

Brokering Semantic Web Services via Intelligent Middleware Agents within a Knowledge-Based Framework

Randy Howard and Larry Kerschberg
George Mason University E-Center for E-Business, <http://eceb.gmu.edu/>
choward@gmu.edu, kersch@gmu.edu

Abstract

The concept of automating Web services, specifically the brokering activities, is an active research topic. We need a comprehensive and overarching framework that seamlessly incorporates intelligent middleware agents within the context of workflow management, and addresses the issues related to Virtual Organizations. The goal is to add semantics to Web services to endow them with capabilities currently lacking in the literature, but necessary for their successful deployment in future systems.

This paper discusses how the Knowledge-based Dynamic Semantic Web Services (KDSWS) Framework, interoperating with the Knowledge Sifter Architecture, can be used to dynamically broker Semantic Web Services. The Functional Agent Services Architecture of the KDSWS Framework contains Intelligent Middleware Agents to enable these dynamic operations. In particular, the specification of the roles of the intelligent middleware agents' in the brokering is presented

1. Introduction

The relatively new concept of *Web services* is important to both e-Business and e-Government in that applications may exchange functionality and information over the Internet using standard XML-based protocols to find publicly-registered services, understand their purpose and operation, negotiate and agree upon usage charges and Quality-of-Service commitments, and invoke the services within the context of Internet-based workflow coordination of these services. "Semantic Web Services" (SWS) is the term that describes our research approach. We view "Web services" as services that use little or no semantic markup, and have little of the enhanced capability described in this paper. Semantic Web technology, on the other hand adds semantic and process oriented information, together with heuristics and constraints that can be used to coordinate the activities of the Virtual Organization (VO).

This paper discusses how the Knowledge-based Dynamic Semantic Web Services (KDSWS) Framework [1], interoperating with the Knowledge Sifter Architecture [2], can be used to dynamically broker Semantic Web

Services. In particular, the specification of the roles of intelligent middleware agents' in the brokering is presented, where brokering requires that services register themselves with a broker, and the broker intermediates to match the request with available services.

Section 1 has introduced the problem, and Section 2 discusses some of the drawbacks and issues associated with existing approaches. The Knowledge-based Dynamic Semantic Web Services Framework is summarized in Section 3. Section 4 presents the processes and data necessary for agents to collaborate for the brokering portion of the framework's methodology, and is couched within the components of the KDSWS Framework. Section 5 presents our conclusions and suggests areas for future research.

2. Related Work

Agents may be thought of as software programs [3] that act on behalf of humans, robots, or other programs, and exhibit autonomous, purposeful behavior. In order to realize the vision of an Virtual Organization consisting of Web services that can be composed dynamically, we need both a comprehensive framework together with a methodology that brings together the concepts of the Semantic Web, multi-agent systems, workflow systems and the functional approach to both data and knowledge management [4].

In order for the various protocols involved in delivering Web services, as shown in [5], to achieve an end-to-end solution involves linking the many disparate protocols and technologies, despite the fact that they are all still based on the XML foundation. The nuances of these various technologies still need to be mediated to some level to be truly interoperable. These approaches do not specify aspects regarding repositories and agents.

As mentioned, SWS attempt to address the shortcomings of Web services with respect to automation, and OWL-S is being established as a primary specification for this endeavor. However, recent research [6] has documented several shortcomings attributed to OWL-S.

3. KDSWS Framework

The KDSWS Framework proposes to deliver a comprehensive and integrated end-to-end solution to dynamically deliver SWS because the 'protocol-independent' KDSWS Design Specification allows the enterprise-wide data, process, backend, middleware and service needs to be stipulated. The data aspects are captured in the meta-model and are modeled by the Knowledge/Data Model and Language (KDM/KDL) [7]. The process aspects are captured in the KDSWS Methodology and are modeled by the Knowledge-based Dynamic Services/Process Model and Language (KDSPM/KDSPL) [1]. The meta-model focuses on developing the 'building blocks' (i.e., constraints, preferences, profiles, capabilities, etc.) versus focusing on the visible and finished products like a service, a profile, or request on which some approaches place their primary attention. This 'meta-model/process'-based approach enables an extensible framework that provides a macro-level shell to plug-in different approaches to the facets of the Web services life-cycle (e.g., configure transactions, policy management, etc.) yet still provides enough structure to guide implementations. The framework positions itself as an enterprise-scale foundation to ultimately provide VOs with the interfaces necessary to interoperate via Semantic Web Services using, and quickly adapting to, the plethora of prevailing technologies and protocols.

