Continuous Delivery
Five Habits of Highly Successful Practitioners

What does it take to make Continuous Delivery successful? Read this eBook to learn five factors that distinguish companies that are doing Continuous Delivery well.
Whether it’s New York Stock Exchange or Netflix, Samsung or Salesforce.com, companies of all kinds have embraced the design practice known as Continuous Delivery.

Continuous Delivery is a development discipline in which software can be released to production at any time. It’s about building a development and release pipeline where early feedback, automated build and test, and incremental deployments into production can dramatically speed up release cycles. Indeed, you accelerate your time to market – providing a continuous flow of “release candidates” for business acceptance – without sacrificing quality.

In the recently published study Continuous Delivery: The New Normal for Software Development, Evans Research Associates finds that more than 28 percent of organizations use Continuous Delivery across all projects, while 37 percent employ Continuous Delivery for some projects. These adoption rates are expected to accelerate.

So what does it take to make Continuous Delivery successful? Here are five factors that distinguish companies that are doing Continuous Delivery well.
Continuous Delivery is focused on more than just “the code.” In fact, all of the elements of a product need to be consistently versioned from development to deployment. Your assets can include chip architectures, CAD designs, documentation, configuration scripts, binaries, etc. After all, applications and products that are popular with consumers today are typically more recognizable for their design, graphics, and interfaces than for underlying functionality.

It’s not just the code that needs to be versioned.

If these additional artifacts are not controlled alongside the rest of the application, you may be releasing incomplete or inconsistent applications, leading to a poor customer experience (or even non-compliance with regulatory requirements).

Therefore, your delivery pipeline should treat non-software assets as first-class components. Developers want to see their commits progressing toward delivery. Showing the same respect to artists and designers will improve overall collaboration and will get integrated into the process. Additionally, when a new version of the product is released, you will have all of the correct configuration scripts, binaries, OS images, etc., ready to launch.
In a highly automated environment, commits immediately initiate and build and tests are executed. A great deal of work starts almost instantly to determine if the changes are acceptable for integration into the delivery package. As a result, if there is a “failure,” that change can be sent back to the development team for correction in an efficient way. Additionally, automating processes enables predictability and repeatability, key attributes to a successful Continuous Delivery approach.

A unified continuous pipeline enables automation at every step of the process. The goal is to enable you to make changes, updates or adjustments quickly and safely while minimizing the introduction of errors. While not every activity can be automated, the more automation you can introduce over time, the more rapidly and productively you can deliver release candidates.

Automation also delivers an additional value: it eliminates manual or ad-hoc processes. ThoughtWorks, a global software delivery and products company, and a thought leader on the subject of Continuous Delivery, recommends introducing automation in stages, starting with monitoring and reporting. From there, you can incorporate automation into the release process. This approach enables you to introduce automation step-by-step.

“A good order is to automate portions that give developers the fastest feedback on the most likely problems, then activities with the highest chances of human errors, and finally things that give developers additional feedback on unlikely problems,” according to the firm.¹ It’s always faster, easier and cheaper to fix an issue on the developer’s desktop than after it’s been handed off for late stage QA testing (or worse, to a customer).

Less manual activity, more automation.

¹ http://www.thoughtworks.com/insights/blog/tackling-your-fear-cd
MAKE EVERYTHING FULLY VISIBLE

Many companies enable universal visibility across the entire production environment. While there might be certain components (e.g., the secure aspects of a company’s trade secrets) that are not available to everyone, what comes out of broad visibility is the opportunity to learn from others in the organization based on their contributions, and the ability to leverage the collective wisdom and output of the group to advance quality and capabilities. Additionally, managers need to be able to see what is in their pipeline at any given point in time across the board.

Improved visibility leads to increased quality as well. Having a warehouse of information that provides a complete catalog of changes across teams, time and projects will enable you to better predict and measure the likely impact of updates in the future.

Sharing a common view of the assets ensures any potential conflicts are found early—not at the last minute before shipping to customers. Errors are reduced strengthening your ability to deliver the highest quality product in the fastest, most repeatable way.

Monitoring and reporting provide the best opportunity to improve visibility. In its eBook Continuous Delivery: What it is and how to Get Started, Puppet Labs highlights the importance of monitoring: “Continuous Delivery is about shortening your release cycles, and that means shortening your test cycles. Doing that is a process of continuous improvement, so you need to continually monitor your testing environment. That lets you figure out how to improve it.”

