International Journal of Advanced Technology in Engineering and Science www.ijates.com
Volume No.03, Issue No. 01, January 2015 ISSN (online): 2348 – 7550

A NOVEL AUTOMATIC WEB SERVICE COMPOSITION FRAMEWORK FOR USER DEPENDENT WEB MINING

Mrs. K. Krishna[1], Dr. D. Murugan [2]

[1]Department of Computer Application, Vivekananda College., Tamil Nadu, (India)

[2]Department of Computer Science and Engineering, Manonmaniam Sundaranar University, (India)

ABSTRACT

Web data which started initially as a repository of information has evolved into a host providing multiple contents and services like map-finding, weather-reporting, e-commerce etc. Real-time applications sense temperature and monitor traffic. Businesses and government integrated data into existing Web applications to provide new services. Web service mining, relatively new area of research is a process aimed at discovering interesting and useful compositions in existing Web services. They are useful since primary goals are undefined. Web services do not assume prior knowledge and rely on component services to explore data with varied results. Organizations which realized this potential took advantage by sharing their data on the Web. This Paper presents a new approach of Web service mining in, a General Purpose Web Composition Framework for Web services.

Keywords: Composition Automatic Web Compositions, General Compositions, Framework, Web Composition, Web Mining

I. INTRODUCTION

Compositions incorporate information from various sources and domains to achieve a task in a Service-Oriented Architecture (SOA). A Single service becomes the basic unit of operation and may be unable to cater to all functional requirements. The services when connected can satisfy complex requirements. Different functional requirements require different compositions . Traditionally, web service composition has been performed manually, making it a difficult and error prone task. Though manual Web compositions are time-consuming and error prone, automatic compositions for composing services into a coherent task may not be available readily. It is tricky to find a single web service that produces the desired output from inputs. A composite service with process logic can be decomposed into subtasks that respond to web services. The Figure 1 is an example of a workflow derived from high-level specifications. The task is subdivided into sub tasks and binding them to single web services based on the logic of the workflow [1]. This paper examines various approaches and proposes a general-purpose framework for automatic service composition.

II. MANUAL WEB SERVICE COMPOSITION PROCESS

In a manual composition a user creates an abstract specification of a high-level task. User manipulation provides autonomy and control over execution but the user needs to possess enough domain knowledge to decompose the specification and create a workflow. An abstract composite workflow is depicted in Fig. 1. Users utilize standard web service languages like WS-BPEL [2] or [3]. The user then binds web services to the workflow. Such methods are widely used in search and discovery services in a web service repository like UDDI [4]. The services will have to be monitored to check if the service is bound to the task while handling faults as they arise. Thus user domain knowledge plays a primary role in such compositions. It is a time-consuming and error-prone process with no guarantees making it imperative to automate compositions.

