Ontology Based Semantic Web Service Discovery and Composition Framework

T.Suganya

Department of Computer Science and Engineering
IFET College of engineering, Villupuram, India
suganya9412@gmail.com

R.Rajmohan

Department of Computer Science and Engineering
IFET College of engineering, Villupuram, India
rjmohan89@gmail.com

Abstract- **Web Service Discovery is the process of finding a suitable Web Service for a given task. A theoretical analysis of ontology-based service composition is proposed in terms of its dependency with service discovery. Web Service composition technique provides the features to users that an individual web service cannot perform. At service composition time, the composition of service depends on the users input outputs parameters and other non-functional parameters. In order to evaluate the best approach an optimal composition search algorithm to extract the best composition from the graph minimizing the length and the number of services and different graph optimizations to improve scalability of the system. The implementation of Floyd-Warshall algorithm can find all the possible way of services integrated and services to the web tasks. We have designed a full frame work for good integrated services for the particular tasks and given good performance of the web.**

Keywords - Ontology, Web service, Semantic web, Discovery, Composition

I. INTRODUCTION

Composition [3] means that the process of joining existing web services together based on the composition rules to satisfy the demand of the user which cannot be satisfied by single service. Service discovery [8] means that the process of finding out the suitable services when users looking for service. Both discovery and com-position engines essentially rely on the processing of service descriptions, which increasingly go beyond syntactic representations to include the semantics of service to enable more advanced computations. In service discovery and composition framework, it requires the considerable effort to perform the complex task (when the number of service is vast in amount).

In the previous work they had use the fine grained services to provide a convenient mechanism to discover the service able to produce the certain type of data during composition. fine grained services is used only to handle the certain amount of services.it sets the target values before providing a service to the given task. When the number of services exceed the range then it cannot handle the task, the output obtained at irregular intervals of time and the output obtained is not optimal.

In order to tackle the previous problems, a composition and discovery framework should consider the following characteristics:

1) To provide a convenient coarse grained discovery mechanisms that could help to discover services able to produce certain types of data as usually required during composition
2) To improve the response time of service discovery to process requests very fast
3) To improve the scalability of the overall composition process
4) Find optimal service compositions by minimizing number of services & length of the composition to avoid complex task.

In this paper we present an ontology-based framework focused on the semantic input-output parameter matching of services. Ontology framework deals with the semantic heterogeneity in structured data. We deployed Floyd–Warshall algorithm it focuses on the all possible way of matching services of web service composition interfaces to give the best services.

II. RELATED WORKS

A. iServe : a Linked Services Publishing Platform

Despite the potential of service-orientation and the efforts devoted so far, we are still to witness a significant uptake of service technologies outside of enterprise environments. A core reason for this limited uptake is the lack of appropriate publishing platforms able to deal with the existing heterogeneity in the service technologies landscape and able to provide expressive yet simple and efficient discovery mechanisms. A novel and open platform for publishing services, called iServe [1] is proposed for providing better service discovery. The disadvantage is Scalable algorithms to synthesize the required control structures are not easy to provide. It provides an initial solution to the generation of such complex plans.

B. Web Service Composition - Current Solutions and Open Problems

Composition of Web service [2] has received much interest to support business-to-business or enterprise application integration. On the one side, the business world has developed a number of XML-based standards to formalize the specification of Web services, their flow composition and execution. This approach is primarily syntactical: Web service interfaces are like remote procedure call and the interaction protocols are manually written. On the other side, the Semantic Web community focuses on reasoning about web resources by explicitly declaring their preconditions and effects with terms precisely defined in ontologies. For the composition of Web services, they draw on the goal-oriented inferencing from planning. So far, both approaches have been developed rather independently from each other.

C. Service-Oriented Computing: State of the Art and Research Challenges

The service-oriented computing (SOC) [6] paradigm uses services to support the development of rapid, low-cost, interoperable, evolvable, and massively distributed applications. Services are autonomous, platform-independent entities that can be described, published, discovered, and loosely coupled in novel ways. They perform functions that range from answering simple requests to executing sophisticated business processes requiring peer-to-peer relationships among multiple layers of service consumers and providers. Any piece of code and any application component deployed on a system can be reused and transformed into a network-available service. The main disadvantage of Service-oriented computing is a vast and enormously complex subject, embracing many technologies that must be integrated in an intricate manner.

