A MULTI-AGENT SYSTEMS ENGINEERING FOR SEMANTIC SEARCH OF REUSE SOFTWARE COMPONENTS

[1]DR. NEDHAL A. AL-SAIYD, [2] DR. MUNA F. AL-SAMARAE, [3]DR. INTISAR A. AL-SAYED,

[1] Assoc. Prof., Computer Science Dept., Faculty of Information Technology, Applied Science University, Amman Jordan

[2] Assoc. Prof., Faculty of Management Information System, Al-Zaytoonah University, Amman Jordan

[3] Assoc. Prof., Computer Science Dept., Faculty of Information Technology, Al-Isra University, Amman Jordan

Abstract
In component-based software development (CBSD) approach, the searching and retrieving of accurate reusable software components that are stored in large, distributed and heterogeneous-structured repositories is a tedious and time consuming process. This paper presents the design and implementation of ontology-based multi-agent software component retrieval system using semantic and structural formalism. A multi-agent system (MAS) is used for searching interconnected repositories and retrieving the desired software component. It consists of integrated tasks that are developing an internal keyword/concept reference table (Terms Vector), understanding the meaning user's query using keyword matching, structuring the component specification in repository, developing a semantic search engine, categorizing the selected components, and presenting the relevant retrieving component in user-friendly way. The users inset a free form of a natural language query without having to know about the vocabulary or the structure of the ontology, and the system produces better relevant results than the traditional keyword-based retrieval systems. An Ontology Web Language (OWL) is used to represent ontologies of component knowledge base and knowledge modeling is built using Protégé and OWL-based representation. The implementation of the ontologies models is separated from the implementation of the semantic retrieval system of reusable components. Therefore, it is easy to maintain.

Keywords: *Component-Based Development, reusable Software Component, Multi-Agent Software Engineering, Semantic Search, Ontology, knowledge Modeling.*

1. Introduction

The growth of component-based software Development (CBSD) increases the use of reusable of software artifacts and components, which) exist in heterogeneous and distributed software repositories. The development requires the design of significant mechanisms to search and retrieve the relevant reusable components. There are at least three main methods to search and retrieve the relevant Web information the users may need [1],[2], [3]:

- *Keyword*-based *matching* searching method, which depends on sending a user's query to a search engine and search for keywords to answer the queries. It depends only on matching the keywords of query to the keywords of Web pages. It is the early traditional method, and the search engine may give partially ambiguous result data. Generally, the keyword-based search engine consists of four basic units crawler, indexer, ranker and query procedure [4].
- The *Web-site browsing* method, which helps a user to follow the hyper-links and navigate through a content of website that seem of interest to them.
- *Categorical information* searching method, which follows existing categories in search engines to assist the user in finding relevant Web pages.

The traditional keyword-based search engines like *Google*, *Yahoo*, *AltaVista* and others still dominated the Web search engines to answer users' queries, while research on semantic web search engines is in the initial stages [4]. The problem of searching for useful reusable software component on the Web is a time consuming and tedious process because the user may not state what he wants precisely and the search engines have not the ability adjust their search mechanisms according to different user's intention. Moreover, the problem will be worsen due to the continuously growing size of software components repositories and they have different structure, different content and are distributed on the Web. Also, the information sources of most of the reusable components are irrelevant to a particular user's interests. These reasons make the understanding of user's query difficult process and sometimes producing different search results due to the differences in data indexing and search process of

different search engines [5]. Therefore, the repositories need to have some sort of semantic structure.

The following parts of this paper are as follows. Section 2 introduces Problems of Keyword-based search engines. Section 2 surveys the main problems of using keyword-based search engines in information retrieval. Section 3 introduces the ontology-based multi-agent software for component retrieval, which includes the details of the multi-agent software engineering phases and embedding ontology-based model. Section 4 delineates the architecture of the architecture of ontology-based retrieval framework for reusable software components, which describes the semantic search application layers. An application experiment is explained in details in section 5. Finally, the conclusions are explained in section 6.

