In-Young Ko, Robert Neches, and Ke-Thia Yao*:

A Semantic Model and Composition Mechanism for Active Document Collection Templates in Web-based Information Management Systems

Authors' affiliations: University of Southern California, Information Sciences Institute , 4676 Admiralty Way, Marina del Rey, CA 90292, U.S.A.

E-mail: {iko, rneches, kyao}@isi.edu

Abstract: *Representing semantic information embedded within documents is important, but not sufficient, for the Semantic Web vision of having machines automatically process data found on the Web. Many Web-based information management service tools, such as GeoWorlds [5,13], deal with collections of documents and the services that operate upon them. Semantic modeling of* ***document collections*** *and* ***Web services*** *is essential. We describe techniques for representing semantics of both collections and services, using a lightweight multi-form ontology. These techniques improve the efficiency and reusability of users' work with Web-based information management systems. They help users to set up complex analyses and structurings of information collections to adapt their work for other analyses or for different collections, and to obtain automatic refreshing and updating for collections with content that changes over time. Our semantic representation facilitates identifying and sequencing appropriate analysis and visualization services for a given task.* ***Templates*** *capture these sequences in terms of active semantic relations between document collections created and manipulated during those tasks. Templates can be dynamically modified and instantiated to generate document collections for similar tasks, or to refresh an information space with time-varying document membership. They can also be exchanged, allowing others to reapply them.*

* Effort sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreements F30602-00-2-0610 and F30602-00-2-0576. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the DARPA, the Air Force Research Laboratory, or the U.S. Government.

1 Introduction

When searching, analyzing, or structuring information from the Web, users deal with large document collections composed of multiple Web documents retrieved from various Web resources. Although some Web resources (e.g., Web directory services like Yahoo) return structured information such as categorized document collections, document collections retrieved from the Web are usually unorganized and too big to easily browse. Information analysis and visualization services are needed to operate upon the document collections in order to characterize, sort, partition and filter them. This paper explains how the semantic modeling and reasoning mechanisms at the *document collection* and *service* level can help improve the efficiency and reusability of users' work with Web-based information management systems.

The Semantic Web [3] is a vision of "having data on the Web defined and linked in a way that it can be used by machines not just for display purposes, but for automation, integration and reuse of data across various applications" [20]. Research on the Semantic Web has focused on representing machine-processable semantics of Web resources. RDF (Resource Description Framework) [18], SHOE [8] and Ontobroker [6] are among the efforts that provide semantic models, languages, and inference mechanisms for representing and processing Web document semantics. These approaches, however, are limited to individual documents. To make full use of document semantics to augment information management systems like GeoWorlds, it is essential to provide separate models and inference mechanisms for semantics of *document collections*. These are the units of information processing for information management services.

Furthermore, the notion of modeling data at the semantic level should be extended to the modeling of services. The Web has the potential of being the repository of not just documents, but of software services. In order for these services to interoperate, they must be described at the semantic level as well. When linking two services together, one must be able to determine at the semantic (and structural) level that the output of the first service is compatible with the second service. This service-centric view may be regarded as the flip side of the Semantic Web vision of "having *services* on the Web defined and linked in a way that it can be used by machines for automation, integration and reuse of services across different *data*." However, these data-centric and service-centric views are not necessarily mutually exclusive. In this paper we describe an ontology to semantically represent the document collections. Then, we use the same ontology to describe the input and output parameters of the services.

Based on the document collections' semantics (content types and organization structures), semantically interoperable (not just syntactically matched) analytic and visual services can be selected to perform context-sensitive information analyses. For example, although underlying data structures for all document collections might be the same, only document collections that have been clustered based on geographic location references can be plotted on a map.

Steps taken to organize document collections in an information space can be scripted by describing active semantic relations between document collections. For example, a document collection plotted on a map can be described as the result of plotting a document collection that contains "place-name-based document clusters." That, in turn, is the result of applying a geographic place-name extraction function to an unorganized document collection retrieved from the Web, which itself is the result of some set of actions on objects (e.g., string searches). We call these descriptions *active* because they describe how a document collection is generated from another. We consider them *semantic level* descriptions because they do not

include any syntactic details such as the data types within the document collections or the specific analysis functions used.

By providing mechanisms to represent and process the active semantic relations between document collections, we can support *active document collection templates*. The advantage is that these can be composed by users in advance without getting bogged down in the syntactic details. Reasoning about the semantics of available tools and data, the system can dynamically instantiate templates on behalf of a user, and execute them to generate completed document collections based on local resources. Not just specific results, but the general methods used to produce them, can therefore be exchanged among users.

Semantically-based service selection and active document collection templates improve the efficiency of information management systems and make it easier for users to develop large scale, task-oriented information spaces. These two mechanisms, in turn, rely upon the following techniques:

- *Explicit semantic representation* of document collections and services
- *Active document collection template composition mechanisms and tools* to define templates for information management tasks by describing active semantic relations between document collections
- *Semantic reasoning mechanisms* that match or retrieve semantically interoperable services for a document collection or another service by examining their semantic descriptions

The next section offers more detail about the target domain, Web-based information management systems. Sections following (from Section 3 to 5) focus in turn on the three enabling techniques. Section 6 describes the status of the current prototype implementation. Related work is reviewed in Section 7 and planned extensions are discussed in Section 8. The benefits and shortcomings are summarized in Section 9.

2 Web-based Information Management Systems

Web-based information management systems such as GeoWorlds [5,13] provide useful tools for retrieving, characterizing, sorting, partitioning, filtering and visualizing of topic-related Web documents, and offer an environment to help users create task-oriented information spaces from raw collections. Figure 1 illustrates the process of using GeoWorlds to organize a sample information space on "High-speed Internet Coverage Areas in the United States." An initial collection obtained by merging string searches is analyzed in various ways (e.g., grouping by frequently occurring phrases, plotting location references on maps) to help identify and populate a topic hierarchy that organizes the collection. Four major types of functions are illustrated: information gathering, information analysis, information visualization, and information organization.

Each type operates on a collection of documents and produces a collection. Using the *information gathering* functions, GeoWorlds users can extract relevant documents from sources such as Web search engines, Web directory services, on-line yellow pages, news video archive databases. They can get help characterizing the resulting initial document collection using a rich set of *information analysis* functions such as noun-phrase extraction, document clustering, category comparisons, language translation. *Information visualization* components applied to the analysis results help users make sense of those results and identify important parts of the document collections to assimilate into the users' information

collections. *Information organization* tools then help identify or impose structure on the results. These organized information spaces can be maintained in persistent storage. Fully organizing a body of information is an iterative process on collections and sub-collections, which repeats until the information space meets its users' needs.

