REALIZING THE NEED FOR SIMILARITY BASED REASONING OF CLOUD SERVICE DISCOVERY

S. BHAMA
Assistant Professor, Dept. of C&IS,
PSG College of Technology, Peelamedu,
Coimbatore, Tamil Nadu-641004, India.
bhamajayabal@gmail.com

DR. G. R. KARPAGAM
Professor, Dept. of CSE,
PSG College of Technology, Peelamedu,
Coimbatore, Tamil Nadu-641004, India.
grkrm@cse.psgtech.ac.in

Abstract:
With the growing abundance of information on the web, it becomes the need of the hour to enrich data with semantics that can be understood and processed by machines. Currently, much of the effort in the area of semantics is focused on the representation of semantic data and its reasoning, which is the processing of semantic information associated with that data. This paper aims at realizing the need for similarity based reasoning of cloud service discovery. It forms a basic requirement of a cloud client to discover the most appropriate cloud service from the list of available services published by service providers. Cloud ontology provides a set of concepts, individuals and relationships among them. The similarity among cloud services can be determined from the semantic similarity of concepts and hence the relevant service can be retrieved.

Keywords: Cloud service discovery, Ontology, Semantic similarity reasoning

1. Introduction

In today's evolutionary world, the clients are always looking towards their desired, beneficial and appropriate services from the providers. Cloud computing is at its infant stage and aims at satisfying the needs of its clients. This led to the modeling of a search engine for discovering services on the cloud [Jaeyong (2011)].

1.1 *Cloud Computing*

Cloud computing is Internet-based computing, which typically involves the provisioning of dynamically scalable and often virtualized resources as services over the Internet. Customers do not own or maintain the physical infrastructure, but consume resources as-a-service from a third-party provider and pay only for the resources that they consume.

1.2 *Cloud Service Discovery*

The basic notion of Service Oriented Architecture is to enable the service specification to be done in an implementation-independent language for better interoperability of services. Earlier, Web Service Description Language (WSDL) and Universal Description and Discovery Interface (UDDI) were the well known standards, but are regarded as failure today [Eurescom (2010)]. Considering cloud computing, services are offered at

different levels, which not only involve software, data and application, but also other capabilities like platform, infrastructure, etc., In this perspective, the discovery and use of relevant services becomes a crucial task.

2. Stages of Cloud Service Discovery

Cloud Service Discovery includes the following stages:

Stage 1: Query Evaluation

The user query is received by the query evaluator, which processes the query and transforms it into concepts and sends them to the semantic reasoning agent.

Stage 2: Semantic Reasoning

The cloud ontology represents the semantic relationships among the cloud concepts and individuals. The similarity reasoner receives the concepts from the query evaluator and processes them to determine the degree of similarity. The reasoning is done using the following similarity reasoning methods [Jaeyong (2010)]:

a) Concept similarity reasoning
b) Object property similarity reasoning
c) Data type property similarity reasoning

Stage 3: Service Ranking

Once the similarity between services is determined, the next step is to calculate the service utility, based on which the services are ranked.

Stage 4: Service Provisioning

The results of service discovery, which include the list of available services and the relevant service(s), are returned to the cloud service consumer.

3. Ontology and Reasoning

Ontology is a conceptual schema about a domain. Cloud ontology provides a set of concepts, individuals and relationships among them [Jaeyong (2011)]. It is typically a hierarchical data structure containing relevant entities, relationships, and rules within the domain, and serves as a machine-understandable dictionary [Sukasom (2006)]. In cloud service discovery, cloud ontology is developed for semantic matching and retrieval of appropriate services from the available services.

Reasoning is a mechanism used for answering queries over ontology classes and instances [Muhammad (2011)].The cloud users pose their requests in the form of queries which are forwarded to the reasoner, which in turn maps the concepts of the query with the concepts of the cloud services.

The similarity between two concepts ‘a’ and ‘b’ can be calculated using the following formula [Jaeyong (2010)]:

$$\mathbf{sim\ (a, b) = \alpha\ sim_{con}\ (a, b) + \beta\ sim_{obj}\ (a, b) + (1- \alpha - \beta)\ sim_{data}\ (a, b)}$$

where, α and β are the weights of each clause and $0 \leq sim(a,b) \leq 1$.