Ontologies and mediation are an integral part of this framework, and are manifested in the mapping and heuristics. The structure and design of the components target, or map to, ontological implementations such as OWL and OWL-S, albeit with potential extensions to facilitate the additional functionality. A primary function of the heuristics is to enhance the mappings ability to dynamically mediate between disparate components. This rules-based approach facilitates the dynamic generation of the framework's profiles (e.g., requestors, providers, situations, services, etc.) in order to drive the fulfillment of requests.

The KDSWS Framework uses an adapted version of the Functional Agent Services Architecture (FASA) [4] that is shown in Figure 1. Note that underlines denote components specified in the diagrams and that the elements that are relevant to this discussion are highlighted in green with dashed outlines. The FASA is a three-layer architecture that is predicated on an agency-based approach in which agents are organized in agencies, and contains virtual agents that specify agents' aggregation and coordinate the activities of other agents. Line agents are involved in work specifications and delegate support functions to support agents.

The Intelligent Middleware Services layer is supported by the Functional Services Agency whose agents include

line agents for Web service brokering and and composition. These agents receive the task specification and plan from the User Agency, and search for appropriate Internet-based Web services that can accomplish the tasks.

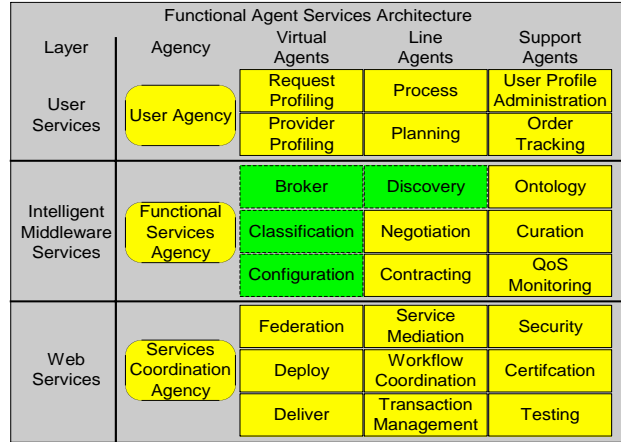


Figure 1. FAS architecture

The profiles of the various agents are embedded throughout the KDSPL. The mapping to agent profiles identifies the points where agents are specified and organizes into a profile that contains the responsibilities of the agent within a given context. From this profile, the specification for a given agent can be engineered into a working component with the necessary social interactions and responsibilities detailed.

4. Collaborating to Broker SWS

4.1. Brokering Context

The brokering activity consists of Discover and Select Tasks. This discussion, as depicted in Figure 2, details the collaborations involved with the Produce and Compile Search Results process portion of the macro Discover process. This diagram shows agent interactions in two manners: the activity internal to the agents conveyed within the vertical separator lines (e.g., Discovery Agent, Classification Agent and Decomposition Agent), and the interactions with agents external to the context of the diagram (e.g., Knowledge Sifter and Broker Agent). The methodology shows the processes within the context of the Management, Workflow and Feedback Threads. The flow of the KDL (data) objects is identified by the dashed line, where the execution of the KDSPL (process) objects is identified via the solid lines.

4.2. Knowledge Sifter Search Agent

Knowledge Sifter (KS) is a scaleable agent-based system that supports access to heterogeneous information sources such as the Web, open-source repositories, XML-

databases and the emerging Semantic Web. User query specification is supported by a user agent that accesses multiple ontologies using an integrated conceptual model expressed in the Web Ontology Language (OWL). A collection of cooperating agents supports interactive query specification and refinement, query decomposition, query processing, as well as result ranking and presentation. The Knowledge Sifter architecture is general and modular so that ontologies and information sources can be easily incorporated. Each Agent is implemented as a Web Service and the external sources are also accessed via Web Service Technology.

