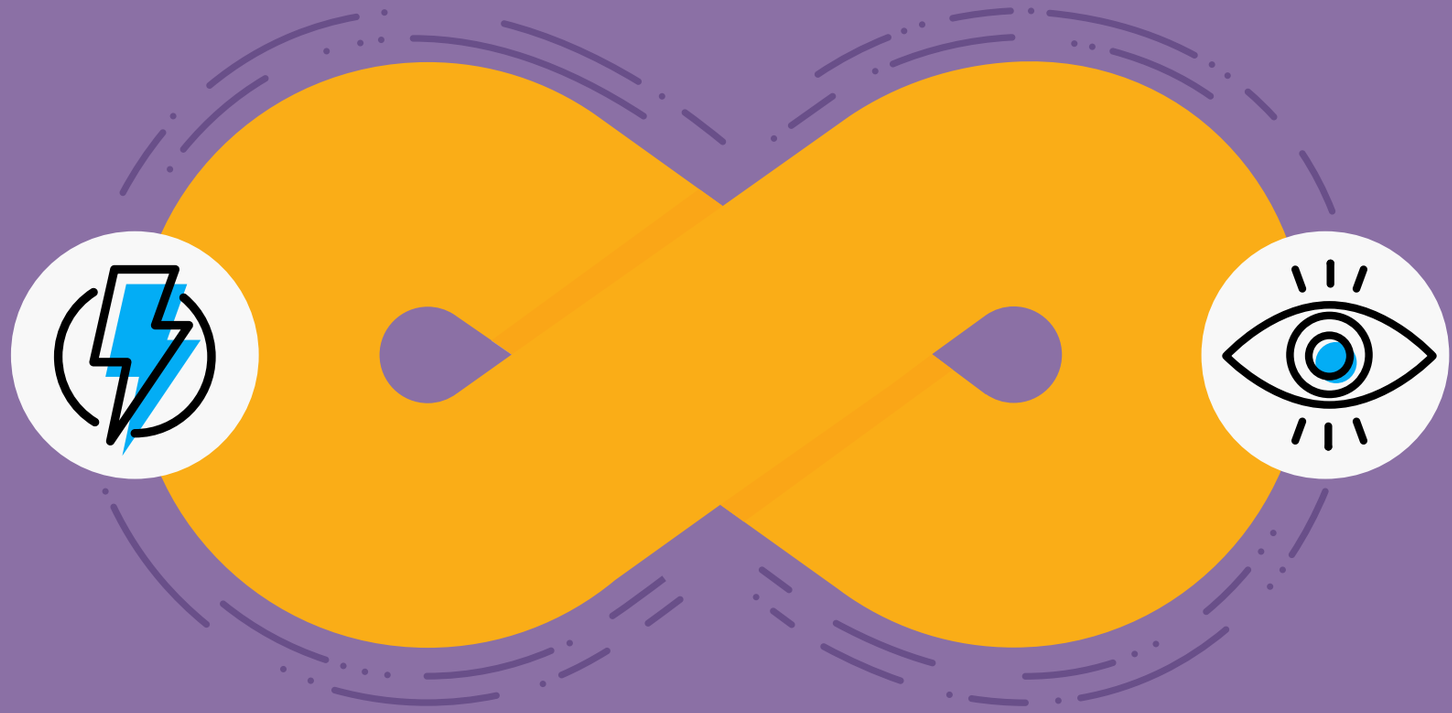
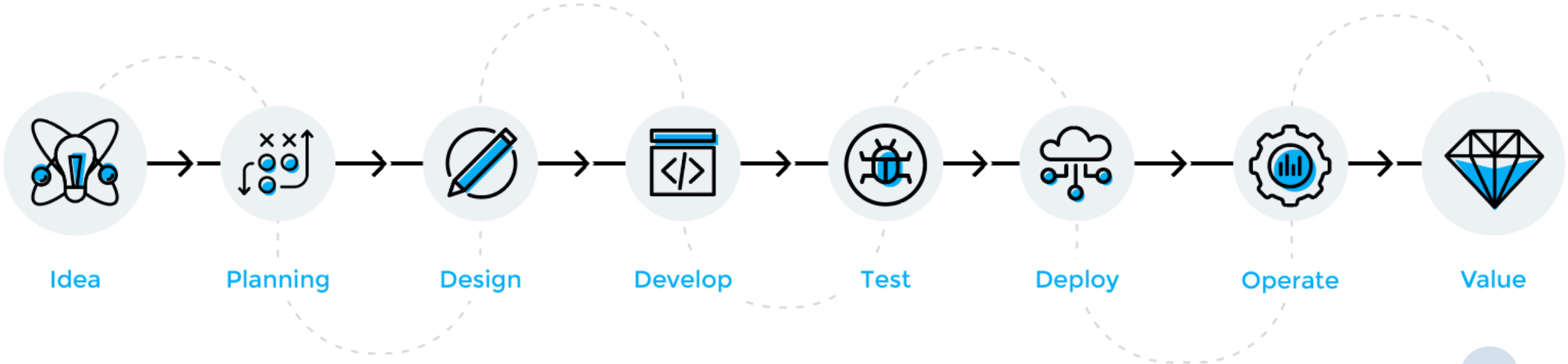


