PLUTORA

WHITE PAPER

Setting Up the Critical DevOps Role of Enterprise Release Management

While the DevOps movement and associated technologies have garnered much attention and fanfare, few have addressed the core issue—the handoff from development to operations. In this white paper, we look at the pitfalls of not acknowledging the importance of Release Management—a critical DevOps function.

Release Management is the bridge between development and operations, and you can strengthen that bridge with the right approach, tools, teams, and processes. This white paper provides a tailored approach to creating an efficient and effective Release Management function within your IT organization.

Bridging the Chasm Between Development and IT Operations

Development teams are "proactive." Developers create software systems, and they often have an entirely different model of business from the Operations teams. These groups have a wide variation in both process and approach to software releases. Some development groups working on slow-moving, back-office systems may be very amenable to the service management model of software delivery.

Highlights

- Bridging the chasm between development and operations.
- Approaches to software release management.
- The importance of Enterprise Release Management for successful DevOps.
- Best practices for setting up an effective Enterprise Release Management function.
- How Plutora helps bridge the DevOps divide.

Development teams are “proactive.” Developers create software systems, and they often have an entirely different model of business from the Operations teams. These groups have a wide variation in both process and approach to software releases. Some development groups working on slow-moving, back-office systems may be very amenable to the service management model of software delivery.

Other groups, who are focused on fast-moving, highly technical systems, are not often aligned with the IT Service Management (ITSM) and Information Technology Infrastructure Library (ITIL) models.

When it comes to tools, developers are more at home managing development processes with issue tracking tools like Atlassian’s JIRA, and configuration management tools like Puppet or Chef.

Traditionally, development teams had the sole purpose of satisfying a particular business need. They tended to focus on software delivery and internal quality metrics, and left it to release managers to act as a buffer between any internal process requirements, or as a bridge across other groups that may have taken a more formal approach to service management.

In short, developers focus their efforts on ‘pushing out code.’ Changes and enhancements to production systems, and attendant failures in providing service, are seen as part of this rush to get the latest and greatest code out to consumers. Developers thus tend to be agile, proactive, and reliant on self-service for accomplishing their goals.

Developers focus their efforts on ‘pushing out code’ and tend to be agile, proactive, and reliant on self-service for accomplishing their goals.

OPERATIONS	**Reactive** Operations reacts to support requests, bugs, and releases. Releases are “change requests”.	**ITSM / ITIL** Operations understands process and decision trees focused on managing change and tracking availability	**Remedy / ServiceNow** Tools like Remedy and ServiceNow model releases as change requests to be reacted to.	**Central Management** Operations groups understand that production and testing environments are tightly-controlled entities in which changes are tracked and risk is managed.
DEVELOPMENT	**Proactive** Development creates change for operations to react to. Software is developed and delivered.	**DevOps / Agile** DevOps is focused on continuous, automated changes to systems. Agile encourages a continuous approach to software delivery.	**JIRA, Puppet, Chef** Development tools are always focused on the changes necessary to deliver the next release.	**Self-service** Increasingly empowered development groups view production and testing environments as self-service systems running on internal or external clouds.

Operations groups, on the other hand, are "reactive." They are focused on service management and incident response. Teams supporting operations tend to use organization models closely aligned with ITSM and ITIL. A site operations team at a high profile website will use the term "customer" to refer to internal end users creating incidents and change requests in a system like BMC Remedy or ServiceNow.

In another example, a team managing cloud infrastructure for forty internal development groups will view interactions with internal groups similar to how service providers view relationships with customers. To accurately track cost and forecast capacity, interactions with these groups would then follow well-established patterns for service management.

In a nutshell, Operations teams are responsible for keeping existing services available, meeting agreed upon Service Level Agreement (SLA), while still helping the business innovate by rolling out new products and enhancing existing products.

Also, IT Operations are always faced with less when it comes to budget and time. The mantra of doing more with less is driven by easy access to technologies such as virtualization and cloud computing that have increased efficiencies and made IT an open endeavor instead of a CapEx investment.

