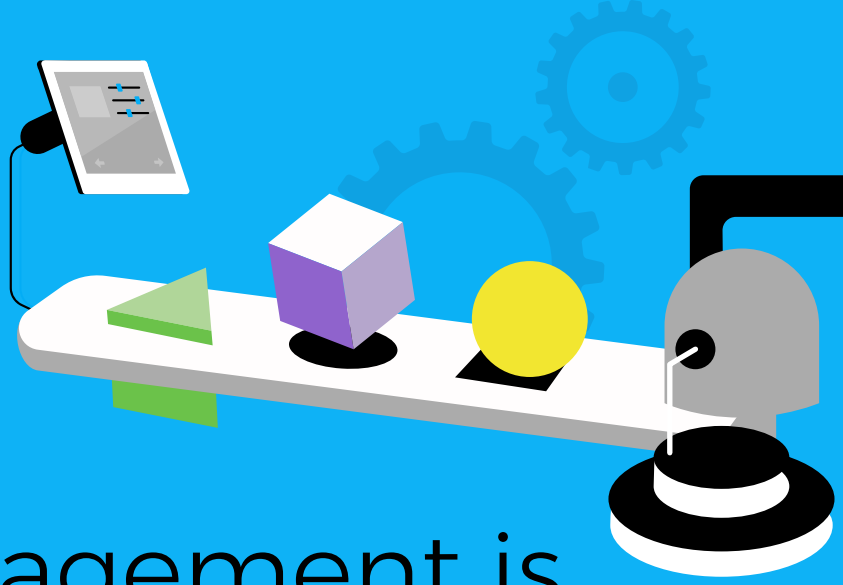
PLUTORA

WHITE PAPER

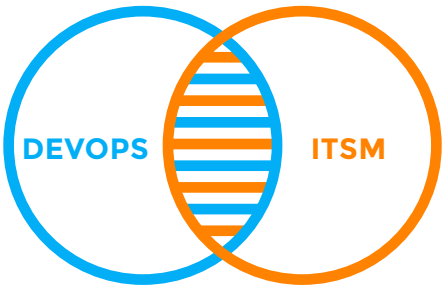


Release Management is Not a "One Size Fits All" Solution

Enterprises supporting critical software releases understand the importance of setting up a strong release management function. Without this function software releases present both unmanageable risks and complexity. While several software tools have been developed to address these risks many attempts to do so from the perspective of IT Service Management (ITSM) using procedures detailed in the IT Infrastructure Library (ITIL).

By forcing organizations into ITSM-focused approaches these tools create a rigid, unrealistic approach to release management which ultimately increases risk as teams are forced to work around release management systems that don't accurately model the reality of evolving approaches to software delivery.

Instead of adopting cookie-cutter solutions for release management based on static, flowchart-driven models enterprises need purpose-built systems designed with a model that can be adapted to the needs of an individual enterprise that incorporate components from emerging DevOps-focused approaches to release management and more standard ITSM-focused approaches to IT Service Management.



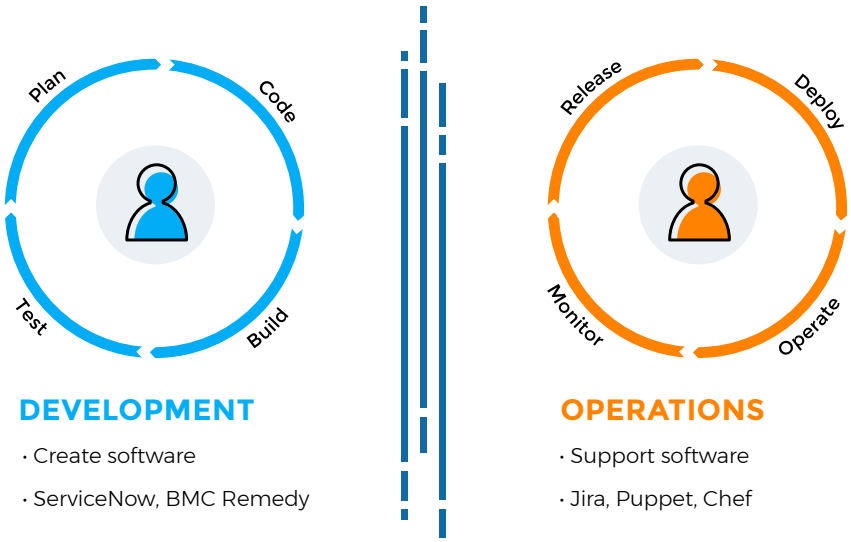
As the boundaries between DevOps and ITSM continue to evolve, tool selection plays a critical role in creating a Release Management function that can adapt and respond to emerging trends.

