A Web based Conversational Case-Based Recommender System for Ontology aided Metadata Discovery

Mehmet S. Aktas, Marlon Pierce, Geoffrey C. Fox
Community Grids Labs
Indiana University
Bloomington, IN 47404, U.S.A.
{maktas, mpierce, gcf}@cs.indiana.edu

David Leake
Computer Science Department
Indiana University
Bloomington, IN 47405, U.S.A.
leake@cs.indiana.edu

Abstract

Locating resources of interest in a large resource-intensive environment is a challenging problem. In this paper we present research on addressing this problem through the development of a recommender system to aid in metadata discovery. Our recommender approach uses Conversational Case-Based Reasoning (CCBR), with semantic web markup languages providing a standard form for case representation. We present our initial efforts in designing and developing ontologies for an Earthquake Simulation Grid, to use these to guide case retrieval, discuss how these are exploited in a prototype application, and identify future steps for this approach.

1 Introduction

The availability of large-scale grid resources makes *resource discovery* a challenging problem. A promising method for alleviating this problem is to develop systems that can support the resource discovery task. This paper proposes the use of *Conversational Case-Based Reasoning* [1], a recommender methodology, to develop retrieval tools, and examines the value of representing cases with semantic web technologies. The work is conducted in the context of an important practical problem, the needs of the Solid Earth Research Virtual Observatory Grid (SERVOGrid) Project [2, 3, 4]. The CCBR retrieval mechanism presented here provides recommendations in metadata discovery on a virtual metadata layer of SERVOGrid Resources.

Resource discovery is the problem of locating resources of interest in large-scale resource-intensive environments. Because of the complex description of grid computing resources and the expected size of the resource set, an efficient and easy-to-use retrieval mechanism is needed to make resource discovery feasible for general users. Consequently, there is a need for not only a retrieval mechanism, but also for a *recommendation system* to suggest resources of interest when the resources may be too difficult to locate with traditional retrieval systems.

Numerous AI methodologies could be considered as the basis for a resource recommender system. For the SERVOGrid environment, however, the problem characteristics restrict the methods that may apply. Because of the volatility of SERVOGrid resources, there is no strong domain theory implicit in its metadata repository; it is hard to make generalizations based on given set of metadata instances. In addition, the SERVOGrid domain is not a domain where the experts can specify set of rules that fully cover the range of possible recommendations that could be made. Because resources can join or leave the system any time, it would be impossible to specify general rules which would be stable for SERVO resources. Consequently, a “lazy learning” method such as *Case-Based Reasoning (CBR)* [5], in which recommendations are made on an as-needed basis by doing reasoning from the current set of cases, appears most suitable for SERVOGrid domain. In CBR, when a similar situation (problem description) is entered, the most similar cases (metadata about resources) are suggested to the user as results.

Conversational Case-Based Reasoning is a type of CBR that relies on question-answer sessions to recommend most similar cases. Due to the complexity of SERVOGrid cases, users may not be able to formulate all the considerations relevant to their resource choices in advance; it is necessary to guide the user at each step of the retrieval. In CCBR, the user interacts with the system to fill in the gaps to retrieve the right cases and the system responds with ranked cases and questions to distinguish them. The question-answer-ranking cycle continues until success, if the user finds an answer to his/her query, or failure, if no satisfactory case is found.

Given the choice of case-based reasoning for resource recommendation, an important question is how to represent cases. The development of representational standards provides advantages such as interoperability and modularity: interoperability because the content of CBR cases could be used by non CBR applications, and provides modularity because CBR cases could be manipulated independent from CBR tool processing the case data. The Semantic Web [8] attempts to define the metadata information model for the WWW to aid in information retrieval and aggregation, and provides general languages for describing any metadata, in addition to advanced capabilities intended to enable knowledge representation and limited machine reasoning. To achieve a standard representation, we adopt Semantic Web languages such as RDF [9] as the representation syntax of metadata, enabling RDF representation of CBR cases to provide a standard means of representation.

Cases for SERVOGrid environment are simply descriptions of SERVOGrid objects such as earthquake simulation codes and data. CBR cases may reflect only some aspects of the knowledge describing SERVOGrid objects. To this end, CBR case data could be considered as a portion of the knowledge (metadata) about a SERVOGrid object. A CBR case data could simply be taken from RDF metadata triples. For our purposes, it is important to represent CBR cases in an RDF format so that it could be integrated with SERVOGrid metadata representation and we could easily extract relevant case data form SERVOGrid metadata.

