

# Web Service Standards for Service Registry and Repository

October 2006

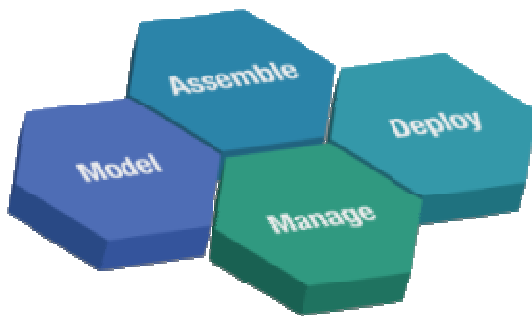
**IBM**

1	Introduction.....	2
2	The role of service registry and repository in Service Oriented Architecture .....	4
3	The relevance of standards for service registry and repository .....	5
4	The use of a service registry and repository in service definition, selection and reuse	5
4.1	Standards relevant to service definition, selection and reuse .....	6
4.1.1	Service Registry and Repository alignment with Web services .....	6
4.1.2	Service registry and repository Data Model .....	9
4.1.3	Service registry and repository Query Language.....	10
5	The use of a service registry and repository in service runtimes .....	11
5.1	Service registry and repository standards relevant to use by service runtimes.	12
6	The role of a service registry and repository in service management.....	13
7	Federation of service registries and repositories.....	13
7.1	Relationship of service registry and repository and UDDI.....	14
7.1.1	Integration of the Service Registry Repository with UDDI.....	16
8	Summary .....	17
9	References.....	18

# 1 Introduction

This paper describes the main concepts and capabilities of a service registry and repository and the standards that deliver the value of a service registry and repository in a heterogeneous environment.

A service registry and repository handles the management of service descriptions and serves as the system of record for this information throughout the complete lifecycle of a service. This paper elaborates on the standards that enhance the value of a service registry and repository throughout the four main stages of a service lifecycle: modeling, assembly, deployment and management (see Figure 1).



**Figure 1. The service lifecycle as defined in IBM's SOA Foundation (see [\[SOA Foundation\]](#))**

In the model and assemble phases, the service registry and repository serves two main functions. First the service registry and repository can store or capture the interfaces and messages used by a new service when it is modeled or a composite service or process when it is assembled. Second, the service registry and repository makes the service information searchable to encourage reuse of common services. A common example of service information gathered or searched in these phases would be documents conforming to the Web Service Description Language (WSDL) or XML Schema (XSD) standards (see [\[WSDL 1.1\]](#) and [\[XML Schema\]](#)). Architects and developers can also annotate this technical information in the registry and repository with extra information using formal classification techniques, such as using the Web Ontology Language (OWL) standard, or by adding other content expressed in XML (see [\[XML\]](#)). Such classification of services allows for meaningful groupings of the services based on business objectives.

The service deployment phase adds additional information to the service registry and repository including the connectivity, security and reliability characteristics of the deployed service. This extra information builds on the information from the modeling and assembly phases and differentiates each deployed service instance in the Service Oriented Architecture (SOA). The deployment information may be expressed in terms of new WSDL content or through the use of policy statements defined using the WS-Policy specifications (see [\[WS-Policy\]](#)).

In deploy and manage phases of the lifecycle, the service registry and repository serves as a point of integration for the SOA environment by reflecting a system of record for each deployed service. The relationships, dependencies and interaction of the services can be documented and exploited in these phases.

It is important to note that the service lifecycle is not purely linear, in that a deployed service typically evolves and changes over time. As the service is versioned, or changes to the policy under which it is deployed are effected, it is important for all stakeholders in the service lifecycle to be aware of such changes. The service registry and repository contains this service information that allows for analysis of services usage and impact.

Since the service registry and repository plays a central role throughout the SOA lifecycle, it is also important to note that it will be integrated with applications from the service development lifecycle, change and release management, service runtimes, service management (operational efficiency and resilience) and other service registries and repositories. These integration points are shown in figure 2.