This includes collecting and analyzing data across environments and the broader infrastructure. Data-driven monitoring provides empirical evidence that testing and automation are, in fact, improving overall performance and can offer insights as to how to improve overall efficiency. For many organizations, visibility through monitoring and reporting is the critical first step towards shifting to a continuous delivery approach.

An ancillary benefit of increased visibility is the effect on team building and culture-shifting.

“Continuous delivery isn’t a one-person show. The entire team needs to see, from moment to moment, if the build is red (broken) or green (working), and what’s actually been released into each environment.”³ This drives collaborative goal achievement through increased communication and teamwork. It also provides the operations team a view into what is happening in development, further streamlining and accelerating the deployment of changes into production and the customer’s hands.

As Andy Singleton, CEO of Assembla, points out, many organizations rely on distributed teams to develop software. However, they all share a common bond: “They are engineers. They focus on facts and they quickly decide what they think is achievable or not achievable.”⁴ There may be a cultural “divide” across teams, but fostering a sense of shared goals (and providing teams visibility into how their work is incorporated) can quickly dissolve those differences.

Visibility enables collaboration across all projects and all teams.
The importance of a “chain of custody” has its roots in regulated industries such as healthcare and finance. However, it’s not only those industries that can benefit. Traceability encourages best practices in development and deployment. Additionally, international standards are now driving adoption of comprehensive version management. For example, ISO 26262 is an emerging requirement for safety in the automotive industry—mandating that a well-managed development lifecycle for electronic components is necessary for increased vehicle safety.

Advanced companies apply a chain-of-custody approach to all aspects of application management, ensuring the tracking of all changes and interdependencies that are delivered as a complete release. Such actions not only facilitate debugging but, in extreme circumstances, protect you in the case of possible litigation.

Tracking is required across the entire product lifecycle to be effective; it’s not just about recording who checked in a file (although that is important). A comprehensive solution will track requirements and how they evolve over time through to validating that the software actually running in production is exactly what it should be.

Always know where all your assets are (and who “owns” each one).
PUT IT ALL IN ONE PLACE

Having a “single source of truth” across the environment and all artifacts can yield immeasurable benefits.

If your assets are distributed in multiple locations and multiple stores, you run the risk of introducing new errors and failing to meet delivery deadlines. You should have a unified repository that can hold all your assets and support virtually unlimited scaling. It should support a heterogeneous environment that might include several different development and continuous integration platforms.

If you unify the collection and management of these assets, you can reduce complexity even as you support dynamic work in a highly collaborative and consistent environment. This is the essence of “version control.”

As a foundational requirement for Continuous Delivery, a unified version control platform provides many powerful benefits. It streamlines siloes from end-to-end, enables teams to work in tighter loops (due to accelerated feedback, automation and testing), and enables the ability to “roll-back” incremental changes if necessary.

A “Single source of truth” should be your truth.
In fact, there are a few key criteria that any versioning platform should meet:

**SCALABILITY**
You must be able to support teams of any size, working anywhere, with no barriers to collaboration. Developers across time zones and geographies must work together as a single team, and use as much build-test-deploy automation as desired.

**UNIVERSAL ADOPTION**
Software engineers, technical writers, artists, hardware designers, and Ops teams all must work together. They should put their assets in a repository that can handle any kind of file.

**SECURITY**
You should have granular access control, applied consistently around the world, that lets you share information with minimal risk. Associated auditing and compliance tools help you with any regulatory or legal concerns.
CONCLUSION: SUCCEEDING WITH CONTINUOUS DELIVERY

The market pressures your organization now faces are enormous. Your customers are more impatient and less forgiving than ever. Under the circumstances, you are expected to produce software much more rapidly without sacrificing high standards of quality.

The streamlining of software development practices will continue. Your challenge is to keep up with the best performers in terms of Continuous Delivery. That’s what it will take to move at the accelerating pace of markets and meet the escalating demands of your customers.

RESOURCES

Kurt Bittner, *Continuous Delivery is Reshaping the Future of ALM*, Forrester (July 22, 2013)


Jez Humble and David Farley, *Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation*, Addison-Wesley (2011)

www.continuousdelivery.com, Jez Humble, Ed


Nathan Wilson, *Achieving Continuous Delivery*, Gartner (January 21, 2013)