WHITE PAPER

The Role of Release Management in a DevOps World

Release management is more important than ever in the fast-paced, distributed DevOps world.

PLUTORA



Executive Sumary

When talking about DevOps, it's easy to focus too much on the speed of each individual step in the software creation and deployment process while losing sight of the both the business objectives and how well everything is working together. Release management takes a more holistic view, looking both at software in the context of its business goals as well as how each piece of the technology stacks works together both in production and during the development process. A key focus of release management is zooming out far enough to understand complex dependencies and thus minimizing risk associated with any change to the application or infrastructure. Release management tools make it easier for businesses to manage the technical and organization complexity of distributed software architectures, distributed teams and distributed responsibility for application health. This makes it possible to accurately evaluate risk, see complex dependencies and, ultimately, connect across teams in the IT department as well as other business stakeholders. Release management isn't just about creating better software, it's about ensuring business alignment, reducing risk, and improving the speed of software delivery for the business.

What Is Release Management?

The role of release management has changed as organizations move to a more DevOps-focused process, but there has never been a 'standard' definition for release management. In the 'old days,' release management referred to shipping a finished piece of software into production. The most expansive definitions of release management in a waterfall-style development environment involved ensuring that a piece of software met security and governance requirements before becoming publicly available. The least expansive definitions referred just to the technical steps required to take a piece of code from the pre-production environment to the production environment.

Release management is evolving as companies adopt a DevOps-style approach... but not everyone is one the same page about the role of release management. "In the IT field, there actually could be quite a few interpretations that are widely varied on what release management is," explains the Director of Release Management at a global pharmaceutical company. On the one extreme, there are people who still think of release management as creating a build and doing a deploy.

Then there are others who think of release management and project management as the same, or release management is something project managers handle. This is a more mature view but still falls short of what true release management is.

True release management is everything you need to do to plan, coordinate, establish resources, develop, test and facilitate the transition into production for a piece of software. Its role is to coordinate the relationship between project management, testing and quality assurance, configuration management and deployment to ensure that every release follows compliance and security best practices and company guidelines. Release management ensures that all the stakeholders are working together and understand how components built by different teams will work together in the production environment.

Friend and Foe

"You are an enabler to get something out, but you're also the gate to say something is not going to get out," explains the Director of Release Management of a global pharmaceutical company, about the role of release management teams. "You could be a friend or you could be a foe to the same team five hours later."

The goal of release management is successful deployment. The more visibility you have into the entire system, the less likely you'll need to put the brakes on further down the pipeline, because you'll be able to identify and fix potential problems much sooner in the development process. Using integrated release management tools gives release management teams the real-time information needed to spend more time being friends and less time slamming on the brakes at the last minute.

Solving the Whole Puzzle

A mature release management system looks at the entirety of the software development process, from identifying a business need and determining how to meet that need with software to developing, testing and deploying that software to end customers. "Release management is the end, think of it as the end to end process of getting an idea to production, all the things that need to happen as part of that," Dalibor Siroky, CEO of Plutora, explains. "And that would include, how does that ideation phase feed into the next phase, how does the design phase build into the construction phase, so those iterative phases … and what are the handoffs, how do people collaborate?"

> **"How does that ideation phase feed into the next phase, how does the design phase build into the construction phase, so those iterative phases … and what are the handoffs, how do people collaborate?"**
>
> — Dalibor Siroky, Plutora CEO

Release management is a way to zoom out and see the entire picture. This includes the end-to-end process of idea to software development to value delivered to customers as well as the complete view of company's technical environment, from where everything is in the application lifecycle to the complete tech stack down to the servers. Release management is about understanding the big picture—how a group of applications moves through its lifecycle, how each application interacts with and depends on other parts of the tech stack, where security vulnerabilities exist and the risk any particular deployment has of blowing up the whole system.

