The Definitive Guide to Modern Software Delivery

CloudBees.

Contents

A Profile of an Enterprise

Over the past decade, it has become clear that modern enterprises succeed or fail on their ability to develop, deliver and leverage software. Whether one is analyzing government agencies or the private sector, a close inspection reveals that adopting modern software development and delivery capabilities has enabled these organizations to increase productivity, reduce delivery risk and drive innovation.

Accepting that software paves the way to a modern enterprise and successfully embracing that reality prove to be very different challenges. Many larger organizations seek to implement the modern approaches to software that are celebrated by the once smaller digital native companies. These practices include agile (interactive, user-centric planning and execution), continuous deployment (deploying every change to production), two-pizza-sized teams (small teams focused on specific features and functionality) and mantras like "fail fast and fix things." Enterprises, however, face unique challenges implementing these practices in a manner that sustainably drives the successful outcomes they seek.

Enterprises have to overcome the challenges of scale and complexity. Such organizations have multiple divisions each with multiple teams, leveraging multiple technologies coordinated across multiple functional groups, and often needing to align to provide a coherent user experience and drive shared objectives. A government agency, for instance, might have a shared services division, various mission programs, a security and compliance office, an IT infrastructure division, a mobile/web applications group, a contracting office, various systems integrators and so on. Each of these groups might have multiple teams using technology specifically suited for unique goals. Some of those technologies might have been adopted decades ago while others might have only recently been introduced. Ultimately, all of the unique efforts have to be coordinated across different functional groups such as project management, application development, quality assurance and operations.

Scale and complexity lead to a diversity of teams, tools and technologies. However, the root problem is not the diversity of the organization but rather the pervasive *disconnection* across teams. Disconnections between development and operations, leadership and practitioners, front-end and back-end teams, etc. all result in lower productivity and failure to innovate. Ultimately, the mission suffers and user satisfaction craters.

Solving the Disconnection Problem

The reality of these pervasive disconnections is that even enterprises that adopt DevSecOps will still find it difficult - if not impossible - to make software delivery rapid, repeatable and scalable. The solution lies in connecting all phases of the software development and delivery lifecycle to achieve a shared and seamless process within a unified view. Ensuring that each team understands how other teams contribute to the pipeline enables the entire organization to work together to remove obstacles and achieve common goals. By further connecting to the enterprise's overall objectives, software teams can align with the business to drive results that matter. The best way to connect these historically disparate views is to focus on the **five pillars of software delivery**: continuous integration (CI), continuous delivery (CD), release orchestration, feature management and value stream management.

Continuous Integration and Continuous Delivery (CI/CD)

Modern development teams have embraced CI/CD as the best approach when implementing automation across the software delivery lifecycle. The first pillar, CI, means committing verified changes to code and more frequently integrating them with the application as a whole, thereby limiting the risk of conflicts and ensuring the code is working properly by itself but also as part of the application. The second pillar, CD, means automatically and continuously verifying that code has been vetted and approved for release, ensuring that the capabilities being built are always production-ready.

In an enterprise, deployments tend to be more complex and must often adhere to rigid compliance, governance and regulatory audit needs. The fear of deploying a change that will "break" production causes many organizations to create unnecessary processes, strict approval gates and even deployment committees or change approval boards in an effort to prevent negative impact to end users. However, these onerous approaches to gate-keeping are negatively correlated with both speed and software quality. Building a disciplined CI/CD strategy that connects engineering teams to the stakeholders requesting - and approving - their changes improves trust and increases the velocity at which new and valuable features can get into end users' hands.

Release Orchestration

Advanced CI/CD pipelines can be very intricate, especially if they include many different test and release stages. Thus, modern software delivery practices in an enterprise require more than a solid CI/CD strategy. Release orchestration glues together CI and CD processes to ensure that pipelines are trustworthy and provide the evidence to prove it - to managers, security and accreditation teams, auditors, etc. Done correctly, release orchestration helps IT leaders manage release pipelines and dependencies across all teams, DevSecOps toolchains and environments, enabling Ops to safely and predictably release new applications and adapt to change at any speed demanded by the mission.

Whereas CI/CD takes software applications from code to production-ready, release orchestration focuses on putting them into production and ensuring they deliver their mission value as intended. Release orchestration enables a high level of governance and visibility into complex enterprise pipelines, making it vital for modern software delivery. Unfortunately, most organizations today either fumble through it manually or neglected it altogether. On the plus side, this reality presents profound growth opportunity for organizations wise enough to invest in building their release orchestration expertise.

Feature Management

For IT leaders, managing which features are consumable in production is a critical aspect of software releases. Feature flags, part of the fourth pillar of modern software delivery, give development teams more granular control over when and how features are delivered to users, allowing teams to progressively deliver fully proven features when stakeholders are ready.

