Requirements and Services for Metadata Management

Knowledge-intensive applications pose new challenges to metadata management, including distribution, access control, uniformity of access, and evolution in time. The authors identify general requirements for metadata management and describe a simple model and service that focuses on Resource Description Framework (RDF) metadata to address these requirements.

Paolo Missier, Pinar Alper, Óscar Corcho, Ian Dunlop, and Carole Goble
University of Manchester, UK

In important application domains, data and service providers are increasingly making their resources publicly available to the community for re-use in complex workflows. To make those resources useful in practice, however, providers must also provide annotations that describe the data and services' nature and function. This is the case for e-science, the realm of *in silico* experiments or "procedures that use computer-based information repositories and computational analysis tools to test a hypothesis, derive a summary, search for patterns, or demonstrate a known fact."[1]

In e-science, it's quite common for providers and consumers to independently add annotations to resources to facilitate their discovery or to record details of their use as part of an experiment. Thus, when scaled to hundreds or thousands of resources and users of those resources, the annotations themselves will form a new and large corpus of heterogeneous metadata distributed over many organizations, with no central control over its maintenance. As a new and complex type of data resource, such metadata requires some form of management to be of any practical use.

In this article, we present a middleware service for metadata management that addresses the issue. Its design is based on the observation that, regardless of their differences in format and content, two simple properties are common to all metadata: namely, that it's invariably associated to some underlying resource, and, optionally, separate meta-information for interpreting metadata — an ontology, for instance — might be available. We refer to such meta-information as a *knowledge entity* to underline the fact that it's used to interpret the metadata. This is

 1089-7801/07/$25.00 © 2007 IEEE

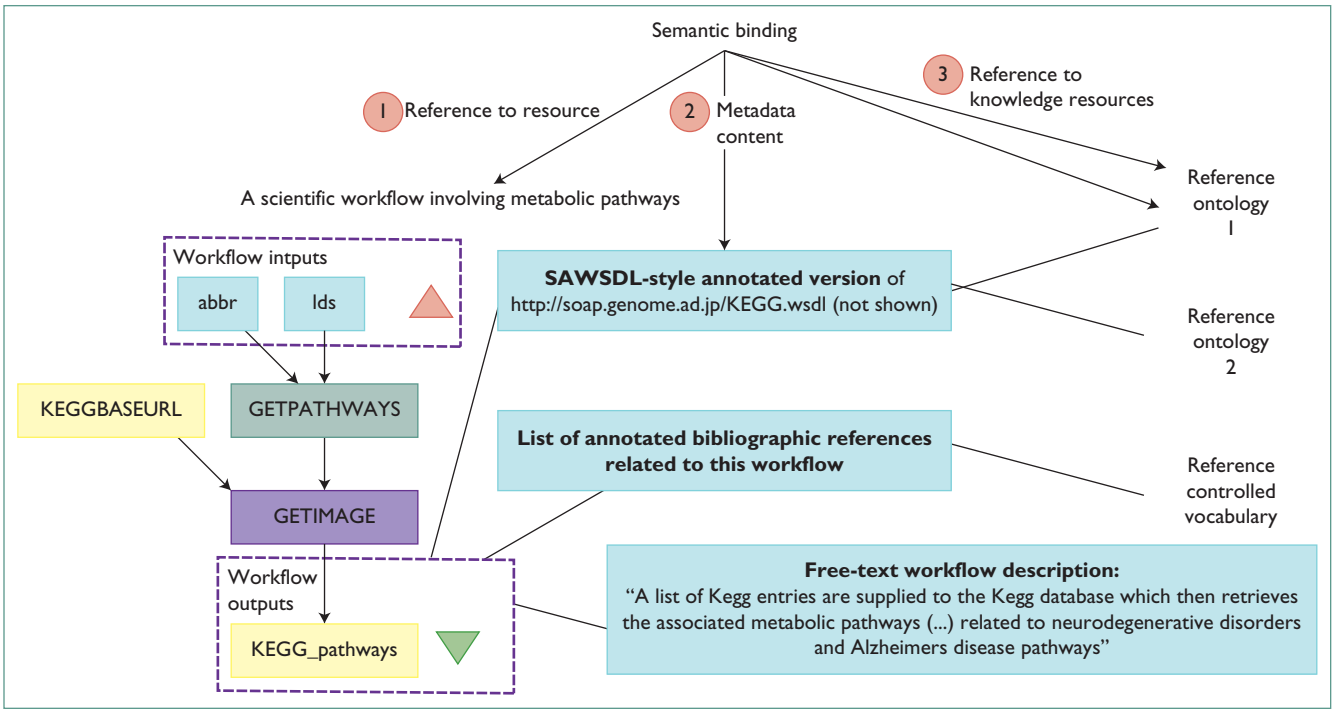

Figure 1. Multiple semantic bindings for a single scientific workflow. Workflow providers and users might provide multiple annotations for the same workflow that serves different purposes, and possibly adopt different annotation styles and formats. These efforts result in multiple semantic bindings, which all share the same reference to a resource, but differ in metadata content and in the references to knowledge resources.

typical but not exclusive of a Semantic Web setting: a schema for relational or XML metadata is also, by this definition, a knowledge entity. We refer to the association between resources and annotation metadata as a *semantic binding*, and to the management service as a *semantic binding service* (SBS).

