Collaborative DevOps with Rational and Tivoli

Overview

This paper describes the challenges that exist between development and operations and how integrations between products from IBM Rational and IBM Tivoli support effective collaboration to achieve improved accuracy, efficiency, agility, and security in the deployment and monitoring of software systems. The scope of the paper spans the areas of strategic planning, deployment planning, automation, and the identification and remediation of production problems.

Background

In virtually all modern organizations, software has become a critical component in achieving strategic objectives and competitive advantage. Internet-enabled connectivity has led to the creation of new business models and has transformed how we collect market intelligence, market and sell products, and manage overall business processes. Recent history has shown that organizations who fail to effectively utilize and manage software simply cannot compete. An IBM CIO study of hundreds of companies revealed that a number of organizations are struggling to just get their software into production consistently. In fact, 50% of deployed applications must be rolled back, with rework accounting for more than 30% of project costs. Reversing this trend requires improvement not only in software development and operational management practices but also in optimizing the collaboration and processes that span these two domains.

Why do so many deployments fail?

A key factor that has led to the high level of deployment failures we see today is a corresponding increase in software and infrastructure complexity over time. Applications today must adapt to an increasing number of business rules and constraints such as compliance mandates and disaster recovery requirements. They usually have numerous interdependencies with other business applications within the IT infrastructure, which itself often spans a large number of distributed environments across multiple geographies. Complex network infrastructures and comprehensive security and monitoring management systems only add to the complexity, as do the architectures required to support the high levels of performance and scalability that business consumers have come to expect. Overcoming all this complexity requires a tremendous amount of coordination and communication, yet the two teams who need to collaborate the most- development and operations- are typically not even managed within the same reporting structure.

What are the key factors to consider when linking development and operations?

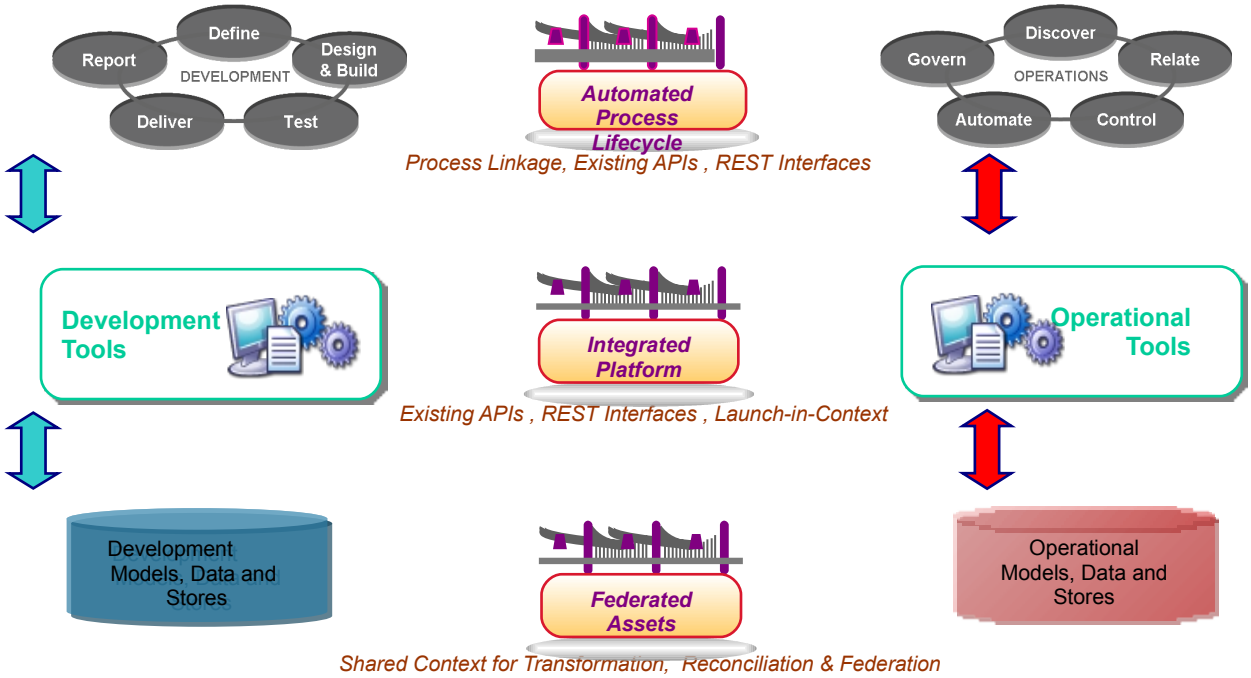
Effective collaboration between development and operations occurs as a combination of three fundamental dimensions. First, process control must be established to clearly define transitions and responsibilities between teams. Second, tools must be instrumented to support more effective collaboration and ensure effective communication across domains. Third, data must be kept accurate, up-to-date, and consistent between the two perspectives to ensure that plans and designs based on that data are not flawed.

