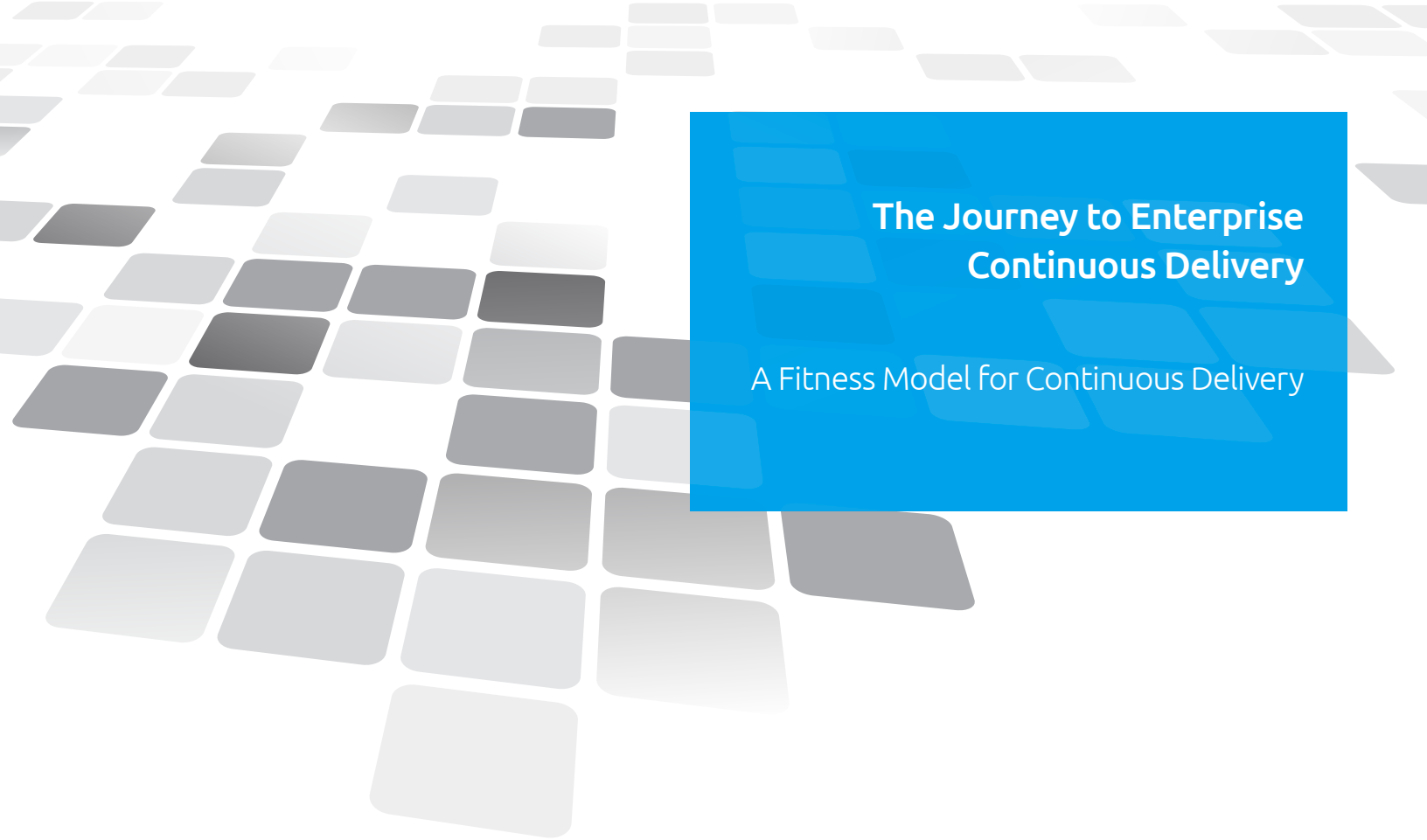
White Paper

Electric Cloud



The Journey to Enterprise Continuous Delivery

A Fitness Model for Continuous Delivery

It is increasingly common to hear businesses describe themselves as software-driven enterprises. From government agencies to companies across all industries, software has become more prominent in addressing faster time to market, competition, and the need for innovation. Organizations are embracing software delivery optimization techniques like Continuous Delivery (CD) in response to these unrelenting pressures and to stay current with modern software development methodologies.

