A QoS-aware approach for runtime discovery, selection and composition of semantic web services

Sanjay Garg and Kirit Modi
Nirma University, Ahmedabad, India, and
Sanjay Chaudhary
Institute of Engineering and Technology (IET), Ahmedabad University, Ahmedabad, India

Received 6 December 2015
Revised 18 February 2016
Accepted 6 March 2016

Abstract

Purpose – Web services play vital role in the development of emerging technologies such as Cloud computing and Internet of Things. Although, there is a close relationship among the discovery, selection and composition tasks of Web services, research community has treated these challenges at individual level rather to focus on them collectively for developing efficient solution, which is the purpose of this work. This paper aims to propose an approach to integrate the service discovery, selection and composition of Semantic Web services on runtime basis.

Design/methodology/approach – The proposed approach defined as a quality of service (QoS)-aware approach is based on QoS model to perform discovery, selection and composition tasks at runtime to enhance the user satisfaction and quality guarantee by incorporating non-functional parameters such as response time and throughput with the Web services and user request. In this paper, the proposed approach is based on ontology for semantic description of Web services, which provides interoperability and automation in the Web services tasks.

Findings – This work proposed an integrated framework of Web service discovery, selection and composition which supports end user to search, select and compose the Web services at runtime using semantic description and non-functional requirements. The proposed approach is evaluated by various data sets from the Web Service Challenge 2009 (WSC-2009) to show the efficiency of this work. A use case scenario of Healthcare Information System is implemented using proposed work to demonstrate the usability and requirement the proposed approach.

Originality/value – The main contribution of this paper is to develop an integrated approach of Semantic Web services discovery, selection and composition by using the non-functional requirements.

Keywords Semantic web, Web services, Semantic web services, Service discovery, Service selection, Web services composition

Paper type Research paper

1. Introduction

The proliferation of the Cloud computing and Internet of Things (IoT) accelerants the adoption of Web services for the development of emerging Web-based applications. A Web service is defined as an application component developed using an XML language to fulfill the emergent need of application development over the web (Booth *et al.*, 2004). Nowadays, a large number of Web services are provided over the Web from companies

International Journal of Web Information Systems
Vol. 12 No. 2, 2016
pp. 177-200

1744-0084
DOI 10.1108/IJWIS-12-2015-0040

such as Google, Microsoft and Amazon. The ability to select and compose heterogeneous Web services over the Web efficiently and effectively at runtime is an important step towards the development of the Web service applications (Modi and Garg, 2016). By using Web services, an end user is able to create composite services to fulfill the requirement when single service unable to do it.

Most of the approaches related to the Web service composition (Sheng, 2014; Moghaddam and Davis, 2014; Shehu *et al.*, 2014; Upadhyaya, 2014) realized the fact that the prerequisite tasks to generate the composition solution are the service discovery and service selection of the candidate Web services stored in the service repository. Although there is a relationship among the discovery, selection and composition tasks to perform the composition process, researchers treat them at individual level, which restricts the effectiveness of the solution. Furthermore, users want to achieve the increased level of satisfaction and quality assurance from the resultant composition solution. At the same time, user wants to reduce the involvement during the process once the request is submitted. This motivates the utilization of a quality of service (QoS)-based approach and ontology model for the discovery, selection and composition of Web services.

In this paper, the proposed approach defined as a QoS-aware approach is based on QoS model to perform discovery, selection and composition tasks at runtime to enhance the user satisfaction and quality guarantee by incorporating non-functional parameters such as response time and throughput with the Web services and user request. In this paper, the proposed approach is based on ontology for semantic description of Web services, which provides interoperability and automation in the Web services tasks.

The key contributions of this paper are represented by following points:

- development of an integrated framework using QoS-aware approach for Web service discovery, selection and composition of Semantic Web services (SWS);
- semantic description of Web services using domain ontology model;
- development of an integrated approach for service discovery, selection and composition by using non-functional requirements such as response time and throughput and ontology model; and
- development of use case for Healthcare Information System (HIS) by using the proposed approach.

The remainder of this paper is organized as follows. Section 2 presents some background detail of the SWS and QoS model. Section 3 discusses the related work and highlights the features of it. Section 4 represents the proposed work. Section 5 discusses the proposed prototype detail. Section 6 shows experimental work with results. Finally, Section 7 concludes the paper.

2. Background

In this section, the concepts which are related to our proposed work such as SWS and QoS model are discussed.

2.1 Semantic Web

Semantic Web offers semantic description to the contents of the web for effective and automated discovery and composition and invocation (Antoniou and Van

Harmelen, 2004; Berners-Lee *et al.*, 2001). SWS McIlraith *et al.* (2001) provides an open, extensible, semantic framework for describing and publishing semantic content, automated service composition and discovery on the internet.

Ontology which is core part of Semantic Web can be used as a knowledgebase to achieve the automation and interoperability by incorporating semantic description with the Web services which is a main component of the Semantic Web technologies. The ontology is an explicit specification of a conceptualization (Gruber, 1995). Ontology can be seen as a declarative model of a domain that represents the concepts existing in that domain, their attributes and the relationships between them. It is generally considered as a knowledge base which becomes available to applications that need to use and share the knowledge of a domain. Over the period the development of ontology has been shifting from artificial intelligence (AI) lab to the area of domain expert. Several disciplines have developed standardized ontologies that domain experts can use to annotate information in their respective fields.

2.1.1 Foundational and domain ontologies. Based on the application domain or generic model, ontologies can be defined within two very different perspectives: Foundational ontologies, also known as upper or top-level, are a model of the common objects applicable across a wide range of domains and developed to represent explicitly a viewpoint of a reality. They are built upon a core vocabulary that contains the terms and associated object descriptions as they are used in various relevant domain sets. The most relevant foundational ontologies are basic formal Ontology (BFO), general formal ontology (GFO), suggested upper merged ontology (SUMO) and descriptive ontology for linguistic and cognitive engineering (DOLCE), among others (Keet, 2011). Domain ontologies, describe a set of representational primitives that model a domain of knowledge or discourse, providing a common and unambiguous understanding of a domain for both the users and the system (Zhang *et al.*, 2013). It models a specific domain, without any pretension of generality.

Despite an ever-increasing number of biomedical ontologies as domain ontologies, dentistry field still lacks high-quality ontology with good coverage of the dental domain. One reason for this could be that the potential uses and applications of dental ontologies have not been adequately described (Smart and Sadraie, 2012). In this paper, we develop ontology model for the dentistry domain.

2.1.2 Semantic Web services. Ontologies are expected to play a central role to empower Web services with expressive and computer interpretable semantics. The combination of these powerful concepts (i.e. Web service and ontology) has resulted in the emergence of a new generation of Web services called SWS (McIlraith *et al.*, 2001; Medjahed and Bouguettaya, 2011; Martin *et al.*, 2004; Miller *et al.*, 2004). SWS provide an open, extensible, semantic framework for describing and publishing semantic content, improved interoperability, automated service composition, discovery and invocation, access to knowledge on the internet (De Oliveira and de Oliveira, 2011). To define the meaning of distinct service components by semantic annotations or enhancements of a service description, it is necessary to have a domain model which can be used as a knowledge base. Most probably, the best-known knowledge base format is ontologies.

Most approaches intend to describe the semantics of Web services, either with novel semantic description languages (Feier and Domingue, 2005) or by extending the

syntactic mechanisms (Lausen and Farrell, 2016) using domain ontologies to annotate data and Web services.

2.2 Web services discovery, service selection and service composition

This section describes the key features related to the Web service discovery, service selection and service composition as follows.

2.2.1 Web service discovery. Web service discovery (Singh and Huhns, 2006; Rajendran and Balasubramanie, 2009) is a process of finding most suitable service from the repository needed by a service requester according to requester's requirement. Web services discovery approaches are classified into four main categories (Zunino and Campo, 2012): information retrieval (IR)-based approaches, Semantic Web-based approaches, context-based approaches and QoS aware approaches.