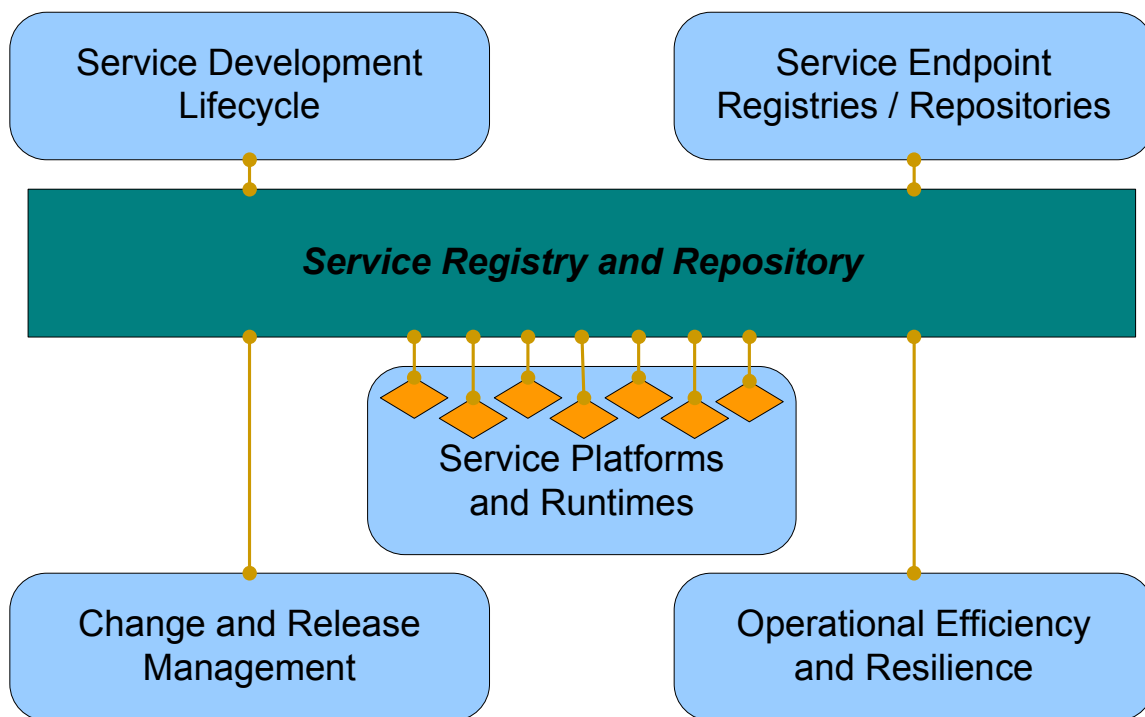


Figure 2. Service registry and repository interactions in a Service Oriented Architecture (SOA)

The interactions shown in Figure 2 also highlight the notion that the complete definition of a service results from the interaction of multiple applications and may also be distributed in multiple registries and repositories. It is necessary then, to consider federation of heterogeneous registries and repositories as a key part of a standards roadmap for registry and repository function. One registry standard that needs to be

considered is defined by the Universal Description, Discovery and Integration (UDDI) specifications.

## **2 The role of service registry and repository in Service Oriented Architecture**

Service Oriented Architecture (SOA) offers the promise of business agility and resilience through reuse, loose coupling, flexibility, interoperability, integration and governance. These are realized by separating service descriptions from their implementations, and using this descriptive information across the service life-cycle. Standards-based service information artifacts capture the technical details of what a service can do, how it can be invoked or what it expects other services to do. Information, annotations and classifications from the service providers 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.

A service registry and repository handles the management of service descriptions and serves as the system of record for this information. A populated and maintained service registry and repository is the place in an organization that catalogs all of the services deployed or used by an organization.

Another function of a service registry and repository is to support the governance of the service lifecycle including promotion of services through phases of that life-cycle, controlled access to service information, and impact analysis and socialization of changes to services with interested parties.

A service registry and repository increases the value proposition of a Service Oriented Architecture by providing a place to organize, manage, find and govern the service descriptions. In heterogeneous deployment environments that are typical in SOA, the service registry and repository needs to provide a standard, interoperable means for access, query and manipulation of the service descriptions. There are several existing standards and specifications that address particular interactions with service descriptions. This paper will describe where these existing standards or specifications can be used to compose the complete functional solution for a service registry and repository. Where the existing standards or specifications do not exist or are inadequate to enable the full set of capabilities for managing the service lifecycle, this paper discusses additional work that should be done as part of a standards effort. One key test for the standards relevant to the service registry and repository component in SOA is that they align with the Web service specifications that can be composed together into a proven interoperable solution.

### **3 The relevance of standards for service registry and repository**

The driving force behind a standards based service registry and repository is that it is a critical integration point in SOA. This means that a service registry and repository will interact with the applications involved in the development lifecycle, change and release management, runtimes and platforms, management components and other service registries and repositories. It is also important to note that Web services built on open standards play a key role in many SOA deployments and a service registry and repository needs to complement the standards used by Web services themselves.

Given the number of interactions with different applications and different users, it is critical for service registries and repositories to implement a well defined, interoperable security model. This security model must address user identification, authentication, and authorization, confidentiality of the messaging and the integrity and privacy of the contents of a service registry and repository. The standards and specifications needed to establish the security model and achieve interoperability are described in the “Security in a Web Services World: A Proposed Architecture and Roadmap” ( see [WS-SecurityRoadmap]). The service registry and repository specifications must complement and remain compatible with the standards and specifications defined in the Web services security roadmap.

Three specific areas where standards yield a benefit for service registry and repository are discussed in three sections below: the use of a service registry and repository in service definition, selection and reuse; in service runtimes and in service management systems. Each section will discuss the relevant standards and specifications that exist today, beyond those mentioned from the security roadmap, as well as the areas where additional specification and standards work is needed.

### **4 The use of a service registry and repository in service definition, selection and reuse**

It is typical for a service runtime to have an ecosystem of tools that complement the service information formats supported natively in the runtime. These tools are used to model, build, assemble and deploy services and can benefit from a service registry and repository that indicates what services are actually running and available for use in an organization.

Web service information such as WSDL files and XSD files are typically first class artifacts that are supported in most service-oriented application development and assembly tooling. Since many of these tools already provide a familiar user experience to deal with these Web service self description files, the mechanism to query the content of those files should complement this existing investment in editors and viewers for WSDL

and XSD files. Web service developers and application assemblers have become familiar with and comfortable with some of the concepts in WSDL and XSD, as have the providers of their tooling. It is therefore advantageous to provide a means of expressing questions or queries about the WSDL and XSD files in a service registry and repository without introducing a new format or requiring a mapping by either the end user or the developers of those tools.

