Web Service Information Systems and Applications

Mehmet S. Aktas[1,2], Galip Aydin[1,2], Geoffrey C. Fox[1,2,3], Harshawardhan Gadgil[1,2], Marlon E. Pierce[1], and Ahmet Sayar[1, 2]

[1]Community Grids Laboratory
[2]Department of Computer Science
[3] Department of Physics
Indiana University
{maktas, gaydin, hgadgil, mpierce, asayar}@cs.indiana.edu
gcf@grids.ucs.indiana.edu

1. Introduction: Information System Requirements for Web Service Grids

As the Service Oriented Architecture (SOA) principles have gained importance, an emerging need has appeared for methodologies to locate desired services that provide access to their capability descriptions. Also, as these services interact with each other within a workflow session to produce a common functionality such as earthquake prediction [PI], another emerging need has also appeared for storing, querying, and sharing the resulting metadata needed to describe session state information.

In SOA-based Grids, Information Services support the discovery and handling of both static and session-related, transitory metadata associated to services. Here, we discuss the limitations in existing Information Services and introduce a novel system as a solution. We design a hybrid Information Service supporting both large amounts of relatively slowly varying data and rapidly updated, dynamically generated information.

Geographical Information Systems provide very useful problems in supporting "virtual organizations" and their associated information systems. These systems are comprised of various archival data services (Web Feature Services), data sources (Web-enabled sensors), and map generating services. All of these services are metadata-rich, as each of them must describe their capabilities (What sorts of features do they provide? What geographic bounding boxes do they support?). Organizations like the Open Geospatial Consortium define these metadata standards. These services must typically be assembled into short-term service collections that, together with code execution services, are combined into a meta-application (i.e. a workflow). Thus we see that we have both stateless and stateful (transient) metadata.

To address these problems, we use and extend two Web Service standards to provide information services: Universal Description, Discovery, and Integration (UDDI) and Web Services Context (WS-Context) in our design. We utilize existing UDDI Specifications [UDDI] and design an extension to UDDI Data Structure and UDDI XML API to be able to associate both prescriptive and descriptive metadata with service entries.

There have been some solutions introduced to provide better retrieval mechanism by extending existing UDDI Specifications. UDDIe [UDDIe] project introduces the idea of associating metadata and lifetime with UDDI Registry service descriptions where retrieval relies on the matches of attribute name-value pairs between service description and service requests. UDDI-M^T [UDDI-MT] improves the metadata representation from attribute name-value pairs into RDF triples. A similar approach to leverage UDDI Specifications was introduced by METEOR-S [METEOR] project which identifies different semantics when describing a service, such as data, functional, quality of service and executions. In our design, we also extend UDDI's Information Model, by providing an extension where we associate metadata with service descriptions similar to existing solutions where we use name-value pairs to describe characteristics of services. Apart from the existing methodologies, we provide both general and domain-specific query capabilities. An example for domain-specific query capability could be XPATH/RDQL queries on the auxiliary and domain-specific metadata files stored in the UDDI Registry.

The main intended use of our approach is to support information in dynamically assembled workflow-style Grid applications where services are tied together in a dynamic workflow to solve a particular problem. There are varying specifications, such as WSRF, WS-Context, WS-Transfer, and WS-Metadata Exchange, introduced to define stateful interactions among services. Among them, we choose WS-Context Specifications [WS-Context] to create a metadata catalog system for storing transitory metadata needed to describe distributed session state information. Unlike the other specifications defining service communications, WS-Context models a session metadata repository as an external entity where more than two services can easily access/store highly dynamic, shared metadata.

2. Research System Architecture

We designed and built a novel architecture [Aktas2005a] for a WS-Context complaint metadata catalog service supporting distributed or central paradigms. The main intended use of our approach is to support information in dynamically assembled Grids where services are tied together in a dynamic workflow to solve a particular problem. Also, the intended scale for our design is in the order of thousand entities that are participating in a workflow session.

We based the programming interface of our system on two widely used specifications: WS-Context and Universal Description, Discovery and Integration (UDDI). We extend both specifications to provide advanced capabilities to support handling and discovery of not only quasi-static, stateless metadata, but also session related metadata. Our approach is to utilize the existing state-of-art systems for handling and discovering static metadata and address the problems of distributed management of dynamic metadata.

