DZone

RESEARCH

DZONE RESEARCH PRESENTS

2014 GUIDE TO CONTINUOUS DELIVERY

BROUGHT TO YOU IN PARTNERSHIP WITH

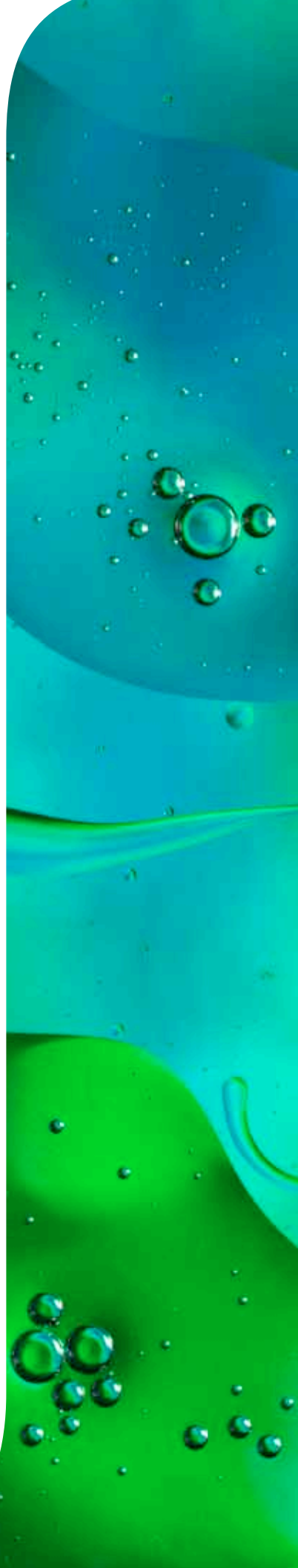
ca technologies

CloudBees

electric CLOUD

XebiaLabs

Automic™



WELCOME

Dear Reader,

In recent years, we have grown accustomed to a constant stream of information that has fueled our news feeds, flooded our inboxes, and triggered constant notifications... and in all of the noise, something has been lost. Clarity, concision, and relevance have become innocent bystanders in an age of information overload. While it is true the task of gathering information has become easier, finding the right information, when and where you need it, has become increasingly complex and, at times, overwhelming.

The guide you hold in your hands (or on your device) is a vital part of DZone's fight against this growing trend. We recognize that as a technology professional, having actionable knowledge at your fingertips is a necessity. Your business challenges will not be put on hold while you try to identify the right questions to ask and sort through endless information. In publishing our *2014 Guide to Continuous Delivery*, we have worked hard to curate and aggregate a wealth of useful topic knowledge and insight into a concise and informative publication.

While this is only our second publication of this nature, we have come a long way so far and we have big plans for the future. I would like to thank you for your support along the way and want you to know that as we continue to publish more research, we are eager for your feedback. Our goal is to make the most informative and practical publications that you can find for free on the web and your input will go a long way in helping us to achieve that goal.

On that note, I hope you find that this guide meets our goal to deliver to you the relevant and useful insight you deserve. Thanks again for taking the time to check it out and, as always, thanks for being a part of our great community.

KELLET ATKINSON
Director of Research
research@dzone.com

TABLE OF CONTENTS

CREDITS

DZONE RESEARCH

KELLET ATKINSON
Director of Research

JAYASHREE GOPAL
Project Manager

MITCH PRONSCHINSKE
Senior Research Analyst

MATT WERNER
Market Researcher

Special thanks to our topic experts Steve Smith, J. Paul Reed, Matthew Skelton, Paul Duvall, Eric Minick, Christopher Little, Benjamin Wootton, Matt Jaynes, Willie Wheeler, Chris Shayan and Andrew Phillips as well as our trusted DZone Most Valuable Bloggers for all their help and feedback in making this report a great success.

DZONE CORPORATE

RICK ROSS
CEO

MATT SCHMIDT
President & CTO

BRANDON NOKES
VP, Operations

ALEX CRAFTS
Senior Account Manager

HERNÂNI CERQUEIRA
Lead Software Engineer

ASHLEY SLATE
Graphic Designer

CHRIS SMITH
Marketing Associate

ALEC NOLLER
Content Curator

BENJAMIN BALL
Content Curator

SARAH ERVIN
Content Curator

SUMMARY AND KEY TAKEAWAYS

There's nothing new about most of today's research on IT's business goals: get applications to market faster, under budget, with less people, and with pristine quality. The best companies will always be driving to achieve these goals. We don't need more research reports to tell us that this is ultimately what business leaders want. What we do need are new ideas and industry data that tell businesses how other organizations are accomplishing these process improvement goals. The DZone *2014 Guide to Continuous Delivery* services this need by providing data, ideas, and solutions that your organization can use to drastically improve its software production process. Explore the content in this guide and use the technology comparison features on its companion site dzone.com/research/continuousdelivery to understand:

- Continuous Delivery, DevOps practices, and their benefits
- Tools and technologies that organizations use to implement Continuous Delivery
- The visual structure of Continuous Delivery pipelines
- Common pitfalls for organizations adopting Continuous Delivery
- Software production metrics and trends from surveying 500+ IT professionals
- Practical strategies regarding software industry best practices for deployment
- Comparison metrics for 30+ solutions to aid Continuous Delivery implementation

KEY TAKEAWAYS

CONTINUOUS DELIVERY: IT'S NOT WHAT YOU THINK IT IS

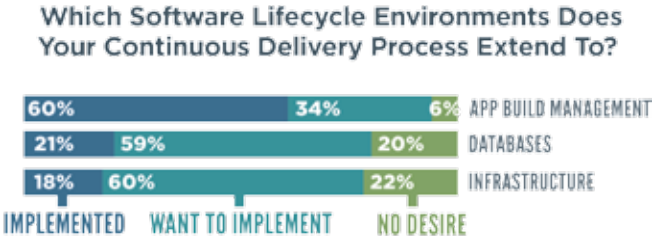
Close to half (41%) of the 500+ IT professionals from our survey believe that they have implemented Continuous Delivery for some or all of their projects. However, based on the questions they answered, only 26% overall are actually performing the basic Continuous Delivery practices. When we filtered respondents by stricter criteria based on three key traits in the definition of Continuous Delivery [1], the number dropped to 8% overall. You can read more about our methodology and breakdown in the *Key Research Findings* section of this guide.

DEVELOPERS SHOULDN'T BE THE ONLY ONES DOING IT

60% of all respondents report that they have Continuous Delivery implemented in their application build management environment. Significantly less have extended these practices to their database management environments and production environments. Industry consultants often observe Continuous Delivery transformations that tend to stop at the development environments, but true Continuous Delivery can't cover just one section of the software production process. It has to be present at every stage.

THE STRATEGIES OF DEVELOPMENT CAN BE EXTENDED TO OPERATIONS

Operations teams can significantly improve their release processes if they employ the tools and techniques of developers. By treating infrastructure configurations in the same way that developers treat source code—employing version control systems, automated testing, and deployment tools—sysadmins can further organize and unify the software deployment process. It's heartening to see that nearly half (48%) of those surveyed use version control for system definitions and environment changes.



MANY ORGANIZATIONS HAVEN'T EVEN MASTERED CONTINUOUS INTEGRATION YET

Continuous Delivery is an extension of Continuous Integration practices into the infrastructure management and the production environment. Many organizations incorrectly assume that they can implement Continuous Delivery even if their CI practices are flawed or incomplete. Supporters of the CD transition must understand that it is an incremental process that starts with CI. A good percentage (38%) of survey respondents are taking the right initial steps toward CD by extending their CI practices to production deployments.

CONTINUOUS DELIVERY IS NOT A FANTASY

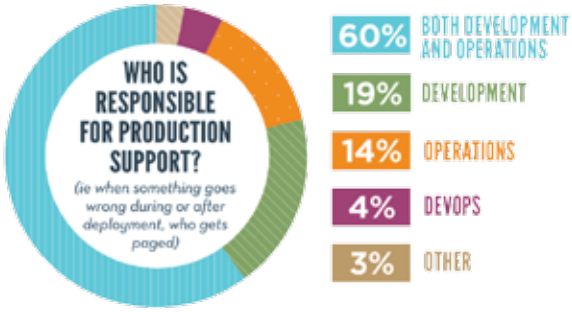
Implementing Continuous Delivery can be more difficult for some organizations, but never impossible. Continuous Delivery isn't just for startups and major web companies like Etsy or Netflix. Any organization can obtain the benefits of Continuous Delivery with consistent effort and the right resources. The content of this guide will show you how.

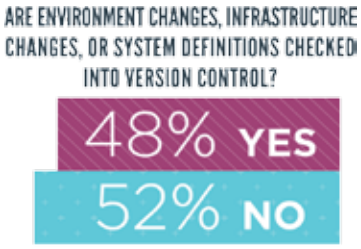
KEY RESEARCH FINDINGS

DZone surveyed over 500 IT professionals for the *2014 Guide to Continuous Delivery*, revealing key insights into DevOps practices, Continuous Delivery adoption, and deployment performance. Respondents came from a fairly even distribribution of large and small companies; 44% came from organizations with over 500 employees and 56% were from organizations with under 500. A large portion of the respondents were developers (68%) or team managers (14%) with smaller portions of operations, QA, and executive management represented. The majority of respondents were headquartered in the US (36%) or Europe (43%).

THE DEVOPS WAY IS CATCHING ON

Continuous Delivery is a practice that fits squarely within the larger philosophy of DevOps, encouraging better technical collaboration and understanding between all areas of an organization, especially development and operations. Some companies have even opted to have a team that is solely focused on this collaboration with cross-compatible skills for multiple disciplines. 30% of the survey respondents have an officially designated DevOps team. Cross referencing this question with company size data shows that these teams are more prevalent in organizations over 100 employees. This is an encouraging sign for the proponents of DevOps and so is the 60% of respondents that said both development and operations were responsible for production support.





MORE PEOPLE ARE TREATING INFRASTRUCTURE AS CODE

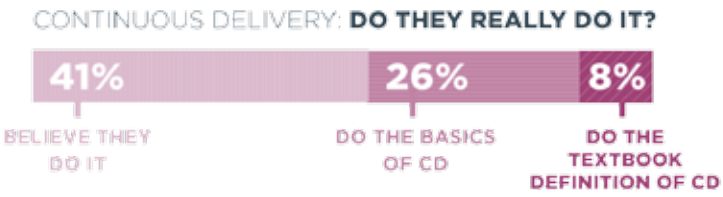
Nearly half of those surveyed are using time-tested development techniques in operational settings. 49% are using using an infrastructure configuration management tool such as Chef or Puppet, and 48% are using a version control system for infrastructure changes and system definitions. However, many organizations aren't automating those infrastructure changes. 73% of respondents still have to use manual scripts for at least half of their infrastructure changes.

ACHIEVING TEXTBOOK CONTINUOUS DELIVERY IS HARD

There are three key traits of Continuous Delivery defined by the authors of the methodology [1] that determine when an organization has fully implemented its practices. The table to the right shows how many survey respondents have these traits in their systems:

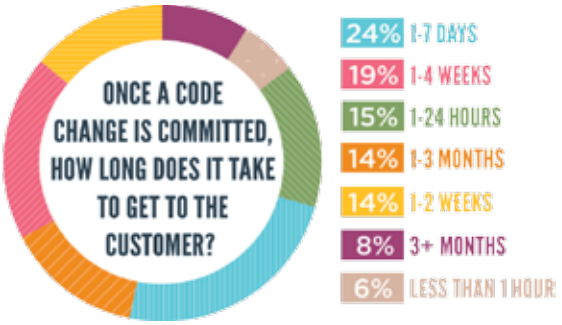
[1] http://martinfowler.com/bliki/ContinuousDelivery.html

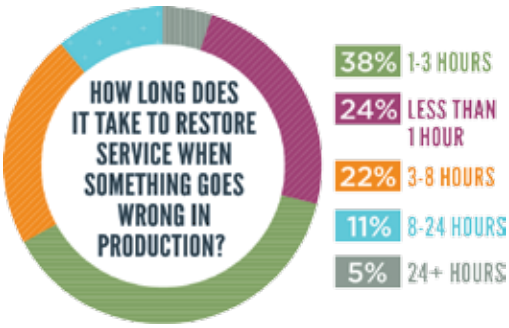
For many, Continuous Delivery is still a work in progress. 27% of those surveyed believe they have achieved CD for some of their projects, and 14% believe they are doing it in all of their projects (41% combined). Respondents were considered to be using basic CD practices if they used version control, issue tracking, resource monitoring, automated at least 30% of their infrastructure changes, and if they frequently confirm their software updates as shippable. Only 26% of all respondents met these criteria. When respondents were filtered by the three key traits of Continuous Delivery from the section above, only 8% said "yes" or "always" for all of the questions.

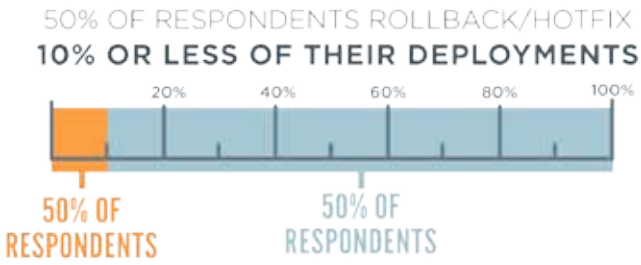


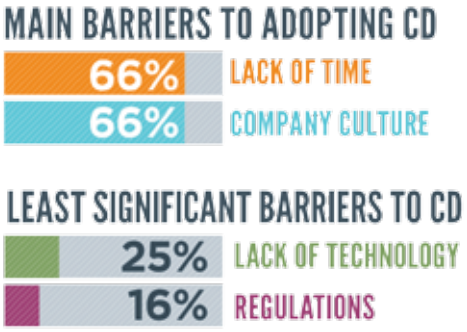
DEPLOYMENT TIMES LEAVE ROOM FOR IMPROVEMENT

Among the 8% of respondents who have all three Continuous Delivery traits, 63% have an official DevOps team and 73% needed to roll back fewer than 10% of their deployments. Compared to the process performance metrics of the entire survey pool, the respondents who exhibit the three key CD traits had faster average deployment times, less rollbacks, and shorter outages. Below are the process performance stats for all respondents:







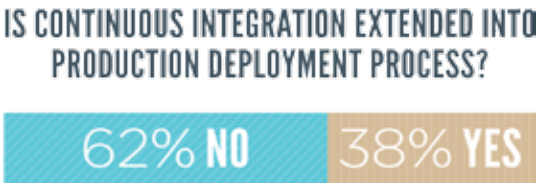


CULTURE AND LACK OF TIME: THE MAJOR CD BARRIERS

The two biggest barriers to Continuous Delivery adoption, both mentioned by 66% of respondents, are lack of time and problems with company culture, such as insufficient collaboration and DevOps practices. Other significant barriers include engineers and sysadmins not having the right skillset (46%), not enough budget (42%), and no support from management (45%). In the survey's free response section, management's support was the most common answer to the question, "what is the one thing that could help you implement Continuous Delivery faster?" The least significant barriers to CD adoption included lack of appropriate technology (25%) and regulation compliance (16%). Surprisingly, 37% of respondents claim that compliance requirements are hindering their ability to automate more of their release process.