This paper introduces our initial effort in representing the CBR cases by using Semantic Web markup languages such as RDF. We design and develop ontologies for an Earthquake Simulation Grid such as SERVOGrid project. We implement a web based Conversational CBR application by utilizing a domain independent CBR engine such as Indiana University Case-Based Reasoning Framework (IUCBRF) [10]. This application is to be used by SERVOGrid scientists seeking for resources (codes, data) to solve a science problem. We use RDF as case representation language and integrate case data within the metadata of SERVOGrid resources. In our application, we provide a retrieval mechanism which interacts with user and where the system responds with ranked cases and ranked questions to distinguish them.

The paper begins with an overview of the SERVOGrid project. It next introduces our effort to design and develop an ontology for the SERVOGrid environment, and describes out implementation of a CCBR retrieval mechanism utilizing instances of an RDF ontology. We then summarize the status of the system and discuss differences and similarities with related work.

2 SERVOGrid

SERVOGrid integrates historical, measured, and calculated earthquake data with simulation codes [2, 3, 4]. SERVOGrid resources are located at various institutions across the country. The primary information managed in the SERVOGrid domain is metadata about data, such as General Positioning System (GPS), Fault and Seismicity data, and Earthquake Modeling codes, such as application, visualization and simulation codes. Figure 1 shows the architecture of SERVOGrid environment.

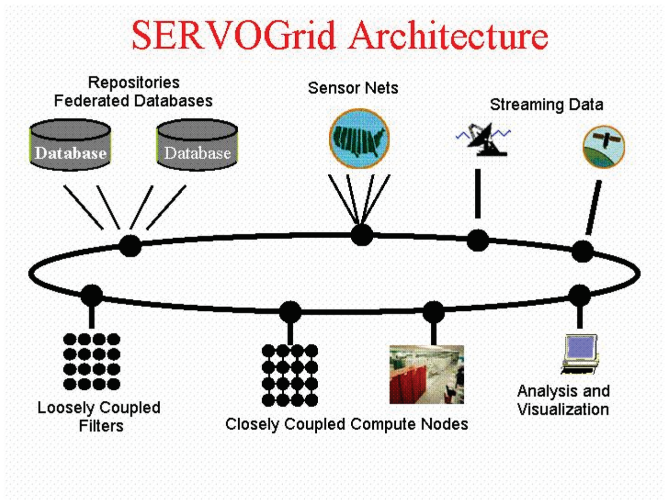


Figure 1: The architecture of SERVOGrid, a large scale grid project integrating earthquake data with simulation codes.

SERVOGrid is a rapidly evolving project. The number of resources, services and their usage frequencies are anticipated to grow quickly. Consequently, there is an emerging need to provide effective ways of locating and accessing desired resources for SERVOGrid environment.

3 SERVOGrid Ontology Design and Development

The SERVOGrid project contains a collection of codes, visualization tools, computing resources, and data sets distributed across the grids, for which we have developed a well-defined ontology using RDF. Here we summarize the development of this ontology; a detailed version of our effort in designing this ontology with pictorial figures is available in the workshop paper [11]. After having built such instances of the ontology, one can pose queries on the ontology instances. This is the basis for our CCBR retrieval system querying ontology instances and recommending users ranked results and questions to distinguish them.

3.1 Defining SERVOGrid Ontology Classes

When designing an ontology, we first need to group together related resources. In SERVOGrid project, there are three major groups of resources. These groups are ServoCodes, ServoData, and ComputingPlatforms. Table 1 shows the high level classification of classes to group together SERVOGrid resources as well as things that are related with these resources.

