Service Composition in the Context of Service Oriented Architecture

Ayaz Mahmood, Muhammad Ibrahim and M.N.A. Khan

Shaheed Zulfikar Ali Bhutto Institute of Science and Technology (SZABIST), Islamabad, Pakistan

Abstract: Service Oriented Architecture (SOA) has been one of the most focused area of research since the last decade for providing various solutions using the concept of services. Various researchers have proposed specific models to customize the service discovery, registry and composition for its effective working. This paper looks into the current level of research published in the contemporary literature relating to the efficient service composition for best utilization of the resources. For this purpose, various models have been studied and evaluated with particular reference to service discovery and efficient composition.

Key words: Service Oriented Architecture (SOA) • Service Composition • Web Service Composition (WSC) • Dependency-aware Service Oriented Architecture (DSOA) • Activity Network (ANet)

INTRODUCTION

Recent technological advancement and development of new standards have lead to the creation of new methods for designing and development of web applications. These web applications are linked through independently published web service components. These web applications are also called web services. A system that has the tendency of integrating multiple web services automatically in a transparent way is considered as part of the web service oriented system. Service Oriented Architecture (SOA) is an architectural method that is used for the creation and usage of business services in the form of web services. SOA also explains the IT infrastructure that makes it possible to allow different applications to exchange data and contribute in business processes. The following two entities are involved in SOA that collaborate to support the “discover, connect and call up” standard [1].

- **Service consumer** is an application web service that needs another service for completing its function. When the user request for a services, then it makes a probe of the service in the registry, connects it to the requesting service and performs the service function based upon the requirements [1].
- **Service provider** is the entity that accepts the demands of service from consumer and executes it. It supplies its services and user interface to service registry [1].
- **Service registry** is used for finding out the services. It contains a list of all the accessible services.

This paper presents a literature review on service composition techniques. In addition, the paper also focuses on the limitations of the existing service composition methods, evaluation techniques, models and their practices. However, the significant of this study is to describe importance of service composition in SOA.

The paper is organized as follow: This section provides introduction to SOA and discusses its importance. Section II provides an overview of existing service composition techniques, methods or frameworks. Critical analysis is provided in Section III which is primarily based on evaluation of the literature review discussed in Section II. Finally, we provide a conclusion summary followed by possible future research dimensions in the last section.

**Literature Review:** Zhou et al. [1] propose an extended SOA model for service composition and service dependency. The authors established a dependency-
aware service-oriented architecture (DSOA) to specify
dependency aware service interactions, i.e., service
publication, discovery, composition and binding.
The authors claim that traditional Service Oriented
Computing (SOC) focuses on service composition
for application development which results in “compose
one time and use one time” tactic. For discovering the
objective of service composition with context of DSOA,
a service composition example for a security notification
service is demonstrated. The registration of security
notification service to service registry after composition
enables user to call the service as many times as he/she
wants, i.e., compose once, use many times. Based
upon the security notification example, the authors
developed upper service dependency ontology. The
paper validates concept of DSOA service composition by
implementing a demo. The limitation of such approach is
that upper service dependency ontology is not capable
enough to support complex DSOA service dependency
description.

Agni et al. [2] highlight salient features of SOA as
well as the challenges faced such as service discovery,
service composition, service interaction, robustness,
quality of service and security. The authors propose
the idea of self-organizing SOA to ensure the
robustness of SOA. The term self-organizing means that in reaction to outer incidents, the service
still remains able to alter its internal structure and
functions. The idea has been taken from the biological
processes like self-healing and applied it to the SOA
problems.

Zhu et al. [3] introduced a new service composition
mechanism based on peer-to-peer (P2P) network.
An extended state machine model is shown to identify
network model from a service. The model describes a
service and its execution patterns. Three execution
patterns (AND, OR and Sequential patterns) are basic
service composition constructs of the model. The model
serves as a basis for service composition algorithm.
Service composition algorithm describes the process of
service composition in detail. A graphic model based on
one of the execution patterns turns out to be the input
of algorithm. Based on the graph model, the algorithm
generates composite service. Furthermore, paper presents
a java based prototype for P2P based service
composition. The authors claim that the presented
algorithm is powerful enough for efficient creation of
composite services. However, analysis of the algorithm
is based upon some assumptions. The main limitation of
this paper is that proposed technique of service
composition method is not feasible for large and complex
systems such as multi-tier applications.