CONTINUOUS DELIVERY STILL PRIMARILY IN DEVELOPMENT

Continuous Delivery isn't a hard sell for developers. 60% say that they have already implemented it for their application build environments, and only 6% have no desire to implement in development. Database and infrastructure environments are still lagging behind though, with only 21% of respondents implementing it for the database environment and 18% implementing it for infrastructure. However, over half of all respondents hope to implement it for those environments. The survey also showed some headway toward extending CI practices to production deployments, with 38% saying they've done this.



Introducing: CONTINUOUS DELIVERY

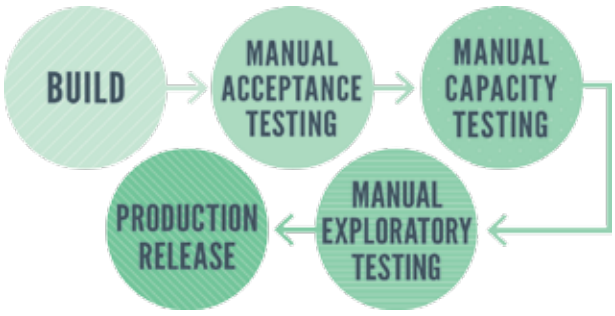
BY STEVE SMITH

"Our highest priority is to satisfy the customer through early and continuous delivery of valuable software." - **The Agile Manifesto**, First Principle

Since the publication of the seminal book *Continuous Delivery* by Dave Farley and Jez Humble in 2010, Continuous Delivery has become a widely discussed topic within the IT industry and an essential competitive advantage for technology companies such as Etsy, Facebook, and Netflix. But where did Continuous Delivery come from, what does it offer, and how does it work?

BEYOND CONTINUOUS INTEGRATION

When Kent Beck published the inaugural eXtreme Programming paper in 1998, his proposal that developers should "integrate and test several times a day" was revolutionary. Enshrined in XP as the Integrate Often rule, the frequent integration of mainline code allows developers to rapidly discover integration problems and reduce development costs. Frequent integration has proven so successful over time that it is now a mainstream development practice known as Continuous Integration. However, since Continuous Integration is focused on development, it can only benefit a fraction of the end-to-end release process, which remains a high-risk, labor-intensive affair in a majority of IT organizations akin to the following:



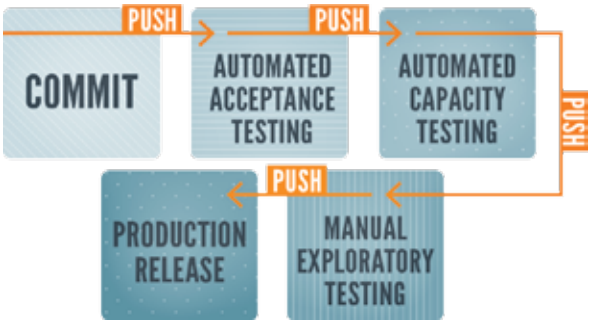
Such a release process will likely involve extensive documentation, overnight scheduling, unversioned configuration management, ad hoc server management, and large numbers of participants. In this situation, software releases inevitably become high cost, high risk events susceptible to human error, and given prominent failures such as the Knight's Capital $440 million glitch [1], there can be an understandable reluctance to release software frequently. However, there is always an opportunity cost associated with not delivering software. This was recently highlighted by reports of Microsoft's decade of e-book/smartphone opportunity costs [2]. This poses a seemingly intractable business problem for many organizations – how can the risk of delivering software be reduced, while simultaneously delivering new features to customers faster?

[1] http://money.cnn.com/2012/08/09/technology/knight-expensive-computer-bug

[2] http://www.vanityfair.com/business/2012/08/microsoft-lost-mojo-steve-ballmer

CONTINUOUS DELIVERY

Inspired by the *Agile Manifesto* which states "*our highest priority is to satisfy the customer through early and continuous delivery of valuable software,*" Continuous Delivery is a method that advocates the creation of an automated deployment pipeline to release software rapidly and reliably into production. The goal of Continuous Delivery is to adopt a holistic end-to-end delivery perspective and optimize cycle time – the average time between production releases – so that development costs are lowered, the risk of release failure is minimized, and customer feedback loops are faster. The result is an automated release workflow similar to the illustration below:



In order to guide Continuous Delivery adoption within an organization, Dave Farley and Jez Humble defined the following principles:

- **Repeatable, Reliable Process:** Use the same deterministic release mechanism in all environments.
- **Automate Almost Everything:** Automate acceptance testing, deployment tasks, configuration management, etc.
- **Keep Everything In Version Control:** Store all code, configuration, schemas, etc. in source control.
- **Bring Pain Forward:** Shrink feedback loops for time-consuming, error-prone operations.
- **Build Quality In:** Fix defects in development as soon as they occur.
- **Done Means Released:** Do not consider features complete until released to production.
- **Everybody Is Responsible:** Align teams and individuals with the release process.

- **Continuous Improvement:** Continuously improve the people and technology involved.

These principles are promoted by the deployment pipeline pattern, which has been described by Dave Farley and Jez Humble as "Continuous Integration taken to its logical conclusion" [3] and lies at the heart of Continuous Delivery. A deployment pipeline is an automated implementation of the build/deploy/test/release cycle that enables self-serviced releases of any application version into any environment. A diagram of a typical deployment pipeline can be found in a free online chapter of Continuous Delivery [4]. A real world pipeline, however, will be specialized to an organization's requirements in the software delivery process.

[3] http://www.amazon.com/dp/0321601912

[4] http://ptgmedia.pearsoncmg.com/images/chap5_9780321601919/elementLinks/fig5_4.jpg

The deployment pipeline pattern is Continuous Integration taken to its logical conclusion.

In the basic deployment pipeline pattern, the commit stage is triggered by a source code or configuration change. This stage includes code compilation, unit tests, static analysis checks, and assembly of the application binaries for the binary repository. A successful run automatically triggers the acceptance stage, which runs the automated acceptance tests against that application version. If the acceptance tests pass, that application version can progress to manual exploratory testing, automated capacity testing, and production usage. This closely follows the deployment pipeline best practices:

- **Build Your Binaries Only Once:** Create immutable binaries to eliminate recompilation errors.
- **Deploy The Same Way:** Use the same automated release mechanism in each stage.
- **Smoke Test Deployments:** Assert deployment success prior to usage.
- **Deploy Into Production Copy:** Create a production-like test environment for testing.
- **Instant Propagation:** Make an application version automatically available to the next stage upon success in the previous stage.
- **Stop The Line:** When an application version fails on a certain stage, automatically stop its progress through the pipeline.

The creation of a deployment pipeline establishes a pull-based release mechanism that reduces development costs, minimizes the risk of release failure, and allows the production release process to be rehearsed thousands of times. It provides visibility into the production readiness of different application versions at any point in time, and drives continuous improvement of the release process by identifying bottlenecks in the system.

ORGANIZATIONAL CHANGE AND DEVOPS

While a deployment pipeline reduces costs and risk, the reality is that Continuous Delivery is vulnerable to Conway's Law [5] and dependent upon organizational change to achieve a significant reduction in cycle time. In the majority of organizations, a deployment pipeline will be used by non-aligned siloed teams, meaning that lead times will be substantially inflated by handover delays between teams regardless of pipeline execution time.

Dave Farley and Jez Humble have repeatedly warned that "where the delivery process is divided between different groups... the cost of coordination between these silos can be enormous," [3] and this is reflected in the *Everybody Is Responsible* and *Done Means Released* principles listed above. Testing and operational tasks must become intrinsic development activities rather than discrete work phases, and the people involved must be integrated into the product development team in what is often a slow and painstaking process of change. It is for this reason that the parallel growth of the DevOps philosophy has been welcomed by the Continuous Delivery community.

DevOps has been defined by Damon Edwards as "aligning development and operations roles and processes in the context of shared business objectives," [6] and aims to increase communication and collaboration between IT divisions such as development and operations. Continuous Delivery and DevOps have evolved independently, but are interdependent upon one another – a deployment pipeline can act as a focal point for DevOps collaboration, and the DevOps integration of Agile principles with operations practices can eliminate the handover delays between development and operations teams.

CONCLUSION

While Continuous Integration has become a mainstream development practice, Continuous Delivery goes much further and is poised to become a mainstream IT practice. Through creating an automated deployment pipeline, a repeatable and reliable delivery mechanism enables an organization to increase product revenues by releasing new features to customers more frequently without fear of failure. However, Continuous Delivery is far more reliant upon organizational change than technology change in order to truly be successful.

[5] http://www.melconway.com/Home/Conways_Law.html

[6] http://dev2ops.org/2010/02/what-is-devops/

WRITTEN BY

Steve Smith

Steve Smith is an Agile consultant and Continuous Delivery specialist at Always Agile Consulting Ltd. Steve is a regular speaker at Skills Matter for the London Continuous Delivery group, and has spoken at conferences such as Agile Horizons and QCon New York about Continuous Delivery.

EXTENDING **CONTINUOUS INTEGRATION** TO **CONTINUOUS DELIVERY** WITH JENKINS

Jenkins has been used widely among developers for several years now, and various polls suggest that it is by far the most adopted continuous integration (CI) server on the market today. Deployment automation is widely accepted and practiced to the point where Jenkins has become mission-critical for many companies, which is amazing.

While a growing number of people embrace automation, the depth of automation among existing users has gotten deeper too. We are now at the point where it is completely practical to build software-managed automation all the way from source code review to production deployment. This process is known as Continuous Delivery (CD). The days when Jenkins was used by developers who only had to automate builds and hand binaries off to the operation guys are over. Instead, an increasing number of teams are responsible for the entire lifecycle from coding to operation, and this is how Jenkins is used by sophisticated users today.

Continuous Delivery is a never-ending practice of pushing the envelope of automation little by little, empowering people to focus and attain higher productivity and visibility into the process.

The CloudBees Continuous Delivery Platform is powered by Jenkins to help people get there. Thanks to the incredible plugin ecosystem of Jenkins, it can be deployed in just about any kind of environment, regardless of whether you use Java or PHP, whether you are writing web apps or mobile apps, or whether you are doing it on premise, in the cloud, or somewhere in-between.

I invite you to take a look at how you can push your automation further with CloudBees' Jenkins-powered Continuous Delivery Platform. Whether you have just started using Jenkins or have been using it for 5 years, there's always an opportunity to take it one step further and get even more out of it. Assisting code reviews, running tests in parallel, tracking deployments, you name it—you can orchestrate all sorts of complex application delivery processes.

WRITTEN BY
Kohsuke Kawaguchi
Chief Technology Officer
CloudBees

CloudBees Continuous Delivery Platform

COMPANY: CloudBees **CI**

CloudBees

PHONE: (323) 843-4483	**LOCATION:** Woburn, MA, USA	**LAUNCH DATE:** January '11	@CloudBees

SUPPORTED TARGET SYSTEMS

- ☑ Windows
- ☑ Linux
- ☑ Mac OS X
- ☑ Solaris
- ☐ AIX
- ☐ Other *(See companion site)*

STRENGTHS

- Jenkins-based Continuous Delivery lifecycle
- On premise, public cloud, or hybrid cloud configuration and hosting options
- Hosted iOS and Android builds for mobile Continuous Delivery
- Manage multiple Jenkins configurations across the enterprise

NOTABLE CUSTOMERS

- ChooseDigital
- Netflix
- Altares
- Viridity Energy
- Alcatel-Lucent

FREE TRIAL

30-day free trial

FULL PROFILE LINK dzone.com/r/aVAt

DESCRIPTION

The CloudBees Continuous Delivery Platform provides a range of Jenkins-based Continuous Integration and Continuous Delivery solutions on premise, in the cloud or in a hybrid configuration. The CloudBees Platform offers solutions for both development and DevOps teams to utilize Jenkins and Platform as a Service technology in a configuration that best meets their needs. The CloudBees CD Platform can also be extended to work with technologies across the application lifecycle.

CUSTOMER SUCCESS STORY

Viridity Energy produces the Java-based VPower software, which allows Viridity's customers to manage their electricity usage and transform energy profiles into financial returns. For VPower, Viridity Energy needed to streamline and accelerate the deployment process, establish an integrated build and runtime environment, and support continuous integration along with the Scala programming language. CloudBees' PaaS helped VPower accelerate reactive application development by providing Jenkins-based continuous integration, reducing deployment windows by 85% and costs by 66%, and scaling with Viridity's needs.

HOSTING OPTIONS

- On premise, public cloud, or hybrid cloud

CONTINUOUS DELIVERY **PITFALLS**

BY J. PAUL REED

There is little question that Continuous Delivery provides a compelling business story. Everyone is looking for ways to increase their own business agility with cloud-based builds, built-in unit and integration testing, and fully automated deployments, all instrumented with monitoring utilities. If there weren't several high-profile companies discussing both their Continuous Delivery successes and failures so publicly, it might all seem like a fairy tale. The incredible complexity of software production makes it extremely difficult to create a clean path for software to travel from a developer's keyboard all the way to the customer's hands.

For every one of the "Continuous Delivery unicorns," like Etsy and Netflix, there are at least ten organizations who struggle significantly to implement CD within their organization. Many of those will ultimately fail entirely to deliver on CD's promises.

Like any new development methodology, implementing Continuous Delivery has a number of pitfalls that can trip up even the most mature organizations. Successful Continuous Delivery processes are predicated on a number of technical and cultural assumptions, and in many cases organizations don't have the foundation necessary for CD. Organizations often try to copy the practices of a company like Netflix wholesale after reading some of their resources, ignoring the differences in their own products or market. These examples are just a few of the reasons why companies find themselves in trouble when adopting Continuous Delivery. Below are four of the most common pitfalls to avoid when implementing Continuous Delivery.