A Service-Oriented Approach to Metadata Management

To illustrate the need for annotations, let's take a look at the myGrid infrastructure for e-Science (www.mygrid.org.uk). myGrid offers a middleware services suite to facilitate the specification of in silico experiments, mainly in the bioinformatics domain.[2] In particular, biologists can compose and execute scientific workflows like the one in Figure 1 (the workflow is part of a myGrid collection, available at http://workflows.mygrid.org.uk/repository/myGrid) that orchestrate access to multiple resources — public databases and data analysis tools, for example — and use them to derive biologically significant results. We can view myGrid workflows, specified using the Taverna workflow model,[3] as a composition of Web services.

Hundreds of Taverna services for biology, for example, are currently available through bioinformatics domain experts' contributions. Service providers can create several types of annotations to facilitate the effective reuse of resources, from natural-language service annotations designed to enable their discovery by domain experts to more formal annotations of the Web Services Description Language (WSDL) interface service to support the automatic composition process.

Let's consider the ways in which we could annotate the workflow example in Figure 1. First, we might have a natural language, informal workflow description. A provider could use this to share the workflow within a social networking environment, as is done in the myExperiment project (www.myexperiment.org). Second, we could annotate each composing service by augmenting its WSDL interface description with new attributes that described the service messages and operations' semantic properties. The provider could achieve this realistic scenario by using the annotation model proposed by the W3C's Semantic Annotations for Web Services Description Language Working Group (SAWSDL; www.w3.org/2002/ws/sawsdl). In this model, we can annotate interface elements using a small number of new WSDL

Current Metadata-Management Approaches and Technologies

Efforts on developing storage and querying technologies for Resource Description Framework Schema (RDF(S)) data dominate the state of the art on metadata management. Jena[1] and Sesame[2] are the most well-known and widely-used examples; these systems, underpinned with traditional relational data storage capabilities provide rich, fine-grained APIs for manipulating and accessing RDF data, as well as for querying it with different languages, including W3Cs Sparql (www.w3.org/TR/rdf-sparql-query). Oracle 10g[3] RDF is a recent addition, with support for very large data sets and rich querying capabilities combining the strengths of RDF query languages with native relational queries. In addition to basic capabilities, enhancements in the areas of contextualization, distribution, and scalability have also attracted attention.

Named Graphs API for Jena (NG4J; http://sites.wiwiss.fu-berlin.de/suhl/bizer/ng4j/) and Sesame2 (www.openrdf.org) provide for grouping of RDF statements into named graphs based on contextual information of choice (for example, ownership). Gergely Adamku and Heiner Stuckenschmidt present a paper[4] in which they developed a mediator layer that provides data-location transparency to clients while operating over multiple Sesame repositories. In Min Cai and Martin Frank's paper,[5] they exploit data structures and P2P protocols networks to store and query very large amounts of RDF(S) data in a fault-tolerant manner. These are only some of the many systems being developed for basic metadata management tasks.

Although these efforts contribute to robust storage and querying of RDF(S) data, the aspects of service encapsulation, secure and controlled access, transactions, replication, metadata evolution/change-management, and change propagation strategies often receive minimal attention. Most recently, we've seen the beginnings of work in some of these essential areas in initiatives such as IBM's Boca RDF store (http://ibm-slrp.sourceforge.net), or the W3C Sparql Protocol (www.w3.org/TR/rdf-sparql-protocol). Boca RDF provides service-based access to an RDF(S) store with replication, transactions, versioning, and change-tracking and notification capabilities. The W3C Sparql protocol is a very lightweight Web service interface for submitting Sparql queries to RDF stores. Yet, with all of these technologies still under active development and testing, industrial-strength,

continued on p. 19

extension attributes, which are typically references to concepts in some ontology. Building on service annotations, we can then formally annotate entire workflows. Third, we could annotate the workflow with a reference list of related work — published papers that describe the scientific hypotheses and experiments that the workflow is designed to support, for example. We could tag these citations using terms from some controlled vocabulary. Finally, in addition to workflows, you can also annotate data resources. A notable example is the Uniprot database (www.uniprot.org), in which expert curators provide annotations that describe protein function and structure.

