Graph Based Service Selection for Composition and Adaptation

Ajaya Kumar Tripathy, Department of Computer Science, Utkal University, Bhubaneswar, India
Manas Ranjan Patra, Department of Computer Science, Berhampur University, Berhampur, India
Sateesh Kumar Pradhan, Department of Computer Science, Utkal University, Bhubaneswar, India
Raisa Rafique, Department of CSE, Silicon Institute of Technology, Bhubaneswar, India
Pujalin Mohanty, Department of CSE, Silicon Institute of Technology, Bhubaneswar, India
Saumyaranjan Dash, Department of CSE, Silicon Institute of Technology, Bhubaneswar, India

One of the core idea of Service Oriented Architecture is the generation of new applications by composing existing services that are available on the web. Several approaches have been proposed to handle the problem of web service composition, but little effort has been devoted so far to the problem of consumer requirement oriented automatic service selection for composing and run-time adaptation of web service based systems (WSBS). In this paper we present a new framework which facilitates automatic service selection for composition of WSBS. This framework also facilitates automatic monitoring of WSBS at run-time and adaptation using an automatic re-composition strategy in case of any discrepancy found by the monitoring process at run-time.

Categories and Subject Descriptors: H.3.5 [Web-based services]: Web-based services
General Terms: Verification, Language, Design
Additional Key Words and Phrases: Web Services, Web Service Composition, Web Service Monitoring

1. INTRODUCTION
A web service is an application that exports a description of its functionality and makes it available using standard network technologies and provides basis for the development and execution of business process that are distributed over the network.

The basic idea of web services is the possibility to develop new applications by composing existing services that are available on the web. For that the manual selection of requirement oriented services for the composition, out of lot of existing alternate services is often difficult and error-prone task. The ability to support the selection of services for the composition of Web Service with semi-automated tools is an essential step to substantially increase quality of service and substantially decrease time and cost.

One of the fundamental idea of web services is distributed in nature and provided by third party. It may happen that at run-time some of the services are down or the quality of service is degraded, then the business analyst need to know about this at run-time, so that the analyst can take required action. For that, it is needed to monitor the composed services. Manual monitoring of web services at run-time is a difficult and error-prone task. The ability to support the WSBS automatic monitoring tool is an essential step to substantially avoid the degradation of the quality of the WSBS.

In case some services in WSBS are down or the quality of services are degraded, it is needed to adapt some new services at run-time to recover from that situation to provide better quality of service constraints. Again if some cheaper alternate services are available at run-time it is also needed to replace those service at run-time. The ability to support this adaptation with automated tool is an essential step to substantially avoid composed service failure, decrease cost and increase quality of service.

Some of the recent works have been addressed in different aspects of service selection for WSBS, e.g., [Marconi et al. 2006; Pistore et al. 2005; Hu et al. 2008; Zhu Yi-an 2009; Silva et al. 2007; Zheng and Yan 2008; Dong 2009]. And some recent works have been addressed in different aspects of WSBS adaptation, [Ly et al. 2008; Brogi and Popescu 2006; Karastoyanova and Leymann 2009; Marconi et al. 2009; Popescu et al. 2010; Kazhamiakin et al. 2010; Kazhamiakin et al. 2010].

Since all these issues are related to composed web service applications, it is needed to formulate an integrated tool to support all these issues.
This paper is an integration of the service composition and runtime monitoring scheme designed by the authors [Tripathy and Patra 2010; 2011]. An integrated framework is designed for service selection for composition as per the requirements, service monitoring at run-time and run-time adaptation. We represent the service domain as a directed graph in which services are vertices and the QOS matrices as weighted edges. We apply a shortest path algorithm as service selection algorithm to find out the shortest path between the starting and ending points of the new application. The services in that shortest path are used to compose to form the desired application. A run-time monitoring framework is used which is designed by the authors to monitor the new application at run-time. If some better new alternate services are available at run-time then it is added in the graph and the service selection algorithm is applied which will decide whether the service will replace some existing ones or not. Also in case of any discrepancy found in the monitoring process then, the involved service is removed from the graph and the service selection algorithm again run and provide the better options to recompose at run-time.