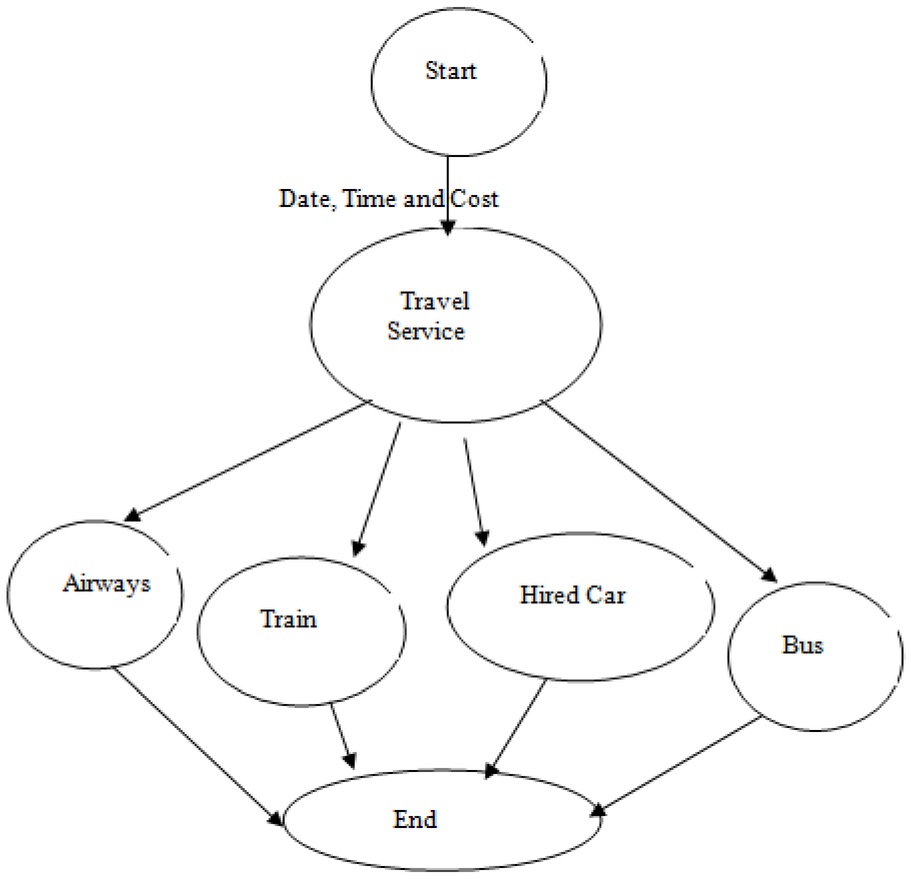

Fig. 1 Abstract Web Composition Workflow

III. AUTOMATIC WEB SERVICE COMPOSITION

Automatic web composition is a five phase process involving specification, planning, validation, discovery and execution. Specification is specifying user goals, requirements or constraints. Languages used for specifying abstract specification can range from OWL-S, WS-BPEL [5]. Planning is automatically composing an abstract workflow while Validation is using techniques to ensure proper outputs. AI planning techniques can generate automatic processes with planning tools like WS-BPEL, OWL-S, or use proprietary languages which provide logic validation like PDDL, CSSL. Validation of an abstract composite workflow is validating syntactically and checking the workflow for proper structures based on the requirements. Many validation algorithms for semantic validation are based on utility theory and path heuristics [6] or check models [7]. Discovery is discovering services which satisfy task specifications. Discovered services are bound to the corresponding subtasks in the workflow. Discovery tools are an open-industry initiative in XML and UDDI (web service repository). The request is then executed using the created composite services. If a service fails during the execution, the recovery can be done using a substituted web service. The substituted service should support original service functionalities or can change the context of the composition [8]. SOA enables dynamic service binding allowing discovery, selection and invocation of services at runtime in Web services technologies [9]. Web services are programmatically accessible and interoperate independently thus offering flexibility and scalability needed by compositions. Such automated compositions can enrich the semantically enabled service-oriented architectures. Fig. 2 depicts the phases in automatic web compositions while Table 1 lists the Comparisons between Web Service Compositions.

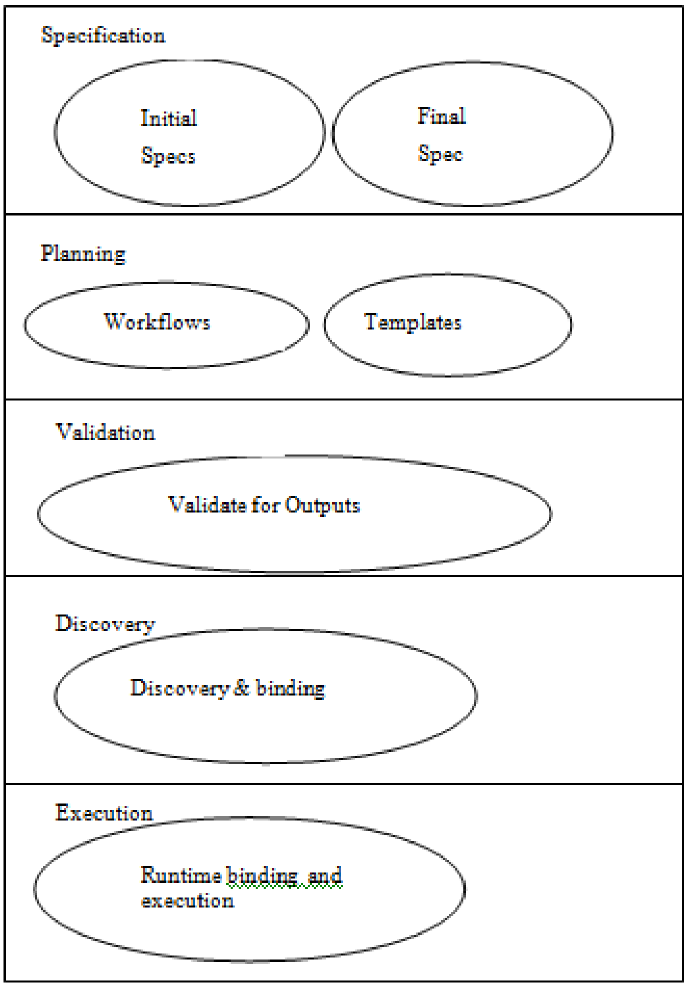

Fig. 2 Automatic Composition Phases

Table 1 - Comparison between Web Service Compositions

	BENCHMARKS			
Approaches	Connectivity	Exception Handling	Scalability	Correctness
E - Flow	Low	Low	Low	Low
PPM	Low	Low	Low	Low
BPEL4WS	Good	Good	Average	Low
OWL - S	Good	Average	Good	Low
WSMO	Good	Good	Good	Low