In a DevOps world that's focused on moving fast, getting this complete picture is the only way to both make sure software is meeting its intended business goals and being delivered in a way that doesn't go wrong regularly. "Fundamentally, the business is saying 'how can I get this idea to production faster, so that I can get generate more benefits for my customer?'" explains Siroky.

"The notion of working DevOps style and delivering fast, to many people, is sort of the equivalent of 'Don't worry about me, I'm just going to spray stuff out there and see what happens," explains the Director of Release Management at a global pharmaceutical company. "The reality is you need Swiss-clock precision."

Release management provides the guardrails to make Swiss-clock precision consistent across an entire enterprise. Here's how it fits into the DevOps world.

The Release Management Ecosystem

Release management involves zooming out, to understand how your organization is using software to meet customers' needs.

Release management refers to managing the entire process, from business logic to customer delivery, for a single application.

Enterprise release management refers to managing the entire portfolio of an organization's applications, from business logic to value delivered to customer, including understanding how these application interact with and depend on each other.

Value stream management is the broadest term, that zooms out to get the widest view of how software is contributing to business value. Value stream management involves creating a feedback loop to evaluate how well the finished software product is meeting the business need as originally identified—and iterating based on how customers interact with the software. Value stream management ties IT in with business metrics and other departments, depending on the software's goal.

Why DevOps Needs Release Management

Release management is crucial to successfully managing the complexity that the combination of DevOps and distributed, microservice-based architectures have introduced into the application landscape. In the real world, organizations that are following DevOps practices without integrating release management often run into the following challenges.

Fragmented information, Informal Collaboration

DevOps prioritizes the creation of small teams and giving them wide latitude to create and deploy applications. This process helps speed up the application development and deployment process, but creates and perpetuates the myth of entirely self-sufficient teams. In reality, both the teams and the software they are creating has complex dependencies. The DevOps process tends to create information silos, making it each team a black box. In the absence of a formal system to organize cross-team and cross-departmental collaboration, everything from information sharing to collaboration is done in an ad-hoc, informal, fragmented way.

> **"The amount of moving parts and dependencies is incredible, and unless everybody's on the same page and you have a flight screen, a radar view of the entire sky, it's a statistically likely that things will break, probably fairly often,"**
>
> — Director of Release Management at global pharmaceutical company

"The vast majority of the project programs focused around business capabilities have dependencies in and out, upstream downstream, but a good portion of them think they live in their world," explains the Director of Release Management at a global pharmaceutical company. Often, individual teams are too low-level and focused on specific functionality—authentication management, for

example—to be able to see the bigger picture or even how authentication management fits into and depends on other parts of the application. DevOps gives the illusion of independence—and when one team changes or deletes a resource another team's app depended on, the consequences can be dramatic.

In addition, it's difficult for anybody to get a complete view of an organization's suite of applications. "Information is often fragmented, informal and not capable of delivering true decision-making information to key stakeholders," Siroky explains. Businesses create software to meet business objectives, but the lack of visibility into the organization's software pipeline makes it difficult to make informed business decisions.

Constant Change

"Everything is shifting all the time," explains the Director of Release Management. "People change, the roles change, there's a lot of shifts in and out of responsibility." This state of constant flux makes it difficult to get everyone in an organization on the same page. Everyone has different cycle times and different responsibilities in the application lifecycle. In addition, each team might follow a different development and deployment procedure, use different tools and write applications in a different language.

Constant change means that information has to be available in real-time, because it can be out-of-date immediately. But relying on the informal, point-to-point information gathering common in the real world makes it impossible to get an accurate view of the changing application landscape.

Decentralized Accountability

In the past, there was more of a centralized gatekeeper that all new code had to go through. DevOps breaks down this centralized role, creating more acceptable pathways to get into production. Cross-functional, product-oriented teams that include experts in development, testing, security and deployment have the autonomy to deliver their code directly—as well as the responsibility to run the application once it deploys. There's no centralized oversight at any part of the application lifecycle. This is autonomy is a key part of DevOps, but in an environment where information is often siloed, dependencies are poorly understood, teams are delivering as fast as possible and everything is in constant flux, the decentralized responsibility for releases is a recipe for disaster.

"Now we're dealing with more pipes, from more practitioners who practice many different ways," explains the Director of Release Management. "The quality of planning, integration, coordination is all over the map. So bringing it all together to something cohesive, that in the end not just works together, but is properly timed and doesn't blow anything else up. That is a challenge."

