Towards A Semantic-based Approach for Software Reusable Component Classification and Retrieval

Haining Yao
Computer Science Department
University of Alabama in Huntsville
Huntsville, AL 35899, USA
+1 256 824 5139
hyao@cs.uah.edu

Letha Etzkorn
Computer Science Department
University of Alabama in Huntsville
Huntsville, AL 35899, USA
+1 256 824 6291
letzkorn@cs.uah.edu

ABSTRACT

In this paper, we propose a semantic-based approach to improve software component reuse. The whole approach extends the software reusable library to the World Wide Web; overcomes the keyword-based barrier by allowing user queries in natural language; treats a software component as a service described by semantic service representation format; enhances the retrieval by semantically matching between a user query semantic representation and software component semantic descriptions against a domain ontology; and finally stores the relevant software components into a reusable repository based UDDI infrastructure. The technologies applied to achieve the goal include: Natural Language Processing, Web services, Semantic Web, Conceptual Graph, domain ontology. The research in the first phase will focus on the classification and retrieval for software reusable components. In the classification process, natural language processing and domain knowledge technologies are employed for program understanding down to code level, and Web services and Semantic Web technologies as well as Conceptual Graph are used to semantically describe/represent a component. In the retrieval process, a user query in natural language is translate into semantic representation formats in order to augment retrieval recall and precision by deploying the same semantic representation technologies on both the user query side and the component side.

Categories and Subject Descriptor:

D.2.2 Design Tools and Techniques; D.2.13 Reusable Software

General Terms

Design, Experimentation

Keywords

Reuse repository, reusable library, semantic matchmaking, domain knowledge base, ontology, system develop

1. INTRODUCTION

Software reuse refers to using software modules that were developed on a previous software project as part of a new software development project. Software reuse is a worthwhile goal since it has been shown to reduce software costs, and improve software quality as well as programmer productivity [13]. Also, the move toward component-based software development, which is a systematic method for using reusable components from in-house component repositories and from commercial component providers, has recently re-ignited interest in software reuse research [26]. Vitharana and Jain [31] argue that a system developed from reusable components is more reliable. Due [7] and Kim and Stohr [14] describe how component-reuse can shorten the development cycle and thus reduce time to market.

In addition to software searches on software libraries/component repositories, one new source for potentially-reusable software components is the World Wide Web. The Internet has become an abundant base of potentially reusable software, with extensive quantities of software freely available for users. Although software licensing can sometimes be an issue when considering this software for commercial applications, still, for many people and in many situations the World Wide Web can be considered to be an immense software reuse library.

Unfortunately, search engines such as Google, which is the most widely used search engine today [1][2][16][17] are general search engines, and are not customized for software component searches. Google will present substantial unrelated information when people are searching for software components; also, Google's search is primarily keyword-based, and there is a lack of domain-knowledge associated with its queries. However, the primary drawback with Google or other search engines such as Yahoo is that they do not specifically consider software as a search target.

This approach focuses on methods to improve software searches and retrievals, both in large software component libraries, and also on the World Wide Web. There are two prongs to this approach: 1) improving the search capabilities of software reuse libraries through annotating software packages in these libraries with a description of the services provided by the software, using program understanding techniques to do this; and using a natural language-based search engine to match software services desired to software services available, in order to pick the best match, and 2) improving searching for software on the World Wide Web through the use of program understanding: web sites are searched for candidate software packages, then program understanding techniques are applied to annotate this software with a description of services. A natural language-based search engine will access this software and perform matching.

The innovative aspects of the approach are: 1) The World Wide Web is treated as a large software library, and software searches on the World Wide Web are specifically addressed. 2) For both existing software component libraries and for the World Wide Web software library, program understanding techniques are employed to tag components with an extensive semantic description; then a corresponding natural language-based search engine is employed to help the user describe the desired component. Since similar semantic techniques are employed on both ends, component-searched-for, and user-search-description, there is great potential for improved matching of existing components to the user-desired component, compared to existing software reuse library search techniques. 3) The techniques employed will extend beyond those employed by Sugumaran and Storey [27]: Semantic Web technologies such as the RDFS/DAML+OIL ontologies will be employed (this is similar to those employed by Suguraman and Storey [27]), as well as WSDL and UDDI. Conceptual graphs will also be employed. 4) Automated conversion between conceptual graphs and semantically augmented WSDL will be employed. Different conversion techniques will be examined. 5) Semantic matching between user queries and component service descriptions will be employed: Conceptual graph matching will be employed similar to that of Puder et al. [25]. Semantically augmented WSDL matching will be employed, similar to that of Paolucci et al. [21][22] and Sivashanmugam et al. [26]. These different matching algorithms will be compared for speed and accuracy, and variations on these algorithms will also be examined.