Table 1: High level classification of classes in SERVOGrid Ontology

Classes	Descriptions
ServoObject	describes SERVOGrid code and SERVOGrid data
ServoDataFormat	describes the types of different data formats used in SERVOGrid project
ServoCompute-Platform	describes the computing platforms exist in SERVOGrid project
ServoCode-Characteristics	describes the characteristics of the ServoCode
ServoObject-Container	describes the various types of containers for code, data and documents
Organization	describes the organizations that are involved in SERVOGrid project
Person	describes the people working in SERVOGrid project
Location	describes the location of the organization involved in SERVOGrid project

The first group of resources ”ServoObject” combines SERVOGrid codes and SERVOGrid data. SERVOGrid codes differ from each other according to their purposes. We defined three different classes for SERVOGrid codes. These three subclasses are simulation codes, visualization codes, and application codes. Simulation codes simulate interacting fault systems with real or syntactically created data using different earthquake models. Examples of these codes are mesh generator, VC (Virtual California), simplex and disloc [2, 4]. We associate simulation codes with zero or more visualization systems. The codes, written for those systems, are referred as visualization codes. Some examples of visualization codes are RIVA and GMT [2, 4]. Application codes are the codes used to facilitate earthquake science, e.g., a statistical analysis program. Likewise further classification can be done for SERVOGrid data. At this classification level we do not intend to represent the data itself, however, the information about the data, such as creator of the data, needs to be represented. To this end, data can be regrouped as GPS, Fault, and Seismicity data.

The second group of resources in the class hierarchy is ”ServoDataFormat” to represent different data formats. Sample instances of this class could be the Swath, Grid and Point data type. For lack of space, these are not discussed here, but preliminary schemas are available from [12].

The third group of resources is computing platforms. These resources can be further classified under ”ComputeResources” and ”InstalledWebServices” subclasses. The descriptions of these subclasses can be seen in Table 2.

Table 2: Classification of ServoComputePlatform

Classes	Descriptions
ComputeResources	describes the computers providing computational environments for the SERVOGrid project
InstalledWebServices	describes the web services providing message oriented computing in SERVOGrid environment

The fourth group of resources is ”ServoCodeCharacteristics”. We classified these characteristics of the SERVOGrid codes as described in the following Table 3.

Table 3: Classification of ServoCode Characteristics

Classes	Descriptions
KnownProblemsInCompiling	describes the general problems encountered while compiling ServoCodes
KnownSuccessfulLibrary	describes the successful libraries that have been tested and used with ServoCodes
UsedModel	describes various earth science models used in ServoCodes
UsedProgrammingLanguage	describes used programming languages

In addition to the classes explained above, we have also defined small ontologies of organizations, people and locations involved in SERVOGrid project. Because the focus of this paper is on SERVOGrid resources, we will not discuss these ontologies here.

3.2 Defining SERVOGrid Ontology Properties

In order to relate ontology classes to each other, we defined our own meaningful properties for SERVOGrid ontology. We benefit from existing ontologies such as Dublin Core [14] and vCard [15] as well. The ServoCode class defines seven different properties in addition to the Dublin

Core definition standard elements. Likewise, ServoData and ServoComputePlatform classes define their own properties. The properties for ServoCode, ServoData and ServoComputePlatform and their meanings are described in Table 4, Table 5,and Table 6 respectively.

Table 4: Properties associated with the ServoCode Class

Properties	Range	Descriptions
isDevelopedWith	UsedProgrammingLanguage	what programming language is used
createsOutputData/ takesInputData	ServoData	what type of data is used for input/output
dependsUpon	ServoCode	what code does it depend before it can be completed
installedOn	ServoComputePlatform	where is the code installed
developedBy	Person, Organization	who developed the code
isOwnedBy	Person, Organization	who owns the resource

Table 5: Associated properties with ServoData Class

Properties	Range	Descriptions
hasDataFormatOf	ServoDataFormat	what is the associated data format
isInputDataFor/ isOutputDataFor	ServoCode	what code is taking this data as input/output

3.3 Generating the Ontology instances with SW Languages

Following the definitions described in the previous sections, we defined a class hierarchy associated with meaningful properties. After designing the ontology, we wrote the description of these classes and the properties in RDF semantic markup language. The SERVOGrid ontology and ontology instances are available from [12, 13]. We will now discuss the details of providing a Conversational CBR recommender system to retrieve the requested metadata satisfying a user query. In the following section, we will assume that our application has an access to all metadata pieces, as if they are stored in a local file store, and will discuss how resources of interest can be sought by using our CCBR retrieval system processing SERVOGrid ontology metadata instances.

