2011 International Conference on Information and Intelligent Computing
IPCSIT vol.18 (2011) © (2011) IACSIT Press, Singapore

A Cloud Portal with a Cloud Service Search Engine

Jaeyong Kang and Kwang Mong Sim

Multiagent and Cloud Computing Systems Lab., Department of Information and Communication, Gwangju Institute of Science and Technology (GIST), Gwangju, Korea

e-mail: kjysmu@gist.ac.kr, e-mail: prof_sim_2002@yahoo.com

Abstract. Cloud computing has become another buzzword after Web 2.0. Many companies, such as Amazon, Google, Microsoft and so on, accelerate their paces in developing Cloud computing systems. However, there is no study that focuses on search engine and web portal for Cloud computing system. Hence, this paper presents Cloud portal with various service categories and Cloud service search engine for Cloud computing system. In Cloud service search engine, we use Cloud ontology to semantically define the relationship among Cloud services. It contains a set of Cloud concepts, individuals of those concepts, and the relationship among those individuals. It is used for determining the similarity among Cloud services with three kinds of reasoning methods (1) concept similarity reasoning, (2) object property similarity reasoning, and (3) datatype property similarity reasoning. The interface and features of the Cloud portal are briefly presented. Finally a proof-of-concept-example demonstrates the features and functionalities of our Cloud portal and Cloud search engine.

Keywords: Cloud Computing; Cloud Portal; Search Engines; Cloud Ontology

1. Introduction

With the emergence of cloud computing, any organization that requires information technology (IT) infrastructure has experienced a significant paradigm shift. Instead of paying and maintaining expensive IT software and hardware, it is now possible to rent required infrastructures at cheap rates. Cloud computing is the collection of virtualization, Web services [1],[2], Service Level Agreements and highly scalable compute servers. The end result is distributed systems that offer over the Internet resources as scalable, pay-per-use services. Although the variety of services and resources are offered in Clouds, there is no portal site and search mechanism that are specialized for Cloud computing. In this paper, we present the Cloud portal with a Cloud service search engine as a main feature and various other features. In our Cloud service search engine, users can specify the type of Cloud services. Furthermore, users can specify three kinds of requirements: 1) functional requirement (category of service), 2) technical requirement (OS, CPU clock, memory, disk capacity and etc.) and 3) cost requirement (maximum acceptable price and timeslot range) as input parameters. Once users send those input parameters to the Cloud service search engine, it returns the list of Cloud services ordered by aggregated similarity (service utility), which is determined by three kinds of similarity reasoning methods by consulting a Cloud ontology. The Cloud ontology provides meta information which describes data semantics. It contains a set of Cloud concepts and their individuals as well as the relationships between individuals. It is used for determining the similarity among Cloud services using three kinds of similarity reasoning methods: (1) concept similarity reasoning, (2) object property similarity reasoning, and (3) datatype property similarity reasoning. With these three kinds of similarity reasoning methods, we demonstrated that in our previous works [7], [8], [9], and [10] that our Cloud service search engine can provide an efficient search mechanism to discover appropriate Cloud services.

This paper is organized as follows. In Section 2, the Cloud ontology and the three kinds of similarity reasoning methods are described. Section 3 presents a Cloud portal and the proof-of-the-concept example of Cloud service search engine, and finally, conclusion and future work are illustrated in Section 4.

2. Similarity Reasoning with Cloud Ontology

2.1. Cloud ontology

Ontology provides a formal, shared specification of concepts, their relationships, and other realities of some domain, which can reduce or eliminate confusion of terminologies, enable computers to process domain knowledge more precisely and conveniently. Since 1990s, ontology has developed from AI field to computer field, and becomes a popular research topic in various communities such as knowledge engineering, natural language processing, intelligent information integration and knowledge management, etc. The existing famous ontologies are CYC [3], KIF [4] and Ontolingua [5]. In a Cloud ontology, the hierarchical relations of Cloud concepts are shown. For instance, the concept "CloudSystem" has five different children nodes (IaaS, PaaS, SaaS, CaaS, and DaaS). By consulting a Cloud ontology, similarity reasoning, which are explained in next sub-section, is carried out.