By adapting existing IR techniques, some researchers have proposed to treat descriptions of Web services as documents to reduce the problem of discovering relevant services. Semantic Web-based approaches propose to annotate the service descriptions with metadata, such as concept definitions from shared ontologies or sometimes referred as semantics, which gave the notion of SWS. Semantic approaches depend on shared ontologies and annotated resources, whereas IR-based ones depend on textual descriptions.

By definition, context is a situation of an entity (person, place or object) that is relevant to the interaction between a user and an application (Newcomer, 2002). Therefore, considering the context in the service discovery process can improve the quality of the retrieved results. However, contextual information is highly interrelated and has many alternative representations (Pokraev *et al.*, 2003) which make it difficult to interpret and use.

QoS is a set of non-functional attributes that may affects the quality of the service provided by a Web service. QoS parameters describe non-functional aspects of Web services, and they are used to evaluate the degree that a Web service meets specified quality requirements in a service request. QoS-aware service discovery provides quality guarantee with increased level of service satisfaction to the user.

2.2.2 Service selection. Web service selection (Pan and Baik, 2010) is the process to select a service from a set of discovered Web services which can fulfill user requirements and to be returned to the service consumer. The Web service selection process is broadly classified into three main approaches (Sathya *et al.*, 2010): functional-based approach, non-functional based approach and user-based approach.

The functional-based service selection approach represents the static and dynamic semantics. Selecting an appropriate service is concerned with retrieving functional descriptions from service repositories and then ensuring that the described and required interfaces match with each other. Static semantics represents the properties of messages and operation semantics. Dynamic semantics represents the properties of behavior and operation logic. With the rapidly growing number of available services, customers are presented with a choice of functionally similar services. This choice allows customer to select services that match other criteria, often referred to as non-functional attributes. The non-functional based service selection represents the QoS and Context in SWS selection. The properties of QoS may be security, reliability, response time, throughput cost, etc., and the properties of context may include context of customer (location and consumer's name) and context of service (provider's details, descriptions, etc.).

User-based approach represents the selection of best service among numerous discovered services based on customer's feedback, trust and reputation.

2.2.3 Service composition. Web services composition (De Oliveira and de Oliveira, 2011; Eid *et al.*, 2008) is the process to combine more than one service to create composite service. Web services composition approaches (Eid *et al.*, 2008; Shiaa *et al.*, 2008) can be categorized in four different ways as follows: Static Web service composition, dynamic Web service composition, manual Web service composition and semi-automatic and automatic Web service composition.

The Web services composition can be classified as static composition or dynamic composition. In static composition, the requestor has to generate an abstract plan of the tasks at design time which should be used at the execution of the Web service. While dynamic composition is performed by developing the abstract plan of tasks and automatically selecting the Web services means without the involvement of the requestor during the process. Dynamic composition could be achieved using optimization technique (Modi and Garg, 2016) when optimized solution is needed.

Service composition support the users to create applications on top of the native service description, discovery and communication capabilities of service-oriented computing. Service composition can be either performed by composing elementary or composite services. Composite services in turn are recursively defined as an aggregation of elementary and composite services. When composing Web services, the business logic is performed by several services. It is identical to workflow management, where the application logic is realized by composing autonomous applications. A client invoking a composite service can itself be exposed as a Web service. Some of the service composition solutions (De Oliveira and de Oliveira, 2011; Zeng *et al.*, 2004; Alrifai *et al.*, 2012) identified the need of QoS attributes to get the optimum solution.

Next sections discuss the work done by the researchers related to Web service discovery, selection and composition problem.

3. Related work

This section presents the work related to Web services research and utilization of Web services for healthcare sector.

Yang *et al.* (2004) have proposed an integrated approach towards the life cycle of service composition which covers the service discovery, composition and selection of composed services without considering Semantic Web and QoS aspects.

Papazoglou *et al.* (2007) have presented the discovery and composition issues of Web services in the proposed extended service-oriented architecture (SOA) architecture. They have also presented the concept of automated service discovery and composition process using SWS, but non-functional properties are not taken into account.

Da Silva *et al.* (2011) have developed DynamiCoS (Dynamic Composition of Services) framework to represent the dynamic service composition life cycle. DynamiCoS supports end-users to perform automatic discovery, selection and composition using Semantic Web. In this work, authors have not considered non-functional characteristics and keep that as a future scope. We have incorporated QoS parameters into service discovery, selection and composition tasks to enhance the user satisfaction and quality of the proposed work.

Hatzi *et al.* (2012) have proposed an integrated approach for SWS discovery and composition automatically using AI planning techniques. In this approach, a service

composition is mapped into a planning problem. PORSCE II and VLEPPO are developed to demonstrate the implementation of the work. Authors have not considered non-functional requirements during the composition process.

Gholam Hassan Tabatabaei, *et al.* (2010) have proposed SCAIMO framework to fulfill security requirements of both service requesters and providers using secure task matchmaker. A prototype called SCAIMO-composer is developed for the validation of proposed work.

Al-Masri *et al.* (2007) have proposed a Web service relevancy ranking function based on QoS parameters to find the best available Web services during discovery process based on a set of client QoS preferences. This work is focuses only on syntactic discovery process.

Ngan and Kanagasabai (2013) have proposed a generic framework for SWS discovery, where discovery and selection are defined as the key tasks of the framework. They have described the benchmarks available to evaluate service discovery system; among these benchmarks, WSC09 (Kona *et al.*, 2009) data sets have been adopted by us for performance evaluation. Authors have focused on the real-life application of SWS discovery systems as an important problem with support of QoS feature. We have used the similar path to develop the integrated solution.

Yu *et al.* (2007) have presented several approaches for service selection problem with multiple QoS constraints using broker-based architecture to offer end-to-end quality guarantee solution for various composition patterns. They have evaluated the algorithms for service selection process only.

Michlmayr *et al.* (2010) have proposeed the QoS-aware Vienna Runtime Environment for Service-oriented Computing (VRESCO) for dynamic binding, invocation and mediation. Authors describe non-functional attributes in their service meta-data model. These QoS attributes can be specified manually using a management service or measured automatically and integrated into VRESCO environment.

Vu *et al.* (2005) have presented a QoS-based Web service selection and ranking algorithm with trust and reputation management support. Authors have given a formal description and validation of the approach with experiments to demonstrate the quality of results under different cheating behaviors. This work mainly concentrates on the QoS-based Web service selection task.

METEOR-S (Verma *et al.*, 2005) is a Semantic Web-based framework for service composition which offers support for semantic annotation of Web services, discovery and composition. The limitation of this approach is that it uses a static-level composition using a template-based approach of processes. Opposite to this, we have focused on dynamic composition approach.

Fujii and Suda (2004) have presented CoSMoS model for semantic description of services at various levels, i.e. at the data, semantic and logic for dynamic composition. In this approach, focus is given on composition process without considering non-functional parameters, while we have considered parameters like throughput and response time in the service composition.

Kona *et al.* (2007) have proposed an approach to perform discovery and composition of Web services semantically. A multi-step narrowing algorithm is used to perform the composition. As Prolog-based technique is used to perform the discovery and composition processes, services are pre-processed from USDL (Bansal *et al.*, 2005) and transformed to Prolog terms. Pre-processing process is time-consuming, especially for

the dynamic service composition. They have not considered non-functional characteristic in the approach.

De Oliveira and de Oliveira (2011) have proposed framework for QoS-based dynamic Web service composition which is divided into three parts: semantics, syntax and implementation. Semantics part contains domain ontology, composer and execution engine and service repository. Our proposed work is inspired by this work by adding matchmaking and service selection approaches to achieve an integrated approach to enhance user satisfaction and personalization.

For dynamic healthcare service composition (DHCS), process-oriented service discovery approach has been proposed by Wang *et al.* (2009) which uses semantic profiles to represent the semantic descriptions of process functionalities using ontology such as HL7 which is domain specific. The presented work is based on web services business process execution language (WSBPEL) standard rather than dynamic approach.