Liu et al. [4] present a technique of web service
composition and optimization to meet Quality-of-Service
(QoS) needs of users. An algorithm is introduced which
is based on the preliminary business process
configuration for QoS oriented web service composition
and optimization. The algorithm mainly consists of three
parts. First part is business process analysis, in which
processes gets transformed into a composition tree where
the structure activities serves as the branch nodes and
basic activities as the leaf nodes. Second part is business
processes and services evaluation. Utility function is
introduced to measure different QoS attributes with
uniform standard formal method to evaluate the web
services. Single-utility and multi-utility functions for QoS
attributes such as response time, throughput and
success rate are designed at this stage. Third part is
service composition and optimization, which gets
done by performing lengthways optimization and
breadthways optimization. The optimization of
response time in parallel or branch structure is known as
lengthways optimization; whereas, the optimization of
throughput in order structure is called breadthways
optimization. To validate the effect of algorithm, a
group of two experiments are carried out to produce
consequential optimized services. However, the technique
should introduce more QoS attributes for better results.
In addition, this algorithm must be combined with
other existing algorithms to attempt a better service
composition.

Kim et al. [5] provide a technique for dynamic web
service composition that extends the meta model of web
service-business process execution language (WS-BPEL)
to apply the concept of AOP. It focuses on service
reliability and dynamic service replacement for the
composition of a new service. Aspect processes are
developed for service replacement and service faults
detection. A special dynamic aspect weaver is
composed for weaving WS-BPEL process and
aspect process. Three main components of dynamic
weaver are process monitor, aspect process manager
and process controller. Process monitor observes
execution of WS-BPEL process and notifies the aspect
manager to activate the corresponding aspect processes.
Aspect manager is responsible for aspect registration,
interpretation and activation. Process controller controls the aspect process which in turn controls the WS-BPEL process at WS-BPEL engine. The technique is proven to be effective for failure prevention of web services composition from unexpected faults. However, the major limitation of the proposed technique is that it did not address other QoS attributes other than reliability.

Li et al. [6] propose a qualitative approach to effort judgment for Web Service Composition (WSC) based SOA implementations. The authors used divide and conquer as their basic strategy to confine effort judgment for WSC-based SOA. Moreover, the authors introduced a classification matrix of WSC which differentiates between context and process dimensions. The paper also includes set of qualitative effort judgment hypotheses in the context of software engineering domain, especially in distributed environment. Authors treated process models and context types as effort factors of WSC in classification matrix. In addition, judgment hypotheses were applied over comparable factors. Finally, the comparison of effort among different WSC approaches was explained from a qualitative perspective. A case study was explained via using effort judgment of WSC approaches. The paper provides a firm theoretical base for a qualitative approach to judge efforts required for different proposals of WSC based SOA. The paper lacks in proving any empirical findings to support their qualitative effort judgment approach for WSCs.

Dasgupta et al. [7] propose an abstraction-based service discovery, selection and composition architecture for event based SOA systems. The authors highlighted problems associated with previous task based architectures and came up with a modified event driven based architectures. Therefore, the new event driven model named “Activity Network (ANet)” solved the problems such as task framing, task analysis, task mapping and analysis and selection incompatibility. One problem associated with ANet is that it gets more complicated as it grows over the time, which was addressed by incorporating “ANet Abstraction” technique. Moreover, a semi empirical evaluation of the Anet Abstraction algorithm was done via setting up a simulation platform to check its time performance and scalability. The paper provides a firm theoretical base for the ANet algorithm, which later on was proved by using a simulation platform. But the paper lacks in comparing ANet with tradition task based modeling. Only comparisons were made with their associated problems and no empirical comparison was done.

Blum et al. [8] propose management of SOA based Next Generation Network (NGN) service exposure, discovery and composition. The authors discuss a policy based mechanism for service exposure, discovery and composition to offer chargeable services and service building-blocks to 3rd party in a customized way. Furthermore, an automatic fault management solution for NGN service compositions offers self-healing mechanisms for SOA-based service compositions. SOA-based NGN also provide rapid service discovery, creation, composition and deployment. Furthermore, semantics with more improved policy mechanisms provides user oriented NGN services, with individually customized service compositions. The papers showed firm basis for the requirements of an appropriate management of NGN’s with respect to IMS processes. Moreover, it also provides self healing mechanisms for its processes. The paper, however, does not discuss dynamic composition.