This formula includes the following similarities:

3.1 *Concept similarity*

The usual term based similarity just finds the string based correspondences between labels [Muhammad (2011)], whereas concept similarity computes the similarity among concepts which have the same meanings but are referred to differently.

$$\mathbf{sim_{con}\,(a, b) = \left|\, Super(A) \cap Super(B) \,\right| / \left|\, Super(A) \,\right|}$$

where,

A, B – the most specific concepts of individuals a and b

Super(A) and Super(B) – the sets of all reachable super concepts from the concepts A and B

3.1.1 *Steps to determine concept similarity*

1. Consider the two individuals for which the concept similarity is to be calculated, ‘a’ and ‘b’.
2. Count all the reachable super concepts from each of the specific concepts, ‘A’ and ‘B’, of the individuals ‘a’ and ‘b’.
3. Determine the commonly reachable super concepts of ‘A’ and ‘B’.
4. Divide the result of step 3 by the result of step 2 (for individual ‘a’).

3.2 ***Object property similarity***

Object properties represent the meanings of concepts and also provide the relationships among concepts in ontology.

$$\mathbf{sim_{obj}\,(a, b) = \bigcup_{(x,y)} Sim\,(x,y) / \left|\, O(a) \,\right|}$$

where,

$U = \{ (x,y) \mid (a,p,x) \in O(a), (b,p,y) \in O(b)\}$

O(a) – a set of triples containing the object properties of the individuals a and b

U – the set of object values, with common predicate p of individuals a and b in each triple O(a) and O(b)

3.2.1 *Steps to determine object property similarity*

1. Consider the two individuals for which the concept similarity is to be calculated, ‘a’ and ‘b’.
2. Identify O(a) and O(b), which are the sets of triples containing the object properties of individuals ‘a’ and ‘b’ respectively.
3. Determine | O(a) |, which is the count of such sets.
4. Identify U as the set of object values with common predicate ‘p’ of ‘a’ and ‘b’ in O(a) and O(b) respectively.
5. Determine sim(x,y) for all object values identified in step 4.
6. Make a summation of all the individual results obtained in step 5.
7. Divide the result of step 6 by the result of step 3 (for individual ‘a’).

3.3 ***Datatype property similarit***

These properties represent the context of concepts and they are the attributes of ‘concepts’ in the ontology.

$$\mathbf{sim_{data}\,(a, b) = \bigvee_{(x,y,p)} Comp\,(x,y,p) / \left|\, D(a) \,\right|}$$

where,

$V = \{(x,y,p) \mid (a,p,x) \in D(a), (b,p,y) \in D(b)\}$

$Comp(x,y,p) = 1 - (\,|x-y|\,) / Max_{distance}\,(x,p)$

$Max_{distance}\,(x,p) = \max_{i \in I(p)} (\,|x-i|\,)$

$I(p) = \{ i \mid (s,p,i) \in Ontology\}$

D(a) – a set of triples, which contains the datatype properties of individual ‘a’

V – a set of datatype values, with common predicate p of individuals a and b

3.3.1 *Steps to determine object property similarity*

1. Consider the two individuals for which the concept similarity is to be calculated, ‘a’ and ‘b’.
2. Identify D(a) and D(b), which are the sets of triples containing the datatype properties of individuals ‘a’ and ‘b’ respectively.
3. Determine | O(a) |, which is the count of such sets.
4. Identify V as the set of object values with common predicate ‘p’ of ‘a’ and ‘b’ in D(a) and D(b) respectively.
5. Determine Comp(x, y, p), which is the similarity between the datatype values ‘x’ and ‘y’ over the common predicate ‘p’.
6. Determine the maximum reachable distance based on ‘x’ value.
7. Using the result of step 6 and the distance between ‘x’ and ‘y’, determine the similarity between the datatype values of ‘a’ and ‘b’.

4. Sample Ontology of Cloud Services

Consider the following ontology of cloud services (Fig:1) which includes various categories of services like infrastructure, software, hardware, data, programming language and its types, etc.,.

