DEVOPS ADOPTION - WHY SHOULD YOUR CFO CARE?

Abishek Bhat

Introduction

IMPACT OF AGILE SOFTWARE DELIVERY ON REVENUE NUMBERS

A recent survey commissioned by a leading technology provider reveals an average of 19 percent revenue increase directly attributed to the adoption of DevOps methodologies. A drill down to the most common financial attributes of DevOps adoption provides more clarity. Automating the software delivery benefits the organization financially in multiple value areas:

- Revenue gains from enhanced developers productivity and reduction of IT headcount waste
- Revenue gains from accelerated time to market of new functionality
- Gains from cost reduction of application failures resulting from increased quality
- Gains from flexibility in the IT environment

To understand how DevOps benefits an organization financially requires us to delve a little deeper into the subject.

As we all know, applications are a key driver to the growth and success of today's businesses. Visibility into application performance and the efficiency of the software development lifecycle reaches far beyond IT and well into business stakeholders. Applications are complex and fast evolving, and with the expansion of mobile and web consumption demanding flexibility to handle constant requirement changes effectively, businesses are forced to reassess their application delivery strategies. Coping with the need for change requires more than a strategy! It calls for an ever-expanding budget without clarity on the financial benefits of adopting agile innovations.

New software development methodologies have emerged to address the need for agility starting from development practices to full automation of the software release process. This collection of best practices has matured into the continuous delivery (CD) process, which applies industrialization concepts to software. Designed to streamline and accelerate software delivery while ensuring that reliable software is released, CD creates an alignment between the application and changing business needs.

The principles of CD find their roots in the DevOps movement, established to bridge traditional gaps between software development and the operations team running applications. DevOps automates manual testing and release processes by replacing them with scripted procedures. It extends Agile development in building applications incrementally to include the full integration, testing, and validation phases. By streamlining application delivery across the entire development lifecycle, applications can be iteratively developed, automatically packaged and tested, and then released to production in a continuous, rapid, consistent manner. The CD archetype establishes the notion of an ever-evolving production-ready version of the application in a functional, deployable state throughout its lifetime.

This whitepaper delves into the disruptions in software delivery focussing on DevOps and its power to positively influence revenue figures.

Background

DISRUPTIONS THAT HAVE IMPACTED SOFTWARE DELIVERY

"The earlier you catch defects, the cheaper they are to fix."[1]

Cloud technology has commoditized IT environments, creating freedom of choice for running production applications. On demand, flexible computing capacity changed the economic models for application infrastructure. Cloud allows infrastructure decisions to shift, based on cost assessments and real-time traffic patterns rather than cumbersome IT dependencies. Consequently, the production environment becomes less predictable and can change multiple times during the life of an application.

Modern applications are becoming infrastructure-agnostic and developers must adjust their processes to support application delivery practices in which the business logic code is portable and can run virtually anywhere. Enabling the organization to take advantage of production flexibility requires the design to anticipate hybrid and changing production environments. Developers cannot rely on a single production target over the lifetime of the application. New systems of engagement, in particular, the vast expansion of mobile and web application access, force companies to react quickly to shifts in end users' behaviour and adjust to scale and usage pattern swings.

The ability to develop and deliver high-quality software faster is crucial to companies trying to capitalize on mobile use expansion, creating rapid and massive shifts in application workloads. The majority of development projects still have not reached a level of maturity that can address a rapid delivery mind-set.

Early DevOps initiatives focused on accelerating software delivery, and changing the culture to establish better collaboration. DevOps promotes stronger communication skills and gives developers a wider responsibility over the application. Overall, the intent is not for a single entity to cover both application development and management, nor to encourage developers and operations to perform each other's duties, but rather smooth the collaboration between these groups.

The requirement for more multidisciplinary skills for developers created a new DevOps role combining application development skills, IT operations understanding, and infrastructure automation techniques. IDC's 2014 survey revealed that 43 percent of the Fortune 1000 companies' respondents were already using some DevOps practices. An additional 40 percent were evaluating DevOps methodologies.[2]

Understanding the Devops Movement

WHY DEVOPS MAKES PERFECT SENSE TO CFOS

The DevOps movement, initiated primarily by developers, laid the groundwork for comprehensive software delivery automation practices that gradually matured. Software can now be delivered at will rather than at planned intervals, hence CD. **Adopting a CD methodology requires not only a change of mind-set but also a change of tools, standardization on new processes, and implementation of best practices.** Integrating the proper tools into the release process automates distinct manual tasks and synchronizing steps in the release cycle. At its best, such a process requires no manual intervention other than quality control and regulatory checks, in essence industrializing software development. Companies implementing CD practices are therefore industrializing software manufacturing by applying an assembly line approach to the software delivery process.

