Electric Cloud

POWERING CONTINUOUS DELIVERY

The Continuous Delivery Effect

Understanding the Benefits of Continuous Delivery

eBook

Software is Eating the World

In just about any industry, the success of an organization is increasingly shaped by the strength of its software. Mobile, Web, and embedded applications define how customers perceive your brand, how employees work and collaborate, and how competitive you are in the market.

Users are increasingly intolerant of inconvenience, and impatient for gratification. Failure to meet customer expectations can bring swift and significant repercussions. To stay competitive, companies like Amazon, Flickr, and Etsy are embracing a development discipline known as Continuous Delivery.

What is Continuous Delivery and how can it help you and your organization succeed in a software-driven economy?

Read on and discover the gains Continuous Delivery can bring to your organization's application owners, developers, and IT operations teams.

Continuous Delivery in a Nutshell

Continuous Delivery is a development discipline that takes the practice of Agile development to its logical conclusion, creating software that is always ready to release. It does this by building upon and extending Agile, Continuous Integration, and DevOps practices and tools to transform the way software is delivered.

By automating routine and time-consuming build, test, and deploy processes—and exercising them early and often—Continuous Delivery brings predictability and repeatability to software pipelines, vastly improving release quality and frequency.

With Continuous Delivery, production deployments become a boring, push-button process that can be accomplished by any entitled user at any time. This encourages smaller, more frequent releases— eliminating the risk inherent in larger and more complex “big bang” deployments.

You’re doing CD when:

- Your software is deployable throughout its lifecycle
- Your team prioritizes keeping the software deployable over working on new features
- Anybody can get fast, automated feedback on the production readiness of their systems any time somebody makes a change to them
- You can perform push-button deployments of any version of the software to any environment on demand

— Martin Fowler, 2013

Continuous Delivery: Benefits for Application Owners

Before Continuous Delivery, organizations tended to focus on big-bang application releases, which required a lot of planning and development time and often ended in missed deadlines or buggy code.

In Continuous Delivery environments, the business moves from a focus on big releases and long-term plans to small, frequent, and incremental releases that continue indefinitely.

At its core, DevOps is not about Dev. Or Ops. It's about the business.

It's hard to define what DevOps is. But it's easy to see what it does. It enables a software delivery pipeline that emphasizes agility, reliability, availability, and security.

Continuous Delivery: Benefits for Application Owners

This fundamental shift provides application owners with the ability to:

Respond to market conditions. Quickly.

Traditionally, application roadmaps were defined based on months of research. Teams commit to fulfilling these requirements by agreeing to long-term plans and deliverables. Often, by the time the plans have been implemented, the market has moved on and requirements have changed.

By delivering smaller batches of value on a regular basis, organizations can find out directly from customers whether the new feature or capability actually delivered the added the value or benefits expected.

Improve quality and compliance.

In large, distributed organizations it is hard, if not impossible, to enforce standards and verify compliance with policies. Without Continuous Delivery, business leaders have no way to determine whether code received 100% of the required testing or security validation.

Because Continuous Delivery leverages automation across the lifecycle, it is much easier to enforce standardized processes, and ensure compliance with testing and security policies. This also results in systems with fewer issues at any given time, because complete testing coverage means bugs are discovered and fixed faster.

Ship product on time.

Have you ever missed a release date? You are not alone. Before Continuous Delivery, looming release dates placed increased stress on the development team, which often meant features had to be dropped in order to deliver a viable final product. QA and IT operations teams lacked the time necessary to perform sufficient testing. Once code is deployed, failures and quality issues were common.

By combining well-rehearsed and automated processes with software that is always ready to release, Continuous Delivery provides the business the ability to ship on-demand. With Continuous Delivery, application owners can choose to deploy every feature the minute it's ready, or hold features and bundle them together into a larger release.

Continuous Delivery: Benefits for Development Teams

Before Continuous Delivery, development teams spent much of their time dealing with the process of software production, creating (and re-creating) scripts to handle CI tasks, documenting test requirements for QA teams, and waiting for feedback from unit and system testing.

In Continuous Delivery environments, automation helps make the process disappear for developers, while ensuring that the tasks necessary to build, test, and release code are carried out efficiently.

Continuous Delivery: Benefits for Development Teams

Continuous Delivery provides developers with the ability to:

Make process bulletproof and invisible.

Without Continuous Delivery, processes associated with building, testing, and deploying software tended to be manual, with many error prone handoffs. When individual teams implement scripts to automate their part of the process, these scripts are difficult to share and aren't easily portable to other projects or environments. This patchwork of processes makes it difficult to track the progress of software through the pipeline. Often, managers and staff members have to reference spreadsheets in order to track test coverage and code reviews.

With Continuous Delivery, code commits immediately trigger processes that automatically compile and test any changes. This allows teams to focus on developing new incremental value, confident with the knowledge that automation will ensure their code is always ready for release into production.

Get faster feedback.