At the same time, competitive pressure and practices like DevOps now require Ops to be nimble, efficient, and cost-effective to ensure they can provide strategic value to the business and not just be considered a cost center.

This balance (business needs vs. accountability) makes operations teams cost-conscious, process-driven, reactive, and wary of sudden and rapid change.

Changes and enhancements to production systems, and attendant failures in providing service, are seen as part of this rush to get the latest and greatest code out to its consumers.

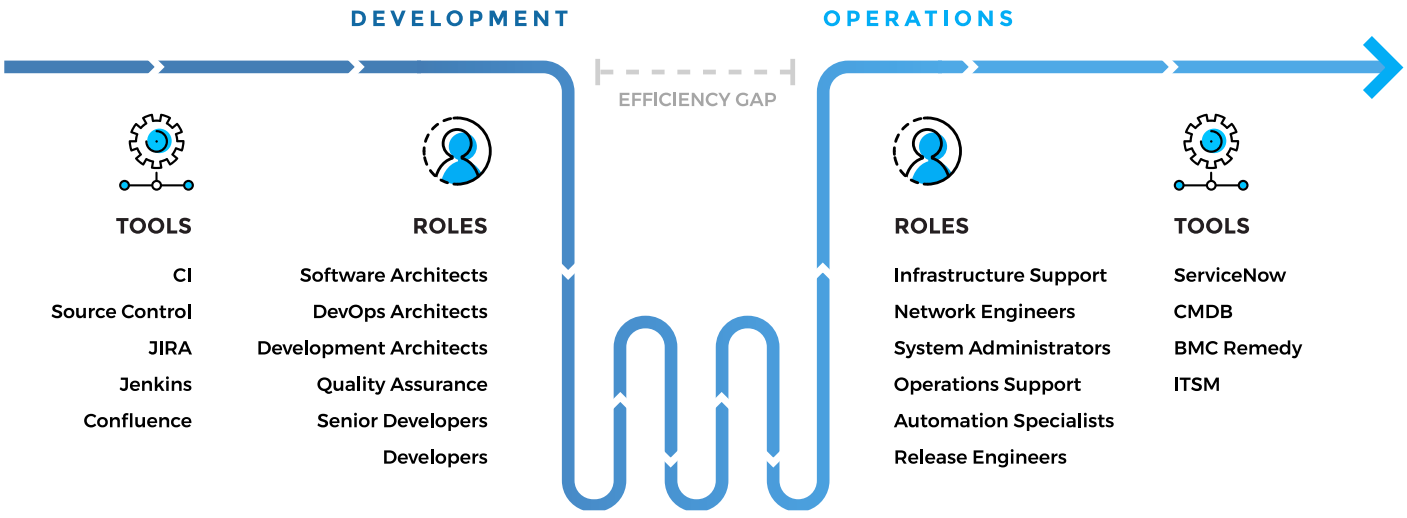

Figure 1: The canyon of distrust–from development to meaningful deployment.

Approaches to Software Release Management

Historically, ITSM and ITIL were both created to enable a set of standard best practices for service management and IT. When an international enterprise needs an internal Help Desk to manage the delivery of a software project to hundreds of thousands of employees, these are the standards that the enterprise can use off-the-shelf.

Different services can be assigned SLAs, and systems like ServiceNow or BMC Remedy can be used to track incidents. Costs are controlled, and system availability is accurately measured. These models work well if you are delivering a predictable set of services to internal "customers." Software releases are predictable events, which can be forecast days or months in advance, and are then tracked and managed in a rigid process that aligns with an ITIL standard.

If you are planning a release, you create a series of changes and aggregate them into release requests. A review board approves the requests, and a service analyst identifies the risks associated with the release process.

Ultimately, detailed analysis of the output of a release-tracking tool—that tracks a release as an incident to be managed and responded to—will identify any further ramifications of the change request.

The DevOps movement emphasizes automation, self-service deployments, and continuous delivery pipelines to support an Agile software development process. When an organization adopts tools and procedures associated with DevOps, this often shifts more of the responsibilities of software deployment and service management onto development teams, who are working closely with operations teams, with developers often driving the initiative.

