ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 1, Issue 6, August 2014

VANET Based Adaptive Traffic Signal Control

N.Priya

Assistant Professor, Ramanujam Centre For Computational Intelligence, Department of Computer Science And Engineering, Bharath University, Chennai,Tamilnadu, India

ABSTRACT: In urban scenarios traffic signal controls are the main mechanism to control vehicular flow at the intersections. However, traditional systems fail to adjust the timing pattern in accordance to the time variability. An alternative to such systems is to develop dynamic systems which will alter the timing patterns according to the traffic demand. In this paper, an adaptive traffic signal control system based on car-to-car communication is designed and developed. This system reduces the waiting time of the vehicles at the intersection along with the reduction in queue length. It's also verified that the proposed solution is collision free at the intersections. The proposed system is compared with a classical pre-timed system. The simulations also show that the data convergence time and the communication delay between vehicle and traffic signal do not compromise the efficiency of the system.

KEYWORDS: VANET; Adaptive Traffic Signal Control; Clustering;

I. INTRODUCTION

The number of web services increased vastly in the last years. Various providers offer web services with the same functionality, so for web service consumers it is getting more complicated to select the web service, which best fits their requirements. That's why many research efforts point to discovering semantic approaches for describing web services including both functional and non-functional properties. This will give consumers the opportunity to find web services according to their QOS requirements such as availabity, reliability, response time, trust, etc.

Most of the current solutions are based on web service definition language (WSDL) and Universal Description and Discovery Interface (UDDI) registry. WSDL documents prove functional description of web services without semantic specifications concerning QOS. UDDI registry provides catalog-based searching without control over the quality of registered services. UDDI API's allow publishing and discovering data for a particular service, but do not provide an opportunity for a quality based retrieval.

The problem becomes more complicated when the discovery process returns several web services with the same functionality. Such mentioned disadvantages motivated me to research principle current approaches fro Q)S-aware web service description and discovery in order to find better solution giving more accurate and productive service retrieval.

II. SERVICE ORIENTED ARCHITECTURE

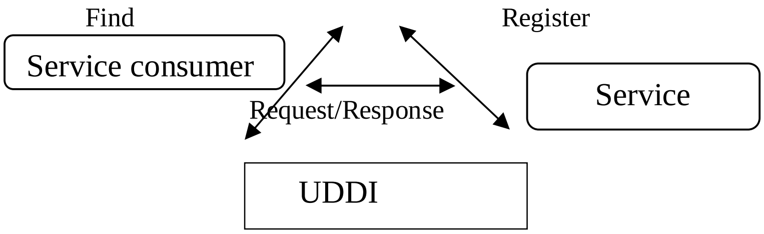
Service oriented architecture is a way of sharing functions in widespread and flexible way. This architecture has been used for years. What distinguishes an SOA from other architectures is loose coupling. Loose coupling means that the client of a service is essentially independent of the service. The way a client (which can be another service) communicates with the service doesn't depend on the implementation of the service. Significantly, this means that the client doesn't have to know very much about the service to use it.

For instance, the client doesn't need to know what language the service is coded in or what platform the service runs on. The client communicates with the service according to a specified, well-defined interface, and then leaves it up to the service implementation to perform the necessary processing.

A web service is a service that communicates with clients through a set of standard protocols and technologies. These web services standards are implemented in platforms and products from all the major software vendors, making it possible for clients and services to communicate in a consistent way across a wide spectrum of platforms and operating environments. This universality has made web services the most prevalent approach to implementing an SOA.

In order for Web Services to be able to work well together, they must participate in a set of shared organizing principles we call a service oriented architecture(SOA).The term service oriented means that the architecture is described and organized to support Web Services dynamic ,automated description, publication, discovery and use.

The SOA organizes Web Services into three basic roles: the service provider, the service requester, and the service registry. The relationships among these three roles are shown below.



Service providers publish (and unpublished) their services to a service registry. Then, service requesters can find the desired web Services by searching for their description at the service registry. Once the requester locates the desired services, its client binds with the service at the service provider and then invokes the service.

The SOA is responsible for describing and organizing the mechanisms and practices for each of these actions. In addition, the SOA is responsible for describing how web Services can be combined into larger services.

