Manage services throughout their lifecycle by federating service metadata
White paper
June 2007

Rational® software

Federated metadata management with IBM Rational and WebSphere software.

Marc-Thomas Schmidt, chief architect ESB and WebSphere Service Registry and Repository
Grant Larsen, STSM, chief architect – asset management

Contents

Introduction

IBM believes that it's important to have different repositories to address different communities. And we acknowledge the need for a federation of information among those repositories—making common content available in the appropriate form to your enterprise's different communities.

This paper illustrates the principles of the IBM metadata management strategy and how these principles are applied in the IBM Rational® Asset Manager development repository and the IBM WebSphere® Service Registry and Repository run-time service metadata registry and repository.

IBM metadata management strategy

IBM offers a number of metadata repositories, each targeted toward a specific user community, each supporting the tasks and processes that are important to that community, and each offering the content relevant to those tasks and communities. Examples are:

- *Repositories used in the development space, such as IBM Rational Asset Manager software.*
- *A repository for managing service-oriented architecture (SOA) run-time-relevant metadata such as IBM WebSphere Service Registry and Repository software.*
- *A repository for managing system and service management metadata such as configuration management database (CMDB) software.*
- *A metadata repository for addressing the needs of an information management audience such as the IBM WebSphere Information Server.*

Highlights

Managing reusable assets for development

A development-time asset repository manages assets that are relevant to development roles, such as the roles of technical managers, analysts, architects, developers and testers. This type of repository governs assets as they are submitted, categorizes assets, provides access control to assets and measures the activity level of assets in terms of their usage. Rational Asset Manager software is a development-time asset management repository delivering these capabilities, and more.

An asset has several major characteristics:

An asset has artifacts, relationships, classification and other items to define it.

- *It is a collection of one or more artifacts—files, binaries, models, tests.*
- *It has relationships to other assets—dependencies, aggregation.*
- *It includes classification—tagged values and terms to classify assets.*
- *It describes usage measurements—who is using the asset, what defects it has.*
- *It enables access control—who can do what with the asset.*
- *It maintains policies—descriptions of proper structure and content.*

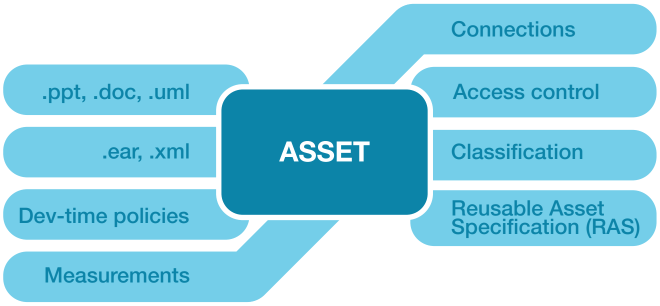

Figure 1. Assets have many characteristics, including relationships to other assets.

Highlights

Examples of assets include applications, components or more fine-grained tools like business process models, services, patterns or a Microsoft® Word design document template. This extensible structure for assets permits enterprises to specify the nature of their high-value assets and to specify how to manage them. Enterprises can specify architectural structures and control through extensible asset types and other mechanisms, which are then managed and enforced by Rational Asset Manager software.

SOA benefits are realized by separating service interface metadata from service implementations.

Managing service descriptions for dynamic SOA

Service-oriented architectures (SOAs) offer the promise of business agility and resilience through reuse, loose coupling, flexibility, interoperability, integration and governance. These advantages are realized by separating service descriptions from their implementations, and by using this descriptive metadata—metadata about service interfaces rather than implementations—across the service lifecycle.

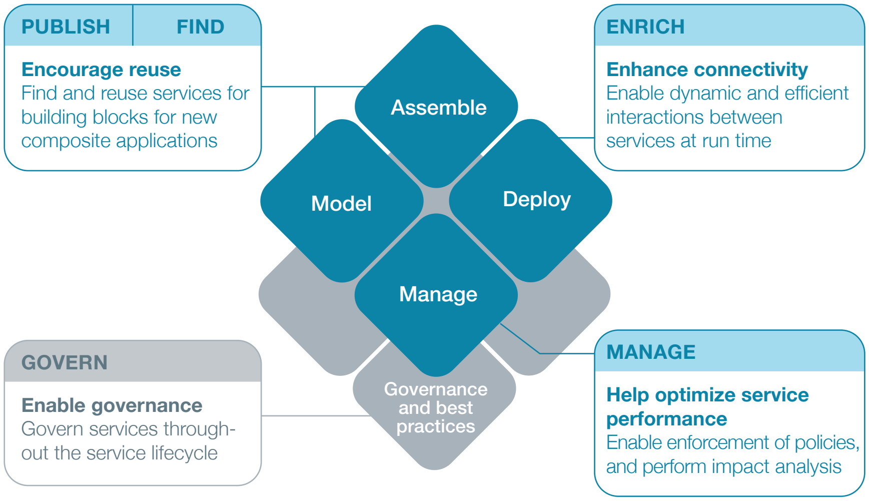
PUBLISH
FIND
Encourage reuse
Find and reuse services for building blocks for new composite applications
ENRICH
Enhance connectivity
Enable dynamic and efficient interactions between services at run time
Assemble
Model
Deploy
Manage
Governance and best practices
GOVERN
Enable governance
Govern services throughout the service lifecycle
MANAGE
Help optimize service performance
Enable enforcement of policies, and perform impact analysis

Figure 2. Service metadata is used throughout the service lifecycle to enable reuse, loose coupling and other benefits.