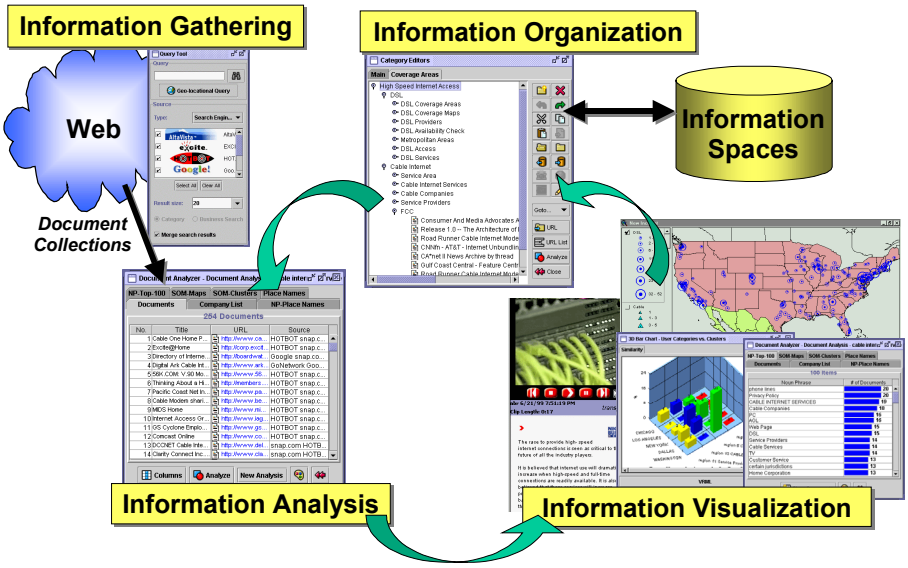

Figure 1: Overview of the information management cycle for organizing the information space "High-Speed Internet Coverage Areas in the US" by using GeoWorlds

The Web-based Information Management System is the target domain for our semantic modeling of document collections and services, and GeoWorlds is used as the test-bed to test the mechanism to compose active document collection templates by using the models developed. The following sections will explain details about the models and the template composition mechanism with giving examples that have been applied to the GeoWorlds system.

3 Explicit semantic representation

One of the critical issues for the users of an information management system is to select appropriate information management services among the various information gathering, analysis, visualization and organization services available in the system. The services must be selected and sequenced in a way in which users' information management tasks are optimized in terms of result quality, time and efforts. Therefore, an automated or semi-automated mechanism that helps users perform optimized information management tasks is necessary.

As the information management cycle is repeated, more and more semantics is imposed on a document collection. For example, if a document collection is formed by an information gathering service that retrieves documents from Web directory sites based on user's query, the document collection will embed its organization semantics as "Topic-based Categorization." When this collection is processed by an analytic service that classifies documents based on the place names in a geographical region the user selected, the resulting collection will have its organization semantics as "Place-name-based Classification." On these two different categorizations,

various category manipulation services such as cross-product and intersection functions can be applied to get more focused and specialized categorizations, and the resulting collection will have more complex organization semantics.

The "appropriateness" of a service for a document collection can be measured based on this semantic information imposed on a document collection, and utilization of this information is the key to provide the automated or semi-automated service selection and composition mechanism. For example, by recognizing the organizational semantics of a document collection that is the result of cross-product operation between the place-named-based and topic-based categorizations, a visualization service that plots multiple types of document clusters on a geographical map can be matched, which will be a hard problem if the system just considers syntactic information in matching services.

To utilize the semantic information, it must be explicitly represented based on a model. This section describes the models we developed for representing document collection and service semantics, and for providing a mechanism to combine multiple services together to support more complex information analysis and to record user's information management tasks.

3.1 Content and structure forms

We have adopted a lightweight multi-form ontology to present the semantics of document collections. Based on our experience with GeoWorlds, this lightweight representation is sufficient for most *current* Web-based information management services. Services that directly operate on document collections from the Web often do not make any semantic assumptions, since most of the documents currently found on the Web are unstructured text with no embedded semantic tags.

Services that operate on the outputs of other services often do make some assumptions, but they do not involve intricate logical reasoning. These document collection assumptions can be divided into two forms: *content* and *structure*. Reasoning based on subsumption is sufficient to determine service applicability to document collections.

Our division of the ontology into multiple forms is consistent with Sowa's observation that the same physical entity can be described by different forms to emphasize its content and physical structure [16]. The *content description* represents the contextual meaning of the collection (e.g., a document collection in which the documents are classified by the major noun phrases). The *structure description* characterizes the organization structure (e.g., a document collection organized in an acyclic graph structure). By dividing the semantic description about a document collection into these two types, the complexity of the semantic representation can be reduced and reasoning performance can be improved [17]. One more element in the document collection semantics is the *active relation* with other document collections, which describes the action of transforming a document collection from one semantics to another (e.g., an initial document collection that is a flat list of documents can be transformed into a hierarchically organized document collection by performing a document clustering service).

Domain-specific ontologies are used to discriminate and classify the document collection content types and organization structures, and service functionalities. Appendix A shows the ontology hierarchies of content types, organization structures, and service types in the current prototype system.

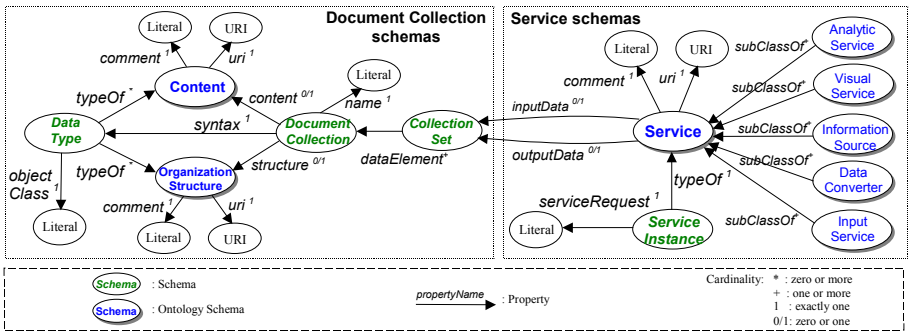

Figure 2: Schema model for representing document collection and service semantics

3.2 Document collection and service schema model

A schema model has been developed to represent the document collection and service semantics in terms of its content, organization structure, and active relations between other document collections. Figure 2 illustrates this schema model as an ER (Entity-Relationship) diagram. Content, Organization Structure, and Service are the top-level ontology schemas that describe the top concepts in the ontology hierarchies of instantiated schemas which we call nodes. Subsumption relations between nodes can be described by *subClassOf* property. In the current implementation, Analytic Service, Visual Service, Information Source, Data Converter, and Input Service are defined as the major service types that are subclasses of Service. The ontology schemas have a *comment* property that is the textual description about the schema and an *uri* property which value is the Universal Resource Identifier (URI) [2] that uniquely identifies the node within ontology hierarchies.

The semantics of a document collection can be represented by Document Collection schema that is composed of *content* and *structure* properties. Since a service may require multiple inputs and generate multiple outputs, the input or output semantics is specified by a set (Collection Set) of document collections. To uniquely identify an element within a set, a *name* property is assigned to each document collection in the set. Table 1 (a) illustrates this by showing the semantic description for the company-name-based document classification service type. This type of service accepts a document collection (regardless of its organization structure) and outputs a document collection that is categorized by the names of businesses found in the original set as well as a list of the company names that were extracted.