Process level integration refers to steps in a process where control transitions between teams. A good example of process level integration is seen in the collaboration across development and operations required for problem identification and resolution. When a problem is identified in production that is caused by a software defect in an application, control must be passed from the operations team to the software development team for defect resolution. Operations must also be aware of what business impact the potential outage will have across the organization so that end users can be notified and provided contingency plans. Once a fix is available and tested, control returns to the operations team to deploy the fix into production and resolve the trouble ticket.

Tool level integration occurs when one tool leverages an interface provided by another to extend the user experience beyond the capabilities provided by the tool itself. The interface may be tool-specific or more preferably based on an open standard specification such as Open Services for Lifecycle Collaboration (OSLC) which will be discussed in more detail later in this paper. From the previous example, tool level integration enables a help desk operator to create and track a defect that resides in a defect tracking tool without leaving the service desk user interface, ensuring more effective collaboration across the two domains.

Data level integration allows information to be shared or linked in a federated manner. For instance, a software release from a development perspective may include quality results and proof of compliance, whereas the same release from an operational perspective may be more focused on usage statistics and installation details, yet both perspectives share some common information about the release as well. The federation of this information allows updates and impacts to be shared to ensure consistency and accuracy between development and operations and help minimize outages.

Figure1- Integration tiers



By integrating processes, tools, and data across development and operations, organizations are not only able to increase their success rate in deploying applications but are also able to more effectively optimize their infrastructure architecture and respond more quickly to production problems related to software defects. Infrastructure inventory information once locked within operations is now available to be leveraged to develop both strategic roadmaps and application-specific deployment architectures. Meta-data captured in design activities is made available to drive automations. Tracking information that once disappeared between control handoffs is made visible within a broader context, making cross-team activities more efficient.

The boundaries between development and operations may be best understood by examining the various capabilities involved in cross-team activities and the relationships between them as seen in Figure 2. This paper will focus specifically on the relationships highlighted in orange which represent integrations between Rational and Tivoli.