However, the only way for CD to flourish is for the enterprise to overcome inertia and abandon the siloed approaches that have conventionally characterized software delivery. But this is just the start – a fresh outlook must also be backed by tightly integrated technology that furnishes an all-inclusive view of the entire delivery pipeline.

Which organizations should implement Continuous Delivery? CD doesn't necessarily equate to continuous releases. CD enables an organization to be highly responsive to their customer requirements and market, no matter how frequently they release. Therefore it's suitable for organizations of every shape, size, and software delivery profile – even those that release software only once or twice a year – to consider incorporating CD into their software delivery process.

But how do you know if your organization is ready for CD? Drawing from insight into the development environments of hundreds of customers, Electric Cloud introduces a Continuous Delivery Fitness Model. This framework measures an organization's readiness for Continuous Delivery along 3 axes: people, processes, and technology. It is the correct alignment of these three axes, relative to an organization's release frequency goals, that sets the stage for a successful software delivery outcome.

In this paper, Electric Cloud describes why CD is such an essential component to the modern enterprise, but more importantly, helps you assess your organization's readiness to properly implement CD given your current release profile and the alignment of people, processes, and technology.

Those individuals responsible for developing and deploying high quality software to help the enterprise become more competitive would benefit from this report. This includes:

- Line of business leaders
- Technology executives
- Product managers
- Software development leaders
- Architects
- Release managers

Why Continuous Delivery is So Important

Continuous Delivery is the new Agile. When properly implemented, Continuous Delivery (CD) achieves the unfulfilled promise of the Agile Manifesto, which values the delivery of working software, developers working together to meet end goals, customers collaborating with vendors, and teams having heightened abilities to respond to change.

Agile, as a sole methodology, will not enable enterprises to bring higher quality software to market more quickly. The complementary practices of Continuous Integration and DevOps won't solve the software delivery challenge by themselves either. Continuous Delivery is the culmination of all of these development practices.

We consider Continuous Delivery to be an end-to-end process that encompasses the entire build, test, deploy, and release cycle. It's the automation of each of these stages that will make achieving CD a reality. It empowers organizations to increase innovation, reduce costs and differentiate themselves in a competitive marketplace – all through a streamlined and efficient software delivery process.

The concept of Continuous Delivery is gaining traction. However, achieving the optimal results of CD can be difficult. Your staff may not know where to start, and the change from infrequent delivery to rapid delivery is daunting. Meanwhile, your processes are probably focused on local or departmental optimization, rather than overall end-to-end delivery. And finally, you may have hundreds of standalone tools meant to support the software development and deployment processes, but they're probably not integrated or streamlined. Fortunately, by evaluating where you stand regarding CD's prerequisites, you can take the necessary steps to overcoming these barriers.

Once more efficient CD practices are implemented releasing high quality software to customers becomes easier and less stressful, whether you consider the release to be major, minor, or merely a patch. CD means that every delivery has the potential to become a release.

Identifying a release becomes a business decision rather than a decision by your software development team. This concept of delivery itself transforms into continuous readiness as opposed to continuous deployment.

CD is important because every company is a software company. Today's enterprise is tasked with delivering software, whether it's from never-ending demands for more sophisticated applications, new compliance regulations, or a better customer experience. Regardless of software environment, release frequency, or destination such as embedded devices, chips, media, supply-chain customers, and OEMs – CD has broad applicability.

Benefits provided by CD

Continuous Delivery yields significant advantages for both your technical teams as well as your business.