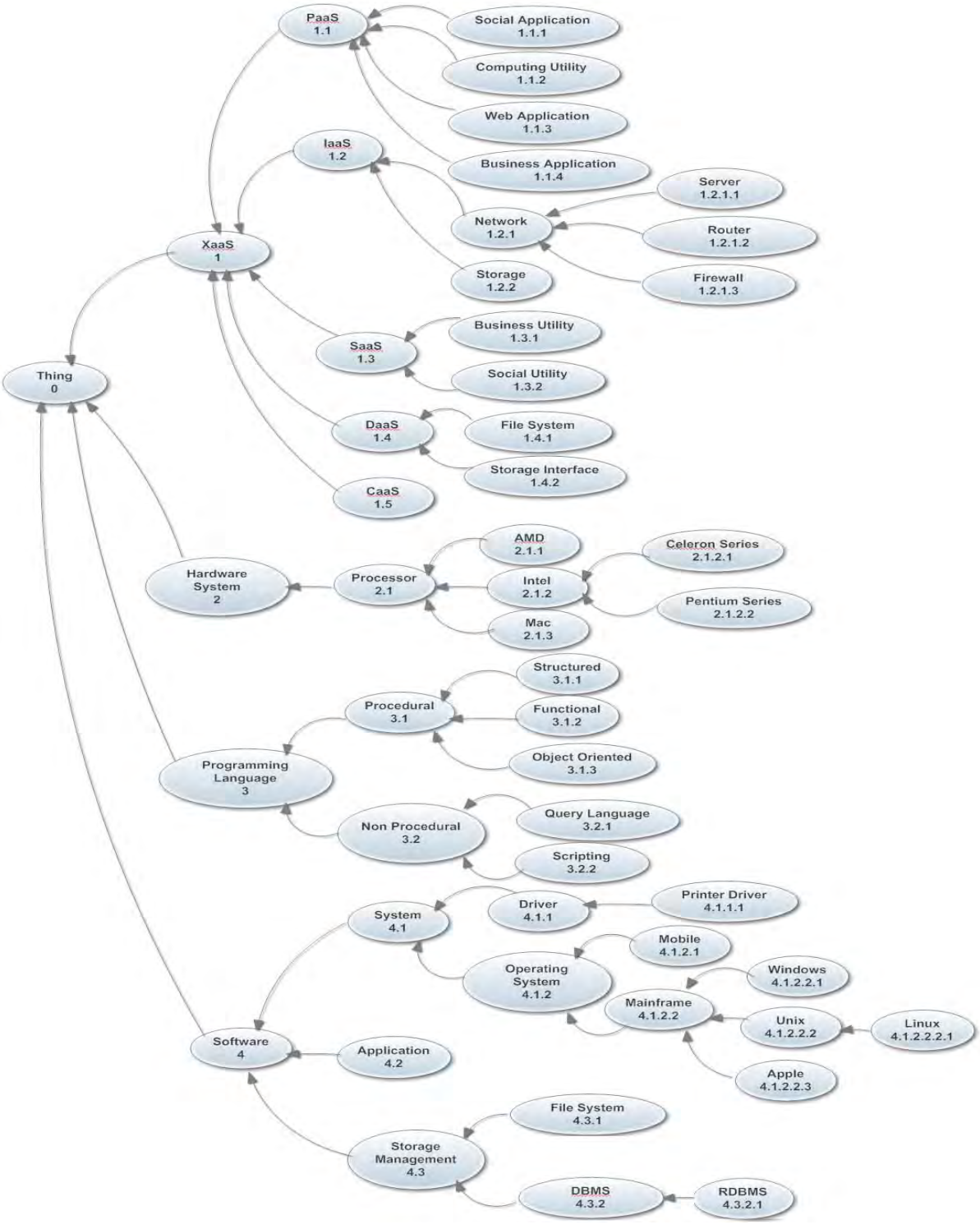


Fig 1: Sample ontology of cloud services

Some examples of concepts and their individuals for this ontology may be tabulated as follows (Table 1):

Table 1: Concepts and their Individuals - Examples

Concept	Individual
Infrastructure as a Service (IaaS)	IProvider1 (IP1 – PSG Cordys Lab)
Software as a Service (SaaS)	SProvider2 (SP2 – PSG SOA Lab)
Data as a Service (DaaS)	Dprovider3 (DP3 – PSG Grid Lab)
Pentium	Proc1
AMD	Proc2
Mac	Proc3
RDBMS	Oracle, SQL Server
Procedural Language	BASIC, FORTRAN, C#
Script Language	VB Script, Python, PHP
Windows	Windows Vista
Linux	Fedora
Apple	Mac
File System	NTFS, FFS