Internet Reasoning System (IRS) is a framework based on SWS where services can be described by their semantics, discovered, invoked and monitored. Different components of IRS II (Motta *et al.*, 2003) are able to communicate through SOAP protocol. The underlying framework of IRS II is based on the Unified Problem Solving Method Development language and is also used for storing knowledge description.

COCOON (Della Valle *et al.*, 2005) is a Web services based project aimed at reducing medical errors which focuses on resolving the problem of integration in healthcare domain through discussion of the problem of integrating components from service discovery to service composition. It is a WSMO compliant project and uses WSMO compliant service discovery engine for resolving the service discovery issue. In COCOON, the most appropriate services are discovered to be used by the specialist, hence providing better healthcare services.

Dogac *et al.* (2006) is an SWS-based project for the semantic discovery and composition of services. It uses OWL -S as the approach for implementing SWS and uses HL7 as a standard for communication. Artemis uses OWL mapping tool (OWLmt) for the communication between sender and receiver providing semantic interoperability. The primary focus of Artemis project is on data interoperability aspect by resolving heterogeneities between HL7 standards V2 and V3.

3.1 Discussions on the literature study

Based on our literature study, the QoS-aware approach can be considered as the most appropriate approach for SWS discovery, selection and composition problem because of the following reasons:

- A QoS-aware approach enhances user satisfaction as well as quality guarantee for the solution.
- Users' preferences will be considered to generate the solution.

Several QoS-based approaches for service discovery, selection and composition proposed by the researchers, still following limitations are identified from the literature:

- QoS-based approaches are proposed for service discovery, selection and composition tasks individually but not collectively. In our proposed work, we have focused to work on service tasks collectively.

- Limited number of use cases is developed for the healthcare sector using SWS. We have identified to develop use case for healthcare sector using proposed work.

To address the above-mentioned problems, the following section describes the proposed work.

4. Proposed framework for web service discovery, selection and composition

In this section, we present the integrated framework and formulate the service discovery, selection and composition problem. We propose the approach by using the proposed framework.

In the proposed framework of Figure 1, a user request is passed to the matchmaking module where request will be matched with the semantically annotated services which are stored in the repository. The domain ontology is used for semantic matching process to compute similarity measure between the concepts using reasoned. Matched results of the services will be selected and filtered using service selection and filtering module using QoS-based method. Selected services will be passed to the composition engine to generated composition plans from the candidate services.

4.1 Quality of service model

This section presents the QoS parameters considered in the proposed work and the formulas of each parameter. QoS for any service can be represented through non-functional aspect. It defines the various non-functional parameters such as throughput, response time, availability, reliability cost, security, etc. (De Oliveira and de Oliveira, 2011; Al-Masri *et al.*, 2007; Vu *et al.*, 2005; Mallayya *et al.*, 2015; Rajeswari *et al.*, 2014; Alrifai *et al.*, 2012). Web Service Challenge 2009 (Kona *et al.*, 2007) data set provides support of non-functional parameters such as throughput and response time; hence, these two parameters are used in the proposed work.

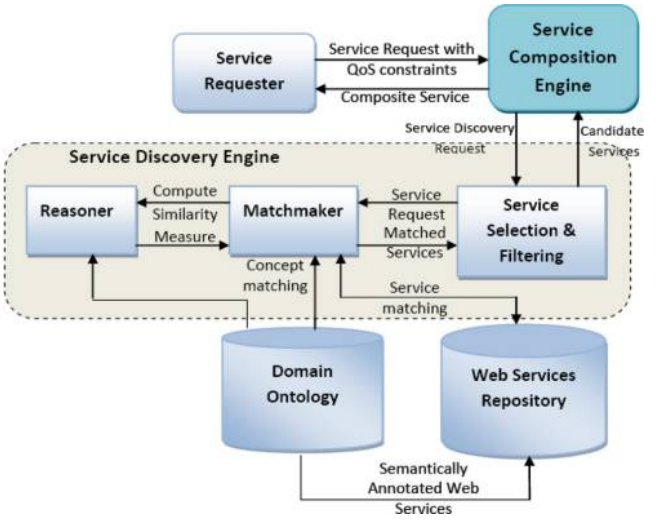

Figure 1.
Proposed framework for Web services discovery, selection and composition

4.1.1 QoS criteria. As a QoS criteria throughput and response time used in our proposed work which are defined as follows.

4.1.1.1 Throughput. The throughput (Yang *et al.*, 2006; Al-Masri *et al.*, 2007) is refers to the number of service requests R that can be processed by a service *s* within a given period. A service's throughput $Q_{TH}(s)$ can be expressed by equation (1) as shown below:

$$Q_{TH}(s) = \frac{\#R}{Time - period} \tag{1}$$

Where $\#R$ is the number of service request. Depending upon the service's characteristics, the period may vary from millisecond (ms) to minute.

4.1.1.2 Response time. The response time (Yang *et al.*, 2006; Al-Masri *et al.*, 2007) refers to the time taken to send a request and receive a response through service execution. A service's response time $Q_{RT}(s)$ can be expressed by equation (2) as shown below:

$$Q_{RT}(s) = ET + WT \tag{2}$$

Where ET is the execution time of service s and WT is the waiting time of service *s*.

Formulas to calculate aggregate values of response time and throughput for sequential execution pattern are presented in Table I. In serial execution pattern, services are executed one after another, and no overlap is considered between execution periods of Web services.

Where $Q_{RT}(si)$ is the response time of a service*si* and $Q_{TH}(si)$ is the throughput of a service *si*. The QoS vector of ith composite service is calculated by equation (3) as follows:

$$Q(S) = Q_{RT}(S) + Q_{TH}(S) \tag{3}$$

4.1.2 Calculation of overall QoS score. The Overall QoS score can be calculated by the simple additive weighting (SAW) method proposed by Yoon and Hwang (1995) and used by Vu *et al.* (2005), Ouzzani and Bouguettaya (2004), Zeng *et al.*, 2004 to select an optimal Web service using local optimization technique. There are two main phases to apply SAW method: scaling and weighting which are defined as follows.

4.1.2.1 Scaling. QoS parameters could be either positive or negative; thus, some QoS values need to be maximized (i.e. the higher the value, the higher the quality), whereas other values have to be minimized (i.e. the higher the value, the lower the quality). To perform this, the scaling phase normalizes each QoS parameter value according to the following formulas. Scaling could be categorized into positive scaling and negative scaling:

Serial no.	QoS criteria	Aggregation function (Sequential)
1	Response time	$Q_{RT}(s) = \sum_{i=1}^{n} Q_{RT}(si)$
2	Throughput	$Q_{TH}(s) = Min_{i=1}^{n} Q_{TH}(si)$

Table I. QoS aggregation function

- *Positive Scaling*: The objective of this scaling category is to maximize the value of QoS criteria. It defines the scaling for positive features, e.g. throughput.
- *Negative Scaling*: The objective of this scaling category is to minimize the value of QoS criteria. It defines the scaling for negative features, e.g. response time.

Equations (4) and (5) presents the formula to calculate the positive and negative scaling values of a candidate service (*CS*) as follows:

$$CSp = \frac{q - qmin}{qmax - qmin}, if\ qmax - qmin \neq 0 \tag{4}$$

$$CSn = \frac{qmax - q}{qmax - qmin}, if\ qmax - qmin \neq 0 \tag{5}$$

Where q is the QoS value of respective service, *CSp* is the formula for positive scaling, *CSn* is the formula for Negative scaling, *qmin* is the minimum QoS value of quality criteria associated with service and *qmax* is the maximum QoS value of quality criteria associated with service

Equations (6) and (7) are derived from (De Oliveira and de Oliveira, 2011) which define a formula to perform scaling on a criterion q of candidate services CS by assuming if q is positive or q is negative, respectively. In brief, the scaled value is determined by considering the highest and lowest values of a given criterion of the candidate services in the candidate list of service repository:

$$s(CS, q) = \begin{cases} \dfrac{overall(SQ, q) - q_m(q)}{q_M(q) - q_m(q)}, if\ q_M(q) - q_m(q) \neq 0 \\ 1, \qquad otherwise \end{cases} \tag{6}$$

$$s(CS, q) = \begin{cases} \dfrac{q_M(q) - overall(SQ, q)}{q_M(q) - q_m(q)}, if\ q_M(q) - q_m(q) \neq 0 \\ 1, \qquad otherwise \end{cases} \tag{7}$$

Where q_M is the $Max\{overall(SQ, q): SQ \in X\}$ and q_m is the $Min\{overall(SQ, q): SQ \in X\}$; here X is the list of candidate services and $overall(SQ, q)$ is the overall QoS score calculation for the selected candidate services *CS* and q is the single criterion.