III. SYSTEM ARCHITECTURE

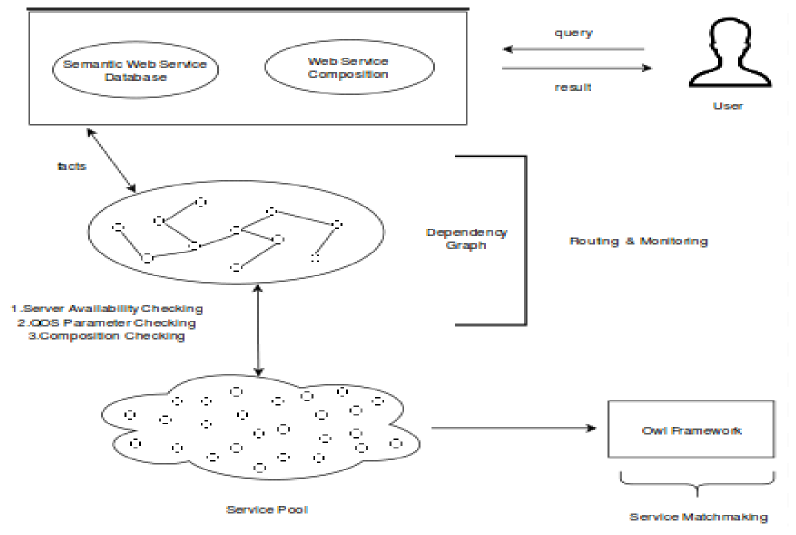

Fig.1.System Architecture

The system architecture is well emphasized in the Fig.1. When the user requests a query, the query may contain the functional and non-functional parameters. In Semantic web service database, semantic discovery and semantic matchmaking will be performed to find out the optimal solution for the user request. Semantic discovery enables the user to search repositories of composable functionality, which is based on functional and non-functional parameters. Semantic matchmaking [5] depends on semantic reasoning to find the relationship between the concepts. Based on the events a dependency graph will be generated, which is LDAG (Layered Directed Acyclic Graph). Optimization is applied to reduce the size of the graph .Routing and Monitoring takes place to improve the optimal composition performance. OWL framework enables the users to automatically determine, appeal, comprise and display web resources offering service under some listed constraints. Floyd Warshall algorithm [10] is deployed here. This algorithm is used to find the shortest path in the weighted graph. It offers length of the path between all pair of vertices. To find the path with maximum values optimal routing is performed.

Algorithm: Floyd Warshall

```
Begin
        n= W.rows
```

```
    $D^{(0)}$= W
    For K= 1to n
    Let $D^{(k)}$= ($d_{ij}^{(k))}$) be a new matrix
    For i= 1to n
    For j= 1to n
    $d_{ij}^{(k)}$= min ($d_{ij}^{(k-1),}$ $d_{ik}^{(k-1)}$+ $d_{kj}^{(k-1)}$)
    return $D^{(n)}$
End
```

IV. SYSTEM IMPLEMENTATION

A. Data Extraction And Preprocessing

When the user gives a query, WSDL document extract the data which is able to search the relevant web services from the repository of web service document to create a document index. To speed up and optimizing the performance document index is used .The search is performed efficiently by searching the query terms instead of searching the keywords. Discovery search is performed by UDDI [9]. Preprocessing is performed to remove trivial words from a document before any text is used on the data. By Service matching documents are riddled to remove duplicate web services. Semantic Discovery is used to discover the relevant service includes the preconditions and effects. Preconditions are said to be as condition should be true before the execution of the process. Effects are used to describe the environment after the conditions are executed.

B. Semantic Discovery

Semantic Discovery allows the functionality to the users to search the repositories of composable functionality. Composable functionality contains both the functional and non-functional parameters. Functional parameters contain input and output parameters along with preconditions and effects. Non-functional parameter contains access control, price, accuracy, quality of service. The following algorithm is deployed for retrieving input relevant set of services.

Algorithm: Retrieving input-relevant and output-relevant sets of services

```
method RELEVANT($Z \subseteq O$, W ,type)
relevantService :={}
for all w = {$I_{Wi}$, $O_{Wi}$} Є W do
if type = Int_Service then
if match(Z, $I_{Wi}$)then
relevantService := relevantService  U $w_i$
end if
else if type = Out_Service then
if match($O_{Wi}$, Z)then
relevantService := relevantService U $w_i$
end if
end if
end for
return relevantService
end method
```

C. Semantic Matchmaking

To generate the composition and discovery services fundamental functionality is required. The ability to analyze the compatibility between different semantic types is said to be as semantic matchmaking. The matchmaking process is carried out by using the OWL frame work [5]. Restful Services is the web based architecture; the protocol which used for the data communication in the REST is HTTP protocol. HTTP standards are used to access the resources. Web crawler is used for the purpose of web indexing. To utilize the logic reasoning and token based IR similarity measure OWLS-MX [5] is used. To retrieve different services from the relevant query for OWL-S, OWLS-MX is applied. SPARQL [4] explains the data access protocol and standard query language to use with RDF. Any data sources can be mapped with RDF by SPARQL. To organize the distributed application action WSC delivers framework, WSC is a Web Service Specification.