Before Continuous Delivery, lengthy build times, insufficient infrastructure resources, non-standard environments, and overworked QA teams bring Agile development to a crawl. These slow feedback loops mean developers either have to wait for test results (destroying productivity) or move on to other work to keep busy (requiring constant, time-sapping context switching). Once feedback is received, going back to fix issues requires developers to re-familiarize themselves with work they did hours, days, or even weeks prior.

Automated testing and standardized environments are the foundation of Continuous Delivery, which means QA tests catch real problems, not configuration errors. Feedback to the developer gets delivered faster, and is more actionable and more effective. As a result, errors can be fixed quickly—before they become expensive to repair.

Eliminate the stress of application release.

Without Continuous Delivery, the application lifecycle is characterized by a series of points at which code has to be "thrown over the wall" to successive QA and operations teams. Each hand off has its own set of efforts, which includes specifying context, configurations, test environments, and tests. Each step presents an opportunity for friction, miscommunication, errors, and delays.

In Continuous Delivery environments, when developers commit code, automation immediately carries it through all required testing processes—eliminating the stress, hassles, and delays of manual project hand offs. Instead of trying to memorize changing processes, and worrying about whether every step was adhered to, developers count on automation to execute the steps and workflows required. And because each commit results in a release-ready build, the decision to actually release can be made without requiring developer interaction or assistance.

Continuous Delivery: Benefits for IT Operations

Before implementing Continuous Delivery, operations teams are saddled with manually building and supporting development, testing, and production environments. Each of these environments may be unique – a "work of art" – each with its own nuances and customer configurations. Fundamentally, this inconsistency breeds issues. Variances mean that, even though the code a developer built ran fine on development machines, issues arise in testing. Worse, code that passes system testing often fails in production – leaving it up to customers to discover the issues.

> *The key test [of whether or not you've achieved CD] is that a business sponsor could request that the current development version of the software be deployed into production at a moment's notice—and nobody would bat an eyelid, let alone panic.*
>
> *— Martin Fowler, 2013*

Continuous Delivery: Benefits for IT Operations

By leveraging Continuous Delivery, your operations teams can:

Make deployments boring.

Prior to implementing Continuous Delivery, each release is large and cumbersome to roll out—a massive, lengthy, and stress-filled undertaking. Further, given releases' scale and complexity, it is hard to roll back to the prior version when issues arise—exacerbating the duration and penalties of having bugs surface in production.

Continuous Delivery environments are characterized by releases that have fewer integration points and code branches, making them simple and stress-free to implement. Further, when finite changes and variables have to be managed, releases can be much more easily and quickly reverted if needed.

Keep up with the rate of change.

Before Continuous Delivery, change was a bad thing for IT operations teams. Because each change is massive, time consuming, and risky, operations staff members can't change as fast as the business needs, and, for that matter, don't really want to.

Leveraging more automated, standardized, and modular approaches, operations can accommodate changes in an efficient, fast, and predictable fashion. Need to scale a test environment to accommodate more tests? Simply spin up additional virtual environments based on pre-defined configurations.

Know what is deployed where.

Before Continuous Delivery, development, testing, and production infrastructures are independently built and maintained. These unique environments make for a dizzying array of patch releases, hardware types, OS versions, and more—with all these variables manually tracked, monitored, and supported.

In Continuous Delivery-driven environments, operations teams take a fundamentally different approach, effectively managing the infrastructure like code. With Continuous Delivery, standardization yields consistency and clarity across environments. Cookie-cutter, well-formulated templates provide a foundation for tracking and managing environmental variables, with a minimum of uncertainty or guesswork.

Electric Cloud
POWERING CONTINUOUS DELIVERY

Conclusion

Competitive advantage, profits, and growth are increasingly tied to the applications a business brings to market. Continuous Delivery represents a vital approach for organizations that want to deliver application innovation to fuel improved business results. With Continuous Delivery, organizations gain the agility needed to deliver innovation faster, while boosting staff efficiency and managing costs.

About Electric Cloud

Electric Cloud powers Continuous Delivery. We help mobile, embedded systems and enterprise web/IT providers deliver better software faster by automating and accelerating build, test, and deployment processes at scale. Industry leaders like Qualcomm, SpaceX, Cisco, GE, Gap, and E*TRADE use Electric Cloud solutions and services to boost DevOps productivity and Agile throughput.

For more information, visit http://www.electric-cloud.com.

Electric Cloud

www.electric-cloud.com

Corporate Headquarters
Electric Cloud, Inc.
35 S. Market Street Suite 100
San Jose, CA 95113
T: 408.419.4300
F: 408.419.4399
info@electric-cloud.com

Electric Cloud Europe
1 Northumberland Avenue
Trafalgar Square, London
WC2N 5BW United Kingdom
T: +44(0)207 872 5500
europe.info@electric-cloud.com

Electric Cloud Japan KK
22F Shibuya Mark City West
1-12-1 Dogenzaka, Shibuya-ku
Tokyo 150-0043 Japan
T: +81.3.4360.5375
japan-info@electric-cloud.com