Automation of manual tasks and consistency of the environments minimizes the amount of repetitive error-prone work done by developers and reduces IT headcount waste. Developers also spent significant time on problem resolution and maintenance versus developing new functionality. A CD process that incorporates comprehensive diagnostics methodologies, including static and dynamic analysis, significantly reduce the time developers spend on problem resolution.

Much like an automated assembly line, CD allows the software manufacturing organization to produce new, reliable code on an ongoing basis. Delivering enhancements to address the current market needs and security threats drive direct revenue and provide a competitive edge. Deriving experience from numerous consulting engagements, Trigent estimates the business impact of accelerating time-to- market credited to shorter cycles depends on the industry and is specific to each company, averaging 20 percent and increasing when software drives the majority of the revenue. Trigent Software finds a 20 percent average reduction in time to market and a 21 percent growth in new functionality delivered to the business. Flexibility in the software manufacturing process not only ensures faster delivery of revenue generating applications, but it also enables more advanced organizations to adjust their IT infrastructure costs based on changing market conditions and price fluctuations. Companies increase their profitability by scaling their environments up or down based on demand, transitioning between cloud vendors or hybrid environments, and taking advantage of the flexibility in the commoditized IT space.

Quality is one of the major concerns and a usual driver of high costs and project delays. Application errors are expensive and negatively impact organizations in the form of reputation damages, customer loyalty, and direct financial impacts. Improving application quality is often the trigger behind building a DevOps practice. The National Institute of Standards and Technology indicates that the cost of finding errors in production are 6-30 times higher compared to identifying them during the development and testing cycle. The actual costs of fixing defects discovered in production can become tremendous when the software is embedded and cannot be easily updated or when the application processes high-value transactions. Quality improvements and earlier defect detection are bigger adoption drivers in such applications. CD provides best practices for optimizing each stage in the application delivery process.

An automated manufacturing assembly line assumes a reliable, consistent, and high-quality supply chain. To achieve code consistency and quality, development teams adopt Agile software methodologies and mandate the use of diagnostics tools at the individual developer level while automated testing and environment provisioning assure that high-quality standards are maintained all the way to production.

Essential Guidance

Communicating DevOps business value is critical to securing additional funding and accelerating the rate and course of change in an enterprise. Finance is the language of business. As such, DevOps teams should consider the following business metrics to communicate success:

- **Productivity:** Speed, velocity, and how much faster the team is executing through faster code development, impact analysis, build and test automation, configuration automation, and time to market
- **Quality:** Improved availability, deeper requirements analysis, early business stakeholder support and involvement, security and compliance risk reduction, and identifying issues earlier through continuous testing and integration
- **Operating expense:** Cost avoidance/optimization, doing more with what you have, fail fast and fail cheap cost modelling, and allocation/bill of IT
- **Capital expense:** Improved utilization, cloud-based systems, and convergence

Understanding DevOps Maturity - Key Principles

For DevOps cultural change to mature into effective CD practice, the IT leadership must adopt and embrace a number of key principals.