To enable classifying a concrete data type (syntax) that represents a document collection with certain content and structure semantics, a Data Type schema has been defined. A *typeOf* relation between a data type and a content or an organization structure indicates that the data type is a representation syntax for document collections that have such content or structure semantics. A data type can be classified under multiple content and/or organization structure ontologies, which means that the data type can be used to represent any document collection which has such semantics. For example, Hash Table data type can be used to represent both a categorized document collection and a document collection with an acyclic graph structure. Each data type description has an *objectClass* property that points to the language-specific object class name such as 'java.util.Hashtable' in Java language.

Specific implementations of services are represented and classified using Service Instance schema. A service instance has a *typeOf* relation to a service

ontology, which indicates that the instance implements the service type. The *serviceRequest* property of a service instance records how to invoke the service (such as a job request entry to the system interface). The I/O parameters of a service instance are described via the same mechanism as describing the I/O document-collection sets of a service ontology. However, each element (a document collection) in a parameter list has a *syntax* property which points to a specific data type, and the order of the elements reflects the order of the parameters in the actual service invocation interface. Table 1 (b) shows the description for the GeoWorlds' implementation of the company-name-based document classification service (specific syntax for I/O parameters have been selected and the output parameters have been reordered based on the service invocation interface).

Table 1 Example service descriptions: (a) the semantic description for the company-name-based document classification service; (b) an instance description of the company-name-based document classification service

(a)

Property	Value		
uri	http://www.isi.edu/geoworlds#CompanyNameBasedClassification		
comment	Services of classifying documents based on company names extracted from a document collection		
subClassOf	KeywordBasedClassification, CompanyNameExtraction		
	name	content	structure
inputData	input$1	DocumentCollection	
outputData	Output$1	CompanyNameBasedCategories	FlatCategoryList
	Output$2	CompanyNameList	ListOfDataItems

(b)

Property	Value			
uri	http://www.isi.edu/geoworlds#GW_CompanyNameExtractor			
comment	GeoWorlds' implementation of the company name extraction service			
typeOf	CompanyNameBasedClassification			
serviceRequest	CompanyNameExtractorJobRequestEntry			
	name	content	structure	syntax
inputData	input$1	DocumentCollection		DASHERCategory
OutputData	Output$2	CompanyNameList	ListOfDataItems	GWCompanyList
	Output$1	CompanyNameBasedCategories	FlatCategoryList	HashTable

As we mentioned earlier, we mainly focus on representing the collection-level semantics instead of individual document level semantics because document collections are the information processing units for Web-based information management services. In addition, we capture collection semantics at the level in which the semantic information can be used to describe the I/O semantics of the information management services. For example, a document summarization service does not require the actual context of individual documents in an input collection (such as the stories embedded in the documents), but the service does need to know information about the language used in the documents. Therefore, we may need to define a 'Spanish Document Collection' as a specific type of the general 'Document Collection' to use it for describing the I/O semantics of a 'Spanish Document Summarization Service'. However, we may not need to define more detailed context information such as "Collection of documents about High-speed Internet Coverage Areas in the US" because none of the information management services in the current GeoWorlds system requires such detailed semantics. We may need to define such detailed context information (more domain-specific ontologies) later as we add more types of services that require the information.

4 Active document collection templates composition mechanism

For some types of information analysis tasks, the information spaces resulting from a sequence of information management cycles might be time-dependent. For example, if the user performed the same information management steps several months apart, an information space on "High-speed Internet Coverage Areas in the US" would show more coverage areas on the map, a different coverage ratio between cable and DSL modems, and a different set of highly cited cable and DSL companies. Such time-sensitive information spaces need to be regenerated (refreshed) regularly to maintain the latest information.

Often, a sequence of analysis steps is repeated many times to organize an information space. For example, to get more detailed information about each high-speed Internet coverage area, the same set of information management steps (retrieve relevant documents from the Web, extract company names, classify the companies based on connection types, etc.) need to be repeated for each area.

Without scripting the information management steps, these information space refreshing and detailing processes will be inefficient and may generate inconsistent results. To meet this need, we provide mechanisms to compose and run *active document collection templates*. These are semantic-level scripts of information management steps. In an active document collection template, semantic requirements for document collections and management functions for organizing an information space are described. Also, the active relations between document collections can be described by specifying the analysis functions that transform a document collection into another with different content and/or structure semantics.

Once an active document collection template is composed, it can be modified and used to quickly generate information spaces for other similar tasks. Many information management tasks are similar in their major analysis steps. For example, the steps for organizing the information space on "High-speed Internet Coverage Areas in the US" can be reused to generate similar document collections for other countries. Only the initial query and the place name set need to be modified to include the geographical information for a different country.

Active document collection templates enable users to exchange their information management scripts with other users. The templates can be dynamically instantiated based on the locally available resources, and will organize the task-oriented information spaces locally.

The following sub-sections explain the details of mechanisms to compose, instantiate and execute active document collection templates.

4.1 Model for representing active document collection templates

The model for representing semantics of document collections and services has been extended to provide a model to represent active document collection templates and their instances. Figure 3 shows the ER diagram of this extended model.

When an information service is matched against a set of document collections, the collections within the set should be bound to the service as marshaled input parameters, so that the service can be performed by using the data at run-time. Also, when a service is combined with another service, the output document collections of the first service should be bound to the input parameters of the second service. *marshaledInput* and *marshaledOutput* properties of Service schema represent such data bindings between nodes. A set of document-collection instances can be represented by using Collection Set Instance schema, which elements are

Document Collection Instance's. An *analysisResultOf* property in a Collection Set Instance explicitly represents the input-output relationship between the set and another document-collection set via a service. Each Document Collection Instance has an *object* property which points to the physical object that keeps the content data of the document collection.

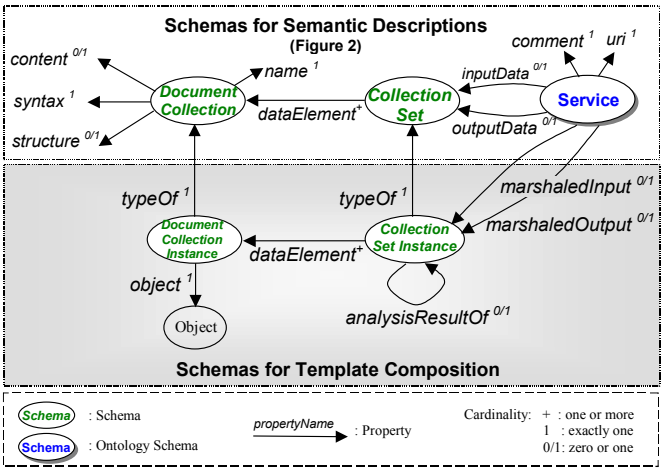

Figure 3: Schema model for representing active document collection templates

A document collection can be a member of multiple I/O data sets and multiple relations with other document collections can be represented. For example, consider a document collection composed of Spanish documents and a document collection categorized based on place names cited in the document contents. These document collections are the results of an English-to-Spanish translation service and a place name extraction service performed on an initial document collection.