Liu et al. [9] present a technique of web service composition based upon Mashup architecture. Mashup is referred to a process which incorporates data/content from heterogeneous sources to create a new service. Proposed technique extends current SOA model with Mashup to facilitate service composition and consists of three major roles. First one is Mashup Component Builder (MCB) that acts as service provider that selects services from Service Catalogue and encapsulates all services in a standard component model with UI presentation. Second one is Mashup Server which publishes services from providers, stores Mashup components created via MCB and monitors the performance. Third is Mashup consumer that selects components from server to compose their own services in the browser. Finally, a case study is presented for the validation of the said technique. The paper provides a good overview of service composition technique for creating user centric web applications. However, the limitation of the paper is that proposed technique cannot be applied in the complex real world scenarios. Also it does not provide runtime management and maintenance of Mashup.

Tan et al. [10] present a lightweight approach to bridge gap between business and service domains. The proposed technique is based on a data-driven
method for creating services to implement given requirements. It aims at finding the relations between business domain data and service domain data. Further, it creates three composition rules-sequential, parallel and choice-based on the augmented data model. A Service Net is generated that encloses all operations in a given service portfolio. Then a Petri-net decomposition method is used to develop a subnet of Service Net, which fulfills the business requirement. The proposed technique makes a combination of bottom-up and top-down approaches in service composition domain. The services obtained via this approach are reusable in nature. Furthermore, introduction of algorithms for data-service composition has brought formalism in the proposed approach. A real-life scenario is used to demonstrate feasibility of the proposed technique. However, it does not address the execution of created Petri nets in a workflow engine. Also, this technique is only limited to data-driven aspect of service composition. It does not provide any mechanism for data sharing between heterogeneous services.

SOA allows distribution of data and enables applications on multiple platforms to access, use and manage data in a flexible way, which is ordinarily not possible in traditional data access methods. This data access requires matching of requests to available resources which can be done by different methods. One such approach is F-Match suggested by Stephen et al. [11]. This approach extends and uses SAW-OWL-S for handling all requests inquiring about the required service and also to respond to these requests by advertising available service resources. This is done by analyzing the request parameters for the service discovery. If the requested parameters indicate functionality which is not available as part of the service then they are filtered out. And if the requested functionality exists, then that functionality is ranked based on the requested parameters. The main weakness of this research work is that it does not have specified any QoS criterion for service discovery and lacks pertinent method to keep intact the NFR of the service for service discovery.

In [12] Ni explained ontology enabled service oriented architecture (OSOA) for assisting ordinary users (non-experts) to utilize devices and combine their functionality. OSOA mingles interoperability and semantic description provided by web services and ontologies respectively. OSOA based ad hoc service composition provides promising solution. Despite interoperability among different services being the main issue, it is possible to exchange data reliably and securely among applications operating in different environments in order to provide a good base for solutions of pervasive computing. Control devices along with unanticipated compositions of any existing services to uncover those which might be of use for non-expert users is provided by ad hoc service composition. The strength of the paper is that ad hoc service composition produces new object with the consumption of old one. If a service has no output in terms of a new object, it is basically the final service that finalizes the composition. The main weakness of this research work is that it does not have specified NFR of the service for service composition.

Sobecki, et al. [13] described an ontology based service discovery using WSMO language in the SOA system. Services are specified through semantic description which facilitates service discovery. WSMO is used for solving service description and discovery issues. One of the strengths of WSMO language is its ability to describe non-functional requirements of a service along with functional ones. While dealing with non-functional requirements in WSMO, capability is the most important functionality. The weakness in the mentioned work is the unavailability of service interface development provided by mediators.

Gao et al. [14] focused on extended service oriented architecture (ESOA) which provides distinct tiers for composing and coordinating services. Moreover, it also helps in services management with the help of grid services while working in an open marketplace. The SGB significantly accelerates application development and deployment with the help of high level services for service management, interaction, aggregation and security. In order to create a single composite service from the aggregation of multiple services, all the required functionality and roles are offered by the service composition tier of ESOA. The strength of this paper is that in contrast to the basic SOA, ESOA addresses overarching concerns like service transaction management, coordination and security etc. Moreover the main advantage of a distinct integration tier is the facility of coupling value added services thus providing packaged solutions for common development needs.
Table 1: Summary of Service Composition Techniques/Models/Practices.