IV. GENERAL PURPOSE WEB COMPOSITION FRAMEWORK (GPWCF)

GPWCF is a different angle for web compositions. It is a framework which has five stages and can be implemented by organizations. The user selects a service and sets constraints. The required service is searched in the services repository and selected. In case a corresponding service is found or not listed, an error message is sent to the user and the error is trapped in the error log. A scope definition is constructed on the selected service and user defined constraints. The scope definition also determines the search space and the return of information sets to the user. All errors like Errors on execution of a service, errors in fetching of results, network failures are logged in the error log, enabling improvement of the compositions in future versions. The error logging is a simple and useful utility not found in many compositions. The GPWCF can be used by organizations as a starting point for web services compositions and developed for future requirements. Fig. 3 depicts the GPWCF framework.

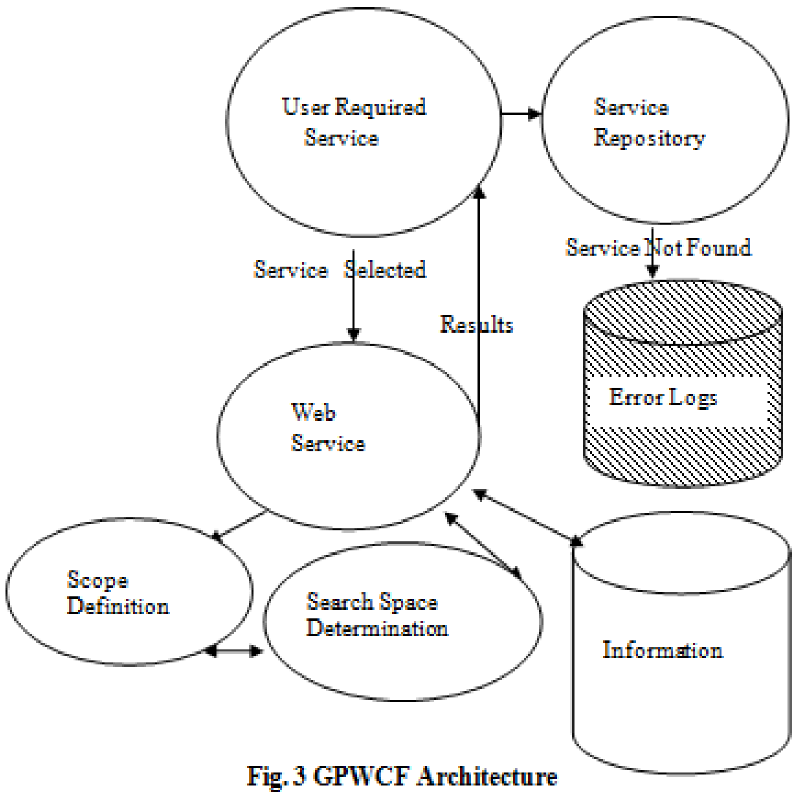

Fig. 3 GPWCF Architecture

4.1 GPWCF Scope Definition

The mining process begins with an evaluation of the user request. User constraints are considered for the specified request and defined as a list of functional activity in the scope specifications of the service. For Example a user request for a cheap travel with an upper limit of 1000, is a constraint to be considered. The scope definition would build a constraint and filter all modes of travel, up to 1000 rupees. Similarly each constraint or request may pertain to a different domain with separate functional parameters. The mining perspective M_P can be formally defined as.

$$M_P = \{f(C) \mid wd \in WD\} \quad (1)$$

where mining perspective is M_P, *WD* is a set of Web service domains, *C* is a set of constraints and f*(C)* is the constraints evaluated in the domain WD. Assume the symbol → denotes *refers to and* the set of all Results referred to in M_P, can be denoted as $R(M_P)$ and calculated using:

$$R(M_P) = \{R \mid \exists f \in C \wedge f \rightarrow Rs\} \quad (2)$$

4.2 GPWCF Search Space Determination

The search space for any mining operation can be defined from its scope. The resultant output of a search space determination phase is a function library from Web service functions of a web service registry WR that are involved in mining perspective M_P, library Lb and set of operations o, defined as .