Figure 4 shows an example of an active document collection template that organizes an information space on "High-speed Internet Coverage Areas in the US." Requirements on document collections and active relations between them are described in terms of the model shown in Figure 3.

Two instances of Web document retrieval services (f_1 and f_3) have been added to generate the two document collections (d_1 and d_3) that are related to DSL and cable modem coverage areas in the US, respectively. By using the place-name-based document classification services (f_4 and f_5), these flat document lists are transformed into categorized document collections (d_4 and d_5) in which documents are classified based on place names cited within their content. To classify the documents, each of the place-name-based document classification services uses a set of US metropolitan area information (d_2) which is provided by the geographic location extraction service (f_2) that extracts place names and location information (e.g., latitude and longitude) for a selected region on a map. The classified document collections are plotted on the map by the document mapping services (f_6 and f_8) to identify geographical locations which can be inferred to be the places where DSL and/or cable modem services are available. The two different categorizations are intersected by the category intersection service (f_7) that produces a document collection (d_6) that is reorganized based on the intersecting categories. These intersecting categories are visualized by a bar chart (f_9) that shows the number of documents for each intersecting place, which helps users identify the places where both of the high-speed Internet connection types are most likely available.

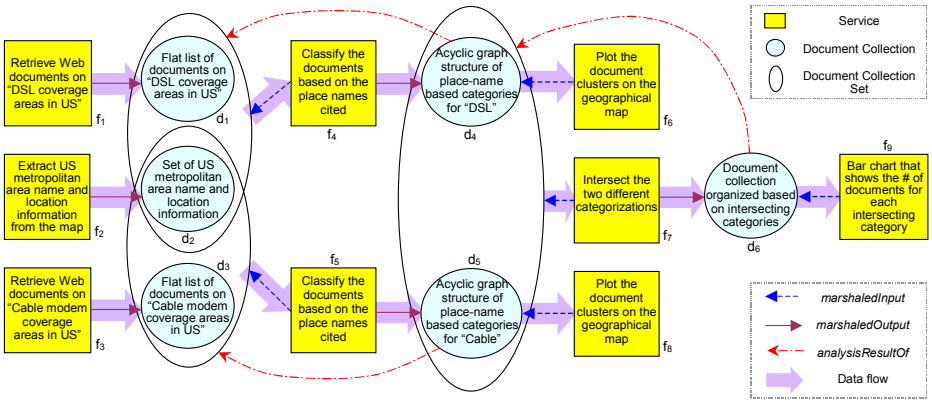

Figure 4: Active Document Collection Template for "High-speed Internet Coverage Areas in the US" [1]

Based on the active semantic relations between document collections, data flow between the services can be deduced (the shaded arrows in Figure 4 represent the data flow). For example, since the service f_4 requires d_1 and d_2 as its inputs, the data flows from the services f_1 and f_2 which are the producers of d_1 and d_2, respectively to the service f_4. Current implementation of the template composition tool allows users compose an active document collection template by specifying data flow between information management services. The detailed input/output relations between document-collection sets and services, and *resultOf* relations between document-collection sets are automatically generated by the composer based on the data flow specified by the user. This makes the template composition task easier for the users. Please see Section 6 for the implementation details and the data flow created for this example by using the template composer.

The graphic representation of an active document collection template, such as Figure 4, can be serialized (the current prototype generates XML data). This can be stored in a template repository or exchanged with other users.

4.2 Instantiation of an active document collection template

An active document collection template can be instantiated by allocating local resources to the nodes in the template. Thus, for example, two users might have different map viewer installed on their systems but could both still run the same template since it would be indifferent as to which map viewer was invoked. In each local system, semantic descriptions about the local resource instances are kept as metadata in a repository. As explained at Section 3, metadata about a resource instance also includes some syntactic information such as data types, job request entries, and I/O parameter ordering[2]. The semantic compatibility measurements that will be explained at Section 5.4 are performed to select semantically compatible local resources for each node in the template. The process of instantiating a template may require human interaction to resolve multiple matches of resource instances and syntactic mismatches between nodes[3].

---

[1] Semantic properties, *typeOf*'s of the services, and the document collections are omitted to reduce the complexity of the diagram.

[2] In the current prototype, Java class names are used to describe the internal data types of document collections. A job request entry is a directive to request for a service.

[3] The current prototype resolves some syntactic mismatches automatically by finding and inserting syntactic converters. See Section 6 for details.

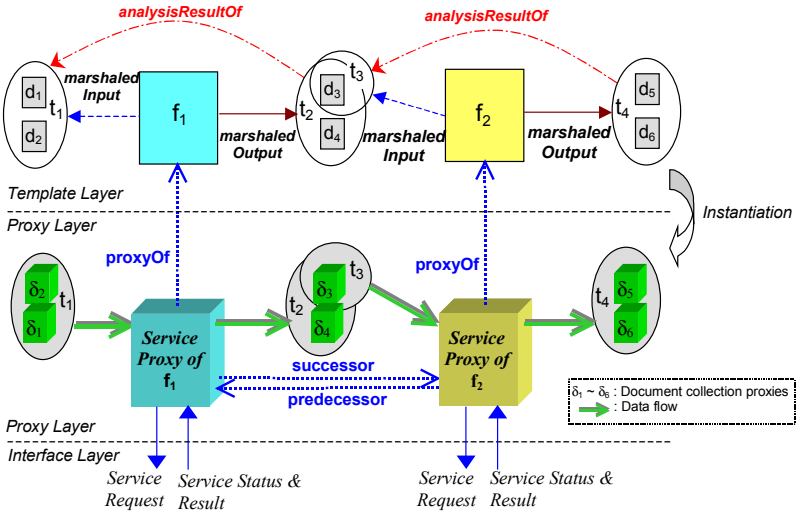

Figure 5: Instantiation of a template by creating proxies

As the result of an instantiation, proxies are created to act as clients to invoke the specific services that are selected to instantiate the template, and to receive the results of these service instances. Each *proxy* represents a resource instance (a document collection or a service) and keeps information for accessing the local resource. Figure 5 illustrates the proxies and relationships between them instantiated for a template. For each functional semantics, a service proxy is created and for each document-collection set, an ordered-list (ordered based on the I/O parameters of the corresponding service instance) of document collection proxies is created. A service proxy keeps precedence relationship between predecessor and successor proxies and a pointer (proxyOf) to the corresponding semantic description in the template. A document collection proxy maintains a pointer to a document collection object.