Some examples of individuals and their properties and values for this ontology may be tabulated as follows (Table 2):

Table 2: Individuals and their Properties and Values - Examples

Individual	Property	Value
IP1	hasOS (Obj)	Mac
IP1	hasCPU (Obj)	Proc3
IP1	hasPL (Obj)	C#
IP1	hasMemory (Data Type)	2000
IP1	hasStorage (Data Type)	10000
Proc1	hasSpeed (Data Type)	3.2
Fedora	hasFileSys (Obj)	FFS
SP2	hasOS (Obj)	Fedora
SP2	hasCPU (Obj)	Proc1
SP2	hasDBMS (Obj)	SQL Server
SP2	hasPL (Obj)	BASIC
SP2	hasMemory (Data Type)	6000
SP2	hasStorage (Data Type)	22000
SP2	hasNWBW (Data Type)	4000
SP2	hasNWLat (Data Type)	600
Proc2	hasSpeed (Data Type)	2.6
Proc3	hasSpeed (Data Type)	4.1

Table 2 (*Continued*)

Linux	hasFileSystem (Obj)	NTFS
DP3	hasOS (Obj)	Windows Vista
DP3	hasCPU (Obj)	Proc2
DP3	hasPL (Obj)	Python
DP3	hasMemory (Data Type)	4000
DP3	hasStorage (Data Type)	15000

4.1 Calculation of similarities

The different types of similarity can be calculated as follows:

4.1.1 *Calculation of concept similarity*

$$\mathbf{sim_{con}(a, b)} = \frac{\left|\mathbf{Super(A)} \cap \mathbf{Super(B)}\right|}{\left|\mathbf{Super(A)}\right|}$$

sim_{con} (IP1, SP2) = $\left|\text{Super(IaaS)} \cap \text{Super(SaaS)}\right| / \left|\text{Super(IaaS)}\right|$

Super(IaaS) = 3, Super(SaaS) = 3, $\left|\text{Super(IaaS)} \cap \text{Super(SaaS)}\right| = 2$

Thus, $\mathbf{sim_{con}(IP1, IP2) = 0.66}$

4.1.2 *Calculation of object property similarity:*

$$\mathbf{sim_{obj}(a, b)} = \frac{\underset{(x,y)}{\cup}\ \mathbf{Sim(x,y)}}{\left|\mathbf{o(a)}\right|}$$

|O(IP1)| = 3

U = {(Mac, Fedora), (Proc3, Proc1)}

sim(Mac,Fedora) = $\left|\text{Super(Apple)} \cap \text{Super(Linux)}\right| / \left|\text{Super(Apple)}\right| = 0.83$

sim(Proc3, Proc1)= $|\text{Super(Mac)} \cap \text{Super(Pentium)}| / \left|\text{Super(Mac)}\right| = 0.50$

Thus, $\mathbf{sim_{obj}(IP1, SP2) = \frac{sim(Mac, Fedora) + sim(Mac, Pentium)}{3} = 0.44}$

4.1.3 *Calculation of datatype property similarity*

$$\mathbf{sim_{data}(a, b)} = \frac{\underset{(x, y, p)}{\vee}\ \mathbf{Comp(x,y,p)}}{\left|\mathbf{D(a)}\right|}$$

| D(a)| = 2

V = {(2000, 6000, hasMem), (10000, 22000, hasStorage)}

(i) $Comp(2000, 6000, hasMem) = 1 - \frac{|x-y|}{Max_{distance}(x, p)} = 1 - \frac{|6000-2000|}{6000} = 0.33$

(ii) $Comp(10000, 22000, hasMem) = 1 - \frac{|x-y|}{Max_{distance}(x, p)} = 1 - \frac{|22000-10000|}{22000} = 0.81$

Thus, $\mathbf{sim_{datatype}(IP1, SP2) = (0.33 + 0.81) / 2 = 0.57}$

Thus, the similarity between the concepts ‘a’ and ‘b’ is given as:

$$\mathbf{sim(a, b) = \alpha\, sim_{con}(a, b) + \beta\, sim_{obj}(a, b) + (1- \alpha - \beta)\, sim_{data}(a, b)}$$

Let $\alpha = \beta = 1/3$, then,

$sim(a, b) = (1/3)\, sim_{con}(a, b) + (1/3)\, sim_{obj}(a, b) + (1- 1/3 - 1/3)\, sim_{data}(a, b)$

= (1/3) * 0.66 + (1/3) * 0.44 + (1/3) * 0.57

= (0.66 + 0.44 + 0.57) / 3 = **0.59**

This value represents the similarity between the service concepts and the concepts derived from the query evaluator.