2. Related Research

Over approximately the past fifteen years, software reuse research has focused on several different areas: examining programming language mechanisms to improve software reusability; developing software processes and management strategies that support the reuse of software; also, strategies for setting up libraries containing reusable code components, and classification and retrieval techniques to help a software engineer select the component from the software library that is the most appropriate for his or her purposes. Recently, although in the past software reuse research has focussed on identifying reusable artifacts, storage and retrieval software component techniques have attracted more attention, because it is essential for software developers to retrieve reusable software components from large software libraries. Also component-repositories have been experiencing exponential growth, with the result that efficient means to search and retrieve software components from large repositories are still a major software reuse research topic [27].

2.1 Software Reuse Classification and Retrieval

"A classified collection is not useful if it does not provide the search-and-retrieval mechanism to use it" [23]. A wide range of solutions to the software reuse classification and retrieval have been proposed and implemented. At different times, based on available software reuse systems and also based on researchers' criteria, software reuse classification and retrieval approaches have been classified differently, but with minor differences.

Ostertag et al. [20] classified the reported approaches by three types: 1) free-text keywords, 2) faceted index, and 3) semantic-net based. Free-text keyword based approaches basically use information retrieval and indexing technology to automatically extract keywords from software documentation and index items with the keywords. The free-text keyword approach is simple, and it is an automatic process. But the free-text keyword based approach is limited by lack of semantic information associated with keywords, thus it is not a precise approach. For faceted index approaches, experts extract keywords from program descriptions and documentation, and arrange the keywords by facets into a classification scheme, which is used as a standard descriptor for software components. To solve ambiguities, a thesaurus is derived for each facet to make sure the keyword matched can only be within the facet context. Faceted classification and retrieval has proven to be very effective in retrieving reuse component from repositories, but the approach is labor intensive. Semantic-net based approaches usually need a large knowledge base, a natural language processor, and a semantic retrieval algorithm to semantically classify and retrieve software reuse components. The semantic-net based approach is also labor intensive, and it is also rigid in a narrow application domain.

Mili et al. [19] classifies search and retrieval approaches into four different types: 1) simple keyword and string match; 2) faceted classification and retrieval; 3) signature matching; and 4) behavior matching. In their summary of component retrieval schemes, Sugumaran and Storey [27] state that the last two approaches, 3) signature matching and 4) behavior matching, are cumbersome and inefficient. The classification here for the first two approaches, 1) simple keyword and string match and 2) faceted classification and retrieval, is mostly the same as other researchers' classification of the two approaches.

Mili et al. [18] designed a software library in which software components are described in a formal specification: a specification is represented by a pair (S, R), where S is a set of specification, and R is a relation on S. The approach is classified as a keyword-based retrieval system, while matching recall is enhanced with sufficient precision: a match is considered as long as a specification key can refine a search argument. Besides, there are two retrieval operations: exact retrieval, and approximate retrieval. If there is no exact retrieval, approximate retrieval can give programs that need minimal modification to satisfy the specification.

The faceted classification scheme for software reuse proposed by Prieto-Díaz and Freeman [23] relies on facets which are extracted by experts to describe features about components. Features serve as component descriptors, such as: the component's functionality, how to run the component, and implementation details. To determine similarity between query and software components, a weighted conceptual graph is used to measure closeness by the conceptual distance among terms in a facet.

Vitharana et al. [32] proposed a scheme to classify and describe business components within a knowledge-based repository for storage and retrieval. Two important steps in their proposed scheme are: 1) classification and coding for business components; 2) knowledge-based repository for storage and retrieval. The classification groups similar parts, and symbols are coded. They borrowed the idea from the facet-based scheme to describe features of reusable software artifacts, whereas their classification and coding scheme considers higher level business oriented features. In their proposed classification and coding scheme, a business component is described by identifiers (structured information such as name, industry type), followed by descriptor facets (unstructured information, such as rules, functionality). In their knowledge-based repository design, a database is used as the repository because it is efficient and effective for storage, search, and retrieval; eXtensible Markup Language (XML) is used to code the knowledge base because XML is suitable to the extensible numbers of descriptor facets.