Web service in web service repository

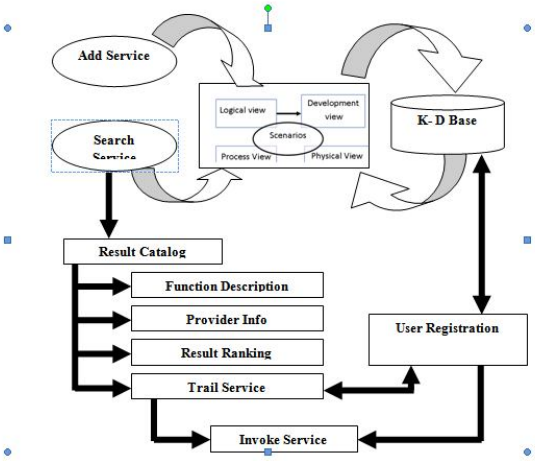
Ws= {Service Key, wsName, wsDesp, QP , OprSet}

- Service Key is the unique identifier;
- wsName represents web service name;
- wsDesp is service functional description;
- QP is published QOS information that is denoted as QP=QN_QD. Where QN represents necessary quality criteria set for all web services and QD represents domain-specific quality criteria set for specific web services.

- OprSet is web operation set denoted as
  OprSet= {opr1, opr2… oprs}.

Where each opri ($1\le i\le s$) can be executed for a specific function task. Based on the QOS requirements of the user, the particular service from the repository is selected and provided to the user.

III. LOGICAL VIEW OF THE WEB SERVICES



IV. MODULES OF WEB SERVICE

QOS calculation:

- QOS values for web services in web service repository is discovered

Selection:

- Selecting web services to meet customers’ QOS requirements

Ranking:

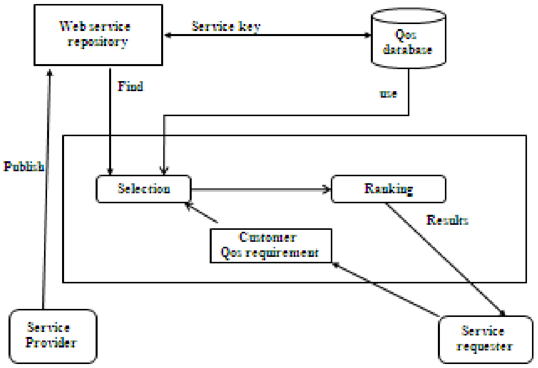
- Selected services are ranked in descending sequence according to the QOS score
- The service is returned to the requester

Updation:

- Refreshes quality criteria value in the QOS database based on the feedback

V. SERVICE SELECTION AND RANKING FRAMEWORK

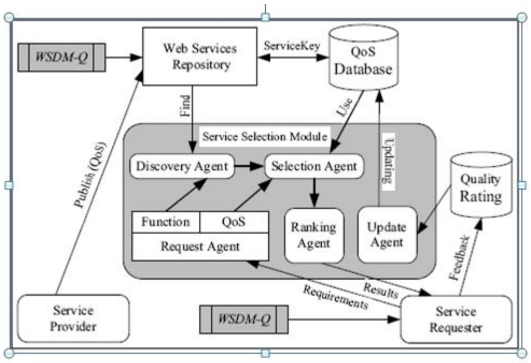
The service selection and ranking module performs initial service selection from web service repository.



Based on the functional requirements, the initial set of services is discovered. Then the service selection is based on the customer Quality of service requirements. The service requesters submit their QOS requirement. The Request Agent which provides interface and communicates with service requester for acquiring functional requirements and QoS constraints. The Discovery Agent which is in charge of finding initial web service set satisfying service requester's functional requirements.

The Selection Agent which collects QOS information from QOS database in terms of initial discovered web service set and then selects web service set fulfilling service requester's QoS constraints. The Rank Agent which is utilized to calculate synthetic QoS score of each selected web services, and then ranks them in a descending sequence according to their QoS marks. Finally, ranked service set is returned back to service requester. The Update Agent which refreshes quality criteria value in the QOS database according to accumulated feedback information in quality rating database.

The selection and ranking framework is designed to support service selection from web service repository and ranking based on the QOS information.



This framework involves

1.The Request Agent which provides interface and communicates with service requester for acquiring functional requirements and QoS constraints.

2.The Discovery Agent which is in charge of finding initial web service set satisfying service requester's functional requirements.

3.The Selection Agent which collects QoS information from QoS database in terms of initial discovered web service set and then selects web service set fulfilling service requester's QoS constraints.

4.The Rank Agent which is utilized to calculate synthetic QoS score of each selected web services, and then ranks them in a descending sequence according to their QoS marks.

VI. SELECTION ALGORITHM

The service providers publishes the web services QoS information. For each service si, its QoS information set is composed of several QoS ternary constraint relations.