5. Realization of similarity based reasoning: Case Study

Consider a software project for an airline reservation system. The requirements specification of this project includes the following:

- The software application is to assist an airline reservation system with transactions related to making ticket reservations, which includes blocking, reserving, canceling and rescheduling tickets.
- The application should provide an easy-to-use, intuitive Graphical User Interface (GUI) as part of the Clerk/Administrator's working desktop environment.
- It should also provide an interactive GUI, on the World Wide Web for the general customers.

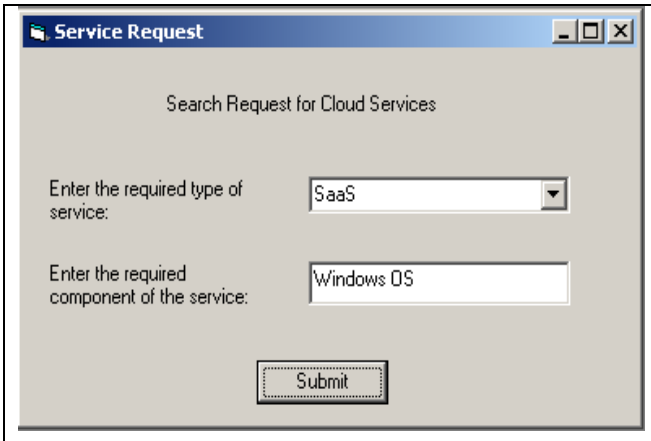
These requirements indeed enforce the following needs from the viewpoint of each project team member:

Member 1 – needs Windows support

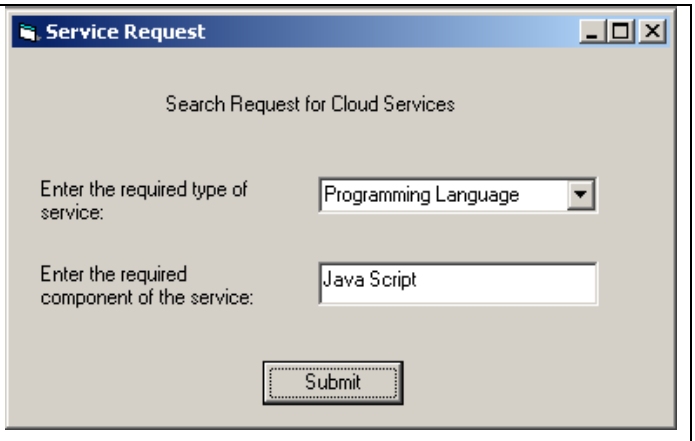
Member 2 – needs to use Java Script for his work