2. Problems of Using Keyword-Based Search Engines

Using general keywords-based search engines to search for software components cause the following serious problems [6] [7] [8]:

- The representation of data in the software components repositories, i.e. software components specifications, lack a proper semantic,
- The keyword-based search engines are incapable to understand the user's query intent due to lack of a generic standard format.
- Searching results are highly sensitive to the input keywords. For example the word 'object' has many definitions including: thing, entity, item, purpose, objective, aim, idea, goal, intent, reason, complain, target …etc. A good refinement words need to be domain specific.
- Keyword-based search engines have high recall, low precision. The keyword-based search engines producing different search results irrelevant to the intended user's query. They are ambiguous or partial ambiguous because of poor interconnection among repository data and the variation in indexing and search process,
- The software component repositories are distributed on web. They have different structures, content and representations of component information.
- The search engines are not compatible to work on variety of component repositories because there is no standard format among global repositories.
- The component-based software engineers, who need reusable components in component-base developing, have different background knowledge.

3. Ontology-Based Multi-Agent Software For Component Retrieval

How can a search engine map user's query to software components in the repositories in an intelligent and meaningful way?
Advanced knowledge management systems map the metadata knowledge to the relevant ontologies, and organize them conceptually according to their meaning. The semantic Web system works on discover the meaning of the keywords and the user's query instead of work on keywords. Therefore, the ontology-based search engine will internally reformulate the query in terms of similarities and close properties. Semantic Web infrastructure uses the ontologies to represent the concepts and to accurately describe them into well-defined vocabulary, which are understood by people and computers [7], [9]. Semantic uses Resource Description Framework (RDF) and Web Ontology Language (OWL), as a W3C standardization, To describe ontologies into representation models [10], [11].

The system finds the most common ontology concept with the highest rank that is fall into pre-defined category. It helps in filtering search results. It involves the clustering of the low-level component features, retrieving, and presenting the component in user-friendly way. The definition and the structure of components information will be well-specified and structured to enable the interoperability, understandability, accessibility, and the integration between distributed components repositories and machine systems.

  - Content-based features of component properties and constraints on properties to web resources. Properties are related to component services, standardized component model (that consists fully documentation, implementation and deployment usage), component composition, interface specification, Naming convention, Meta-data access, Customization, some component platform, Evolution support, Component communications, Component trustworthiness (i.e. trusted source code), Component quality certification, size, development history, component data and control dependencies.
- Use Ontologies and semantic annotations to specify the software components [**3**]. They are accompanied by a rich set of unstructured and a nonstandard metadata that describe various details related to each software component, which are later structured to form a knowledge representation that is understandable and is processable by a computer. There is a separate ontology for each category correspond to the collection software components, and these can be used for searching purposes. The data that is required to search for software components depending on platforms, is

significantly different from the data required for service searching. The *Annotation* ontology is used to conceptualize and capture all common data in a structured way, where the specific data does not overlap across component categories.

- Use Ontologies to specify meaning of user's query using explicit semantic information, which can be used by intelligent agents to solve complex query-answer problems.

We adopt a multi-agent software (MAS) and ontology-based approaches, where these issues are considered in developing intelligent software agents, since they can connect between the ontology knowledge and content-based features of software components. Intelligent software agents are widely used in significant model-based solutions that provide an interoperable interface and can run autonomous without human intervention [12]. Due to its appropriateness for open environments, the agents are commonly applied to distributed, large-scale, and dynamic systems. They improve the goal-directed processes of information finding filtering and information interpreting, and decision making [12], [13]. The agent also applies a set of predefined rules that are specified by the user that allow to transform conditions into decisions.

Task = Agents + Interactions (Inter-agent and/or intra-agent) + set of predefined rules

Each agent has attributes to define agent identity, authority, and an access to the resources that the agent can have. Different software gents may have different internal structure, but they need to communicate with other agents. [14] [15]. The general MASE development phases is shown in Figure 1. The rounded rectangles are used to capture the output of each step, they denote the MASE models. The arrows between them show how the models affect each other. Every object created during the analysis and design phases can be traced forward or backward through different steps to other related objects [16], [17].

In general, the adopted multi-agent software engineering (MASE) has the following phases and processes:

i. Domain-Specific Analysis Phase: It consists of three steps:

a. Capturing Goals, transforming it into a hierarchical structure of sub-goals
b. Translating goals into roles and tasks, and generating a use case and sequence diagrams
c. Refining roles, and identifying the concurrent tasks.

ii. Top-Down Design Phase: convert roles and tasks into a representation of agents and their interactions that is more suitable for implementation. Design phase consists of four steps:

a. Creating Agent Classes, and producing agent class diagram..