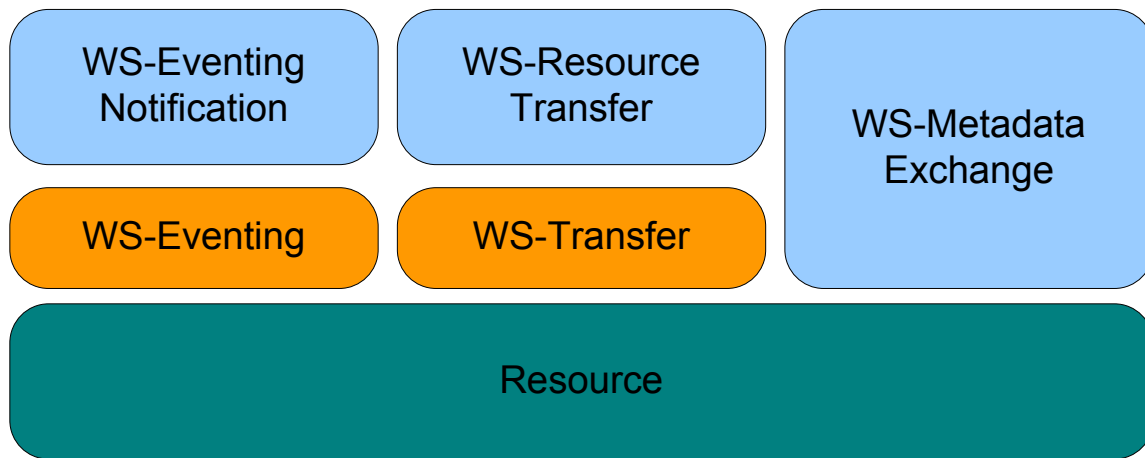
Once a critical volume of service descriptions exist in a service registry and repository, it is an important resource for developers and application assemblers to understand what services exist. Service developers are often constrained by an existing data type or information model that must be supported by the service. While searching for a suitable service, a developer may need to inspect the service based on the typing of the message as indicated in the XSD. As the system of record, this use requires that a service registry and repository have the knowledge to combine all of the service description documents, such as message formats, interfaces and policies, for each registered service.

## ***4.1 Standards relevant to service definition, selection and reuse***

One key aspect of service definition in a service information repository is providing an interoperable means of, adding and updating the content using the tooling from multiple vendors. Similarly, service selection in the model and assemble phases may require tooling from multiple vendors to be used and thus an interoperable means of querying and selecting service definitions is needed.

### **4.1.1 Service Registry and Repository alignment with Web services**

The widespread adoption of Web services has led to a common development pattern where WSDL and XSD files are shared as Web resources accessible over HTTP. This access pattern for service descriptions treats service descriptions as Web resources that can be accessed independent of the service binary, executable code, or source code. The Web service metadata and resource specifications take this access pattern a step further and provide a means of accessing service descriptions over any transport capable of sending and receiving SOAP (see [[SOAP](#)]) messages and identifying the resources using WS-Addressing (see [[WS-Addressing](#)]). Treatment of the service description as a Web service resource means that the service description is part of the state of the service, also referred to as the service metadata resource, which can be accessed, manipulated and subscribed to using the specifications shown in Figure 3.



**Figure 3. View of the existing specification for manipulating, querying and subscribing to Web service resources.**

As mentioned earlier, the service registry and repository standards should build on existing Web service specifications where possible. For retrieval, creation, update and deletion of service description resources, the WS-Resource Transfer (WS-RT) specification provides the consistent framework for interacting with resources (see [[WS-ResourceTransfer](#)]).

In order for developers, architects and integrators to be aware of the lifecycle and changes to the service, it is also important for a service registry and repository to implement an interoperable means to subscribe to a resource representing service information and monitor it's progress through it's lifecycle. The forthcoming WS-EventingNotification (WS-EvN) specification provides this framework and will be another specification tracked by a service registry and repository standards effort (see [[WS-ManagementRoadmap](#)]).

WS-MetadataExchange is a complementary specification for aggregating all of the details and properties of a specific resource, also referred to as the metadata, into a single message (see [[WS-MetadataExchange](#)]). WS-MetadataExchange defines a single

operation, `getMetadata`, which allows for a single point of service information integration for a specific service. Similarly a service registry and repository is a single point of service information aggregation and integration for multiple services. Service registry and repository standards will directly complement WS-MetadataExchange so that the complete definition of a service can be retrieved at the service endpoint itself, or a direct equivalent can be retrieved from a service registry and repository instance. As WS-MetadataExchange is more widely deployed, a service registry and repository instance should establish a formal relationship between the WS-MetadataExchange provider for the metadata resources of a given service and the entries stored in a service registry and repository for that service. If a service does not have a WS-MetadataExchange provider, the capabilities of a service registry and repository can be used to provide the service descriptions in the format needed by WS-MetadataExchange..

Manipulating the service definition can be accomplished using the mechanisms defined in WS-RT. WS-RT defines body elements for Create, Get, and Put that support creating, retrieving, and updating partial elements of a resource. Working with partial elements of a given piece of service information allows for improved performance when a user or application needs a small fragment of the resource(s) representing the complete service description.

WS-RT defines support for XPath and an extension mechanism to add new dialects for querying the resources. The use of XPath and queries is described in greater detail in a subsequent section of this paper.

A service registry and repository will directly support WS-RT so that the act of manipulating service information is nearly identical to what is done when using WS-RT to accomplish the same task at the service endpoint. This approach is consistent with the means of accessing service descriptions discovered at service endpoints using WS-MetadataExchange.