QP ={(ci1(qi1, opi1, vi1),…, cim(qim, opim, vim))(ci(m+1)(qi(m+1), opi(m+1), vi(m+1)),…, cih(qih,opih,vih))}

WHERE,

- S Web services repository
- Si A web service, si∈ S
- SD Discovered service set, SD ⊆ S
- SS Selected service set, SS ⊆ SD
- SR Ranked service set, SR ⊆ SS
- qij QoS name at position j of si
- vij Constraint value of qij
- QpPublished QoS information set of si
- QR Submitted QoS requirement set
- cij A constraint relation at position j of si
- ck Request ternary relation at position

Similarly,

- q represents quality attribute name,
- v gives constraint value, and
- op is constraint operator between q and v.
- QR consists of QN and QD.
- constraint operator set {≤,≥} is adopted in ck (1≤k≤n) for service requesters to submit their QoS requirements

```
Algorithm
 Service selection with QoS (SSA-Q).
Input: SD and QoS requirement QR;
Output: Selected web service set SS;
1. S D S ←S;
2. If =NULL R Q then
3. Return SS;
4. Else if NULL N Q ≠ then { // N R Q ⊆Q
5. len← QN.length;
6. SS← SelectWithQoS (SD, QN, len, 1); }
7. If NULL S S ≠ then {
8. len← QD.length; // D R Q ⊆Q
9. SS← SelectWithQoS (SS, QD, len, 0); }
10. Return SS; }
11. Else
12. Return null
```

VII. RANKING ALGORITHM

After the service selection process, r web services are picked out from SD by the Selection Agent. The selected service set is denoted as SS= {s1, s2… sr}.with this input, the ranked service set is generated.

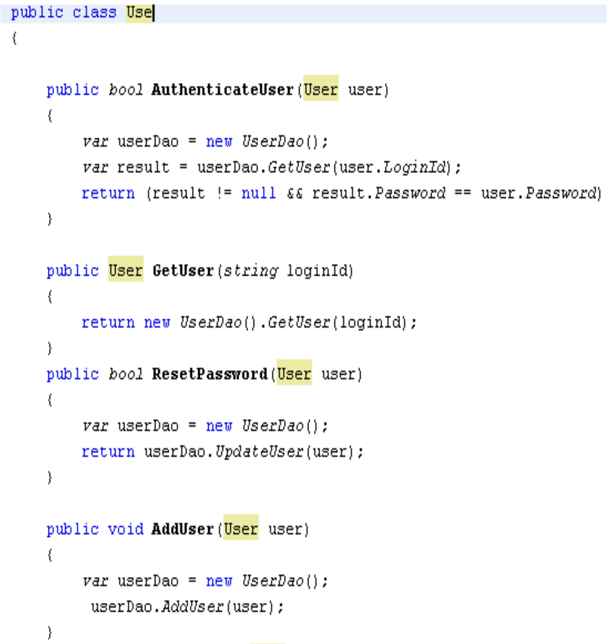
In the first step, QR is taken as benchmark for yielding n columns and r rows are formed by each candidate service si (1≤i≤r). Each row represents a candidate service, and each column contains QoS values of a quality attribute in a QR's ternary constraint relation. i.e., qosij is generated by service si and cj (qj, opj,vj). If there exists a cuw(quw, opuw,vuw) sharing the same quality name with cj, vuw is used as qosij's value. Otherwise, qosij is set 0.

In the second step, each matrix element in MS is normalized with metrics function and mapped in the range of [0,1]. For each qosij, its normalized value qos′ij is calculated by qmax and qmin (maximum/minimum value in column j). All QoS values of column j are normalized in a monotonically increasing way. Otherwise, it is measured in a monotonically decreasing

Finally, for each candidate service si (1≤i≤r) in SS, its synthetic QoS value is calculated based on weight array W and normalized quality criteria matrix where each row corresponds to a service. Then, we rank and append them into SR according to their comprehensive QOS marks.

Service ranking with QoS (SRA-Q).