1. *Attempting to Build Continuous Delivery on Top of an Unstable (or Non-Existent) Continuous Integration Foundation*

Continuous integration is a foundational requirement for Continuous Delivery. The constantly referenced Continuous Delivery deployment pipeline is really just the practices of continuous integration extended to infrastructure management and the production environment. Thus, any successful CD implementation begins with a CI system that is stable, operationalized, and able to provide actionable data. The organization should also foster a company culture that knows how to react to that data.

Many environments, however, have no continuous integration currently deployed or a poor CI foundation for CD. Common issues with CI infrastructures include:

- Operating system configuration, libraries, and tool inconsistencies between developer and build/test/production environments.
- Inconsistencies among CI agent configuration (despite how much it's emphasized as an issue by the DevOps movement, it remains a disturbingly common problem).
- CI master servers or agent hosts running on individual employees' computers or in unofficial/personal cloud accounts. This is often observed in organizations that need to support Mac OS X.
- CI masters with no access control, with job configurations that can be modified without notification or an audit trail.
- Unactionable CI, which can be caused by insufficient communication mechanisms from the CI infrastructure (email, which everyone promptly filters for instance), or a cultural barrier, where CI errors or build/test failures are not considered by everyone in the organization to be worthy of a timely response.

In the worst cases, organizations get *excited* about CD and try to *skip over* CI.

The bad news: the solutions to these problems are varied. Some are simple, while others (especially those involving company culture) may be tougher to overcome.

The good news: once the organization has achieved stable, operationalized, actionable continuous integration, you've already done 50% of the work to build a legitimate Continuous Delivery environment.

2. *Confusing Continuous Delivery with Continuous Deployment*

Since Continuous Delivery and continuous deployment have very similar themes, it's easy to understand the origin of this confusion. Continuous Delivery is the discipline and infrastructure around the *ability* to deploy changes *whenever it makes sense for the business to do so*. It does not require that every single developer commit directly out to production in real-time.

Getting every single improvement out to customers the moment that it is checked in is so tantalizing that business stakeholders often believe that this is the ultimate goal of Continuous Delivery. The metric of achieving commit-to-release for all situations starts to be overly emphasized, leading to the Hawthorne Effect [1], where people improve performance on the metrics they know they're being measured by. Unfortunately, the effect is often temporary.

To emphasize the difference between Continuous Delivery and continuous deployment, one just has to understand that Continuous Delivery is possible even in low-change tolerance, high risk environments, like a nuclear power plant. In such situations, the focus isn't on continuously

deploying updates to plant software and infrastructure, but rather on integrating and deploying it to staging infrastructure so that even infrequent deployments will have a higher probability of success.

To be clear, continuous deployment is not a fantasy: organizations who have long focused on their Continuous Delivery pipelines and processes, like Etsy and Facebook, do deploy changes at an attractively high rate. But for organizations starting with Continuous Delivery, focusing on the path commits take through the delivery pipeline, as opposed to the rate at which they're deployed, will produce better initial results and a higher chance of permanent adoption.

[1] http://en.wikipedia.org/wiki/Hawthorne_effect

3. *Complicated or Inconsistent Source Code Workflows*

Continuous Delivery father Jez Humble often conducts an informal survey during talks. He'll ask the audience to raise their hand if they do continuous integration. Most hands go up. His second question: "Put them down if all the devs on your team don't check into trunk/mainline/master at least once a day." Many of the hands invariably go down.

This survey indicates that feature branches and complex branching/merging processes are an enemy of continuous integration, and therefore of CD. In fact, in its pure form, Humble argues there is no need for any branch other than (what Git calls) the *master* branch.

This is especially hard for users of Git to accept because of its ability to easily create and publish infinite feature branches. Overuse and reliance on an unmerged codeline can make code flow through a delivery pipeline difficult for anyone to track. Creating release branches and other automation-related branches can quickly add complexities that make it almost impossible to answer simple questions like, "show me all the commits that are in this release, but not in the integration branch."

This doesn't mean that Git is a bad tool for CD, but users should know that complicated branching/merging and different individual workflows will make Continuous Delivery very difficult, regardless of the tools used.

4. *Attempting to Switch to Continuous Delivery Without any Supporting Infrastructure*

The ability to deploy code to production at a moment's notice requires visibility into the state of your application. Your application may require configuration management, monitoring and alarms, and rollback conditions. Setting up this infrastructure is not very exciting and many times new features will get priority and investment instead of infrastructure support. This is a mistake that will make the implementation of CD more difficult.

A great example of this is Mozilla Corporation's move from 18+ month release cycles for its Firefox web browser to six week release trains. The move required a massive investment in QA process and infrastructure, including a cultural change so that *every* component of the browser and *every* bug fixed had a comprehensive suite of unit tests, ensuring that any mistakes that would "break the web" would be easy to spot as the rate of change to the code-base increased. It required an investment in build/release infrastructure to support running all of these tests on the supported platforms and mobile devices at a scale previously not encountered.

> *In the worst cases, organizations get excited about CD and try to skip over CI.*

In short, it required that the business invest heavily in infrastructure which, to put it frankly, isn't sexy and usually doesn't provide *direct* business value. In many software shops, infrastructure has been chronically underinvested in already, so they are already starting their CD initiative at a disadvantage. Successful Continuous Delivery stories commonly emphasize the organization's commitment to investing in infrastructure throughout the CD transformation and beyond.

5. *Difficult, but Worth It*

Moving to Continuous Delivery as the release mechanism for your software is not a trivial undertaking. Similar to implementing DevOps, it requires committed investment in new tooling and a re-examination of the organizational culture from all technical and business teams before any benefits can be gained. This can be very difficult if the cultural foundation for Continuous Delivery is initially unstable. It's even harder if you've fallen into one of the pitfalls mentioned in this article and still believe that you're going down the right path. It takes work, humility, perseverance, and a commitment to balancing feature development with infrastructure development. If your organization is willing to put in the extra effort to get things right, you will put your business on the path to reliably ship software any time.

WRITTEN BY

J. Paul Reed

J. Paul Reed has over a decade's experience in the trenches as a build/release and tools engineer, working for such companies as VMware, Mozilla, and Symantec. In 2012, he founded Release Engineering Approaches, a consultancy incorporating a host of tools and techniques focused on helping organizations "Simply Ship. Every time." He also hosts a DevOps podcast at theshipshow.com.

YOU WRITE CODE.
WE GET IT TO YOUR USERS. SIMPLE.

START BUILDING YOUR CONTINUOUS DELIVERY PLATFORM WITH XEBIALABS

- Start with your current release process and transform it into Continuous Delivery pipelines

- No More Deployment Scripts! Automate your deployments with a ton of out-of-the-box logic for on-prem middleware, cloud-hosted services, database updates and more.

- Manage your Agile test suites: You have lots of automated tests running in many different test tools, and need to try to get a overall picture to analyze and optimize them? **We can help!**

- Create and scale your target environments: no more troubleshooting sessions trying to figure out why your app doesn't run in your test, QA or production environment!

- Analyze configuration drift: define, discover, track and compare application and system configuration from Dev to Production!

- Continuous Delivery Success!

BUILD ON WHAT YOU HAVE, ADD WHAT YOU NEED:
XL Platform integrates with Jenkins and all leading CI tools, as well as ALM, system provisioning and configuration tools, and much more.

THE SOONER YOU GET STARTED, THE FASTER YOU CAN GET YOUR GREAT FEATURES TO YOUR USERS.
EVERYBODY WINS!

Learn More **Try XL Platform**

XebiaLabs
Deliver Faster

Continuous Delivery **One Improvement at a Time**

Getting your code and great new features into the hands of your users is the most important part of your development lifecycle. We all want to get away from painful month-long release cycles, stressful troubleshooting sessions trying to figure out why our system is not working in the next environment, and endless early-morning status calls.

Continuous Delivery makes life so much easier for developers, allowing you to focus on the important stuff: writing cool code and implementing great features. It's also easy to sell to your managers, allowing you to speed up time to market while maintaining service quality, and to close the feedback loop between developers and users.

> *You write great code. We get it to your users. It's that simple.*

One of the fundamental principles—perhaps the fundamental idea—of Continuous Delivery is the notion of *steady, incremental improvement*. Getting CD right in a real-world situation, especially in the enterprise, means having a clear and pragmatic vision of what your goals are—initial and long-term—and introducing tools and practices to build the Continuous Delivery tool chain and culture that is right for *you*.

Our delivery automation is designed with one goal in mind: to allow you to get started on your road to Continuous Delivery *today*. Whether it's deployment automation, on-demand environment provisioning, Agile test set management, or the full coordination and orchestration of your release process and delivery pipelines, our tooling allows you to get started in your *current* situation and with your *current* processes. Improve them incrementally to build the delivery pipelines that *you* need, and grow your DevOps teams and culture, day by day.

We have the vision and knowledge of what fully-automated delivery pipelines and a strong DevOps culture can do—it's how we develop our own software. But we also have the hands-on, hard-nosed experience of what Continuous Delivery implementations look like in the real world—the kind of experience that allows us to build tools that will help *you* succeed.

WRITTEN BY

Andrew Phillips

VP Product Management
XebiaLabs

XebiaLabs XL Deploy

COMPANY: XebiaLabs **ARA**

XebiaLabs

PHONE: (323) 843-4483	**LOCATION:** Boston, MA, USA	**LAUNCH DATE:** May '11	@xebialabs

SUPPORTED TARGET SYSTEMS

- ☑ Windows
- ☑ Linux
- ☑ Mac OS X
- ☐ Solaris
- ☑ AIX
- ☐ Other *(See companion site)*

STRENGTHS

- Agentless architecture
- Declarative automation that eliminates scripting and workflow design
- Fully automated rollback and recovery
- Enterprise security and centralized auditing
- Visualize Continuous Delivery pipelines with template reports

NOTABLE CUSTOMERS

- 3M
- GE
- John Deere
- Expedia
- Digital Globe
- Tribune

FREE TRIAL

Fully functional, time-limited evaluation version. Trial license extensions available upon request.

FULL PROFILE LINK dzone.com/r/M3aA

DESCRIPTION

XL Deploy is an agentless application release automation solution. XL Deploy combines intelligent deployment automation, the richest content set for enterprise middleware, and seamless integration into the application delivery ecosystem. XL Deploy's unique, agentless architecture means less maintenance, lower costs, and support for much more diverse deployment scenarios that will become common with the Internet of Things.

CUSTOMER SUCCESS STORY

The Rabobank Group is a multinational banking and financial services company that maintains hundreds of in-house and customer-facing applications. Before using XL Deploy, it took them four weeks to manually modify the infrastructure configuration for each application across every environment. XL Deploy allowed Rabobank to automate the entire deployment process and develop infrastructure configurations along with the application code. By integrating Continuous Delivery into the deployment process with XL Deploy, Rabobank has accelerated their time-to-market while eliminating errors and reducing costs.

SUPPORTED CI ENGINES

- Jenkins/Hudson
- Atlassian Bamboo
- Microsoft TFS

THE CONTINUOUS DELIVERY TOOLCHAIN

BY MATTHEW SKELTON

Unless your software is very simple, no single tool, automation product, or deployment pipeline implementation will provide you with Continuous Delivery. Effective Continuous Delivery requires an organizational understanding of the intent and purposes of the activities you undertake, not merely the automation of those activities. However, Continuous Delivery is impossible without some core capabilities that tools provide. Each section below will examine a link in the core Continuous Delivery toolchain with examples from each tool category in the boxed sections.

ORCHESTRATION AND DEPLOYMENT PIPELINE VISUALIZATION

Orchestration tools are the backbone of any CD system. They allow teams to build an effective sequence of deployment pipeline steps by integrating with their entire toolchain. These tools can also provide visualization utilities, which are important for enabling the full involvement of stakeholders from all departments. For pipeline orchestration and visualization, you can use a dedicated deployment pipeline tool or you can use an application release automation (ARA) solution. Whichever direction you take for your orchestration tool, you should be sure that it helps your team detect and expose delays at each stage of the pipeline, including wait times between stages. Visualization and orchestration working in tandem allow teams to quickly identify the places they should optimize first.

Dedicated Deployment Pipeline Tools: *Jenkins, Travis CI, ThoughtWorks GO, CircleCI, JetBrains TeamCity, Atlassian Bamboo*
ARA: *ElectricCommander, CA LISA, IBM UrbanCode, XebiaLabs XL*
Orchestration Engine: *MaestroDev, CollabNet*

VERSION CONTROL

Most software development teams use a version control system for the files they consider source code. However, many organizations forget to include configuration files, such as the configuration that defines the build and release system. All text-based assets should be stored in a version control system that everyone can easily access. The code changes should be very easy to review, line-by-line (ideally in a web browser), with a pull or merge request.

Version Control: *Git, Mercurial, Perforce, Subversion, TFS*

CONTINUOUS INTEGRATION

CI tools can support orchestration and visualization, but their core functionality is to integrate new code with the stable main release line and alert stakeholders if any new code would cause issues with the final product. This makes it easy for teams to combine work on different features while keeping a master code branch ready for release. It should feel natural to integrate many times a day. Teams should also connect a code metrics and profiling utility that can stop integrations if certain metrics reach an undesirable threshold.

CI: *Jenkins, Travis CI, ThoughtWorks GO, CircleCI, JetBrains TeamCity, Atlassian Bamboo*
Code Metrics: *SonarQube, SLOC (and variants), SciTools Understand*

ARTIFACT MANAGEMENT

Packaged artifacts, rather than the application's raw source code, are the focus of deployment pipelines. Artifacts are assembled pieces of an application that include packaged application code, application assets, infrastructure code, virtual machine images, and (typically) configuration data. Artifacts are identifiable (unique name), versioned (preferably semantic versioning), and immutable (we never edit them). Together, these artifacts allow developers to build a bill of materials (BOM) package describing the exact versions of all the artifacts in a particular version of their software system. Package metadata identifies when and how the package was tested or deployed into a particular environment.

Ensure that versioned, traceable artifacts are the key unit of currency for the deployment pipeline.

Artifact management is most effective with an artifact repository manager. Artifact repositories contain a complete artifact usage history, similar to the way version control systems track source code changes. They use dependency resolution between the package versions and allow the system to build a dependency graph from the hardware all the way up to the user interface of the application. The ability to verify dependencies through an entire system is powerful for tracking exactly what was (or will be) tested or deployed.

Version Control Systems: *Git, Subversion, Mercurial*
Artifact Repository Managers: *Archiva, Artifactory, Nexus (roll-your-own with zip files, metadata, shared storage, and access controls)*
Language-specific Package Managers: *Composer (PHP), Ruby Gems, npm (Node.js), Python PIP*
OS-level Package Managers: *APT, Chocolatey, RPM*

TEST AND ENVIRONMENT AUTOMATION

The only manual testing in a deployment pipeline should be for tests that are tough for a computer to handle, such as exploratory testing, inspection of user interface designs, and user acceptance tests. The rest should be automated. Tools for automated testing should operate in a completely headless (non-interactive) manner and be lightweight enough to run across many test servers simultaneously. Teams also need to create testing environments on-demand by using environment automation tools that can provision a VM and configure an environment template.