A key goal of service registry and repository specifications will be to eliminate the differences between integration and interaction with the metadata provided at the service endpoint and interaction with the service information in a service registry and repository. In this regard a service registry and repository can act as a broker for an individual resource or an aggregator of service information from multiple sources (see Figure 4).

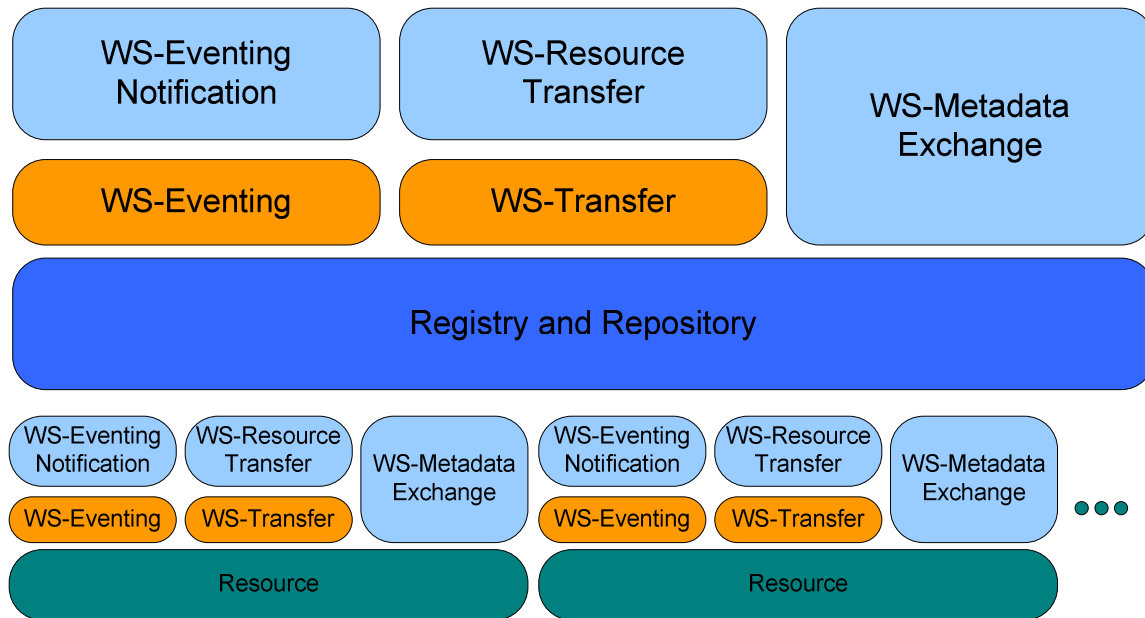


Figure 4. View of registry and repository as an aggregation of service description resources.

#### 4.1.2 Service registry and repository Data Model

One part of the interaction that is not defined by WS-RT is the data model that is manipulated using the methods defined in the specification. To facilitate interoperability between vendor implementations, it will be necessary to define a service registry and repository data model. This data model should consist of two main parts. The first part consists of the physical documents, or files, such as WSDL and XSD that constitute the syntactic definition of a service. The second part of the data model consists of three types of service semantics metadata types: Annotations, Relationships and Classifiers. The three types of service semantics metadata can apply to a whole document, a set of documents or just a fragment of a document. The service registry and repository provides a means of expressing these concepts in XML and associating these semantic constructs to the document, fragment or set of documents.

An annotation is a piece of XML that can be associated with an entry but normally would not be inside the original physical file. An example of an annotation one might associate with a WSDL document is an author as follows: `<author>Me</author>`. An example of an annotation associated with a fragment such as the `wsdl:service` element might be `<supportedBy>Not Me</supportedBy>`

Of particular importance to service definition and selection will be the user-defined category systems that are imported and shared through the use of documents encoded using the existing Web Ontology Language (OWL) standard. Any valid OWL document

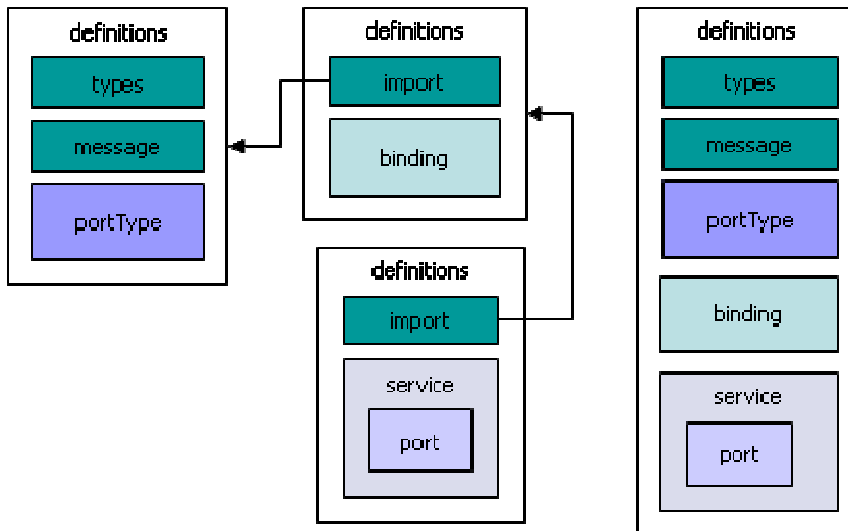
can be used as a classification system, and a service registry and repository should at least exploit the features that establish hierarchical classifications.