When we scale these annotation efforts to very large numbers of resources, new metadata management issues begin to emerge. We focus on three such issues: storage and retrieval of heterogeneous metadata, metadata evolution and lifetime management, and access control to metadata.

Independent researchers can freely add new metadata, often without the help of tools that might otherwise enforce some homogeneity in format and content types.[4] Formats can range from free text to Javadoc pages, XML documents, and Resource Description Framework graphs (RDF; www.w3.org/RDF). The latter is the standard used to describe resources in Semantic Web applications — often in conjunction with Web Ontology Language (OWL) ontologies, although we can indeed write valid RDF triples whose elements (subject, object, and property) aren't defined elsewhere. We refer to RDF graphs whose elements are interpreted in the context of some ontology as *semantic metadata*.

Metadata evolution follows naturally from the evolution of biological knowledge itself, in which the description of a service that retrieves, say, "Kegg pathways," might change when an e-scientist refines the reference ontology that describes the concept of a metabolic pathway. This represents a potential problem for applications that rely on metadata to retrieve all experiments regarding a particular concept.

Finally, regarding access control, we note that privacy issues with metadata follow from access restrictions on the resources it describes — that is, the description of a service whose access is restricted to a defined set of users might also need to be restricted.

Metadata Management Requirements

Our SBS's goal is to provide a uniform set of primitives for the management of metadata resources — that is, creating, accessing, tracking the lifetime of, and destroying those resources. Note, however, that

Current Metadata-Management Approaches and Technologies, cont.

real-world applications don't make use of their metadata-management capabilities.

Several models exist for maintaining the association of explicit metadata with the resources that it describes; however, they rely on ad hoc mechanisms. One of the most common ways to include metadata in HTML or XHTML documents is to use the `<meta>` element anywhere in the document, as Ben Adida and Mark Birbeck describe.[6] This element allows the description of document properties using property–value pairs. Normative properties include author, expiration date, keyword lists, and so on, although we can use other user-defined properties (such as those from Dublin Core). This was also extensively used in pre-Semantic Web initiatives like Knowledge Annotation Initiative for the Knowledge Acquisition Community ((KA)^2) and Simple HTML Ontology Extensions (SHOE). XHTML 2.0[7] proposes an alternative way to link entities and their descriptions so that metadata isn't intertwined with the document contents, rather it's contained in a separate file or separate part of the document itself. Files containing metadata can be attached to XHTML documents by using the `<link>` element in the head of the document. The properties `rel`, `about` and `href` allow reference to resources inside or outside the document.[6]

References

4. G. Adamku and H. Stuckenschmidt, Implementation and Evaluation of a Distributed RDF Storage and Retrieval System," *Proc. 2005 IEEE/WIC/ACM Int'l Conf. Web Intelligence* (WI 05), IEEE CS Press, 2005, pp. 393–396.
6. B. Adida and M. Birbeck, "RDFa Primer 1.0: Embedding RDF in XHTML," tech. report, W3C working draft, May 2006; www.w3.org/TR/xhtml-rdfa-primer/.
3. J. Axelsson et al., "XHTML 2.0," tech. report, W3C working draft, July 2006; www.w3.org/TR/2006/WD-xhtml2-20060726/Overview.html.
2. J. Broekstra, A. Kampman, and F. van Harmelen, "Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema," *Proc. Int'l Semantic Web Conf.* (ISWC 02), LNCS 2342, Springer-Verlag, 2002, pp. 54–68.
5. M. Cai and M. Frank, "RDFPeers: A Scalable Distributed RDF Repository Based on A Structured P2P Network," *Proc. Thirteenth Int'l World Wide Web Conf.* (WWW 04), ACM Press, 2004, pp. 650–657.
1. J.J. Carroll et al., "Jena: Implementing the Semantic Web Recommendations," tech. report HPL-2003-146, HP Labs, 2003.
7. E. Chong, S. Das, G. Eadon, and J. Srinivasan, "An Efficient SQL-based RDF Querying Scheme," *Proc. 31st Int'l Conf. Very Large Data Bases* (VLDB 05), VLDB Endowment, 2005, pp. 1216–1227.

it doesn't provide interoperability among heterogeneous metadata, which is a problem in its own right because different RDF annotations (each a graph of RDF triples) can refer to different ontologies, thus making it difficult for third-party applications to integrate them. Rather, the SBS offers a uniform way to maintain correct associations among resources, metadata, and knowledge entities whenever they change, regardless of the differences in format and content among the metadata elements.

