Introduction

As application development becomes more complex, organizations need to stay quick and agile about delivering mature products that delight customers. Continuous Delivery is a critical approach to transforming the people, process, and technology to ensure delivery toolchains are seamless, reliable, and efficient.

This whitepaper provides a blueprint for mature Continuous Delivery pipelines. It explains why well-engineered, mature delivery pipelines are important to provide both agility and quality.

We will explore:

- The importance and benefits of Continuous Delivery pipeline maturity
- Best practices for configuring mature Continuous Delivery pipelines
- A Continuous Delivery Maturity Model framework to help you self-assess and plan a roadmap to advance your maturity level

The suggested framework and guidelines can help you improve both the agility and quality of your application delivery.
The abbreviation “CD” is ambiguous because it is often used to refer to the “Continuous Delivery pipeline,” “Continuous Delivery,” and “Continuous Deployment,” which are three related but separate things. According to Jez Humble, Continuous Delivery is the ability to get changes into production or into the hands of users safely and quickly in a sustainable way.

A key distinction is that Continuous Delivery is about the ability to get changes into production but does not necessarily include deployment to production. There are a lot of cases, such as new platform products, in which it is not desirable to deploy all software releases to production, even though, from an internal process point of view, there are many reasons to get the software release ready to deploy to production.

Continuous Deployment is a set of practices that enable every change that passes automated tests to be automatically deployed to production. Continuous Deployment takes continuous delivery to a higher level of automation. Continuous Deployment depends on Continuous Delivery but takes the changes all the way to production.

“Continuous Delivery and Deployment both depend on Continuous Delivery pipelines that continuously integrate software developed by the development team, build release candidate artifacts, and run automated tests on those artifacts to detect problems. The best practice for CD pipelines requires pushing artifacts into increasingly production-like environments to minimize the chance that software may fail when deployed to the production environment.”

Book Reference “Engineering DevOps” by Marc Hornbeek
Why Is Continuous Delivery Pipeline Maturity Important to DevOps?

Benefits of mature, well-engineered Continuous Delivery Pipelines include the following:

- A well-engineered Continuous Delivery pipeline provides visibility for software as it propagates through each stage.
- Quick lead times and more frequent releases provide quicker access to user feedback.
- Smaller, incremental, controlled releases are less painful, and failure events are lower risk.
- Reduced lead improves time-to-market for innovative new features.
- Software quality and stability are improved when each change is following a disciplined, well-engineered Continuous Delivery pipeline.
- Cost of software change is reduced when lean engineering practices are applied to the Continuous Delivery pipeline.
- Customer and employee satisfaction improve when they see positive results of the Continuous Delivery practices.

Mature, well-engineered Continuous Delivery and Deployment avoids the following types of problems:

- Inefficient Change Review Board meetings required to approve releases
- Voluminous release documentation instead of automated deployments
- Reliance on manual error prone testing
- Frequent corrections to the manual release process
- Manual configuration errors
- Lengthy manual deployments
- Frequent release roll-backs caused by manual errors
- Unexpected interruptions during a long release
- Sitting bleary-eyed in front of a monitor during release hell nights and weekends
How are Continuous Delivery Pipelines Engineered for DevOps?

Figure 1—DevOps Continuous Delivery Pipeline Blueprint shows how mature continuous delivery pipelines are configured.

The CD pipeline starts with release candidate artifacts delivered from the CI pipeline into an artifact repository. The CD pipeline is automated, and artifacts are deployed to a pre-production staging environment for system and user acceptance testing. These tests are automated as much as possible to minimize processing delays and bottlenecks. Once the tests pass, the artifacts are assessed with release policies to win approval for deployment. The release acceptance criteria determined by a Change Approval Board (CAB) are implemented as codified policies as much as possible. Once approved, the artifacts are deployed to production according to a deployment schedule. To minimize risk, the deployments are conducted using the safe Green/Blue and Dark Launch methods that are explained in this section. Once deployed, there may be additional post-deployment tests to evaluate software options using A/B testing and Canary testing methods.
The following are engineering practices consistent with mature well-engineering continuous delivery pipelines:

- Automate the build, deploy, test, and release process. This ensures consistent configuration of the system, environments, and the release process. This enables your processes to be less error-prone, more traceable, and easier to control, and it will help you remediate problems. Overall, you get a better-quality result.

- Frequent small releases are safer. Smaller, frequent releases reduce the risk associated with bigger, less frequent releases. It’s much easier to roll back the application and its associated configuration (including its environment, deployment process, and data).

- Fast feedback is essential. Any change, of whatever kind, needs to trigger the feedback process. The feedback must be delivered as soon as possible. The delivery team must receive feedback and act.