DevOps seems to be working for some enterprises. In fact, it is not uncommon to hear success stories from the likes of Facebook, Netflix, Amazon, or Etsy. Claims of multiple (in some cases hundreds) of deployments a day are not uncommon. In fact, Etsy makes the bold claim that "a new developer commits to production on Day 1."

Is this reality to be found across the IT spectrum or is it the viewpoint of a few unicorns?

The DevOps movement emphasizes automation, self-service deployments, and continuous delivery pipelines to support an Agile software development process.

Most enterprises still feel stranded when it comes to DevOps. While they understand the benefits of being Agile, they do not know where to focus or how to start down the DevOps path. A big reason for this (as outlined in previous sections) is the inherently different nature, approach, tooling, and incentives between development and operations teams.

In short, there is a much overlooked yet huge disconnect within the IT department between teams tasked with supporting software and teams tasked with creating software. In these organizations, the intersection or handoff from development to operations continues to be a bottleneck.

Development still perceives operations as a brick wall they run into despite their best intentions and approach (Agile) to application development. To development teams, it still feels as if they are tossing code over this brick wall, hoping it will be deployed correctly by IT operations.

Operations, on the other hand, look at development suspiciously. Has adequate testing been done? What about ticket sign-offs, authorizations and approvals, and due process? How about ensuring the code has been acceptance-tested, been through various stages and gates, and is indeed ready for prime time?

There is a much overlooked yet huge disconnect within the IT department between teams tasked with supporting software and teams tasked with creating software.