- **Production readiness –** The fundamental principle behind CD is the ability to deliver a production-ready release on demand. The organization must reach a maturity level in which the application code is always in a production ready state. Production readiness does not necessarily mean that the application is deployed to production on a continuous basis, but rather the organization retains the ability to do so at will.
- **Uncompromised quality –** Software quality cannot be compromised at any point and the development organization has to prioritize high quality over new features. Ensuring a consistent high quality requires developer's responsibility and proper tooling. It demands tiers of comprehensive testing: Unit testing and static analysis before build and automated functional testing, load, and endurance testing with proper runtime diagnostics tools in place. Quality failures abort the build process until resolution.
- **Repeatable delivery –** The entire software delivery process from build through staging to production must be reliably repeatable so that each iteration is performed in an identical manner. This is achieved by adopting IT tasks automation. Repeated manual tasks that are prone to errors and inconsistencies are also wasting expensive IT resources and time. Automation of these tasks is a prerequisite to any successful CD process.
- **Frequent build and integration –** A CD environment operates with the notion that changes to the application code between build cycles are minimal. Agile, incremental development is practiced alongside CD to ensure that the development project is broken into short iterations. Builds are triggered on all code checked-in to ensure that problems are isolated and addressed quickly.
- **Application stack consistency –** The application stacks should be consistent and automatically provisioned to eliminate environment configuration inconsistencies. Consistency also accelerates the developer's and IT problem resolution capability as it reduces the failures related to application external dependencies.
- **Diagnostics and application management –** High code quality requires problem detection and immediate resolution as defects occur. Fast and meaningful diagnostics data becomes critical to a successful CD implementation. Static analysis and dynamic analysis tools are sequentially deployed during the build cycle providing developers with the insight and drill down data. Lack of developer insight and diagnostics information allows defects to slip through and delay the ability to deliver a quality build.

- **Broad test automation coverage –** Test automation is a prerequisite to ensure high quality and production readiness. Unit tests and multiple layers of automated functional tests are implemented to identify potential issues and regressions. Developers are required to develop unit tests for each submitted piece of code.

Summary

MAINSTREAM ADOPTION OF DEVOPS

DevOps practices see a significant rise in adoption across industries, though misconceptions tend to delay implementation of a complete CD process. That gap is often a factor of initiatives driven by developers compared to a comprehensive company strategic move. The perceived upfront investment is often considered too high, but moreover, there is a notion that establishing a CD practice is a massive undertaking that interrupts the organization for a considerable length of time. Trigent has found companies that have adopted or are looking to adopt DevOps practices see cost and time as their primary barrier to move forward. Since CD is evolving from a collection of DevOps practices that can be implemented independently, over time and to different degrees of maturity, each of these — continuous integration, test automation or release automation delivers business value autonomously. It is, therefore, more common to see companies adopting distinct elements of CD first with less disruption to the organization, and maturing gradually to demonstrate end-to-end automated release capability.

Netflix, Amazon, Facebook, and other industry leaders are publicly known for their ability to deliver new features quickly to production. Facebook, for example, developed principles and a management style that pushes its developers to continuously innovate and improve. The developers are moving and learning fast even if things get broken in the process.[3] Facebook released new code to production twice a day5, using a dedicated release team and developing tools to ensure that new changes do not cause functionality outages. Amazon established its leadership as an Agile powerhouse releasing code to production hundreds of times each day. Impressive as these capabilities are, most organizations do not need to operate at such production release rates. That said, adoption of CD delivers significant productivity, efficiency, and financial benefits even when the organization doesn't require frequent deployments to achieve its business goals. This is a fundamental difference between continuous deployment that suits primarily vendors providing online services and continuous delivery, which provides organizations with the flexibility to release at, will. In fact, many enterprise DevOps organizations that implement CD keep sufficient and strict acceptance validation and authorization gates prior to actually pushing an application package to production.

Organizations tend to adopt DevOps over time and gradually progress along the maturity curve. The route to DevOps typically begins with developers' commitment to code quality, version control, and proper unit testing.

About Trigent

Trigent is a privately held, professional IT services company and a Microsoft Gold Partner with its U.S. headquarters in the greater Boston area and its Indian headquarters in Bangalore. We provide consulting services in various technologies including Microsoft Solutions. Our operating model is to conduct sales, customer relationships and front-end consulting (e.g., business case, requirements, architecture) onsite with our clients and perform the detail design, development, integration, testing and quality assurance offshore at our world class development and support center in Bangalore. We are a SEI CMM Level 4 company and is ISO 9001:2000 TickIT certified organization.

For sales contact sales@trigent.com or call 508-490-6000.

CERTIFIED ISO 9001:2008 COMPANY

Microsoft Partner
Gold Application Development
Gold Collaboration and Content

2016 SOFTWARE 500 COMPANIES

[1] - David Farley, Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation

[2] http://www.idc.com/

[3] https://www.wired.com/2012/02/zuck-letter/