Many organizations have not achieved the level of continuous delivery pipeline maturity described above. For an organization to mature their continuous delivery pipeline capabilities they need to know their current state of maturity and target future state. **There are three dimensions to consider at each level: People, Process and Technology.** Technology such as better test tools will not accomplish a higher level of maturity unless the People and Processes use the tools effectively.
Continuous Delivery Pipeline Maturity Levels

There is no standard continuous delivery pipeline maturity model. As a practical matter, I have found it useful to apply a five-level maturity model adapted from the “standard” software Capability Maturity Model. I define five levels of continuous delivery pipeline maturity as described in the following paragraphs.

Continuous Delivery Pipeline Maturity Level 1: Chaos

*Figure 2* shows key characteristics of People, Process and Technology evident at this level of maturity. At this level there is little evidence of continuous delivery skills. Pipeline processes are not well integrated and pipeline automation technology has major gaps. Typical outcomes are releases are unpredictable, reactive, and the pipeline is wasteful.

<table>
<thead>
<tr>
<th>People</th>
<th>Process</th>
<th>Tech</th>
</tr>
</thead>
<tbody>
<tr>
<td>□ Silo team organization</td>
<td>□ Requirements, planning and tracking processes poorly defined and operated manually</td>
<td>□ Manual builds and deployments</td>
</tr>
<tr>
<td>□ Little communication</td>
<td>□ Unpredictable and reactive</td>
<td>□ Manual quality assurance</td>
</tr>
<tr>
<td>□ Blame, finger-pointing</td>
<td></td>
<td>□ Environment inconsistencies</td>
</tr>
</tbody>
</table>

*Figure 2—Continuous Delivery Pipeline Maturity Level 1: Chaos*

Continuous Delivery Pipeline Maturity Level 2: Continuous Integration

*Figure 3* shows key characteristics of People, Process and Technology evident at this level of maturity. At this level there is some knowledge of continuous delivery pipelines. The builds, integrations, and build tests are well supported by processes and technology, End-to-end continuous delivery, especially delivery and deployment stages are not well supported with automation. Build quality is good, but typically there are deployment quality problems, and the infrastructure is used inefficiently.

<table>
<thead>
<tr>
<th>People</th>
<th>Process</th>
<th>Tech</th>
</tr>
</thead>
<tbody>
<tr>
<td>□ Managed work backlog</td>
<td>□ Processes defined within silos</td>
<td>□ Source code version management</td>
</tr>
<tr>
<td>□ Communication between silos</td>
<td>□ Lack E2E process standards</td>
<td>□ Automated builds, release artifacts and automated tests</td>
</tr>
<tr>
<td>□ Limited knowledge sharing</td>
<td>□ Can repeat what is known, but can’t react to unknowns</td>
<td>□ Painful but repeatable releases</td>
</tr>
</tbody>
</table>

*Figure 3—Continuous Delivery Pipeline Maturity Level 2: Continuous Integration*
Continuous Delivery Pipeline Maturity Level 3: Continuous Flow

*Figure 4* shows key characteristics of People, Process and Technology evident at this level of maturity. At this level continuous delivery processes extend from end-to-end across the pipeline. Dev and QA teams cooperate to ensure a good level of test coverage is automated. Release standards are using automated testing metrics. Typical outcomes include repeatable quality releases with some bottlenecks and inefficiencies.

<table>
<thead>
<tr>
<th>People</th>
<th>Process</th>
<th>Tech</th>
</tr>
</thead>
<tbody>
<tr>
<td>□ DevOps leadership</td>
<td>□ End-to-end CI/CD pipeline automated</td>
<td>□ Toolchain orchestrates and automates builds, tests and packaging deliverables</td>
</tr>
<tr>
<td>□ Collaboration between cross-functional teams</td>
<td>□ Standards across the org for applications, releases processes and infrastructure</td>
<td>□ Infrastructure as code</td>
</tr>
<tr>
<td>□ DevOps training program</td>
<td></td>
<td>□ Metrics and analysis for release acceptance and deployment</td>
</tr>
</tbody>
</table>

*Figure 4*—Continuous Delivery Pipeline Maturity Level 3: Continuous Flow

Continuous Delivery Pipeline Maturity Level 4: Continuous Feedback

*Figure 5* shows key characteristics of People, Process and Technology evident at this level of maturity. At this level more advanced knowledge of continuous delivery pipelines is apparent. Goals and metrics are set for each stage in the pipeline. The culture includes training and mentoring for continuous delivery. There is a focus on end-to-end performance trends rather than spot results. Automation is applied to test environment orchestration and analytics. Typical outcomes at this level reflect a general confidence to obtain repeatable quality releases, with a metrics-driven culture.