D. Service Composition Graph Generation

Service Composition generates the graph when the system receives the request. The graph describes the semantic relation between the existing services for the user request. The request denotes the input and output concepts. An input concept denotes the initial set of inputs and output concepts denotes the composite service. The composition graph generated is Layered Directed Acyclic Graph (LDAG). The service composition and graph generation process is depicted in figure 2.

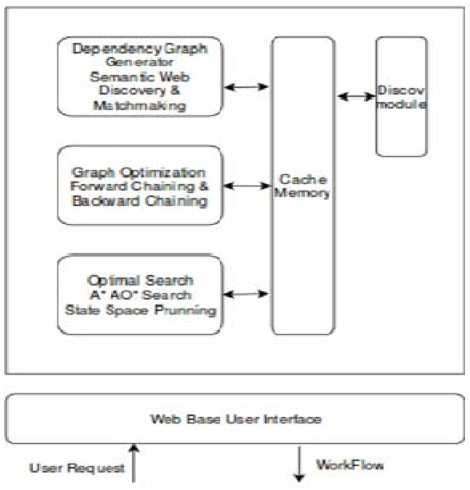
Dependency Graph Generator Semantic Web Discovery & Matchmaking
Graph Optimization Forward Chaining & Backward Chaining
Optimal Search A* AO* Search State Space Prunning
Cache Memory
Discov module
Web Base User Interface
User Request
WorkFlow

Fig.2.Composition Framework

E. Graph-Based Optimization

To reduce the size of the graph different graph optimization technique is used. The optimal composition search performance is improved. Both the composition and discovery phase are independent to each other. For optimal composition search all the information in the graph are required. Information states the relevant services and semantic relation between the input and output services.

F. Optimal Composition Search

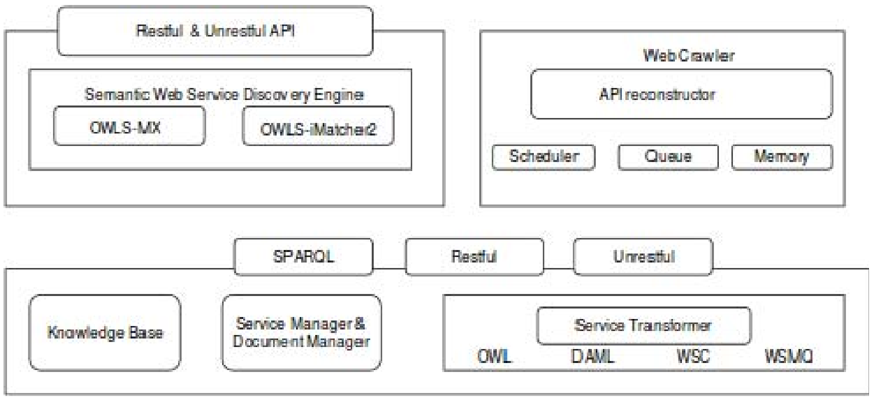

Fig.3.Discovery Framework

The composition search is performed to find the best composition from the available composition by satisfying the input and output request which is depicted in fig 3. The search in the graph is performed both in the forward and backward directions. We have devised a reference implementation of the framework on the basis of two pre-existing separate components, namely iServe [1] and ComposIT [7].Dependency graph generator generates the directed graph, which represents the dependencies of numerous objects towards each other. Semantic web encourage common data formats and exchange protocols on the web. Data formats generally represent the Resource Data Format (RDF).Semantic Web Discovery enables the user to search for composable functionalities. Semantic matchmaking is used to find the relation between the relevant services based on the semantic reasoning. Graph optimization is used to analyze all the possible outcomes and find out the best from it. Optimal search is the binary search, the search is performed until it finds the smallest time for the possible sequences. A* algorithm is used to find the shortest path and it is flexible, it can be used for wide range of contexts and AO* search is used to find the one solution, which is the best optimal solution. The process of searching process is explained in the figure.4.