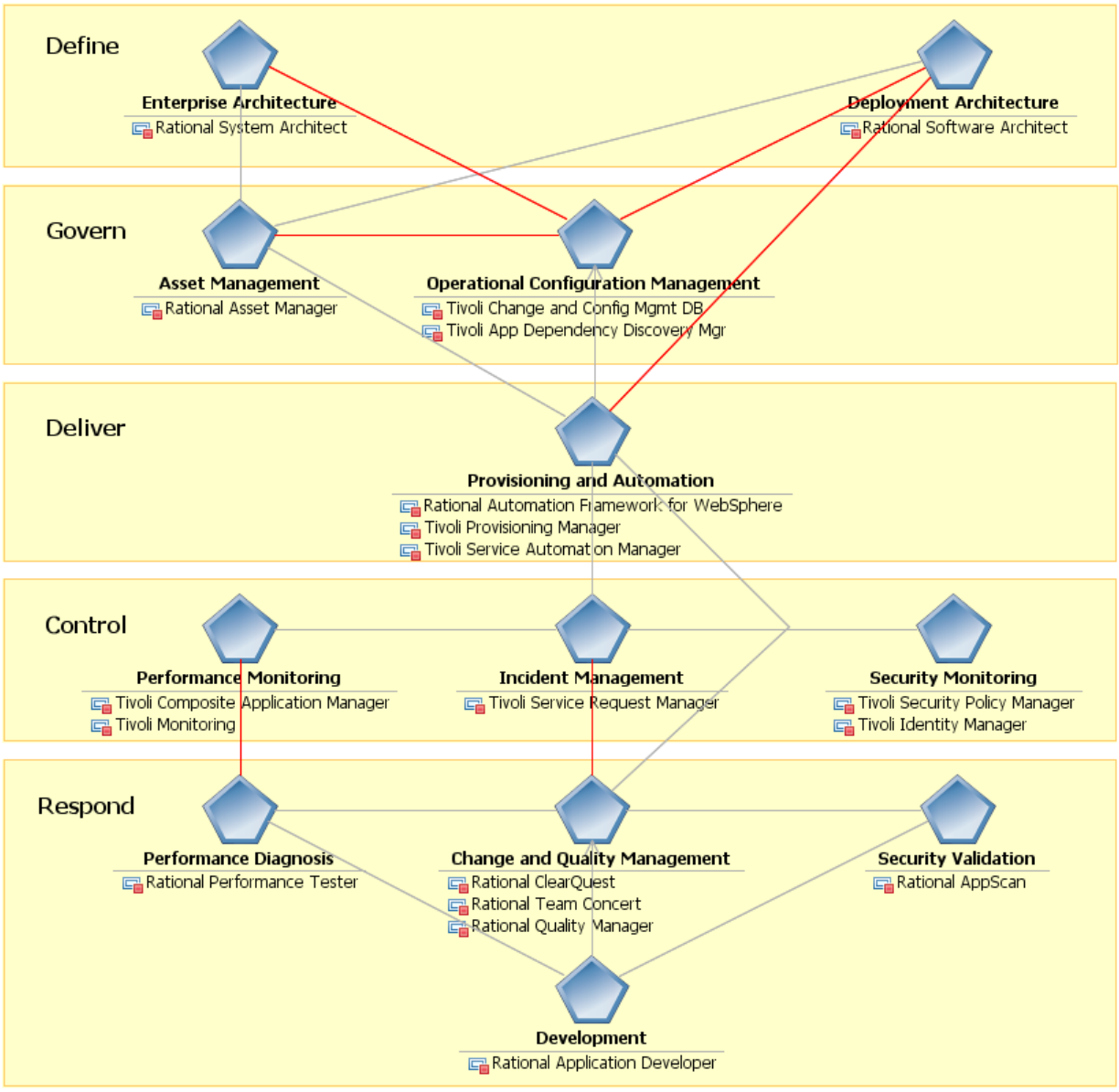


Figure 2- Capabilities Related to Collaborative DevOps

Architecture

Traditional approaches to integration, such as the creation of a single repository or point-to-point integrations, fail to scale well due to performance and complexity issues and are often not feasible in large enterprises. Not only that, but tight interdependency caused by proprietary APIs leads to vendor lock-in and upgrade issues that many organizations with heterogeneous environments reject. Rational and Tivoli are moving toward more flexible integration architectures where tools leverage open standard interfaces to communicate rather than proprietary API's.

In 2008, IBM started the Open Services for Lifecycle Collaboration project (http://open-services.net), a multi-vendor effort to help define specifications where resources can be loosely-coupled. Each OSLC resource must have a universal address that can be referenced using a Uniform Resource Locator (URL), and services must be accessible via RESTful web protocols. The content of each resource must conform to a format defined in the relevant OSLC specification, allowing tools to access the information without being tightly-coupled. OSLC provides a mechanism by which information is easily accessible through Internet protocols and eliminates the need for point to point integrations.

Tools that realize or consume OSLC interfaces do not require knowledge of other tools with which they may integrate. For example, a tool may call the Change Management OSLC interface to log a defect in a defect tracking tool without knowing which defect tracking tool will fulfill that request. Any defect tracking tool that realizes the Change Management OSLC specification can then be registered as the provider of Change Management services. This loose coupling allows for unprecedented extensibility and scalability, enabling a wide variety of integration scenarios.

Current OSLC specifications of particular interest to DevOps include Change Management, Asset Management, Architecture Management, and Automation. Additional specifications may be added as integrations evolve and mature over time.

Collaboration in Strategic Planning and Transformation

Enterprise architects play a key role in driving strategic transformation. By modeling the as-is and to-be states of the enterprise, they identify the solution components necessary to realize strategic objectives and then establish roadmaps to incrementally implement the changes needed. Understanding the current state of the enterprise is critical to building and maintaining effective enterprise architecture, yet infrastructure configurations are rarely static. Enterprise architects need to leverage automated discovery mechanisms to harvest and synchronize configuration information from the live infrastructure in order to ensure that the decisions they make are based on up-to-date information. They also need to know how various items in the infrastructure are being used within the enterprise and by whom (or what) so that transitions can be planned and managed with minimal disruption to the business.