<table>
<thead>
<tr>
<th>People</th>
<th>Process</th>
<th>Tech</th>
</tr>
</thead>
<tbody>
<tr>
<td>□ Collaboration based on shared metrics to remove bottlenecks</td>
<td>□ Proactive monitoring</td>
<td>□ Applications, pipelines and infrastructure fully instrumented</td>
</tr>
<tr>
<td>□ SLIs, SLOs and SLAs to ensure stakeholder alignment</td>
<td>□ Metrics collected and analyzed against business goals</td>
<td>□ Metrics and analytics dashboards</td>
</tr>
<tr>
<td>□ DevOps Mentors and Guilds</td>
<td>□ Visibility and repeatability</td>
<td>□ Orchestrated deployments with automated rollbacks</td>
</tr>
</tbody>
</table>

*Figure 5*—Continuous Delivery Pipeline Maturity Level 4: Continuous Feedback
Continuous Delivery Pipeline Maturity Level 5: Continuous Improvement

*Figure 6* shows key characteristics of People, Process and Technology evident at this level of maturity. At this level there is a high level of knowledge and confidence regarding continuous delivery pipelines. Dev and QA teams are tightly integrated to optimize knowledge and efficiency. End-to-end processes focus on the end-customer experience and more sophisticated risk-based strategies. Typical Outcomes include an extremely high confidence and satisfaction by all stakeholders, with an innovation-driven culture. Achievement at this level provides a platform for autonomous continuous improvement strategies in which automation and intelligence drive innovation.

**People**
- Culture of continuous experimentation and improvement

**Process**
- Self-service automation
- Risk and cost optimization
- High degree of experimentation

**Tech**
- Zero downtime deployments
- Immutable infrastructure
- Actively enforce resiliency by forcing failures

*Figure 6—Continuous Delivery Pipeline Maturity Level 5: Continuous Improvement*
**Continuous Delivery Pipeline Maturity Assessment**

While the above five levels of continuous delivery pipeline maturity provide a practical guide for defining maturity against characteristics of People, Process and Technology, they are not an absolute measure of maturity. Organizations, or specific applications within an organization, may match some of the characteristics for different levels. *Figure 7—Continuous Delivery Pipeline Maturity Model* is a useful tool to determine the “best fit” for the maturity of an organization or application within an organization. By marking the characteristics that best match, gives a visual picture of the dominant level of maturity. This also is a quick way to determine areas to address to improve the level of maturity.

<table>
<thead>
<tr>
<th>People</th>
<th>Process</th>
<th>Tech</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chaos</td>
<td></td>
<td></td>
</tr>
<tr>
<td>☐ Silo team organization</td>
<td>☐ Requirements, planning and tracking processes poorly defined and operated manually</td>
<td>☐ Manual builds and deployments</td>
</tr>
<tr>
<td>☐ Little communication</td>
<td>☐ Unpredictable and reactive</td>
<td>☐ Manual quality assurance</td>
</tr>
<tr>
<td>☐ Blame, finger-pointing</td>
<td>☐ Environment inconsistencies</td>
<td></td>
</tr>
<tr>
<td>Continuous Integration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>☐ Managed work backlog</td>
<td>☐ Processes defined within silos</td>
<td>☐ Source code version management</td>
</tr>
<tr>
<td>☐ Communication between silos</td>
<td>☐ Lack E2E process standards</td>
<td>☐ Automated builds, release artifacts and automated tests</td>
</tr>
<tr>
<td>☐ Limited knowledge sharing</td>
<td>☐ Can repeat what is known, but can’t react to unknowns</td>
<td>☐ Painful but repeatable releases</td>
</tr>
<tr>
<td>☐ Ad hoc training</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Continuous Flow</td>
<td></td>
<td></td>
</tr>
<tr>
<td>☐ DevOps leadership</td>
<td>☐ End-to-end CI/CD pipeline automated</td>
<td>☐ Toolchain orchestrates and automates builds, tests and packaging deliverables</td>
</tr>
<tr>
<td>☐ Collaboration between cross-functional teams</td>
<td>☐ Standards across the org for applications, releases processes and infrastructure</td>
<td>☐ Infrastructure as code</td>
</tr>
<tr>
<td>☐ DevOps training program</td>
<td></td>
<td>☐ Metrics and analysis for release acceptance and deployment</td>
</tr>
<tr>
<td>Continuous Feedback</td>
<td></td>
<td></td>
</tr>
<tr>
<td>☐ Collaboration based on shared metrics to remove bottlenecks</td>
<td>☐ Proactive monitoring</td>
<td>☐ Applications, pipelines and infrastructure fully instrumented</td>
</tr>
<tr>
<td>☐ SLIs, SLOs and SLAs to ensure stakeholder alignment</td>
<td>☐ Metrics collected and analyzed against business goals</td>
<td>☐ Metrics and analytics dashboards</td>
</tr>
<tr>
<td>☐ DevOps Mentors and Guilds</td>
<td>☐ Visibility and repeatability</td>
<td>☐ Orchestrated deployments with automated rollbacks</td>
</tr>
<tr>
<td>Continuous Improvement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>☐ Culture of continuous experimentation and improvement</td>
<td>☐ Self-service automation</td>
<td>☐ Zero downtime deployments</td>
</tr>
<tr>
<td>☐ Risk and cost optimization</td>
<td>☐ High degree of experimentation</td>
<td>☐ Immutable infrastructure</td>
</tr>
<tr>
<td></td>
<td></td>
<td>☐ Actively enforce resiliency by forcing failures</td>
</tr>
</tbody>
</table>