Highlights

A service registry and repository manages standards-based service metadata, such as Web Services Description Language (WSDL), XML Schema, Web Services Policy (WS-Policy) or Service Component Architecture (SCA) documents, that capture the technical details of what a service[1] can do, how it can be invoked or what it expects other services to do. Semantic annotations and other metadata can be associated with these artifacts to offer insight to potential users of the service on how and when it can be used, and what purposes it serves.

Service metadata is used in each of the phases of the SOA lifecycle.

This service metadata is used by analysts, architects, integrators and developers during the model and assemble phases of the SOA lifecycle[2] to locate services and policies to use or reuse, and to evaluate the impact of changes to service configurations. The primary source of information about service assets for these users would be Rational Asset Manager software, but they would follow Rational Asset Manager references to WebSphere Service Registry and Repository software-managed content for more detailed information about services and related run-time-relevant information about those services. WebSphere Service Registry and Repository software is used by deployment professionals and administrators in the deploy phase of the SOA lifecycle, and exploited by the SOA foundation run times for dynamic selection of service endpoints and configuration of the SOA environment. It's also used in the manage phase of the lifecycle to support policy enforcement required by service-level agreements (SLAs), and to present a more comprehensive view of the managed service environment.

Rational Asset Manager overview

Rational Asset Manager software supports various communities, tasks and individual needs.

Communities and tasks

Several categories of users for Rational Asset Manager software include:

- *Business and technical management—team leads, project managers.*
- *Administrators—integration administrators, community space administrators.*
- *Practitioners—analysts, architects, developers, testers.*

Highlights

Service metadata is leveraged by many kinds of users to support development-time asset management scenarios.

For each of these users, the Rational Asset Manager solution provides support for asset management scenarios, as introduced in figure 3. These scenarios may be conducted from Web clients as well as Eclipse clients. The five blue-highlighted scenarios are briefly covered in this paper.

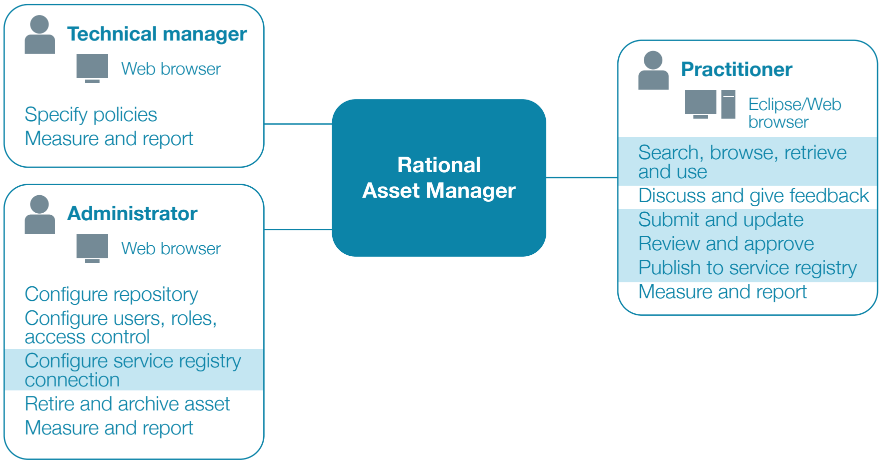

Figure 3. Leveraging Rational Asset Manager, several kinds of users work with services.

Organizing communities to address individual needs

The Rational Asset Manager solution helps create and support multiple communities with unique members, access controls and policies.

Content

Rational Asset Manager software manages metadata on assets to support asset management scenarios. The metadata includes obvious items such as the asset name, description, version and state. Other metadata includes the artifact name, description, version and reference (or location). To support these scenarios, Rational Asset Manager software uses the Reusable Asset Specification (RAS) structure from the Object Management Group (OMG) as the core structure for asset metadata. Additional metadata is required for integrating with tools and other repositories and registries such as WebSphere Service Registry and Repository software.

Highlights

Using RAS to represent service assets your organization can configure service asset types to align with your development processes.

RAS provides a structure for unique asset identification as well as metadata extension. This combination provides the basis for managing references between Rational Asset Manager and WebSphere Service Registry and Repository software. Using RAS in the Rational Asset Manager solution, your enterprise can configure multiple asset types to meet your particular needs. Examples of asset types include applications, components, services and Word templates.

A *service* can be defined as a single asset type containing all necessary artifacts, or it can be decomposed into many asset types such as service interface, service implementation and service test. When defining asset types, you need to do so from the perspective of someone who will be using the asset; that is, you need to define artifacts and relationships. Assets are created for a specific type, and they possess the characteristics defined by that type.

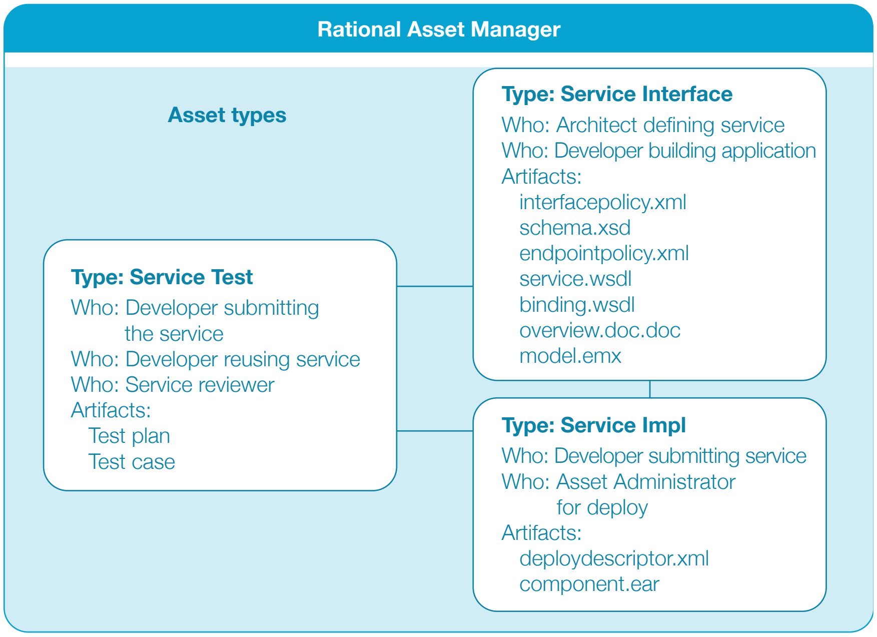

Figure 4. Configurable asset types in Rational Asset Manager are used to represent services.