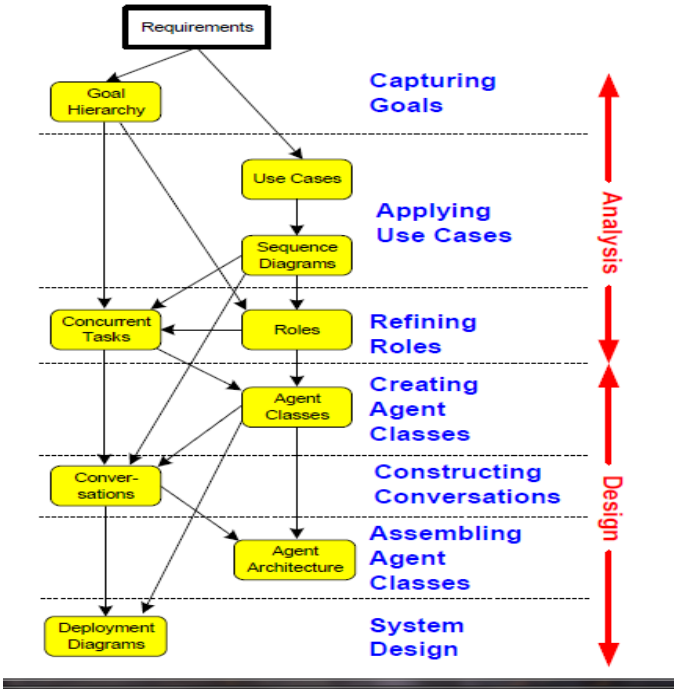

Fig. 1 MASE Development Phases [15]

b. Constructing Conversations, and producing a conversation diagrams.
c. Combining Agent Classes, and producing agent architecture diagrams, as it is shown in figure 2.

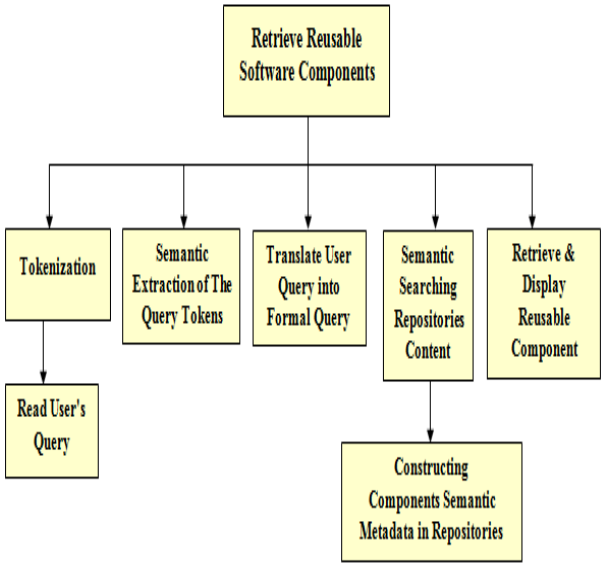

Fig. 2 Goal Hierarchal Diagram of Intelligent Agents and Semantic Web Services

d. Performing system Design and generating deployment diagrams. JADE (Java Agent DEvelopment Framework) is used for heterogeneous and interacting agents and agent-based systems [18] [19], [20]. JADE supports converting between formats, which is suitable for content exchange and content manipulation. This support is integrated with Protégé as ontology tool to create ontologies graphically and validate the exchanging messages among agents [21]. Agents interact by asynchronous exchanging message. An agent just sends a message to a destination that is identified by destination name under certain conditions [22], [23]. An agent provides a set of services that are requested by another agent. A service may have a set of actions depending on the goals.

iii. Ontology Engineering Using Protégé

Our model is based upon embedding ontology in the multi-agent system to communicate among software agents. Ontology specifies the conceptualization in terms of concepts, properties, relations and constraints [23]. Ontologies are usually organized into a taxonomy tree where each node represents a concept with its attributes, as shown in figure 3. Each concept in taxonomy is associated with a set of instances. They share common understanding of the information structure that can be used to extract information to answer queries.