The rest of the document is organized as follows. Section 2 describes the state of the art in the web service technology and existing research approaches to the automated service composition, monitoring and adaptation. Section 3 describes the proposed integrated framework to support all the three issues. Finally Section 4 concludes the paper and introduces the future work directions.

2. STATE OF THE ART

Web Service Technologies

Web Services are platform-independent, self-contained, self-describing, modular components that can be published, located and invoked over the Web. In order to achieve interoperability in such heterogeneous frameworks, standards are of vital importance [Leymann 2003]. A whole stack of different standards has already been proposed with the aim of supporting the description, discovery, and interoperability of distributed, heterogeneous applications as services.

The functional description of a Web Service is provided by the Web Services Description Language (WSDL) [Christensen et al. 2001]. WSDL describes a set of operations it offers, in-coming and out-going messages, and data types used by the Web Service (defined in terms of XML Schemas). Concrete protocol bindings and physical address port specifications complete a service description, providing a mechanism to locate a Web Service. WSDL defines what a Web Service does, not how it does; it characterizes the service only in terms of its interface, without providing any behavioral description. Such dynamic aspects are crucial for a complete understanding of a web service so that it can be recognized and used by autonomous applications.

Web Service Composition Requirements

When addressing a WSBS composition problem, the (manual or automatic) development of the new composite WSBS must be driven by the analysis of published abstract process specifications of component services and by requirements and constraints the composite service has to satisfy.

Several approaches have been proposed to model WSBS compositions and Web Services in general. In the approach proposed in [Ponnekanti and Fox 2002] component services are defined in terms of their inputs and outputs; given the inputs and outputs of the service, a rule is then defined which indicates which outputs can be obtained by the service from given inputs. When a developer wishes to create and deploy a new composite service, he specifies the inputs and outputs of the composite service and submits it to SWORD which determines if the composite service can be realized using the existing services. This rule based approach can be adopted exclusively for modelling simple composition problems, where component services are atomic and deterministic. The methodology proposed in [Lau and Mylopoulos 2004] is founded on Tropos [Bresciani et al. 2001], an agent-oriented software development
technique that supports early and late requirements analysis, as well as architectural and detailed design. The key idea of this approach is that Web Services are designed by starting from stakeholders goals, that are then analyzed and refined into subgoals and tasks. However, the starting point for Web Service compositions are usually abstract workflows, which are derived from business scenarios, rather than high level business goals. In [Skogan et al. 2004] the authors propose a UML-based model-driven method for Web Service composition. They show how composite Web Service models can be transformed into executable models by means of UML model transformations. The weakness of UML-based approaches is that modeling complex composition scenarios is a demanding and difficult task since they require to use several different UML diagrams (e.g. activity, class, collaboration diagrams); moreover, the relationships between these diagrams do not have a clear and formal semantic.

Web Service Automated Composition

The manual analysis of abstract BPEL4WS processes, and the implementation of programs that interact with them, is a very difficult, time consuming, and error prone task; efficient automated support is needed to achieve cost effective, practically exploitable Web Service composition. With automated composition of Web Services we mean the generation of a new executable composite service that interacts with a set of existing component services in order to satisfy given composition requirements.

3. COMPOSITION, MONITORING & ADAPTATION FRAMEWORK

Fig. 1 depicts the framework. The major four components of this framework are Service Selection Engine, Composition & Deployment, Web-Service Monitoring Engine and Service Adaptation Engine.

