DEVOPS ADVICE FOR SMALL, MEDIUM, & LARGE ORGANIZATIONS



DevOps can take many forms. In fact, every organization adopting DevOps can have their own definition, policies and implementation strategy for the SDLC methodology. Small, medium size and large organizations may adopt various DevOps principles based on applicability, availability of resources and business goals. However, some general DevOps best practices can be adopted consistently from a high level perspective across organizations of all sizes and verticals. Here are a few DevOps tips and advice to help your organization fine-tune your DevOps approach with the industry-standard best practices:

(This article is part of our DevOps Guide. Use the right-hand menu to navigate.)

Go for Fast and Short Release Cycles

DevOps transformation is all about speed by adopting the technology, cultural and operational changes to deliver software builds with faster release cycles. In this context, organizations of all sizes must adopt Lean principles to introduce agility across all stages of the SDLC pipeline and thereby accelerate innovation. For instance, establish policies and governance structure that doesn't interfere with cross-functional and inter-department collaboration. From an SDLC perspective, its easier to establish agility when the release cycles are fast and short, with fewer controls. Faster release cycles ensure that new features are tested and validated quickly. Developers don't just produce a software build and push the code into test environment. Instead, Devs work with testing teams to ensure that iterative, functional builds are released through to production environment. Furthermore, if a feature release does break the system, the modular approach to software release

on an iterative basis makes it easier to identify the root cause of the problem. As a result, DevOps organizations spend less time identifying issues and fixing them only too late into the SDLC cycle, and instead are able to deliver working software code at a higher frequency.

Establish Continuous Improvement

DevOps thrives as it approaches toward continuous improvement in every stage of the SDLC pipeline. This improvement results from continuity across the DevOps process flow, that allows DevOps teams to reduce waste, focus on improvements and ultimately deliver better quality software at a faster pace. Practices such as Continuous Integration, Delivery, Deployment, Monitoring and Feedback are the key components of this strategy. Continuous Integration involves regular update and merging of software changes to a centralized shared repository. Each update can be automatically tested, and an up-to-date functional version of the software build is always available for release. These verified changes can then be packaged pushed into production at the push of a button (Continuous Delivery) or automated entirely (Continuous Deployment). Finally, a culture of continuous improvement is ensured as each release is carefully monitored during and post-release. The resulting changes in terms of software performance, quality, security, customer feedback and end-user experience is then evaluated and continuously considered for future builds as part of the Continuous Feedback cycle. A key consideration for continuity of these processes is to eliminate waste and bottlenecks, both technical and cultural, that may impact the pace of the DevOps process flows.

Balancing Automation in DevOps

Continuous process flows within the SDLC pipeline are bottlenecked by tasks that require manual intervention. These tasks may include configuring and provisioning the infrastructure, orchestration of the DevOps pipeline, code testing, collecting data and metrics, and communicating updates, among other tasks. By automating these tasks, DevOps teams are able to foster speed, consistency and reliability within their process workflows. However, DevOps isn't all about automation and in fact, too much automation can adversely affect the progress of the software development project. Automation is intended for processes that are repeatable and no manual changes are necessary across the repeated process iterations. However, if a process is flawed and requires manual changes to eliminate the issue, automation will only repeat the flawed process and yield suboptimal results. It's important to break down the meta-development workflows and identify the phases that can be automated for the unique application of your particular organization. Then, select tools that can address those unique requirements and integrate with the legacy technologies that may be in use. Finally, it's important to understand that the focus of automation should be on the SDLC processes, and not the tooling itself. In addition to the tooling and processes, the automation strategy should also encompass the organizational structure and the approach of DevOps team members to contribute toward automated processes. New set or responsibilities as part of CI/CD may be required and organizations may need to streamline communication and collaboration channels among members from disparate teams.

Culture Change: Shared Motivation and Failing Fast

DevOps is inherently a cultural change regarding the way organizations develop software products. The change relates with the way individuals and teams collaborate and approach the SDLC process

workflows. The most important element of the cultural change is to establish a mechanism of trust among team members. Along with fostering a mindset of trustworthiness, organizations must eliminate the silos across team members and organizational departments to ensure unified agreements and communication across all phases of the SDLC. Create a sense of shared responsibility between DevOps team members and eliminate the philosophy of “it’s not my job” at the workplace. A mature DevOps culture is free from the blame-game and embraces failure. In fact, failure is often celebrated in fast moving innovative companies such as Facebook that follow the principle of “fail fast and early”. With this approach, DevOps members are encouraged to try new ideas and have a bias toward action. When a process fails early, its easier and less expensive to restart from the beginning as opposed to identifying flaws only at the end of the SDLC when it’s too late, complicated and expensive to fix the issues.

These best practices are applicable for SMBs and large enterprises alike. The advantage for SMBs is the ability to embrace DevOps and move away from traditional SDLC practices with little friction. Large enterprises on the other hand may have a range of governance controls, policies and strong culture of SDLC practices that go against the modern DevOps philosophy. Additionally, for large enterprises coming from traditional industry verticals and gradually shifting toward a technology-centric business segment, integration with legacy technologies, complex IT infrastructure and the lack of right skillset may bottleneck DevOps adoption.