ACCELERATING AGILE DELIVERY WITH ITSM INTEGRATION

IT leaders are increasingly asking how their teams can speed service and applications delivery, rolling out new and cutting-edge technologies to their user communities at breakneck speed. This mandate, providing cost-efficient and value-added services, demands that CIOs work diligently to improve collaboration within their developer teams while striving to reduce the developer's overall level of effort.

Accelerating any development delivery demands that Development and Operations teams execute their tasks with greater synchronization, less manual effort and fewer hand-off's. This sounds easy and intuitive; however, the process slows down because developers are often required to input and maintain similar types of information in both development and defect management tools (e.g., Jira and Rally), in operational tools (e.g., IT Service Management platforms) and may also be required to hand off the information to the operations team to administer. As a result, the development process slows significantly and becomes very repetitious and tedious to execute. Additionally, information common to both Agile and ITSM platforms (e.g., development or defect lifecycle information needed within change or problem management tasks) is often not updated or synchronized due to the significant time and effort required to manage work across multiple systems. Hence, to improve the speed of delivering applications results to the enterprise while reducing operational risk, these duplicative and sequential efforts need to be reduced or eliminated.

I will share in this blog that this is best achieved by:

- Enabling application developers to focus on their agile development work in tools they are familiar with; AND
- Integrating those tools with operational service management capabilities.

As you integrate these capabilities, your development and operations teams will benefit from the increased agility you achieve within your end-to-end Dev/Ops lifecycle.

To arrive at this seamless, agile state, I will illustrate my recommended three step best practice that you can begin using right way. These steps are:

1. Identify, prioritize and select the use cases that support your required outcomes;
2. Define and design the use case details (e.g., triggers, data, tools, etc.) for your integration; and
3. Implement using existing API's or integration tools.

Let's get started with the first step.

Identify, prioritize and select the use cases that support your required outcomes

While many IT leaders and their teams have an awareness of where they struggle and where they can improve with Dev/Ops integration, the ability to gain the greatest and lasting benefit begins with clearly identifying and prioritizing the outcomes and use cases needed to achieve success. There is no single use case or prescriptive set of priorities that applies to all the potential interactions and integrations between Development and Operations. Dev/Ops integration needs are triggered in numerous development and operational scenarios with some common, bi-directional use cases.

Operationally, within ITSM these cases are supported by Incident Management, Change Management and Problem/Defect Management and may require initiation from either the Service Management platform or from within the development lifecycle and the Agile development capabilities. In virtually all cases, some level of bi-directional integration capability is necessary to accomplish the desired outcome.

Below, I describe the use cases most often encountered when integrating Development and Operations using ITSM capabilities. These common use cases should be of assistance in the identification and prioritization of Dev/Ops integrations for your company's needs:

- **Request and Incident Dev-Ops Cases** – Requests for new development and enhancements from the business occur constantly. Development tools do not provide an enterprise level digital catalog and request capability that enables a requestor to initiate and track the status of their requests along with all other work they request from IT. Utilizing a tool like BMC Helix Digital Workplace enables development requests to be managed like all other IT requests. What follows are a couple of use cases emerging that integrate the request process for development work between BMC Helix Digital workplace and a set of development tools (Jira):
    - **Use Case 1: Create Jira User Story from a Digital Workplace Request**
    - **Use Case 2: Jira User Story Updates – update the originating Request Progress**

- **Change and Development System Use Cases** – New development is often initiated from within the development cycle in the form of Agile "User Stories," or other identified development triggers and events. Developers initiate and track this work within their development tools such as Jira or Rally as a part of the Agile development cycle. Yet, much of that work also needs to initiate Change Records to ensure the proper levels of rigor and compliance are applied. As a result, information that is tracked in the development tools must also be established and synchronized with change records, thus requiring duplicate effort and lost time. The following use cases that support integration between Change Management and

development tools, such as Jira, can significantly reduce the amount of time spent managing change records and tasks as well as ensure change compliance and risk requirements are fully addressed with reduced effort:
    - **Use Case 3: Jira Development project needs to initiate a change request CRQ**
    - **Use Case 4: Jira Project needs to update a CRQ/Task based upon defined development activities**
    - **Use Case 5: Operational updates within Change trigger actions in Jira**

- **Problem/Defect Management Use Cases** – New problems are often identified in both operations and within the development and release cycle as "Defect" backlog for developers to work. For new problems identified in ITSM operations, this requires creating new defect records and then tracking constant updates to those records back in the parent problem record. Conversely, defects found during application/service testing and not fixed need to be established as problems and potentially known error records within ITSM. Managing all these records and the constant updates is very time-consuming. If done poorly, IT operational teams may not have visibility into known problems or errors. Teams are demanding a better way of streamlining the information between problem management and defect management to minimize manual efforts and ensure information is current within both systems. The use cases supporting the integration with problem management and defect management that help minimize work and increase visibility to defect work include:
    - **Use Case 6; A Problem Record is raised. Create Problem Backlog Record in Jira**
    - **Use Case 7: Problem management monitors Jira for Updates to problems**
    - **Use Case 8: On Demand Update to the Problem Record**
    - **Use Case 9: New Defects found during development testing (not addressed) – are transferred to backlog and need a Problem/Known error (and possibly workarounds) created.**

Define and Design the Use Case Details

Once the use cases that provide the most value to your business have been agreed upon, you should design the detail that will deliver the integration. This includes defining the data that will pass back and forth, the timing and events within the process, and the tool that triggers the execution of the automation/integration. This will also help you identify the best tools to support the integration described in the last step below. We recommend taking a rapid, Agile development approach to the integration design and starting with designing the initial set of attributes and capabilities needed. Lastly, I recommend executing in your integration design using incremental sprints with regular check-ins.

Implement the integrations

To implement your well-defined requirements, look to your existing, out-of-the-box integrations within your BMC Helix ITSM. Remedy provides foundational support for some of the basic use cases such as items 1 and 2 listed above. Rest API's are also available within Remedy to enable direct integration where the design requires. In addition, BMC's tools such as TrueSight Orchestration and Control-M can help you deliver integration outcomes. For unique use cases that aren't delivered with out-of-the-box capabilities, BMC Helix Platform provides capabilities to quickly build and enhance integrations as needed between Remedy and your Agile development platforms to meet and extend your unique outcomes.

Hopefully, through this blog it is clear that as technology advances, IT leaders are more easily able to evolve their Dev/Ops capabilities into a single, integrated approach. This means IT can streamline time to market as well as focus developer efforts on delivering rapid outcomes. Additionally, this can all be done while ensuring risk and compliance policies are fully met. Even further, with Dev/Ops integrated in this way, Developers are then free to use their preferred development tools and minimize duplication of effort in multiple platforms. Once defined and prioritized, implementing the Dev/Ops the use cases I've reviewed here can be done leveraging existing integration to help reduce the manual, often tedious work required to maintain data. Finally, by arriving at this integrated Dev/Ops approach, your development and operations teams can work more seamlessly and a much faster pace.

If your organization could use assistance with planning and transitioning to a better integration of Dev/Ops, please fill out our form to be contacted by a Consulting expert to discuss your needs.