2.2. Similarity reasoning

The similarity between the request from users and the advertisements from providers can be determined by 1) concept similarity reasoning, 2) object property similarity reasoning, and 3) datatype property similarity reasoning, as follows:

$$Sim(p,q) = \alpha Sim_{con}(p,q) + \beta Sim_{obj}(p,q) + (1-\alpha-\beta) Sim_{data}(p,q) \tag{1}$$

where α, β, and $(1-\alpha-\beta)$ are the weights of each clause, and the range of the evaluated value is $0 \le Sim(p,q) \le 1$.

1) Concept similarity reasoning: The concept similarity can be determined as follows [6]:

$$Sim_{\text{con}}(p,q) = \frac{|Super(\text{P}) \cap Super(\text{Q})|}{|Super(\text{P})|} \tag{2}$$

where *P* and *Q* are the most specific concepts that individuals *p* and *q* belong to, respectively, and $Super(\text{P})$ (respectively, $Super(Q)$) is a set of all reachable super-concepts from concept *P* (respectively, concept *Q*).

2) Object property similarity reasoning: The object property similarity can be determined as follows:

$$Sim_{\text{obj}}(p,q) = \frac{\sum_{(x,y)\in U} Sim(x,y)}{|O(p)|},$$
$$U = \{(x,y) \mid (p,r,x) \in O(p), (q,r,y) \in O(q)\} \tag{3}$$

where $O(p)$ is a set of triples that contain the object properties of the individual *p*, and *p* is the subject. Each triple consists of (1) the subject, (2) a predicate, and (3) an object value to express the ontology. For instance, if we want to express the individual '*Provider1*', which has the property '*hasOS*', and its value '*Devian*', we can simply express using a triple as '(*Provider1, hasOS, Devian*)'. *U* is the set of object values that has the common predicate *r* of individuals *p* and *q* in each triple $O(p)$ and $O(q)$ respectively. For instance, the common predicates, which have an object value of *Provider1* and *Provider2* in Table 2 are *hasCPU* and *hasOS*. Hence, the set of object values of the common properties of individuals *p* and *q* is *U* = {(*CPU1, CPU2*), (*Devian, WindowsVista*)}.

3) Datatype property similarity reasoning: The datatype property similarity can be determined as follows:

$$
\begin{aligned}
&Sim_{\text{data}}(p,q) = \frac{\sum_{(x,y,r)\in V} Comp(x,y,r)}{|\,D(p)\,|},\\
&V = \{(x,y,r)\,|\,(p,r,x)\in D(p),(q,r,y)\in D(q)\},\\
&Comp(x,y,r) = 1 - \frac{|\,x-y\,|}{MAX_{\text{distance}}(x,r)},\\
&MAX_{\text{distance}}(x,r) = \max_{i\in I(r)}(|\,x-i\,|),\\
&I(r) = \{i\,|\,(s,r,i)\in \text{Ontology}\}
\end{aligned}
\tag{4}
$$

where $D(p)$ is a set of triples that contains the datatype properties of the individual p and p is the subject. Each triple consists of (1) the subject, (2) a predicate, and (3) a datatype value to express the ontology. For instance, if we want to express the individual '*Provider2*', which has the property '*hasNetworkLatency*', and its value '*200*', we can simply express using a triple as '(*Provider2, hasNetworkLatency, 200*)'. V is a set of datatype values that has the common predicate r of individuals p and q in each triple $D(p)$ and $D(q)$, respectively. For instance, the common predicates that have a datatype value between *Provider1* and *Provider2* in Table 2 are *hasMemory*, *hasCache*, and *hasStorage*. Hence, the set of datatype values of the common properties of individuals p and q is V = {(*5000, 12000, hasMemory*), (*8, 8, hasCache*), (*25000, 30000, hasStorage*)}. With each of the elements in V, *Comp(x,y,r)*, which is the similarity between datatype values x and y over predicate r is determined. For instance, the example used to calculate *Comp(5000, 12000, hasMemory)* is shown in Fig. 4.