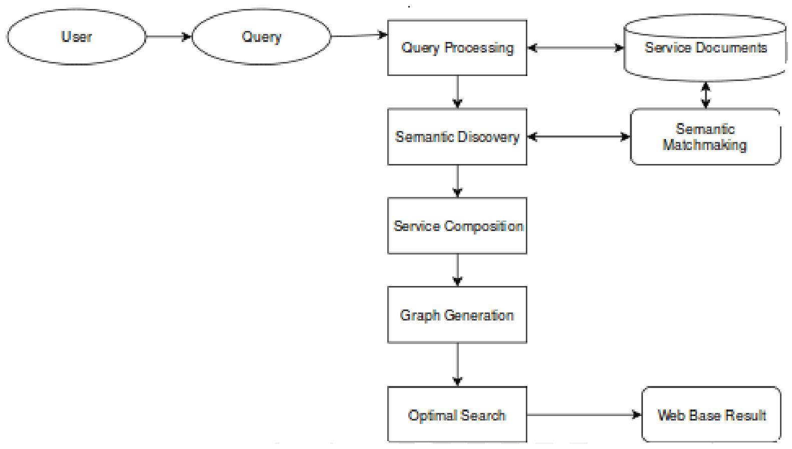

Fig.4.Optimal searching

V. EXPERIMENTAL DESIGN

In order to perform a standard and comparable evaluation we selected the OpenFlight dataset [11]. The OpenFlight dataset scenario is explained in the fig .These datasets allow us to measure the scalability with an increasingly large set of services (from 158 t0 8,119 services). Services were imported to iServe [1] using a specific transformer plugin which translates each service description in the OpenFlight XML format into MSM, and the XML concept taxonomy into an equivalent OWL-MX representation. The Open Flight database scenario is explained in figure.5.

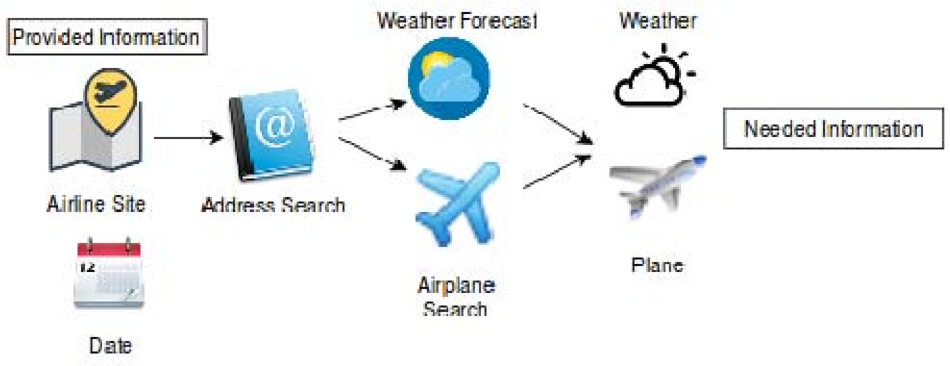

Fig.5. OpenFlight Database

Experiments were run under Fedora 20 64-bit on a PC with i3 processor at 3.06 GHz and 8 GB of RAM. OWLM-Lite 5.4 with OWL Horst reasoning was chosen in iServe as the RDF triple store for the semantic register and deployed within Glassfish server. The parameters and the concepts used in OpenFlight dataset are explained in the Table 1.

TABLE 1. OpenFlight Dataset

Web service & user request	**Input**	**Output**	Description
Request(R)	Airline site(A) Date (B)	Weather(C) Plane name (D)	The request provides tourist site, date and needs the weather and airline of the city
Address search (A1)	Airline(A)	City(E)	It finds the city which the tourist site in.
weather forecast (A2)	City(E) Date(B)	weather (c)	It returns the weather of the city on the date.
Plane search (A31,A32)	City(E)	Planes(D)	It finds airline in the city.
weather forecast (A4)	Airline(A) Date(B)	weather(c)	It returns the weather of the tourist site on the date.
Plane locate (A5)	Plane site(A)	Planes(D)	It finds airline near the tourist site.

VI. RESULT EVALUATION

TABLE 2. Characteristics of the OpenFlight Datasets

Dataset	#Serv.	#Con.	#Serv.Sol.	Length
OpenFlight’01	128	1,100	5	2
OpenFlight’02	348	1,195	3	2
OpenFlight’03	204	2,389	20	13
OpenFlight’04	941	1,035	3	4
OpenFlight’05	1,190	8,228	8	6
OpenFlight’06	1,198	1,007	10	7
OpenFlight’07	2,213	10,237	8	10

Table 2 shows the characteristics of each OpenFlight dataset. The number of services and concepts in the ontology of each dataset are shown in columns #Serv and #Con. Respectively. Experiments were done using one instance of iServe in order to measure the effect of the Discovery/Matchmaking over the whole composition process.