Similarly, infrastructure managers need to understand the business alignment of the operational systems that they maintain and easily assess and respond to changes in business priorities. Without this perspective, their ability to effectively prioritize their efforts and implement appropriate optimizations is limited.

To meet the needs of enterprise architects and infrastructure managers and to improve the collaboration between the two roles, IBM has developed an integration between Rational System Architect, which supports enterprise architects in strategic planning, organizational impact analysis and business optimization, and Tivoli Change and Configuration Management Database(CCMDB), which serves as the authoritative source for the inventory and the status of applications, products, technologies, and computing platforms. The integration also leverages Tivoli Application Dependency Discovery Manager (TADDM), which performs automated discovery of infrastructure elements and dependency mappings. The solution leverages live configuration and dependency information discovered by TADDM and stored in CCMDB to populate the infrastructure architecture model in Rational System Architect, enabling improved decision-making and alignment and reducing the risk of business disruption during transformation activities.

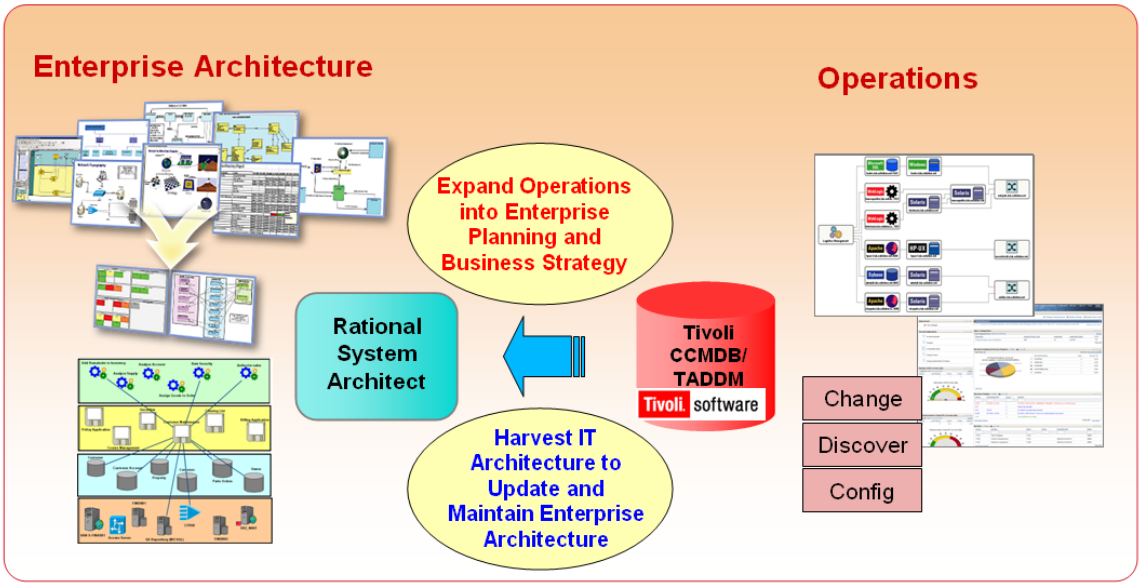


Figure 1- Integrating Enterprise Architecture and Operational Configuration Management

The impact of this integration on the business appears in several areas. First, confidence in strategic decisions improves as a result of more accurate data and more timely data collection from the existing operational environment. This translates to a reduction in course corrections and delays which can stifle momentum and waste valuable time and resources. Next is a reduction in the risk associated with transformation activities. Users and system owners which depend on infrastructure that needs to be migrated are easily identified and notified prior to implementing any changes. Additional risk reduction is gained by querying the enterprise architecture to identify gaps in compliance, such as which servers lack adequate disaster recovery or contain sensitive customer data.

The demand for this integration came from organizations that were challenged with understanding their existing application portfolio and, as a result, found it difficult to make good investment decisions in their IT infrastructure based on business drivers. The integrated solution is helping them to accurately identify opportunities for cost reduction and innovation, retire applications that add little business value, and react more quickly to changing market conditions.

(NOTE: This integration is currently available via a reusable service asset, configured for unique client environments.)