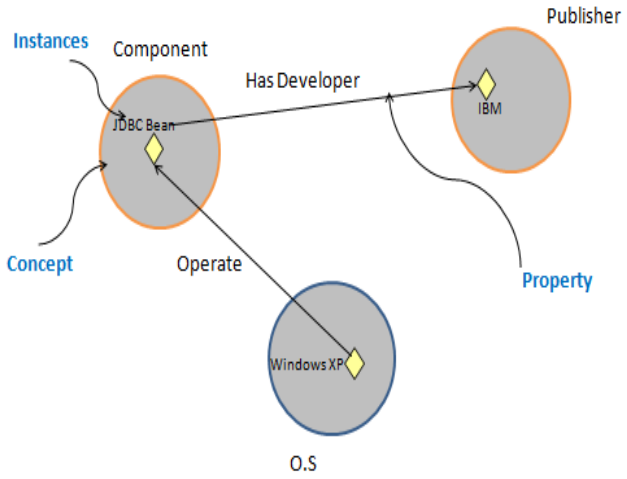

Fig. 3 An Example of Ontology

OWL Protégé, which is based on the Resource Description Framework (RDF), is used to describe properties, relation between classes, cardinality, characteristics of properties. Agents can then read ontologies and interpret them using interaction with other agents. OWL models are compatible with World Wide Web and add certain capabilities to ontologies to develop Semantic Web application [24] [25], [26] OWL ontology consists of instances, properties and concept (or classes). An example of component ontology is shown in figure 4. It shows instances of classes or concepts. The properties are relationships between two entities:

- Object property: from concept/instance to concept instance
- Data type property: from concept/instance to RDF

4. The Architecture of Semantic Search For Reusable Software Components

A modular design is used for structuring the architecture of semantic search for reuse software components. if the is a need to extend knowledge base, it will affect only the knowledge level is affected not the process level. Therefore, it will characterize the system as adaptive, consistent, and maintainable system. The Semantic Search Application Layers for Reuse Software Components, as it is shown pictorially in figure 4, involves the following:

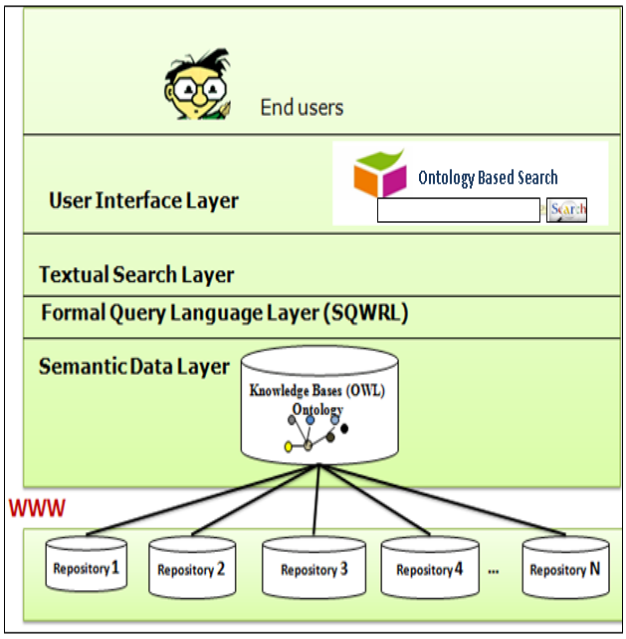

Fig. 4 Semantic Search Application Layers for Reuse Software Components

4.1 User Interface Agent

The interaction with the user is done through *Interface agent*. It depends on user model, user ontologies, and natural language processing for the user's query. The user inserts a free form natural language query, without having to know about the vocabulary, semantic data or the

structure of the ontology. The query specifies the required component that is needed to be reused in new software.
The query is preprocessed to understand what is the user's intend (to find out the semantic meaning of his query), and to identify what components he is looking for. All uppercase words become lowercase. The query is decomposed into tokens and the meaningless tokens are removed such as: articles, pronouns, prepositions, conjunctions, etc. and identify only the important tokens (terms). *Interface agent* then sends a group of terms or ontologies to *ontology* agent. The resultant terms are represented as input to the *Term Extracting* Agent.

4.2 Textual Search Layer (the *Term Extracting* Agent)

This process identifies ontology-based concepts for user's query terms. It determines the semantic meaning of the extracted terms, which will support the semantic search engine. The terms in the query are stored in a query array, and is defined as: $Q = \{T_1, T_2, T_3, ...T_n\}$. For each term T_i in Q, all its concepts and attributes of ontology are determined to replace the user's query terms. There may be more than one concept or attribute for each term. The output concept list, $C_{Ti} = \{C_1, C_2, ..., C_m\}$; will be use as entries to the semantic searching agent. Now, the list of query terms is expanded with words that represent the semantic meaning.