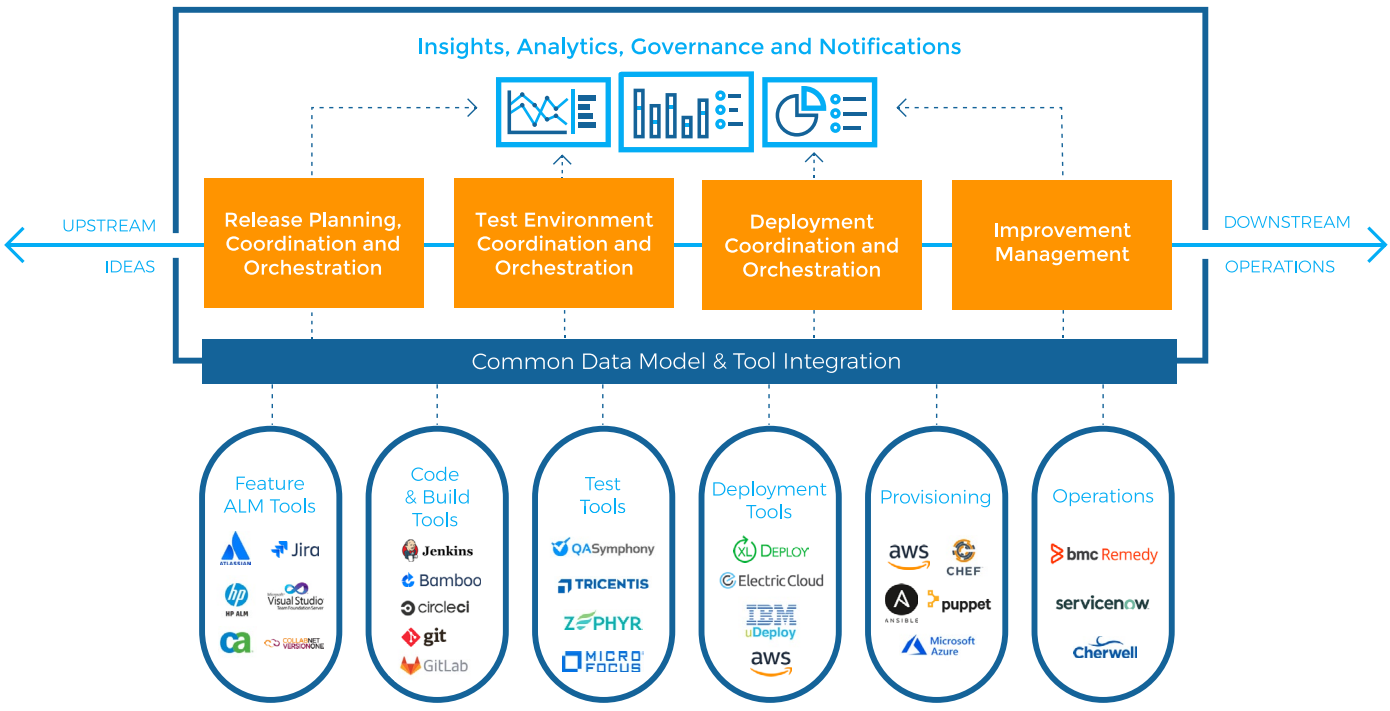

Figure 2: The application release workflow from a tools perspective–a critical diagram for IT management to understand. It shows the variety of tools available across the management and engineering disciplines to ensure a complete and efficient software delivery process. It further segments the tools by the functions or phases that typically use it. The Plutora platform brings all functions together for faster, better software delivery.

The Importance of Release Management

It is in this context of development meeting operations that Release Management becomes significant. It is indeed the lynchpin for a smooth transition of applications from code completion to deployment into live production environments.

It is easy to get carried away with thinking that DevOps begins and ends with ALM tools in conjunction with automation (Continuous Delivery) tools. While these tools serve an important purpose and have been readily embraced by development teams, they still leave huge gaps in the application delivery workflow.

Implementing proper Release Management functionality can help empower IT staff to drive change and help address these issues.

Poor Release Management Practice

Many IT organizations have embraced the Agile practices prescribed for development as well as the control processes (such as ITIL) prescribed for operations.

However, as these organizations and their releases have grown more complex, their Release Management function has been either absent or has failed to adapt with the changing times. As complexity grows, many issues will impact application release capabilities. Examples include:

- A lack of knowledge and visibility of other teams and their activities in the release.
- Difficulty collaborating across teams and their tools (spreadsheets, intranets, and wiki pages).
- Inability to produce real-time Release Management status reports.
- Test environment clashes, inefficient usage, and capacity mismanagement.
- A lot of time spent in trying to gain a single view of how a managed release is tracking.
- Unknown ownership of various release tasks.

These issues not only plague Release Management processes, but they can also lead to deployment delays and failed releases as well as to a higher incidence of issues in production. Overcoming these issues will require a culture change, but implementing proper Release Management functionality can help empower IT staff to drive change and help address these issues.

What is Good Release Management and What Does it Achieve?

Release Management is primarily concerned with the flow of change throughout various pre-production environments, culminating in successful deployment into the production IT environment in the least disruptive manner.

Release activities should include planning, designing, configuration, rollout planning, testing, communication, and deployment. Release Management should make the Change Management process more proactive and predictable and is crucial to managing the volume of independent change within any IT organization.

Pre-production environments like development, system testing, integration testing, performance testing, and user acceptance testing, all fall outside the formal Change Management controls of IT operations.

Given the velocity at which these environments change during build and test, an appropriate balance needs to be found between agility, flexibility, and control. Release Management and the Release Manager ensure that the functions of Development and Operations and the disciplines of Agile programming and ITIL come together.

Release Management and the Release Manager ensure that the functions of development and operations and the disciplines of Agile programming and ITIL come together.

Best Practices for Release Management

While it is common to think of Release Management as the final promotion of a component into a production environment, we take a far broader view of Release Management ranging from:

- The strategy of establishing quarterly, monthly, or biweekly release windows.
- Prioritizing the contents of each release.
- Understanding the complex dependencies and tracking delivery as it meets the integrated delivery targets.
- Aligning the necessary non-production or test environments to support the coordinated release strategy.

Enterprise Release Management success depends on the following ten best practice approaches to improving the Release Management function.

1. Review Existing Release Management Processes

Start by reviewing the existing Release Management function — the current personnel, processes, and tools. The characteristics of a successful Release Management function are capable people, a clearly defined, regular process, and a toolset that supports all participants in the process.