Highlights

Regardless of the service asset type configuration in Rational Asset Manager, you can synchronize with WebSphere Service Registry and Repository.

The key point here is that, regardless of the service asset type configuration in Rational Asset Manager software, the synchronization can still work with WebSphere Service Registry and Repository software. A service asset that has the WSDL, XML Schema Definition (XSD) or XML configuration is the asset wherein the ID will be used for maintaining references between repositories.

In addition to the service asset metadata such as the ID that is used to create a reference between Rational Asset Manager and WebSphere Service Registry and Repository software, other metadata elements and classification are synchronized between Rational Asset Manager software and WebSphere Service Registry and Repository software. A sample of the service asset metadata is illustrated in figure 5 in the RAS box.

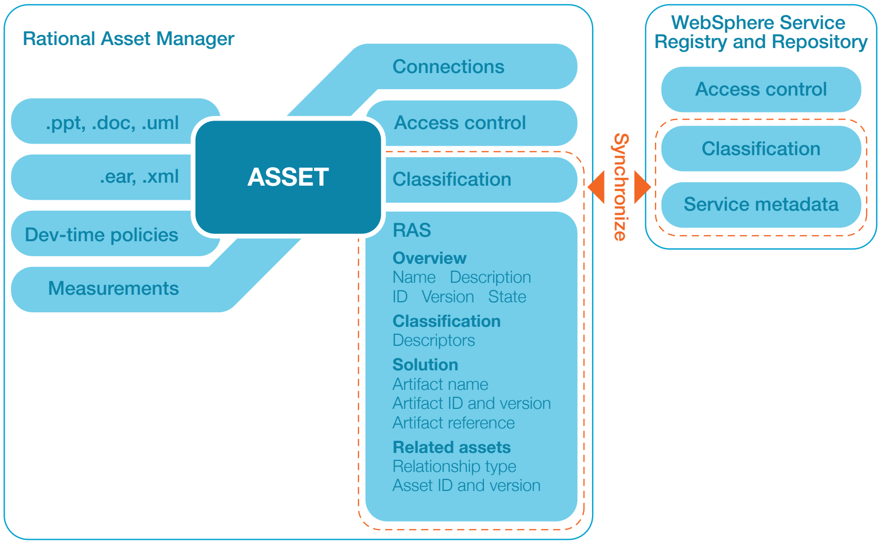

Figure 5. RAS service metadata in Rational Asset Manager is federated with WebSphere Service Registry and Repository service documents.

Synchronizing service classification and metadata reduces administrative overhead and simplifies developer use.

Highlights

WebSphere Service Registry and Repository overview

The WebSphere Service Registry and Repository solution offers the service metadata management services required to implement flexible SOA composite applications.

Community and tasks

WebSphere Service Registry and Repository provides detailed descriptions of services and composite applications.

Here the user community represents parties involved in assembling, configuring and managing SOA composite applications. The main interest of these users is in managing detailed descriptions of services and composite applications, and in governing the lifecycle of those SOA applications.

The users need a central place to advertise and find information about any service deployed in or used by (in case of an external service) an enterprise. They need to understand the capabilities and requirements of those service endpoints, and they want to enforce policies regarding who can provide, configure or use a specific service.

This community needs rich query capabilities that allow users to navigate the potentially large set of services to exploit the potential SOA offers in terms of rapidly assembling, configuring and, more importantly, reassembling and reconfiguring composite applications in reaction to changing business requirements.

The users also need a governance model that allows them to enforce governance policies for services. They use the service metadata to effect governance of the underlying services; governance policies are represented as part of the service metadata and can be enforced by SOA run times such as the enterprise service bus (ESB), which enforces governance policies controlling interactions

Highlights

of services that are based on information managed in the WebSphere Service Registry and Repository system. Controlling who can create and delete content in WebSphere Service Registry and Repository software means controlling who is allowed to make services visible and invisible for others to use. Controlling updates to the lifecycle state model of WebSphere Service Registry and Repository content drives usage of services by SOA run times (e.g., only services in state “production” can be targets of ESB-managed interactions).

Content

WebSphere Service Registry and Repository software is the definitive repository for all the deployed applications you wish to present as services. It stores information about service endpoints as well as information about composite applications using those endpoints. WebSphere Service Registry and Repository content falls into three categories:

WebSphere Service Registry and Repository content is in three categories: service metadata, semantic annotations and governance-related information.