The Open Grid Services Architecture Data Access and Integration project (OGSA-DAI; www.ogsadai.org.uk) managed by the DAIS Working Group at the Open Grid Forum, presents a similar but more limited approach, in which a single interface with abstract operations is defined for standardized access to any data resource in a Grid computing environment, regardless of schema or data model. The standard includes several realizations to provide concrete, model-specific access to the data — OGSA-DAIR for relational data access and, more recently, OGSA-DAI-RDF for RDF data. The OGSA-DAI model provides uniform access to heterogeneous data resources by encapsulating queries within a generic service while remaining agnostic regarding data models and content.

The SBS follows a similar pattern, but it provides additional concrete metadata-management primitives. Existing metadata repositories — document-management servers, RDF storage managers, or file systems — provide low-level data management that's unaware of the data's role. The SBS leverages this functionality by offering a uniform metadata-management layer that is aware of the relationships among resources, their annotations, the annotations' lifetimes, and the reference knowledge entities.

We've implemented the work we describe here in a full prototype as part of the S-OGSA architecture for the Semantic Grid[5] proposed by the European Union-backed OntoGrid project (www.ontogrid.net). The implementation is deployed within the Globus Toolkit 4 (GT4) Grid computing infrastructure, and provides specific support for RDF metadata and Sparql Protocol and Query Language queries (Sparql; www.w3.org/TR/rdf-sparql-query/).

SBS's benefit to applications that store and access metadata is twofold: it offers semantic bindings as a simple, uniform model to maintain a stateful (that is, lifetime-aware) relationship between resources and their annotations and offers a single-service interface for management, which we can deploy in a distributed fashion within a Grid computing environment.

The metadata-management requirements that drive our design include:

- *Metadata distribution*. Metadata is naturally distributed because multiple organizations should be able to produce and store it. Most metadata-management systems provide repositories that are designed for centralized storage and use, with metadata consumers and producers acting as local-access clients using specialized APIs.
- *Uniform access*. Metadata access and management should be uniformly encapsulated within a single type of service that exposes simple common properties — specifically, associating the metadata with some underlying resource and, optionally, with knowledge entities that enable its interpretation: ontologies.
- *Metadata evolution*. Metadata is naturally dynamic; annotations on a document often update or supersede others, for example. Yet, although some work has been done with respect to the semantics of propagating data updates to associated metadata, current technologies poorly support metadata dynamics.[6,7] Stateful metadata management should involve detecting when it becomes invalid, and informing consumers of the state changes. Completely automating these tasks might not be possible, but the management architecture should make performing them as cost-effective as possible.
- *Access control*. Metadata producers might enforce access restrictions on the metadata they produce or store. A choice of existing mechanisms should be available from distributed computing standards, such as Web services and service-oriented Grids, to meet the requirement of a common access-control model. These include the Extensible Access Control Markup Language (XACML: www.oasis-open.org/committees/xacml), WS-Security (www.oasis-open.org/committees/wss/), and open-source reference implementations such as the Globus Security Infrastructure (www.globus.org/toolkit/docs/4.0/security) for global user identification, single-sign-on, communication encryption and representation, and decisions about resource-sharing policies.

Liming Chen and coauthors have proposed architectures that address some of these requirements,[8] but they focus exclusively on managing service annotations for service discovery — a more specific goal than capturing arbitrary metadata types. Unlike in our approach, they assume the presence of semantic annotations that applications can interpret according to some reference ontology. Moreover, their proposed architecture doesn't address issues such as invalidation of stale annotations. Perhaps more relevant for future work,[9] Divesh Srivastava and Yannis Velegrakis recently proposed representing associations between data and metadata as SQL queries rather than maintaining extensional lists of associations, assuming that the resources to be annotated are stored in relational databases.

The Metadata Management SBS

The SBS is a stateful service that manages a collection of semantic bindings that represent associations between data and metadata and between metadata and knowledge entities.

- *Data and metadata*. This is the relationship between a resource's unique identifier and its descriptive metadata. We assume that the same metadata might contribute to describing multiple resources, and different SBs might describe a given resource or set of resources.
- *Metadata and knowledge entities*. Metadata might be associated with a set of schemas or ontologies. In the particular but important case of knowledge entities that are OWL ontologies, coupled with RDF metadata, an application might use the knowledge entities to perform formal reasoning on the metadata — using an OWL reasoner,[10] for example — to match a service description to users' requirements.

Elsewhere, Óscar Corcho and colleagues present a detailed SB model,[5] in which the only assumption on the metadata's nature is that an application used to generate annotations can encapsulate it within an SB, regardless of the SB's format (RDF, natural language, social tags, and so on).