Our architecture consists of various modules such as Query and Publishing, Expeditor, Access, Storage and Sequencer Modules. The Context Query and Publishing Modules receive/process client requests through a uniform web service interface for publishing/discovering dynamic and static metadata. The Expediter Module is a generalized caching mechanism. One consults the expediter to find how to get (or set) information about a dataset in an optimal fashion. The Access Module supports request distributions by publishing messages to topics in a topic-based publish-subscribe based brokering network. It locates the nodes that are closest in terms of network distance with lowest load balance from the node requesting access to the communal node in question. The Storage Module runs storage algorithm which decides whether a metadata is to be replicated. If the metadata is decided to be replicated, then Storage module advertise this replication by multicasting it to available peers through publish/subscribe mechanism. The Sequencer Module ensures that an order is imposed on actions/events that take place in a session. The Sequencer Module interacts with Storage Module and labels each metadata which will be replicated in this replicated metadata hosting environment. For further detail reading on design details and architecture of Information Services, we refer the reader to [Aktas2005a].

3. Implementation Details

We implemented a centralized version of Information Services handling discovery of both static and dynamic, session related metadata. We use/extend both UDDI and WS-Context Specifications in our design.

The Information Service applies following programming logic to serve client requests, upon receiving querying/publishing metadata requests. First, the system separates dynamic and static portions of the metadata. Here, static metadata could be throughput or location of a service whereas dynamic metadata could be session identifier pointing to a workflow session in which the service is participating. Second, the system delegates the task of handling and discovery of static portion of the metadata to one of the existing state-of-art technologies handling with static metadata such as UDDI, a widely accepted and WS-I compatible standard. As for the UDDI Service, we use our own implementation of Extended UDDI XML Metadata Service [Aktas2005b]. Third, the system itself provides handling and discovery of dynamic metadata, using session-related portions of the metadata query.

Here, session related metadata is short-lived and dependent on the client. The Information Service keeps track of context information shared between multiple participants in Web Service interactions. The context here has information such as unique ID and shared data. It allows a collection of action to take place for a common outcome.

Each session is started by the coordinator of an activity. The coordinator service publishes the session metadata to Information Service and gets a unique identifier in return. The uniqueness of the session-id is ensured by the Information Service. Sessions can obviously be composed from other “sub” sessions hierarchically. Here, each session is associated with the participant services of that session. Dynamic session information, i.e. context, travels within the SOAP header blocks among the participant entities within the same activity. In order to correlate (collective) work participants may propagate more contexts using the same session-id created by the coordinator.

Upon receiving the system response to a request for session creation, the user can store the context associated to the unique session identifier assigned by the Information Service. This enables the Information Service to be queried for contexts associated to a session under consideration. Each context is stored with a lifetime as the Information Service is being used as third-party repository for dynamic information.

4. Usage Scenarios

In order to present the applicability of our system, we outline following example usage scenarios. The first example use domain is the PI GIS Grid. This example illustrates a workflow based GIS Grid Application where session metadata needs to be managed to enable services to better interact with each other for common outcome. The second usage domain is The Virtual Laboratory for Earth and Planetary Materials (VLab) Application. This example illustrates a Grid/Web Service-based system for enabling distributed and collaborative computational chemistry and material science applications where the user inputs needed to be archived as serialized Java Bean Objects within a session to enable users to access/reuse their previously stored user-system interactions. The third usage domain is the HandHeld Message Service and Handheld Flexible Representation Project [HHMS]. This project investigates a fast web service communication model in collaborative mobile computing environment.

In the first example usage domain, Information Services are used for storing transitory metadata needed to describe distributed session state information. In the current test system, it is used to store information needed by a workflow engine (HPSearch) to orchestrate system interactions. The HPSearch is simply a scripting environment for managing distributed workflows. Also, geospatial data sources are available online through Geographical Information System (GIS) enabled data services (Web Feature Services). These data services provide different data and data formats with varying spatial coverage. Here, Information Services provide domain specific metadata catalog for GIS domain and enable users to pose queries to locate GIS data.