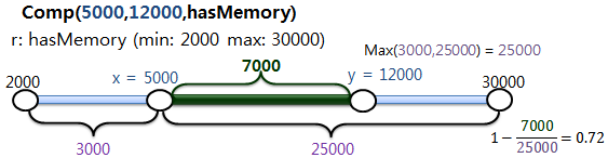

Figure 1. Example to calculate Comp(5000,12000,hasMemory)

The predicate r is *hasMemory* with the range of 200 to 30000. Based on the value x, the maximum reachable distance with the range of 2000 to 30000 is decided. With the maximum reachable distance and the distance between x and y, the similarity between datatype values 5000 and 12000 over the predicate *hasMemory*, *Comp(5000, 12000, hasMemory),* can be determined.

With the above three kinds of similarity reasoning methods, we can determine the similarity between two individuals. For instance, let *Provider1* and *Provider2* be individuals in the concepts *PaaS* and *CaaS*, respectively, in a Cloud ontology representing providers. Also, we assume that *Provider1* and *Provider2* have some properties. Table 1 shows the concepts and their individuals, and Table 2 shows those individuals and their properties.

TABLE I. EXAMPLE OF CONCEPTS AND THEIR INDIVIDUALS

Concept	Individual
InfrastructureAsAService (PaaS)	Provider1
CommunicationAsAService (CaaS)	Provider2
IntelCPU	CPU1
AMDCPU	CPU2
RelationalDBMS	Oracle
WindowsSeries	WindowsVista
LinuxSeries	Devian

TABLE II. EXAMPLE OF INDIVIDUALS AND THEIR TYPE AND VALUE

Individual	Property Name (Type)	Value
Provider1	hasCPU (Object)	CPU1
Provider1	hasOS (Object)	Devian

Provider1	hasDBMS (Object)	Oracle
Provider1	hasMemory (Datatype)	5000
Provider1	hasCache (Datatype)	8
Provider1	hasStorage (Datatype)	25000
CPU1	hasSpeed (Datatype)	2.8
Provider2	hasCPU (Object)	CPU2
Provider2	hasOS (Object)	WindowsVista
Provider2	hasMemory (Datatype)	12000
Provider2	hasCache (Datatype)	8
Provider2	hasStorage (Datatype)	30000
Provider2	hasNetworkBandwidth (Datatype)	500
Provider2	hasNetworkLatency (Datatype)	200
CPU2	hasSpeed (Datatype)	3.4

To calculate $Sim_{con}(Provider1, Provider2)$, we know that $|Super(PaaS)|=3$ and $|Super(PaaS) \cap Super(CaaS)|=2$ (see Fig. 2). Hence, the concept similarity is $Sim_{con}(Provider1, Provider2) = 2/3$.

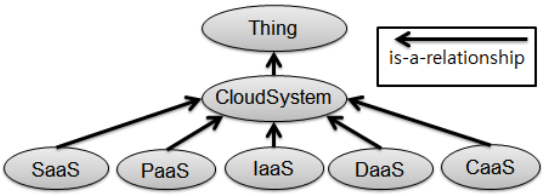

Figure 2. Relation in terms of Cloud

To calculate $Sim_{obj}(Provider1, Provider2)$, we know that $|O(Provider1)|=3$, and the set of the values of common object properties of *Provider1* and *Provider2* is $U = \{(CPU1, CPU2), (Devian, WindowsVista)\}$. The similarity of each of the members in U can be calculated by a recursive procedure. Using the three kinds of similarity reasoning methods that have been previously explained and will be explained, we can give the similarity values of each member of U, which are *Sim(CPU1, CPU2)* = 0.64, *Sim(Devian, WindowsVista)* = 0.82. Hence, the calculated object similarity is as follows:

$$Sim_{obj}(Provider1, Provider2) = \frac{0.64 + 0.82}{3} = 0.49 \quad .$$

For the datatype property similarity, we know that $|D(Provider1)|=3$ and the set of the values of common datatype property between two individuals, which is $V = \{(5000, 12000, hasMemory), (8, 8, hasCache), (25000, 30000, hasStorage)\}$. We have to calculate numerical similarity for all the members of the set V. The numerical similarity for the first member of the set V is as follows:

$$Comp(5000,12000,hasMemory) = 1 - \frac{|5000-12000|}{\max(|5000-2000|,|5000-30000|)} = 0.72 \quad .$$

with a memory range from 2000 to 30000. The numerical similarity for the other members can be calculated in the same way, which are *Comp(8, 8, hasCache)* = 0.96, *Comp(25000, 30000, hasStorage)* = 0.94. Hence, the datatype property similarity can be calculated as follows:

$$Sim_{data}(a,b) = \frac{0.72 + 0.96 + 0.94}{3} = 0.87$$

We assume that α and β are 1/3 each, the same weight. Finally, the similarity between two individuals, *Provider1* and *Provider2*, can be calculated as follows:

$$Sim(Provider1, Provider2) = \frac{1}{3}0.67 + \frac{1}{3}0.49 + \frac{1}{3}0.87 = 0.68 \quad .$$