Table 6: Associated properties with ServoComputePlatform Class

Properties	Range	Descriptions
isOwnedBy	Person, Organization	who owns the resource
hasData /hasCode	ServoData /ServoCode	what ServoData/ServoCode is available in this compute platform
isMaintainedBy	Person	who maintains this compute platform

4 A Web based CCBR Recommender System for Ontology aided Metadata Discovery

We have implemented a prototype web-based CCBR system [16] to aid in metadata discovery for the SERVOGrid project. This system is to be used by SERVOGrid users (scientists) who are looking for resources (codes, data) to solve a science problem. This application uses the previously-described ontology-aided semantic metadata representation in RDF, asking the user questions about desired resource characteristics and responding to queries with ranked cases. At each step, the system also provides a ranked set of questions, which the user may answer to further distinguish the proposed cases [17, 18]. This process narrows down results and incrementally eliminates cases which prove to be irrelevant as the user's needs are elaborated.

Our application has been developed using IUCBRF [10] as a domain-independent CBR Engine. IUCBRF is an open source framework written in Java. It provides standard implementations of necessary components (such as domain definition, case construction, retrieval and so on) to develop CBR/CCBR applications. Because IUCBRF is domain-independent, and the domain-specific information for the system is captured entirely in the RDF ontology and ontology instances, the developed system could be easily transferred to other domains as well.

In the CCBR application, problems are described by metadata concerning desired characteristics of a SERVOGrid resource, and the solution to the problem is a pointer to a resource described by metadata. Cases contain both a problem description—used for similarity assessment—and solution pointers. Our prototype focused only on discovery of earthquake modeling codes and data. However, the prototype can easily be extended to other resources of SERVOGrid project. The library of cases—the "case base"—is initially generated from a file store where each case is represented with RDF syntax. For case storage, we currently use the "flat case base" general indexing scheme implemented in IUCBRF, which stores cases in an unordered list. This is sufficient for our proof-of-concept demonstration, but would not be sufficiently efficient to access large resource sets. However, IUCBRF is currently being extended to enable efficient retrieval directly from a database, following [19], which will enable its use for large-scale sets of resources.

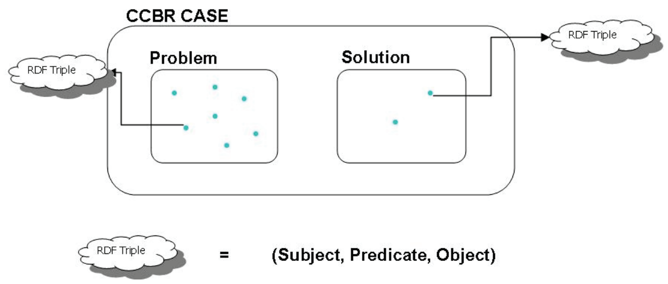


Figure 2: A CBR case consists of a problem and a solution along with additional information such as time of creation, case source, etc. Problem and solution parts of a case are defined as RDF Triples.