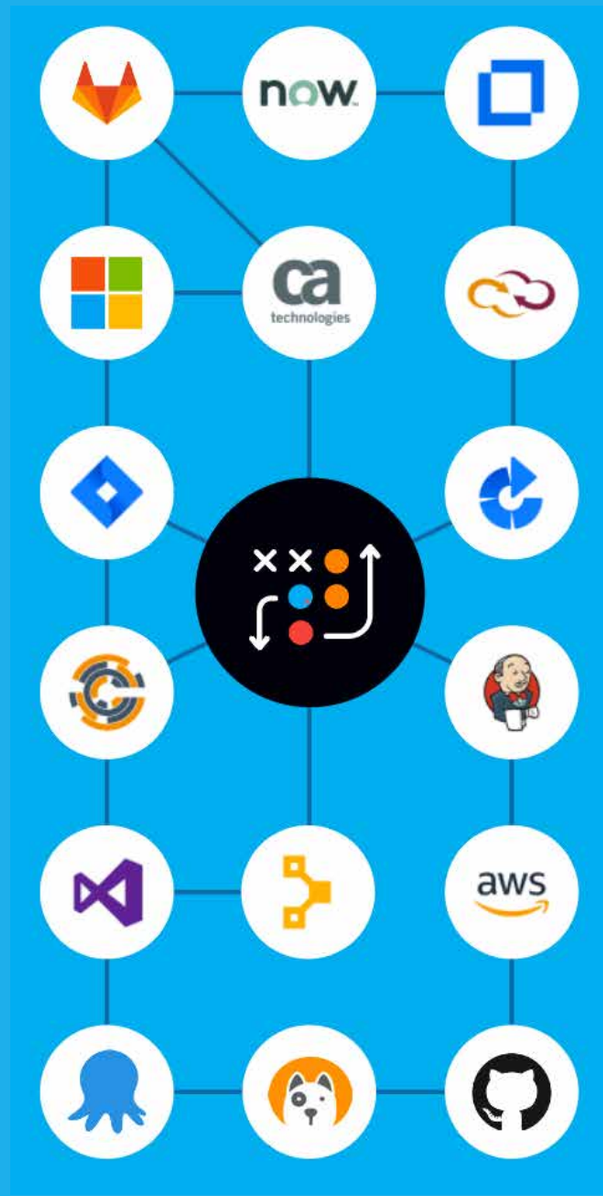
Misunderstood Risk

Constant change, decentralized responsibilities and lack of information make it hard for anybody to truly understand the risk associated with any individual deployment, which requires granular visibility into an applications entire stack, starting with the server operating system as well as all the upstream and downstream dependencies.

The inability to understand risk profiles goes to the core of what happens when you have distributed team, diverse working methods and incomplete information sharing—in other words, DevOps without release management. “The amount of moving parts and dependencies is incredible, and unless everybody’s on the same page and you have a flight screen, a radar view of the entire sky, it’s a statistically likely that things will break, probably fairly often,” explains the Director of Release Management at a global pharmaceutical company. “The question is what breaks? Is it something you could live with, or is it a real issue?”

Integrating Automated, Holistic Release Management with DevOps

When release management is integrated into the automated pipelines and processes that are at the core of the DevOps development paradigm, there’s a way for information to be shared between engineering teams as well as with other business-level stakeholders. Creating a formal, automated information repository, as well as standardizing the release steps and making it easy to visualize complex dependencies, brings the DevOps chaos back under control. Here’s how an integrated release management helps organizations tackle the challenges DevOps creates without infringing on the speed and autonomy that makes DevOps successful.



Risk Transparency

Comprehensive release management tools provide transparency into the entire toolchain as well as the web of upstream and downstream dependencies, making it possible to accurately evaluate different kinds of risk factors, from whether or not the deployment will introduce a security risk to how likely it is to break another team's application. Understanding what the possible consequences of a failed deployment are can both inform whether or not a deployment is ready for production as well as what needs to be monitored post-deployment.

A major part of the risk transparency is the ability to understand dependencies before one causes an outage—ideally, in fact, before an application is even at the testing phase. Comprehensive release management surfaces dependency information automatically before developers alter or create an application, providing better information for that practitioner to keep in mind from the very beginning.

Without release management tools, release managers are asked to evaluate risk by looking at past releases and using past experience to predict future performance—a process that inevitably involves gut feelings. Using a tool like Plutora takes the guesswork out of this process by looking at the complexity and dependencies of the release to make accurate risk assessments as well as recommend ways to reduce that risk.