### 4.1.3 Service registry and repository Query Language

Beyond the support provided for accessing and manipulating service information described in WS-RT and WS-MetadataExchange, the service registry and repository queries select information that crosses the boundaries of multiple resources or documents which potentially represent multiple services. A second key difference is that when targeting a single known service, the query patterns are centered around producing subsets of the metadata, whereas in service registry and repository, the query patterns are centered around finding the service endpoint or a collection of service endpoints and the information that differentiates the services in the collection.

Since a service registry and repository deals with a number of metadata resources or a number of service information documents, the XPath 1 dialect in WS-RT does not provide the necessary document navigation and document reference resolving functions (see [[XPath1](#)]). It will be necessary for service registry and repository specifications to define a dialect based on the XPath 2 standard (see [[XPath2](#)]). Specific service registry and repository XPath functions should be provided to simplify the querying of standard service information documents, such as WSDL and XSD. Many of these functions could be written in XPath 2 directly, however, and the functions in the specification are intended to reduce the complexity of authoring the query and allow for implementations to optimize the execution of these functions.

As an example of the type of XPath 2 function that should be specified for the service registry and repository, consider the following diagram that illustrates two alternative means of modeling the same service in WSDL. On the left side, the WSDL definition is split across three files in a convention that is recommended to encourage reuse through separation of the interface, protocol binding and service endpoint. On the right side is a single WSDL definition file that could specify the exact same service.



**Figure 5.** The diagram represents two different approaches to package the service description for the same service in WSDL definitions documents. On the left, the definition is split between three documents and on the right the complete definition is in a single file.

In order to find the service based on the port type name, the XPath selection is different based on the design choice to separate the definition into one file or two.

By providing functions that encapsulate the navigation to the binding and portType elements with the document loading required by the import statement, the service registry and repository will allow an expression to be written consistently for both of the cases above:

## 5 The use of a service registry and repository in service runtimes

At the core of a service registry and repository are the artifacts that describe the operational services in the organization. In order to encourage providers of service runtimes to integrate their service description artifacts into the service registry and repository, the content model of the service registry and repository must directly complement the artifacts that are available from the service runtimes.

One key challenge for IT operations and SOA governance councils who have deployed registries or repositories is the lack of information propagated from the service runtime. In many service registries today, the changes made to service information in the deployment phase, such as policy configuration or service endpoint details, is not captured in the service description due to lack of integration with service runtimes and the deployment tooling for those runtimes. For a Web service, it is typical for the service runtime to at least be able to export WSDL and XSD files. The service registry and repository should accept that critical Web service information as exported by the service

runtime and should not require the service runtime to provide a library mapping the service information into a new registry or repository specific format. Making a service registry and repository easy to populate with service information from all the service runtimes used in an organization is one key design point for a service registry and repository standard.

Having established a service registry and repository that is populated with approved services also enables a new class of invocation patterns that make use of the loose coupling in SOA to mediate or route service requests to providers selected from a subset of the repository entries.

Taking advantage of this loose coupling to allow for some runtime configurable invocation patterns, such as policy based service routing is another use case driving service registry and repository standards.

## ***5.1 Service registry and repository standards relevant to use by service runtimes***

The publication of service information from runtimes into a service registry should be identical to the methods described above in the standards for service definition, selection and reuse. The additional performance requirements of accounting for runtime invocation patterns can also be achieved through the standards described above making greater use of the relationship and category information in the service registry and repository.

For any particular service interface, there may be multiple implementations of that service interface in a given environment. These implementations could be delivered with varying properties in the service agreement such as response time of the service. If the variations in the service implementations are described as a set of SLA policies, the service infrastructure, such as an enterprise service bus (ESB), could select a particular service endpoint address based on the policies or agreements related to the service endpoint.

Another key requirement when runtimes query the service registry and repository is to keep the number of interactions to a minimum. The service registry and repository specifications will provide a query language that can resolve the combination of service interface information and policies to service endpoints in a single call. It is further useful to a runtime to only receive the fragment of information that will be used to route a message to the service, such as the service endpoint or URL. The subset or fragment part of the runtime query can be met through the use of XPath specified filters on the interaction with the service registry and repository. The resolution of related documents requires some additional specification work on a query dialect that provide functions to resolve and evaluate the contents of related documents in a single XPath call.

The service registry and repository specification will need to detail new query dialects that allow for navigation of the relationships in the service description.

## **6 The role of a service registry and repository in service management**

Service management and monitoring systems will need to search the registry for types of services that can be enhanced through management or monitoring. The service management system may also add observed characteristics of the service to the service definition. It is expected that multiple implementations of management and monitoring tools would interact with a service registry and repository, resulting in another area where standards need to be employed for the service registry and repository. The management system will likely be searching for services based on their status in the service lifecycle to determine if there is a new service requiring monitoring. The service registry and repository specifications will describe a method and content model to associate service lifecycle information with the technical service descriptions in the service registry and repository.

Having established some management relationship with a service found through the service registry and repository, management systems should be able to add information to the service definition, including but not limited to, the relationship of the service to the management system and observed performance of the service. The service registry and repository data model must allow for the complete definition of a service to be provided from multiple sources throughout the lifecycle of the service.

## **7 Federation of service registries and repositories**

Within a given organization, it is likely that multiple service registries and repositories will exist as a result of organizational, geographic or other reasons that typically result in some duplication of IT components. It is also likely that some applications will require information from both a service registry and repository and another repository, such as a development asset repository or a configuration management database. In this environment, it is likely that multiple and overlapping data sources exist for a service definition.