3. Proof-Of-Concepts Example with Cloud Portal

The main page of Cloud portal is shown in Fig 3. The features of Cloud portal are as follows.

Figure 3. The main page of Cloudle portal.

The main page of Cloud portal contains the latest news about Cloud computing and the latest updates of Cloud portal as well as each menu of four kinds of services: 1) Cloud search, 2) News & Articles, 3) Books, and 4) Events. In Cloud search, an example will be provided to illustrate the major functionalities of the Cloud service search engine in Cloud portal.

4) Cloud search

In the Cloud search page, users can search the Cloud services based on their service requirements. At first, a user must select the type of the Cloud service to narrow the search range to find appropriate Cloud services. Keywords, when needed, can be added as many as a user wants to sort out the services. When user knows the name of the service to be searched, a service name can be assigned to limit it for only the services with the assigned name. In the technical requirements field, the user can specify information such as OS, CPU name, CPU clock, RAM, and HDD. In the cost and time requirements field, user can assign the maximum price that is acceptable for the service, and the timeslot that the service should be available. When a user sends a query to the Cloud search engine, it returns the list of information of Cloud services ordered by similarity in an increasing order. A user can select one of the Cloud services from a list of services. Finally, the user can pay and get the Cloud service. An example to find Cloud services is described as follows.

Step 0: The screen in Fig. 4 shows the service registration query of a Cloud provider. A Cloud provider can specify the information of Service name, Web address, Keywords, Technical Information (e.g., OS, CPU name, etc.), and Cost and Time Information (e.g., Min Price, several timeslots). By specifying these parameters, Cloud providers can register their services into the database of our Cloud service search engine.

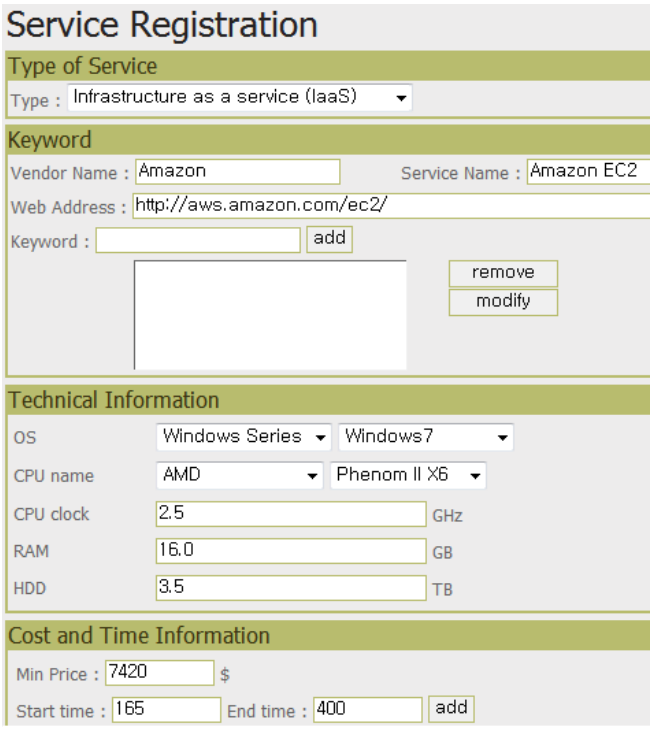

Figure 4. The service registration page

Step 1: The screen in Fig. 5 shows the input query of a user which contains a service type (e.g., "Infrastructure as a service (IaaS)"), technical requirements (e.g., OS = "Windows7", CPU name = "Phenom II X6", CPU clock = "3.0", RAM = "3.0", HDD = "4.0"), and cost and time requirements (e.g., Max price = "3000", Start time = "200", End time = "600", Price weight = "0.5", Time weight = "0.5").