Service Selection Engine:. Selecting services for a single composition plan is one of the most important task in designing WSBS. And the real challenge in this design is the selecting services based on consumers quality of service (QOS) preferences. In this section we propose a graph based approach for QOS requirement oriented service selection scheme. A collected set of related(related to desired WSBS) available services with their QOS matrices and client requirements are represented in a weight directed acyclic graph called "Service Graph". In "Service Graph" the service set represents the vertices. Two extra vertices represents the starting and ending point of composition. In this graph, all possible composition plans are represented by adding a directed edge from predecessor vertex to successor vertex with weight of the edge as the QOS matrices of the successor vertex. A weight estimation scheme is proposed to map all QOS matrices to similar scale. Finally a shortest path based approach
is designed to select proper services for composing and deploying the required composed application to achieve the customers need.

**Service Graph Design**

Once the business process flow requirement for the composition and available services are known, one technical expert has to design a service graph as per the following scheme.

The service graph $SG = (S, E, W)$, is a weighted directed acyclic graph. Where,

- The vertex set $S = \{S_1, S_2, ..., S_n\} \cup \{S_0, S_f\}$ represent the vertex set such that $S_i, i=1$ to $n$ represent all available atomic services for composition. Two extra vertex $S_0$, $S_f$ represents starting and ending points of composition.
- The edge set $E = \{(S_i, S_j) :$ there is a predecessor & successor relation between service $S_i$ & $S_j$ as per the composition requirement to achieve the business flow requirement for the composition.\}
- The weight set $W = \{w_{ij}, \forall (S_i, S_j) \in E\}$

$$w_{ij} = \begin{cases} EQM_j & \text{If } S_j \neq S_f \\ 0 & \text{If } S_j = S_f \end{cases}$$

Where, $EQM_j$ is the Estimated QOS Matrix of $S_j$. $EQM_j$ is a function of QOS matrices $Q_i$ and user requirement, which we will discuss in detail further. Figure 2 depicts a Service Graph for an example scenario.

![Service Graph](image)

**3.0.1. EQM Estimation.** $EQM$ is a function of quality of service matrices of the available services and the user requirements. The specification of QOS matrices and user requirements are as follows.

- $Q_i = \langle q_{i1}, q_{i2}, ..., q_{ik} \rangle$ is the QOS matrix vector of service $S_i$, where $k$ is the number of QOS matrices collected.
- $U_i = \langle u_{i1}, u_{i2}, ..., u_{ik} \rangle$ is the upper limit of the matrices.
— $L_i = \langle l_{i1}, l_{i2}, ..., l_{ik} \rangle$ is the lower limit of the matrices.
— $P_i = \langle p_{i1}, p_{i2}, ..., p_{ik} \rangle$ is the projection of $Q_i$ in to the range of 0 to 1.

$$p_{ij} = \frac{q_{ij} - l_{ij}}{u_{ij} - l_{ij}}$$

— $T = \langle t_1, t_2, ..., t_k \rangle$ is the QOS matrix type.

$$t_i = \begin{cases} -1 & \text{If } i^{th} \text{ QOS matrix } \propto \text{ preference of service} \\ 1 & \text{If } i^{th} \text{ QOS matrix } \propto \text{ dislike of service} \end{cases}$$

— $I_i = \langle i_{i1}, i_{i2}, ..., i_{ik} \rangle$ is the importance % of the matrices for vertex $S_i$.

$$EQM_i = \sum_{j=1}^{k} \frac{i_{ij} * p_{ij} * t_{ij}}{k},$$

### 3.0.2. Service Selection Algorithm

The "Service Selection Algorithm" selects services for single execution plan for the required WSBS in such a way that the overall cost of the composed services is relatively low and provide better QOS as per the requirement. We use Bellman Ford’s single source shortest path graph algorithm [Cormen et al.] to design Service Selection Algorithm. The Service Selection Algorithm is defined in Algorithm 1.