*Figure 7—Continuous Delivery Pipeline Maturity Assessment Model*
Conclusion

Software development organizations must move beyond the “chaos reigns” state of low repeatability and reliability to one that brings cloud-native applications to market faster and more securely than ever before. Automating the build, test, deploy process with frequent small releases is critical to deliver value to customers quickly and get fast feedback. This, in turn, requires a Continuous Delivery solution that facilitates the AWS well-architected design benchmarks of operational excellence and performance efficiency.

Getting started with the right tools

Finding the tools that work best for your organization is critical for automating and scaling your application delivery pipelines. AWS Marketplace provides third-party software that is purpose-built for implementing a Continuous Delivery practice in AWS. These solutions help apply security, provide support, and integrate with a broad range of DevOps tooling.

Continuous Delivery tools available in AWS Marketplace

- **Armory**
  - Enterprise-grade open source based CD platform that deploys workloads in Amazon EC2, Amazon EKS, Amazon ECS, and more.
  - Try now

- **CircleCI - CI/CD**
  - CI/CD delivery platform that enables teams to orchestrate complex workflows and automate software delivery at scale.
  - Try now

- **CloudBees Products**
  - Connect, automate and orchestrate tools across development, operations and shared service teams to optimize software delivery.
  - Try now

- **GitLab Ultimate**
  - Source code management, CI/CD, monitoring, and more all in a single application to enable concurrent DevOps.
  - Try now

- **Harness Continuous Delivery**
  - CD-as-a-service platform with machine learning to detect the quality of deployments and automate rollbacks.
  - Try now

- **Stackery**
  - Point-and-click serverless development and deployment platform.
  - Try now

View all solutions and learn more about CI/CD >
About AWS Marketplace

AWS Marketplace is a curated digital catalog that simplifies software discovery, procurement, provisioning, and management. With AWS Marketplace, customers can also utilize features that speed up product evaluation, improve governance and cost transparency, and enhance control over software spend. AWS Marketplace offers third-party solutions across software, data, and machine learning tools that enable builders to find and deploy solutions to expedite innovation.

Customers can launch pre-configured solutions in just a few clicks in both Amazon Machine Image (AMI) formats and software-as-a-service (SaaS) subscriptions, with entitlement options such as hourly, monthly, annual, and multi-year contracts.

AWS Marketplace is supported by a global team of solution architects, product specialists, and other experts to help IT teams connect with the tools and resources needed to streamline migration journeys to AWS.

Key benefits of building your DevOps practice with solutions in AWS Marketplace

- **Find solutions 46% faster**
  Find the market-leading tools you need to orchestrate your ideal DevOps toolchain.

- **Adopt new tools 53% faster**
  Try leading-edge developer tools and simplify procurement with integrated AWS billing.

- **Deploy your way, 48% faster**
  Deploy DevOps solutions with methods that best fit your use case using containers, SaaS, AMIs, APIs, or CloudFormation Templates, and more.

- **Trust AWS interoperable technologies**
  Count on tools that are designed for AWS interoperability and are regularly scanned for security vulnerabilities.

About DevOps Institute

DevOps Institute works to advance the human elements of DevOps. We create a safe and interactive environment where members can network, gain knowledge, grow their careers, support enterprise transformation, and celebrate professional achievements.

We are not just an information source; we connect and enable the global member community to drive human transformation in the digital age.

Contributors:

- **Marc Hornbeek**
  Ambassador – DevOps Institute
  CEO – Engineering DevOps Consulting
  Analyst – Accelerated Strategies Group
  Author – “Engineering DevOps”

- **Alex Jones**
  Sr. Digital Demand Generation Manager
  AWS Marketplace

*2020 Amazon Web Services Marketplace online survey with 200 IT decision-makers and influencers in the U.S.*