<table>
<thead>
<tr>
<th>Author</th>
<th>Technique</th>
<th>Key Points</th>
<th>Limitations</th>
<th>Suggested Improvements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zhou et al. [1]</td>
<td>Dependency-aware service-oriented architecture (DSOA) approach</td>
<td>The Service Oriented Computing (SOC) focuses on service composition for application development which results in &quot;compose one time and use one time&quot;.</td>
<td>The proposed research does not explain the technical implementation details of the service composition. Upper service dependency ontology is not capable enough to support complex DSOA service dependency description.</td>
<td>The possible solution of this problem would be to introduce a layer that can provide support for managing DSOA service dependency description.</td>
</tr>
<tr>
<td>Agni et al. [2]</td>
<td>Self organizing SOA</td>
<td>Authors propose the idea of self organizing SOA to ensure the robustness of SOA.</td>
<td>The major issue with this is that it lacks the implementation of the proposed architecture to any of the real world case study.</td>
<td>A validation of the model would provide real essence of self organization of the services.</td>
</tr>
<tr>
<td>Zhu et al. [3]</td>
<td>P2P network based approach</td>
<td>A technique of extended state machine model is shown to identify network model from a service.</td>
<td>The proposed technique of service composition method is not feasible for large and complex systems such as multi-tier applications.</td>
<td>A prototype based implementation on the multi-tier application would conquer the limitation.</td>
</tr>
<tr>
<td>Liu et al. [4]</td>
<td>Quality-of-Service (QoS) based approach</td>
<td>The proposed algorithm is based on preliminary business process configuration for the QoS oriented web service composition and optimization.</td>
<td>This technique only deals with the Security attribute of QoS. It does not account for other QoS attributes such as availability, reliability etc.</td>
<td>This algorithm must be combined with other existing algorithms to attempt a better service composition.</td>
</tr>
<tr>
<td>Kim et al. [5]</td>
<td>Dynamic web service composition</td>
<td>The proposed technique extends the meta model of web service-business process execution language (WS-BPEL) to apply the concept of AOP.</td>
<td>The major limitation of the proposed technique is that it does not address other QoS attributes other than reliability.</td>
<td>The proposed technique should take into account the other QoS attributes as well.</td>
</tr>
<tr>
<td>Li et al. [6]</td>
<td>Qualitative approach using Divide and Conquer strategy</td>
<td>The authors used Divide and Conquer strategy to confine effort judgment for Web Service Composition based SOA.</td>
<td>The paper lacks in proving any empirical findings to support their qualitative effort judgment approach for WSCs.</td>
<td>A validation or simulation would enhance the proof of concept.</td>
</tr>
<tr>
<td>Dasgupta et al. [7]</td>
<td>Activity Network (ANet) Model</td>
<td>The authors highlighted problems associated with previous task based architectures and propose a modified event driven based architectures.</td>
<td>The paper lacks in comparing ANet with traditional task based modeling.</td>
<td>An empirical comparison with the traditional task based modeling would increase the research strength.</td>
</tr>
</tbody>
</table>

**Critical Evaluation:** The critical analysis of the literature review in provided in Table 1.

**CONCLUSION**

In this study, we have made an attempt to provide an understanding of the concept of service composition related techniques, practices and frameworks in the context of service oriented architecture. We have highlighted some of the papers relating to dependency aware. Mashups have been discussed that are used for service composition. The central idea behind this work was to review the various composition techniques used for SOA. This study is particularly useful for software developers in the dispensation of their job to clearly specify model and implement web services in SOA-based application. On the basis of the literature review provided in this paper, we conclude that most of the techniques discussed are effective for applications based upon SOA. However, a limitation observed during the conduct of this study shows that various techniques fall short of addressing cross-cutting concerns in design and
implementation of web services. Also, it lacks to provide methodology for implementing or utilizing the SOA in the cloud computing paradigm. To overcome this limitation, our proposed future work will focus on introduction of Aspect Orientation in Service Composition and the utilization of SOA in the context of cloud computing. The main intended advantage would be to bridge the gap between SOA and cloud computing.

REFERENCE