Girardi and Ibrahim's [11] solution for retrieving software artifacts is based on natural language processing. Both user queries and software component descriptions are expressed in natural language. Natural language processing at the lexical, syntactic, and semantic levels is performed on software descriptions to automatically extract both verbal and nominal phrases to create a frame-based indexing unit for software components. Both user queries and software component descriptions are semantically formalized into the internal representation, canonical forms. Then the matching in between is reduced to compute the closeness of the query and the software component description, which is the distance of the two canonical forms. The retrieval system looks for closeness and candidate software components can be ranked by the quantified value of closeness. A public domain lexicon is used to get lexical information for both the query classification and the software classifications. To avoid rigidity, this solution employs partial canonical forms to allow some ambiguities and to infer all possible interpretations of a sentence.

Sugumaran and Storey [27] present a semantic-based solution to component retrieval. The approach employs a domain ontology to provide semantics in refining user queries expressed in natural language and in matching between a user query and components in a reusable repository. The approach includes a natural language interface, a domain model, and a reusable repository. The three major steps in the approach are: 1) initial query generation, 2) query refinement, and 3) component retrieval and feedback. Initial query generation is heuristic-based natural language processing to abstract keywords from user queries. Their approach for natural language processing of user query and component descriptions is based on the ROSA system [10]. Query refinement is achieved by mapping the initial query against an ontology to ensure correct terms are used in the query. In the component retrieval and feedback step, a closeness measure proposed by Girardi and Ibrahim [11] is employed to identify which are the most relevant components to a user's query.

2.2 Program Understanding

Program understanding is one of the steps involved in code reuse [23]. "Reusable software resources are considered worthless if they cannot be understood and if the likelihood of their successful incorporation into a new software system can not be assessed" [14]. Program understanding by humans is very time-consuming, thus an automatic tool employing a knowledge-based program understanding approach is necessary to semantically and structurally understand a program and identify its reusability.

2.3 Semantic Service Description and Discovery Technologies

Web services are an industry wide project for service description and discovery. Simply, Web services defines a "standardized mechanism to describe, locate, and communicate with online applications"[12]. Three major areas of Web services framework are: communication protocols, service descriptions, and service discovery. The most salient and stable specifications in each area are: Simple Object Access Protocol (SOAP) as the communication protocol, Web Service Description Language (WSDL) for the services description, and Universal Description, Discovery, and Integration (UDDI) as the registry directory for service discovery [3]. WSDL [33] is an XML-based grammar used to describe a business service. Critical information described by WSDL includes: interface information for publicly available methods, data type information for messages, binding information for transport protocols, and address information for locating services. However, as one of the XML based standards, WSDL does not support semantics for services descriptions [21][22]. UDDI [30] is an industry wide framework to provide open and standard specification for Web services: describe services; discover business and services; create and deploy an XML-based registry. The goal of UDDI is to provide an Internet registry for web services. UDDI search is only a keyword-based search, and considered poor on searching because UDDI is also an XML based standards which share the limitation of lack of an explicit semantics [21][22].

Semantic Web [28] technologies are based on RDF (Resource Description Framework) [24]. RDF is a general-purpose knowledge representation language to provide a basic object-attribute-value data model for metadata. An RDF description is a list of triples: an object (a resource), an attribute (a property), and a value (resource or free text). This object-attribute-value data model provides a common framework for expressing machine processable information so it can be exchanged between applications without loss of meaning. And the object-attribute-value structure can be nested, which provides natural semantic units in which objects are independent units. Compared with XML, RDF has a significant advantage in semantic interoperability, so it has turned out as an excellent complement to XML [5][6][15][34]. But both XML and RDF are considered as lightweight schema language when compared with DAML+OIL [4] which is an ontology language specially designed for use in the Semantic Web. DAML+OIL is based on RDF, but extends RDF with new modeling primitives and formal semantics to RDF schema. A DAML+OIL knowledge base is a collection of RDF triples.

Trastour et al. [29] experimented with a semantic web approach for service description and matchmaking on a B2B marketplace application by both RDF and DAML+OIL technologies. Based on an ontology described in semantic web representations, service matching is reduced to match service advertisements in an RDF graph by using a matching algorithm. Paolucci et al. [21][22] and Sivashanmugam et al. [26] combine Web services and Semantic Web technologies, and the combination powers an automated search to the current UDDI registry, whose inherent inhibition now is lack of semantics in the discovery process.