#### ALGORITHM 1: Service Selection

1: for each vertex $S$ in service graph SG do
2: \hspace{1em} $OCW[S] \leftarrow \infty$
3: \hspace{1em} $Predecessor[S] \leftarrow NIL$
4: end for
5: \hspace{1em} $OCW[S_0] \leftarrow 0$
6: for number of vertices - 1 times do
7: \hspace{1em} for each edge $(S_i, S_j)$ do
8: \hspace{2em} if $OCW[S_j] > OCW[S_i] + W_{ij}$ then
9: \hspace{3em} $OCW[S_j] \leftarrow OCW[S_i] + W_{ij}$
10: \hspace{3em} $Predecessor[S_j] \leftarrow S_i$
11: \hspace{2em} end if
12: end for
13: end for
14: \hspace{1em} $RevOrderSerList[] \leftarrow \{S_f\}$
15: currentnode $\leftarrow S_f$
16: while rootNode $\neq S_0$ do
17: \hspace{1em} $RevOrderSerList[] \leftarrow RevOrderSerList[] \cup Predecessor[currentNode]$
18: \hspace{1em} currentNode $\leftarrow Predecessor[currentNode]$
19: end while

Composition & Deployment: Manual composition of web services is a time consuming and error prone task. Therefore automatic service composition becomes the basic need for deploying WSBS. In this framework, after the services are selected for composition, the desired WSBS is composed and deployed using a simulated automatic composition framework proposed in [Marconi et al. 2006; Pistore et al. 2005].

Web-Service Monitoring Engine: In most cases services used in the WSBS are provided by third party, distributed in nature and are totally in control of the service provider. Therefore consumers need to ensure the QOS of WSBS over the life period. A run-time WSBS monitoring framework [Tripathy and Patra 2010] designed by the authors is responsible for measuring QOS matrices of the services used in the WSBS. The monitoring framework is depicted in Figure 3.
Fig. 3. SBS monitoring Framework

Service Adaptation Engine: In case of WSBS failure or performance degradation of a service used in WSBS detected by the Web-Service Monitoring Engine, the Service Adaptation Engine adapt suitable new services in the WSBS to maintain the QOS of WSBS.

In case of the used services perform as usual, but some alternate services providing better quality of service. In that case we need a run-time performance evaluation and proper replacement of services. We design a algorithm for run-time performance improvement of designed WSBS, which is shown in Algorithm 2. This algorithm runs in a regular interval of time to provide best possible quality of WSBS.

**ALGORITHM 2: Performance Updation**

1. if New related service arrive then
2. Update Service Graph by adding a new service vertex as described in "Graph Construction" section.
3. end if
4. Service Selection();

It is also assumed that at run-time, the monitoring framework notify the service failure or significant quality degradation of used services in the WSBS. In case of significant degradation of performance, it is necessary to replace an alternate service or a set of services. In that case a run-time monitoring of WSBS and a adaptation scheme is needed to adapt these WSBS failure situation. The authors designed a run-time WSBS monitoring framework [Tripathy and Patra 2010] to do the required monitoring and a adaptation scheme is designed for the WSBS adaptation, which is shown in Algorithm 3. After getting the fault notice from the WSBS monitoring framework [Tripathy and Patra 2010], this algorithm make a run-time alternate service selection to provide best possible quality of WSBS.

**ALGORITHM 3: SBS Adaptation**

1. if New fault notice from monitoring framework then
2. Update Service Graph by deleting the fault responsible services.
3. end if
4. Service Selection();

4. CONCLUSION AND FUTURE WORK DIRECTIONS

An integrated WSBS composition, monitoring and adaptation framework is proposed in this paper. QOS requirement oriented composition is made by semi supervised service selection approach. An event based non-intrusive monitoring framework is used to monitor the WSBS at run-time. In case of quality degradation at run time an automation adaptation scheme is proposed. In future we would like to test the proposed framework, by composing, monitoring and fault handling of real WSBS at run-time.
ACKNOWLEDGMENTS

We thank the team members of the Service Oriented Application unit of FBK, Trento, Italy for their valuable suggestions and validation of concepts incorporated in this work.

REFERENCES


Cormen, T. et al. Introduction to algorithms, chapter 24.1 (pp. 558-592).