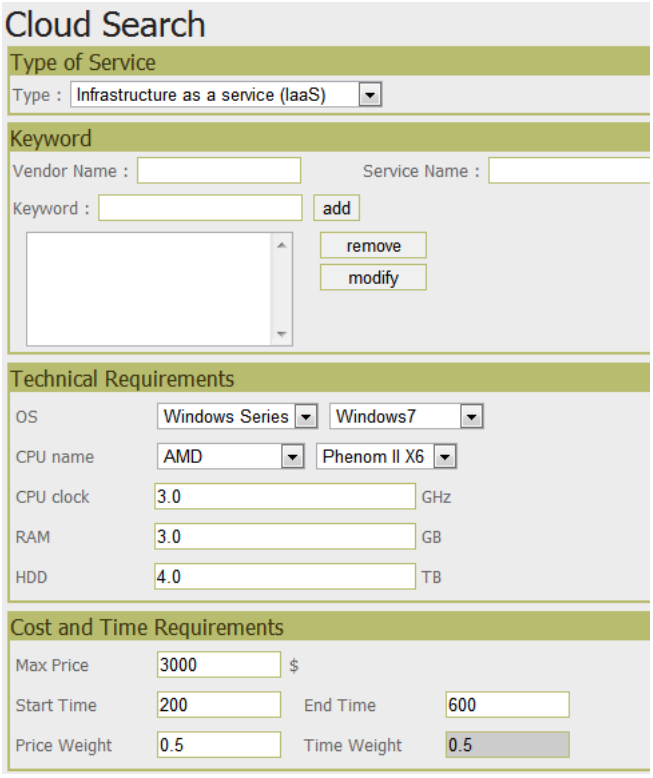

Figure 5. Cloud service search engine

Step 2: Once user sends the input query to our system, by referring to the value of service type (e.g., service type = "IaaS"), the list of registered services in the database is filtered out as shown in Fig 6. With the technical requirements input parameters, the similarity of each filtered service can be determined using (1) similarity reasoning, (2) object property similarity reasoning, and (3) datatype property similarity reasoning by

consulting the Cloud ontology. With the Cost and Time requirement, the price and timeslot utility can be determined.

```
1 select * from services where service_type='IaaS';
```

id	service_name	service_type	os_category	os_name
0	Zenith Savior v9	IaaS	Unix series	Solaris
2	Massive Worker 2	IaaS	Moblie OS	iPhoneOS
9	Iconic Cruiser v8	IaaS	Moblie OS	WindowsMobileOS
13	Yota Builder v10	IaaS	Unix series	AIX
16	Relational Generator v3	IaaS	Unix series	AIX
17	Tera Mark 3	IaaS	Unix series	Solaris
19	Exa Power v8	IaaS	Moblie OS	iPhoneOS
32	Smart Worker v7	IaaS	Unix series	AIX
38	Zenith Savior v10	IaaS	Moblie OS	WindowsMobileOS
39	Absolute Generator 500	IaaS	Moblie OS	iPhoneOS
47	Massive Builder 400	IaaS	Unix series	Mandrake
...		...		

Figure 6. The list of services after filtering by service type

Step 3: After the similarity of each service is determined and rated, the list of Cloud services is displayed to the user as shown in Fig 7. From the result, we can see that the number of retrieved Cloud service is 82, and the Cloud service "Amazon EC2" has the highest similarity (e.g., similarity = 0.9858) among all other Cloud services. (see comparison table III).

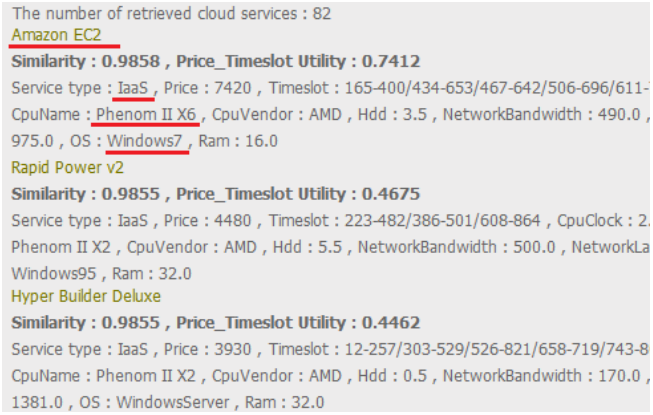

Figure 7. The list of searched Cloud services.