	Benefits for developers and testers	Benefits for the business
1.	Elimination of mundane, repetitive manual tasks	Faster time to market
2.	Increased agility for all phases of the delivery cycle	Better resource utilization and augmented productivity that includes timely and accurate visibility into project status
3.	Tighter feedback loops that reduce time-wasting error detection and correction	Enhanced predictability when bringing software to market: the elimination of "trial and error"
4.	Faster, more efficient iterations between development and testing	Reduced cost and complexity that yields automatic compliance to enterprise standards and policies
5.	A migration away from stressful 'all-or-nothing' infrequent releases towards a more automated daily - or even hourly - release profile	Perpetual, ongoing product and process improvements that cultivate innovation

What Does it Take to Get Ready for CD

Every enterprise is unique, so it should come as no surprise that each organization has its own distinct qualifications when evaluating how to move forward with CD. At Electric Cloud, we work with hundreds of software development organizations which gives us unparalleled insight into various software release frequencies and how to most effectively implement CD.

Software Release Profile

Five software release profiles are shown below in Table 1. Based on each release frequency, we provide the common development methodologies used and the types of businesses that fall into these categories. Understanding your particular profile will help determine how CD might benefit your organization.

Before you review the profiles, it's important to keep the following in mind:

- The job of releasing software assets actually has two dimensions. The first relates to cycle time. In other words, how quickly do you perform the dev/test/delivery cycle? The second refers to release frequency: how often do you ship product? The answer to this second question will determine which profile you fall under.
- A single enterprise may need to operate in more than one delivery profile since there may be many unique applications or systems.
- The profile may change based on where an asset is in its life cycle.
- Even legacy applications can move to a faster release cycle once they're modularized.

Release Frequency	Common Methodology	Types of Business
6 months or longer	Generally compatible with waterfall or spiral methodologies	• Embedded hardware • Mature enterprise software • Very large enterprise software products like Microsoft Office, operating systems • Any software with a very large install base that's reluctant to upgrade frequently
Quarterly	Waterfall or spiral methodologies; often times Agile	• Mobile hardware or software • Major open source initiatives • SaaS major releases • Internal enterprise software and add-ons
Monthly	Requires Agile to hit these frequent milestones and to eliminate feature branching. The rapid delivery pace usually means that software engineers are involved in testing and quickly fixing problems.	• SaaS minor releases • Apps on mobile devices • E-commerce site with seasonal/holiday releases
Weekly	Pace is too fast for Scrum to keep up with these demands. At this point, Kanban (or even continuous Kanban) is required. This build frequency also changes your software architecture: you need to build a platform instead of a product so that you can continually add or substitute components	• Apps on mobile devices • Minor SaaS releases • E-commerce site • Social media • Entertainment
Daily	At this level, automated deployments and continuous release are essential to maintain frenetic release pace.	• Large consumer-facing websites that have high customer responsiveness • Online retail marketing

Table 1: Software release profiles

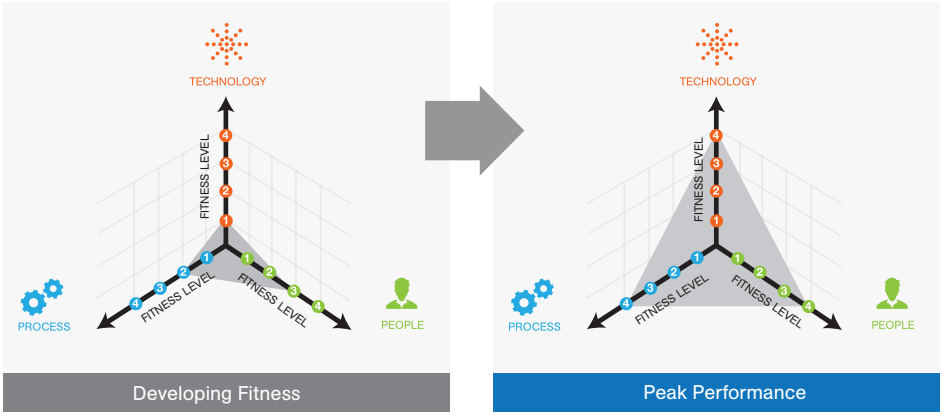
Organizations with monthly, weekly, and daily release cycles would obviously benefit from CD. But CD is also pertinent for organizations that have much longer release cycles, because:

