Supporting Change-Aware Semantic Web Services

Annika Hinze
Department of Computer Science,
University of Waikato, New Zealand
a.hinze@cs.waikato.ac.nz

Abstract. The Semantic Web is not only evolving into a provider of structured meaningful content and knowledge representation, but also into a provider of services. While most of these services support external users of the SW, we focus on a vital service within the SW – change management and adaptation. Change is a ubiquitous feature of the SW. In this paper, we propose a service architecture that embraces and utilises change to provide higher quality services. We introduce pilot implementations of two supporting services within this architecture.

1 Introduction

Typical interaction with the Semantic Web [3] (SW) involves the retrieval of data using search engines and agents. However, these services do not monitor changes that occur on the SW. The SW can be expected to change in more ways and to a larger extent than the current Internet. Changes may occur in many SW components: content, knowledge representation, ontologies, sources, services and agents. Thus, services that simply retrieve content and metadata (knowledge and ontologies) are not sufficient. A new type of service is needed that supports the fast integration of new or changed components (services, agents and sources) and data (content and metadata). The service needs to provide the following functionalities: observation and filtering of changes in SW components and data, and active notification of interested parties (e.g, user agents or services). This functionality is typically provided by event notification services (ENS).

The support of *event notification services for the Semantic Web* is a vital enhancement of SW functionality in order to provide much-needed adaptivity. ENS inform interested parties about events that occurred at providers' sites. Events can be as diverse as the registration of a new service or changed web-page data. Our proposed event notification service offers content-based filtering of change information from diverse providers. We identify and address the following research challenges in the SW:

1. *Components:* Adaptation to and integration of changed sources, services or agents;
2. *Data:* Integration and distribution of changed content and metadata;
3. *Architecture:* Open infrastructure for aggregation and inter-operability of services.

To address these challenges, we propose the design of a service architecture which supports dynamic adaptation to changes in a Semantic Web context: adaptation to and change in both components and data. In this paper, we report about our initial steps towards an adaptive and change-aware SW service architecture. We focus initially on two services: the identification and consolidation of information about changed components, and the identification of changes within RDF/S documents.

The remainder of this paper is organised as follows: Section 2 shows that related work did not sufficiently addresses the challenges identified above. In Section 3, we introduce the high level design of the architecture and give details about the implementations of the two services for change awareness. Section 4 summarises the contributions of the paper and describes plans for future work.

2 State of the Art

We have highlighted the need for supporting change awareness and adaptivity in the Semantic Web. Despite the large research effort focussed on SW technologies, only a few projects address the challenges we identified here. Note that the consolidation of evolving ontologies constitutes a separate problem that has been widely recognized.

Development of frameworks and standards (e.g., RDF/S, XML, OWL, SOAP) are key efforts towards the Semantic Web [13]. To date, no standard method has been proposed for the announcement of new or changed components. One approach is DAML-S [7]: a Web Service Ontology and supporting tools that enable automation of SW services; agents and sources may be described similarly. Currently, agent descriptions are propagated in the network using an intermediary agent [11]; no common framework has been established for the propagation of source information. An ENS could control the propagation of change information in the SW: We propose a service for integrating information about data sources, services and agents with differing descriptions using approximate filtering techniques.

Data in the SW is typically represented in XML and RDF/S format. Existing XML-based notification services (e.g., NiagaraCQ [6], XFilter [1]) only process *new* documents submitted to the service; filtering of changes or deletions in the SW are not supported. Effective support for these more advanced features requires, for example, the identification of changes. This is more demanding than the identification of XML documents using a common query language (such as XML-QL (NiagaraCQ) or Xpath (XFilter)). Existing query languages for XML and RDF could be extended to filter languages. So far, no filter languages have been proposed to identify changes in XML documents. RDF data is typically queried using RQL or RDQL [9]; neither of them includes elements to query for changes in RDF documents. Currently, several researchers are attempting the enrichment of XML/RDF repositories with active functionality [4,10]. We believe that active features should address the demands of the SW services and not be primarily driven by a focus on storage systems. We therefore suggest the extension of RDF query languages with active components and the implementation of an event notification system for RDF/S data.

The subject of a service architecture for the semantic web has been addressed by the Semantic Web Services Initiative Architecture committee in a requirements analysis [2]. The requirements analysis introduces several use cases; the ones for ubiquitous computing (UC-U) and web (UC-W), respectively, address parts of the issues discussed here. UC-U points our the extremely dynamic variation in available web services, UC-W motivates dynamic discovery and combination of web services. However, both use cases fail to cover dynamic in other SW components or data. Consequently, the requirements developed fail to cover the concepts of change, client interest in changes (i.e.,