3. Semantic-Based Approach for Software Reusable Component Classification and Retrieval

The approach described in this paper focuses on methods to improve searching for and retrieving software reuse components, both in large software component repositories, and also on the World Wide Web. There are two prongs to this approach. 1) Improve the search capabilities of software reuse libraries through annotating software components and packages in these libraries with a semantic description of the services provided by the software. Techniques used to achieve the goal includes: natural language processing on queries, and also on program understanding, reuse metrics to evaluate reusability, semantic service description, and domain knowledge base applied for whole process for semantic description and retrieval. 2) Improve searching for software on the World Wide Web through the use of program understanding. Web sites are searched and downloaded for candidate software packages, then program understanding techniques are applied to annotate this software with a description of services. A natural language-based search engine will access this software and perform matching. The approach will include three major sub sections:

Section 1: An intelligent, natural language-based user interface. The user will specify queries in basically unrestricted natural language. These queries will be represented in a conceptual graph semantic representation within a knowledge-base, and will also be translated into Semantic web-based representation. DAML+OIL ontologies are employed to support domain knowledge for the Semantic web based presentation.

Section 2: An analysis and annotation tool employing program understanding to identify and describe the functionality of the software components using semantic representation. The semantic representation will be in a conceptual graph knowledge-base, and will also be translated into a Semantic-web-based representation. DAML+OIL ontologies are employed to support domain knowledge for the Semantic web based presentation.

Section 3: A semantic matchmaker based on a domain knowledge base. The matchmaker will compare a user query in a conceptual graph with component service descriptions in conceptual graphs.

The full web-based approach, to consider the World Wide Web as a software library, would also employ:

Section 0: An intelligent Internet search engine to automatically search and download software components from the Internet based on user requests, to be annotated by Section 2.

The full software reuse library approach would also include:

Section 4: A software component repository that uses UDDI as its infrastructure, and WSDL and DAML-S as service description languages. Components in the repository would be annotated with WSDL/RDF service descriptions.

As proof of concept for this approach, a working system that implements the approach will be developed. Figure 1 describes this system.

The proposed system described in Figure 1 will implement software reuse analysis, classification and retrieval as follows:

Accept a user query in natural language; apply natural language processing technology to analyze a user query based on semantics, and translate the query into conceptual graphs, then into WSDL/RDF.

A program understanding tool is applied to analyze downloaded software packages [8][9]; software services and features associated with the package will be translated into conceptual graphs also, then into WSDL/RDF.

Finally a semantic matchmaker will search for desired software components by matching the user query conceptual graphs to components' conceptual graphs. Then WSDL/RDF representations of the user query and components will be matched; a comparison will be made between conceptual graph matching and WSDL/RDF matching.

Conceptual graph matching similar to that used by Puder et al. [25] will be employed. Puder et al.'s [25] matching algorithm was originally applied for an AI Trader system to match service requests between Service Requester and Service Provider. In our approach, a similar algorithm will be applied to match between query conceptual graph and software reusable component conceptual graph. Both service and request are described in weighted conceptual graphs, the weights in the conceptual graph reflect the importance of the nodes, and then an akin index is computed based on the semantic intersection between the two graphs. The akin index shows how similar the two graphs are, then indicate how close the user query is to the reusable components.

However, a semantic matching based on WSDL annotated with RDF will be employed as well as the conceptual graph description and matching. A reusable component is a service and its functionality can be semantically described. Service description and matching technologies of Semantic web and Web services are naturally applicable for software reuse component description, classification, and retrieval. WSDL is an industry standard description language for service description, and WSDL service description usually describes interface information for publicly available methods, data type information for messages, binding information for transport protocols, and address information for locating services. Considering the description requirements for features of a reusable software component, WSDL is naturally suitable to meet the need for software reusable component classification and retrieval. Although since WSDL does not support semantic description, it can not support semantic matching. So using Semantic Web technologies to annotate WSDL descriptions will augment the classification and retrieval process by connecting the query and descriptions to domain. Sivashanmugam et al.'s [26] approach of annotating WSDL by DAML+OIL ontologies will be adopted: reusable software component description depicted in conceptual graph will be converted to WSDL description, which will then be annotated with DAML+OIL ontologies. Paolucci et al.'s [21] matching algorithm will be