1. When releases happen so infrequently, their underlying build/test/deploy processes tend to be manual and error-prone.
2. This greatly increases the likelihood of bugs or other potential quality issues, and customers are often the first to discover these embarrassing flaws.
3. Correcting these issues with bug fixes and patches is costly, time-consuming, and damaging to the brand.

By automating and streamlining the process of delivering software, CD counters each of these troubling outcomes, resulting in higher quality and lower cost.

How is CD different than DevOps?

DevOps is a related movement with the goal of increasing collaboration between the engineering and IT teams, thereby enabling more frequent and streamlined delivery. This is a radical departure for most organizations, where engineering commonly doesn't understand how to deploy, operate, and scale, and the IT team is resistant to frequent updates from engineering because every update introduces risk that can damage uptime. In a nutshell, DevOps can be seen as both a prerequisite and an enabler of CD.



Assessing your Continuous Delivery Fitness

In this section, we present a Continuous Delivery Fitness Model to help you evaluate your overall state of readiness for implementing CD. The model includes three dimensions:

1. Technology
2. People
3. Processes

For each dimension, we've identified five levels of fitness: 0 (not fit) all the way to 4 (extremely fit). For each level, we itemize a few common traits that we've observed across our large customer install base, as well as how these traits have affected our clients and their CD readiness.

Note that while each of these dimensions is independent, we've observed that enterprises typically mature in each dimension more-or-less concurrently. In other words, it's hard to find a technology-challenged team that collaborates well, or a team of lone wolves that also manages to adhere to consistent delivery practices. Figure 1 illustrates that peak performance is achieved when all three dimensions measure at the highest end of the spectrum (level 4).

State of Fitness: People

First, an organization needs the right culture to support and encourage CD. Unfortunately, far too many organizations perceive the hard work that's necessary to establish effective communication and collaboration as an extraneous obstacle that detracts from overall productivity. It's only when these teams mature and become more ready for CD that they realize just how important these cooperative efforts are.

While smoothly coordinated teams are necessary to implement CD, organizations need to design and maintain logical, streamlined processes to support the actual job of software delivery.

Level	Common Traits	Results
0	• Small teams, a collection of people assigned to a skunkworks project, or an advanced research group in a large enterprise, where there is a no clear thought leadership or a project management leadership structure. • Programmers have strong academic or "prima donna" personalities • They focus on idealistic solutions	Creativity is vital, and is often expressed using free form programming. Work is not consistent in quality because proper software engineering principles are not applied. Issues at integration time typically occur because people are fighting to defend their own individual point of views.
1	• People are isolated from each other: everyone works in cubicles or closed door offices • Communication – when it happens – is via email, and is typically reactive • Leadership tends to be "old school" that meets once or twice a week • A lot of junior people that aren't offered mentoring or apprenticing	Collaboration is poor, and peer feedback is not welcomed. The team always works in a reactive mode. There is limited cross-team communication; primarily only up and down the hierarchy. The project plan is very important: everyone scrutinizes where they are on the plan. When the plan has to change, people panic.
2	• Some collaboration between teams • People tend to focus on the individual components that they're developing • Components are then "tossed over the wall" for others to integrate, test or deploy • Attempts at mentoring junior members are strictly ad-hoc	There's some collaboration, but it's not being conducted effectively. People are somewhat more conscious of business goals, but their primary focus is still tied to protecting individual agendas.
3	• People are very aware of business goals • They are eager to collaborate on how to achieve these business objectives • They seek out and accept feedback • Some attention paid to formalized mentoring of junior team members.	Adaptability to change is high. People understand goals and respect project plans, but appreciate that plans are always subject to change. They care for customer satisfaction, and believe in Jidoka (anyone can stop the "production" line). They also subscribe to concept of Kaizen (continuous process improvement).
4	• Collaboration and cooperation are paramount • There's strong interest in communication and mentoring of more junior team members	Teams are driven by continuous improvement. They adjust to business needs by continually examining requirements and real-time team performance.