Collaboration in Deployment Planning and Automation

Planning and deploying a release into production requires a high degree of accuracy and more often than not results in a rollback to a previous version due to errors in the configuration of the application. Software architects, deployment engineers, testers and release engineers need an accurate and consistent understanding of the target deployment environment(s) for a release to be successful.

Deployment engineers are also challenged in deploying a full software release as a single automation execution. Typical deployment automations stop short of full deployment and require a combination of automated and manual configuration steps, which introduces risk into the process. They need to leverage automation at both the infrastructure and application layers to make frequent releases more consistent, complete, and timely and less error-prone.

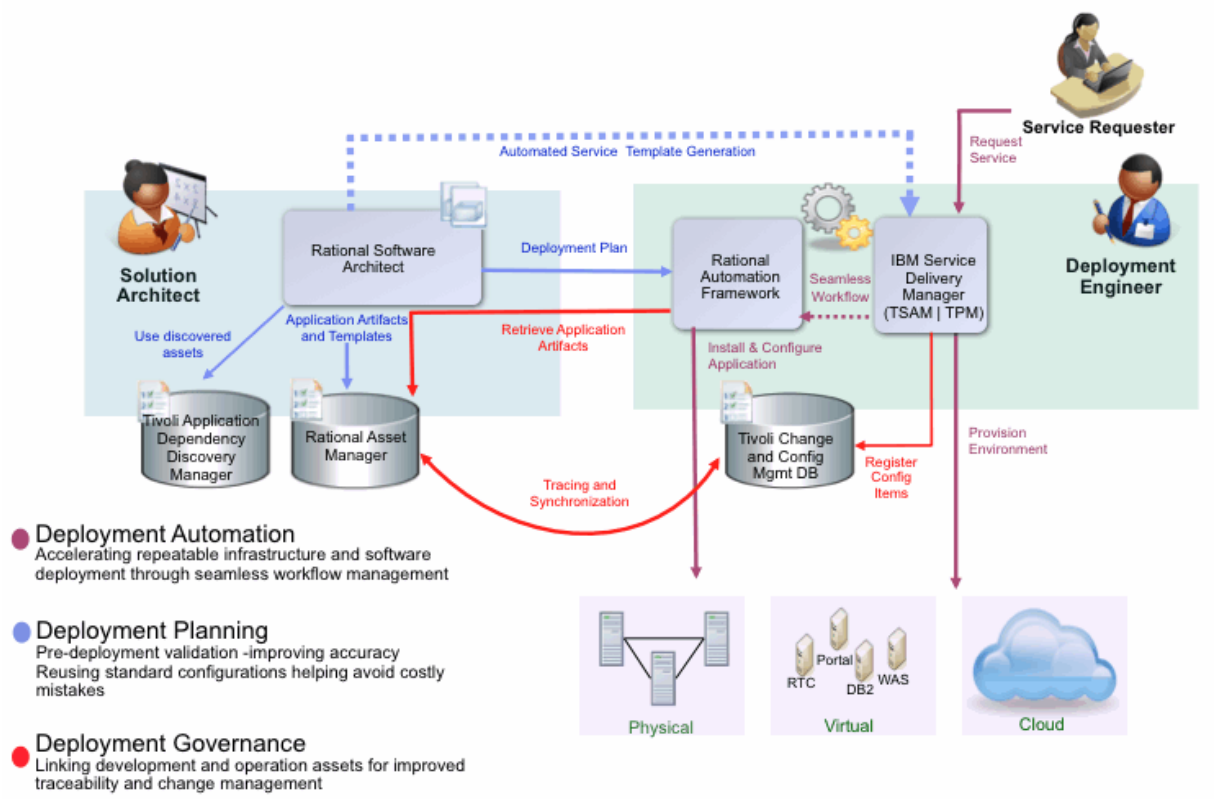


Figure 2- Integrating Deployment Planning and Automation

To achieve consistency across the enterprise, enterprise architects or IT architects can specify standard templates using the Rational Software Architect Extension for Deployment Planning. These standard templates codify and prescribe the approved deployment patterns and configurations for the enterprise. They can govern and share these templates with various teams using Rational Asset Manager. By applying these templates to deployment plans, solution architects can ensure that their deployment plans align with enterprise standards.