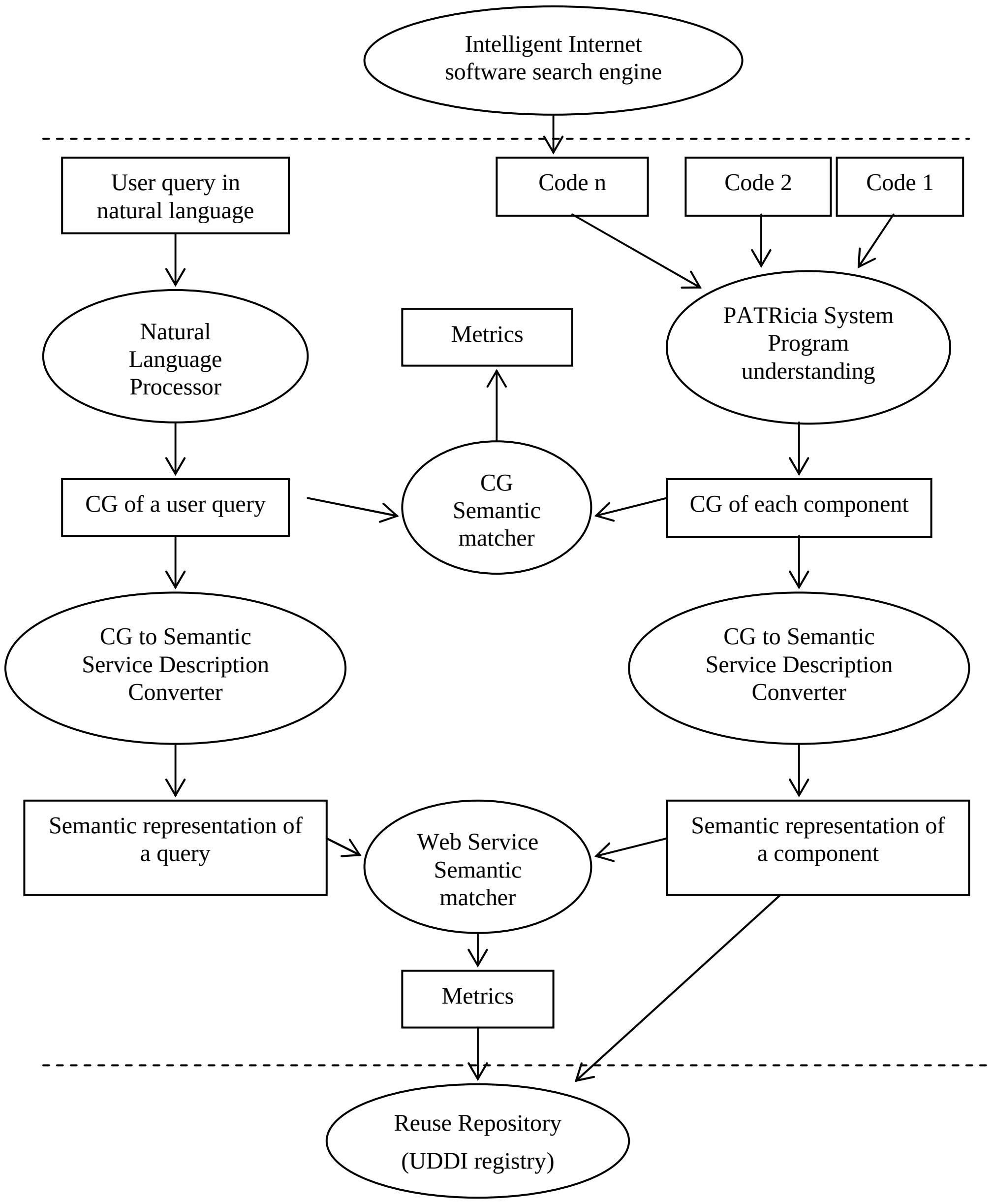
Intelligent Internet software search engine
User query in natural language
Code n
Code 2
Code 1
Natural Language Processor
Metrics
PATRicia System Program understanding
CG of a user query
CG Semantic matcher
CG of each component
CG to Semantic Service Description Converter
CG to Semantic Service Description Converter
Semantic representation of a query
Web Service Semantic matcher
Semantic representation of a component
Metrics
Reuse Repository (UDDI registry)

Figure 1: System Data Flow Diagram

adopted to compute closeness between the query and the semantic annotations in DAML+OIL to find the components that best matched the query.