Test Automation: *JMeter, Selenium/WebDriver, Cucumber (BDD), RSpec (BDD), SpecFlow (BDD), LoadUI (Performance), PageSpeed (Performance), Netem(Network Emulation), SoapUI (Web Services), Test Kitchen (Infrastructure)*
Environment Automation: *Vagrant, Docker, Packer*

SERVER CONFIGURATION AND DEPLOYMENT

Current deployment tools support three models:

- **Push model:** Manages the distribution and installation of packages to multiple remote machines. It's a good choice for smaller systems because it's usually simple and quick.
- **Pull model:** Requires an infrastructure configuration tool such as Chef or Puppet. Supporters say it scales better than push and like that it treats the deployment of application code as another step in configuring the infrastructure.
- **Hybrid model:** Uses a push tool to trigger a pull client on target servers.

For any of the three models, your team must ensure that the process is fully automated, provides detailed information with standard output and error messages, and allows easy and rapid rollback to a stable state.

Push deployment: *Capistrano, Fabric, ThoughtWorks Go, MSdeploy, Octopus, RunDeck, various CI and build tools, various ARA tools*
Pull deployment: *Ansible, Chef, CFEngine, Puppet, Salt.*

MONITORING AND REPORTING

Monitoring your system logs is essential for spotting problems and halting the deployment pipeline. Rather than manually collecting logs from each machine in an environment, logs should be shipped to a central store that indexes them and makes them available for searching via a web browser. This is a crucial capability for a Continuous Delivery environment. The log store should be connected to all environments (including the developer's system) to speed up problem diagnosis and resolution. Most monitoring tools should work in a dynamic infrastructure and integrate with yours through scripted configuration.

Log Aggregation & Search: *Fluentd, Graylog2, LogStash, nxlog, Splunk*
Metrics, Monitoring, Audit: *Collectd, Ganglia, Graphite, Icinga, Sensu, ScriptRock*

A FINAL LOOK AT SOME GUIDING PRINCIPLES FOR TOOLS

It's important to understand all the capabilities your team needs before selecting the tools to build your Continuous Delivery system. The following list details the key areas you should always keep in mind when selecting tools:

- **Visibility:** Look for tools that have clear, comprehensive visualizations for everything that your organization needs to track.
- **Traceability:** Select tools that will allow you to easily track important metadata from your source code, infrastructure code, binary artifacts, application and infrastructure configuration, VM images, and the deployment pipeline configuration.
- **Full Coverage:** Tools must cover all of the applicable environments in order for delivery to be consistent. If a tool is too expensive to have in every environment, you shouldn't choose it.

In addition to these tool selection suggestions, you should consider using separate tools for CI and visualization/orchestration, because these capabilities have different requirements. You should also ensure that versioned, traceable artifacts are the key unit of currency for the deployment pipeline. Avoid simplistic linear sequential pipeline stages where parallel flows would better meet business needs. Finally, insist on a system that tracks, measures, and visualizes the flow of artifacts toward production so that all stakeholders can effectively engage with your software production process.

> *Insist on a system that tracks, measures, and visualizes the flow of artifacts toward production so that all stakeholders can effectively engage with your software production process.*

WRITTEN BY

Matthew Skelton

Matthew Skelton is an independent software consultant who helps organizations to design and operate effective software delivery practices, with a focus on the re-design and evolution of software in order to support Continuous Delivery. He initiated and helped to run PIPELINE 2014, the first conference in Europe dedicated to Continuous Delivery, having co-founded and organized the London Continuous Delivery meetup group since 2012. He is writing two books: one on package management in a Windows environment, and one on software operability.

@matthewpskelton | matthewskelton.net

The **Need for Speed and Quality** *in Release Cycles*

In today's fast-moving, ultra-competitive business landscape, the days of enterprises spending months—or even years—building, testing, and releasing an application, service, or new set of capabilities to the market are over. Today's customers, technology users, and the market as a whole are more nimble than ever before, and the demand for innovative new functionality has grown in concert.

In response to this change in the marketplace, many enterprises have begun focusing on ways they can accelerate their application releases without sacrificing quality and cost. The problem is, IT development and operations teams are often operating in silos and relying on inefficient and error-prone manual deployment methods. The traditional manual methods slow release cycle times and ultimately delay time to revenue.

> *You can think of release automation as the central nervous system of DevOps.*

An enterprise-class Continuous Delivery solution, fully automating the deployment and promotion of changes from development through production, significantly helps enterprises overcome these challenges. You can think of release automation as the central nervous system of DevOps, constantly integrating information and coordinating activities for all parts of the deployment process. Organizations automating the release-deployment process have the opportunity to develop a true DevOps culture, tearing down barriers between development and operations, and speeding communication and collaboration to improve innovation and time to revenue.

CA Technologies has a solution designed to help you along your DevOps journey. CA Release Automation enables customers to automate the delivery process and reduce the complexity of promoting new applications through the development lifecycle and into production. It also provides unique capabilities specifically engineered to enable reuse, segregation of duties, collaboration, and deployment processes. Ultimately these features will help you to achieve your DevOps goals.

WRITTEN BY

Ruston Vickers

SVP, Product Management & Strategy Application Delivery

CA Technologies

CA Release Automation

COMPANY: CA Technologies **ARA**

ca technologies

PHONE: (800) 225-5224	**LOCATION:** Islandia, NY, USA	**LAUNCH DATE:** January '07	@cainc

SUPPORTED TARGET SYSTEMS

- ☑ Windows
- ☑ Linux
- ☐ Mac OS X
- ☑ Solaris
- ☑ AIX
- ☑ Other *(See companion site)*

STRENGTHS

- Action packs and plug-ins for automating and integrating third-party solutions without scripting
- Designed to handle complex, multi-tier distributed solutions on various infrastructures
- Graphical workflow for creating deployment processes
- Defines and manages release activities across application components

NOTABLE CUSTOMERS

- Atkia Bank
- BUPA
- Tesco
- Liquidnet
- Logicalis SMC B.V.
- British United Provident Association LTD.

FREE TRIAL

No free trial

FULL PROFILE LINK dzone.com/r/wkdP

DESCRIPTION

CA Release Automation is an enterprise-class, continuous delivery solution that automates complex, multi-tier release deployments through orchestration and promotion of applications from development through production. CA Release Automation helps companies speed up application release cycles, achieve higher quality and reduce the cost of application deployments, while promoting collaboration and alignment between Development and Operations.

CUSTOMER SUCCESS STORY

A consumer electronics company that manufactures home entertainment devices requires customers to connect to a website through their devices in order to configure them. However, during the busy holiday season, many customers reported that they were unable to use their new devices due to performance issues. After the incident, they learned that their problems were related to their deployment processes, and introduced CA Release Automation in order to standardize deployments across releases. As a result, the website and software are now consistently reliable, earning customer satisfaction and retention.

SUPPORTED CI ENGINES

- Jenkins/Hudson
- JetBrains TeamCity
- Microsoft TFS

CONTINUOUS DELIVERY: *visualized*

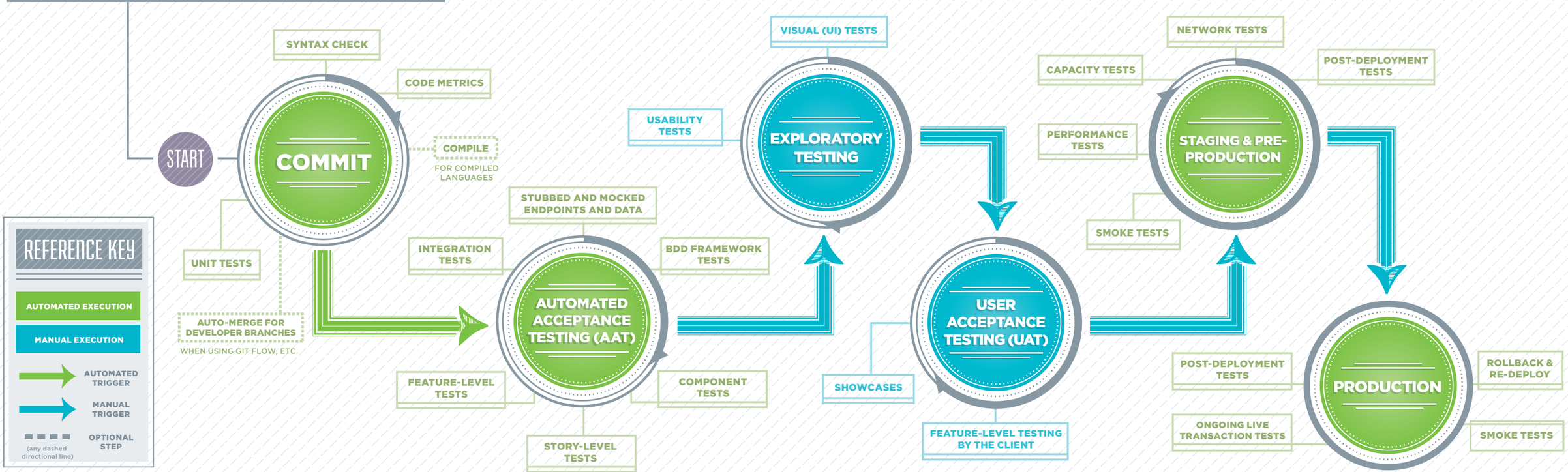
Continuous Delivery advocates the creation of deployment pipelines, which are illustrated below. The first is a long, linear diagram with all of the possible steps, and below that are two common deployment pipelines with parallel-running steps. The parallel (wide) pipelines allow teams to deploy to production sooner or to wait for test results before deploying.

- Any long-running step, such as UAT, Pre-Production testing, or Exploratory Testing, can happen even after the change has already been deployed to Production.
- If significant issues are found in any long-running step, and the change has not been deployed to Production, the team should manually halt the pipeline.
- If significant issues are found in any long-running step, and the change has already been deployed to Production, the team should rollback Production to the last working release.

Diagrams are based on Jez Humble's diagrams from the Continuous Delivery blog (http://continuousdelivery.com/2010/09/deployment-pipeline-anti-patterns)

Special thanks to Matthew Skelton for helping build these diagrams.

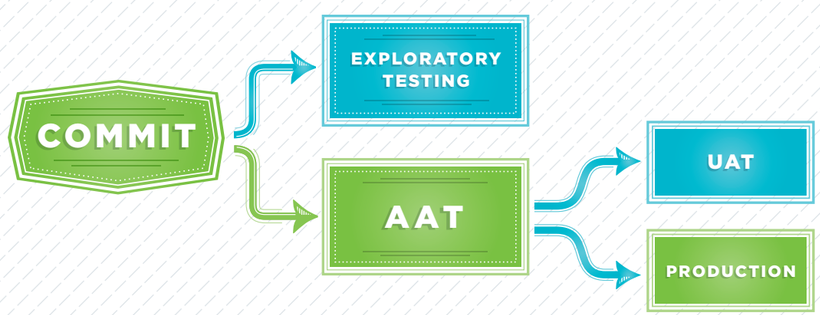
DZone RESEARCH



COMMON DEPLOYMENT PIPELINE #1

CSS CHANGE IN THE UI

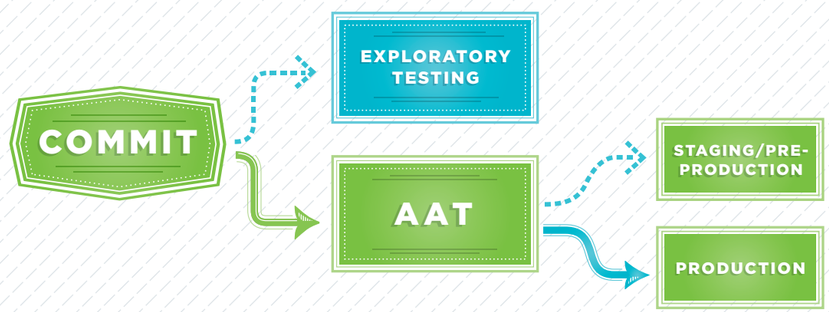
A minor CSS change that could adversely impact a user's ability to use a low-traffic area of the site



COMMON DEPLOYMENT PIPELINE #2

CHANGE IN TRANSACTION PROCESSING ENGINE

A change in the core processing engine with a low chance of affecting performance



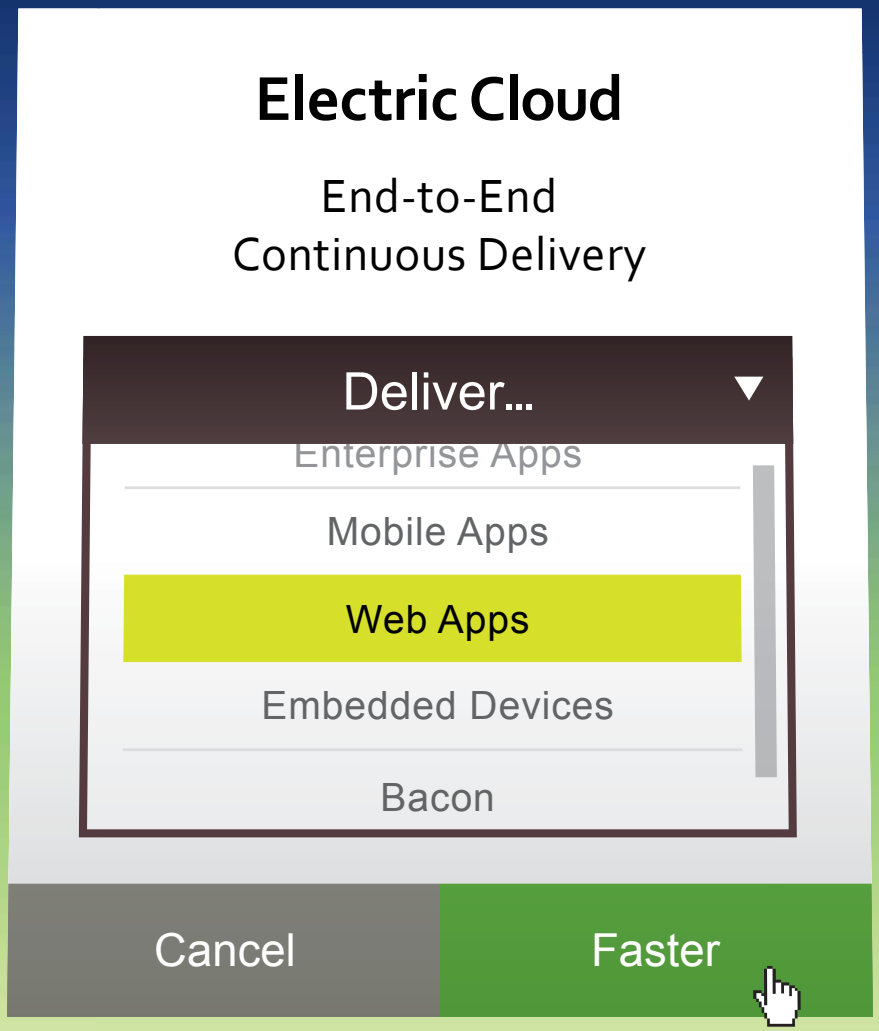
ABOUT DZONE

Continuous Delivery and DevOps are just a couple of the dozens of technology topics covered by DZone. At DZone.com you will find a wide variety of resources including more guides, code cheat sheets (RefCardz), hundreds of articles written by technology experts, and useful links shared by other technology professionals. If you are passionate about technology, then DZone has something to offer you. **Come explore more at:**

DZONE.COM

Electric Cloud

End-to-End
Continuous Delivery



www.electric-cloud.com/cd

Used by leading companies to:

Accelerate builds • Increase developer productivity • Improve product quality

SPACEX DreamWorks Animation SKG splunk> HUAWEI E*TRADE

Accelerate Software Delivery.

www.electric-cloud.com

@electricCloud

electric CLOUD

FROM COMMIT TO CONSUMER:

3 "Must-Haves" for Speedy Delivery

So, you've just checked-in another killer feature. What happens next? Well, that depends. Some of you may be able to build, test, and deploy code to production within minutes. Most of you probably aren't so lucky—it may be hours, days or even weeks before it sees the light of day.