In the second example application, Information Services are used as a lightweight, Web Services based archival data store. VLab project forms a computational grid environment where users may upload input data to execute scientific applications on remote computers. The input data is usually given trough input form pages which are tedious to fill out and it is more likely that uses will have minor changes on the input parameters to a particular job and resubmit it later. The input data can be preserved as serialized Java Bean Objects to be reused later in user-system interaction. Here, each Java Bean Object is simply being stored as “context”, i.e., metadata associated with a session into WS-Context Information Service. Contexts may be arranged in parent-child relationships. When storing a context, we first create a session in the WS-Context Information Service. Here, a session can be considered an information holder; in other words, it is a directory where contexts with similar properties are stored. Similar to an access control list in a UNIX file system, each session directory may have associated metadata, called “session directory metadata.” Session directory metadata describes the child and parent nodes of a session. This enables the system to track the associations between sessions. One can create a hierarchical session tree where each branch can be used as an information holder for contexts with similar characteristics. This enables the WS-Context compliant Information Service to be queried for contexts associated to a session under consideration. Each context is stored with unlimited lifetime as the Context Store is being used as an archival data store.

The third example use scenario is fast web service communication model in mobile communication environment. In this scenario, a user has a PDA, which is running a videoconferencing application packaged as a "lightweight" web service. Such service could be a conferencing, streaming, or instant messaging service. Here, the Information Service is being used as a third party transitory metadata store, to store/maintain redundant parts of the SOAP messages which are being exchanged between the mobile service and its clients. This way, the size of SOAP messages is being minimized to make the service communication much faster. The redundant parts of a SOAP message can be considered as XML elements which are encoded in every SOAP message exchanged among two services. These XML elements are stored as "context", i.e. metadata associated to a conversation", into WS-Context store. Each context is referred with a system defined URI where the uniqueness of the URI is ensured by the Information Service. The corresponding URI replaces the redundant XML elements in the SOAP messages which in turn reduce the size of the message for faster message transfer. Upon receiving the SOAP message, the corresponding parties in service conversation interact with WS-Context compliant Information Service to retrieve the context associated with the URIs listed in the SOAP message.

5. References

[PI] Tiampo, K. F., Rundle, J. B., McGinnis, S. A., & Klein, W. Pattern dynamics and forecast methods in seismically active regions. Pure Ap. Geophys. 159, 2429-2467 (2002).

[UDDI] Bellwood, T., Clement, L., and von Riegen, C. (eds) (2003), *UDDI Version 3.0.1: UDDI Spec Technical Committee Specification.* Available from http://uddi.org/pubs/uddi-v3.0.1-20031014.htm.

[UDDIe] Ali ShaikhAli, Omer Rana, Rashid Al-Ali and David W. Walker.., UDDIe: An Extended Registry for Web Services, Proceedings of the Service Oriented Computing: Models, Architectures and Applications, SAINT-2003 IEEE Computer Society Press. Oralndo Florida, USA, January 2003

[UDDI-MT] Miles, S., Papay, J., Dialani, V., Luck, M., Decker, K., Payne, T., and Moreau, L. Personalized Grid Service Discovery. Nineteenth Annual UK Performance Engineering Workshop (UKPEW'03), University of Warwick, Conventry, England, 2003.

[METEOR] Verma, K., Sivashanmugam, K. , Sheth, A., Patil, A., Oundhakar, S. and Miller, J. "METEOR–S WSDI: A Scalable P2P Infrastructure of Registries for Semantic Publication and Discovery of Web Services" , Journal of Information Technology and Management.TM03-006.pdf

[WS-Context] Bunting, B., Chapman, M., Hurlery, O., Little M., Mischinkinky, J., Newcomer, E., Webber J, and Swenson, K., Web Services Context (WS-Context), available from http://www.arjuna.com/library/specs/ws_caf_1-0/WS-CTX.pdf

[Aktas2005a] Mehmet S. Aktas, Geoffrey C. Fox, Marlon Pierce Information Services for Dynamically Assembled Semantic Grids Proceedings of 1st International Conference on SKG2005 Semantics, Knowledge and Grid Beijing China November 27-29 2005
Also: Please see, http://www.opengrids.org/wscontext/index.html

[Aktas2005b] Mehmet S. Aktas, Galip Aydin, Geoffrey C. Fox, Harshawardhan Gadgil, Marlon Pierce, Ahmet Sayar, Information Services for Grid/Web Service Oriented Architecture (SOA) Based Geospatial Applications, Technical Report, June, 2005
Also: Please see, http://www.opengrids.org/extendeduddi/index.html

[HHMS] Sangyoon Oh and Geoffrey C. Fox HHFR: A new architecture for Mobile Web Services: Principles and Implementations Technical report September 2005 Also, please see: http://grids.ucs.indiana.edu/ptliupages/hhms/