4.3 Formal Query Language Layer (SQWRL)

It transforms the concept list, $C_{Ti} = \{C_1, C_2, ..., C_m\}$; into formal structure of the concepts and the relationships among the concepts of different terms of the query. It reflects the semantic meaning of the user's query, by associating each term with content-based and ontology-based features.

component (?c) ∧ used_ for_ developing (?pl, ?c) ∧ sameAs(?pl, java) ∧ provided_by (?pr, ?c) ∧ sameAs(?pr, barcode) → sqwrl: select(?c)

They are considered as the input to the Ontology-Based Semantic Search Engine to search a specified meta-data of the repositories of reusable software components, to find out the semantic relationships and to retrieve the relevant components from its beneath layer.

4.4 Semantic Data Layer (Ontologies)

The large amount of meta-data is used to describe the reusable software components in the distributed and heterogeneous repositories. The structure of information varies in different repositories, and in order to unify the structure of these repositories, we find that software components have common features and related well-defined ontologies that are understood by people and computers. Knowledge base is constructed for reusable components specification and the Web Ontology Language (OWL) is used to add semantic to data. Therefore, knowledge base is transformed into OWL formatted documents to represent the data representation model. Intelligent agents are enabled to share ontologies that provide the vocabulary to facilitate communications among the agents. Agents also can search the semantic knowledge base using automated rule-based reasoning process. Then, ontological knowledge base; for components specification, with OWL format can be queried and reasoned over. This process extracts information and identifies the classes, properties and instances using a pre-defined template to produce a filled or semi-filled template. All the extracted features compose the basic constructs of classes, properties and instances and predicates of the searched semantic meta-data (semantic data repository). These basic constructs and predicates can be combined via Boolean operators (i.e., AND, OR, NOT) to create complex predicates. It represents the global properties and instances of the software components.

5 Application Case Study

This process developed using multi-agent system environment, composed of multiple interacting intelligent agents.

Step1: Insert the user's natural language query to the query engine interface. The user's query specifies the required software component; such as: "I want a component that is written in java and provide barcode generation for a given String". The query is preprocessed by decomposing the query into its tokens and removing the words that have no sense, as it is shown in figure 5. The natural language of user's query is transformed into a group of terms in form of terms vector that is extracted from user's query. Therefore the terms vector = {JAVA, BARCODE ,STRING}.

Step 2: In (SQWRL), semantic matches of more than one keywords are identified and combined together using the OWL-based formal language to reflect the semantic of the user's query, as it is shown in shown in Figure 6.

component (?c) ∧ used_ for_ developing (?pl,?c) ∧ sameAs(?pl,java) ∧ provided_by (?pr,?c) ∧ sameAs(?pr,barcode) → sqwrl:select(?c)

The goal of this process is to find out the semantic meaning of each important keyword but there may be more than one semantic matching for one

keyword entry. Therefore, we use semantic entities labels that represent the meaning.

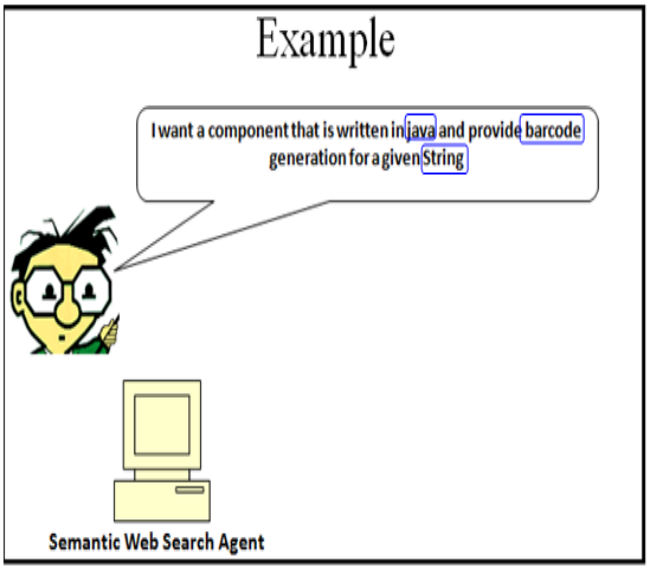

Fig. 5 Query Filtering using Keyword Matching

Step 3: The SQWRL query is executed by querying the back-end semantic representation of reuse software repositories. SQWRL uses many Boolean-valued features for software components. 'Information Retrieval' agent also uses the vector model that employs similarity measures to find the relevant reusable component to a query, and in response a list of ranked Web sites is generated and the knowledge about the desired software component(s) is identified.