2. Establish and Enterprise Release Strategy

It is equally important to establish an enterprise release strategy that clearly articulates regular release cycles.

As Enterprise Release Management is concerned with ensuring all releases, projects, and work packages arrive at the same time within a defined window for integrated testing, it is essential to establish a release strategy with regular windows that ensures the deployment of new features to customers at regular intervals.

It is critical to define very early on what your Release Management mission and goals are. Achieve this by using de-facto policies, or by taking a more formal approach.

But it is critical that an organization's Release Management goals are well articulated. Goals may be metrics centered on successful releases, decrease in release-based downtime and outages, or more broader and strategic metrics such as measuring and growing the top line by an agreed-upon percentage.

Next, define the rules of engagement: the entry, journey through, and exit out of the Release Management pipeline.

Lastly, it is important to define critical success factors that will define the success or failure of a release. These range from measuring the number of incidents caused by releases, the number of failed releases, to the number of releases implemented but not tested, or the number of releases without operational assurance.

Other examples of metrics may be number of releases implemented late, volume of major and minor releases, percent of releases requiring a back-out plan to be implemented, average best and worst time to implement by release type, etc.

It is critical to define very early on what your Release Management mission and goals are. Achieve this by using de-facto policies, or by taking a more formal approach.

Rules of Engagement for Release Management

Examples

- Empowering Release Management to veto a new release based on specific acceptance criteria.
- Allowing the Release Manager to sit in on meetings and empowering him or her to assist in analyzing and approving changes.
- Organizing changes into units and groups with tactical rules before releasing them into production.
- Testing both pre-release and post-release into production quality and approving them with a sign-off from Release Management.
- Developing a back-out strategy depending on release and production environment criticality.

Enterprise Release Management is successful when it is exercised regularly and tweaks its processes slightly for each release based on the lessons learned.

Successful organizations strive to hold steadfast when it comes to their broader Release Management strategy, have well-defined and executed rules of engagement, as well as a feedback loop that takes the critical success factors into account.

Successful organizations strive to hold steadfast when it comes to their broader Release Management strategy, have well-defined and executed rules of engagement, as well as a feedback loop that takes the critical success factors into account.

3. Define the Optimal Release Management Process

The next step in an effective Release Management function is to define an optimal Release Management process. This has the following ingredients:

Identify the Release Management Process Inputs

The Release Management process should consider the following as inputs to the process:

- Portfolio and program management systems
- Service management systems
- Quality management systems
- Configuration management systems and deployment solutions

Identify Key Activities for Release Management

Key activities for an effective Release Management process workflow include:

- Release planning
- Coordination
- Design and building
- Configuring of releases
- Coordinating release acceptance
- Conducting rollout planning
- Coordinating release communications
- Training activities
- Coordinating distribution and deployment of releases into production
- Measuring and providing management with an overview of Release Management processes and key KPIs

As you may surmise, these closely mimic the Release Management Process Inputs except, for example, with Incident Management you must show which release fixed which incident. Again, the areas to focus on are:

- Incident Management
- Problem Management
- Change Management
- Configuration Management
- Service-Level Management
- Service Monitoring

Invest in the Right People

It is critical to invest in the right people to be custodians of the Enterprise Release function. The team with the best players wins, and it is no different in corporate teams and Release Management.

Program Managers and Project Managers will manage a broad set of workstreams and activities to deliver key milestones. Development managers will manage developers and produce work packages for deployment.

The Release Manager with executive support is responsible for all releases. He or she should coordinate the various functions and work activities at all levels, provide the authority or ability to promote releases, as required, and manage the process end-to-end so as to ensure optimal overall performance and quality.

The Environment Manager should ensure proper capacity utilization, configuration, and uptime of environments. Test, staging, and production support environments are critical. They each have their own needs, differing degrees of flexibility, and multiple stakeholders with vested interest in using them. The task of the Environment Manager is to coordinate a limited supply of environments across various release stages and stakeholders.