4.1.2.2 Weighting. This phase computes the overall score by taking into account all the criteria and the weight assigned by the user to represent the user's priority for each criterion. The overall QoS score of a single service can be defined by an equation (8) as follows:

$$OverallQoS\left(\frac{Sqos}{R_{qos}}\right) = \sum_{i=0}^{n}\left(\frac{CSp}{CSn}\right) * W \tag{8}$$

Where $OverallQoS(Sqos)$ is the overall QoS score of all Criteria of a single service $OverallQoS(R_{qos})$ is the overall QoS score of all criteria of user request. The weight $W \in [0,1]$ which represents the weight of each criterion and it is like a coefficient. The overall QoS score for candidate service can be expressed by value which uses

the values derived through scaling and weighting for the user request R for each criterion:

$$OverallQoS(SCQ) = \sum_{\forall(q,Q(S),W)\in R_{qos}} s(CS,q) * W \quad (9)$$

The equation (9) expresses the formula to calculate the overall QoS score by taking weighted sum of QoS values of selected candidate services for a composite service.

4.2 Web service composition problem

To formulate the composition problem, following terminology need to be defined with necessary notations:

(1) *Request (R_q)*: A composition request R_q comprises a collection of provided inputs R_{qin}, a collection of required outputs R_{qout} and a set of QoS constraints R_{qos} imposed by the requester.

(2) *Ontology (Ont)*: An Ontology *Ont* represents a collection of concepts used to define the specific domain. These concepts have relationship with each other as per the subsumption relation, i.e. a concept $c_1 \in Ont$ subsumes another concept $c_2 \in Ont$ if c_1 is a super class of c_2; c_2 subsumes c_1 if c_1 is a subclass of c_2; c_1 and c_2 subsumes each other if they are the same.

(3) *Web service (S)*: A web service S is defined as $\{S_i, \ldots, So\}$, where $S_i = \{S_1, S_2, \ldots, S_i\}$ is a finite collection of required input concept and $S_o = \{S_1, S_2, \ldots, S_j\}$ is a finite set of provided output concept.

(4) *Repository (SR)*: A repository *SR* represents a collection of services. Every service $S \in SR$ comprises a required inputs S_i, a collection of provided outputs S_o, and, a collection of QoS criteria $S_{qos} = \{(q_1, v_1); (q_2, v_2), \ldots, (q_k, v_k)\}$, where qi $(i = 1,2,\ldots, k)$ is a quality criterion, v_i is the value of the criterion q_i associated with the service and k is the total criteria provided.

(5) *Composition (SC)*: A composition *SC* is a represented as acyclic form of directed graph with vertices $C_v = \{S \mid S \in SR\}$ and edges $C_e = \{(u,v) \mid \forall u,v \in C_v \wedge \exists\ c_1 \in u_{out} \wedge \exists\ c_2 \in v_{in} : c_2$ subsumes $c_1\}$.

(6) *Comparator function (C_{cmp})*: A comparator function (C_{cmp}) sort the service composition (*SC*) and indicating whether *SC1* ($r < 0$) or *SC2* ($r > 0$) should be expanded next.

(7) *Sorted list of composition (X)*: A sorted list of composition X contains a set of compositions sorted according to comparator function. X may be 0 or a set of single service compositions.

(8) *Requesters' QoS constraints (R_{qos})*: A set of QoS constraints provided by requester for desired services. $R_{qos} = \{(q_1, v_1); (q_2, v_2), \ldots, (q_k, v_k)\}$, where qi $(i = 1,2, \ldots, k)$ is a quality criterion, v_i is the value of the criterion q_i provided by the requester and k is the total criteria provided.

(9) *Semantic similarity between concepts*: The semantic similarity between concepts refers to the degree that the two concepts can match, and it is a

quantitative definition given on the basis of the four match types (Bellur *et al.*, 2008) presented as follows:

- *Exact*: It means outR and outS are exactly the identical concepts.
- *Plug-in*: It means outR is subsumed by outS, means outR is a subclass of outS.
- *Subsumes*: It means outR is subsumes outS, means outR is a superclass of outS.
- *Fail*: When subsumption relationship between outR and outS is not available.

4.2.1 Composition algorithm

Algorithm 1. Web services composition algorithm

Input: Rq_{out}→Composition Request, SR→ Service Repository
Data: X→ A set of candidate compositions **Output**: SC→ the composition, or ∅

1. **Begin**
2. **foreach** $outR \in Rq_{out}$ **do**
3. **foreach** $s \in SR$ **do**
4. $s \leftarrow matchmaking(outR, outS);$
5. $SL \leftarrow selection\ (s, R_{qos})$
6. $SC_v \leftarrow \{SL\};$
7. **Endforeach**
8. **Endforeach**
9. **foreach** $S_{qos} \in SC$ **do**
10. **if** $OverallQos(S_{qos}) > OverallQos(R_{qos})$ **then**
11. $X \leftarrow$ append(X, SC);
12. **Endif**
13. **Endforeach**
14. **while** $X \neq \emptyset$ **do**
15. $X \leftarrow$sortServices(descending, X, C_{cmp});
16. $SC \leftarrow$popLastComposition(X);
17. **if** isSolution(SC) **then**
18. **return** SC;
19. **Endif**
20. **foreach** $S \in SR$ **do**
21. $s \leftarrow matchmaking(outR, outS);$
22. $SL \leftarrow selection\ (s, R_{qos})$
23. $SC_{v\ new} \leftarrow SC_v \cup \{SL\};$
24. $SC_{e\ new} \leftarrow SC_e \cup \forall s2 \in c_v \{(s, s2)\};$
25. **if** $OverallQos(S_{qos}) > OverallQos(R_{qos})$ **then**
26. $X \leftarrow$append(X, SC_{new});
27. **Endif**
28. **Endforeach**
29. **Endwhile**
30. **return**∅
31. **End**

An algorithm for Web service composition depicted in Algorithm 1 is inspired by the work proposed in (De Oliveira and de Oliveira, 2011). Algorithm 1 starts to perform the composition process for a request R_q by searching all services which can offer a concept as an output which is semantically equal to the outputs required defined in $outR \in Rq_{out}$.

The matchmaking process is performed to calculate semantic similarity measure for a given request against available services. Matched results will be used for selection of candidate services (Steps 2-8). A service may be a candidate for service composition if it satisfies QoS requirements specified by user request $R_{qos.}$ Quality parameters Q_{qos} are extracted from the services and compared with quality constraints imposed in the request to get overall value of QoS. Candidate compositions are filtered those no longer meet the quality constraints (Steps 9-13). The filtered candidate compositions are appended into X.

The candidate compositions are sorted in descending order using comparator formula by comparing two candidate level compositions SC_1 and SC_2 and provide a value (r) less than zero if SC_2 is more promising than SC_1, more than zero if SC_1 is more promising than SC_2 and zero if both are equally evaluated. The comparator function proposed by Weise *et al.* (2008) where they have used four composition characteristics: known concepts – consists of input concepts specified by the user and output concepts specified by the output all services in SC; unknown concepts – consists of output concepts of the requester and input concepts of each service; eliminated concepts – consists of already provided unknown concepts; and the total participating services in composition.