It's not enough to simply automate the software delivery process. To remain competitive, it also needs to go faster.

Here are three ways the right Continuous Delivery (CD) platform keeps you and your team productive and happy:

1. ACCELERATED, ERROR-FREE BUILDS

Agile development creates more frequent builds. A CD platform that optimizes and parallelizes large builds across hundreds, or even thousands of cores is good. A solution that automatically detects dependencies to eliminate broken builds is even better.

2. FASTER FEEDBACK

You shouldn't have to wait until builds or tests are 100% complete before receiving feedback. A CD platform that allows real-time drill-down into warnings and errors as they occur helps eliminate wasted time and CPU cycles.

3. BULLETPROOF AND PAINLESS PROCESSES

CD isn't continuous if jobs in flight are lost when a CI server goes down, or when it takes days for QA machines to be provisioned, or when deployments fail because of differences between QA and production. A CD platform should automate and normalize the build, test, and deploy process across ANY environment (public, private, or hybrid) with one-click simplicity.

CD should enable developers to spend their time developing code, not worrying about infrastructure. Look for a CD platform that can take it one step further to accelerate the end-to-end process. Your code is worth it!

WRITTEN BY

Rohit Jainendra

Chief Product Officer
Electric Cloud

ElectricCommander/ElectricAccelerator

COMPANY: Electric Cloud **CI, ARA**

electric CLOUD

PHONE: (408) 419-4300	**LOCATION:** Sunnyvale, CA, USA	**LAUNCH DATE:** September '06	@electriccloud

SUPPORTED TARGET SYSTEMS

- ☑ Windows
- ☑ Linux
- ☑ Mac OS X
- ☑ Solaris
- ☐ AIX
- ☐ Other *(See companion site)*

STRENGTHS

- Visual authoring of complex multi-path workflows
- Ability to create data-driven dynamic processes
- Dynamic creation of steps, procedures and workflows
- Process-specific user impersonation and fine-grained access control

NOTABLE CUSTOMERS

- Qualcomm
- Cisco
- GE
- HP
- SpaceX
- Intel

FREE TRIAL

30-day free trial

FULL PROFILE LINK dzone.com/r/Tdv6

DESCRIPTION

Electric Cloud helps organizations accelerate the delivery of high-quality software. They automate and parallelize manual build, test, and deploy processes across pools of cloud infrastructure to significantly boost DevOps productivity and throughput. Software-driven companies like Intel, Cisco, Qualcomm, Siemens, HP, GE, SpaceX, GAP, E*TRADE, and GM use Electric Cloud to reduce costs and accelerate innovation.

CUSTOMER SUCCESS STORY

Urban Science is a global automotive retail sales analytics leader, with 20+ customizable products that are deployed to 6 environments over 18 global data centers. Before working with Electric Cloud, Urban Science was using scripts and manual installation instructions customized for each environment, resulting in inconsistent environments, slow deployment times, and poor quality. Using ElectricCommander, Urban Science was able to standardize deployment procedures, automate every step of their deployment workflow, optimize their use of IT resources, and gain security and visibility which they previously did not have.

HOSTING OPTIONS

- On-premise and cloud hosted

Infrastructure as Code:
WHEN AUTOMATION ISN'T ENOUGH

BY MITCH PRONSCHINSKE

Some of the best ideas in the tech industry, many of which have revolutionized the way organizations build software, have come from the cross-pollination of strategies and best practices that occurs when all teams from the organization leave their silos and work together on their most difficult problems. It's widely understood that progress often comes from the sharing of ideas, so it's no surprise that the fundamental principles behind DevOps are widely accepted as beneficial practices.

Agile workflow methodologies were embraced early on by the development community, and today they are widely shared with business and operations teams. Continuous Delivery is an expansion of the concepts behind continuous integration. Continuous integration started as a development strategy and expanded to encompass production deployments, which meant bringing the operations team into the mix. When the principles of DevOps came into play, the concept of Continuous Delivery emerged.

One of the most powerful innovations spawned by the DevOps movement was the idea that operations departments could manage the configurations of servers and virtual machines the same way that developers manage the configurations in their software's source code. This doesn't simply mean command line scripts, which sysadmins were already familiar with. It means using a description language or full-fledged programming language to manipulate infrastructure in an automated, repeatable way.

INFRASTRUCTURE AS CODE

With the arrival of tools like CFengine, Puppet, and Chef, the concept of *Infrastructure as Code* was born. These tools enable developers and sysadmins to abstract their problems around maintaining modular, automatable infrastructure, and being able to define them using a high-level language. Once teams started using these tools and techniques, building and maintaining their server environments began to closely resemble the way that software developers build and maintain application source code.

All of the testability, repeatability, and transparency of a modern software development process suddenly became available for people who were building infrastructure. The lofty goal for many of these cutting edge organizations was to be able to completely rebuild a business' software systems with nothing more than physical server resources, a complete backup of their databases, and source code. Today, that's how many modern successful tech businesses operate. They have the ability to do exactly that.

INFRASTRUCTURE AUTOMATION AT SCALE

Infrastructure automation is the process of automatically running a set of custom actions to install an operating system on a virtual or physical server and then configure it according to specifications. When virtualization and cloud infrastructure became widely available, the new IT strategies that emerged were only achievable through some form of infrastructure automation. The ability to spin up new virtual machines in minutes meant that a manual installation with step-by-step instructions for the sysadmin wasn't going to cut it anymore. Those previously manual steps would now need to be handled by software in conjunction with the creation of the VMs. Otherwise, the full value of cloud and virtualization strategies wouldn't be realized.

However, infrastructure automation brings its own set of challenges. Any form of automation can cause a host of new issues if it is not carefully managed. Server image templates and copies (clones) are the first step in handling the scale of cloud and virtualization management, but if those templates or copies have any instructions that will cause problems, bad configurations get copied again and again, propagating the issue to a large number of servers. Automating correct infrastructure settings on a large scale can be a powerful ability, but it can be equally devastating if automation instructions are not composed carefully.

There are also dangers that come with the ease and speed of growing an organization's infrastructure portfolio when they have automation in place. It's very difficult with the basic automation that many organizations use to keep up with an infrastructure that is constantly growing and changing. The biggest problem that infrastructure automation can cause is *Configuration Drift*.

From the Continuous Delivery Report glossary:

CONFIGURATION DRIFT: *A term for the general tendency of software and hardware configurations to "drift," or become inconsistent with the baseline or template version of the system due to manual ad hoc changes (like hotfixes) that are not introduced back into the template.*

Many of today's deployment problems and IT outages are caused by Configuration Drift. Configuration Drift isn't limited to infrastructure configuration, either. Middleware settings also experience a great deal of untracked changes in many organizations. This causes numerous Configuration Drift-related errors, especially in testing and acceptance environments. Disaster recovery and high availability failures are also common results of Configuration Drift. Having a feature in your deployment automation solution, whether it's custom or provided by a product vendor, is highly recommended. Detecting configuration drift is the first step to overcoming it.

It's important to note that infrastructure automation is not the same as Infrastructure as Code. Infrastructure automation can take repeatable actions and enact them across any number of servers, but Infrastructure as Code includes tools and capabilities from the software development world to build, maintain, and test those actions in a more manageable way.

DEVELOPER LESSONS FOR INFRASTRUCTURE AS CODE

The biggest danger when trying to integrate an Infrastructure as Code-enabling configuration management tool like Chef, SaltStack, or Ansible in your software production process is not applying the same lessons learned by the development world in the last ten years. Infrastructure code needs to be version controlled, tested, maintained, quickly deployed, and crafted for easy usability.

Development principles that date all the way back to the beginning of Extreme Programming will provide the best guidance for making Infrastructure as Code into a blessing rather than a curse. These are the principles that organizations who are new to Infrastructure as Code should focus on:

- Include infrastructure code in the software development lifecycle by putting it through build, testing/QA, and production along with application code.
- Start with and maintain a simple design for your infrastructure code and any new features. Use design patterns.
- Have the infrastructure team agree on the design and direction of the system.
- Institute code reviews with the help of team notifications for each commit. Pair programming is also helpful.
- Maintain a shared set of coding standards for the infrastructure code.
- Continuously test with as many of the same styles of development testing that could be applicable to Infrastructure as Code. This ensures that the code definitions produce the correct environments and don't introduce problems.
- Employ infrastructure monitoring for testing and pre-production environments as well as the production environment.
- Refactor the infrastructure code to reduce the amount of maintenance needed.

TOOLING EXPECTATIONS

Organizations don't want to waste their time or money using tools that don't provide the features needed to avoid the biggest pitfalls in Continuous Delivery. Beware of tools that let you easily create endless server templates but don't allow proper traceability for template changes like the inevitable quick fixes that are prevalent in software production. The software should be able to externalize infrastructure code definitions that can be stored in common version control systems. Organizations should be wary of tools that don't allow the usage of community tools that benefit from large community knowledgebases. It's better to use tools that are flexible and allow you to produce future-proof systems that can take advantage of the many innovations in the wider development tooling community.

> *One of the most powerful innovations spawned by the DevOps movement was the idea that operations departments could manage the configurations of servers and virtual machines the same way that developers manage the configurations in their software's source code.*

Infrastructure as Code emerged from the development community through open source configuration management tools that were shared because so many organizations were having the same problems when using basic, non-programmatic infrastructure automation. The best piece of advice to internalize about using Infrastructure as Code is to stay connected to the vast community of innovative developers. If your systems remain open to the rapid changes in that community, you'll be able to share and benefit from the cutting-edge ideas that will make your organization successful.

WRITTEN BY

Mitch Pronschinske

Mitch Pronschinske is the Head Analyst for DZone's Research division. He has been writing, curating, and editing content for an audience of IT professionals for over four years. In that time he has learned the complexity that software producers deal with on a daily basis, and he strives to make their world easier to understand and digest.

CONTINUOUS DELIVERY MATURITY CHECKLIST

Check the boxes next to the practices you currently perform to see your maturity in each area of Continuous Delivery.
Add up your score at the end based on the highest levels you checked.

SOURCE CONTROL

BASELINE
- ☐ Early branching
- ☐ Branches tend to remain apart

NOVICE
- ☐ Branches are used for isolating work
- ☐ Merges are common

INTERMEDIATE
- ☐ Pre-tested commits
- ☐ Integration branch is pristine

ADVANCED
- ☐ All commits are tied to tasks
- ☐ History used to rewrite features before pushing to central repository
- ☐ Version control DB schema changes

EXPERT
- ☐ Traceability analysis and release notes auto-generated
- ☐ Commits are clean enough for the master branch/trunk

BUILD PROCESS

BASELINE
- ☐ Official builds are not performed on developers' machines
- ☐ Self-service build or nightly build

NOVICE
- ☐ System polls source control and builds on commit
- ☐ Build artifacts are managed, some manual scripts still used

INTERMEDIATE
- ☐ Build artifacts are managed by purpose-built tools, no manual scripts
- ☐ Dependencies are managed in a repository

ADVANCED
- ☐ Distributed builds on build cluster, can be done in sequence
- ☐ Source control tells system when to build, no polling

EXPERT
- ☐ Build environments based on VMs
- ☐ Streams are never "broken"
- ☐ Gated commits

TESTING & QA

BASELINE
- ☐ Automatic unit testing with every build
- ☐ Code coverage is measured

NOVICE
- ☐ Peer-reviews
- ☐ Mockups & proxies used

INTERMEDIATE
- ☐ Periodic static code analysis
- ☐ Automated functional testing

ADVANCED
- ☐ Integrated management and maintenance of the test data
- ☐ Automated performance & security tests in target environments

EXPERT
- ☐ Automated acceptance testing

DEPLOYMENT

BASELINE
- ☐ Fully scripted deployments

NOVICE
- ☐ Push-button deployments to test environments

INTERMEDIATE
- ☐ Auto deploy to first test environment
- ☐ Standard deployments across all environments
- ☐ Push-button deployments to production

ADVANCED
- ☐ Automated deployments after tests pass
- ☐ Database deployments
- ☐ Multi-tier deployments

EXPERT
- ☐ Ability to implement continuous deployment

VISIBILITY

BASELINE
- ☐ Build status notification is sent to committer

NOVICE
- ☐ Latest build status is available to all team members

INTERMEDIATE
- ☐ Trend reports are automatically generated from build server events
- ☐ People outside the team can subscribe to build statuses

ADVANCED
- ☐ Stakeholders have dashboards with real-time product and dependency stats

EXPERT
- ☐ Cross-team data mining and analysis

OVERALL MATURITY SCORECARD

For each check mark add the assigned number of points and total them for each section:

TALLY YOUR SCORES:

POINT KEY:
- **BASELINE** 0 points
- **NOVICE** 1 point
- **INTERMEDIATE** 2 points
- **ADVANCED** 3 points
- **EXPERT** 4 points

SOURCE CONTROL	
BUILD PROCESS	
TESTING & QA	
DEPLOYMENT	
VISIBILITY	
TOTAL	

0-7 ADEQUATE **8-11** AVERAGE **12-14** SKILLED
15-17 ADEPT **18-20** MASTER

Inspired by Chris Shayan and Eric Minick

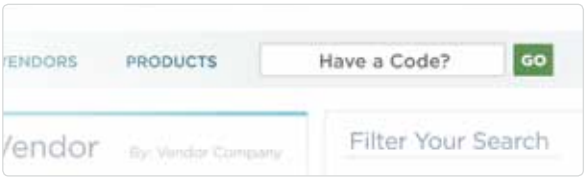
SOLUTIONS **DIRECTORY**

In the following Continuous Delivery Solutions Directory, you will find a side-by-side comparison of the many solutions available to build out your continuous delivery toolchain. Each solution profile will give you a glimpse of the essential set of criteria for categorizing and assessing each tool.