Release Management for Two Audiences

There's an often overlooked disconnect within the IT department between teams tasked with supporting software and teams tasked with creating software.

These teams rarely share the same terminology, and they certainly don't share the same tool sets. Some teams are using tools like ServiceNow and BMC to model IT Portfolio management and governance while other teams are managing development process with issue tracking tools like Atlassian's Jira and DevOps tools like Puppet or Chef. In an increasingly automated and tool-driven release workflow, this disconnect between these two audiences - development and operations - influences the tools and terminology different groups employ when managing a software release.

When designing a release management function for your enterprise, it is important to use tools that allow you to customize your model of software releases such that "release management" can act as a necessary bridge between competing views of the organization. Enterprises need to use tools that can interoperate with different groups in IT - a release management tool that ignores IT Portfolio Management and capacity planning is as useless as a release management tool that cannot account for source control, continuous integration, and development-focused issue tracking.



Highlights

- Release Managers bridge two audiences: a DevOps-focused view of releases and deployments, and an ITSM-focused view of service management.
- Enterprises should select tools that can adapt to both DevOps and ITSM-focused approaches as they evolve in an ever-changing environment.
- Enterprises standardizing on rigid, ITSM-focused tools force organizations into unrealistic models requiring workarounds and sacrificing agility.
- Enterprises adopting a DevOps-focused approach to release management fail to provide predictability and cost accounting to the business.
- Plutora is a purpose build tool for release management providing a flexible model incorporating both DevOps and ITSM-focused components to provide a comprehensive model for software delivery.

Two Different Groups, Two Different Models

The divisions between teams in IT fall into two well-defined categories:

Operations groups are "reactive."
They are focused on service management and incident response. Teams supporting operations tend to have a model of the organization that is more 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.

A team managing cloud infrastructure for 40 internal development groups will also view interactions with internal groups as service provider relationships with customers. To accurately track cost and forecast capacity, interactions with these groups should follow well-established patterns for service management.

Development groups are "proactive".
Developer create software systems and they often have an entirely different model of the business from 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 focused on fast-moving, highly technical systems are not often aligned with the ITSM and ITIL models at all.

Most development teams focus on software delivery, internal quality metrics leaving it to release managers to act as a buffer between an internal process and other groups which may take a more formal approach to service management.

Modeling Releases from an ITIL Perspective

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

Different services can be assigned Service Level Agreements (SLA) and incidents can be tracked in a system like ServiceNow or BMC Remedy. The availability of systems can be accurately measured and the costs can be controlled. 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 change requests which are then aggregated into a release request. These requests are then sent to a review board for approval, this board then assigns a service analyst to the requests to identify risks associated with the release process, and ultimately any further ramifications of the change. Requests are identified via detailed analysis of the outputs of a tool that tracks a release as an incident to be managed and responded to.

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

Modeling Releases from a DevOps Perspective

DevOps is a recent trend in software delivery which places an emphasis on 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 deployments and service management on a development team working closely with an operations team with developers often driving the initiative.

While there are several examples of larger, technology-driven organizations such as Facebook or Google achieving a remarkable level of success with a more agile approach to IT service management using techniques associated with DevOps, DevOps often requires both a heavy investment in development talent along with a greater exposure to short-term risk as development-focused teams often struggle to provide the business with reliable and predictable cost accounting for IT service management.

ITSM-focused systems fail to fully understand how software is developed while DevOps-focused systems fail to understand how enterprise software is supported. Release management exists to bridge this gap.

The Reality of Large Software Releases

Both the ITSM and DevOps-focused approach fail to provide the perfect combination of agility and predictability for the business. The reality of large, software-driven enterprises is that a release management function exists to facilitate continuous software releases while providing the necessary translation layer between development and operations.

The ITSM and ITIL model breaks down when you are dealing with a fast-moving technology project across several development groups which may require a release cadence that doesn't align with the built-in assumptions of an ITSM-focused standard such as ITIL.

The DevOps model breaks down when multiple teams require strong orchestration across multiple departments while providing a clean hand-off to IT groups responsible for managing incidents, providing centralized cost accounting, and providing the business with a stable interface for support across multiple departments.

A release management function exists to facilitate continuous software releases while providing the necessary translation layer between development and operations.