Table 2: State of fitness for people

State of Fitness: Process

Fully implemented CD means that you can promote any given software build into a released product. Clearly, a software development team that's releasing software on a more ad-hoc basis won't be able to live up to this standard. But given proper attention and time, any organization can set up the foundational processes that CD needs.

Level	Common Traits	Results
0	• No plans or processes • Ad-hoc software delivery procedures	Building software is considered to be all about creativity, not engineering processes. Majority of developer time is spent in contemplation and brainstorming, rather than following mandated best practices.
1	• There are plentiful project plans but a notable lack of well-defined processes for change requests (requirements, designs, escalation) • No process for continuous testing	Lack of procedures paired with business demands results in many emergencies; everything is a last minute "fire drill". This even impacts the source code, which is frequently scattered among multiple branches.
2	• Several well-defined processes exist (e.g. Continuous Integration, Continuous Testing, Continuous Release), but execution is not consistent nor integrated end-to-end • Agile may be used in small batch sizes	Since processes are both cumbersome as well as un-integrated, people tend to take shortcuts.
3	• Well defined, end-to-end processes are in place • Development, Test, and Release are part of one pipeline, and automation is present throughout the lifecycle • Code is kept in a single branch • More evolved Kanban Agile is utilized	Since the processes are so carefully planned and supported, they "disappear" into the background. This lets people focus on their primary roles and responsibilities without being asked to do "unnatural" tasks. They can focus on deliveries and innovations instead of process compliance-related tasks.
4	• Well defined continuous adjustment and improvement process to meet fast moving business requirements by leveraging data generated by the end-to-end CD monitoring tool	Processes continue to improve by using real-time data as the feedback loop.

Table 3: State of fitness for processes

State of Fitness: Technology

Developers and testers have more infrastructure possibilities than ever. Organizations have numerous tools to select from to streamline development, test, and delivery processes. But technology on its own isn't a panacea, and can introduce as many headaches as it resolves. As organizations become more ready for CD, they find ways to put technology to work for them, rather than the other way around.

Level	Common Traits	Results
0	• Homegrown scripts or manual processes • Basic open source tools	Other than elementary scripting, there's little-to-no automation.
1	• Many siloed tools - often stand-alone and open source • Independently used to solve specific issues such as build, testing, or integration • Commonly operated by different individuals without common understanding, process or oversight	Relies on people for manual integration. While this may work in small groups, people quickly become a bottleneck. The loss of one key member can have a significant impact to the entire team's effectiveness.
2	• Creation of shared infrastructure (testing, preflight, delivery) that helps reduce bottlenecks • However, this newly-deployed infrastructure quickly becomes individual islands of automation with minimal interaction	Scarce developer talent is dedicated to tool creation, customization, and maintenance. There's no integration across regions, and handoff becomes very expensive.
3	• Automation throughout the entire software build/test/deploy life cycle • Automated provisioning and configuration of necessary assets • Centralized, self-service infrastructures that are elastic • End-to-end visibility and traceability	Technology takes care of orchestration in the background. There's very little awareness of tools: they seem to "disappear." Builds are fast and reliable, and when everything happens normally, success is noticed. When errors do occur, they're reported in real-time.
4	• Executive dashboards allow decision makers to make real-time adjustment based on business needs • The business determines which content should be released and when	When adjustments are needed, processes and tooling can be modified in real-time without much downtime or impact to the existing processes.

Table 4: State of fitness for technology

A well-integrated software infrastructure can go a long way towards ensuring CD success. But as we identified above, even the best supporting technology will be under-utilized if the organization's culture and processes doesn't encourage CD.