Along with a basic snapshot, each profile also includes a DZone companion site "shortcode" URL that looks like this:

By visiting this URL, you will be taken to an expanded profile for the given solution where you can explore a much more comprehensive set of features and details. In addition to the ability to see expanded profiles, the companion site also gives you the ability to run a side-by-side comparison of selected solutions by your chosen features and criteria.

Alternatively, you can visit http://dzone.com/research/continuousdelivery and enter any shortcode in the search bar to find your selected vendor.



Ansible Open Source

CM

PRODUCT LAUNCH: Feb-12 | **TWITTER:** @ansible | **LOCATION:** Santa Barbara, CA, USA

SUPPORTED TARGET SYSTEMS:

- ☐ Windows
- ☑ Linux
- ☑ Mac OS X
- ☑ Solaris
- ☑ AIX
- ☑ Other *(See companion site)*

STRENGTHS

- Free and open source
- Agentless model reduces complexity of managing systems
- Self-documenting YAML-based automation language
- All user-submitted and officially tested modules packaged with Ansible

AGENT MODEL

- Agentless

FULL PROFILE LINK

dzone.com/r/9szr

DESCRIPTION

Ansible is an open source IT configuration management engine, designed to be minimal and lightweight. Ansible models infrastructure by looking at how all systems inter-relate, and manages systems parallel to each other. To further its goal of minimal design and requirements, Ansible does not use remote agents. It also uses SSH to ensure system security, removing complexities around upgrading software and security patches.

FREE TRIAL

Open source

Ansible Tower

CM

PRODUCT LAUNCH: Jul-2013 | **TWITTER:** @ansible | **LOCATION:** Santa Barbara, CA, USA

SUPPORTED TARGET SYSTEMS:

- ☐ Windows
- ☑ Linux
- ☑ Mac OS X
- ☑ Solaris
- ☑ AIX
- ☑ Other *(See companion site)*

STRENGTHS

- Self-documenting YAML-based automation language
- Automates deployment, config management, provisioning, and orchestration
- Pulls cloud inventories automatically and assists auto-scaling scenarios
- Full-REST API browser shows off every resource

AGENT MODEL

- Agentless

FULL PROFILE LINK

dzone.com/r/tRUJ

DESCRIPTION

Ansible is an IT orchestration engine that models your infrastructure by looking at how all of your systems inter-relate, rather than just managing one system at a time. Ansible Tower is the browser-based Ansible UI, available as a commercial product. It features push-button automation, LDAP integration, scheduling, and a full REST API. Tower also features out-of-the-box integration with AWS and Rackspace.

FREE TRIAL

Free up to ten nodes

Atlassian Bamboo

CI

PRODUCT LAUNCH: Feb-07 | **TWITTER:** @Atlassian | **LOCATION:** San Francisco, CA, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [ ] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Build results and deployment status inside JIRA issues
- Automatically applies your CI scheme to new branches in your repository
- Deployment automation from master or dev branches
- Easy parallelization of build steps

HOSTING OPTIONS

- On-premise and cloud hosted

FULL PROFILE LINK

dzone.com/r/4Lks

DESCRIPTION

Bamboo has all the capabilities of a CI server by running builds and tests. It also connects issues, commits, test results, and deploys using other pieces of the Atlassian tool suite, such as JIRA. This gives the entire project team, including developers, project managers, testers, and sysadmins, wide visibility into the entire project from commit to production. To support Continuous Delivery, Bamboo can automate the entire delivery pipeline.

FREE TRIAL

30-day full-feature trial

Attunity RepliWeb

ARA

PRODUCT LAUNCH: Jan-07 | **TWITTER:** @attunity | **LOCATION:** Burlington, MA, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [ ] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Accelerated large-file deployments across WANs and LANs
- Ad hoc and programmatic package deployments
- Defines multi-tiered, multi-stage deployment topologies
- Provides lines-of-business with job execution and visibility into deployment

SUPPORTED CI ENGINES

- Microsoft TFS

FULL PROFILE LINK

dzone.com/r/7Muh

DESCRIPTION

RepliWeb is a deployment platform for applications and large-file content. The platform addresses a wide range of deployment, replication and data transfer scenarios for operations, developers, content creators, and business continuity teams. RepliWeb includes a template deployment processes, customized UIs with role-based access, proprietary transfer engines, comprehensive monitoring and reporting, and full rollback across server and application states.

FREE TRIAL

15-day free trial with approved POC definitions

Automic Deployment Manager

RESEARCH PARTNER

ARA

PRODUCT LAUNCH: Jan-10 | **TWITTER:** @automic | **LOCATION:** Wolfsgraben, Austria, EU

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [x] AIX
- [x] Other *(See companion site)*

STRENGTHS

- Use the same workflow in every environment (Dev - QA - UAT - Production)
- Automatic rollback
- Deployments done with managed file transfer

SUPPORTED CI ENGINES

- Jenkins/Hudson
- Atlassian Bamboo
- JetBrains TeamCity
- Microsoft TFS

FULL PROFILE LINK

dzone.com/r/RdL7

DESCRIPTION

Automic's ONE Automation Platform provides a centralized platform for developers, QA, and operations to coordinate and automate the entire application release. Deployments and releases can be coordinated with other critical business processes such as workload, batch, or approval, ensuring that releases do not have any negative impact on business as usual.

FREE TRIAL

7-day free trial

BMC Release Lifecycle Management

ARA

PRODUCT LAUNCH: Oct-11 | **TWITTER:** @BMC_DevOps | **LOCATION:** Houston, TX, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [ ] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Includes planning, coordination, and automation execution
- Compatible with many existing prominent open source and commercial products
- Trend-spotting features for the release process in visualization interface

SUPPORTED CI ENGINES

- Jenkins/Hudson
- Atlassian Bamboo
- JetBrains TeamCity
- Microsoft TFS

FULL PROFILE LINK

dzone.com/r/YQWw

DESCRIPTION

BMC Release Lifecycle Management is a process flow and application deployment solution that includes technology from Varalogix, a deployment automation firm that BMC acquired in 2012. The broad ARA solution now features a centralized management console and a release repository for archiving application components, along with a drag-and-drop web UI for building multi-tier application deployment templates.

FREE TRIAL

No free trial

CA Release Automation

RESEARCH PARTNER

ARA

PRODUCT LAUNCH: Jan-07 | **TWITTER:** @cainc | **LOCATION:** Islandia, NY, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [ ] Mac OS X
- [x] Solaris
- [x] AIX
- [x] Other *(See companion site)*

STRENGTHS

- Plug-ins for automating and integrating third-party solutions without scripting
- Application-centric, manifest-driven deployment provides reusable and repeatable processes to simplify application releases.
- Graphical workflow for creating deployment processes
- Defines and manages release activities across application components

SUPPORTED CI ENGINES

- Jenkins/Hudson
- JetBrains TeamCity
- Microsoft TFS

FULL PROFILE LINK

dzone.com/r/wkdP

DESCRIPTION

CA Release Automation is an enterprise-class continuous delivery orchestration solution to design, test, and automate complex application release deployments with visibility into the entire process for all stakeholders. Integrating technologies acquired from Nolio, the CA LISA platform helps enterprises simplify and standardize the application release processes, increasing the speed of application release cycles while improving software quality.

FREE TRIAL

No free trial

CFEngine Community

CM

PRODUCT LAUNCH: Jan-94 | **TWITTER:** @cfengine | **LOCATION:** Oslo, Norway, EU

SUPPORTED TARGET SYSTEMS:

- [ ] Windows
- [x] Linux
- [ ] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Free and open source under the GNU GPL license
- Designed to scale
- Continuously monitors and self-repairs IT systems
- Able to use any integrated policy editor

AGENT MODEL

- Agents Installed

FULL PROFILE LINK

dzone.com/r/hQyX

DESCRIPTION

CFEngine Community is an open source infrastructure automation framework, originally developed out of an academic setting. CFEngine helps engineers, system administrators, and other stakeholders in an IT organization manage and understand IT infrastructure throughout its lifecycle. Solutions running CFEngine use decentralized, autonomous agents to scale and automatically configure IT systems. CFEngine was first built in 1993, well before its present competitors in the CM space.

FREE TRIAL

Open source

CFEngine Enterprise

CM

PRODUCT LAUNCH: Jan-94 | **TWITTER:** @cfengine | **LOCATION:** Oslo, Norway, EU

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [ ] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Designed to scale, with up to 5,000 hosts per server hub
- Perfect security record based on nist.gov standards
- "Sketches" help users perform sysadmin tasks without in-depth knowledge of the product
- CFEngine AS provides training and consulting

AGENT MODEL

- Agents Installed

FULL PROFILE LINK

dzone.com/r/ALtU

DESCRIPTION

CFEngine is an infrastructure automation framework that helps engineers, system administrators and other stakeholders in an IT organization manage and understand IT infrastructure throughout its lifecycle. It takes systems from build to deploy, manage, and audit. CFEngine was first built in 1993, well before its present competitorsin the CM space. In contrast to its competitors, it originates from an academic setting rather than a commercial one.

FREE TRIAL

Free trial limited to 25 managed hosts. Community edition also available.

CircleCI

CI, ARA

PRODUCT LAUNCH: Jul-12 | **TWITTER:** @circleci | **LOCATION:** San Francisco, CA, USA

SUPPORTED TARGET SYSTEMS:

- [ ] Windows
- [x] Linux
- [ ] Mac OS X
- [ ] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Build machines are managed as part of the product
- Automatic test parallelization
- SSH access to builds
- Built-in artifact management
- Tight GitHub integration
- Runs virtual machines with user's choice of database.

SUPPORTED CI ENGINES

- CircleCI

FULL PROFILE LINK

dzone.com/r/uNhy

DESCRIPTION

CircleCI is a hosted continuous integration and deployment platform that can infer how to run CI and CD from the details of a user's code based on the first test. It supports web applications on a Linux stack, including Ruby, Python, Node, PHP, and Java applications. This includes optimized build speed, automatic test parallelization, SSH access to builds, built-in artifact management, tight GitHub integration, and instant setup of new repositories.

FREE TRIAL

2 weeks unlimited

CloudBees Continuous Delivery Platform

RESEARCH PARTNER

CI

PRODUCT LAUNCH: Jan-11 | **TWITTER:** @CloudBees | **LOCATION:** Woburn, MA, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Jenkins-based Continuous Delivery lifecycle
- On premise, public cloud, or hybrid cloud configuration and hosting options
- Hosted iOS and Android builds for mobile Continuous Delivery
- Manage multiple Jenkins configurations across the enterprise

HOSTING OPTIONS

- On premise, public cloud, or hybrid cloud

FULL PROFILE LINK

dzone.com/r/aVAt

DESCRIPTION

The CloudBees Continuous Delivery Platform provides a range of Jenkins-based Continuous Integration and Continuous Delivery solutions on premise, in the cloud, or in a hybrid configuration. The CloudBees Platform offers solutions for both development and DevOps teams to utilize Jenkins and Platform as a Service technology in a configuration that best meets their needs. The CloudBees CD Platform can also be extended to work with technologies across the application lifecycle.

FREE TRIAL

30-day free trial

Codeship

CI

PRODUCT LAUNCH: May-11 | **TWITTER:** @codeship | **LOCATION:** Cambridge, MA, USA

SUPPORTED TARGET SYSTEMS:

- [ ] Windows
- [x] Linux
- [ ] Mac OS X
- [ ] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Integration with GitHub and BitBucket
- Web based configuration
- Completely hosted solution
- Build pipelines for deployment

HOSTING OPTIONS

- Cloud hosted

FULL PROFILE LINK

dzone.com/r/CWKT

DESCRIPTION

Codeship is a hosted Continuous Integration and deployment service. Users can connect their Github or BitBucket repositories and Codeship will automatically run the user's tests when they push new commits. Upon successful completion of the tests, users can also deploy to staging or production systems, so their applications are always up to date with the latest changes.

FREE TRIAL

50 builds per month for free

Drone.io

CI

PRODUCT LAUNCH: Feb-14 | **TWITTER:** @droneio | **LOCATION:** San Francisco, CA, USA

SUPPORTED TARGET SYSTEMS:

- [ ] Windows
- [x] Linux
- [ ] Mac OS X
- [ ] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Single binary file installation with only Docker as a dependency
- Drone is open source
- Dashboard optimized for large monitors or TV screens
- Drone CLI for local builds that run like cloud-based builds

HOSTING OPTIONS

- On-premise and cloud hosted

FULL PROFILE LINK

dzone.com/r/JvfV

DESCRIPTION

Drone is a Continuous Integration solution built on Docker, an open source tool to pack, ship, and run any application as a lightweight container. Every build is virtualized in a Docker container. Drone provides dozens of pre-built Docker images for nearly every popular language and database, eliminating the need to setup and configure your build environment. Drone is free and open source with a cloud-hosted offering available.

FREE TRIAL

Open Source

ElectricCommander ElectricAccelerator

RESEARCH PARTNER

CI, ARA

PRODUCT LAUNCH: Sep-06 | **TWITTER:** @electriccloud | **LOCATION:** Sunnyvale, CA, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Visual authoring of complex multi-path workflows
- Ability to create data-driven dynamic processes
- Dynamic creation of steps, procedures and workflows
- Process-specific user impersonation and fine-grained access control

SUPPORTED CI ENGINES

- Jenkins/Hudson
- Microsoft TFS

FULL PROFILE LINK

dzone.com/r/Tdv6

DESCRIPTION

Electric Cloud helps organizations accelerate the delivery of high-quality software. They automate and parallelize manual build, test, and deploy processes across pools of cloud infrastructure to significantly boost DevOps productivity and throughput. Software-driven companies like Intel, Cisco, Qualcomm, Siemens, HP, GE, SpaceX, GAP, E*TRADE, and GM use Electric Cloud to reduce costs and accelerate innovation.