Recall the annotated workflow in Figure 1, for example: the SAWSDL-based annotation results in an SB of the form, `WF`, [`<list of SAWSDL annotations>`, `<list of ontology references>`] in which we assume that all `WF` references to the workflow and annotated WSDL files are through URI-formatted IDs (in the implementation, the SBS identifies the SB, a first-class resource, using a WS-Addressing endpoint reference). An annotation application can express the reference list of papers for the same workflow using an RDF graph representing the relationship between a paper's ID and its topics, resulting in an SB of the form, [`WF`,

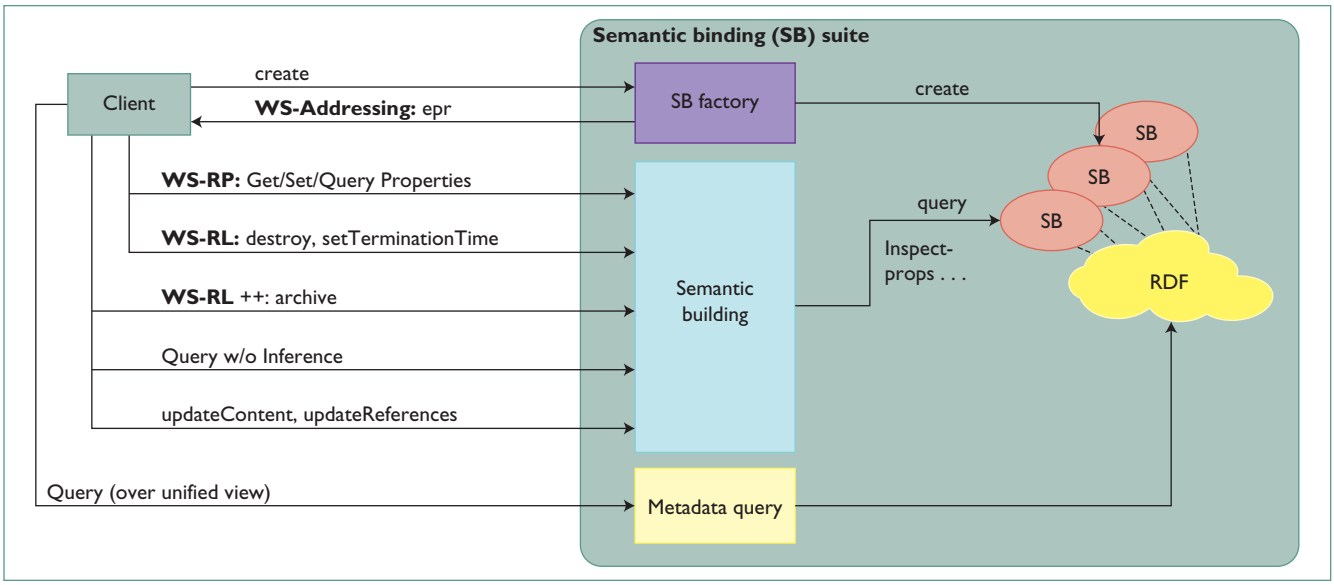

Figure 2. Semantic binding service (SBS) functionality. The WS-Resource Properties (WS-RP) specification defines protocols for accessing and querying a resource's dynamic state information. The WS-Resource Lifetime (WS-RL) specification defines very basic interfaces for destruction of WS-Resources.

`<citation list>`, `<references to controlled vocabularies>]`. Note that a free text annotation might not have any references to knowledge entities — that is, `WF`, `<annotation text>`, `_`.

The SBS is the single contact point for metadata-aware applications wishing to submit new metadata to be represented as an SB, and to retrieve SBs and their associated metadata content. Figure 2 shows the SBS's functionality, which includes creating and destroying SBs and maintaining their logical state with regard to metadata lifetime (described in detail later). The SBS also provides service-based access to metadata content by forwarding application-specific queries, specified as part of the service request, to the underlying metadata repository without interpretation. This is consistent with the SBS's generic nature, which is unaware of the specific data model used for the metadata. The current implementation, however, specifically supports RDF data and can issue application-provided Sparql queries to several interchangeable RDF stores, such as Sesame and OpenRDF (www.openrdf.org).

To generalize the approach, we envision adding a simple introspection extension to the SBS interface to let clients discover metadata-specific metainformation, such as access language and schema. We haven't implemented this feature yet.

Combining SBs and an SBS enables the seamless distribution of metadata and provides a common access interface. This uniformity of access makes it possible to broaden the deployment to a federation of services, providing large-scale access to metadata regardless of each individual back-end metadata repository's limitations.

Lifetime Management for Semantic Bindings

Metadata evolution concerns the dynamic nature of many metadata types and suggests that we should model SBs as stateful resources and notify clients when a SB's state changes. Note that each SB component might change independently. Recall in Figure 1 that the workflow is annotated in different ways. Now consider a modified workflow where the designer has added or removed some services, possibly resulting in invalid SAWSDL annotations of those services. When new relevant publications describing the workflow appear, the citation list also needs updating (the second annotation). The referenced ontologies might also change, causing it to become invalid, as well.