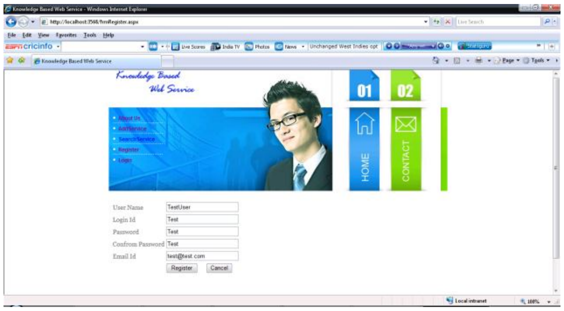
```
Input: SS, QR and quality criteria weight array W;
Output: Ranked web service set SR;
Step 1: generate quality criteria matrix MS.
    Input: Ss, QR and quality criteria weight array w
    Output: Ranked web service set SR
    Calculate and rank each service's QoS value.

    1. For i← 1 to r do {
    2. qSumi ←Σk-1 to n ( wk * Ms [i k]) ;
    3. SR .rank (qSumi si) ; }
    4. Return SR;
```



```
public class Use
{

    public bool AuthenticateUser(User user)
    {
        var userDao = new UserDao();
        var result = userDao.GetUser(user.LoginId);
        return (result != null && result.Password == user.Password)
    }

    public User GetUser(string loginId)
    {
        return new UserDao().GetUser(loginId);
    }
    public bool ResetPassword(User user)
    {
        var userDao = new UserDao();
        return userDao.UpdateUser(user);
    }

    public void AddUser(User user)
    {
        var userDao = new UserDao();
         userDao.AddUser(user);
    }
```

Fig: Function call for each service

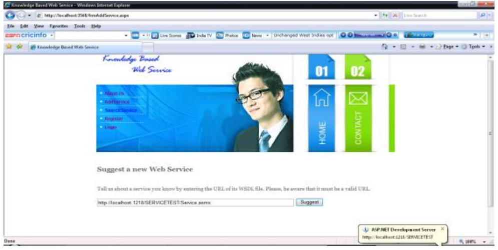
VIII. EXPERIMENTAL VIEW OF WEB SERVICE



Knowledge Based Web Service - Windows Internet Explorer
http://localhost:3568/frmRegister.aspx
File Edit View Favorites Tools Help
Knowledge Based Web Service
Knowledge Based Web Service
About Us
AddService
SearchService
Register
Login
01
02
HOME
CONTACT
Local intranet

User Name TestUser
Login Id Test
Password Test
Confrom Password Test
Email Id test@test.com
Register Cancel

Fig: User Registration view



Knowledge Based Web Service - Windows Internet Explorer
http://localhost:3568/frmAddService.aspx
File Edit View Favorites Tools Help
Knowledge Based Web Service
Knowledge Based Web Service
About Us
AddService
SearchService
Register
Login
01
02
HOME
CONTACT
Done

Suggest a new Web Service

Tell us about a service you know by entering the URL of its WSDL file. Please, be aware that it must be a valid URL.

http://localhost:1218/SERVICETEST/Service.asmx Suggest

ASP.NET Development Server
http://localhost:1218/SERVICETEST

Fig: suggestion web service

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 1, Issue 6, August 2014

IX. CONCLUSION

In this paper we presented a detailed road-map for innovation for web service for dynamic integration for project development as well service selection based on the web service quality of service parameters through web site.

The first stage of the development was producing service registration that provides the discover the service for consumer needs.

The development perspective the code re writing for the same operation are unavoidable one. This research provides an initiative of development can dynamically use the existing developed components for the required features.

This research main aspect is provide the service selection is based on service quality of service parameters like availability, response time , connection time, cost.

REFERENCES

1] F. Curbera et al. Unraveling the Web Services: an Introduction to SOAP, WSDL, and UDDI, IEEE Internet Computing, Mar/Apr issue 2002.

[2] Cardoso J., Sheth A., Miller J., Arnold J., Kochut K. Quality of service for workflows and Web service processes, Journal of Web Semantic, 2004, 13：281~308.

[3] Zeng, L., Benatallah, B., Ngu, A.H.H., Dumas,Kalagnanam, J., Chang, H., Chang H. QOS-aware middleware for Web services composition,IEEE Transactions on Software Engineering,2004,30(5)： 3l1~327.

[4]Hefeng Cao, Xingzhi Feng, Yang Sun, Zhiwei Zhang, Quanyuan Wu. A Service Selection Model with Multiple QOS Constraints on the MMKP, 2007 IFIP International Conference on Network and Parallel Computing - Workshops.

[5]KO, Chang Ouk Kim, Ick-Hyun Kwon. Quality-of service oriented web service composition algorithm and planning architecture, The Journal of Systems and Software.81 (2008)2079C209.

[6] H. C.-L and K. Yoon. Multiple Criteria Decision Making, Lecture Notes in Economics and Mathematical Systems, Springer-Verlag, 1981.