardization, teams can increase velocity and issue new releases more frequently and with fewer problems. Automated test and review cycles prevent release delays as incident response times improve. Organizations that are less mature on the DevOps spectrum benefit from incremental automation and adoption of integration toolchains.

5 Continuous Delivery Example

To ensure that code or features are always in a production-ready state, convert defined processes into actionable software and validate the resultant code on a continuous basis (as depicted in **Figure 2**).

A critical success driver in this process is user feedback that is integrated into the loop as new requirements are identified. In this way, key stakeholders' voices are heard and accounted for in each delivery cycle, improving communication, collaboration, and ultimately, the end result.

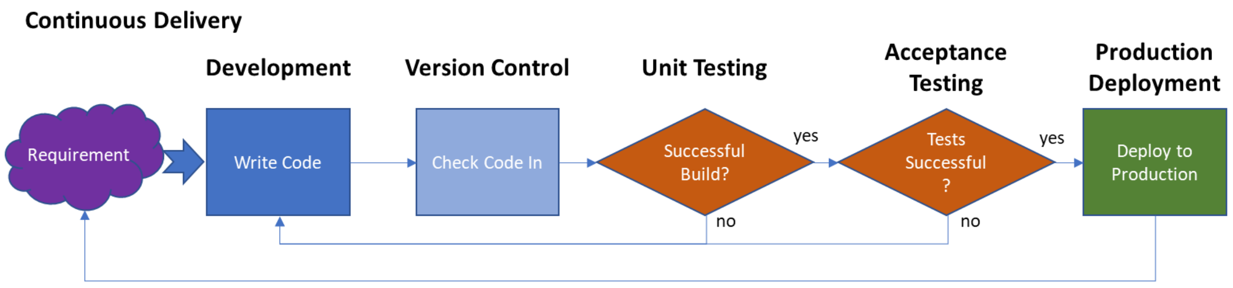


Figure 2 The Continuous Delivery Process Loop. Short, quick cycles bring incremental feature enhancements to production smoothly

As mentioned earlier, another strength of DevOps is its well-defined and flexible workflow. All enterprises have different structures and environments in which they deploy code or features. One enterprise may operate completely on premises ("on-prem"), another may access multiple clouds on-demand, and a third may integrate on-prem and cloud resources. While the architecture may be different, each environment requires certain tasks, tests, and approvals along the journey from a feature request to a functioning feature in the production system. One key step to orchestrating matriculation is the use of CD/CI pipelines or delivery pipelines.

The delivery pipeline in the DevOps toolset integrates code, initiates builds, executes tests, and deploys or packages code for deployment. A mature automated delivery pipeline identifies tasks in a traceable, reportable, and auditable manner. It also reports how long each individual step and the aggregate of all steps takes to complete. The delivery pipeline has the ability to run tasks in serial, parallel, or mixed fashion and the completion of one task can initiate or stop another. Delivery pipeline tools are normally API-driven and can accommodate new tests, scans, or approval gates as required.

Figure 3 depicts a simple delivery pipeline that accomplishes several tasks in serial and parallel fashion. This particular workflow:

- Reviews code from a versioning repository;
- Runs tests and code analysis;
- Builds the deployment package; and
- Deploys the deployment package to a User Acceptance Test environment for functional testing.

Each action produces a log that describes facts about the task such as build number, elapsed time, and status. The tools that were used within this workflow are:

- Version repository
- Configuration Management tool
- Continuous Integration tool

There are a number of well-constructed commercial and open source tools that can fill these needs and each organization should feel free use the tools that are most appropriate for them. Devis takes a "vendor-agnostic" approach to tool selection, preferring to focus on each client's specific environment, system requirements and configuration, and DevOps maturity level to recommend tools to address client specific needs most effectively.