Finally, the annotation application might explicitly give an SB a limited lifetime, perhaps because we know some annotations will become invalid or unreliable after a certain time period — even if we aren't aware of new relevant publications, for example, we might decide to automatically invalidate a list of references when it becomes too old. In each of these cases, a manual or automatic validation process is required to make sure that the SB remains valid.

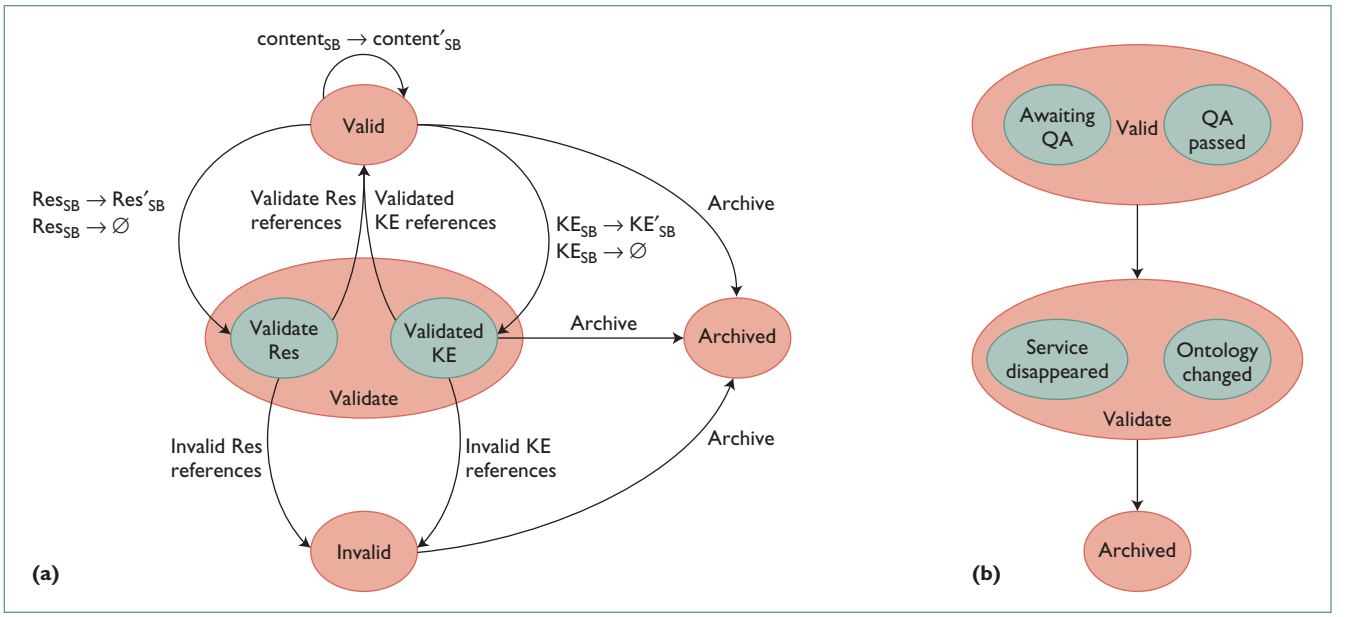

Figure 3. State-transition diagram for (a) generic semantic bindings (SBs) and (b) myGrid-specific SBs. Each SB has a state that applications can inspect to verify that it's safe to use or valid. The arc labels denote external events that cause the SB to change its state. Some state transitions cause a dedicated application to perform validation on the SB, possibly resulting in a new state.

These considerations lead to the definition of the state diagram shown in Figure 3a, in which we denote the set of data and knowledge resources that are part of the association as Res_{SB} and KE_{SB}, respectively, and the metadata within SB as $content_{SB}$. On creation, the SB is in the valid state. The diagram illustrates transition-triggering events as $Res_{SB} \rightarrow Res'_{SB}$ (change in the data resource), $KE_{SB} \rightarrow KE'_{SB}$ (change in the knowledge resource), and $content_{SB} \rightarrow content'_{SB}$ (change in the metadata content). Note that the latter always leads to a new valid SB.

An SB in one of the two interim validate states calls for a validation process, which updates any or all of Res_{SB}, KE_{SB}, or $content_{SB}$, resulting in a transition to a valid or invalid state. For a `ValidateRes` SB, such a procedure determines whether the existing metadata is still valid for the new resources and provides an update to the SB references to Res'_{SB}. For example, following a change in workflow that's annotated with metadata, the procedure determines whether the same metadata can be associated with the new workflow. For a `ValidateKE` SB, the problem is to determine whether we can use the new ontology to interpret the old metadata. The problem of assessing the ontology evolution's impact on an existing knowledge base has been addressed elsewhere.[6,7] Finally, note that the archived state provides a way to retain old SBs that are no longer valid. This state will be supported only by implementations that use some versioning mechanism, in which an archived SB might be brought back to an active state.