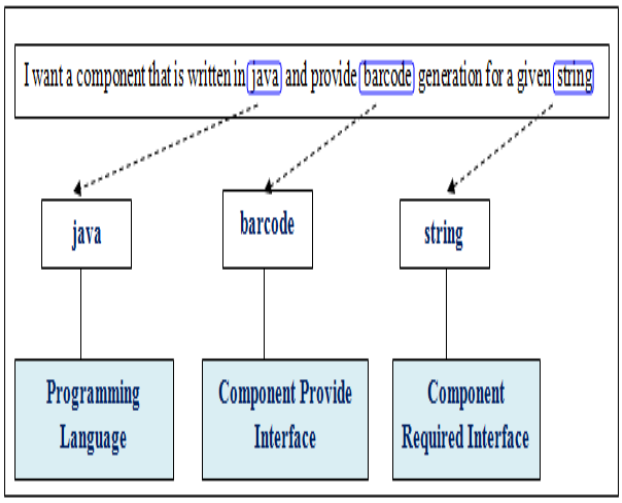

Fig. 6 Extracting Keywords and Assigning Keyword Semantics

Step4: The search engine tries to understand the concept of the user's query and find semantically the reusable component with the highest rank that is fall into pre-defined , as a measure of importance, and suit the user's query.

$$\text{erm frequency} = \frac{\text{Freq}_{ij}}{\text{Max}_T \text{Freq}_{ij}} \quad (1)$$

where:

Freq_{ij} : is the number of occurrences of term t_i in document d_j

Max_T: is the number of documents in the system

Step 5: Retrieving the relevant software components are evaluated by: precision and recall measures. Recall represents the percentage of relevant Software components that are retrieved from the component repositories. Precision measure represents the percentage of relevant software components that are in the retrieved software components. Their equations are:

$$\textbf{Precision} = \frac{|\textbf{Relevant} \cap \textbf{Retrieved}|}{|\textbf{Retrieved}|} \quad (2)$$

$$\textbf{Recall} = \frac{|\textbf{Relevant} \cap \textbf{Retrieved}|}{|\textbf{Relevant}|} \quad (3)$$

Where 'Relevant' represents the set of software components in the component repositories that are relevant to a particular query and 'Retrieved' is the set of the retrieved software components for user's query. The maximum values are preferred for both recall and precision measures.

F_β measure: is used to combine precision and recall and is defined as:

$$\mathbf{F_\beta} = \frac{\left((\beta^2+1)\,\textbf{Recall} * \textbf{Precision}\right)}{\left((\beta^2 * \textbf{Precision}) + \textbf{Recall}\right)} \quad (4)$$

Where: $\beta \in [0, 5]$. When $\beta = 5$, precision and recall are given the same weight in the F_β -measure rating Web software components on a scale of 0.0 to 4.0. This numeric scale is then mapped into five categories (Very Good, Good, Appropriate, Inappropriate, and Very Inappropriate) in order to simplify the retrieving components of labeling Web software components.

6. CONCLUSION

In this paper, we proposed a prototype of semantic search of reusable software component, which is employed using multi-agent system. An ontology-based model is embedded in a multi-agent system to semantically search the software component in an opening, distributed and heterogeneous environment. It empowers the semantic

search and increases the recall and precision of a query. The technique depends on analyzing the user's query to find out user's intent, extracting the meaning of terms from their associated ontologies, comparing and ranking the concepts, and aggregating the information that are provided by the different resources. The components are categorized into various classes, programming language, interfaces, services, ..etc. The information of reusable components are semantically specified and structured, as a universal content representation, in the repositories. Therefore, the search engines and computers can understand the provided information. The distributed repositories ensures different levels of content to cover the usage and intents of different users. Intelligent agents are enabled to share ontologies to enhance the coordination and communication among agents and to link the data of components from distributed heterogeneous repositories. The prototype has been tested on small-scale databases with promising results.

Acknowledgment

The authors are grateful to the applied science private university, Amman-Jordan, for the partial financial support granted to cover the publication fee of this research article.