- *Documents containing metadata about services that are created outside the scope of the WebSphere Service Registry and Repository system and imported into it for management. WebSphere Service Registry and Repository software not only stores the document, but also establishes a fine-grained model for the content of such a document, enabling users to navigate and query into the coarse-grained metadata documents.*
- *Semantic annotations of those documents or elements of their content model describing relationships, properties or classification of the artifacts conveying semantics not obvious from the document content. These semantics allow WebSphere Service Registry and Repository users to advertise services when they publish them in ways that make it easy for service metadata consumers to find “interesting” information.*
- *Governance-related information. This information could be a lifecycle state model for governed WebSphere Service Registry and Repository content, access constraints on it, validation policies to be applied, etc.*

Highlights

Rational Asset Manager and WebSphere Service Registry and Repository integration scenarios

Enabling the enterprise to manage and govern the service across multiple communities and repositories is a key motivator to federating metadata between Rational Asset Manager and WebSphere Service Registry and Repository.

Principles of federation

Rational Asset Manager and WebSphere Service Registry and Repository serve different communities with a shared focus on services.

As explained earlier, the user communities and content managed by Rational Asset Manager software and the WebSphere Service Registry and Repository system are rather different. Many assets have nothing to do with deployed services, and not every deployed service automatically is an asset an enterprise might want to reuse. However, there are a number of artifacts that can be of interest to both user communities: basically any document (e.g., WSDL, XSD, WS-Policy) stored in WebSphere Service Registry and Repository software could be of interest as part of a reusable asset; and WebSphere Service Registry and Repository users might be interested in understanding design specifications, implementation or documentation about a specific service.

Service metadata is federated to provide community-relevant information.

We apply the general principles of our metadata management strategy here to deal with the situation—federating information to provide community-relevant view and context. In some cases this is achieved by physically replicating information, while in other cases we provide links that users can select to navigate from one repository to another. For any artifact that is replicated between the repositories, we establish which repository owns the master copy and which holds a cached copy of the original for the purpose of providing a context-specific view on it.

Different communities require different governance and authorization models. Often in development context a rather coarse-grained workgroup model is used to facilitate collaboration where any member of a workgroup can perform any operation on an artifact once in the scope of the workgroup. On the other hand, governance requirements for deployed services

Highlights

often imply a much more fine-grained and restrictive model for specifying who can perform which tasks on specific artifacts. Rather than trying to unify these authorization models, we introduce in each community bridging roles that are authorized to make content from one environment available in another environment.

As part of federating metadata between Rational Asset Manager software and WebSphere Service Registry and Repository software, there is federation of governance. Governance is the establishment of:

- *Chains of responsibility to empower people.*
- *Measurement to gauge effectiveness.*
- *Policies to guide the organization to meet its goals.*
- *Control mechanisms to ensure compliance.*
- *Communication to keep all required parties informed.*

Governance needs are driven by the communities that each repository serves.

The governance needs are driven by the communities that each repository serves. The governance model for service assets in the Rational Asset Manager solution is focused on asset submission, review and approval, procurement of the assets and their retirement, as well as measurement and reporting. The governance model for services in the WebSphere Service Registry and Repository system is focused on managing the lifecycle of a service and metadata artifacts related to the service, potentially from the point where it is deployed into a test environment to approval and provisioning of its deployment into production and onto retirement of the service. These two governance models operate in a related, yet independent, fashion with each other. An operational service endpoint with matching lifecycle governance in WebSphere Service Registry and Repository software can be

Highlights

Federating service metadata requires repository authentication and connection information to be configured.

selected to become part of an asset and go through an asset approval process in Rational Asset Manager software. Or a service declaration that is part of an asset that was approved in Rational Asset Manager software can be published into WebSphere Service Registry and Repository software, reflecting the service's transition from development into deployment and production.

Rational Asset Manager software to WebSphere Service Registry and Repository software

There are several scenarios for navigating the metadata relationships from Rational Asset Manager to WebSphere Service Registry and Repository software. Each integration scenario is introduced with a summary paragraph and accompanying text and images.

Configure service registry connection

Summary: An administrator configures the connection from a Rational Asset Manager community to one or more WebSphere Service Registry and Repository service registries.

The Rational Asset Manager software synchronizes with WebSphere Service Registry and Repository software to provide coordination of asset and service metadata. To enable this synchronization, the administrator describes one or more WebSphere Service Registry and Repository connections for a Rational Asset Manager community.

For example, the administrator may configure one connection to a WebSphere Service Registry and Repository test registry, and another connection to a WebSphere Service Registry and Repository production registry, as shown in figure 6. Each connection needs information such as the address to the host,

Highlights

which port to use, the WebSphere Service Registry and Repository login credentials, as well as the name of the owner of the assets. Each asset in Rational Asset Manager software has an owner, so when WebSphere Service Registry and Repository shadow assets are created, the proper asset owner will be set.

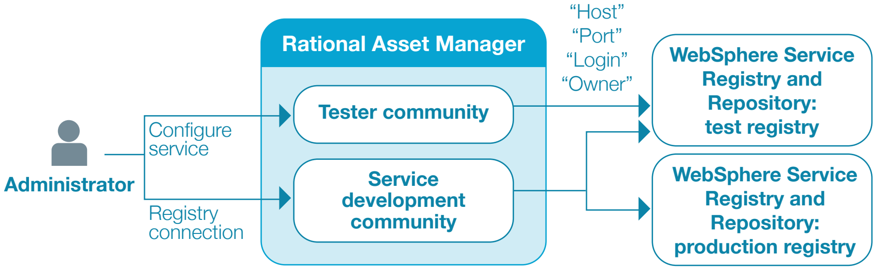
Administrator
Configure service
Registry connection
Rational Asset Manager
Tester community
Service development community
"Host"
"Port"
"Login"
"Owner"
WebSphere Service Registry and Repository: test registry
WebSphere Service Registry and Repository: production registry

Figure 6. The Rational Asset Manager administrator configures a WebSphere Service Registry and Repository connection for Rational Asset Manager communities.