4.3 Template execution

The instantiated template can be executed by running the service proxies in a sequence governed by the precedence relations. An activated service proxy submits a service request to the system interface, monitors the job status, and receives the result. In the current prototype implementation, this service access mechanism is implemented based on the GeoWorlds' asynchronous service invocation architecture that is described in [17].

Multiple service proxies can be run in parallel and the proxies can be synchronized by using the precedence relations. When a proxy receives the result from its service instance, it updates the object pointer fields in its output document collection proxies to point to the result objects. Then, it invokes all the successor service proxies. A service proxy will not be activated unless it receives signals from all the predecessors. Usually the terminal services are visualization services that have no successor nodes.

4.4 Template update

Active document collection templates for complex information management tasks can be developed *incrementally* by repeating the composition, instantiation and execution steps. After/while executing a template, the user can modify the template by adding new nodes or removing unnecessary nodes or replacing certain nodes with other semantically compatible ones (to specialize, generalize or alter the

functionality). When a template node is modified, only the proxies that are affected are regenerated (usually from the modification point to the terminals). Therefore, when the template is re-executed, the unaffected proxies will not be rerun and all the intermediate data can be reused for the newly initiated services.

This incremental development of active document collection templates is especially effective when the information management task has to deal with large document collections and is composed of many analysis steps.

The next section describes how the semantics of document collections can be compared to select and combine semantically interoperable services. This is a key to composing an active document collection template.

5 Semantic reasoning mechanisms

A service can be applied to a document collection if the semantics of the document collection is equivalent to, or subsumed by, that service's input parameter semantics. For example, a category comparison service, which can compare two document collections that are categorized by a set of keywords, can also compare two document collections categorized by a set of place names because the place name-based categorization is a specialization of the keyword-based categorization.

The *semantic inclusion* relation is defined to compare such relationships between document collections or document-collection sets. This relation is the basis for matching services against document collections and measuring semantic interoperability and compatibility among services.

To explain the reasoning mechanism, the following formal representation is defined:

- Document collection semantics: $D = (c, s)$, where c is the content ontology and s is the structure ontology
- Semantics of a document-collection set: $T = \{ D_1, D_2, \ldots, D_n \}$, where $D_1=(c_1,s_1)$, $D_2=(c_2,s_2)$,…, $D_n=(c_n,s_n)$
- Functional semantics of a service: $F=(O,T_I,T_O)$, where O is the functional ontology, T_I is the semantics of the input data set, and T_O is the semantics of the output data set

5.1 Semantic inclusion relation

Semantic comparison between two document collections is done by comparing their content and structure semantics. Let $D_1=(c_1,s_1)$ and $D_2=(c_2,s_2)$ be the document collections to compare. If both the content and structure semantics of D_1 subsumes the content and structure semantics of D_2 (i.e., $c_1 \geq c_2$ and $s_1 \geq s_2$), we say that the document collection D_1 *semantically includes* the document collection D_2 and denote the relation by

$$D_1 \sqsupseteq D_2$$

The semantic inclusion relation is reflexive and transitive:

$$D \sqsupseteq D$$

$$D_1 \sqsupseteq D_2 \wedge D_2 \sqsupseteq D_3 \Rightarrow D_1 \sqsupseteq D_3$$

For example, document collections which are hierarchically categorized based on the major keywords semantically subsume document collections with a flat structure based on the company names, because company names are more specific than keywords and the flat category structure is a special case of the hierarchical category structure.

Many of the information management services accept and/or generate multiple document collections as their inputs and/or outputs. Therefore, a comparison mechanism between sets of multiple document collections is required. To support this, semantic inclusion relations between document-collection sets are measured by comparing each pair of related document collections from both sets. Let $T = \{ D_1, D_2, \ldots, D_n \}$ and $T' = \{ D_1', D_2', \ldots, D_m' \}$ be the two document-collection sets to compare. Then,

$$T \sqsupseteq T' \text{ iff } (\forall D_x \in T)\,(\exists D_y \in T')\; D_x \sqsupseteq D_y$$

If there exists any element in T that semantically includes multiple document collection semantics in T' (i.e., the semantic inclusion relation between T and T' is not a function), the semantic inclusion relation is *map ambiguous*.

For example, a set of two document collections, either of which is a keyword-based categorization semantically includes a set of three document collections each of which is a document classification based on a set of place names or company names or keywords because each collections in the first set semantically includes any of a document collection in the second set. However, it is map ambiguous because there are multiple mappings between the sets. In this case, human interaction is required to select a specific mapping.

5.2 Semantically-based service selection

Given a set of document collections with semantics T_d, a service F with input parameter semantics T_I is *selectable* if T_I semantically includes T_d (i.e., $T_I \sqsupseteq T_d$). If the semantic inclusion relation between T_I and T_d is map ambiguous (i.e., there exist multiple mappings between the data sets), human interaction is required to resolve the ambiguity by explicitly specifying the mapping between T_I and T_d.

For example, a category comparison service can be selected to process the set of three document collections with different categorizations because the service requires two categorized document collections (regardless of their categorization types) as inputs which semantically include the set of three document collections. Mapping ambiguity exists because any two document collections from the collection set can be mapped to the service inputs. Manual selection among possible mappings is required to resolve this.

5.3 Semantic interoperability

Semantic interoperability characterizes the requirements for two services in an active document collection template to be composable, $F' \bullet F$ (output of F is the input of F'). Let the functional semantics of F be (O, T_I, T_O) and the functional semantics of F' be (O', T_I', T_O'). Then,

F' is *semantically interoperable* with F iff $T_I' \sqsupseteq T_O$

i.e., if the output document-collection set of F can be accepted by F', F' can be combined with F.

For example, a noun-phrase extraction service which analyzes frequently cited

noun-phrases in a document collection, and returns a list of noun-phrases extracted from the documents and a document classification based on the noun-phrases can be combined with a document clustering service which accepts a list of keywords and a keyword-based document classification to analyze contextual similarity between documents, and to return clusters of similar documents. It is because the inputs (the keyword list and the keyword-based categorization) of the document clustering service semantically include the outputs (the noun-phrase list and the noun-phrase-based classification) of the noun-phrase extraction service.

5.4 Semantic compatibility

Semantic compatibility in an active document collection template characterizes the requirements for one service to replace another service.

Let $F=(O,T_I,T_O)$ be the functional semantics of the original service and $F'=(O',T_I',T_O')$ be the functional semantics of a replacement service. The most conservative way to check the semantic compatibility is to compare the I/O data semantics of the two functions as follows:

$$F' \text{ is } \textit{semantically compatible} \text{ with } F \text{ iff } T_I' \sqsupseteq T_I \wedge T_O \sqsupseteq T_O'$$

This rule ensures that the replacement function accepts all input data semantics accepted by the original function, and only generates output data semantics generated by the original function. For example, a place-name-based classification service in a template can be substituted by a geo-containment analysis service because input semantics of the two services are the same[4] (a general document collection), and the output (a hierarchical document classification based on geo-containment) of the geo-containment analysis service is more special than the output (a flat place-name-based document classification) of the place-name-based classification service.