To simplify the navigation of the service definitions in an enterprise, the vision is to create a registry and repository federation specification that allows clients to simply create and configure virtual service registries and repositories of various scopes and topologies. Each participating service registry and repository can interact with and read data from any other participant, subject to constraints established by the implementing IT organization. The resulting federated data provides a base for organization wide service selection, reuse, management and operational resiliency. Data sources beyond just service

registry and repository over a broad spectrum of complexity should be able to participate in this federation if the specification mandates only a few behaviors, additional capabilities are optional and it supports extensibility.

In this vision, many service registries and repositories exist, each of which can be considered an authoritative source for some set of service information. The federation model should allow for incremental evolution toward the use of shared data. While organizations will have the option but not the requirement to consolidate key data from many sources into a service registry and repository, the specifications must acknowledge the distribution of the complete service description across multiple sources. This approach to creating a federated service registry and repository accommodates diverse tools and processes.

In addition to specifying a means to incrementally federate content, it is also necessary consider existing standard sources for service descriptions. The UDDI registry standard provided one means of describing services and support for the UDDI specifications exists today in multiple vendor products (see [\[UDDI\]](#) ). The following section describes how to integrate and federate the content of a UDDI registry with the portion of the service description that is contained in a service registry and repository.

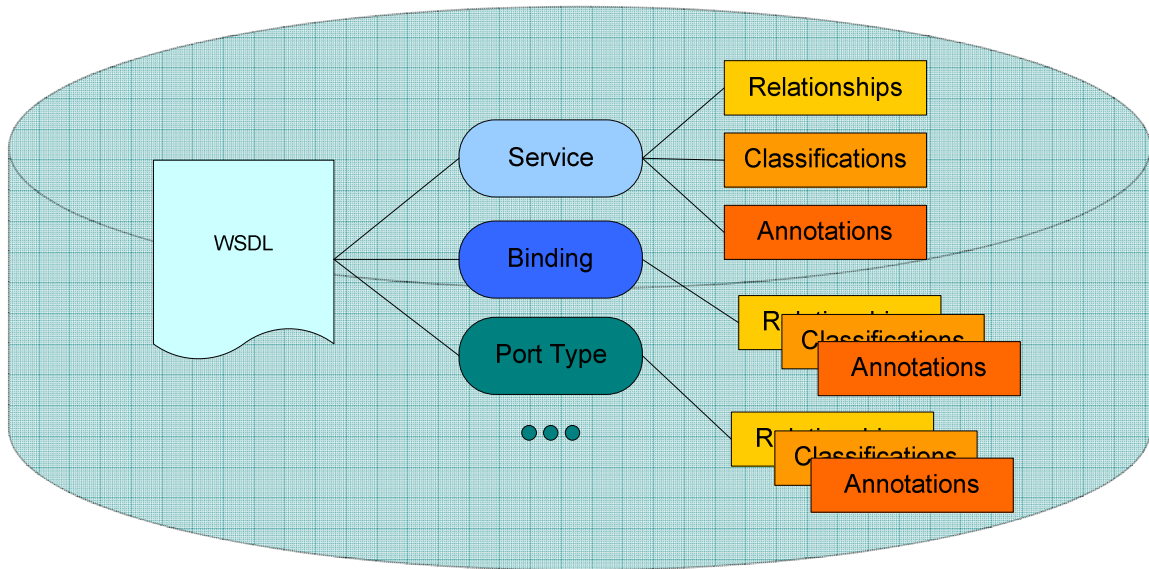
## ***7.1 Relationship of service registry and repository and UDDI***

In addition to the Web service specifications mentioned above, a service registry specification does exist today that is targeted at a subset of the use cases described above. The Universal Description, Discovery and Integration (UDDI) specification defines a registry that evolved from an Internet service directory to a Web services registry specification. UDDI provides a model for service description and corresponding methods to manipulate and query the UDDI model. The SOA requirements described above go beyond the capabilities of UDDI specifically in that it requires integration with repository and service lifecycle.