Member 3 – needs a Pentium IV Processor for the entire work

Member 4 – needs a network router to enable network connection



Request 1



Request 2

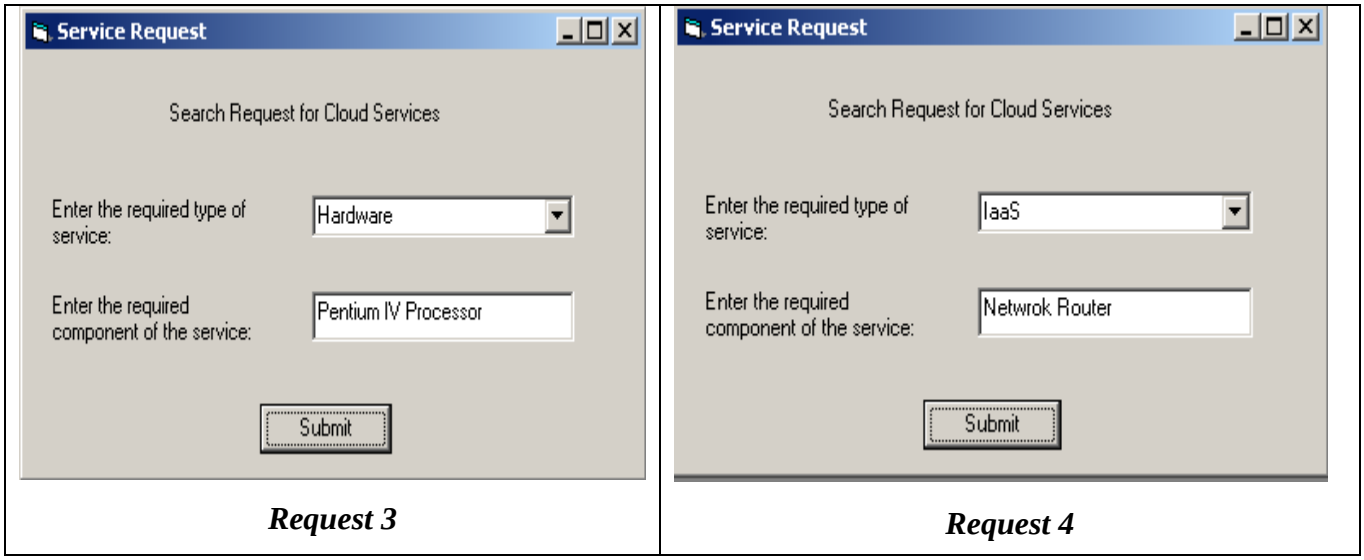


Fig 2: Sample Cloud service requests from users

For the above queries of users, the output paths through the tree structure of the ontology are shown in Fig3. It can be seen from the results that semantic mapping of terms is considered rather than the exact term matching. Table 3 shows the output paths and the corresponding outputs.

Table 3: Output paths and the corresponding outputs

Request No.	***Output Path***	***Sample Output***
1	0 – 4 – 4.1 – 4.1.2 – 4.1.2.2 – 4.1.2.2.1	
2	0 – 3 – 3.2 – 3.2.2	
3	0 – 2 – 2.1 – 2.1.2 – 2.1.2.2	
4	0 – 1 – 1.2 – 1.2.1 – 1.2.1.2	