However, by looking at the neighboring services (predecessors and successors) in the active document collection template we can derive a less restrictive *context-dependent semantic compatibility* rule. Suppose that T_x is the least upper bound (least general generalization) of the semantics that the predecessors of F can generate, and T_y is the greatest lower bound (most general specialization) of the semantics the successors of F can accept, and $F''=(O'',T_I'',T_O'')$ is a replacement service then

$$F'' \text{ is } \textit{context-dependent semantically compatible} \text{ with } F \text{ iff}$$
$$T_I'' \sqsupseteq T_x \wedge T_y \sqsupseteq T_O''$$

This implies that context-dependent semantic compatibility is less restrictive than the semantic compatibility, i.e., the set of services that accept T_I'' and generate T_O'' is a superset of the set of services that accept T_I' and generate T_O'. For example, if the service after the place-name-based classification service in the template is a general category viewer (such as a tree viewer) that can visualize a hierarchical category structure (which is less specific than the place-name-based categorization), any type of document classification services (such as the keyword-based classification service) that produce a categorized document collection can be selected to substitute

[4] The place-name-based classification service requires a place name list and the geo-containment analysis service requires a geo-containment hierarchy as an additional input. However, these inputs are not considered in this compatibility measure because the data can be provided by input services that can be inserted to the template without affecting other services.

the place-name-based classification service.

The act of replacing a service in a template with another compatible service can be classified into three cases, based on the relationship between the functional ontologies of the existing and the replacement services:

Generalization:
F' is a *generalization* of F if F' is semantically compatible with F and O' ≥ O.

Specialization:
F' is a *specialization* of F if F' is semantically compatible with F and O ≥ O'.

Alteration:
F' is an *alteration* of F if F' is semantically compatible with F and there is no subsumption relations between O and O'. A service with the functional semantics F can be replaced by a service with function F' to perform an alternative functionality with preserving the data flow semantics in the template.

Using these relationships, we can build a system that helps in a number of ways. It can create and serialize templates or scripts. It can automatically select services and collections to instantiate a template, and can even make some repairs and substitutions to overcome some mismatches.

6 Prototype

A prototype system that embodies the capabilities outlined above has been implemented in Java. Persistence of resource descriptions and active document collection templates is provided via XML serialization. The major components in the prototype are the *inference engine*, *metadata editor*, and *active document collection template composer*.

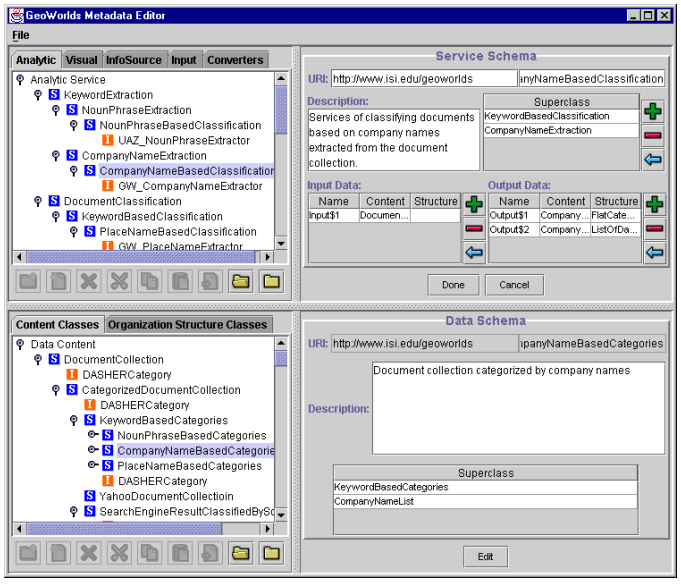

Figure 6: Metadata Editor

The inference engine implements the mechanisms required to retrieve, compare, select and substitute among collections and among services, along the formal lines described in Section 5. It also provides consistency-checking functions by which the

system can ensure that only valid semantic descriptions can be added and appropriate resource instances can be classified.

When the inference engine is generating or testing candidates for use in a template, it can bridge across some semantic mismatches by adding additional nodes. For example, it can add a user input service if a document collection is required by a service but is not available at the stage of editing the template[5]. Similarly, a structure converter (if available) can be inserted between services to convert between data formats when they are matched in content semantics but not matched in structure semantics.

The metadata editor is the system's GUI for registering and modifying semantic descriptions of document collections and services. By using this tool, local resource instances can be classified under the ontology hierarchies and their syntactic descriptions can be edited. Figure 6 shows a screen shot of the metadata editor. The upper part of the window is for editing service nodes (analytic, visual, information source, user input, and data conversion services). The lower part is for editing content and structure semantics of document collections. In both cases, the left side displays the ontology hierarchy and the right side provides forms to edit node property values[6].

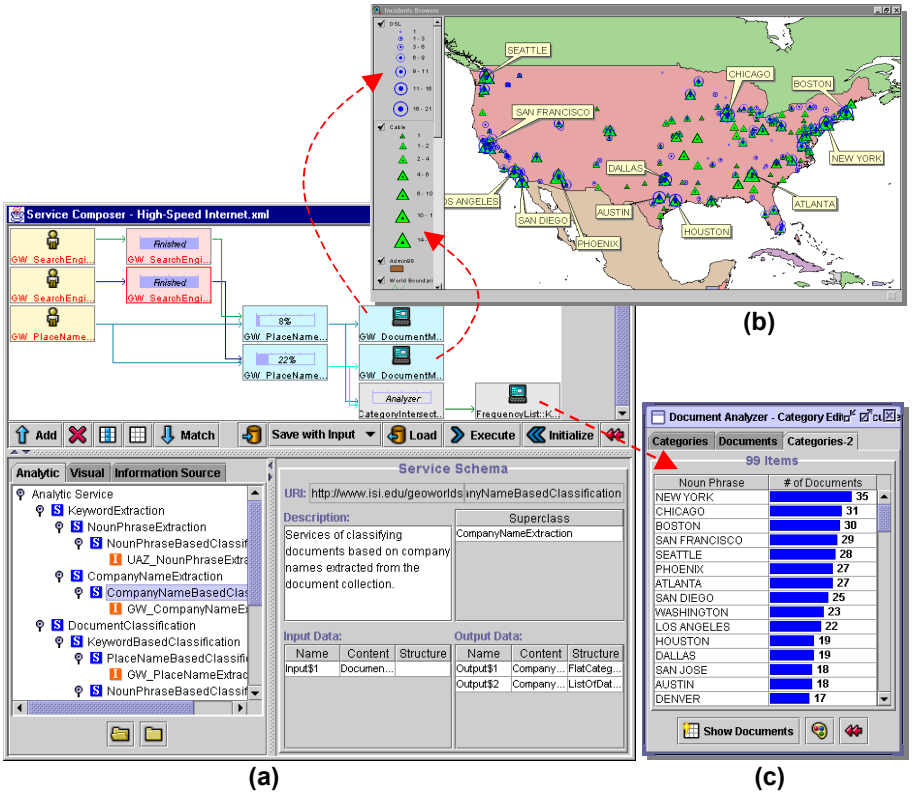

Figure 7: (a) Active Document Collection Template Composer; (b), (c) template execution results

---

5 Examples of the data that can be provided by user input services: a query string, a set of place names to be extracted from a document collection.