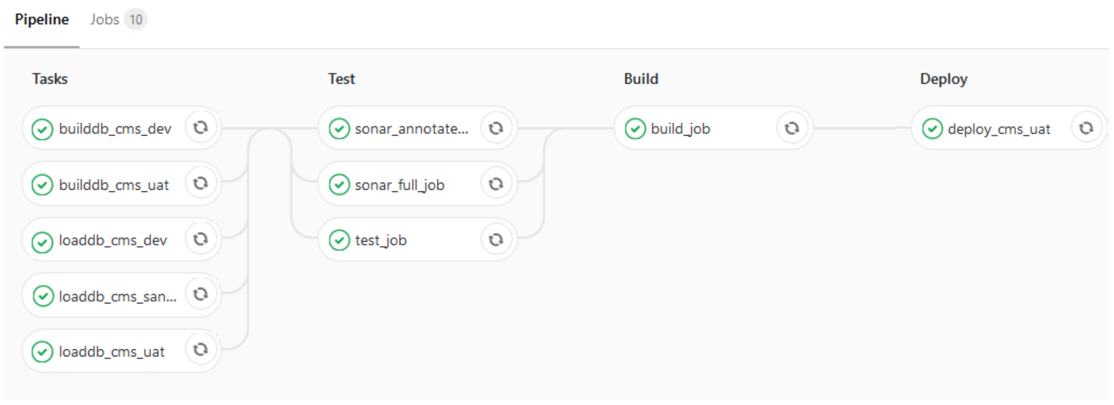


Figure 3 Sample delivery pipeline. Providing automation and a simple to follow audit trail for each step in a process.

Documenting the delivery pipeline in real time is an important element of a DevOps methodology. Most automation tools created to support DevOps include the ability to produce artifacts that document the process as a byproduct of the work. For example, log files that are created throughout the process are sent to the versioning repository and stored as artifacts of any build. These artifacts are valuable to auditing entities in order to support an assertion of the status of an environment at any point in time. An organization also can use these artifacts to quickly recreate an environment if necessary.

6 Conclusion

There are many benefits to a properly implemented DevOps environment. Each organization may have different needs, but the basic DevOps framework is flexible enough to support each. It is important to remember that while each discipline can be siloed, joining disciplines can increase the capability, flexibility, accuracy, precision, and efficiency with which features, enhancements, or entire applications go from inception to production. In return, a DevOps methodology will manifest itself as cost savings, higher quality products, and greater user satisfaction. Devis smooths the implementation process and supports organizations across the maturity spectrum with our team of developers and mentors.

About Steve Curtis

Steve Curtis is Devis' Vice President of Development Services. He is an industry veteran, with over 20 years of experience in the information technology field. He specializes in the design, implementation, and management of data-centric solutions – combining a background in database design and object-oriented programming with the industry-leading standards of the Project Management Institute.

Mr. Curtis is a Certified Project Management Professional (PMP), a certified ScrumMaster (CSM), an Agile Certified Practitioner (ACP), a Certified Scrum Product Owner (CSPO), and has completed graduate work in Software Engineering at Virginia Polytechnic Institute.

About Devis

Devis is a minority, woman-owned small business (WOSB) with more than 25 years as a leading provider of IT solutions to the Federal Government and international development community. We have built our practice by solving the information sharing problems faced by public and private organizations with dispersed stakeholders. Our core areas of expertise include:

- Agile Application Development
- Tier 2 & 3 Help Desk Support
- Secure, Cloud-Based Managed Services
- Worldwide IT Deployment
- Systems Integration
- Knowledge Management
- IT Consulting
- Section 508 Accessibility
- Software and Business Process Training

Our accomplished staff has an average of more than 15 years of experience in the IT field, and more than six years of experience at Devis. They have collectively travelled on hundreds of TDYs to over 70 counties. Our proven success is a direct result of our certified staff and our ability to develop, deploy, and maintain systems while understanding and supporting our clients' goals.

For More Information

Chris Kagy

VP Business Development

ckagy@devis.com

703-525-6485 ext 126