The Test Manager should ensure proper testing protocol is followed throughout the release process. He or she should ensure that each stage (unit testing, integrated test suites, user acceptance testing, etc.) is tested and relevant testing gates are maintained. The Test Manager should coordinate and communicate with the Release Manager, the Environment Manager, and the Development Manager.

It is critical to invest in the right people to be custodians of the Enterprise Release function. The team with the best players wins, and it is no different in corporate teams and Release Management.

Release Team

Roles

- Release Manager
- Environment Manager
- Test Manager
- Implementation Manager

Skills

- Leadership
- Organization and planning
- Technical
- Project Management
- Communication and teamwork

5. Choose the Right Tools

As seen earlier, a variety of tools across development, test, and operations are already being used. In addition, a robust Release Management tool is essential to ensure the success of the Release Management process. Such a tool should have:

- Stakeholder management
- Communications
- A master release calendar
- Automated Release Management workflow capabilities
- Data extraction and reporting
- Collaboration and views based on roles and function
- Auditing and process capabilities (setting up gates, inflow, outflow, stoppages)
- A dashboard for various stakeholders
- The ability to integrate with existing tooling
- A robust API for use by other tools in the environment

6. Plan for Test Environment Usage and Optimization

All phases of the release process require IT environments to be in place for test execution and validation well before the completion of any code. The release infrastructure covers the hardware, storage, network connections, bandwidth, software licenses, user profiles, and access permissions.

In complex integrated and secure environments, this is no trivial matter and requires thorough planning, understanding of interdependencies, alignment of specialist skill sets and resolving contention with competing initiatives. Critical environment bottlenecks must be eliminated as early as possible before they hold up delivery.

All phases of the release process require IT environments to be in place for test execution and validation well before the completion of any code. Critical environment bottlenecks must be eliminated as early as possible before they hold up delivery.

7. Ensure Transparency and Control

Releases encompass many moving parts. Transparency and control of each phase within a release are critical to alignment to a set test window. As releases move through their key phases, integrated gates, and milestones, work packages are promoted at a physical level through various environments for various forms of testing and validation. A transparent baseline of the environments as well as a clear understanding of the composition of promoted work packages prevents significant rework.

Releases encompass many moving parts. Transparency and control of each phase within a release are critical to alignment to a set test window.

8. Engage Stakeholders

A sign of an efficient and working Release Management function, and a worthwhile goal for any IT organization, is to be able to publish the target release plan for the next twelve months. While the composition of releases may not be certain, the intent is to lock in the release windows so that all teams work towards not only the final release date, but also the intermediate targets, such as completion of integrated testing, completion of user acceptance testing, and so on.

A sign of an efficient and working Release Management function, and a worthwhile goal for any IT organization, is to be able to publish the target release plan for the next twelve months.

In an environment where targets are continually shifting, being able to set and communicate release windows twelve months into the future essentially shifts the discussion to release composition rather than release date. Once the release dates are defined and approved, stakeholders should be engaged to prioritize outstanding feature requests and allocate them into future releases. Stakeholders fill releases as far into the future as practical. There should be certainty around the immediate next release and less definition about the composition of releases scheduled further into the future. Regular structured releases give customers confidence that they can order something and have it delivered.

9. Encourage Continuous Communication

Regular communication with stakeholders, delivery teams, and suppliers is essential to ensure all parties have a consistent view of the expected outcomes and the manner of achieving them. Where possible, information relating to the progress of the release should be available always in a frictionless manner and should not be limited to the summarized information presented in the form of dated presentations. Rather, all parties should have a systematic way of accessing the information they need in real time.

Regular communication with stakeholders, delivery teams, and suppliers is essential to ensure all parties have a consistent view of the expected outcomes and the manner of achieving them.

10. Establish Sponsorship and Metrics

Practitioners should consider having a sponsor. Senior, active sponsors bode well for any Release Management function. Additionally, having visible metrics to monitor end-to-end release health is critical. Taking the time to define these release health metrics and ensuring they are consistently measured and published is essential to establishing credibility. Understanding the business impact is critical to the Release Management function being taken seriously, getting funded, sponsorship and priority. Ensuring that technology concerns do not crowd out business impacts can be more effort than one expects.