As mentioned in the equation (9), the comparator function considers overall QoS along with the four composition characteristics. Once the sorted list is generated by the comparator formula, the composition is returned as a solution which has high overall QoS score (steps 14-19). In undesired case, the selected composition is explored to form new candidate compositions (steps 20-28).

4.2.2 Algorithm analysis. Based on the step-by-step analysis, the time complexity of the Algorithm 1 is defined as follows:

$$Time = O(p^3)$$

Where, p is the total services participating in the composition process.

The algorithm proposed in (De Oliveira and de Oliveira, 2011) have shown time and memory complexity $O(p^m)$ where p is the participating services in the composition and m is the total services in a composition. We have achieved similar value of time complexity of our proposed approach. In the best case, the time complexity of our approach is $O(p^2)$, it defines the situation when composition solution will be found at first attempt. In the worst case and average case the time complexity is $O(p^3)$, it define the situation when all the possible candidate services participating in the composition required to be considered.

4.3 Web service matchmaking algorithm

The matchmaking algorithm proposed in Algorithm 2 is inspired from (Choi *et al.*, 2005), and it is also followed by Bellur *et al.* (2008). This matchmaking algorithm is mainly based on the required concepts and offered concepts.

Algorithm 2. Web services matchmaking algorithm

Input: ***outR*** (required concepts), ***outS*** (offered concepts)
Output: ***ExactConcepts*** **or** ***Plug-inConcepts*** **or** ***SubsumeConcepts*** **or** ***Failed***
1. **if** ***outR*** = ***outS*** **then**
2. return ***ExactConcepts***

3. **else if** ***outS*** subsumes ***outR*** **then**
4. return ***PluginConcepts***
5. **else if** ***outR*** subsumes ***outS*** **then**
6. return ***SubsumeConcepts***
7. **Else**
8. return ***Failed***
9. **end if**

For the matchmaking Algorithm 2, the best case, worst case and average case time complexity is O(1). It shows the constant value for the time complexity.

4.4 Web service selection algorithm

Web service selection algorithm is shown in Algorithm 4. Following definitions have been used to formulate the algorithm:

- *A Set of Discovered services (SD)*: It represents the services retrieved from the discovery process.
- *Users' criteria (Cj)*: It defines the QoS criteria specified by the user in the form of the request.
- *Normalized value of QoS (N_{qos})*: It provides the normalized value of QoS parameters.
- *Sorted list of selected services (SL)*: It presents the ranked and sorted list of selected services as an output of the selection proves.

Algorithm 3. Web services selection algorithm

Input: $\boldsymbol{R_{qos}}$(User Request), ***SD*** (Set of Discovered services)
Output: ***SL*** (Sorted list of Selected services)

1. **Begin**
2. $\boldsymbol{S \rightarrow S_{qos}}$
3. **foreach** $\boldsymbol{Si \in SD}$ ***do***
4. **if** overall $\boldsymbol{S_{qos} > R_{qos}}$
5. **Endforeach**
   // filtering
6. **while** $\boldsymbol{i <= n}$ **do**
7. **for** $\boldsymbol{j}$ = 1 to 2,
8. **if** $\boldsymbol{qi(Si) < Cj}$ **then** filterout ***Si***
9. **Endif**
10. **Endfor**
11. **Endwhile**
    // ranking
12. calculate ***CSp*** & ***CSn*** **foreach** ***Si***,
13. $\boldsymbol{Q_{score} = N_{qos} + w}$
14. ***SL*** ← sort ($\boldsymbol{Q_{score}}$desc)
15. **Return** ***SL***
16. **End**

In the Algorithm 3, Steps 2 to 5 represent the list of candidate services for requesters with their QoS values. Steps 6 to11 represent the filtering mechanism. During the filtering process, matching of the user request criterion (c_j) with each service criterion (q_{ij}) is performed. The services which are unable to satisfy the requester's specified

constraints will be filtered out. The aim of the service filtering process is to generate refine result to the composition module for to increase the quality and performance of the composite result. Steps 12 to15 represents ranking mechanism and select the best services according to highest QoS score. The ranking process is performed as follows. First, scaling method is applied to get normalize value of all QoS parameters for each service then after the weighted value (w) is calculated. The total QoS score will be calculated by combining the normalized value and weighted value. Once the total score is computed, sort the services in descending order according the total QoS score. The selected list of services will be generated as result of the service selection process.

5. Experimental work and results

This section describes the detail about the experimental work carried out and results which are generated based on the experiments. All the experiments presented here have been performed on with Intel Core 2 Duo 2.0 GHz processor with 3.0 GB RAM memory, Eclipse Europa with JDK 1.6.0, protégé 4.3, Jena API.

To evaluate the proposed approach, we have used the Web Service Challenge (WSC 2009) data sets. In the following experimentation, the effect of increasing the size of Web services on the performance of proposed work is presented. To fulfill the task, different data sets containing 136, 1,023, 2037, 4,018, 8,016 Web services are run. In these experiments, two QoS attributes including: response time and throughput are considered.

We have measured the execution time of algorithms by running them multiple times and get the average value of the results. A sample query could be: response time $< 1{,}000$ ms, throughput $>=25$. Then the query vector is set as (1,000, 25). A sample preference vector for this query could be (1, 2).

5.1 Comparison of proposed approach with quality of service-based approach

A comparison of proposed approach with QoS-based approach (De Oliveira and de Oliveira, 2011) is depicted in Figure 2. The presented results demonstrate the performance and scalability of the proposed work. Results show that proposed approach performs better in comparison with the QoS-based approach.

The results show that for different values of services of data set the proposed approach performs well in comparison with the QoS-based approach which has focused on composition process only while to make an approach more efficient integrated approach is used in the proposed work. The same environment of experimental work has been considered for comparison purpose.

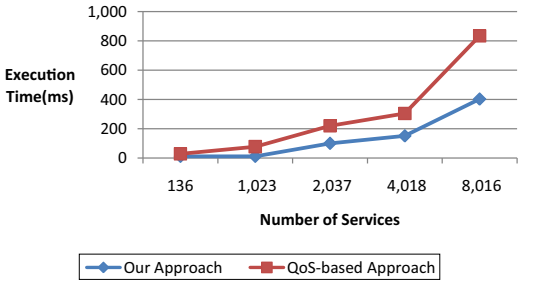

Figure 2.
Comparison of proposed approach with QoS-based approach

5.2 Comparison of proposed approach with hybrid approach

A comparison of proposed approach with Hybrid approach (Ma *et al.*, 2015) is presented in Figure 3. The results show that the execution time of hybrid approach is much more than our approach.

The Hybrid approach proposed in (Ma *et al.*, 2015) is an integration of genetic programming technique with greedy search method which increases execution time of composition process, while our approach is based on greedy search technique by integrating discovery, selection and composition process which performs composition process in reasonable time and the generated solution is found to be near to optimal solution.

6. Healthcare information system

An HIS integrates the healthcare's business process and information systems to deliver better healthcare services (Almunawar and Anshari, 2011). The Healthcare information is offered through electronic medium, i.e. electronic health record (EHR), but different organizations uses different formats which create problems of standardization and interoperability of the information. Some key challenges of EHR adoption are cost, ownership, standards and human factors (Seckman, 2013). An ideal solution is required to make healthcare information interoperable and easily accessible by the end users. Another requirement is sharing of information for better and quick solution. By considering these challenges, we have developed a prototype for HIS using the proposed approach as shown in Figure 4. The main components of the prototype framework are discovery engine, selection and filtering engine and composition engine. At first, users' requirements are translated into XML-based format using request formulation module and forwarded the request to the discovery engine.

The discovery engine will perform semantic matchmaking based on user's request and semantically enabled services derived from the Web service repository. Discovered services will be used for selection and filtering based on user specified QoS constrains. The resultant services are composed through composition engine and composite services will be delivered to the user as a solution based on users' requirement.

6.1 Web services for the healthcare information system

To describe the dynamics of the prototype framework, we present below a use case from the electronic-health domain. We define several Web services as shown in Table II related to healthcare sector.