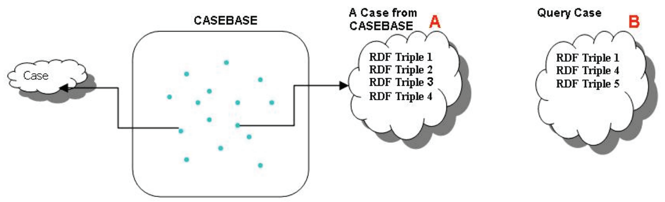


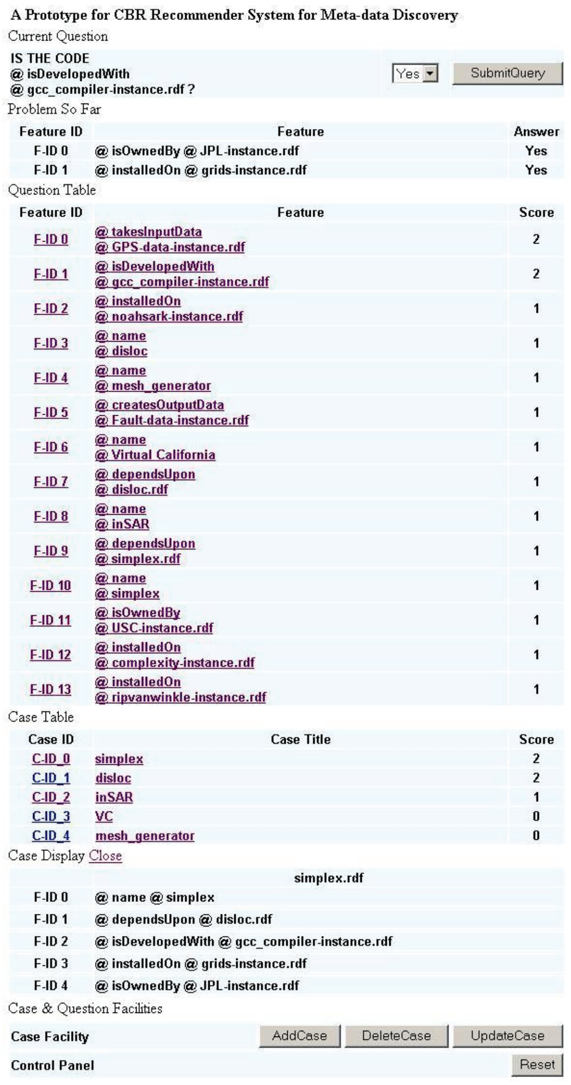
Figure 3: Cases are represented with RDF triples. Queries are incrementally constructed as a set of triples reflecting a question-answer conversation between the system and user.

In IUCBRF, cases contain a description of the problem they address, the solution to a current situation, and bookkeeping information such as the time of case creation, contexts in which they apply, use counts and source information [10]. Both the problem and solution consist of elements of a predefined set of features. We defined domain features as (predicate, predicate value) pairs of an RDF Triple. In the initialization phase, all possible (predicate, predicate value) pairs, derived from an ontology, form pre-defined features of the CCBR Domain, with following qualification: We only take into consideration RDF Triples where the subject is an instance of "ServoObject" class or its subclasses in the high level hierarchy of SERVOGrid ontology. We implemented an RDF Triple feature which implements the Feature Interface defined by IUCBR Framework. Using RDF triples as feature sets of both problem and solution of a case is illustrated in Figure 2. As an example, we can illustrate a feature of a case about a simulation code "simplex.c" with an RDF triple as (servo:simplex-instance, servo:developedBy, servo:JPL-instance). In this example, the feature of case "simplex.c" indicates that "simplex.c" is developed by NASA Jet Propulsion Laboratory. Likewise, a query case, constructed by user-system interaction according to the conversation so far, also contains RDF Triples as sets of features as shown in Figure 3.

At each step of the retrieval, ranked cases and questions to distinguish them are displayed. Our system utilizes a "threshold retrieval method" implemented in IUCBRF. In this method, all cases within a given threshold of difference are retrieved. Cases are eliminated if they are in conflict with problem description according to the conversation so far. When refining a case list, for each case under consideration, each known feature (an RDF triple) is checked against the problem's corresponding features. If the feature is unknown for the problem, no action is taken against or in favor of the case under consideration. If the feature is known, then the value of this feature (of case under consideration) is compared against the value of that feature of the problem. If both values are consistent, the case is given a higher ranking in the list. If the value is inconsistent, the case has an inconsistent feature and the case is eliminated. In this case refinement methodology, cases are ordered by the number of consistent features at each step of the retrieval and cases that have inconsistent features are eliminated.

The system determines which question to be displayed to the user depending on what is known about the problem so far, which cases are under consideration, and the domain. Questions are driven from (predicate, predicate value) pairs. The system decides which questions to be displayed next based on that question's frequency in cases under consideration. A feature which is most frequently known by the cases under consideration gets the highest rank. The question which is driven from this feature is the question to be asked next. Only features that exist within the cases that are under consideration can qualify to be ranked, otherwise those features would be eliminated.

Figure 4 shows the system's user interface. The interface presents a feature list with prioritized questions. Each feature has a hyperlink pointing to SERVOGrid ontology to let the user learn about the description of the predicates. This user interface shows also a list of cases that are still under consideration. Case titles have hyperlinks pointing to RDF representation of the cases; case contents can be displayed by clicking the case identifier on the left. The system also provides facilities to add, delete and update cases.



A Prototype for CBR Recommender System for Meta-data Discovery

Current Question

IS THE CODE
@ isDevelopedWith
@ gcc_compiler-instance.rdf ?