FREE TRIAL

30-day free trial

Enterprise Chef

CM

PRODUCT LAUNCH: Jun-11 | **TWITTER:** @chef | **LOCATION:** Seattle, WA, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Pushes workloads to the edges of the network
- Efficiently uses server resources and bandwidth
- RBAC, multi-tenant support, and cryptographic validation for clients and servers
- Out-of-box integrations with popular tools and programmatic extensions for other tools

AGENT MODEL

- Agents Installed

FULL PROFILE LINK

dzone.com/r/rfNp

DESCRIPTION

Enterprise Chef models IT infrastructure and application delivery as code, providing an integrated tool chain and automation platform. Ruby on Rails developers will find Chef's syntax especially familiar, as will many users with programming backgrounds. Enterprise Chef enables rapid provisioning and deployment of IT resources and the automated delivery of applications and services at massive scale.

FREE TRIAL

Open source version available

EpicForce Leroy

ARA

PRODUCT LAUNCH: Dec-13 | **TWITTER:** @LeroyDeploy | **LOCATION:** Carrboro, NC, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Agents compiled for each platform
- Multi-credential agent model without having to store user credentials
- Agents contain an embedded Python scripting environment
- Application and deployment engine configs are in one scope

SUPPORTED CI ENGINES

- Jenkins/Hudson
- JetBrains TeamCity

FULL PROFILE LINK

dzone.com/r/47ks

DESCRIPTION

Leroy is an application deployment automation engine, written in C++, that is free to use. It allows engineers to describe the entire deployment, properties, and application configs using a simple XML format. This allows the application deployment to be treated like a versioned component in the repository, like the rest of the code. Leroy is relatively new, but it is being used in production environments at companies like Fleetmatics.

FREE TRIAL

Free solution

Hudson Continuous Integration Server

CI

PRODUCT LAUNCH: May-06 | **TWITTER:** @hudsonci | **LOCATION:** Ottawa, Canada

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Self-contained easy to install by running the WAR file
- Massive plugin ecosystem
- Architected to meet the needs of enterprise scale development
- Java-based, runs on a wide range of environments

HOSTING OPTIONS

- On-premise and cloud hosted

FULL PROFILE LINK

dzone.com/r/3hjG

DESCRIPTION

Hudson is an open source Continuous Integration (CI) server, hosted by the Eclipse Foundation. Hudson's job is to automate the process of building, testing and deploying software over a wide range of technologies and platforms in an open and extensible manner. Hudson has a low barrier to entry and is simpler to set up and run than the equivalent manual processes. The Jenkins CI engine is a fork of the Hudson project.

FREE TRIAL

Open Source

IBM UrbanCode Deploy

ARA

PRODUCT LAUNCH: Sep-11 | **TWITTER:** @UrbanCode | **LOCATION:** Armonk, NY, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [x] AIX
- [x] Other *(See companion site)*

STRENGTHS

- Direct support for mobile, distributed, and mainframe components
- Builds, database changes, config, etc bundled into snapshots for consistent testing
- Support for true active-active horizontal scalability and failover
- Tracks any discrepancies that could cause configuration drift

SUPPORTED CI ENGINES

- Jenkins/Hudson
- JetBrains TeamCity
- Microsoft TFS
- Rational Team Concert
- AnthillPro
- Rational Build Forge

FULL PROFILE LINK

dzone.com/r/sh3j

DESCRIPTION

IBM UrbanCode Deploy (formerly uDeploy) is an application deployment automation solution to deploy and test new builds. It integrates with many CI and test tools to drive rapid feedback, and is designed to facilitate collaboration in agile development environments. UrbanCode Deploy also provides audit trails, versioning, and approvals. The tool also tracks which components make up an application so they can be deployed and monitored together.

FREE TRIAL

30-day free trial

Inedo BuildMaster

ARA

PRODUCT LAUNCH: Oct-09 | **TWITTER:** @inedo | **LOCATION:** Berea, OH, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Designed for use by entire team
- Flexibility & Extensibility
- Growing integrations library
- Can manage from source control through production
- Web UI allows users to set up as many deployment plans as they need

SUPPORTED CI ENGINES

- Jenkins/Hudson
- JetBrains TeamCity
- Microsoft TFS

FULL PROFILE LINK

dzone.com/r/bWCK

DESCRIPTION

Inedo BuildMaster utilizes approval and promotion requirements to efficiently automate deployment processes from source control through production. It uses automatic builds to create builds and run unit tests at specific times, trigger builds via source control activity, ping a URL using another tool, or integrate with an existing CI system. Users can see the state of all planned and past releases, with detailed historical data and auditing.

FREE TRIAL

45-day Full-Featured Enterprise Trial or Non-expiring Feature-limited Express Edition (5 users, 10 application, 5 servers)

JetBrains TeamCity

CI

PRODUCT LAUNCH: Oct-06 | **TWITTER:** @teamcity | **LOCATION:** Czech Republic, EU

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Reporting determines new failures and assigns investigations
- Support for non-linear build pipelines
- Reuse settings with templates, hierarchical projects, and parameter references
- Bundled with other JetBrains tools like IntelliJ IDEA

HOSTING OPTIONS

- On-premise

FULL PROFILE LINK

dzone.com/r/dJPM

DESCRIPTION

TeamCity is a continuous integration and build server. It features out-of-the-box integration with many version control systems and it has dedicated "no-configuration" integration with many build tools and testing frameworks in Java, .NET, and Ruby. TeamCity also features reporting both in the web browser and IDE plugins.

FREE TRIAL

Free professional version. Can build up to 20 different builds. Some plugins are open source.

MidVision RapidDeploy

ARA

PRODUCT LAUNCH: Jun-08 | **TWITTER:** @RapidDeployTM | **LOCATION:** London, UK, EU

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Agent and agentless deployments
- File & API based deployments to support cloud and legacy platforms
- Broad range of support out-of-the-box for third-party tools
- Supports several compliance and audit capabilities

SUPPORTED CI ENGINES

- Jenkins/Hudson
- CruiseControl
- RTC

FULL PROFILE LINK

dzone.com/r/ytX9

DESCRIPTION

RapidDeploy is an application release automation originally created for a large financial institution. It features integrations with many popular source control, build, and repository tools, and has support for many middleware, messaging, and database solutions. It provides self-service deployment capabilities as well. Integrations are available for monitoring and ITSM, as well as image-based provisioning and cloud providers.

FREE TRIAL

5 nodes free forever or unlimited 30 day free trial.

Octopus Deploy

ARA

PRODUCT LAUNCH: Jul-12 | **TWITTER:** @octopusdeploy | **LOCATION:** Australia

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [ ] Linux
- [ ] Mac OS X
- [ ] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Built-in conventions for .NET configuration settings and transforms
- Conventions for IIS configuration and Windows Service deployment
- Encrypted database
- Secure variable management for storing sensitive settings
- Comprehensive REST API

SUPPORTED CI ENGINES

- JetBrains TeamCity
- Microsoft TFS

FULL PROFILE LINK

dzone.com/r/KdTv

DESCRIPTION

Octopus Deploy is a deployment automation system built for .NET developers. Octopus works with a build server to enable secure, automated releases of ASP.NET applications and Windows Services into test, staging and production environments, whether they are in the cloud or on-premises. Red Gate's Deployment Manager is based on a copy of the Octopus Deploy codebase.

FREE TRIAL

All features free for 45 days. Free version available for 5 projects or 10 agents.

Open Source Chef

CM

PRODUCT LAUNCH: Jan-09 | **TWITTER:** @chef | **LOCATION:** Seattle, WA, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Free and open source
- Integrations with Amazon EC2 and Rackspace
- Able to support multiple environments
- Chef servers can manage over 10,000 nodes

AGENT MODEL

- Agents Installed

FULL PROFILE LINK

dzone.com/r/RrL7

DESCRIPTION

Open Source Chef is an open-source configuration management tool, and is the basis of Chef Enterprise. Chef allows users to manage the infrastructure behind applications by automatically configuring each resource to the desired state. Users can also create QA and pre-production environments.

FREE TRIAL

Open source

Puppet Enterprise

CM

PRODUCT LAUNCH: Feb-11 | **TWITTER:** @puppetlabs | **LOCATION:** Portland, OR, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [ ] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- DSL that is easy to learn and use for operations teams and developers
- A secure declarative model that enables true simulation runs
- Tight integration with VMware, Cisco, Microsoft, and others.
- Large, active community of users and contributors

AGENT MODEL

- Agents Installed

FULL PROFILE LINK

dzone.com/r/j9GR

DESCRIPTION

Puppet Enterprise is an IT automation product that gives system administrators the ability to automate repetitive tasks, deploy critical applications, and proactively manage infrastructure on premise or in the cloud. Puppet Enterprise automates tasks at any stage of the IT infrastructure lifecycle, including: discovery, provisioning, OS and application configuration management, orchestration, and reporting.

FREE TRIAL

Free up to ten nodes, Open source project available

Puppet Open Source

CM

PRODUCT LAUNCH: Jan-05 | **TWITTER:** @puppetlabs | **LOCATION:** Portland, OR, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [ ] Mac OS X
- [ ] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Free and open source under Apache 2.0 license
- Large, active community of users and contributors
- DSL that is easy to learn and use for both operations teams and developers
- A secure declarative model that enables true simulation runs

AGENT MODEL

- Agents Installed

FULL PROFILE LINK

dzone.com/r/TMv6

DESCRIPTION

Puppet Open Source is an IT automation product under the Apache 2.0 license that gives system administrators the ability to automate repetitive tasks using a declarative, model-based approach to IT automation. Puppet Open Source automates tasks at many stages of the IT infrastructure lifecycle, including: provisioning, OS and application configuration management, and orchestration.

FREE TRIAL

Open source

Rocket ALM Hub

ARA

PRODUCT LAUNCH: Jan-01 | **TWITTER:** @rocket | **LOCATION:** Waltham, MA, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [ ] Mac OS X
- [ ] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Multi-platform management
- Point and click process setup
- Integrated issue tracking
- Integrated service desk
- Automated promotion, build, and deployment

SUPPORTED CI ENGINES

- Jenkins/Hudson

FULL PROFILE LINK

dzone.com/r/yaX9

DESCRIPTION

The Rocket ALM Hub is an end-to-end solution that provides multi-platform application lifecycle management automation for deployment automation. Rocket focuses on organizations managing multiple releases, ensuring applications are deployed to the correct locations with the most up-to-date builds. The Hub provides automation, visibility to everyone, security, and repeatable process creation. The software also includes reporting and IT compliance.

FREE TRIAL

Free trials are limited by time

Rundeck

ARA

PRODUCT LAUNCH: Jul-11 | **TWITTER:** @rundeck | **LOCATION:** Redwood City, CA, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Designed for automating all ongoing operational tasks
- Fully open source software project
- Extensible through first and third-party plugin systems
- Built for enterprise scale concerns, including security and auditing

SUPPORTED CI ENGINES

- Jenkins/Hudson
- Atlassian Bamboo

FULL PROFILE LINK

dzone.com/r/HGQ4

DESCRIPTION

Rundeck is an open source utility designed to help operations teams automate routine operational procedures and deployment. Teams can collaborate through Rundeck to share how processes are automated, delegate execution of those processes, and share visibility into operational activity. Rundeck also includes access control, workflow building, scheduling, logging, notifications, and plugins for integration with external data sources and tools.

FREE TRIAL

Open source software

SaltStack Enterprise

CM

PRODUCT LAUNCH: Feb-11 | **TWITTER:** @saltstackinc | **LOCATION:** Salt Lake City, UT, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- High-speed, always-on communication bus
- Instantly react to events spanning networks, data centers, or continents
- Infrastructure-wide deployment and orchestration
- Self-healing capabilities
- Imperative or declarative model

AGENT MODEL

- Can Install Agents or Go Agentless

FULL PROFILE LINK

dzone.com/r/jxGR

DESCRIPTION

SaltStack Enterprise delivers an infrastructure communication bus for real-time infrastructure automation, cloud provisioning, and orchestration, as well as application stack configuration management. SaltStack is known for its speed and its ability to scale up to support environments with tens of thousands of systems, unmodified, with no performance or functional degradation. SaltStack Enterprise is identical to the open source Salt project, but offers enterprise support.

FREE TRIAL

Open source

Semaphore

CI

PRODUCT LAUNCH: Mar-12 | **TWITTER:** @semaphoreapp | **LOCATION:** Serbia, EU

SUPPORTED TARGET SYSTEMS:

- [ ] Windows
- [x] Linux
- [ ] Mac OS X
- [ ] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Quick setup process through GitHub integration
- Automatic project configuration that works without any changes in source code
- Parallel testing capabilities for languages and frameworks
- Live project timeline and server deployment history

HOSTING OPTIONS

- Cloud hosted

FULL PROFILE LINK

dzone.com/r/yHX9

DESCRIPTION

Semaphore lets developers create a continuous delivery workflow with a testing and deployment solution in the cloud, built on dedicated hardware. Semaphore automatically configures the build and deploy environment for a growing list of languages and frameworks. It works without any change in source code. Collaborators can follow the progress on a live project timeline. Semaphore requires that you use GitHub for source control, and it is tightly integrated with their system.

FREE TRIAL

30-day free trial

Serena Release Manager

ARA

PRODUCT LAUNCH: Apr-08 | **TWITTER:** @serenasoftware | **LOCATION:** San Mateo, CA, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Insights through dashboards and reporting
- Approval and notification system can be calendar-based, rule-based, or on-demand
- Reusable processes that can be tailored to release type with supporting audit trail
- Single UI that supports multi-platform apps

SUPPORTED CI ENGINES

- Jenkins/Hudson
- JetBrains TeamCity
- CloudBees

FULL PROFILE LINK

dzone.com/r/7Nuh

DESCRIPTION

Serena Release Manager is an ARA and configuration management solution that allows developers to manage the implementation of software changes in the production environment. Release Manager is designed to support the application release management lifecycle from demand to deployment, including planning and tracking the steps of releasing applications, greater visibility into those processes, and a way to enforce release policies.