References

[1] L., Chen and K. Sycara, WebMate : A Personal Agent for Browsing and Searching. Proceeding of AGENTS '98 Proceedings of the second international conference on Autonomous agents, ACM, NY, USA, 1998. Available at: http://www.cs.cmu.edu/~softagents/papers/aa98-webmate.pdf. https://www.ri.cmu.edu/pub_files/pub1/chen_liren_1998_1/chen_liren_1998_1.pdf.

[2] J. Hsiang and H. Tu, Personalized Web Retrieval: Three Agents for Retrieving Web Information, Multi-agent Platforms Lecture Notes in Computer Science, Vol. 1599/1999, PP. 118-132, 1999. Available at: http://link.springer.com/chapter/10.1007/3-540-48826-X_9#page-1

[3] P. N. Gupta, P Singh., P. P. Singh, P. K. Singh, and D. Sinha, A Novel Architecture of Ontology based Semantic Search Engine, International Journal of Science and Technology, Vol. 1, No. 12, Dec. 2012.

[4] M. A. Naeem, N. Asif, A Web Smart Space Framework for Intelligent Search Engines, International Journal of Emerging Sciences ISSN: 2222-4254 1(1) April 2011. Available at: http://cogprints.org/7294/1/4254111.pdf.

[5] G. Sudeepthi1, G. Anuradha, S. P. Babu, A Survey on Semantic Web Search Engine, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 2, No 1, March 2012 Available at: http://ijcsi.org/papers/IJCSI-9-2-1-241-245.pdf

[6] J.R. Jenkins, Where Angels Fear To Tread: The Problems Of Keyword Search In E-Discovery, MLIS, FTI Consulting, Inc. FTI Technology is a business of FTI Consulting, 2010. Available at: http://www.ftitechnology.com/doc/White-Papers/whitepaper-ediscovery-keyword-search-2009.pdf.

[7] G. Madhu, A. Govardhan, and T.V. Rajinikanth, Intelligent Semantic Web Search Engines: A Brief Survey, International journal of Web & Semantic Technology (IJWesT) Vol.2, No.1, January 2011. Available at: http://airccse.org/journal/ijwest/papers/0111ijwest03.pdf.

[8] W. Webber, Evaluating the Effectiveness of Keyword Search. Available at: http://ftp.research.microsoft.com/pub/debull/a10mar/webber-paper.pdf

[9] S. K. Malik, N. Prakashm S. Marwaha , Role of Search Engines in Intelligent Information Retrieval on Web, Proceedings of the 2nd National Conference; INDIACom-2008.

[10] G. Madhu1, A. Govardhan, and T. K. Rajinikanth, Intelligent Semantic Web Search Engines: A Brief Survey, International journal of Web & Semantic Technology (IJWesT), Vol.2, No.1, January 2011. http://web.ebscohost.com/ehost/pdfviewer/pdfviewer?vid=4&hid=14&sid=bff4b941-1957-4590-aa38-a57e7cd6b734%40sessionmgr10

[11] V. Dhingra, and K. K. Bhatia, Towards Intelligent Information Retrieval on Web, International Journal on Computer Science and Engineering (IJCSE), Vol. 3, No. 4, PP. 1721- 1726, Apr 2011. Available at: http://www.enggjournals.com/ijcse/doc/IJCSE11-03-04-144.pdf

[12] O. Z. Akbari, A survey of agent-oriented software engineering paradigm: Towards its industrial acceptance, Journal of Computer Engineering Research Vol. 1, No. 2, PP. 14 - 28, April 2010. Available at: http://www.academicjournals.org/article/article1379926489_Akbari.pdf

[13] S. Bhat, N. S. Sidnal, B. S. Malashetty, and S. Manvi, Intelligent Scheduling in Healthcare Domain, International Journal of Computer Science Issues (IJCSI), Vol. 8, Issue 5, No. 3, September 2011.

[14] S.A Deloach, Analysis and Design using MaSE and agent Tool, In 12th Midwest Artificial Intelligence and Cognitive Science Conference (MAICS 2001), Miami University, Oxford, Ohio, 2001.

[15] V. M. B. Werneck, and R. M. E. M. Costa, and L. M. Cysneiros, Modeling Multi-Agent System using Different Methodologies, Modeling, Interactions, Simulations and Case Studies, Dr. Faisal Alkhateeb (Ed.)m InTechm 2011. Available at: http://cdn.intechopen.com/pdfs-wm/14487.pdf

[16] S. A. DeLoach, E. T. Matson, and Y. Li. Exploiting Agent Oriented Software Engineering in the Design of a Cooperative Robotics Search and Rescue System, *The International Journal of Pattern Recognition and Artificial Intelligence*, Vol. 17, No. 5, August 2003, PP.1-19.