$$Lb = \{o \mid o \in WR\} \quad (3)$$

4.3 GPWCF Service Selection

The most suitable web service is selected from the composite web service and executed. Selection of relevant web services is performed by matching inputs of the user request to the web services which are operation unique [10] and so the required services are selected from the composite. The service selection can be defined as

$$S_P = \{f(p) \in P\} = O_p \quad (4)$$

where S_P is the Services Perspective , *P* is a set of perspectives of Web services in the directory, f*(p)* is the chosen service from a the select list of all services in the Directory and O_p is the operations matching the scope perspective

.Input: Service Selection $S_P = \{f(p) \in P\}$

Output: Selected Service f(p)

```
    for all $P \in S_P$ do
        if service = f(p) do
            add service to $O_p$;
        end if
    end for
  if $O_p = 0$ then
     add to error_log
     return failure
    end if
return $O_p$;
```

4.4 GPWCF Function Selection with Filtering Based On Mining Perspective

Algorithms can select the functions required for mining and be a medium for the Web service. The filtering algorithms can be applied to the mining Perspective.

Input: Mining context $M_P = \{f(P) \mid d \in D\}$, Web service registry WR.

Output: Function library RLi.

Variable: Operation Perspective $O_P(M_P)$

```
    for all $f(P) \in M_P$ do
        for all $O_P \in f(P)$ do
            add $O_P$ to $O_P(M_P)$;
        end for
    end for
    for all $r \in WR$ do
        if $r = O_P(M_P)$ then
        add r to RLi;
        end if
        end for
        if operations count  = 0 then
            add to error_log
           return failure
        end if
return RLi;
```

V. SERVICES INTER-RELATIONS

Two services of a web service composition have a relationship if they can be plugged together to perform a value added service or if one of the service can be substituted by the other. Relationships between services can be of the following types.

- Service A → Service B   // Calling the next service
- Service A // Service B   // parallel execution
- Service A ←→ Service B // Service substitution
- Service A ϶ Service B   // Inclusion

In a relationship between Services the first service has to finish before the other starts. In parallel processing both execute in parallel but combine the results. In substitutions the services, substitute the functionalities with each other, functionally. In Inclusion, one service includes the services functionalities. Web compositions use JAXRPC and Java RMI services.

VI. RESULTS

Several parameters need to be considered while evaluating a Web Composition like Exception Handling, Accuracy and constraint attributes of the composition. . Connectivity and Reliability in are the basic requirements of a Composition and in its absence even the best composition cannot function. Exception handling must deal with exceptional cases of failure for the system to adapt in future versions. Accuracy pertains to the behavior of a composition and in matching the results. Constraints are user requirements passed as parameters to produce the end results in a composition service. Automated compositions are perceived to be intelligent processes with specifications on services and requirements, though design involves human intervention. Performance evaluation of Web services can help implementers understand the behavior of the services in a composed process[11]. The performance of a single Web service can potentially affect the overall performance in a composition. It is customary to evaluate the performance of the services within a process before making it line for usage. A common approach to obtain performance results of a given Web service is by performance testing from a user‘s perspective the total execution time of a process is a measure of its efficiency. When a Web Service application client connects to a service, submits a request and terminates it is unlikely to load server‘s CPU, or the network. Such services perform well compared to distributed object systems. Table 2 shows relative performances for three implementations an application with a request for a single data structure retrieved according to an argument key and Fig. 3 shows the Throughput of different technologies graded against time factor and depicts the performances of different technologies in web service composition in terms of throughput network conditions and response times.

Table 2 Approximate Costs of Single Web requests in Technologies

Technology	Latency	Packets	Data transfer in Bytes
WS	0.15s	16	3338
CORBA	0.43s	8	1111
Java RMI	0.31s	48	7670