Problem So Far

Feature ID	Feature	Answer
F-ID 0	@ isOwnedBy @ JPL-instance.rdf	Yes
F-ID 1	@ installedOn @ grids-instance.rdf	Yes

Question Table

Feature ID	Feature	Score
F-ID 0	@ takesInputData @ GPS-data-instance.rdf	2
F-ID 1	@ isDevelopedWith @ gcc_compiler-instance.rdf	2
F-ID 2	@ installedOn @ noahsark-instance.rdf	1
F-ID 3	@ name @ disloc	1
F-ID 4	@ name @ mesh_generator	1
F-ID 5	@ createsOutputData @ Fault-data-instance.rdf	1
F-ID 6	@ name @ Virtual California	1
F-ID 7	@ dependsUpon @ disloc.rdf	1
F-ID 8	@ name @ inSAR	1
F-ID 9	@ dependsUpon @ simplex.rdf	1
F-ID 10	@ name @ simplex	1
F-ID 11	@ isOwnedBy @ USC-instance.rdf	1
F-ID 12	@ installedOn @ complexity-instance.rdf	1
F-ID 13	@ installedOn @ ripvanwinkle-instance.rdf	1

Case Table

Case ID	Case Title	Score
C-ID_0	simplex	2
C-ID_1	disloc	2
C-ID_2	inSAR	1
C-ID_3	VC	0
C-ID_4	mesh_generator	0

Case Display Close

	simplex.rdf
F-ID 0	@ name @ simplex
F-ID 1	@ dependsUpon @ disloc.rdf
F-ID 2	@ isDevelopedWith @ gcc_compiler-instance.rdf
F-ID 3	@ installedOn @ grids-instance.rdf
F-ID 4	@ isOwnedBy @ JPL-instance.rdf

Case & Question Facilities

Case Facility

Control Panel

Figure 4: User interface of our CCBR retrieval system. In this facility, a user can interact with the system and the system responds with ranked cases and questions to distinguish them.

5 System Status

Testing the system on a sample case base of 20 cases has provided some initial observations and preliminary results. We anticipate that the online processing time will take more time than traditional retrieval systems as the number of cases increased, though refinements of the retrieval methods used should keep the time manageable. However, the system does facilitate selection of the right code, even for codes with only slight differences, and the user-system interaction seems natural to users. In addition, earthquake modeling codes, which are difficult to be described with textual description, can easily be described with the RDF syntax and developed ontologies.

6 Relevant Work

In recent years, several CBR systems have been developed using XML case representations [6, 7]. In [21], an XML based markup language is introduced as a standard way of representing cases. XML as a semi-structured data representation model focuses on the syntax of the data, whereas the main focus of semantic web languages such as RDF is to provide more descriptive information. RDF graph model is distinct from its XML tree syntax, so complicated relationships can be more easily modeled [9]. For the purposes of our research, this content information is highly important; thus we find the combination of XML and RDF promising for CBR applications.

Personalization of CBR recommender systems has also been explored by various researchers [17, 22, 23]. As a next step, we would also like to explore performance improvements if personal characteristics and preferences of users are taken into consideration.

Recommender systems have been an active research area for some years [24]. Recommender systems have received limited attention in scientific computing environments, but have been applied in projects such as [20, 25]. Such recommenders are promising for environments for which it may be difficult for users to identify the right alternatives.

7 Conclusions and Future Work

This paper has described an effort to design and develop ontologies for an earthquake simulation grid such as SERVOGrid project, and to exploit them to aid users as they select resources. It introduced a prototype web-based CCBR retrieval system which operates on an RDF file store. This system combines RDF representation and CBR recommendation methodology to do code selection for earthquake simulation codes; thus it applies a CBR approach with RDF data models.

Work remains to further develop our preliminary ontology, perhaps by systematic interviewing of code developers, and to refine and evaluate the system through user testing, as well as to extend the system to provide another type of support, guidance for the use of complex earthquake modeling codes.

Acknowledgement. The research is funded through the NASA CT and AIST divisions with contract number of 1253656.