These two kinds of matching, by conceptual graph and by WSDL annotated by DAML+OIL, will be compared. Additional matching metrics will also be gathered and examined.

4. Validation

An experiment involving human experts will be organized. A component repository containing several different components of similar types will be developed. In this case, the component repository will consist of C++ software in several subdirectories of a main directory. Experts will request components and be provided with the most appropriate matching components. Experts will fill out a questionnaire regarding how well the match was performed. Metrics for recall, precision, and overgeneration will be calculated on this result. The results of the component matching experiment will be evaluated for recall, precision, and overgeneration.

Recall is a measure of completeness of components matching. Recall can be defined as the proportion of the number of relevant, retrieved components to the number of all relevant components in the repository. Precision is a measure of the accuracy of component matching. Precision can be defined as the ratio of the number of retrieved, relevant components to the number of the all retrieved components. Overgeneration is a measure of spurious generation of component matching. Overgeneration is complementary to the precision ratio.

The second experiment will also be performed by human experts. These experts will search for a particular type of software package using Google or Yahoo, they will pick 5 packages each that will be downloaded. Experts will request their preferred component and will be provided with the most appropriate matching component. Again, metrics for recall, precision, and overgeneration will be calculated on this result. The definitions for recall, precision, and overgeneration are same as the definitions given above.

5. Summary

The purpose of the approach is primarily to automate the process for software search and retrieval: automated reusable software searches on the World Wide Web; natural language processing, program understanding, and semantic representation techniques employed to annotate a software component with a semantic description of the services and operations provided by the component; a natural language interface provided to users for a semantic description of the queries. Various Semantic Web and conceptual graph matching techniques will be employed to match the queries to the components.

6. REFERENCES

[1] Brin, Sergey; Page, Lawrence. (1998). "The anatomy of a large-scale hypertextual Web search engine," *Proceedings of the seventh international conference on World Wide Web 7*, April 1998, Brisbane, Australia, pp.107-117.

[2] Broder, Andrei. (2002). "A taxonomy of web search," *ACM SIGIR Forum*, P. 3-10.

[3] Curbera, Francisco; Duftler, Matthew; Khalaf, Rania; Nagy, William; Mukhi, Nirmal; Weerawarana, Sanjiva. (2002) "Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI," *IEEE Internet Computing*, vol. 6, no. 2, pp. 86-93.

[4] DARPA Agent Markup Language (DAML) http://www.daml.org, accessed Oct. 12, 2003.

[5] Decker, Stefan; Melnik, Sergey; Harmelen, Frank Van; Fensel, Dieter; Klein, Michel; Broekstra, Jeen; Erdmann, Michael; Horrocks Ian. (2000). "The Semantic Web: The Roles of XML and RDF," *IEEE Internet Computing*, Vol. 4, No. 5, pp. 63-74.

[6] Decker, Stefan; Mitra, Prasenjit; Melnik, Sergey. (2000). "Framework for the Semantic Web: An RDF Tutorial," *IEEE Internet Computing*, Vol. 4, No. 6, pp. 68-73.

[7] Due, Richard T. (2000). "The Economics of Component-based Development," *Information Systems Management*, Vol. 17, No. 1, pp. 92-95.

[8] Etzkorn, L.H.; Davis, C.G. (1997). "Automatically Identifying Reusable Components in Object-Oriented Legacy Code," *IEEE Computer*, Vol. 30, No. 10, pp.66-71.

[9] Etzkorn, Letha H. (1997). *A Metrics-based Approach to the Automatic Identification of Object-oriented Reusable Software Component*, doctoral dissertation, University of Alabama in Huntsville, Huntsville, AL.

[10] Girardi, M.R.; Ibrahim, B. (1993). "A Software Reuse System based on Natural Language Specifications," Proceedings of the 5th International Conference on *Computing and Information (ICCI'93)*, Sudbury, MA, May 27-29, 1993, pp. 507-511.

[11] Girardi, M.R.; Ibrahim, B. (1995). "Using English to Retrieve Software," *Journal of System and Software*. vol. 30, no. 3, pp.249-270.