Traditional software release practices require engineering and business teams to work in lock step in order to align business and technical requirements before a release. Unfortunately, these teams rarely work in perfect sync due to different priorities, practices and timelines. Feature flags allow features to be deployed to production when the development team is ready but revealed to users only when the business team is ready, removing the need for lock-step work and allowing each team to work at its own pace.

Value Stream Management (VSM)

Successfully meeting the demands of rapid capability delivery requires IT and engineering teams to fully align with the overarching business goals of the mission. Yet all too often they're left out of the decision-making process and forced to react to, rather than anticipate, the rapidly changing landscape of end user needs and new or improved technologies. Value stream management allows engineering teams to define and prioritize their goals as concrete mission outcomes and demonstrate their value back to the mission or business.

Closely monitoring the performance of the value stream in the "software factory" allows engineers to prioritize work to make certain that it advances mission/business objectives. A strategic view of software value streams allows managers to objectively and reliably measure and monitor improvements to software delivery teams. Demonstrable and easily consumable metrics, such as those defined by DORA, enable efficient communication with executive stakeholders.

Best Practices to Support the Five Pillars

In order to break down silos and connect across people, tools and processes, it is imperative to adopt all five pillars of modern software delivery as fundamental operating principles. Deceptively attractive as standalone practices, implementing just one or a few separate from the whole simply shifts the problems out of sight and into other areas of the software pipeline. Only by understanding the entire lifecycle of a desired change from both technical and business perspectives can organizations achieve the speed, safety and value offered by modern software delivery.

Integrated, Automated, End-to-End Software Delivery

By harnessing the power of automation and orchestration, while following the operating principles described above to forge connections across diverse teams, technologies and functional groups, enterprises can achieve end-to-end visibility into, and management of, the software delivery lifecycle. This approach ensures repeatable, reliable and rapid hand-offs, which reduces the risks and cost of software delivery. Embracing the complexity of the enterprise software landscape by implementing shared platforms, with a shared process and a shared view, instead of redesigning the entire software delivery toolchain, enables teams to use the tools and technologies they trust and the organization to leverage its existing investments.

Software Delivery Automation and the Future of Modern Software Delivery

How do enterprises make this a reality? The answer is Software Delivery Automation, connecting the end-to-end software delivery lifecycle across the organization's teams tools and technologies with a unified automation and orchestration layer. It empowers teams to weave purpose-fit tools and technology into an end-to-end delivery platform. Organizations can go faster without compromising quality or security.

CloudBees Software Delivery Automation solutions enable enterprises to leverage connected, proven solutions to overcome the disconnection problem by delivering on all five pillars: continuous integration, continuous delivery, release orchestration, feature management and value stream management. By leveraging the CloudBees platform, these software delivery automation solutions reduce the risk and cost of software delivery and allow organizations to get the most out of existing investments.

Software Delivery Automation enables organizations to:

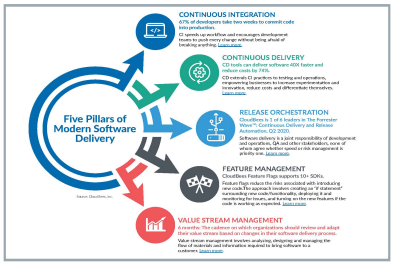
- Use real-time, actionable insights into the value stream to optimize software delivery, end-to-end
- Track performance metrics across projects and teams
- Make sense of software releases in a consistent, reliable and repeatable way
- Embed security into CI/CD and release pipelines, and collect bodies of evidence, for DevSecOps

...so that they can:

- Start development with the confidence that teams are following a trusted approach
- Produce proof that delivery is going well using concrete, industry standard methods
- Expedite approvals and provide insight into the readiness of capabilities to proceed into Ops
- Consistently deliver on time, on budget and at scale

By providing deep integration between market-leading solutions for CI, CD, release orchestration and feature flag management across all teams and tools, CloudBees Software Delivery Automation solutions enable any organization to implement all five pillars of modern software delivery at scale and across a complex, diverse enterprise portfolio.

Learn More



Replay the DevOps World Session

Audit-Ready Release Pipelines are Table Stakes for DevSecOps

Read the Case Study

IRS Modernizes with Agency-wide DevOps Initiative

Watch the Webinar

Accelerated ATO with CloudBees SDA

Learn More:

www.cloudbees.com/software-delivery-automation

CloudBees CI is built on top of Jenkins, an independent community project. Read more about Jenkins at: www.cloudbees.com/jenkins/about

© 2021 CloudBees, Inc. CloudBees is a registered trademark and CloudBees CI, CloudBees CD, CloudBees Engineering Efficiency, CloudBees Feature Management, CloudBees Build Acceleration and CloudBees CodeShip are trademarks of CloudBees. Jenkins® is a registered trademark of the Linux Foundation. Other products or brand names may be trademarks or registered trademarks of their respective holders.

0221v00

CloudBees, Inc.
4 North Second Street | Suite 1270
San Jose, CA 95113
United States
www.cloudbees.com
info@cloudbees.com