Rational Asset Manager software will allow service asset artifacts to be published to the WebSphere Service Registry and Repository system when the following conditions are met:

Service assets can be published to WebSphere Service Registry and Repository when certain conditions exist in Rational Asset Manager.

- *The community is configured with one or more WebSphere Service Registry and Repository connections.*
- *There is at least one WSDL file, XSD file, XML file, SCA module or WS-Policy file in a service asset.*
- *The service asset needs to be in the approved state.*
- *The person publishing the service documents needs to have publish access/ entitlement on the asset.*

Highlights

When these conditions are met, the *Publish to service registry* option becomes visible in the service asset details. If more than one WebSphere Service Registry and Repository connection is available, the person publishing selects the appropriate one. Multiple WebSphere Service Registry and Repository registries are not required.

Publish service asset to the WebSphere Service Registry and Repository system

Summary: A developer or administrator publishes the WSDL, XSD and XML from a Rational Asset Manager asset to the WebSphere Service Registry and Repository system.

In this scenario, the developer submits to the Rational Asset Manager repository the assets that the service comprises. The service assets are updated in the repository with relationships, and then are submitted to the review process. In this case three kinds of assets represent the service, including Service Interface, Service Test and Service Impl.

The developer submits service assets to Rational Asset Manager, using the preconfigured asset types.

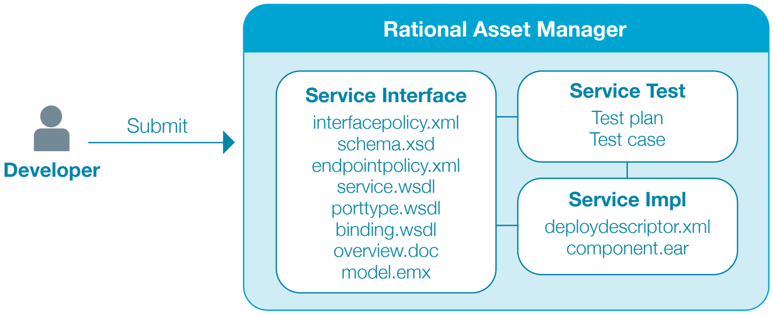

Figure 7. In this example, the developer submits a Service Interface asset with bidirectional relationships to the Service Impl and Service Test assets.

Highlights

The tester is notified as one of the reviewers who then publishes the service asset artifacts to the WebSphere Service Registry and Repository test registry and then reuses the service assets to test the service.

The service is validated and approved in the development-time repository.

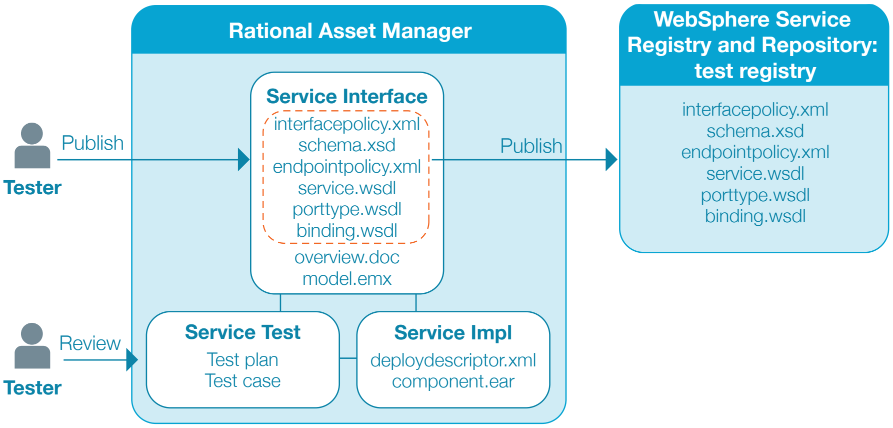

Figure 8. The tester publishes the service asset to a WebSphere Service Registry and Repository test registry and validates the service.

The review process is completed, and for our purposes here, we will say that the service assets are approved. The appropriate stakeholders are notified of the results of the review.

The service is published to WebSphere Service Registry and Repository.

The administrator now publishes the service asset artifacts to the WebSphere Service Registry and Repository production registry for further lifecycle activities and for ultimate reuse in the development of enterprise solutions. After the service is published, a synchronization mechanism creates WebSphere Service Registry and Repository shadow service assets in the Rational Asset Manager software. This is described below.

Search and retrieve service

Summary: A developer searches for a service and its supporting development assets. A developer retrieves the service and related assets into the development workspace to build an application.

Highlights

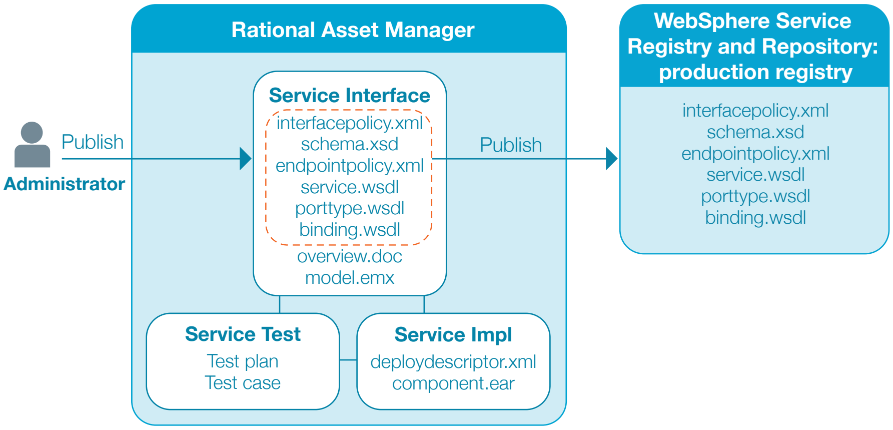

Figure 9: The administrator publishes the service asset to a WebSphere Service Registry and Repository production registry for the service to be reused.