[17] S. Maalal, and M. Addou, A new approach of designing Multi-Agent Systems With a practical sample, International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 2, No. 11, 2011. Available at: http://arxiv.org/ftp/arxiv/papers/1204/1204.1581.pdf

[18] http://www.fipa.org/
[19] F. Bellifemine, G. Caire, D. Greenwood, Developing Multi-Agent Systems with JADE, John Wiley & Sons Ltd., UK, 2007.
[20] M. Laclav´ık, Z. Balogh, M. Bab´ık, and L. Hluchy, AGENTOWL: Semantic Knowledge Model and Agent Architecture, Computing and Informatics, Vol. 25, 419–437, 2006. Available at: http://agentowl.sourceforge.net/publications/AgentOWL_cai2006.pdf.
[21] S. A. DeLoach. Multi-agent Systems Engineering of Organization-based Multi-agent Systems. 4th International Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS'05). May 15-16, 2005, St. Louis, MO. Springer LNCS, Vol. 3914, Apr 2006, PP. 109 - 125. Available at: http://people.cis.ksu.edu/~sdeloach/publications/Conference/sdeloach-selmas05.pdf
[22] S. A. DeLoach, M. F. Wood and C. H. Sparkman, Multi-agent Systems Engineering, The International Journal of Software Engineering and Knowledge Engineering, Vol. 11, No. 3, June 2001.
[23] R. Choren, and C. Lucena, Modeling Multi-Agent Systems with A Note, Software System Model, Vol. 4, 2005, PP. 199–208.
[24] H. Knublauch, M.A. Musen, and N. F. Noy: Tutorial: Creating Semantic Web (OWL) Ontologies with Prot´eg´e. Web Conference (ISWC2003), Sanibel Island, Florida, USA, 2003. Available at: http://iswc2003.semanticweb.org/pdf/Protege-OWL-Tutorial-ISWC03.pdf
[25] M. Wilson and B. Mathew, *The Semantic Web: prospects and challenges,"* IEEE2006. Available at: http://ieeexplore.ieee.org/iel5/11087/35308/01678469.pdf?isnumber=35308&arnumber=1678469
[26] P. J. Radadiya, K.R. Rakholiya, and D. R. Kathiriya, Intelligent Semantic Web Search Engine, Computer Science, Vol. 1, Issue 7, Dec. 2012, PP. 34- 35. Available at: http://theglobaljournals.com/ijsr/file.php?val=December_2012_1354294962_26e08_13.pdf

Dr. Nedhal A. Al-Saiyd. She got her B.Sc. degree in Computer Science from University of Mosul-Iraq in 1981, M.Sc. and PhD degrees from University of Technology, Baghdad-Iraq in 1989 and 2000 respectively. She is an Associate Prof. at Computer Science Dept., Faculty of Information Technology, in the Applied Science University, Amman, Jordan. She has got more than 24 years of teaching experience. Her research interests include Software Engineering, Ontology Engineering, Intelligent Systems, User Authentication, Security and Speech Processing.

Dr. Muna F. Al-sammarai. She obtained her B.Sc. degree from Al-Mansour University Baghdad-Iraq in 1994, M.Sc. and PhD degrees from University of Technology, Baghdad-Iraq in 1997 and 2002 respectively. She is an Associated Prof. at MIS Dept., Faculty of Faculty of Economics and Administrative Sciences in Al-Zaytoonah University, Amman, Jordan. She has got more than 17 years of teaching experience. Her research interests include: Software Engineering, Image Processing, Ontology Engineering and Intelligent Systems.

Dr. Intisar A. Al-Sayed. She got her B.Sc. from University of Technology, Baghdad-Iraq, in 1986, M.Sc. degree from Al-Nahreen University, Baghdad-Iraq, in 1993, and PhD from University of Technology, Baghdad-Iraq in 2000. She is an Associated Prof. at Computer Network Dept., Faculty of Information Technology, in Al-Isra University, Amman Jordan. She has got more than 21 years of teaching experience. Her research interests include: Genetic Algorithms, Computer Engineering, Software Engineering, Ontology Engineering, and Intelligent Systems.