To improve the accuracy of deployments into existing environments, IBM has integrated the Extension for Deployment Planning of Rational Software Architect (RSA), which is used by software and solution architects to design the deployment topology of an application, with Tivoli Application Dependency Discovery Manager (TADDM), which performs automated discovery of infrastructure elements and dependency mappings. From within RSA, discovered content from TADDM, such as servers, networks, L2 interfaces, IP interfaces, operating systems, and installed software, is used as a basis to plan a deployment. An architect or deployment engineer then refines the model to reflect the installation of new or changed release components, which are managed in Rational Asset Manager, as well as required middleware or platform adjustments to the existing environment. The configuration information captured in the deployment model is validated using the rich semantic domains of the planning tool as well as the standard enterprise deployment templates to ensure it is consistent. Once validated, the configuration data can be exported to a deployment specification document and/or automation engine for purposes of executing the deployment.

To make releases more consistent and complete and less error-prone, deployment engineers need to leverage automation at both infrastructure and application layers. IBM has integrated Tivoli Provisioning Manager, which automates the provisioning of infrastructure items, with Rational Automation Framework for WebSphere, which automates the configuration of middleware and deployment of applications and application artifacts, to support the automation requirements at both layers. This is a key requirement not only in simplifying the deployment of traditional production environments but also for Cloud environments where self-service provisioning is common. Deployment engineers are able to automate the creation of service definitions using the updated configuration information is then pulled from the model using the Rational Software Architect 8.0.1 Extension for Deployment Automation where it is leveraged to generate a service definition in Tivoli Service Automation Manager (TSAM) and automation workflows in both Tivoli Provisioning Manager and Rational Automation Framework for WebSphere.* The service definition in Tivoli Service Automation Manager defines the overall deployment plan which orchestrates the infrastructure-related automation workflows in Tivoli Provisioning Manager with application configuration automation workflows in Rational Automation Framework for WebSphere to support the complete, end-to-end provisioning of a software release.**

The reason that this solution is important to the business is that every delay in getting a software system deployed carries a lost opportunity cost or a financial risk. The business case for which the software was built cannot be realized until the system is in production, and if the deployment fails due to inaccuracies in the configuration or is delayed due to manual processes, the business cannot benefit from the new capabilities and may, in fact, suffer financially if the failure or delay impacts business continuity or compliance.

Demand for this set of integrations came from organizations with complex manual deployment processes and, as a result, high deployment failure rates. Many currently manage their configuration specifications in spreadsheets and find it difficult to keep the information current. Several report that ‘the last mile’ of the deployment of complex applications can take several months. This integrated solution helps to reduce the cycle time of a release to a matter of days or even hours.

**The ability to create a service definition archive from RSA and import into TSAM is possible with the IBM Deployment Planning and Automation for the Cloud package (http://www-01.ibm.com/software/brandcatalog/ismlibrary/details?catalog.label=1TW10TS02) available on the Integrated Service Management Library (ISML).*
***The integration is made possible with the Build Forge Automation package (http://www-01.ibm.com/software/brandcatalog/ismlibrary/details?catalog.label=1TW101090) available on the Integrated Service Management Library (ISML).*

Collaboration in Problem Identification and Resolution

Any production outage of any duration can have a direct, negative impact on a company's revenue. Outages related to software defects carry additional risk due to the coordination of resources required to identify, analyze, and correct the problem, as well as to test the fix and deploy it into production without breaking dependent applications or components. Software developers, performance engineers, release engineers, and operational staff need to effectively collaborate and share information to resolve these issues in a timely manner that minimizes disruption to the business.

To optimize the communication between development and operations to resolve production issues related to software defects, IBM has developed an integration between Tivoli Service Request Manager (TSRM), which is a service desk application, and Rational ClearQuest, which is a defect tracking system. If the support staff suspects that a production issue is caused by a defect in the software, she can search for the reported defect in ClearQuest via the TSRM user interface to determine if the problem has already been reported. If she does not find it, she can log a new defect in ClearQuest, relate it to the trouble ticket, and monitor its progress, again using the TSRM user interface.