TABLE.3 Discovery and Composition

	Discovery/Matchmaking(D/M)						Composition		
	1) SPARQL		2)Index D/SPARQL		3) Full Indexed D/M				
Dataset	***G.size***	***G.time (a)***	***#SPARQL***	***G.time (a)***	***#SPARQL***	***G.time (a)***	***#SPARQL***	***G.size(opt)***	**Comp.time (A)**
OpenFlight’01	25	17.42	2256	3.57	524	0.12	0	12(-27%)	0.06
OpenFlight’02	25	53.80	5349	10.56	1630	0.35	0	10(-27%)	0.04
OpenFlight’03	95	152.70	34619	10.00	2984	0.67	0	30(-18%)	0.11
OpenFlight’04	34	116.10	11828	11.11	2281	0.49	0	15(-37%)	0.10
OpenFlight’05	87	303.40	39148	16.00	2417	0.54	0	32(-34%)	0.15
OpenFlight; ’06	169	851.10	89682	38.11	8011	1.02	0	55(-20%)	1.00
OpenFlight’07	104	983.10	100881	25.66	6176	1.20	0	60(-36%)	0.13

Results with each configuration are shown in Table 3. The second column shows the size (number of services) of the resulting composition graph for each dataset. The next columns show the time taken to generate the composition graph (G.time) in seconds and the number of SPARQL queries generated during the process. The last three columns show the size of the graph after the graph-based optimization, the time of the composition search (graph optimization + optimal A* backward search) and the number of services and length of the optimal composition found.

The analysis of these results reveals that the discovery and matchmaking phases take most of the time of the composition, even using the optimal configuration to avoid the latency of the SPARQL queries. This is graphically represented in Fig. The discovery and matchmaking are responsible for the majority of the computation that needs to be performed to compose services.

VII. CONCLUSION

A hypothetical analysis of service composition in terms of its dependency with service discovery has been discussed in our work. We have defined a formal integrated graph-based composition framework anchored on the integration of service discovery and matchmaking with composition process. Our empirical analysis shows that with the satisfactory interface and indexing, discovery engines can support high efficient composition. The scalability and flexibility of our proposal shows how integrated composition systems can be designed in order to achieve good performance in real scenarios and distributed and controlled by diverse organizations.

REFERENCES

[1] C. Pedrinaci, D. Liu, M. Maleshkova, D. Lambert, J. Kopecky, and J. Domingue, "iServe: a linked services publishing platform," in CEUR Workshop Proceedings, vol. 596, 2010.
[2] B. Srivastava and J. Koehler, "Web Service Composition - Current Solutions and Open Problems," in ICAPS 2003 workshop on Planning for Web Services, 2003, pp. 28–35.
[3] J. Rao and X. Su, "A Survey of Automated Web Service Composition Methods," in Semantic Web Services and Web Process Composition, vol. 3387, 2004, pp. 43–54.
[4] K.Jayasri, R.Rajmohan, D.Dinagaran, ".Analyzing the query performances of description logic based service matching using Hadoop", IEEE conference, DOI:10.1109/ICSTM.2015.7225382, pp. 1-7, May 2015.
[5] R.Rajmohan, N.Padmapriya, "A Domain ontology based service matching for CHORD based Super Peer network", IEEE conference, DOI:10.1109/ICDSE.2012.6282324, Cochin University, pp. 214-219, July 2012.
[6] M. P. Papazoglou, P. Traverso, S. Dustdar, and F.Leymann, "Service-Oriented Computing: State of the Art and Research Challenges," Computer, vol. 40, no. 11, pp. 38–45, 2007.
[7] P.Rodriguez-Miler, M.Mucientes, J.Vidal, and M.Lama, "An Optimal and Complete algorithm for Automatic Web Service Composition, " Int. J. of Web Services Research(IJWSR), vol.9m no.2, pp. 1-20, 2012.
[8] G. Alonso, F. Casati, H. Kuno, and V. Machiraju, Web Services, ser.Data-Centric Systems and Applications. Springer, 2004.
[9] R Rajmohan, N Padmapriya, SKV Jayakumar, "A survey on problems in distributed UDDI", International Journal of Computer Applications, Volume 36– No.3, December 2011.
[10] Floyd Warshall Algorithm, https://en.wikipedia.org/wiki/Floyd–Warshall_algorithm.
[11] Open Flight datasets, https://datahub.io/dataset/open-flights.