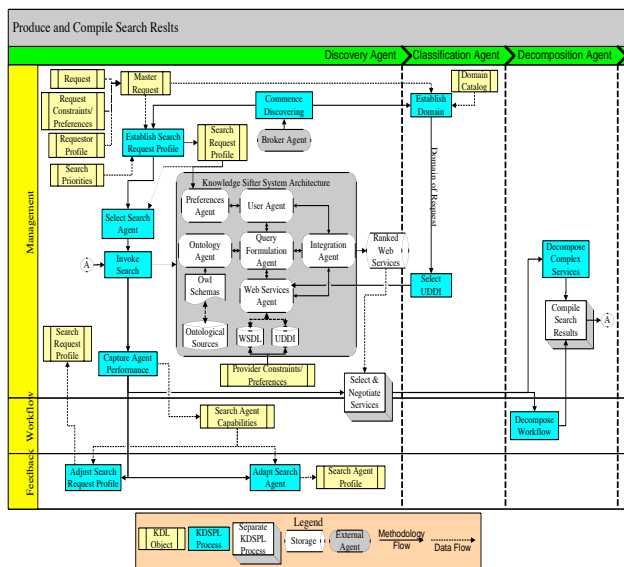


Figure 2. Methodology and agent interactions

Two particular agents are of particular interest to this research because the collaboration presents them as interfaces: the Preferences Agent and the Integration Agent. The Preferences Agent manages user preferences that are typically gathered by the User Agent interacting with the user. These preferences include the relative importance attributed to terms used to pose queries, the perceived authoritativeness of Web search engine results, and other preferences to be used by the Integration Agent.

The Integration Agent is responsible for compiling the sub-query results from the various sources, ranking them according to user preferences, as supplied by the Preferences Agent, for such attributes as: 1) the authoritativeness of a source which is indicated by a weight – a number between 0 and 10 – assigned to that source, or 2) the weight associated with a term comprising a query [2].

4.3. The Collaboration Process

The Commence Discovering process picks up its tasking from the Broker Agent which oversees the entire Discover process; therefore, it is the OWNER and the Discovery Agent is the STEWARD of the Produce and Compile Search Results process. The Discovery Agent delegates work to the Classification Agent when the Establish Domain process uses the Master Request and the Domain Catalog to determine the domain, which is in turn used to select a UDDI registry that best fits the request. The Master Request is an aggregate of such information as the original request, information maintained about the requestor and constraints and preferences of the request.

The Establish Search Request Profile uses the Master Request as well the Search Priorities that the requestor wishes to have on the various metadata fields (i.e. price might be deemed more important than availability). Since multiple search devices are available, Select Search Agent uses the Search Request Profile determine which search vehicle can best handle the request and the priorities set forth. In order to show interactions with an external agent, this discussion specifies that Knowledge Sifter is selected. Once the search resource (Knowledge Sifter, KS or short) is selected, the Invoke Search is started where the Search Request Profile is sent to the KS Preferences Agent (the preferences agent vs. the user agent because the profile is already structured and does not require human interaction) to help the search vehicle deliver more accurate results.

The KS architecture suggests that UDDI and WSDL be selected as the data sources because the desired Web resources are Web services. This approach proposes that the WSDL and UDDI be expanded with Provider Constraints/Preferences to provide increased coverage and more accurate and reliable matches.

The KS Integration Agent delivers the Ranked Web Services back to the main process's Select & Negotiate Services, which is another process that is detailed outside of this discussion. Because our methodology handles Web services that call other Web services (or Complex Web services) and those that specify workflow steps, the Decompose Complex Services and Decompose Workflow processes decompose Web services down to Simple Web services (those that have a specified endpoint). The call to this Decomposition Agent is another example of delegating work to support agents. The Compile Search Results, another external process, stores and compiles the results for further processing. Notice the on-page connector sends the flow back to the Invoke Search process in order to search for decomposed Web services that specify only a profile for a Web service versus an endpoint.

Because our methodology focuses on learning capabilities of agents, the Capture Agent Performance process captures the measurement of performance and accuracy of the results in Search Agent Capabilities. Adjust Search Request Profile process uses these measurements to refine the Search Request Profile for future searches. The Adapt Search Agent process refines the knowledge conveyed in the profile for the agent in Search Agent Profile.