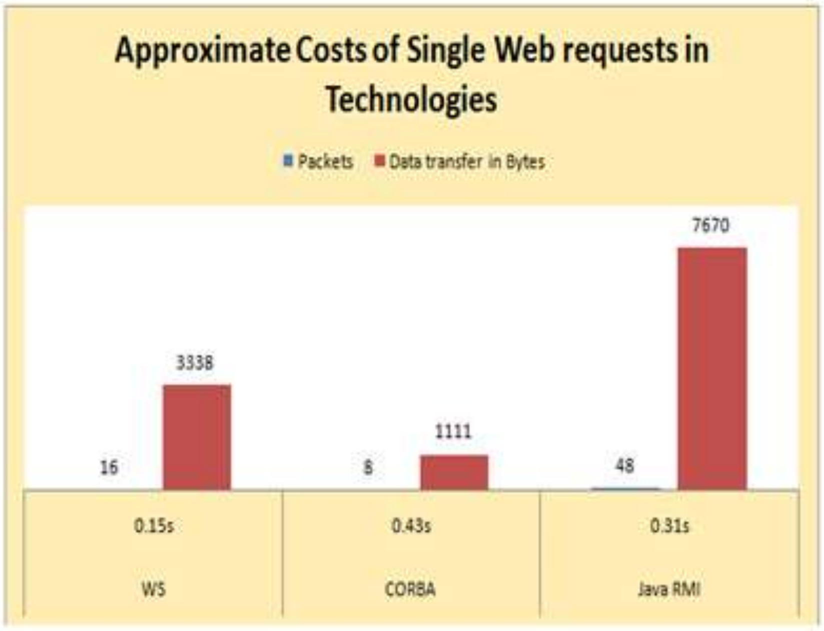

Fig 3 Throughput of Technologies

VII. CONCLUSION

Web services mining has advantages in many fields and more so in the field of business intelligence. Process mining can be used for automatic execution of web services and interoperations between web services. Efficient mining algorithms and techniques are required to analyze and infer from the data collected from process logs. This Paper proposes a new and novel framework for automatic web compositions. This framework can be adapted in any technologies listed above in Fig. 3 for better or faster throughput. Manual web service composition can be error-prone and a tedious process. Automatic web service composition frees the configurer from the burdens of manual composition. Real-world applications need to find an acceptable middle stage between manual and automatic composition that relieves the user from the complexity but gives the user enough involvement and confidence for a predicted end result. Though there are many approaches to web service compositions, there are very little proper solutions for the approaches. Guidelines on composition processes are also varied. The approaches suggested enable the proactive discovery of interesting pathways for web services. The suggested Web service mining framework allows the mining of interesting composite services to be carried out with the presence of execution logs. The framework can be a base for future works and development.

REFERENCES

[1] Anya Kim, Myong Kang and Catherine , "A Framework for Automatic Web Service Composition", Naval Research Laboratory, Washington, DC 20375-5320, NRL/MR/5540--09-9191, April 30, 2009.

[2] Martin, D., et al., OWL-S: Semantic Markup for Web Services. 2004 http://www.w3.org/Submission/OWL-S/.

[3] OWL-S [UDDI Spec Technical Committee, UDDI Version 3.0.2. 2004: http://www.uddi.org

[4] Agarwal, S., A Goal Specification Language for Automated Discovery and Composition of Web Services, in Proceedings of the IEEE/WIC/ACM, International Conference on Web Intelligence (WI 07). 2007, IEEE Computer Society: Silicon Valley, USA. p. 528—53.

[5] Agarwal, S., A Goal Specification Language for Automated Discovery and Composition of Web Services, in Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI 07). 2007, IEEE Computer Society: Silicon Valley, USA. p. 528—53.

[6] Yue, P., et al. Semantics-enabled Metadata Generation, Tracking and Validation in Geospatial Web Service Composition for Mining Distributed Images in Proceedings of the 2007 IEEE International Geoscience and Remote Sensing Symposium (IGARSS07). 2007. Barcelona, Spain.

[7] Bianculli, D., C. Ghezzi, and P. Spoletini. A Model Checking Approach to Verify BPEL4WS Workflows in International Conference on Service-Oriented Computing and Applications (SOCA 07). 2007. Newport Beach, California, USA

[8] Pathak, J., S. Basu, and V. Honavar, On Context-Specific Substitutability of Web Services in 5th IEEE Intl. Conference on Web Services (ICWS 2007). 2007: Salt Lake City, UT, USA

[9] Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web services: concepts, architectures and applications. Springer-Verlag (2004).

[10] Pramodh N, Srinath V, Sri Krishna A, Optimization and Ranking in Web Service Composition using Performance Index, International Journal of Engineering and Technology (IJET), ISSN : 0975-4024 Vol 4 No 4 Aug-Sep 2012

[11] Senthilanand Chandrasekaran, Gregory S. Silver, John A. Miller, Jorge Cardoso, and Amit P. Sheth. "Web Service Technologies and Their Synergy with Simulation" Proceedings of the 2002 Winter Simulation Conference. Dec. 2002.