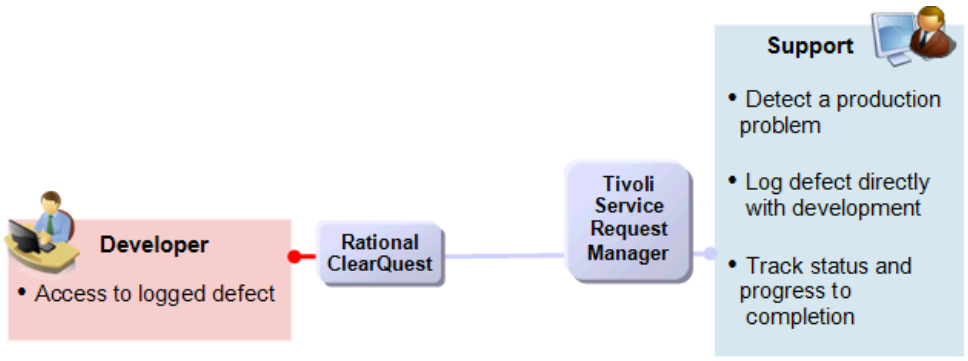


Figure 3- Integrating Incident Management and Defect Tracking

Some software defects are discovered while an application is being monitored for performance. To accelerate the analysis of these defects by developers, IBM has developed integration between IBM Tivoli Composite Application Management (ITCAM)/IBM Tivoli Monitoring (ITM), which are used to monitor applications and detect performance problems, and Rational Performance Tester (RPT) which is used to test software performance and diagnose performance issues. When a production problem is first detected within the production environment, ITCAM and ITM package the logs and attach them to a new TSRM trouble ticket, which, as we saw above, can flow to the development team via a new defect in ClearQuest. To understand the nature of the performance problem and isolate it to an area within the source code, a developer imports the ITCAM or ITM data into Rational Performance Tester (RPT) and runs an analysis of the problem which leads him to the area of code where the problem exists. The developer may also import resource monitoring data from ITM to help further diagnose the problem. Furthermore, resulting RPT test scripts can be uploaded to ITCAM for Transactions to help verify the ongoing availability and responsiveness of the production system (RPT is an optional install component of ITCAM for Transactions).

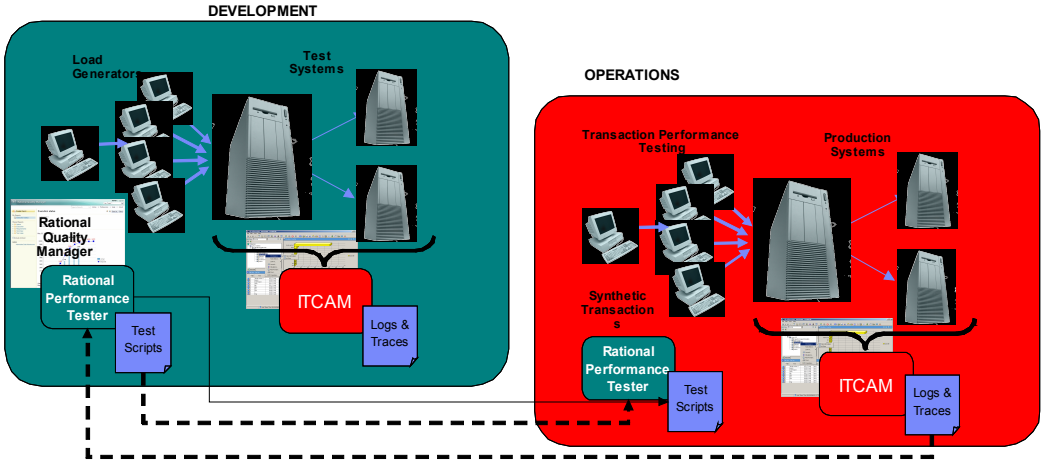


Figure 4- Integrating Performance Monitoring and Diagnosis

Understanding the full context of a particular release of software can further accelerate the resolution of a production issue and help maintenance developers avoid creating fixes that are inconsistent with the original architecture. Likewise, understanding the production dependencies of a deployed release is needed to avoid undesirable disruption to affected systems. To support these critical aspects related to problem resolution, IBM has developed an integration between Rational Asset Manager (RAM) and Tivoli Change and Configuration Management Database (CCMDB) which allows navigation between the development and operational perspectives of a software release.