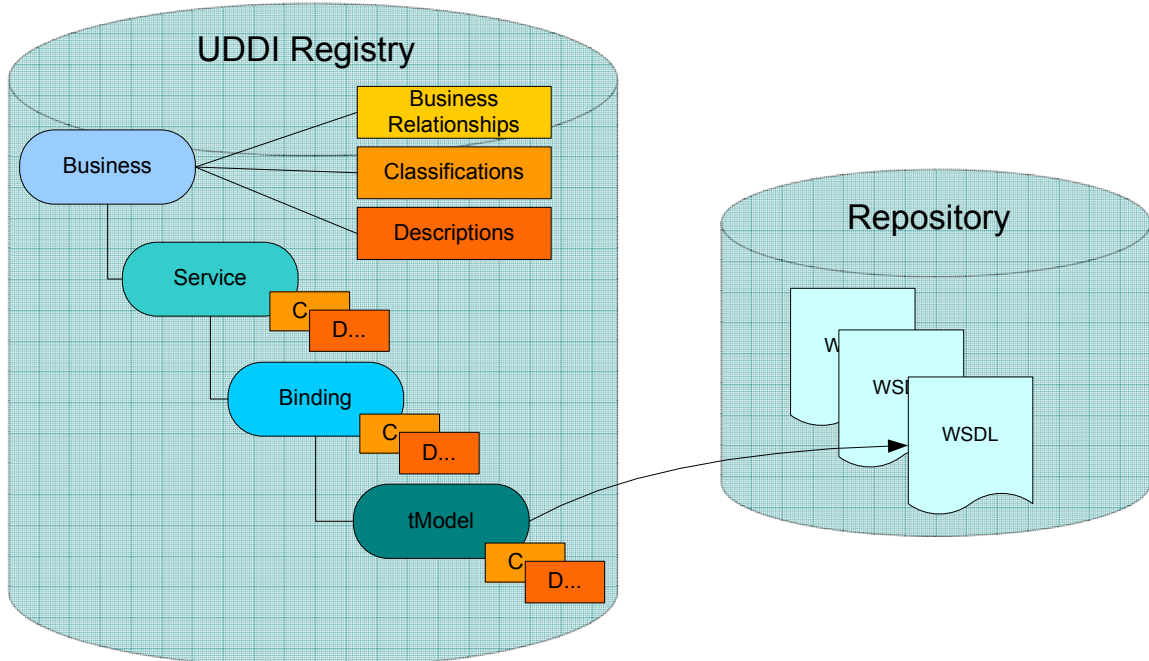
It is necessary to recognize that UDDI is a mature service registry standard and some organizations have made it a part of their architecture model. This section will describe how to map content and integrate UDDI with a service registry and repository

A documented practice exists for mapping pieces of information from WSDL and XSD documents into the UDDI data model to build a registry entry. The remaining part of the UDDI data model is focused on business ownership of the services and less focused on the variance in the technical details which are abstracted into a single structure, the

tModel.



**Figure 6. – Service information stored in service registry and repository where the WSDL file can be queried according to any of it's main types, and further any part of the WSDL can be annotated, related or classified**



**Figure 7 – Service as stored in the UDDI Registry under a business entity with external reference to the actual WSDL document.**

Figure 6 illustrates that each significant part of the WSDL model including the Service, Binding, Port Type and Operation elements can be queried in the service repository. Any given item in the WSDL model in the repository can be decorated with relationships, classifications and properties to further differentiate the registry contents. It is important to note that the user defined information in these two figures is similar enough that if either model is populated it is possible to import, export and possibly even synchronize information between the two models

### **7.1.1 Integration of the Service Registry Repository with UDDI**

Organizations needing the features of a service registry and repository that also have existing UDDI infrastructure or tooling can move towards a combined service registry and repository in stages. The different stages of integration are as follows and will be discussed in detail.

1. **Manual Repository Integration with UDDI**– Publishing of WSDL and WS-Policy to the repository and a separate publish of tModels to UDDI. In this scenario, the repository can serve as the physical storage of files that are referenced by URL in the UDDI registry. Part of the value to the organization would be the storage and control point for WSDL files. To enable both the repository query and UDDI query, the organization must maintain some physical process that defines what gets registered into the UDDI registry
2. **Automatic Tight Integration with UDDI** – Integrate the implementation of the UDDI registry with the repository. The automatic tight integration of the repository with UDDI would require the coupling of these two data models. This would mean that all information in one registry would be mapped completely into the other registry and would be kept in sync at all times. The effort to automate this might result in constricting the repository model to a point where every SOA artifact must be “owned” by a single business organization unit. This has proven unrealistic when capturing the nature and relationships of service artifacts in many SOA deployments. The business ownership also prevents bulk harvesting of service information from runtimes since it requires user defined information on business ownership to be defined before services can exist in the repository.
3. **Automatic Loose Integration** – Automatic import of data from UDDI to the repository and the automatic export of data from the repository to UDDI. The advantages are similar to the manual integration, however the process is enhanced with UDDI subscription and some automation of what subset gets mapped between the UDDI registry and the repository. One key challenge is how to map each subset of repository data under the rigid business structure imposed by UDDI. Assuming the organization can establish some consistent convention that makes sense within their organization this loose integration could be accomplished.

The key advantage to this third approach is to allow UDDI tooling to continue to access the service information, while not constraining what can be stored in a repository to the UDDI business structures that do not accurately reflect or model the structure of an organization.

The current approach is to move from the manual integration and interoperability that exists today towards automatic loose integration and interoperability of the service registry and repository and UDDI registries. This will enable most organizations to benefit from the stability of the UDDI data model while not applying a relationship in the master copy of the service information that enforces a single organization to service relationship.

## 8 Summary

A number of key specifications and standards related to interoperability of service registries and repositories exist today. These include the resource and metadata specifications which themselves build on the robust landscape of Web services specifications. Further, in the modeling space, a mature model for classifying services also exists with the OWL standard. This standard forms one key part of a data model that goes beyond the technical documents that need to be stored in a service registry or repository.

It is clear, however, that additional work remains to achieve a fully interoperable service registry and repository. This additional work is in establishing an industry definition of the data model for service registry and repository that aligns with WS-MetadataExchange and provides a means to maintain user defined annotations, classifications and relationships along with the technical documents such as WSDL and XSD. The new or extended queries that could be carried using WS-RT will also need to be standardized along with that service registry and repository data model.

Previous specifications efforts have laid out a significant foundation of understanding what is important to include in a service registry data model. The emergence of new specifications that provide a common mechanism to manipulate resources now enables the service information repository to be standardized as well. The result of combining the critical features and experience from service registry efforts and using emerging standards related to service and resource management will establish a more functional point of integration with interoperability that does not exist today.