REVENUE

Accelerated Time-to-Value

Good Release Management provides accelerated time to value. It does so by delivering services to customers on ever-shorter release cycles. End users realize the benefits of changes as quickly as possible.

Higher Release Throughput

Release Management delivers higher release throughput by absorbing higher rates of changes to systems while maintaining IT service quality through a unified, well-understood, and controlled release process.

Enhanced Agility and Flexibility

Release Management enhances agility and flexibility by responding to new and emerging needs and competitive threats as they arise. With all parties involved operating from consistent information, they can quickly understand the impact of new changes to the release schedule and risk profile of the release in order to respond positively to these demands.

COST

Increased Productivity

Effective Release Management increases productivity through the creation and enforcement of standards and best practices across the release process, as well as by ensuring more efficient allocation of test environments to support releases. It delivers smoother transitions of releases from development activities (projects) to final destination environments.

Increased Collaboration

Release Management increases collaboration by bringing together disparate teams and skill sets involved in the Release Management process through information sharing and instant communication. All release stakeholders have complete and timely insights into schedules, changes, priority, and status. Release Management metrics are clear across the organization.

Duplicate and Manually-Intensive Activities Eliminated

Release Management removes duplication and manually-intensive activities in the planning, management, and deployment stages of the release process. It also eliminates the need to be always reconciling inconsistent information sources from day-to-day activities.

RISK AND UNCERTAINTY

Mitigate Release Failure

Good Release Management practice mitigates release failure, the bane of any IT organization. It does so through strong policy and governance that enables stakeholders to take preventative action based on superior information.

Real-time visibility into enterprise-wide release statuses enables stakeholders to pinpoint the root cause of potential release failures and mitigate them quickly.

Plutora and Release Management

Plutora is an all-in-one, SaaS-based software tool that manages all aspects of the Release Management lifecycle. Plutora enables teams at a portfolio or enterprise team level to enforce a repeatable release framework. With Plutora, users can architect, plan, coordinate, and govern all aspects of the Release Management lifecycle.

Plutora is aligned to the ITIL v3 Release Management Process. Plutora also aligns well with the DevOps initiative, and the planning and collaboration processes of the DevOps approach.

With Plutora, one can plan releases, build the release pipeline and release scope, handover deployment plans and run books to Operations, and improve collaboration between Development and Operations.

With Plutora Release, users can architect, plan, coordinate, and govern all aspects of the Release Management lifecycle.

Every aspect of Plutora can be customized, due to its powerful admin customization features, and Plutora is integrated into third party tools and systems you may already be leveraging. Examples include CA Clarity, JIRA, ServiceNow, IBM RTC, Remedy, and many more.

Value from Plutora cuts across the organization. It also provides numerous out-of-the-box reports and executive dashboards which are parameterized. It provides relevant reporting by working with the IT organization to understand their requirements and subsequently developing reports that meet these requirements.

Conclusion

Release Managers and the Release Management function sit at the DevOps divide. We have seen how they sit between an operations group tasked with accepting, deploying, and supporting a software release. This group wants a predictable process, they view releases as incidents to be managed, and they answer to a business that wants predictability and accountability.

On the other side of the divide are the Development groups. They are tasked with creating software to be released. These groups tend to be more diverse in that they may be using bleeding-edge technology to create a next-generation website as compared to another group that may be developing more traditional applications, for example, applications based on an Oracle database. Both development and operations teams use different tools and different vernacular when they describe their respective roles.

One can truly think of Release Management as a heat shield protecting a release during the re-entry process from a DevOps orbit to a more manageable ITSM approach that the largest companies have come to expect. In this context, Release Management is critical, and it is important to establish a sound Release Management practice within the organization. A good Release Management tool such as Plutora helps organizations achieve the right balance between development and operations, and bridges the DevOps gap.

Release Management is critical, and it is important to establish a sound Release Management practice within the organization. A good Release Management tool such as Plutora helps organizations achieve the right balance between development and operations, and bridges the DevOps gap.

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