6 In the example shown, the upper-left part currently shows the ontology hierarchy of analytic services. The upper-right part shows the node properties for one of these services: "Company Name Based Classification." The lower part displays the outputs currently being edited in that service description, in this case, the ontology hierarchy for the document collection contents, and the node properties for "Company Name Based Category."

The active document collection template composer provides GUI-based tools to help users set up analysis and structuring activities by composing, instantiating and executing templates. Figure 7 (a) shows the template composer window. The upper part displays the current template including services and connections between them. Given a selection of nodes in the template, the lower part shows the partial ontology hierarchies that are semantically compatible with the selected nodes. This lets users see what options are available to them for adding steps to their analysis.

As described in Section 4, the composer creates proxies and proxy connections (a directed acyclic graph of control and data flow between proxies) when a template is instantiated. This enables the system to invoke, monitor and synchronize the services. Also, with the help of the inference engine, it automatically finds and inserts syntactic converters (e.g., data type converters) between syntactically mismatched proxies if appropriate converters are available. When a template is executed, each proxy node in the graph displays the status (progress bar and messages) of the service.

The example template shown in Figure 7 (a) is the script to organize an information space on "High-speed Internet Coverage Areas in the US" which is explained at Section 4.1. The active document collection template is represented in terms of a data flow diagram that shows data flow connections between information management services. All the properties to describe the active semantic relations between document collections are automatically constructed and internally kept by the composer. As a result of executing the template, a map (Figure 7 (b)) is displayed, which plots geographical locations where document clusters on the DSL and cable modem coverage areas are mapped[7]. Figure 7 (c) is the visualization of the category intersection result, which shows a sorted list of the places in which both categories (DSL and cable modem) of documents are mapped. The frequency bar displayed for each place name is the total number of documents that cite the place name within their content. A place with a longer frequency bar means that the place is more likely to be a place where both of the high-speed Internet connection types are available.

This template can be modified and rerun by using the editing features in the composer to refresh the information space or to organize high-speed Internet coverage information for other countries. The template can be serialized in XML and stored in the template repository, and reloaded later for a reuse by using the persistency functions (save and load buttons) in the composer. Users can also exchange the XML template data with other users who are working on the similar information management tasks to collaboratively populate or refine the information space.

7 Related work

As networked resources such as the Web have become available, digital library collections increasingly need to be specified by description in the form of *criteria* for selecting resources or *tools* for resource discovery, rather than being defined by enumeration in physical document collections [11]. Our active document collection templates are an implementation mechanism for such descriptively specified, Web-based digital library collections. The semantic descriptions of document collections and the active relations between them are the criteria and tools to organize and manipulate document collections for information management tasks.

[7] Size of a circle or triangle plotted on the map reflects the size of the document cluster mapped to the geographical location, i.e., the number of documents referencing that location.

Sheth pointed out in [15] that the focus of information system interoperability research is changing from system/syntactic level to semantic level. As heterogeneity of digital data, operations and computations is increased, users' demands upon information systems are shifting from the mere data level to information and knowledge levels. Sheth's observation is consistent with our experience with GeoWorlds. As more information types are available from the Web and as we add more analytic and visual services to the system, users want to specify their information management requirements at a high level (instead of spending their energy figuring out which services can be applied to a certain type of document collection). Our semantically-based service selection and the active document collection template composition mechanisms were developed to satisfy this need.

A significant discussion of issues of semantic interoperability in large object systems appears in [9]. They argue that explicit representation and run-time manipulation of semantic information can reduce the time and effort to build large software systems composed of COTS and legacy components. Our semantic representation scheme and active document collection templates extend this into the domain of information management, by suggesting that semantically-based mechanisms enable efficient and rapid organization of large-scale, task-oriented information spaces.

The approach of using explicit semantic information for integrating heterogeneous information sources have been used by various information mediation projects such as SIMS [1], InfoSleuth™ [14], and GINF [12], which characterize the information sources by extracting the semantic information (domain ontologies and relationships between ontologies) and expressing it using high-level representation languages. Semantics-based query processing and context-sensitive provision of analysis functions are closely related to our active document collection templates and the semantically-based service selection mechanism. However, their queries are for retrieving initial document collections and cannot specify the full information management cycles. Also, their analysis functions are mostly limited to information integration functions that are tightly bound with particular information sources.

There are similarities between our semantic inclusion relation and the semantic proximity measurement in the Semantics-Based Information Brokering system [10]. The semantic proximity represents semantic similarities between database objects based on their context. Similar to the semantic description of a document collection (or a document-collection set), the context of a database object is represented by referring to pre-existing ontologies. Also, the semantic similarity between database objects is measured by comparing corresponding ontologies.

Web Services Description Language (WSDL) [4] is a proposal submitted to the W3C for using XML to describe network services as collections of communication endpoint (*ports*) capable of exchanging *messages*. Many elements of our approach and WSDL are comparable. The concept of a WSDL *operation*, consisting of an input message and an output message, corresponds to our concept of a service with *inputData* and *outputData* collection sets (see Figure 2). The *types* used to define the WSDL messages corresponds our document collection semantics, which we further refine to content and organization structure semantics. Although the WSDL proposal prefers XSD (XML Schema) type system, it allows room for other type systems. We see our semantic reasoning mechanisms (see Section 5) as a potential alternative typing mechanism that directly supports document collections. WSDL allows services to explicitly state *bindings*, such as SOAP, HTTP or MIME, to attach specific protocols and data formats to the messages, operations and endpoints. Currently, in our system we implicitly bind to JAVA using Jini/JavaSpaces entries. One area of our work that is missing from WSDL is our conception of templates that provides a composition mechanism to combine individual services to form higher-level services (see Section 4).

8 Future work

The current template model has some limitations on its representational power (e.g., looping and dynamic behaviors between services cannot be specified). We are investigating information management cases and analysis functions that require more representation power. We plan to enhance the model based on that investigation's results.

By applying *semantic distance* measurements such as [7] to the semantic inclusion decision, some cases involving semantic ambiguity and multiple matching services can be resolved automatically. Instead of a binary decision, our inference engine can measure the semantic distance between document collections or between document-collection sets, i.e., measure how much a component semantically includes another one. Based on this distance measure, the system can select the most semantically close components.

USC ISI's TBASSCO (Template-Based Assurance of Semantic interoperability in Software COmposition) project addresses concerns about quality in adaptive composition of component-based software. The semantic measurement and the active document collection template composition mechanisms described in this paper are being generalized and extended for the TBASSCO semantic gauges. We are developing semantically-based software gauges that measure the level of semantic interoperability between components. These help system engineers evaluate components' functional and data equivalence compatibility, find pertinent data conversion mappings, and predict performance (time, space, network) of a component architecture.