The Web services such as getHospitalWithDoctor finds the nearest hospital at given location and a doctor with his/her expertise, getTransportation finds the transportation

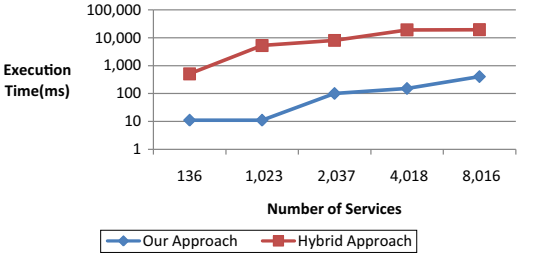

Figure 3.
Comparison of proposed approach with hybrid approach

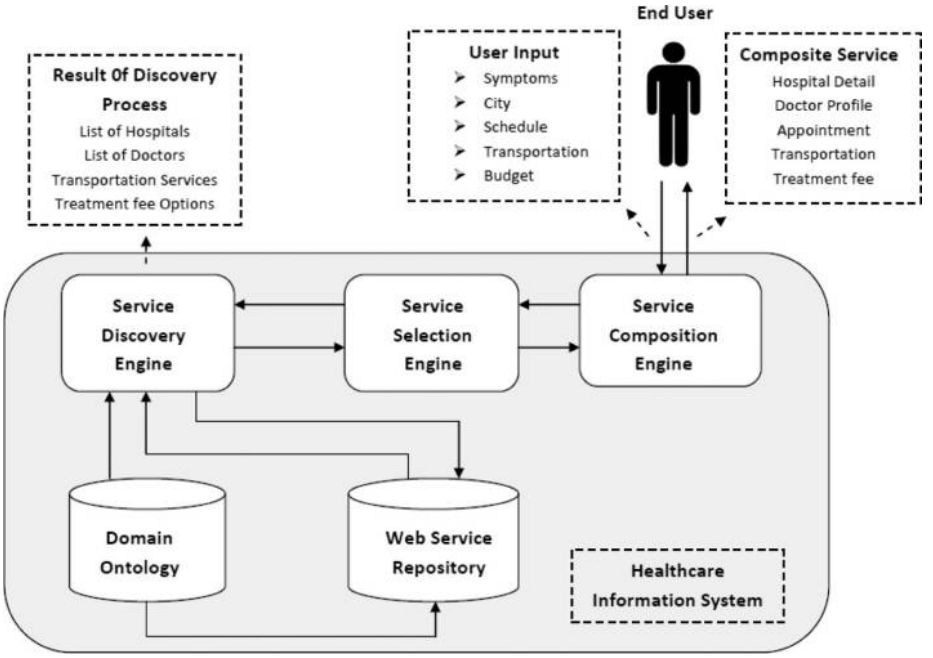

Figure 4.
A prototype for HIS

related detail, getAppoinment retrieves the appointment detail and getPackage finds the payment related options. We have added semantic description to these Web services using the ontology developed by us. These semantically enabled services are stored into the service repository for the discovery, selection and composition. We have considered the repository of 1,000 services of specified types. Based on the knowledgebase, we define a user request that results towards the composition solution to fulfill the request.

6.2 Ontology model for healthcare information system

The sample ontology presented in Figure 5 is the dentistry ontology, described using OWL and developed using protégé 4.3.

Various subclasses such as symptoms, doctor and hospital can be described along with their properties and relationship with each other in the form of concepts. The sample ontology can be used as the knowledgebase in the prototype framework.

Serial no.	Functionality of the Web service	Inputs	Outputs
1	getHospitalWithDoctor	Symptoms, Location	Hospital, Doctor
2	getTransportation	Pickup-point, Landmark	Transport facility
3	getAppoinment	Date, Time	Appointment
4	getPackage	Consulting fee	Payment
5	getOffer	Offer inquiry	Offer detail

Table II.
List of Web services

6.3 Composition process for healthcare information system

The scenario implemented in this paper as a use case concerns the utilization of healthcare services associated with the hospital. The user wishes to provide as inputs a symptoms, location, pickup-point, landmark and date and time for appointment. The outputs of the composite service are an appointment with a dentist of a nearest hospital as per the date and time specified by the user.

User requirements for the desired composite service are expressed in the natural form through user interface which are mapped to the semantically relevant concepts for the semantic matching, selection and composition. A composition plan presented in Figure 6 demonstrates the internal mechanism of the composition process once user has submitted the requirements. User will not take part in the intermediate phases of the discovery, selection and composition process which reflects the automatic behavior of the proposed use case.

6.4 Performance results of healthcare information system

To demonstrate the performance of the prototype, we have used the two QoS constraints such as response time and throughput. The performance is measured by varying the number of services in the range of 100 upto 500 service for the purpose of comparison with DynamiCoS as shown in Figure 7. The obtained measurements show that the execution time of our prototype increases along with the number of services and show

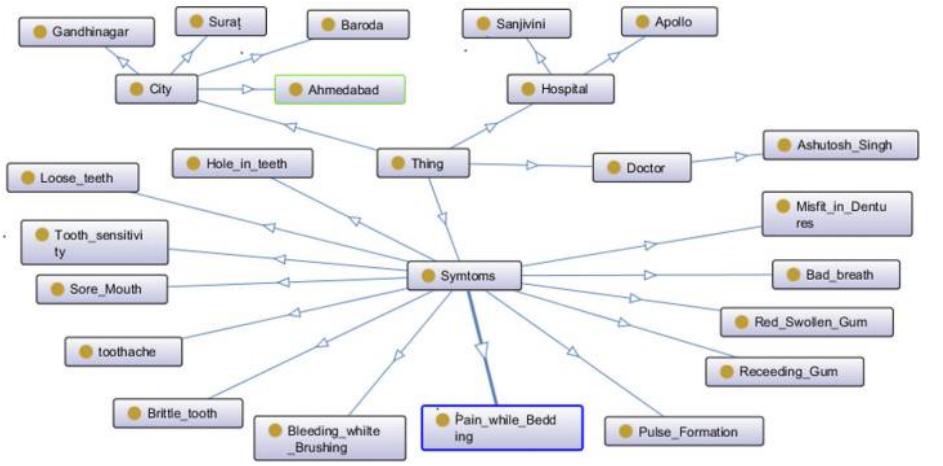

Figure 5.
A sample of dentistry ontology model

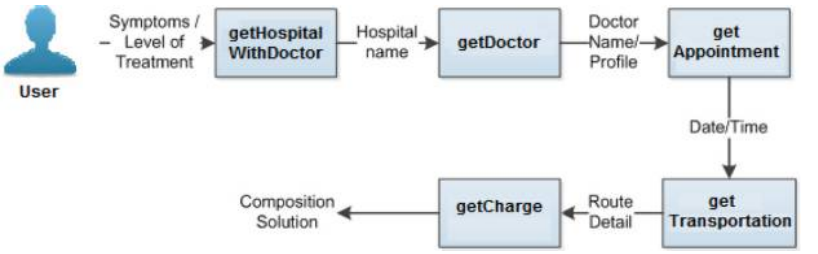

Figure 6.
Conceptual representation of composition plan

improved performance in comparison with DynamiCoS. At this juncture, it is important to note that the DynamiCoS approach has not considered non-functional parameters to perform service discovery, selection and composition tasks.

6.5 Comparative study with existing approaches

A comparative study among DHCS (Wang *et al.*, 2009), DynamiCoS and our proposed approach is presented in Table III by considering some aspects such as functionality, results and novel features considered in the proposed approach. The main objective of all these approaches is to offer the healthcare-related services to the end user. For the semantic description, DHCS use HL7 standard ontology, while DynamiCoS and our proposed approach have considered domain specific ontology. In terms of number of services offered, our approach has accommodated 1,000 services in the repository which show the scalability support in the proposed use case. QoS-aware feature is supported by our approach only in comparison with DHCS and DynamiCoS as shown in the table.