[12] Guha, R.; McCool, Rob; Miller, Eric (2003) "Semantic Search," *Proceedings of the twelfth international conference on World Wide Web*, Budapest, Hungary, pp. 700 - 709

[13] Hooper, J.W.; Chester, R.O. (1991) *Software Reuse and Methods*, Plenum Press, New York.

[14] Kim, Yongbeom; Stohr, Edward A. (1998) "Software Reuse: Survey and Research Directions," *Journal of Management Information Systems*, Spring 1998, Vol. 14, No. 4, pp. 113-147.

[15] Klein, Michel (2001) "XML, RDF, and Relatives," *IEEE Intelligent Systems*, Vol. 16, No. 2, pp. 26-28.

[16] Kobayashi, Mei; Takeda, Koichi. (2000). "Information Retrieval on the Web" *ACM Computing Surveys*, Vol. 32, No. 2.

[17] McFedries, P. (2003). "Google this," *IEEE Spectrum*. vol. 40, no. 2, p.68.

[18] Mili, Rym; Mili, A.li; Mittermeir, Roland T. (1997). "Storing and retrieving software components: a refinement based system," *IEEE Transaction on Software Engineering*. vol. 23, no. 7, pp. 445-460

[19] Mili, Rym; Mili, A.li; Mittermeir, Roland T. (1998). "A Survey of Software Storage and Retrieval," *Annal of Software Engineering*, vol. 5, no. 2, pp. 349-414.

[20] Ostertag, Eduardo; Hendler, James; Prieto-Díaz, Rubén; Braun, Christine. (1992). "Computing similarity in a reuse library system: an AI-based approach," *ACM Transaction on Software Engineering and Methodology*, vol. 1, no. 3, pp. 205-228

[21] Paolucci, Massimo; Kawamura, Takahiro; Payne, Terry R.; Sycara, Katia P. (2002). "Semantic Matching of Web Services Capabilities," *Proceedings of the First International Semantic Web Conference on The Semantic Web*, Pages: 333 – 347, June 09 - 12, 2002.

[22] Paolucci, Massimo; Kawamura, Takahiro; Payne, Terry R.; Sycara, Katia P. (2002). "Importing the Semantic Web in UDDI," *Web Services, E-Business and Semantic Web Workshop*, 2002.

[23] Prieto-Díaz, Rubén; Freeman, Peter. (1987). "Classifying Software for Reuse," *IEEE Software*, vol. 4, no. 1, pp. 6-16.

[24] Resource Description Framework (RDF) http://www.w3.org/RDF, accessed Oct. 12, 2003

[25] Puder, A.; Markwitz, S.; Gudermann F. (1995). "Service Trading Using Conceptual Structures," *International Conference on Conceptual Structures (ICCS'95).*

[26] Sivashanmugam, K.; Verma, K.; Sheth, A.; Miller, J. (2003). "Adding Semantics to Web Services Standards," *The 2003 International Conference on Web Services (ICWS'03)*, June 2003.

[27] Sugumaran, Vijayan; Storey, Veda C. (2003). "A Semantic-Based Approach to Component Retrieval," *The DATA BASE for Advances in Information Systems – Summer 2003*, Vol. 34, No. 3, p. 8-24

[28] Semantic Web (SW) http://www.w3.org/2001/sw/, accessed Oct. 13, 2003

[29] Trastour, David; Bartolini, Claudio; Gonzalez- Castillo, Javier (2001). "A Semantic Web Approach to Service Description for Matchmaking of Services," Technical Report, HPL-2001-183, 20010730, HP Labs Bristol, UK.

[30] Universal Description, Discovery Integration (UDDI) http://www.uddi.org, accessed Oct. 13, 2003.

[31] Vitharana, Padmal; Jain, Hemant (2000) "Research issues in testing business components," *Information and Management*, vol. 37, no. 6, pp. 297-309

[32] Vitharana, Padmal; Zahedi, Fatemeh M.; Jain, Hemant (2003) "Knowledge-based repository scheme for storing and retrieving business components: a theoretical design and an empirical analysis," *IEEE Transactions on Software Engineering*, vol. 29, no. 7, pp. 649-664

[33] Web Service Description Language (WSDL) http://www.w3.org/TR/wsdl, accessed Oct. 12, 2003.

[34] eXtensible Markup Language (XML) http://www.w3.org/XML, accessed Oct. 12, 2003.