After the service asset artifacts (WSDL, XSD, XML) are published to WebSphere Service Registry and Repository software and the repositories are synchronized, a shadow service asset is available in Rational Asset Manager software to support search and retrieval scenarios. In the Rational Asset Manager application the original service asset and the shadow service asset point to each other through an asset relationship.

The developer searches for the service in Rational Asset Manager, and navigates to WebSphere Service Registry and Repository for additional service information.

The developer can search the service asset and the service shadow asset metadata and content. A link is also stored in the original service asset (in this case the Service Interface asset) and the service shadow assets pointing to the service documents in WebSphere Service Registry and Repository software. This link is navigable, allowing the developer to go from Rational Asset Manager software to WebSphere Service Registry and Repository software for further browsing, such as information that ITCAM publishes into the WebSphere Service Registry and Repository system. There is one shadow service asset created for each of the documents published to WebSphere Service Registry and Repository software, as shown in figure 10.

Highlights

Service searching and retrieval scenarios are enabled through shadow service assets in Rational Asset Manager.

Providing searching and indexing at the service asset level and the service document level provides the developer with the granularity to work with a specific service document, as well as the abstraction to understand related assets such as service tests, and business processes.

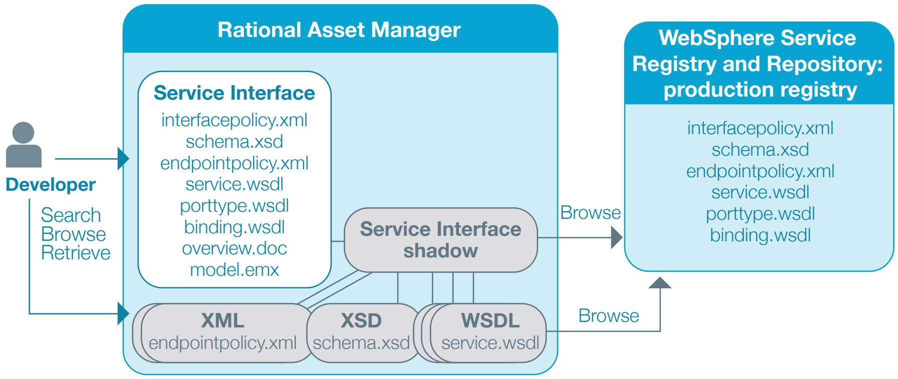

Figure 10. A shadow service asset with fine-grained service document assets are synchronized with Rational Asset Manager.

The service assets, shadow assets and other related assets are retrieved into the developer's Eclipse workspace for application development.

WebSphere Service Registry and Repository software to Rational Asset Manager software

There are several scenarios for navigating the metadata relationships from WebSphere Service Registry and Repository software to Rational Asset Manager software.

Discover development service assets for a service problem

In this scenario, an information technology (IT) administrator is notified of a problem with a service in production. The IT administrator finds the service document in the WebSphere Service Registry and Repository system (step 1 in figure 11) and navigates to the service asset in Rational Asset Manager software (step 2 in figure 11).

Highlights

On the service asset in Rational Asset Manager software, the IT administrator submits a defect and subscribes to the service asset (step 3 in figure 11) to be notified as the next version is modified and prepared to be published. The submitted defect is transferred to IBM Rational ClearQuest® software, and the Rational Asset Manager service asset notifications are sent either through Really Simple Syndication (RSS) feeds or through e-mail, as shown on the *Notify* line in figure 11.

Federated service metadata permits the enterprise to understand development-time service assets to solve service problems.

If a developer or other stakeholder is notified of the defect through Rational ClearQuest software, a link is stored in the defect record, allowing the developer to navigate from Rational ClearQuest software, to the service asset in Rational Asset Manager software, and on to the WebSphere Service Registry and Repository system if needed.

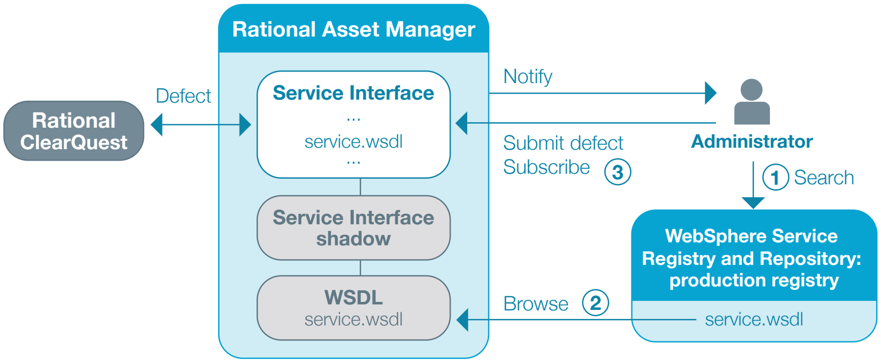

Figure 11. The WebSphere Service Registry and Repository service documents link to Rational Asset Manager service assets to support problem determination scenarios.

The IT administrator can navigate to other service assets, such as service tests, and business processes through the asset relationships.

Highlights

Development-time service assets can be created for newly added trading partner services.