7. Conclusion and future work

In this paper, we have presented an integrated approach on the support of runtime Web service discovery, selection and composition based on Semantic Web and non-functional characteristics to facilitate the end user to search, select and compose the services with increased satisfaction. We have proposed a framework to show the inter-relationship among the discovery, selection and composition tasks. Most of the frameworks proposed for the service discovery, selection and composition have considered these tasks on individual basis. Our proposed framework has considered these service tasks collectively. Based on the proposed framework, we have presented the approaches for service discovery, selection and composition by incorporating ontology as knowledgebase and non-functional characteristics. We have developed a prototype for the HIS to offer end user to use the healthcare services for performing the routing task such as to make the appointment and treatment from the nearest healthcare centers. We have performed the experiments on the proposed approach using standard data sets such as WSC2009 to evaluate the performance and comparison with existing work. We conclude that runtime service discovery, selection and composition using non-functional characteristics can be achieved. However, with this work, we demonstrated that end user has to play an important role to generate the efficient composition solution. We have evaluated the proposed work using real-life scenario to show

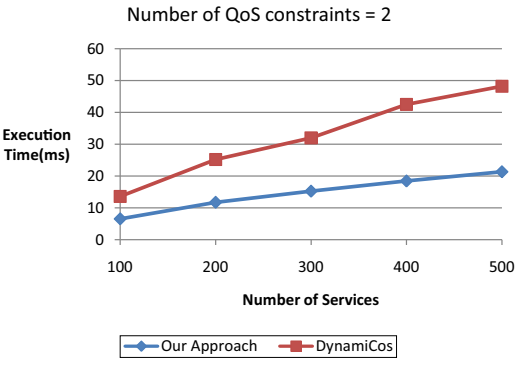

Figure 7.
Composition process time

Table III.
Comparison with existing approaches

Aspects	Dynamic healthcare service composition (Wang *et al.*, 2009)	DynamiCoS (Da Silva *et al.*, 2011)	Proposed prototype
Description	This enables the flow of healthcare information within organizations and the end user can extract the relevant information such as health status	This enables the user to discover, select and compose Web services such as FindHospital, FindDoctor, MakeMedica – Appointment for appointment between the doctor and users	Our work enables the user to discover, select and compose Web services such as getHospitalWithDoctor, getAppoinment, getTransportation and getPackage for the appointment, transportation and treatment services
Semantic description	HL7 ontology	Domain ontology	Domain ontology
Information provided	Information of patient within organization only Patient's report to the nurses Doctors can only prescribe the treatment	Provides hospital location information only Makes appointment with doctor	Provides service detail offered by the hospital information Provides appointment with doctor for the treatment Provides transportation detail Provides package detail for the treatment
Output	Patient's current report Treatment given by the doctor	Appointment detail only	Appointment schedule Information of the doctor Information of the hospital Information of the transportation Approximate charges of the treatment
Number of services in repository	Not specified	500	1,000
Performance comparison	Not given	Not given	Provided
QoS support	No	No	Yes

the evidence of its usability. We have made comparison of proposed approach with existing approaches to demonstrate the effectiveness of the proposed work.

As a future work, it could be possible to extend the proposed system through integration of handheld devices to get the advantage of pervasiveness. Cloud-based platform could also be incorporated to provide services to large-scale level from local access to remote area where healthcare facility is not easily accessible.

References

Al-Masri, E. and Mahmoud, Q.H. (2007), "Qos-based discovery and ranking of web services", *Computer Communications and Networks, ICCCN, Proceedings of 16th International Conference on*, *IEEE*, pp. 529-534.

Almunawar, M.N. and Anshari, M. (2011), "Health Information Systems (HIS): concept and technology", *In Proceedings of the International Conference on Informatics for Development*.

Alrifai, M., Risse, T. and Nejdl, W. (2012), "A hybrid approach for efficient Web service composition with end-to-end QoS constraints", *ACM Transactions on the Web (TWEB)*, Vol. 6 No. 2, p. 7.

Antoniou, G. and Van Harmelen, F. (2004), *A Semantic Web Primer*, MIT Press, Massachusetts.

Bansal, A., Kona, S., Simon, L., Mallya, A., Gupta, G. and Hite, T.D. (2005), "A universal service-semantics description language", *Web Services, 2005: ECOWS 2005 Third IEEE European Conference on*, *IEEE*, p. 12.

Bellur, U., Gupta, A. and Vadodaria, H. (2008), *Semantic Matchmaking Algorithms*, INTECH Open Access Publisher, Rijeka.

Berners-Lee, T., Hendler, J. and Lassila, O. (2001), "The semantic web", *Scientific American*, Vol. 284 No. 5, pp. 28-37.

Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C. and Orchard, D. (2004), "Web services architecture", W3C Working Group Note.

Choi, O., Han, S. and Abraham, A. (2005), "Extended semantic web services model for automatic integrated framework", *Next Generation Web Services Practices, 2005, NWeSP 2005, International Conference on*, *IEEE*, p. 8.

Da Silva, E.G., Pires, L.F. and Van Sinderen, M. (2011), "Towards runtime discovery, selection and Composition of semantic services", *Computer Communications*, Vol. 34 No. 2, pp. 159-168.

Della Valle, E., Cerizza, D., Celino, I., Gadda, L. and Savoldelli, A. (2005), "The COCOON project", *Demos and Posters of the 2nd European Semantic Web Conference (ESWC 2005)*, *Heraklion*, Vol. 29.

De Oliveira, F.G.A, Jr and De Oliveira, J.M.P. (2011), "QoS-based approach for dynamic web service composition", *Journal of Universal Computer Science*, Vol. 17 No. 5, pp. 712-741.

Dogac, A., Laleci, G.B., Kirbas, S., Kabak, Y., Sinir, S.S., Yildiz, A. and Gurcan, Y. (2006), "Artemis: deploying semantically enriched Web services in the healthcare domain", *Information Systems*, Vol. 31 No. 4, pp. 321-339.

Eid, M., Alamri, A. and El Saddik, A. (2008), "A reference model for dynamic web service composition systems", *International Journal of Web and Grid Services*, Vol. 4 No. 2, pp. 149-168.

Feier, C. and Domingue, J. (2005), "D3.1v0.1 WSMO Primer", available at: www.wsmo.org/TR/d3/d3.1/v0.1/#S1,2005

Fujii, K. and Suda, T. (2004), "Dynamic service composition using semantic information", *Proceedings of the 2nd International Conference on Service Oriented Computing, ACM*, pp. 39-48.

Gholam Hassan Tabatabaei, S., Vahid Dastjerdi, A., Wan Kadir, W.M., Ibrahim, S. and Sarafian, E. (2010), "Security conscious AI-planning-based composition of semantic web services", *International Journal of Web Information Systems*, Vol. 6 No. 3, pp. 203-229.

Gruber, T.R. (1995), "Toward principles for the design of ontologies used for knowledge sharing?", *International Journal of Human-Computer Studies*, Vol. 43 No. 5, pp. 907-928.

Hatzi, O., Vrakas, D., Nikolaidou, M., Bassiliades, N., Anagnostopoulos, D. and Vlahavas, I. (2012), "An integrated approach to automated semantic web service composition through planning", *Services Computing, IEEE Transactions on*, Vol. 5 No. 3, pp. 319-332.

Keet, C.M. (2011), "The use of foundational ontologies in ontology development: an empirical assessment", *The Semantic Web: Research and Applications*, Springer, Berlin Heidelberg, pp. 321-335.

Kona, S., Bansal, A., Blake, M.B., Bleul, S. and Weise, T. (2009), "WSC-2009: a quality of service-oriented web services challenge", *Commerce and Enterprise Computing: CEC'09, IEEE Conference on, IEEE*, pp. 487-490.

Kona, S., Bansal, A., Gupta, G. and Hite, D. (2007), "Automatic composition of semantic web services", *ICWS*, Vol. 7 No. 1, pp. 150-158.