We are currently developing an automatic template generator that returns active document collection templates based on users' high-level specification. It will generate all possible scripts that can achieve a user's information analysis goal. It will allow users to select and modify the template(s) to be instantiated for their tasks. We believe this will make their information management tasks go much easier and faster.

Persistence both of semantic description about document collections and services, and of active document collection templates, is being implemented using RDF (Resource Description Framework) [18], and RDF Schema [19]. Our schema model for representing semantics maps directly to RDF's graph-based model, and our active document collection templates can be represented in a more structured way using RDF syntax. RDF will also enable remote schema referencing, an attractive alternative to copying entire domain-specific ontology descriptions when active document collection templates are exchanged between users.

9 Conclusion

Our goal is to fully utilize semantic information about collections retrieved from Web resources and to support large-scale, task-oriented information management systems. To this end, we have developed an initial set of semantic representation and processing mechanisms for document collections. We have complemented this with mechanisms to compose and run active document collection templates. These facilitate scripting complex information management steps at a semantic level. Our system is able to instantiate them dynamically and repeatedly based on locally available resources.

Our semantically-based component description and selection mechanism

facilitates component-based information management software architectures. It provides an infrastructure in which data and service components can be added dynamically without requiring major changes to the system or to existing components. It improves software component reusability and enhances flexibility in choosing and using information management components. Complexity of component specification and reasoning is reduced by dividing the document collection semantics into two independent types: content and organization structure.

We believe the active document collection template composition and execution mechanisms serve to improve information management performance. By using the templates, expired document collections can be quickly refreshed. Development of new information analysis plans can be speeded up by reusing and modifying existing templates. Also, by exchanging relevant information management templates, users can collaborate with each other on organizing task-oriented information spaces.

The Semantic Web will make task-oriented information management systems more effective by providing infrastructure and tools to retrieve and manipulate more accurate and rich document semantics. Web-based information management systems that provide ways to manipulate *document collection semantics* and *active relations* between document collections, will make the Semantic Web much more usable and productive for its users.

Information and questions

For more information about GeoWorlds and TBASSCO projects:

http://www.isi.edu/geoworlds
http://www.isi.edu/tbassco

References

1. Yigal Arens, Craig A. Knoblock and Chun-Nan Hsu. Query Processing in the SIMS Information Mediator, Advanced Planning Technology, editor, Austin Tate, AAAI Press, Menlo Park, CA, 1996.
2. Tim Berners-Lee. Universal Resource Identifiers in WWW: A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web. RFC 1630, CERN, June 1994.
3. Tim Berners-Lee. Semantic Web Roadmap. World Wide Web Consortium (W3C), 1998. http://www.w3.org/DesignIssues/Semantic.html
4. Erik Christensen, Francisco Curbera, Greg Meredith, and Sanjiva Weerawarana. Web Services Description Language (WSDL) 1.1., March 2001. http://www.w3.org/TR/2001/NOTE-wsdl-20010315
5. Murilo Coutinho, Robert Neches, Alejandro Bugacov, Ke-Thia Yao, Vished Kumar, In-Young Ko, Ragy Eleish, and Sameer Abhinkar. GeoWorlds: A Geographically Based Information System for Situation Understanding and Management. In Proceedings of the First International Workshop on TeleGeoProcessing (TeleGeo '99), Lyon, France, May 1999.
6. Stefan Decker, Michael Erdmann, Dieter Fensel, and Rudi Studer. Ontobroker: Ontology Based Access to Distributed and Semi-Structured Information. In R. Meersman et al. (eds.), Semantic Issues in Multimedia Systems, Proceedings of DS-8, pp. 351-369, Kluwer Academic Publisher, Boston, 1999.

7. Harry S. Delugach. An Exploration Into Semantic Distance. Lecture notes in artificial intelligence, No.754, pp. 119-124, Springer-Verlag, Berlin, 1993.

8. Jeff Heflin, James Hendler, and Sean Luke. SHOE: A Knowledge Representation Language for Internet Applications. Technical Report CS-TR-4078 (UMIACS TR-99-71). 1999.

9. Sandra Heiler, Renée J. Miller, and Vincent Ventrone. Using Metadata to Address Problems of Semantic Interoperability in Large Object Systems. First IEEE Metadata Conference, Silver Spring, Maryland, April, 1996.

10. Vipul Kashyap and Amit Sheth. Semantics-Based Information Brokering. In Proceedings of the Third International Conference on Information and Knowledge Management (CIKM), Gaithersburg, MD, November, 1994.

11. Carl Lagoze and David Fielding. Defining Collections in Distributed Digital Libraries. D-Lib Magazine, November 1998, ISSN 1082-9873. http://www.dlib.org/dlib/november98/lagoze/11lagoze.html

12. Sergey Melnik *et al.* Generic Interoperability Framework, Working Paper, Department of Computer Science, Stanford University. http://www-diglib.stanford.edu/diglib/ginf/WD/ginf-overview/

13. Robert Neches, Sameer Abhinkar, Fangqi Hu, Ragy Eleish, In-Young Ko, Ke-Thia Yao, Quan Zhu, and Peter Will. Collaborative Information Space Analysis Tools. D-Lib Magazine, October 1998, ISSN 1082-9873. http://www.dlib.org/dlib/october98/dasher/10dasher.html

14. Marian Nodine, William Bohrer, Anne Hee Hiong Ngu. Semantic Brokering over Dynamic Heterogeneous Data Sources in InfoSleuth™. 15th International Conference on Data Engineering, March, 1999, Sydney, Australia.

15. Amit P. Sheth. Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics, *Interoperating Geographic Information Systems*. M. F. Goodchild, M. J. Egenhofer, R. Fegeas, and C. A. Kottman (eds.), Kluwer, 1998.

16. John F. Sowa, *Knowledge Representation: Logical, Philosophical, and Computational Foundations*, Brooks Cole Publishing Co., Pacific Grove, CA, ©2000.

17. Ke-Thia Yao, In-Young Ko, Ragy Eleish, and Robert Neches. Asynchronous Information Space Analysis Architecture Using Content and Structure Based Service Brokering. In Proceedings of Fifth ACM Conference on Digital Libraries (DL 2000), San Antonio, Texas, June 2000.

18. Resource Description Framework (RDF) Model and Syntax. World Wide Web Consortium (W3C) Recommendation, February 22, 1999. http://www.w3.org/TR/REC-rdf-syntax/

19. Resource Description Framework (RDF) Schema Specification 1.0. World Wide Web Consortium (W3C) Candidate Recommendation, March 27, 2000. http://www.w3.org/TR/rdf-schema/

20. W3C Semantic Web Activity Statement. World Wide Web Consortium (W3C), February 9, 2001.

Appendix A: Ontology hierarchies of document collection semantics (content and structure types), and service types for the GeoWorlds information management system