## 9 References

### [OWL]

D. McGuinness, et al. "OWL Web Ontology Language Overview" February 2004. (See <http://www.w3.org/TR/owl-features/> )

### [SOA Foundation]

R. High, et al. "IBM's SOA Foundation" November 2005. (See <http://www-128.ibm.com/developerworks/webservices/library/ws-soa-whitepaper/> )

### [SOAP]

M. Gudgin, et al, "SOAP Version 1.2 Part 1: Messaging Framework," June 2003. (See [http://www.w3.org/TR/2003/REC-soap12-part1-20030624/.](http://www.w3.org/TR/2003/REC-soap12-part1-20030624/))

### [UDDI]

L. Clement, et al, "UDDI Version 3.0.2" October 2004. (See [http://uddi.org/pubs/uddi\\_v3.htm](http://uddi.org/pubs/uddi_v3.htm) )

### [WS-Addressing]

M. Gudgin, et al, "Web Services Addressing 1.0 (WS-Addressing)," May 2006. (See [http://www.w3.org/2005/08/addressing/.](http://www.w3.org/2005/08/addressing/))

### [WSDL 1.1]

E. Christensen, et al, "Web Services Description Language (WSDL) 1.1," March 2001. (See [http://www.w3.org/TR/2001/NOTE-wsdl-20010315/.](http://www.w3.org/TR/2001/NOTE-wsdl-20010315/))

### [WS-Management Roadmap]

IBM, HP, Intel and Microsoft "Evolving Web services standards for managing system resources" (See <http://www-128.ibm.com/developerworks/webservices/library/specification/ws-roadmap/> )

### [WS-MetadataExchange]

K. Ballinger, et al, "Web Services Metadata Exchange (WS-MetadataExchange)", August 2006. (See <http://schemas.xmlsoap.org/ws/2004/09/mex.>)

### [WS-Policy]

S. Bajaj, et al, "Web Services Policy Framework (WS-Policy)," September 2004. (See <http://schemas.xmlsoap.org/ws/2004/09/policy.>)

### [WS-ResourceTransfer]

I. Robinson, et al, "Web Services Resource Transfer" August 2006. (See <http://schemas.xmlsoap.org/ws/2006/08/resourceTransfer/> )

### [WS-Security Roadmap]

IBM and Microsoft "Security in a Web Services World" (See [www.ibm.com/developerworks/library/ws-secmap/](http://www.ibm.com/developerworks/library/ws-secmap/))

### [XML]

T. Bray, et al, "Extensible Markup Language (XML) 1.0 (Second Edition)", October 2000. (See <http://www.w3.org/TR/2000/REC-xml-20001006.>)

### [XML Schema]

H. Thompson, et al, "XML Schema Part 1: Structures," October 2004. (See [http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/.](http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/))

### [XPath1]

J. Clark, et al,, "XML Path Language (XPath) Version 1.0", November 1999. (See <http://www.w3.org/TR/xpath> )

### [XPath2]

A. Berglund, et al,, "XML Path Language (XPath) 2.0", June 2006. (See <http://www.w3.org/TR/xpath20> )

## Copyright Notice

© 2006 International Business Machines Corporation. All rights reserved.

This is a preliminary document and may be changed substantially over time. The information contained in this document represents the current view of International Business Machine on the issues discussed as of the date of publication. Because IBM must respond to changing market conditions, it should not be interpreted to be a commitment on the part of IBM, and IBM cannot guarantee the accuracy of any information presented after the date of publication.

The presentation, distribution or other dissemination of the information contained in this document is not a license, either expressly or impliedly, to any intellectual property owned or controlled by IBM and/or any other third party. IBM and/or any other third party may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. The furnishing of this document does not give you any license to IBM's or any other third party's patents, trademarks, copyrights, or other intellectual property. The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, places, or events is intended or should be inferred.

This document and the information contained herein is provided on an "AS IS" basis and to the maximum extent permitted by applicable law, IBM provides the document AS IS AND WITH ALL FAULTS, and hereby disclaims all other warranties and conditions, either express, implied or statutory, including, but not limited to, any (if any) implied warranties, duties or conditions of merchantability, of fitness for a particular purpose, of accuracy or completeness of responses, of results, of workmanlike effort, of lack of viruses, and of lack of negligence, all with regard to the document. ALSO, THERE IS NO WARRANTY OR CONDITION OF TITLE, QUIET ENJOYMENT, QUIET POSSESSION, CORRESPONDENCE TO DESCRIPTION OR NON-INFRINGEMENT OF ANY INTELLECTUAL PROPERTY RIGHTS WITH REGARD TO THE DOCUMENT.

IN NO EVENT WILL IBM BE LIABLE TO ANY OTHER PARTY FOR THE COST OF PROCURING SUBSTITUTE GOODS OR SERVICES, LOST PROFITS, LOSS OF USE, LOSS OF DATA, OR ANY INCIDENTAL, CONSEQUENTIAL, DIRECT, INDIRECT, OR SPECIAL DAMAGES WHETHER UNDER CONTRACT, TORT, WARRANTY, OR OTHERWISE, ARISING IN ANY WAY OUT OF THIS OR ANY OTHER AGREEMENT RELATING TO THIS DOCUMENT, WHETHER OR NOT SUCH PARTY HAD ADVANCE NOTICE OF THE POSSIBILITY OF SUCH DAMAGES.