Discover development service assets for a service document

In this scenario a trading partner provides a WSDL for its service. This scenario implies there are initially no service assets in the Rational Asset Manager software. The IT administrator enters the WSDL into the WebSphere Service Registry and Repository system (step 1 in figure 12). The synchronize mechanism creates a shadow service asset in Rational Asset Manager software (step 2 in figure 12). The developer creates and submits development service assets and their relationships to support the service (step 3 in figure 12). In step 3 the developer creates the relationship from the Service Interface asset to the WSDL asset.

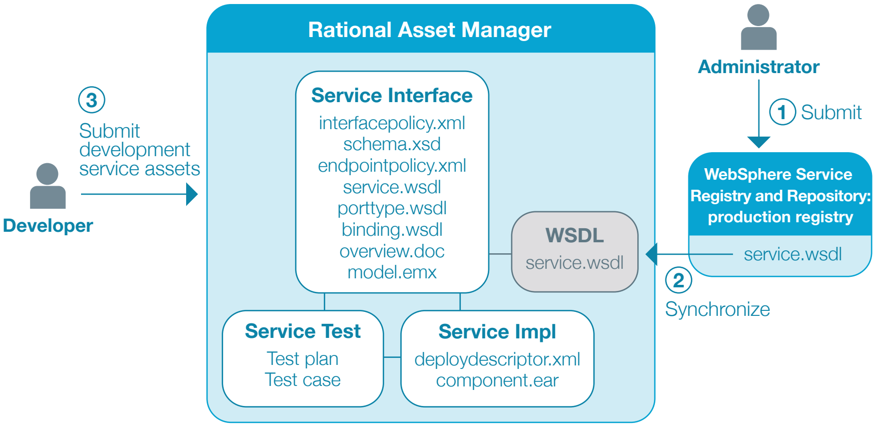

Figure 12. The WebSphere Service Registry and Repository service documents can link to new Rational Asset Manager service assets to support new service documents provided by trading partners.

Highlights

Publishing service assets creates service documents and metadata in WebSphere Service Registry and Repository.

Architectural overview

The major components providing federated metadata between Rational Asset Manager and WebSphere Service Registry and Repository software are introduced in this section.

Publishing Rational Asset Manager assets to WebSphere Service Registry and Repository software

Assets in Rational Asset Manager software can contain many artifacts. These artifacts may be WSDL or XSD files as well as other items. When a service asset is published from the Rational Asset Manager system, the developer selects the WSDL, XSD and other relevant files from the asset content.

Upon invoking the *Publish* function, a WebSphere Service Registry and Repository GenericObject is created for the service asset. This GenericObject contains service asset metadata that points back to the originating Rational Asset Manager service asset. In figure 13, only two of the service asset files were published, but certainly others can be as well.

When the service asset is published, WebSphere Service Registry and Repository software creates relationships between the GenericObject and the service documents, as well as relationships among the service documents.

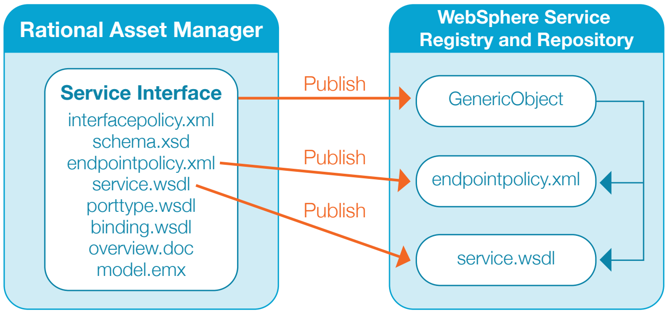

Figure 13. Publishing a service asset to WebSphere Service Registry and Repository creates a GenericObject for the service asset and its constituent service documents and relationships.

Highlights

Federated service metadata is synchronized using a background process in Rational Asset Manager.

Synchronizing WebSphere Service Registry and Repository documents to Rational Asset Manager software

When the WebSphere Service Registry and Responsibility service documents are brought into Rational Asset Manager software, the service document owner is preserved and stored in the Rational Asset Manager asset, unless the user authentication systems are different and the owner cannot be resolved. In Rational Asset Manager software, for each WebSphere Service Registry and Repository connection a default asset owner is specified and will be used if the WebSphere Service Registry and Repository service document owner cannot be resolved.

While publication of service assets from Rational Asset Manager software into WebSphere Service Registry and Repository software is an explicit, user-driven process, synchronizing service interface metadata from WebSphere Service Registry and Repository software into Rational Asset Manager is performed in a more automated fashion. Rational Asset Manager software and WebSphere Service Registry and Repository software are kept in synchronization with each other through a background process. This process is initiated by an indexing interval that is set per a WebSphere Service Registry and Repository connection in a Rational Asset Manager community, as shown in figure 14.

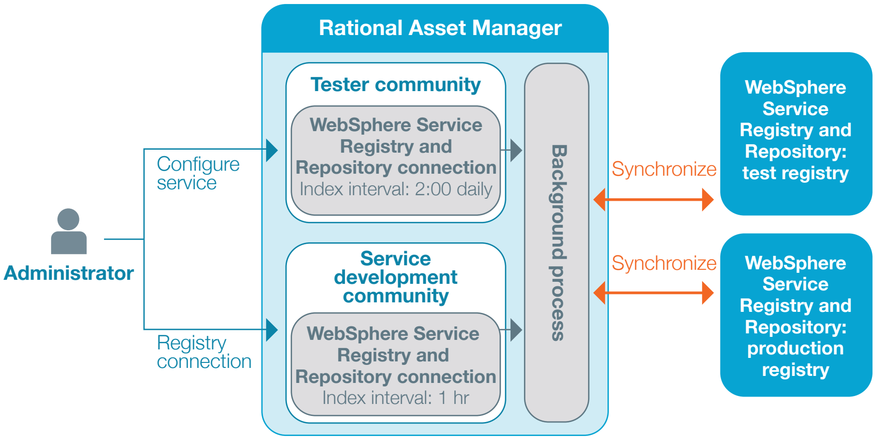
Rational Asset Manager
Tester community
WebSphere Service Registry and Repository connection
Index interval: 2:00 daily
Service development community
WebSphere Service Registry and Repository connection
Index interval: 1 hr
Background process
Configure service
Registry connection
Administrator
Synchronize
Synchronize
WebSphere Service Registry and Repository: test registry
WebSphere Service Registry and Repository: production registry

Figure 14. Synchronization of Rational Asset Manager and WebSphere Service Registry and Repository is managed by a background process.