5. Conclusions

In order to accelerate the introduction of new concepts and systems, we need approaches which automate the entire Web services life-cycle and address the issues related to Virtual Organizations. This paper presents an agent-based approach to managing the delivery of Semantic Web Services for use within a Virtual Organization, and is part of a larger methodology, called the KDSWS Framework. The framework provides a formal model-based approach to implementing Web services that specifies the modeling, specification, design, implementation and deployment of systems composed of SWS. The goal of the framework is to support the automatic brokering and management of Web services within the Virtual Organization in a protocol-independent manner. The agency-based approach of the Functional Architecture organizes agents functionally to support intelligent middleware services and Web services.

The framework provides a comprehensive solution because it addresses the backend specification for federating enterprise resources, agents and knowledge repositories. It is integrated because the data and process specifications are created using the same foundation - the KDM/KDL. Interoperability is facilitated by the integrated design space and mediation structures. The framework is knowledge-based because it uses heuristics to associate the knowledge to objects and services, and captures usage context as well. This rules-based approach facilitates the dynamic profiling of resources as well as quick adaptation to the rapidly-changing Web services standard and protocol base. The term 'semantic' denotes the knowledge-based semantic specification of relevant features and functions provided by the Web Service.

The KDSWS Design Specification viewpoint models objects, relationships, constraints, heuristics, and processes using the KDM/KDL and extensions KDSPM/KDSPL to handle special features of Semantic Web Service specifications, and the agents' responsibilities. The advantage provided by a separate and integrated specification is the VO can adjust to a constantly changing protocol base more rapidly by developing the mappings from the consistent base versus having to recode the affected interfaces.

The KDSWS Functional Architecture provides an agent-based architecture to implement systems via

composable Semantic Web Services. The architecture includes Functional Architectures for knowledge represented in repositories, federation policy, rules, agents, Web service protocols, and agent services to manage various aspects of deploying SWS.

This research is still at an abstract level, and an implementation of the framework is planned where the facets of the research can be matured. Future work for the framework includes creating a modeling facility for the unique approach of the methodology that integrates the meta-model.

Finally, our research indicates that the KDSWS semantic modeling techniques and methodology, when applied to service-oriented systems exemplified by Semantic Web Services, helps to address the plethora of issues needed to successfully deploy them in real-world applications. The semantically-enabled workflow and feedback processes provide a managed approach to the dynamic delivery of Web services. The multiple viewpoints help to isolate and identify important issues and the mappings from viewpoint to viewpoint assure that the structures, operations, and constraints are properly mapped and preserved.

Acknowledgements

This work was sponsored by a NURI from the National Geospatial-Intelligence Agency (NGA). This work was also supported in part by the Advanced Research and Development Activity (ARDA). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U. S. Government.

6. References

- [1] R. Howard and L. Kerschberg, "A Framework for Dynamic Semantic Web Services Management," *International Journal in Cooperative Information Systems*, Accepted for Publication, 2004.
- [2] L. Kerschberg, M. Chowdhury, A. Damiano, H. Jeong, S. Mitchell, J. Si, and S. Smith, "Knowledge Sifter: Ontology-Driven Search over Heterogeneous Databases," presented at SSDBM 2004, International Conference on Scientific and Statistical Database Management, Santorini Island, Greece, 2004.
- [3] M. Wooldridge, "Issues in Agent-Based Software Engineering," *Cooperative Information Agents*, pp. 1-18, 1997.
- [4] L. Kerschberg, "Functional Approach to in Internet-Based Applications: Enabling the Semantic Web, E-Business, Web Services and Agent-Based Knowledge Management," in *The Functional Approach to Data Management*, P. M. D. Gray, L. Kerschberg, P. King, and A. Poulouvassilis, Eds. Heidelberg: Springer, 2004, pp. 369-392.
- [5] L. Wilkes, "The Web Services Protocol Stack," CBDI Web Services Roadmap, 2004, URL: <http://roadmap.cbdiforum.com/reports/protocols/index.php>
- [6] M. Paolucci, J. Soudry, N. Srinivasan, and K. Sycara, "A Broker for OWL-S Web services," presented at 2004 AAAI

Spring Symposium Series, Stanford University Palo Alto, CA, 2004.

[7] W. D. Potter and L. Kerschberg, "The Knowledge/Data Model: An Integrated Approach to Modeling Knowledge and Data," in *Data and Knowledge (DS-2)*, R. A. Meersman and A. C. Semadas, Eds.: Amsterdam: North Holland, 1988.