Release Management: Questions to Consider

When selecting a tool to support your organization's release management function consider the following questions:

- How does one model configuration changes made across several distinct teams in form-based approach to release management?
- How does continuous delivery factor into an ITIL based process? What if a software project is in a constant state of change?
- Should every change be modeled as a change request? Or should release managers create placeholder change requests?
- If several teams with different release cadences all update related systems do you need a change ticket for each system?
- What about tiered releases or tests? If you are ramping up traffic between an old and a new system, does each event require a ticket?
- What if you need to perform an emergency fix to production, immediately without the direct involvement of several levels of management?

The answers to these questions are varied, but the fact is that a large enterprise that attempts to use an ITIL-based tool to manage software releases across both dev and ops groups creates a situation where models don't fit reality.

ITIL-based tools such as ServiceNow and BMC Remedy are entirely disconnected from the models and procedures used by developers to create software. Using an ITIL-focused tool such as ServiceNow or BMC Remedy forces the entire organization to view the most complex release orchestration events through the lens of standards that weren't designed with today's complexities in mind.

Trends in Software Delivery

The reality of modern software development and delivery mean a few things.

Software projects are increasingly agile

The term "Agile" is often overused — what does it mean in the context of release management? While "Agile" is a development process, it can also translate to software projects that are delivered more frequently with less ceremony surrounding a given release.

This trend can be developed to a point where a large software system is pushed to production multiple times a day, and these are systems that are commonplace in these organizations, such as Google and Facebook. These companies push mission-critical systems to production every day in a way that would make the use of heavy ITIL-based release management tools impractical.

Deployments are increasingly self-service

This means that developers and development groups have direct access to deploy code to production. This is a dramatic shift made possible by the ease with which developers can now provision virtual machines in cloud environments.

If your organization uses an internal cloud on OpenStack, your developers may also have direct access to tools that allow for self-service configuration management and software deployment. In these organizations, release managers are an important bridge between self-service deployments and supportable service delivery.

Releases are increasingly distributed

Take a large e-commerce or social networking website as an example and you'll quickly see that a software release requires the coordination of several independent systems.

As more and more enterprises move to a server-oriented architecture software, releases have become less amenable to modeling as a single release event.

Emerging Trends & New Realities

Increased Agility

Releases are no longer "exceptional" events, there is a constant cadence of change which makes strict adherence to ITSM difficult.

Self-service Releases

Developers and development groups are more empowered than ever to manage software releases making oversight and accountability more challenging.

More Distributed

Today's software releases often involve the entire organization. Creating a change requests in an ITSM-focused tool doesn't begin to capture the challenges of enterprise-wide release orchestration.

As more and more enterprises move to a server-oriented architecture software, releases have become less amenable to modeling as a single release event.

Modern Software Releases Demand Flexibility

Large enterprises with several development groups, agile development practices, and frequent releases are finding themselves in a continuous state of releasing software. It is this always on nature of software releases that is most incompatible with ITSM models. To read the ITIL standard is to understand the model of software delivery from nearly two decades ago: when software releases were rare events to be managed with process and ceremony.

Modern software developers are dynamic and distributed and large organizations are effectively managing the risks of software releases without adopting strict ITSM guidelines across the entire effort. If an enterprise is looking to adopt a mixed array of IT standards it needs to adopt a tool that can be tailored to fit the individual idiosyncrasies that develop in any large organization.

Whether you have multiple development teams using slightly different strains of Agile development or teams that have adopted procedures more aligned with ITSM, your release management function and the tools you adopt to support it need to be able to adapt and evolve with your organization.

Plutora: Purpose-built for Release Management

Plutora is a purpose-built tool designed to support your release management process that can be adapted to work with a wide variety of tools. Plutora doesn't dictate a standard and it won't force you to adopt an antiquated, process based on a flowchart from an outdated standard. It is a tool that can provide a comfortable interface both to teams that expect to see ITSM-based standards and to other teams that might not even know what ITSM stands for.

This is the challenge today. A large, software-driven enterprise has so many different actors, different groups collaborating with one another to produce quality software. Some of them understand the business from the perspective of ITSM and it is important to provide an ITSM-aware interface to those groups. Other groups have little patience for form-driven, ticket interfaces of ServiceNow and BMC Remedy as they release to software releases. In these organizations Plutora acts as a necessary bridge.

We allow different teams to use the tools they are comfortable with and we facilitate the creation of a Release Management function which can bridge the dynamic, creative qualities of a development team to the more predictable, accountable standards that IT demands.

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