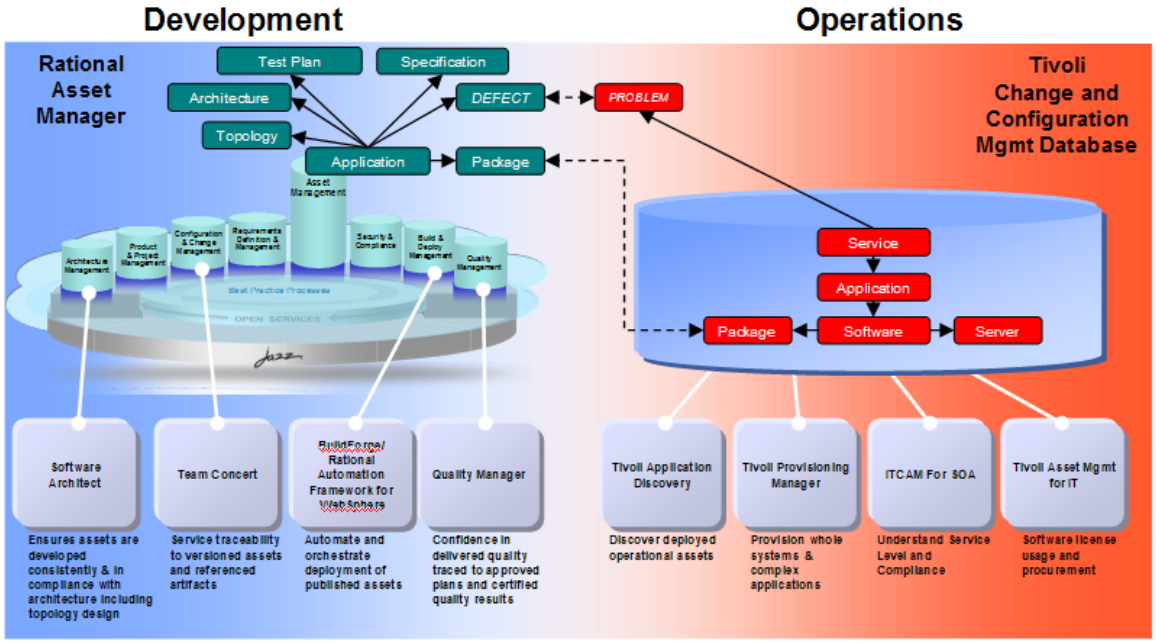


Figure 5- Integrating Release Asset and Operational Configuration Management

This integration not only accelerates analysis activities but also serves as a governance mechanism to ensure that only approved releases are available for deployment. Rational Asset Manager (RAM) becomes the Definitive Media Library (DML) which serves as the authoritative source of all software configuration items, including licenses and documentation. Tivoli Release Process Manager, which is part of CCMDB, can be configured to use RAM as its DML. Once configured, release assets in RAM can be used to generate corresponding configuration items in CCMDB which are then updated to a "ready" status. Using the CCMDB, the operator can determine exactly which versions of the modules were causing the error condition throughout the entire operational environment and where the software module must be updated to help avoid the current and future performance problems.

Customers who have implemented this solution report improvement in resolution time of defects, especially performance defects, and a reduction in defects related to fixpack dependencies. These improvements are attributed to improved collaboration, efficiency, and control provided by the integrations.

Collaboration in Securing Enterprise Software

Software security vulnerabilities pose one of the most significant risks for an enterprise that depends on software to run its business. The consequences can include loss of intellectual capital and brand value and can even lead to legal consequences. Securing a software application requires proactive and reactive measures at all stages of the application lifecycle, from initial planning through deployment and operations/management of the deployed application in production.

Threat models created during application design can be mitigated in a variety of ways, ranging from application design and implementation (guided by Rational design tools and tested using Rational Security testing tools) all the way to being deployed in a computing environment that contains security-related protections and monitoring using Tivoli Security offerings. Threats that are not mitigated in application development are identified early so that they can be accounted for during deployment and operations phases of the application's implementation.

Continued Investment in Collaborative DevOps

IBM is continuing to invest and innovate in the area of Collaborative DevOps. For example, we are actively working with customers in the automotive sector to keep software configurations in cars up-to-date with changes in design specifications. We expect these types of challenges to become more important with the proliferation of embedded devices required to enable a Smarter Planet. For similar reasons, we are continuing to develop more efficient ways of creating and managing cloud environments to accelerate adoption and reduce operating control costs. In all of these endeavors, we strive to leverage a sound architectural approach and a deep understanding of our customers' challenges to drive innovation and synergy between Rational and Tivoli that brings value to our customers.