Highlights

Shadow service assets are added to Rational Asset Manager, providing navigation of the service assets across the repositories.

The indexing interval can be set on several increments, such as by minutes, hours, days and weeks. When the background process runs, it looks for new WebSphere Service Registry and Repository documents and creates shadow assets with the WebSphere Service Registry and Repository types and WebSphere Service Registry and Repository classifications in Rational Asset Manager software. Creating shadow assets is necessary to reflect the WebSphere Service Registry and Repository type and classification information in Rational Asset Manager software. A custom attribute called "bsrURI" is added to the shadow assets. This attribute is a navigable link taking the user from the Rational Asset Manager shadow asset to the service interface documents in WebSphere Service Registry and Repository software.

When the Rational Asset Manager to WebSphere Service Registry and Repository *Publish* function occurs, an ID property value is placed on the WebSphere Service Registry and Repository GenericObject. This value is used to find the Rational Asset Manager asset and create a "deployed" relationship from the shadowed GenericObject asset to the originating service asset, as shown in figure 15.

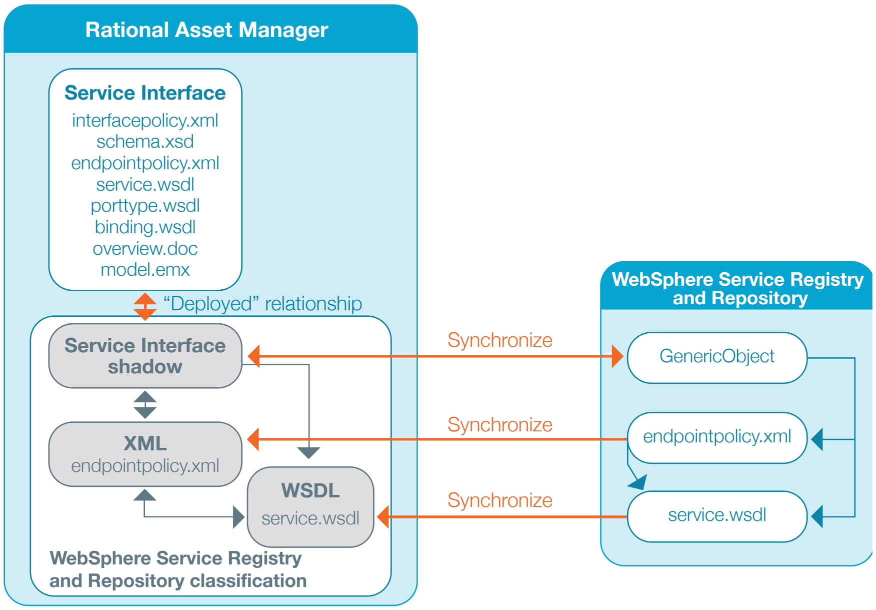

Figure 15. The WebSphere Service Registry and Repository service documents are synchronized with Rational Asset Manager service assets, creating a "deployed" relationship to the originating service asset.

Summary

IBM offers a number of metadata repositories, each targeted toward a specific user community, supporting the tasks and processes important to that community and offering the content relevant to those tasks and community.

This paper illustrates the principles of our metadata management strategy and how they are applied in the Rational Asset Manager development repository and the WebSphere Service Registry and Repository run-time service metadata registry and repository as examples.

The Rational Asset Manager to WebSphere Service Registry and Repository scenarios were introduced, including:

- *Configure service registry connection.*
- *Publish service asset to WebSphere Service Registry and Repository software.*
- *Search and retrieve service.*

The WebSphere Service Registry and Repository software to Rational Asset Manager scenarios were introduced, including:

- *Discover development service assets for a service problem.*
- *Discover development service assets for a service document.*

Finally the paper concluded with an architectural overview of the synchronization mechanisms that provide federated metadata for the Rational Asset Manager and WebSphere Service Registry and Repository solution.

For more information

To learn more about IBM Rational Application Manager and IBM WebSphere Registry and Repository software, visit:

ibm.com/software/rational

IBM Corporation
Software Group
Route 100
Somers, NY 10589
U.S.A.

Produced in the United States of America
06-07
All Rights Reserved

The information contained in this documentation is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this documentation, it is provided "as is" without warranty of any kind, express or implied. In addition, this information is based on IBM's current product plans and strategy, which are subject to change by IBM without notice. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this documentation or any other documentation. Nothing contained in this documentation is intended to, nor shall have the effect of, creating any warranties or representations from IBM (or its suppliers or licensors), or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

[1] Here, the word "service" represents a service endpoint that is operational in a test or production environment, or information such as schemas that may be used by one or more services.

[2] For a detailed description of the IBM SOA foundation, the SOA lifecycle and the reference architecture, visit: download.boulder.ibm.com/ibmdl/pub/software/dw/webservices/ws-soa-whitepaper.pdf