FREE TRIAL

30-day free trial

Shippable

CI

PRODUCT LAUNCH: Mar-14 | **TWITTER:** @BeShippable | **LOCATION:** Seattle, WA, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [ ] Mac OS X
- [ ] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Utilizes dedicated Docker containers for each user
- Users can trigger multiple builds to test language runtimes and environments
- Integrated test results and code coverage visualization
- Offers granular team-based permissions for each member

HOSTING OPTIONS

- On-premise and cloud hosted

FULL PROFILE LINK

dzone.com/r/khs3

DESCRIPTION

Shippable is a continuous integration and deployment cloud service. It is free to use for private repositories on GitHub or BitBucket and natively supports Docker. Users can sign in with their GitHub or BitBucket credentials and turn CI on for repositories they want to enable. Shippable offers real time notifications, matrix builds, integrated test/code coverage visualizations, and build history.

FREE TRIAL

Unlimited builds, unlimited public repositories, up to 5 private repositories, 1 parallel build minion

Thoughtworks Go

CI

PRODUCT LAUNCH: Jul-10 | **TWITTER:** @thoughtworks | **LOCATION:** Chicago, IL, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [x] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Deployment Pipelines as a first-class concept
- Dashboard provides automatic, traceable bug reporting
- Allows for custom automation of testing functions
- Model complex workflows using job, tasks, stages, and pipelines

HOSTING OPTIONS

- On-premise

FULL PROFILE LINK

dzone.com/r/N3pH

DESCRIPTION

ThoughtWorks Go is a Continuous Delivery solution that offers development, business, and operations team members a common platform for collaboration. Go automates and streamlines the build-test-release cycle for continuous delivery of applications. It was built with the Continuous Delivery book's principles in mind. Go is now open source as of March 2014, and the source code will be available in April 2014.

FREE TRIAL

Open Source

Travis CI

CI

PRODUCT LAUNCH: Feb-11 | **TWITTER:** @travisci | **LOCATION:** Berlin, Germany, EU

SUPPORTED TARGET SYSTEMS:

- [ ] Windows
- [x] Linux
- [x] Mac OS X
- [ ] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Integration and collaboration around GitHub pull requests
- Support for Mac and iOS builds
- Large community contributions and following
- Command line tooling
- Completely free for open source projects

HOSTING OPTIONS

- On-premise and cloud hosted

FULL PROFILE LINK

dzone.com/r/94zr

DESCRIPTION

Travis CI is a hosted continuous integration and deployment platform that began life as a free platform for open source projects. Now Travis CI is a fully hosted product, supporting both Linux and Mac/iOS build environments. Travis CI requires GitHub as its version control system, and it tightly integrates with GitHub to test pull requests. Travis CI also has integrations with several databases and supports most programming languages.

FREE TRIAL

100 builds

XebiaLabs XL Deploy

RESEARCH PARTNER ARA

PRODUCT LAUNCH: May-11 | **TWITTER:** @xebialabs | **LOCATION:** Boston, MA, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [ ] Solaris
- [x] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Agentless architecture
- Declarative automation that eliminates scripting and workflow design
- Fully automated rollback and recovery
- Enterprise security and centralized auditing
- Visualize Continuous Delivery pipelines with template reports

SUPPORTED CI ENGINES

- Jenkins/Hudson
- Atlassian Bamboo
- Microsoft TFS

FULL PROFILE LINK

dzone.com/r/M3aA

DESCRIPTION

XL Deploy is an agentless application release automation solution. XL Deploy combines intelligent deployment automation, the richest content set for enterprise middleware, and seamless integration into the application delivery ecosystem. XL Deploy's unique, agentless architecture means less maintenance, lower costs, and support for much more diverse deployment scenarios that will become common with the Internet of Things.

FREE TRIAL

Fully functional, time-limited evaluation version. Trial license extensions available upon request.

Zend Server

ARA

PRODUCT LAUNCH: Feb-08 | **TWITTER:** @zend | **LOCATION:** Cupertino, CA, USA

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [ ] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Web APIs for integrating CI and config management tools
- Multi-mode PHP app performance optimization
- Developer tools that provide enhanced debugging
- Long-term support for the open source PHP runtime

SUPPORTED CI ENGINES

- Jenkins/Hudson
- Atlassian Bamboo

FULL PROFILE LINK

dzone.com/r/rVNp

DESCRIPTION

Zend Server is an application server with a supported PHP runtime that gives PHP developers, DevOps engineers, and sysadmins tool integrations to automate and accelerate app delivery. Integrating with tools like Jenkins, Chef, and Nagios, Zend Server orchestrates app deployment automation, scalability, performance monitoring, request analysis, and configuration management.

FREE TRIAL

Free edtion. 30 days full access to all Enterprise features.

ZeroTurnaround LiveRebel

ARA

PRODUCT LAUNCH: May-11 | **TWITTER:** @liverebel | **LOCATION:** Estonia, EU

SUPPORTED TARGET SYSTEMS:

- [x] Windows
- [x] Linux
- [x] Mac OS X
- [ ] Solaris
- [ ] AIX
- [ ] Other *(See companion site)*

STRENGTHS

- Automated rollbacks with no downtime
- Version-controlled configuration management
- Self-contained deployment script management with no downtime
- Pre-written scripts to integrate with third party tooling from many categories

SUPPORTED CI ENGINES

- Jenkins/Hudson, Bamboo, and TeamCity

FULL PROFILE LINK

dzone.com/r/ULJf

DESCRIPTION

ZeroTurnaround's LiveRebel enables the release of Java, .Net, PHP, and other apps automatically with configuration changes. Instead of a constrictive plugin system, LiveRebel features a powerful command line with plenty of simple scripts to help make flexible integrations with other tools. LiveRebel fits in with continuous integration and provisioned infrastructure to help create a full Continuous Delivery pipeline that ties application builds to infrastructure deployments.

FREE TRIAL

2 free licenses

GLOSSARY OF TERMS

A

AGENT - A program installed on specific physical servers in order to handle the execution of various processes on that server.

AGILE SOFTWARE DEVELOPMENT - An iterative and incremental software development workflow methodology with several sub-methodologies.

APPLICATION RELEASE AUTOMATION (ARA) - A type of continuous software release solution that handles packaging and deploying applications into a variety of environments and ultimately into production.

B

BEHAVIOR-DRIVEN DEVELOPMENT (BDD) - A deployment methodology that asserts software should be specified in terms of the desired behavior of the application, and with syntax that is readable for business managers.

BUILD - A compiled version of software, and typically a work in progress.

BUILD AGENT - A type of agent used in continuous integration that can be installed locally or remotely in relation to the continuous integration server. It sends and receives messages about handling software builds.

BUILD ARTIFACT REPOSITORY - A tool used to organize artifacts with metadata constructs and to allow automated publication and consumption of those artifacts.

C

CAPACITY TEST - A test that is used to determine the maximum number of users a computer, server, or application can support just before failing.

COMMIT - An action that makes a set of potential revisions into permanent changes, also known as a "check in."

CONFIGURATION DRIFT - A term for the general tendency of software and hardware configurations to drift, or become inconsistent, with the template version of the system due to manual ad hoc changes that are not introduced back into the template.

CONFIGURATION MANAGEMENT - A term for establishing and maintaining consistent settings and functional attributes for a system. It includes tools for system administration tasks such as IT infrastructure automation (e.g. Chef, Puppet, etc.).

CONTINUOUS DELIVERY - A software production process where the software can be released to production at any time with as much automation as possible for each step.

CONTINUOUS DEPLOYMENT - A software production process where changes are automatically deployed to production without any manual intervention.

CONTINUOUS INTEGRATION (CI) - A software development process where a continuous integration server rebuilds a branch of source code every time code is committed to the source control system. The process is often extended to include deployment, installation, and testing of applications in production environments.

D

DEPLOYMENT - A term that refers to the grouping of every activity that makes a program available for use and moving it to the target environment.

DEPLOYMENT PIPELINE - An automated process for getting software from version control to the production environment by moving those builds through multiple stages of testing and deployment.

DEVOPS - An IT organizational methodology where all teams in the organization, especially development teams and operations teams, collaborate and implement technology to increase software production agility and achieve business goals.

E

EXPLORATORY TESTING - A manual testing strategy where human testers have the freedom to test areas where they suspect issues could arise that automated testing won't catch.

I

INFRASTRUCTURE-AS-A-SERVICE (IAAS) - A self-service computing, networking, and storage utility on-demand over a network.

INTEGRATION TESTING - Testing that occurs after unit testing, but before validation testing, where individual software components are combined and tested as a single group.

ISSUE TRACKING - Also known as bug tracking or defect tracking, this is a process that allows programmers and quality assurance personnel to track the flow of defects and new features from identification to resolution.

P

PLATFORM-AS-A-SERVICE (PAAS) - An application-server-as-a-service (aPaaS) provided over a network along with additional components that may include middleware, development tools, and application controllers.

PRODUCTION - The final stage in a deployment pipeline where the software will be used by the intended audience.

R

ROLLBACK - An automatic or manual operation that restores a database or program to a previous defined state in order to recover from an error.

S

SELF-SERVICE DEPLOYMENT - A feature where deployment processes are automated enough for developers to allow project managers or even clients to directly control push-button deployments.

SOURCE CONTROL - A system for storing, tracking, and managing changes to software. This is commonly done through a process of creating branches (copies for safely creating new features) off of the stable master version of the software and then merging stable feature branches back into the master version. This is also known as version control or revision control.

SYSTEM ADMINISTRATOR (SYSADMIN) - A member of the operations department who is responsible for the upkeep, configuration, and stable operation of a server, multi-user computer, local area network (LAN), large program, or community.

U

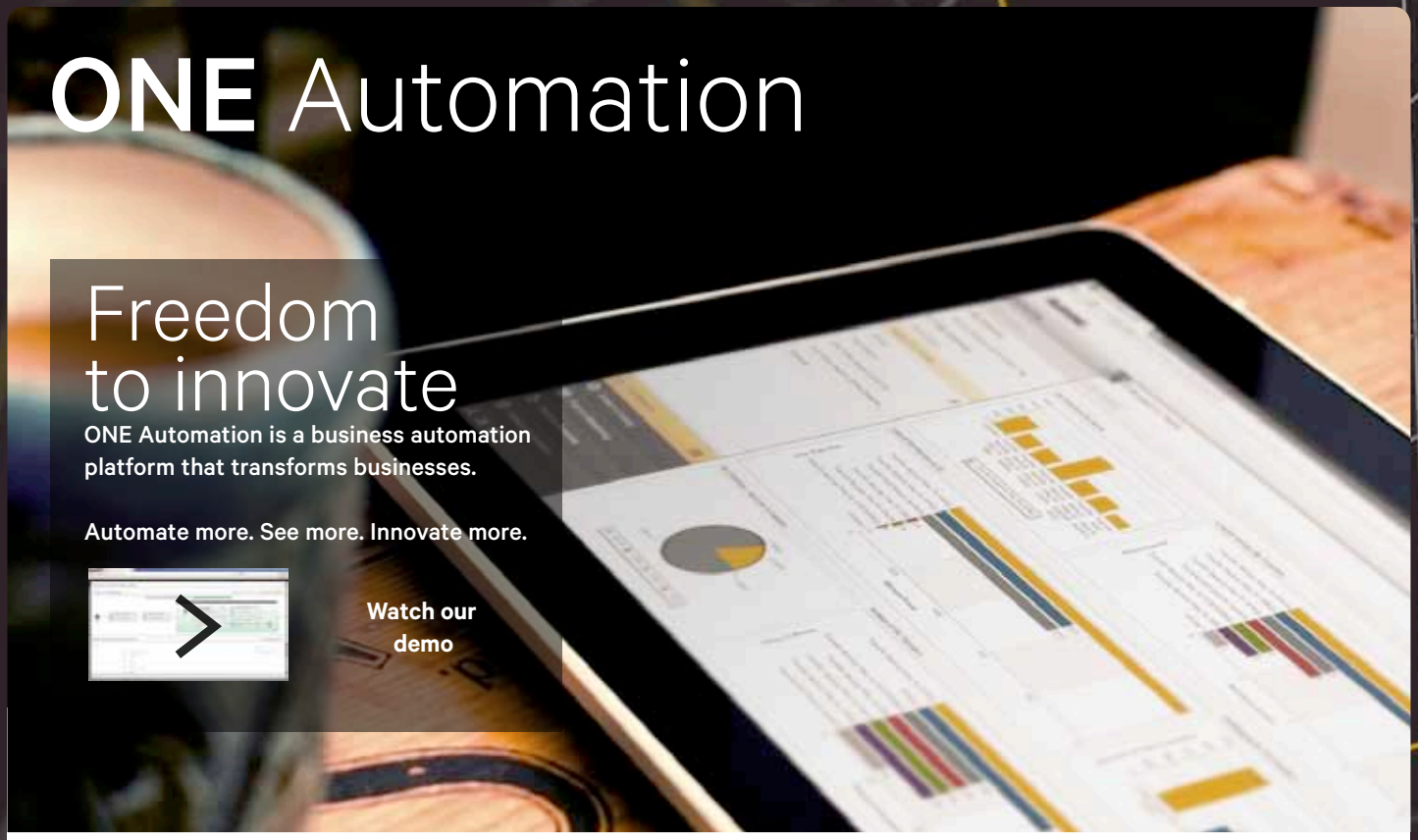
UNIT TESTING - A testing strategy in which the smallest unit of testable code is isolated from the rest of the software and tested to determine if it functions properly.

USER ACCEPTANCE TEST - The final phase of software testing where clients and end-users determine whether the program will work for the end-user in real world scenarios. This stage is also known as beta testing.

V

VIRTUAL MACHINE (VM) - A software emulation of a physical computing resource that can be modified independent of the hardware attributes.

ONE Automation



Freedom to innovate

ONE Automation is a business automation platform that transforms businesses.

Automate more. See more. Innovate more.

Watch our demo

ONE Automation means less complexity, more innovation

Your business, development and operations teams are busier than ever. Confronted with an ever growing list of applications and infrastructure they are consumed with the complexity of running the business leaving them little time for innovative transformation.

ONE Automation lets you give them the one thing they need most - time.

Benefits for business

- Simple to use interface
- Visibility of the entire business process
- Easily understand up and downstream applications
- See the availability of services that matter most
- Predict and respond to errors using analytics

Benefits for development

- Drag & drop deployment workflows
- Smart deployment model
- Hundreds of built-in run books
- Integration with source control, binary repositories, build and continuous integration servers

Benefits for operations

- Centralized script repository
- Reuse the same workflows across thousands of servers
- Dynamic workload management
- Self service catalog

What's possible with ONE Automation

Learn how Automic lets you go faster, for less cost whilst staying in control. ONE Automation is the only automation platform that combines all automation disciplines and operates across business, application and infrastructure layers.

Automic™

Let's Automate Business.

www.automic.com