Assessing your Overall Fitness

Your organization's release frequency, or the frequency you strive to be at, determines how far along each axis you NEED to be in order to have a healthy, effective CD implementation.

As shown in the Fitness Model diagram on page 6, peak performance (level 4 along each axis) enables an organization to operate at a daily release frequency. The table below highlights the typical alignment of people, processes, and technology for each of the software release profiles. For example, organizations that are releasing quarterly need to score roughly in the 2's along each axis.

Scoring less than the designated number in any of the three areas is fine, but it should help organizations determine which areas should be of focus in order to get the most out of Continuous Delivery and to reach their release frequency goals.

Release Profile	People	Process	Technology
6 months or longer	1	1	0-1
Quarterly	2	1-2	2
Monthly	3	2-3	3
Weekly	3	3	3-4
Daily	4	4	4

Moving forward with Continuous Delivery

Shortening the feedback loop is one of CD's primary objectives. This lets developers know when their tasks have been completed satisfactorily, thus freeing them to move onto other responsibilities. These tight feedback loops are critical for any organization seeking innovation, improving quality, and speeding time-to-market. When appropriately employed, CD can serve as a strategic competitive advantage that happens to save money and heighten morale at the same time. Companies with longer release frequencies will have a longer feedback loop but as they become more mature across the 3 axes, they will be able to adapt to a more frequent software release cycle.

A CD Success Story – FamilySearch

FamilySearch is the largest genealogy organization in the world. They are a non-profit business, but as with any organization today, they must be mindful of the quality of their products and the productivity of their software employees. The executives at FamilySearch realized that their long lead-times to get software to customers were slowing down their customer responsiveness. The labor intensive and error prone process of building, testing and deploying software also led to increased frustration for the hundreds of developers.

Software releases took 3 months at FamilySearch given their development complexities. They were automatically building software with Maven, running unit and acceptance tests with Junit, TestNG, and Selenium, and packaging the application for deployment using OS native packaging. They had multiple tools, different teams, and disparate products.

Using Electric Cloud's ElectricCommander solution, FamilySearch's development and ops teams were transformed in how they worked. Bob Hartley, Development Manager at FamilySearch, explained, "In the past we released products in one to three months; now we can go from commit to production every 10 minutes. Automation is a key part of making that happen."

FamilySearch realized that Continuous Delivery was fundamental to improving their alignment to the customer. Faster deployments meant more opportunities for user testing. Errors were detected more quickly, and the development team gets immediate feedback from its users on the value and quality of new software releases. The solution also provided a common workflow and process, improving the automation-centric culture at FamilySearch. The shared workflow promotes consistency and best practices across the FamilySearch organization.

In Summary

At Electric Cloud, we've observed that just about every software development organization is a candidate for Continuous Delivery, but their ability to move forward and achieve their desired release frequency is heavily influenced by their fitness in terms of people, process, and technology. Unfortunately, because of relative immaturity in one or more of these areas, many organizations fail to adequately implement CD despite years of effort.

We offer a collection of holistic, well-integrated products and services that can help you properly deploy CD in your organization. Our Professional Services organization can also provide a site-specific workshop to assess your enterprise's fitness and develop an action plan.

About Electric Cloud

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

For more information, visit electric-cloud.com.

Electric Cloud

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

Electric Cloud China
New City Center Plaza, No.70,
Room 908, Tong Chuan Road,
Shanghai, 200333, China
T: +86 13601825314 / +86 13761649476
china.info@electric-cloud.com

Electric Cloud Europe
1650 Arlington Business Park
Theale, Reading
Berkshire RG7 4SA United Kingdom
T: +44 (0) 0207.872.5500
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

©Electric Cloud, Inc. All rights reserved. Electric Cloud, ElectricAccelerator, and ElectricFlow are trademarks of Electric Cloud, Inc. All other names are used for identification purposes only and are trademarks of their respective companies.