TABLE III. COMPARISON USER QUERY WITH SEARCH RESULT

Information	User query	Search result
Service type	IaaS	IaaS
OS	Windows7	Windows7
CPU name	Phenom II 6	Phenom II 6
CPU clock	3.0	2.5
RAM	3.0	16.0
HDD	4.0	3.5

5) News & Articles

In the News & Articles page, users can see the list of news and articles related to the Cloud computing. It contains the title of news and several beginning sentences.

6) Books

In the Books menu, the list of books that is relevant to Cloud computing is shown. It contains the information of title, authors, and price of the books.

7) Events

In the Event menu, the list of conferences and events are shown. It contains the information of the name, date, submission deadline, venue and country of the conference.

4. Conclusion and Future Works

This paper presented a Cloud portal that contains a Cloud service search engine. While the previous work on Cloud service search engine which is called "Cloudle" was presented in [7],[8], and [9], and [10] compared the performance of Cloudle with two Cloud ontologies, the Cloud service search engine in this paper is introduced as the one of features in Cloud portal. The contributions of this work is to build the Cloud portal that provides (1) Cloud service search engine with three kinds of reasoning methods, (2) latest news and articles regarding Cloud computing, (3) Books about Cloud computing, and (4) Events such as upcoming Cloud computing conferences and workshops. From the proof-of-the-concept example, we demonstrate that Cloud service search engine can potentially assist users in finding Cloud services that are closely matched with users' requirements. To the best of the authors' knowledge, this work is the first attempt in building a Cloud portal with a Cloud service search engine and other various features. At present, there are few Cloud service providers and there may not be many Cloud services available. However, when Cloud computing is more widely used in the near future, our Cloud portal can be helpful tool for Cloud users for finding Cloud services under their specific preference and utilizing other various features.

5. Acknowledgment

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MEST) (KRF-2009-220-D00092) and the DASAN International Faculty Fund (project code: 140316).

6. References

[1] D. Booth, et al. Web Services Architecture. Accessed 18 Feburary 2009, http://www.w3.org/TR/ws-arch/.

[2] F. Curbera et. al. Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI. IEEE Internet Computing, V. 6, I. 2, 2002.

[3] Douglas B. Lenat and R. Guha: Building large knowledge-based systems. Representation and inference in the Cyc project, Addison-Wesley, Reading, Massachusetts, 1990.

[4] M. R. Genesereth: Knowledge Interchange Format. In Proceedings of the Second International Conference on the Principles of Knowledge Representation and Reasoning (KR- 91), J. Allenet al., (eds), Morgan Kaufman Publishers, 1991, pp 238-249.

[5] Thomas R. Gruber: A Translation Approach to Portable Ontology Specifications, Knowledge Acquisition, 5:199-220, 1993.

[6] Troels Andreasen, Henrik Bulskov, and Rasmus Kanppe, "From Ontology over Similarity to Query Evaluation", 2nd International Conference on Ontologies, Databases, and Applications of Semantics for Large Scale Information Systems (ODBASE), 3-7 November 2003, Catania, Sicily, Italy, 2003.

[7] Jaeyong Kang and Kwang Mong Sim, "Cloudle: An Agent-based Cloud Search Engine that Consults a Cloud Ontology," Cloud Computing and Virtualization Conference (CCV 2010), 2010.

[8] Jaeyong Kang and Kwang Mong Sim, "Cloudle: A Multi-criteria Cloud Service Search Engine," IEEE Asia-Pacific Service Computing Conference (APSCC 2010), 2010

[9] Jaeyong Kang and Kwang Mong Sim, "Cloudle: An Ontology-enhanced Cloud Service Search Engine," The 1st International Workshop on Cloud Information System Engineering (CISE 2010), 2010

[10] Jaeyong Kang and Kwang Mong Sim, "Ontology and Search Engine for Cloud Computing System," International Conference on System Science and Engineering (ICSSE 2011), 2011