We can extend this basic model by introducing substates, resulting in finer-grain definitions of metadata behavior. In myGrid, for example, both the valid and validate states are extended with two substates, as Figure 3 shows. The substates distinguish between metadata that's been reviewed by human experts — quality assurance (QA) — and metadata that's awaiting QA. Note that, in both cases, the metadata is indeed valid in the sense that the annotation is plausible. The substates add explicit information regarding the annotation's quality, which some applications might want to take into account. As Khalid Belhajjame and coauthors point out,[11] this is important with automatically generated annotations that require experts' inspection prior to their release.

New state transitions might also be associated with the substates:

- $KE_{SB} \rightarrow KE'_{SB}$ triggers the invocation of a change-detection tool[12] that analyzes the SB content and issues a report to the annotator.
- A transition to the awaiting QA state triggers a notification to the annotator to carry out the QA task.

To alert the application when SBs require attention, the SBS uses the notification component, which we describe next.

SB State Change Notification

Along with state management, the SBS also provides a notification service to inform interested clients of any changes in the state of SBs, according to the proposed WS-Notification standard (www.oasis-open.org/committees/tc_home.php). In this model, clients can subscribe to any set of predefined topics, such as "state of a specific SB changed to `ValidateRes` due to an update in a data resource D." In the current model, the client must initiate revalidation actions on receiving a notification because the SBS doesn't attempt to interpret the metadata and thus can't associate any behavior with state transitions.

In our workflow example, an application that subscribes to topics associated with the SB might partially automate these actions. On receiving notification of a change to the workflow, the application might interact with the user to create annotations for the new services or update existing ones. Likewise, the user might be alerted to update the affected annotations when ontologies change.

Our current prototype will soon include a new SB housekeeping service, which will act as a single activation point for all revalidation processes. Recognizing that some states have similar meaning to many applications, the service will subscribe to all the available SBS topics and react to notifications by launching application-specific validation processes.

SB Access Control

To address the need to provide fine-grained access control to metadata resources, we note that having defined metadata as first-class resources makes available to us all the standard access-control mechanisms from the underlying middleware, including the aforementioned XACML and WS-Security. In particular, the current SBS implementation supports the Grid Security Infrastructure (GSI) framework. As an example of a fine-grained access-control mechanism, we could specify that scientists from an agency can't access metadata for resources that have been requested by scientists in another agency unless access is granted explicitly.

Implementing and Using the SBS

We implemented the SBS as a stateful S-OGSA service,[5] compliant with the Web Service Resource Framework standard (WSRF).

The choice of metadata storage largely determines the effort required to support more advanced SBS functionality — specifically, access control and state-change notification. In particular, Sesame doesn't currently offer any native support for the Globus security framework (www.globus.org/toolkit/docs/4.0/security/), nor does it include event generation following RDF statement updates. An alternative implementation that might simplify the effort is IBM's Boca (http://ibm-slrp.sourceforge.net/), an RDF management engine that offers a rich programmatic interface based on OpenRDF. Boca uses the DB2 relational database management system natively, and we can configure it to use other database back ends, as well. Although it doesn't support the AuthZ framework, Boca offers traditional rule-based access control at the data level. In addition, Boca 2.0 now includes a change-detection subsystem, based on Sun's Java Messaging Service, that clients can use to receive notifications of changes to single RDF statements or to entire named graphs.

From the application developer's viewpoint, we argue that it would take limited effort to refactor existing metadata-intensive applications to adopt SBS as a single point of contact. Regardless of their specific architectures, such applications are generally characterized by collections of metadata producers — resource-annotation components and metadata consumers, such as service-discovery components. The main refactoring steps are writing adapters as clients of the SBS and obtaining X.509 certificates from third-party certification authorities, which GSI recognizes and the SBS accepts. Reacting to change-event notifications is likely to require the development of a new component, as well.

We've implemented an early prototype of our service-oriented model for format- and location-independent metadata management, and we're testing it on myGrid and myExperiment use cases.

We're enhancing the current design in several ways. We're simplifying security management based on GSI by providing predefined configurations to cover common cases. We're also redesigning the current SB naming scheme (based on WS-Addressing EPRs) to follow the WS-Naming specification, which extends WS-Addressing with

the use of URIs. Finally, we're adding a new SB housekeeping service that represents a single (logical) delivery point for notifications and manages registered third-party applications that are activated in response to SB state changes.

Additionally, we're addressing engineering issues of scalability and effectiveness by migrating the SBS from the current in-memory GT4 storage model for SBs to a scalable model supported by a persistent back end, and by deploying a federation of distributed and cooperating SB services.