Lausen, H. and Farrell, J. (2016), "Semantic annotations for WSDL and XML schema", W3C, W3C Recommendation, available at: www.w3.org/TR/2007/REC- sawsdl-20070828/

McIlraith, S.A., Son, T.C. and Zeng, H. (2001), "Semantic web services", *IEEE Intelligent Systems*, Vol. 2 No. 1, pp. 46-53.

Ma, H., Wang, A. and Zhang, M. (2015), "A hybrid approach using genetic programming and greedy search for qos-aware web service composition", *Transactions on Large-Scale Data-and Knowledge-Centered Systems XVIII*, Springer, Berlin Heidelberg, pp. 180-205.

Mallayya, D., Ramachandran, B. and Viswanathan, S. (2015), "An automatic web service composition framework using qos-based web service ranking algorithm", *The Scientific World Journal*, Vol. 14 No. 4.

Martin, D., Paolucci, M., McIlraith, S., Burstein, M., McDermott, D., McGuinness, D. and Srinivasan, N. (2004), "Bringing semantics to web services: the OWL-S approach", *Semantic Web Services and Web Process Composition*, Springer, Berlin Heidelberg, pp. 26-42.

Medjahed, B. and Bouguettaya, A. (2011), *Service Composition for the Semantic Web*, Springer, Science & Business Media, Berlin.

Michlmayr, A., Rosenberg, F., Leitner, P. and Dustdar, S. (2010), "End-to-end support for qos-aware service selection, binding, and mediation in vresco", *Services Computing, IEEE Transactions on*, Vol. 3 No. 3, pp. 193-205.

Miller, J., Verma, K., Rajasekaran, P., Sheth, A., Aggarwal, R. and Sivashanmugam, K. (2004), "Wsdl-s: adding semantics to wsdl-white paper", LSDIS Lab, University of Georgia, GA.

Modi, K.J. and Garg, S. (2016), "Dynamic web services composition using optimization approach", *International Journal of Computer Science & Communication*, Vol. 6 Number 2 April-Sep 2015, pp. 285-293, ISSN: 0973-7391. doi: 10.090592/IJCSC.2015.630.

Moghaddam, M. and Davis, J.G. (2014), "Service selection in web service composition: a comparative review of existing approaches", *Web Services Foundations*, Springer, New York, NY, pp. 321-346.

Motta, E., Domingue, J., Cabral, L. and Gaspari, M. (2003), "IRS–II: a framework and infrastructure for semantic web services", *The Semantic Web-ISWC*, Springer, Berlin Heidelberg, pp. 306-318.

Newcomer, E. (2002), *Understanding Web Services: XML, Wsdl, Soap, and UDDI*, Addison-Wesley Professional, Boston.

Ngan, L.D. and Kanagasabai, R. (2013), "Semantic web service discovery: state-of-the-art and research challenges", *Personal and Ubiquitous Computing*, Vol. 17 No. 8, pp. 1741-1752.

Ouzzani, M. and Bouguettaya, A. (2004), "Efficient access to web services", *Internet Computing, IEEE*, Vol. 8 No. 2, pp. 34-44.

Pan, Z. and Baik, J. (2010), "A QOS enhanced framework and trust model for effective web services selection", *Journal of Web Engineering*, Vol. 9 No. 2, pp. 186-204.

Papazoglou, M.P. and Van Den Heuvel, W.J. (2007), "Service oriented architectures: approaches, technologies and research issues", *The VLDB Journal*, Vol. 16 No. 3, pp. 389-415.

Pokraev, S., Costa, P.D., Pereira Filho, J.G., Zuidweg, M., Koolwaaij, J.W. and van Setten, M. (2003), "Context-aware services state-of-the-art", *Telematica Instituut*, Vol. 30 No. 1.

Rajendran, T. and Balasubramanie, P. (2009), "Analysis on the study of QoS-aware web services discovery", *Journal of Computing*, Vol. 1 No. 1, pp. 119-130.

Rajeswari, M., Sambasivam, G., Balaji, N., Basha, M.S., Vengattaraman, T. and Dhavachelvan, P. (2014), "Appraisal and analysis on various web service composition approaches based on QoS factors", *Journal of King Saud University-Computer and Information Sciences*, Vol. 26 No. 1, pp. 143-152.

Sathya, M., Swarnamugi, M., Dhavachelvan, P. and Sureshkumar, G. (2010), "Evaluation of qos based web-service selection techniques for service composition", *International Journal of Software Engineering*, Vol. 1 No. 5, pp. 73-90.

Seckman, C. (2013), "Electronic health records and applications for managing patient care", *Elsevier and Typesetter Toppan Bes*, pp. 87-105.

Sheng, Q.Z., Qiao, X., Vasilakos, A.V., Szabo, C., Boume, S. and Xu, X. (2014), "Web services composition: a decade's overview", *Information Sciences*, Vol. 280 No. 1, pp. 218-238.

Shiaa, M.M., Fladmark, J.O. and Thiell, B. (2008), "An incremental graph-based approach to automatic service composition", *Services Computing, SCC'08, IEEE International Conference on, IEEE*, Vol. 1, pp. 397-404.

Shehu, U., Epiphaniou, G. and Safdar, G.A. (2014), "A survey of QoS-aware web service composition techniques", *International Journal of Computer Applications*, Vol. 89 No. 12.

Singh, M.P. and Huhns, M.N. (2006), *Service-Oriented Computing: Semantics, Processes, Agents*, John Wiley & Sons, New York, NY.

Smart, P.R. and Sadraie, M. (2012), "Applications and uses of dental ontologies", IADIS e-Society.

Upadhyaya, B. (2014), "Composing heterogeneous services from end users' perspective", available at: http://hdl.handle.net/1974/12261

Verma, K., Gomadam, K., Sheth, A.P., Miller, J.A. and Wu, Z. (2005), "The Meteor-S approach for configuring and executing dynamic web processes", (TR6-24-05).

Vu, L.H., Hauswirth, M. and Aberer, K. (2005), "QoS-based service selection and ranking with trust and reputation management", *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE*, Springer, Berlin Heidelberg, pp. 466-483.

Wang, S., Brown, K., Capretz, M.A., Hines, P. and Boyd, J. (2009), "A process oriented semantic healthcare service composition", *Science and Technology for Humanity (TIC-STH), IEEE Toronto International Conference*, IEEE, pp. 479-484.

Weise, T., Bleul, S., Comes, D. and Geihs, K. (2008), "Different approaches to semantic web service composition", *Internet and Web Applications and Services, ICIW'08, Third International Conference on*, IEEE, pp. 90-96.

Yang, J. and Papazoglou, M.P. (2004), "Service components for managing the life-cycle of service compositions", *Information Systems*, Vol. 29 No. 2, pp. 97-125.

Yang, S.J., Zhang, J. and Lan, B.C. (2006), "Service-level agreement-based QoS analysis for web services discovery and composition", *International Journal of Internet and Enterprise Management*, Vol. 5 No. 1, pp. 39-58.

Yoon, K.P. and Hwang, C.L. (1995), *Multiple Attribute Decision Making: An Introduction*, Sage publications, New York, NY, Vol. 104.

Yu, T., Zhang, Y. and Lin, K.J. (2007), "Efficient algorithms for Web services selection with end-to-end QoS constraints", *ACM Transactions on the Web (TWEB)*, Vol. 1 No. 1, p. 6.

Zhang, X., Hou, X., Chen, X. and Zhuang, T. (2013), "Ontology-based semantic retrieval for engineering domain knowledge", *Neurocomputing*, Vol. 116 No. 1, pp. 382-391.

Zeng, L., Benatallah, B., Ngu, A.H., Dumas, M., Kalagnanam, J. and Chang, H. (2004), "Qos-aware middleware for web services composition", *Software Engineering, IEEE Transactions on*, Vol. 30 No. 5, pp. 311-327.

Zunino, A. and Campo, M. (2012), "A survey of approaches to web service discovery in service – oriented architectures", *Innovations in Database Design, Web Applications, and Information Systems Management*, Vol. 107 No. 1.

Corresponding author
Kirit Modi can be contacted at: kirit.modi@ganpatuniversity.ac.in