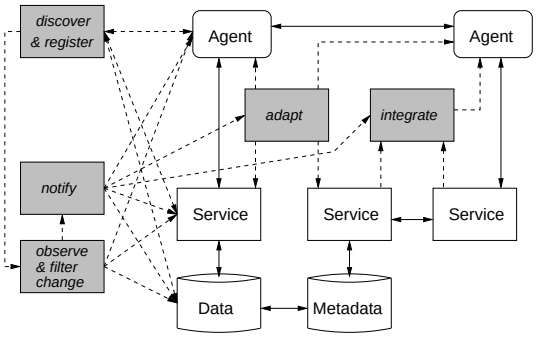

Fig. 1. Change adaptation: Concept of interactions between SW components (sources, services and agents) in white, with *supporting services* for change awareness and adaptation in gray

profiles), notification, integration, and adaptation. An extension of the requirements would be necessary, such as extending the description of 'candidate service query protocols' so that changes in existing services and the search for services over a period of time are supported. Several approaches have been proposed for managing SW services, such as SWSA [12] and WSMF [8]. They chiefly focus on selection and interaction of services; none of these covers the issue of change and adaptation.

3 Supporting Change-Awareness

This paper reports our initial steps within a project towards the design and implementation of an open infrastructure for aggregation and interpretability of SW services. In this section we will introduce our current architecture for a SW service framework that supports change awareness. For the pilot implementation, we focussed initially on two services for the integration of components and the identification of changes within RDF/S documents, respectively.

3.1 Architectural Considerations

The proposed service architecture has to support dynamic adaptation to changes within a Semantic Web context in both components and data. This includes the automatic registration of components, the propagation of changes at active components and the adaptation of existing components to a changing SW context. Here, we report on our initial design considerations.

Our service architecture incorporates a set of *supporting services* that are part of the Semantic Web and make use of the data, semantic annotations and ontologies. Figure 1 shows the concept for interactions between SW components and the supporting services. The architecture will provide services for identifying, filtering, and forwarding change information regarding components and data: Components have to register or be otherwise accessible, and interested parties may register an interest in new or changed

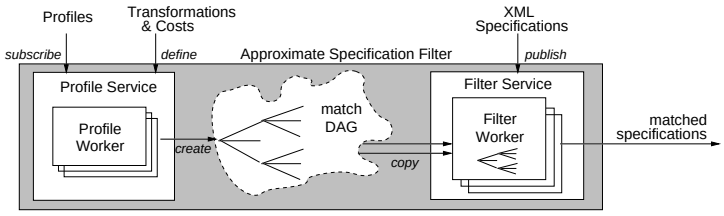

Fig. 2. Service for approximative filtering of component specifications in XML

components (*discover & register*); changes in the functionality/descriptions of components and in the data have to be detected (*observe & filter change*) and interested parties have to be notified (*notify*). The subsequent adaptation of components (*adapt*) may depend on their (temporally changing) context. The integration of information (*integrate*) should allow for changes/diversity in descriptions and data.

Service architectures have proven very successful in areas where general services can be reused for building specialized applications. The architecture will be open to services that embed and integrate existing or changed generic components as well as application-specific components. Possible application scenarios in the context of the Semantic Web are E-Learning, Tourist Information and Health Care.

We evaluated the option of extending existing architecture approaches. The concept of client profiles needs to be introduced and the supporting services indicated in Figure 1 need to be added (e.g., observe and adapt) or extended (discover and register). Several features of our architecture (changes in agents, sources, data) cannot be mapped into a simple extension of, e.g, SWSA. A comprehensive infrastructure is needed that incorporates these heterogeneous aspects.

3.2 Integration of Components: Approximate Specification Filtering

Components within the SW (such as data sources, services, and user agents) are described using differing specifications. As discussed above, changes in these specifications have to be filtered to be identified and integrated (service discovery and integration, see Figure 1). We see the most promising starting point for the integration of the differently structured information in the approximate filtering of the specifications. Approximate filtering allows for variations in the specifications' structures and data. We propose an XML filter service that uses approximate and ontology-based filtering.

Figure 2 sketches the design of our implemented ApproxFilter service for flexible integration of components in the SW: Clients subscribe using profiles that describe the specifications they are interested in and by defining allowed transformations in the data and structure of specifications (top left in Figure 2). A distributed profile service unit builds a matching tree from the profiles (extended by possible transformations). Our approximative filter is executed by distributed filter worker threads working on local copies of the filter tree. These threads accept incoming XML specifications and filter them according to the user profiles. If the resulting costs (for the necessary transformations to match the profile) are below a given threshold, the client receives the matching

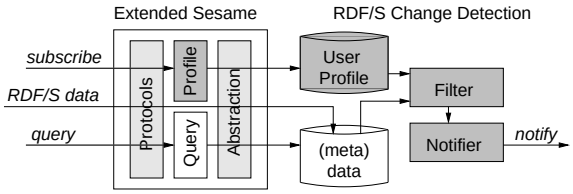

Fig. 3. Service for change detection in RDF/S data; original Sesame components in white, extended components in light gray and new components in gray

specification in a notification. By using the set of transformations and their assigned costs, the filters can be adapted to be strict, or allow for variations either in the data or the structure of the XML specifications or in both.