Make Coordination a Reality

DevOps might be about breaking down the barriers between development and operations, but it can create barriers between individual teams as well as between engineering teams and other stakeholders, like security, compliance or legal departments. Integrated, automated release management tools surface information seamlessly, in real time, without a need to interrupt developers' workflow.

Information sharing makes it possible not only to know that your application's dependencies, but also to see how where other teams, whose applications have upsteam or downstream dependencies on yours, are in their workflow. This makes it possible for teams to see how changes to the application affect others, and vice versa, at every step of the process. This advance notice of possible issues down the line makes it possible to coordinate and prevent potential collisions from cause production outages.

> **"In a desire to increase frequency, you typically get a high degree of automation, you get a higher degree of collaboration and you can't do it without information sharing. That information sharing comes from more of a decentralized structure, and that decentralized structure needs to support and supply the information seamlessly"**
>
> — Dalibor Siroky, Plutora CEO

See the Big Picture

Value stream management tools provide a catwalk over the factory floor, giving business stakeholders as well as engineering leaders a view of the entire software ecosystem that includes everything from idea to production. It's easier to see bottlenecks in the process or places where current procedures aren't working as well as they should In addition, this zoomed-out view includes not just information about how the software is performing from a technical perspective, but also how it contributes to business goals.

Optimize for Customer Value

Release management is about more than creating better software and reducing outages. Ultimately, comprehensive release management tools allow organizations to worry less about the technical nitty-gritty and more about how they use software to create value for customers and value for the business, leading to better customer experiences and higher net promoter scores. "The starting point of a release is at the planning and scoping phase, what's the idea that we're building, why are we building it?" explains Siroky. "The other part is, is it delivering value to the customer?"

If you don't have a view of the end-to-end process, you can only optimize individual pieces of the pipeline rather than the entire process as a whole. When you lay it out and look at how everything connects to each other, you can start to optimize the entire process, from business logic through customer feedback. This makes it possible to iterate based on how well a piece of software is meeting customer needs and delivering its intended business value.

Release Management in Product-Oriented Teams

A key part of the DevOps transition has been moving to self-contained, product-oriented teams that can handle an increasingly large share of the application lifecycle, including designing, building and running the application. As companies move to more mature DevOps models, they incorporate more roles into the DevOps team. At one point, testing was handled by testers external to the development team—now testing is the product-oriented team's responsibility. Release management is currently handled almost exclusively by dedicated release management teams, outside of the product-oriented development and operations team. But as DevOps adoption becomes more mature, release management responsibilities will be transferred to one of the product-oriented team's members or the team as a whole, though possibility still with coaching from a release management group.

While responsibility for release management will become more distributed, the need for centralized control will never disappear—especially in industries that have strict compliance requirements. Automating as much of the release management process allows a centralized release manager to create the guardrails that teams throughout the company will have to respect. Building release management into the development process at all stages, inside of pre-defined guardrails and policies, gives product-oriented teams the agility they need to move fast while ensuring that compliance, security and release management best practices are followed consistently.

Conclusion

Most organizations adopt DevOps in an effort to move faster and provide value to internal and external customers more quickly. Particularly at an enterprise scale, however, DevOps adds complexity from a technical and human resources perspective. Integrating comprehensive release management increases the visibility into the application lifecycle and creates guardrails to ensure everyone is following best practices. Surfacing information about dependencies between teams, getting a complete view of the risk profile and enforcing security and compliance best practices company-wide makes it possible to move at the speed companies expect from DevOps without an unacceptable level of risk that any given deployment will blow up the entire system in unpredictable, hard-to-debug ways.

True release management gives companies a zoom-out view of the application lifecycle that makes it possible to identify inefficiencies and stitch together the creative parts of application design with the tasks best handled by automation. Zooming out also puts the application in the context of the business and the business goals, allowing organizations to optimize not just for software quality, but for value delivered to customers.

About Plutora

Plutora, the market leader of value stream management solutions for enterprise IT, improves the speed and quality of software creation by capturing, visualizing and analyzing critical indicators of every aspect of the delivery process. Plutora orchestrates release pipelines across a diverse ecosystem of development methodologies, manages hybrid test environments, correlates data from existing toolchains, and incorporates test metrics gathered at every step. The Plutora Platform ensures organizational alignment of software development with business strategy and provides visibility, analytics and a system of insights into the entire value stream, guiding continuous improvement through the measured outcomes of each effort.

PLUTORA

Learn more: www.plutora.com
Email: contact@plutora.com