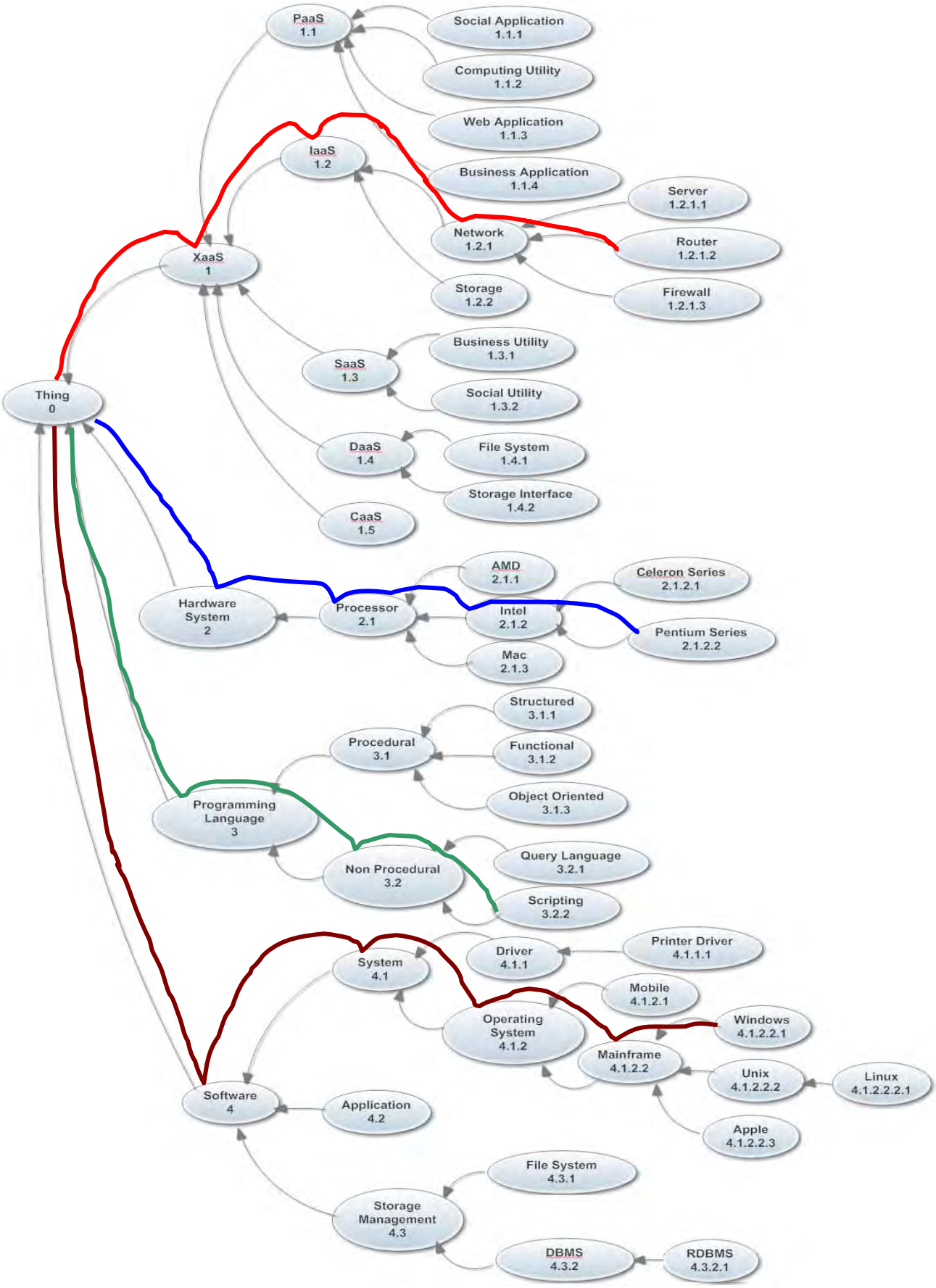


Fig 3: Sample output paths based on similarity reasoning

6. Discussion

This paper focuses on the similarity reasoning methods and the evaluation of different types of similarity among concepts. The concepts that are compared are those derived from the query evaluator which processes the user queries and those of the cloud services that are stored in the database. Similarity reasoning is a type of reasoning where inferences are made based on parallelism between two entities or domains [Christoph (2008)]. This method of reasoning is used to derive additional knowledge from existing knowledge base. The strength of a semantic architecture is the ability to derive new knowledge from existing knowledge, i.e., to reason out new facts from basic facts, which is called deductive description logic reasoning. There are also two new approaches, namely, similarity or analogical reasoning and inductive reasoning. Similarity reasoning uses similarity measures, where it derives new triples from existing triples. Inductive reasoning is realized using statistical techniques, where new triples are derived inferring about an object using statistics from an observed sample. Among these, this paper used the method of similarity reasoning for finding similarities among concepts.

7. Conclusion and Future Enhancements

As the demand for services grows, expertise will be required in the provision of appropriate services as requested by the users. This is one of the major challenges in the field of cloud computing. To face this challenge, it is required to seek the support of semantics and reasoning. The semantic representation of the concepts of a service helps in better reasoning of relevant services by consulting the ontology. This paper focused on the similarity reasoning methods which included the three types of similarity. In future, the one other important factor of ‘quality’ may be included for reasoning. This might involve analyzing the functional and non-functional aspects of services.

8. References

[1] Christoph Kiefer. (2008): Non-Deductive Reasoning for the Semantic Web and Software Analysis, Doctoral Thesis.
[2] Jaeyong Kang, Kwang Mong Sim. (2011): Ontology and Search Engine for Cloud Computing System, Proceedings of International Conference on System Science and Engineering, Macau, China.
[3] Jaeyong Kang, Kwang Mong Sim. (2010): Cloudle: A multi-criteria Cloud Service Search Engine, IEEE Asia-Pacific Services Computing Conference.
[4] Muhammad Fahad, Nejib Moalla, Abdelaziz Bouras, Muhammad Abdul Qadir, Muhammad Farukh. (2011): Towards Classification of Web Ontologies for the Emerging Semantic Web, Journal of Universal Computer Science, Vol. 17, No.7, pp. 1021-1042.
[5] Sukasom Chaiyakul, Kati Limapichat, Avani Dixit and Ekawit Nantajeewarawat. (2006): “A Framework for Semantic Web Service Discovery and Planning”, IEEE.
[6] Eurescom Projects and Studies, P2053 (2010): Dynamic Service Discovery and Use in a Cloud Environment, http://archive.eurescom.eu/Public/Projects/P2000-ries/P2053/default.asp.