3.3 Identification of Changed Data: Filtering of RDF/S Data

Currently, change in SW data is mainly addressed for ontology integration. But changes on the data level (e.g, RDF and RDFS) are also influential for the cooperation of services. In addition, users and agents may be interested in the new or changed data. We argue that an event notification service is required for the detection of and alerting about change events in RDF/S documents (as shown bottom left in Figure 1).

We extended the RDF/S storage system Sesame [5] to detect changes in the stored RDF/S data. Figure 3 shows our extensions to the Sesame architecture and the additional components of the change detection service. The definition of profiles is an extension to the system: Sesame supported only querying for data. The client profiles are stored in a database. The filter component compares new and changed data with the stored client profiles. Internally, we use triggers in a relational database (Oracle) to detect changes in the stored RDF/S data.

Our current implementation supports 40 different event types. The detectable changes are new, deleted, and updated data; the affected items can be classes, types, resources, properties, and domains in RDF/S models. Simple data changes may have complex model implications; therefore, identification of changes in the RDF/S models involves filtering of data on several model levels. The service supports filter languages based on RDF triples or graphs, such as RQL and RDQL. Currently, we provide only restricted support for profile definition for external users. An extension of RDF/S query languages into filter languages is planned as a next step in the project.

4 Summary and Future Work

This paper has reviewed the current state of the art in SW systems, and identified a lack of support for change awareness in a dynamic environment. We discussed selected initial requirements for a change-aware architecture, and proposed a framework of supporting services that better addresses those requirements than either existing systems or modified forms of current systems could provide. To demonstrate the integration of

components and support of change awareness, we introduced an approximate filter service for XML specifications. Furthermore, we introduced an event notification service that complements existing SW facilities with the detection of and alerting about change events in RDF/S documents. We briefly described both service implementations.

The next steps in our project will see extensions of the two pilot services described here and further development of the architecture. The adaptive filter service for SW components will be extended to support the automatic integration of the identified components into the SW network. This service will also be used as a mediator between different services that offer different but similar functionality. We plan to extend the notification service for data to detect patters of events in RDF/S documents and to fully support RDF/S filter languages. Further work on the implementation of our service architecture will focus on inter-operability and standardisation issues.

Acknowledgements We would like to thank Takanori Kozuka and Yann Michel for their work on the pilot implementations.

References

1. M. Altinel and M. Franklin. Efficient Filtering of XML Documents for Selective Dissemination of Information. In *Proceedings of the VLDB, Cairo, Egypt*, Sept. 2000.
2. Architecture Committee. Semantic web services architecture requirements. Working draft, Version 1.0, June 2004. available at `http://www.daml.org/services/swsa/swsa-requirements.html`, last accessed in April 2005.
3. T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):35–43, May 2001.
4. A. Bonifati, S. Ceri, and S. Paraboschi. Active rules for XML: A new paradigm for e-services. *The VLDB Journal*, 10(1):39–47, 2001.
5. J. Broekstra, A. Kampman, and F. van Harmelen. Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In *Proceedings of the First International Semantic Web Conference (ISWC'02), Sardinai, Italy*, June 2002.
6. J. Chen, D. DeWitt, F. Tian, and Y. Wang. NiagaraCQ: A scalable continuous query system for internet databases. In *Proceedings of the ACM SIGMOD*, July 2000.
7. DAML Services Coalition. DAML-S: Web service description for the semantic web. In *The First International Semantic Web Conference (ISWC'02), Sardinia, Italy*, June 2002.
8. D. Fensel and C. Bussler. The Web Service Modeling Framework WSMF. Technical report, Vrije Universiteit Amsterdam, Computer Science Department, 2002.
9. P. Haase, J. Broekstra, A. Eberhart, and R. Volz. A comparison of RDF query languages. In *Proceedings of the International Semantic Web Conference, Hiroshima, Japan*, Nov. 2004.
10. G. Papamarkos, A. Poulovassilis, and P. T. Wood. Event-condition-action rule languages for the semantic web. In *Proceedings of Workshop on Semantic Web and Databases (SWDB'03), Berlin, Germany*, Sept. 2003.
11. T. Payne, R. Singh, and K. Sycara. Facilitating message exchange through middle agents. In *The First International Joint Conference on Autonomous Agents and Multi-Agent Systems, Bologna, Italy*, July 2002.
12. T. Sollazzo, S. Handschuh, S. Staab, and M. Frank. Semantic web service architecture - evolving web service standards toward the semantic web. In *Proc. of the 15th International FLAIRS Conference, Pensacola, Florida*, May 2002.
13. W3C. Semantic web activity statement. available at `http://www.w3.org/2001/sw/Activity`, last accessed March 2005.