Acknowledgment

This work was partially supported by OntoGrid, a European Union FP6 project (STREP 511513) in the area of Grid-based systems for solving complex problems.

References

1. M. Greenwood et al., "Provenance of e-Science Experiments – Experience from Bioinformatics," *Proc. UK OST e-Science All Hands Meeting* (AHM 03), Simon Cox, ed., Engineering and Physical Sciences Research Council, 2003, pp. 223–226.
2. C. Wroe et al., "A Suite of DAML+OIL Ontologies to Describe Bioinformatics Web Services and Data," *Int'l J. Cooperative Information Systems*, special issue on data management and modeling support in bioinformatics, vol. 12, no. 2, 2003, pp. 197–224.
3. T. Oinn et al., "Taverna: A Tool for the Composition and Enactment of Bioinformatics Workflows," *Bioinformatics*, vol. 20, no. 17, 2004, pp. 3045–3054.
4. S. Bechhofer et al., "The Semantics of Semantic Annotation," *Proc. 1st Int'l Conf. Ontologies, Databases, and Applications of Semantics for Large-Scale Information Systems* (ODBASE), LNCS 2519, Springer-Verlag, 2002, pp. 1152–1167.
5. Ó. Corcho et al., "An Overview of S-OGSA: A Reference Semantic Grid Architecture," *J. Web Semantics*, vol. 4, no. 2, 2006, pp. 102–115.
6. L. Stojanovic, *Methods and Tools Ontology Evolution*, doctoral thesis, Univ. Karlsruhe, 2004.
7. M. Klein and F. N. Noy, "A Component-based Framework for Ontology Evolution," *Proc. 18th Int'l Joint Conf. Artificial Intelligence: Workshop on Ontologies and Distributed Systems* (IJCAI 03), CEUR-WS, 2003.
8. L. Chen et al., "Managing Semantic Metadata for Web Grid Services," *Int'l J. Web Services Research*, vol. 3, no. 4, 2006, pp. 73–94.
9. D. Srivastava and Y. Velegrakis, "Intensional Associations between Data and Metadata," *Proc. ACM SIGMOD Int'l Conf. Management of Data* (SIGMOD 07), ACM Press, 2007, pp. 401–412.
10. I. Horrocks and U. Sattler, "Ontology Reasoning in the SHOQ (D) Description Logic," *Proc. 17th Int'l Joint Conf. Artificial Intelligence* (IJCAI 01), Morgan Kaufmann, 2001, pp. 199–204.
11. K. Belhajjame et al., "Automatic Annotation of Web Services Based on Workflow Definitions," *Proc. 5th Int'l Semantic Web Conf.* (ISWC 06), LNCS 4273, Springer-Verlag, 2006, pp. 116–129.
12. P. Plessers and O. De Troyer, "Ontology Change Detection Using a Version Log," *Proc. 4th Int'l Semantic Web Conf.* (ISWC 05), LNCS 3729, Springer-Verlag, 2005, pp. 578–592.

Paolo Missier is a research fellow at the School of Computer Science, University of Manchester. His research interests include data and data quality management, and distributed and service-oriented software architectures. Missier has a MSc in computer science from the University of Udine, Italy and from the University of Houston, Texas. He has been an ACM and IEEE member for nearly 10 years. Contact him at pmissier@cs.man.@ac.uk.

Pinar Alper is a research associate with the Information Management Group at the School of Computer Science, University of Manchester. Her research interests include semantic web services and service-oriented Grids. Alper has a MSc and M.Phil. in computer science from the University of Manchester. Contact her at Pinar.Alper@manchester.ac.uk.

Óscar Corcho is a research fellow at the Information Management Group of the University of Manchester. His research interests include the semantic grid, the semantic web and ontological engineering. Corcho has a PhD in artificial intelligence from Universidad Politécnica de Madrid (UPM), Spain. Contact him at ocorcho@cs.man.ac.uk.

Ian Dunlop is a software engineer at the School of Computer Science, University of Manchester. His research interests include Grid computing, web services, web 2.0, and software engineering principles. Dunlop has a MSc in mathematical sciences from the University of Paisley, UK. Contact him at Ian.Dunlop@manchester.ac.uk.

Carole Goble is a professor in the School of Computer Science, University of Manchester and the coleader of the Information Management Group. Her research interests include the Semantic Web, e-Science, and the Semantic Grid. Goble is the director of the large UK e-Science myGrid/Taverna program of work for workflow based middleware for life scientists. She is the co-chair of the Open Grid Forum Semantic Grid Research Group and the Technical Director of the EU Strep OntoGrid. She sits on 15 national and international oversight committees concerning e-Science, bioinformatics, semantic technologies and Grid computing.