A Novel Approach for Verifying Web Services Composition Using Muller Transition Algorithm

N. Danapaquiame¹ Dr. E. Ilavarasan² and J. Sankari³

¹Research Scholar, Department of Computer Science and Engineering, Pondicherry Engineering College, Pondicherry-605014, India, n.danapaquiame@gmail.com
²Associate Professor, Department of Computer Science and Engineering, Pondicherry Engineering College, Pondicherry-605014, India, eilavarasan@pec.edu
³Student, Pondicherry Engineering College, Pondicherry-605114, India, sankarij68@pec.edu

Abstract. Web Services are a very appropriate communication mechanism to perform distributed business processes among several organizations. A concept of service composition, combining existing Web services together to from new added services. The main problem is the verification of the Composed Web Services. It has to depend on formal verification method which helps to check the correctness of Services Composition. Only limited number of verification approaches available in Web Services Composition. This paper proposes a new automata called Muller Automata is introduced for verifying the composed web services and also a new algorithm called Muller Transition algorithm (MTA) is introduced for avoiding Dead Transition in Automata Representation. The Web Services are composed using BPEL4WS and it is transformed into Promela (It is the input language of SPIN model checker) and then verified using the tool called SPIN.

Keywords: Composition, Muller Automata, Promela, MTA and BPEL4WS and SPIN.

1 Introduction

Web services are distributed and independent computational elements that solve specific tasks, varying from simple requests to complex business processes, and they will be interacting the information using XML messages following the SOAP standard. Composition of services has received much interest to support B2B. Developing a number of XML-based standards to formalize the specification of web services, their composition, and their execution. WSC should satisfy several fundamental requirements: Connectivity, support for non-functional quality of service metrics, correctness, scalability, and in the desiderative situation, automatization.

In orchestration, the involved web services are under control of a single endpoint central process (another web service). This process coordinates the execution of different operations on the Web services participating in the process. Choreography is based on collaboration and is mainly used to exchange messages in public business
processes. In comparison with Choreography, Orchestration is obviously more efficient and flexible when it comes to compose Web services to execute business processes.

A Net Beans module is a set of Java classes written to communicate with the Net Beans APIs, for extending the IDE or for creating your own application on the Platform. The following web service programming models are supported by the IDE:

### 1.1 Muller Automata

Muller automaton is a type of a $\omega$-automaton ($\omega$-automata are finite automata on infinite words). The Muller automata is defined using Muller acceptance condition, i.e. the set of all states visited infinitely often must be an element of the acceptance set. Muller-automaton is a tuple $M = (Qm, \Sigma m, \delta m, q0m, Fm)$ that consists of the following information:

- $Qm$ is a finite set. The elements of $Qm$ are called the *states* of $Qm$.
- $\Sigma m$ is a finite set called the *alphabet* of $M$.
- $\delta m : Qm \times \Sigma m \rightarrow Qm$ is a function, called the *transition function* of $A$.
- $q0m$ is an element of $Qm$, called the initial state.
- $Fm$ is the *accept component*, $Fm \subseteq pow(Qm)$ if it is used with the following acceptance condition:

**Muller Acceptance:** We say, $M$ accept a $\omega$-word $\alpha \in \Sigma^{\omega}$ if and only if there exist a run $r$ of $M$ on $\alpha$ satisfying $\text{Inf}(r) \in Fm$ i.e. the set of infinitely often visited states are exactly one of the set in $Fm$.

The main contribution of this paper will be verification of the composed web services using an algorithm can be summarized as follows:

- We present a novel approach to verification of composed web services Called Muller Automata. It is suitable for both deterministic and non-deterministic system.
- We develop the new algorithm called Muller Transition Algorithm (MTA). It avoids the Dead transition problems.
- We also compare with our most recent system with the Timed Automata [2], Interface Automata [1], Which is the verification technique for deterministic system.
- We have converted Muller Automata Representation to Promela.
- Finally, we used the tool called SPIN for verifying the composed web services (input is the Promela)
- We report our system is giving efficient result than other system by means of accepting regular languages.
- We show that our system is readily applicable for existing applications, and that it can be used for verifying composed web services.

The rest of this paper is organized as follows. In Section 2, describes a Verification algorithm. In Section 3, we implemented with real time applications i.e. case study. In Section 4 discuss related work. In Section 5 depicts the summarize our main results and give an outlook on future research.
2 Verification Algorithm

2.1 Muller Transition Algorithm for dead transitions

There are no activities in the process that cannot be realized. If initially dead transition exists, then the composition was bad designed. The figure 1 shows the Muller Transition Algorithm (MTA). It removes the dead transition problem.