References

[1] D. Aha and L. Breslow and H. Munoz-Avila: Conversational Case-Based Reasoning, Applied Intelligence Journal, Volume 14, Pages 9-32, 2001

[2] Choonhan Youn, Marlon Pierce, and Geoffrey Fox: Building Problem Solving Environments with Application Web Service Toolkits, ICCS03 Australia June 2003

[3] Marlon Pierce, Choonhan Youn, Ozgur Balsoy, Geoffrey Fox, Steve Mock, and Kurt Mueller: Interoperable Web Services for Computational Portals. SC02 2002

[4] Marlon Pierce, Choonhan Youn, and Geoffrey Fox: Interacting Data Services for Distributed Earthquake Modeling. ACES Workshop at ICCS June 2003 Australia

[5] D. Leake, editor. *Case-Based Reasoning: Experiences, Lessons, and Feature Directions.* AAAI Press/MIT Press, Menlo Park, CA, 1996

[6] Gardigen D., Watson I (1998). A web based Case-Based Reasoning System for HVAC Sales Support. Proceedings of British Expert Systems Conference 1998.

[7] Hayes, C., Cunningham, P., Doyle, M., Distributed CBR using XML. In Proceedings of the KI-98 Workshop on Intelligent Systems and Electronic Commerce, number LSA-98-03E. University of Kaiserslauten Computer Science Department, 1998

[8] W3C Semantic Web Site:
http://www.w3.org/2001/sw

[9] W3C - Resource Description Framework Site:
http://www.w3.org/RDF

[10] Steve Boegarts, David Leake, A Framework For Rapid and Modular Case Based Reasoning System Development: http://www.cs.indiana.edu/sbogaert/CBR/IUCBRF.pdf

[11] Mehmet S. Aktas, Marlon Pierce, and Geoffrey C. Fox, Designing Ontologies and Distributed Resource Discovery Services for an Earthquake Simulation Grid GGF-11 Global Grid Forum Semantic Grid Applications Workshop Hawaii June 10 2004

[12] Link to the SERVOGrid ontologies:
http://ripvanwinkle.ucs.indiana.edu:4780/examples-/download/schema

[13] Link to the SERVO ontology instances:
http://ripvanwinkle.ucs.indiana.edu:4780/examples-/download/data

[14] Dublin Core Metadata Initiative Web Site:
http://dublincore.org/documents/dces

[15] The vCard version 3.0 Specifications:
http://www.ietf.org/rfc/rfc2426.txt

[16] Link to the web site of CCBR project presented here:
http://complexity.ucs.indiana.edu/ maktas/servo-/project.html

[17] Goker, M., Thompson, C.: Personalized conversational case based recommendation Proceedings of the Fifth European Workshop on Case Based Reasoning. Trento, Italy

[18] Aha D., Breslow L., ”Refining Conversational Case Libraries”, in Leake D., Plaza E. (eds.) ”Case Based Reasoning Research and Development, 2nd International Conference on Case Based Reasoning ICCBR 1997’, pp.267-268, Springer Verlag, Berlin 1997.

[19] Leake D., ”The Experience Sharing Architecture: A Case Study in Corporate-Wide Case-Based Software Quality Control”, in Leake. D., ”Case-Based Reasoning: Experiences, Lessons, and Future Directions”, pp. 235-268,AAAI Press,Menlo Park, CA, 1996

[20] David C. Wilson, David B. Leake, Randall Bramley Case Based Recommender Components for Scientific Problem-Solving Environments Proceedings of the Sixteenth IMACS World Congress, 2000.

[21] Lorcan Coyle, Conor Hayes, Padraig Cunningham, Representing Cases for CBR in XML, In Proceedings of 7 th UKCBR Workshop, Peterhouse, Cambridge, UK, 2002.

[22] Robin Burke, The Wasabi Personal Shopper: A Case Based Recommender System, Prooceedings of the 11th Conference on Innovative Applications of Artificial Intelligence

[23] Goker, M., Thompson, C.: The Adaptive Place Advisor: A Conversational Recommendation System, Proceedings of the 8th German Workshop on Case Based Reasoning. Lammerbuckel, Germany.

[24] Kautz, H and B. Selman, Creating Models ofReal-World Communities with ReferralWeb, Workingnotes of the Workshop on Recommender Systems,held in conjunction with AAAI-98, Madison, WI,1998

[25] E.N. Houstis and A. Catlin and J. Rice and V. Verykios and N. Ramakrishnan and C. Houstis, PYTHIA II: A Knowledge/Database System for Managing Performance Data and Recommending Scientific Software, ACM Transactions on Mathematical Software Journal, Volume 26, pp. 227-253, 2000