![Fig 1. Muller Transition Algorithm](image)

2.2 Service Composition Verification

During verification process the following properties are verified. They are

- Web service is composed by using BPEL. For verification BPEL is converted into Muller Automata. Using this automata a new algorithm called DM (Dead Transition algorithm) can be verified. In verification process the following property is verified. It is defined as

  **Dead transitions** means that the transitions which will never be enabled. There are no activities in the process that cannot be realized. If initially dead transition exists, then the composition was bad designed.

3 Implementation

In library service, it provides the detail about book, issuing and returning of the book. This service includes the following operation like insert student details, get student details, book details and return details etc. Hotel reservation lists out the availability of hotel in a given district. Also, it provides the services for checking the room availability and reserving the room. Bank service provides detail about the customer who is having account, branch details, checks the borrower and depositor, loan details etc. The three applications are designed and the corresponding testing can be made i.e unit testing.
3.1 Composition of Web Services

Figure 2 shows the full composed BPEL diagram of the web service composition. The above diagram shows the composition of web services using BPEL4WS. This composed web services can be converted into Muller automata. Using the Muller transition algorithm dead transition can be verified. If dead transition exists, the composition was bad designed that should be recomposed. Figure 3 shows the automata diagram for the composed web services fig 2.

3.2 Verifying Safety and Reachability Properties

Table 1: Comparing existing system with proposed system

<table>
<thead>
<tr>
<th>Name</th>
<th>Transaction</th>
<th>Uniform Initial</th>
<th>Uniform Initial</th>
<th>Distributed</th>
<th>Tool Used</th>
<th>Proof Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aaa</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bbb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ccc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ddd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig 4. Verifying Safety and Reachability Properties
The figure 4 shows verifying the safety and reachability property is checked by selecting the safety option and by clicking the verify button the property is satisfied.

The above tabular shows the overall survey of my research works. Here only limited number of approaches is available for verifying the composed Web Services. Even though the methods are available, it is not satisfying the properties like Emptiness, Reachability Dead Transition. Table 1 shows the survey report regarding the existing system with proposed system.

4 Related Works

We now discuss related work on Web Service Composition Verification. Recent survey [1] which can be roughly divided into (i) approaches that are based on Timed Automata [2, 9]. These papers based on Timed Automata and Computation Orchestration. WSCDL used for composition of Web Services, converted into Timed Automata and verified using the tool called UPPAAL for verification of the web service composition. Workflow net models are used and define a fully-automatic translation of this formalism into conceptual BPEL by means of the tools 4BPEL framework. The automatic generation of BPEL code by using the workflow net formalism as design model, which can be checked of being free of deadlocks and live locks. This proposal then takes the orchestration viewpoint, whereas, we take choreographic one. Timed Automata applicable only for Deterministic system rather than non-Deterministic System. We discuss the (ii) Approaches that are based on Interface Automata [1]. It is based on the verification of the composed web services using interface automata.

Initially composition done BPEL4WS, converted into Interface automata mapped into Promela and then verified using the tool called SPIN. This approach is applicable for Deterministic system rather than Non-Deterministic system. We now discuss (iii) Approaches that are based on Petri net [3, 5, 6, 7and 8] used Petri-nets presented a methodology for the design, verification and validation of composite Web Services using WS-CDL as the language for describing Web Service interactions, and Petrinets as a formalism (that allows us to analyze the described systems). In this work they have considered timed automata and prioritized collaboration in composite Web Services, so the considered model of Petri nets is a prioritized version of Timed Petri nets. The last approaches (iv) that are based on Model checking [4]. In this paper they have discussed about the verification using SPIN model checker. But it is applicable for deterministic system rather than non-deterministic system.

5 Conclusion and Future Work

Web services technologies are becoming as popular standard to integrate distributed applications and systems using XML-based standards. Developing applications that support web services interfaces will not be enough to provide complete and coordinated business processes. Thus, we need a new approach to compose these web services together in order to form web services orchestration and processes definition.
In this paper, an algorithm is proposed (Muller Transition Algorithm) for the verification of web services composition using Muller automata. Web services are created and composed using BPEL4WS, and then it is mapped into Muller automata notations and transformed into Promela. SPIN tool is used for verifying the composed web services. The experiment shows that our approach can reduce Reach ability, Dead Transitions and verify the safety correctness property effectively in verification process. Even though the techniques are available for verifying the composed web services, but is applicable only for deterministic system rather than non-deterministic system. Here selecting the accepting state or final state time duration will be very less. Because $\text{INF}(r) \in F$, ie it can have more than one final state, selecting by nearest final state than Buchi automata. Further work will be testing by using different parameters for composed web services.

References

8. YanPing Yang, Qing Ping Tan, Yong Xiao, “Verifying Web Services Composition Based on Hierarchical Colored Petri Nets”, November 4, 2005, Bremen, Germany.ACM 1-59593-184-5/05/0011...$5.00.