Modeling and Analysis of Web Services Composition Based on Higher-Order Petri Net

Ali M. Meligy¹, Hani M. Ibrahim², Amal M. Aqlan³
¹, ², ³ Dept. of Mathematics, Faculty of Science, Menoufia University, Egypt

Abstract— Web service composition is a most mature and effective way to realize the rapidly changing requirements of business in service-oriented solutions. Testing the compositions of web services is complex, due to their distributed nature and asynchronous behaviour. Petri Nets (PNs) provide a framework for the construction and analysis of distributed and concurrent systems. This work focuses on the modeling and analysis of web services composition using Higher-Order Petri Nets (HOPNs). The modeling emphasizes on aspects of composition, i.e. a modeling capable of verifying and analyzing the performance of composition at present. The analysis of composition can adopt other approaches, for example Reachability graph.

Keyword— Web services composition (WSC), Reachability graph (RG), Petri Nets (PN), Higher-Order Petri Net (HOPN), Open Workflow net (oWF-net).

I. INTRODUCTION

With the continuous development of computer technology, constructing the loosely coupled collaborative software system, which can flexibly integrates program or legacy system distributing in heterogeneous environment, has become an important research field of software engineering. Service oriented computing (SOC) appear in order to solve above problems. And web service has become one of the most important computing resources on the Internet. Meanwhile, along with its combination standards and technology continuing to develop, web service has been the core technology of SOC [1].

According to application requirements, web service composition may consist of many autonomous web services, thus it may provide new, more powerful and value-added functions.

In order to realize this composition [2], several approaches have been proposed.

- Web service composition based on workflow.
  The researchers adopted several languages to implement this approach, namely eFlow.
- Web service composition relied on XML.
  The most popular languages that are based on this approach are XLANG and BPEL4WS. Unfortunately, these methods suffer from a total lack of semantic representations of services, which has led researchers to propose another approach based on ontology such as OWL-S. It enables the exchange of information in a meaningful way. However, these approaches cannot verify certain properties such as QoS, accuracy, etc.
- Web service composition based on formal method.
  This approach utilizes mathematics model and formal tools to describe web service composition. Petri net is a formal model of concurrency. Process algebra is formal language, which can describe and reason process behaviour.

The authors in [2] presents a modeling the WSC based on Petri nets, they use open workflow nets to model Web services which communicate with other services. Open workflow nets (oWF-nets) result from the extension of workflow nets (WF-nets) by adding external places for the communication of the external environment of the considered Web services.

The classical Petri nets cannot model the behavior of the complex system accurately. To solve this problem, researchers proposed many extensions to the classical Petri nets. Examples are: Higher-Order Petri nets (HOPNs) [3], which use the similarities between neural networks and Petri nets to exploit the properties of higher-order neural networks.

In this paper, we propose a new Higher-Order Petri net called the Higher-Order Open workflow nets (HOWFNs) by adding some amendments to the oWF-nets Model, depending on the definition of Higher-Order Petri net and a Conversion Procedure.

The rest of this paper is organized as follows. Section II is dedicated to present background on Web services composition (WSC), Petri nets, as well as Open workflow net. The definitions proposed of Higher-Order Open Workflow Nets presents in section III. Section IV and V discusses Web Services Composition model with Petri Net and analysis using Reachability Graph, respectively. In section VI presents related work. Conclusions are drawn in Section VII.
II. BACKGROUND

A. Web Services Composition
Web services composition (WSC) consists on the construction of new Web services by composing existing ones [2]. A composite service is a new service with added value obtained by combining other existing services. It can be the allocation of basic services or composite services. Fig.1 shows the obtaining composite service.

![Fig.1 Schematic description of WSC](image)

B. Petri Nets
Petri nets were originally developed to meet the need in specifying process synchronization, asynchronous events, concurrent operations, and conflicts or resource sharing for a variety of industrial automated systems at the discrete-event level. For that reasons, we propose to model WSC using Petri nets.

Definition 1 (Petri Nets): A Petri net is a 4-tuple $PN = (P, T, F, W)$ where
- $P$ and $T$ are two finite non-empty sets of places and transitions respectively, $P \cap T = \emptyset$,
- $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation, and
- $W: F \rightarrow N$ is the weight function of $N$ satisfying $W(x, y) = 0 \iff (x, y) \notin F$.

If $W(u) = 1 \ \forall u \in F$ then $PN$ is said to be ordinary net and it is denoted by $N = (P, T, F)$. For all $x \in P \cup T$, the preset of $x$ is $\text{pre} = \{y \mid (y, x) \in F\}$ and the postset of $x$ is $\text{post} = \{y \mid (x, y) \in F\}$. A marking of a Petri net $N$ is a function $M: P \rightarrow N$. The initial marking of $N$ is denoted by $M_0$. A transition $t \in T$ is enabled in a marking $M$ (denoted by $M[t]$) if and only if $\forall p \in \text{pre}(t)\ M(p) \geq W(p, t)$. If transition $t$ is enabled in marking $M$, it can be fired, leading to a new marking $M'$ such that: $\forall p \in P: M'(p) = M(p) - W(p, t) + W(t, p)$. The firing is denoted by $M[t]M_0$. The set of all markings reachable from a marking $M$ is denoted by $[M]$. For a place $p$ of $P$, we denote by $M_p$ the marking given by $M_p(p) = 1$ and $M_p(p') = 0 \ \forall p' \neq p$.

Petri nets are represented as follows: places are represented by circles, transitions by boxes, the flow relation is represented by drawing an arc between $x$ and $y$ whenever $(x, y)$ is in the relation and the weight function labels the arcs whenever their weights are greater than 1. A marking $M$ of a Petri net is represented by drawing $M(p)$ black tokens into the circle representing the place $p$.

Definition 2 (Higher-Order Petri Net): An HOPN is formally defined in the same way as the classical Petri nets, $HOPN = (P, T, F, W, M_0)$, where [3]
- $F \subseteq (P \times T) \cup (P^2 \times T) \cup ... \cup (P^n \times T) \cup (T \times P)$ is a set of arcs;
- $W: F \rightarrow N$ is a weight function;
- $M_0: P \rightarrow N$ is the initial token distribution, called the initial marking;

The main difference is the definition of the set of arcs.

Definition 3 (Firing Rule of HOPN): A transition $t$ is said to be enabled or firable if there exist at least one of its $k$th-order input arcs such that each of this arc’s places have at least as many tokens as the weight of this $k$th-order arc. Such an arc is defined as an enabled arc.

C. Open Workflow Petri Nets (oWF-net)
Open workflow nets [2] (oWF-nets) result from the extension of workflow nets (WF-nets) by adding external places for the communication of the external environment of the considered Web services. Now for WF-nets, they result from the application of Petri nets to workflow management.

Definition 4 (Workflow net, WF-net): A Petri net $PN = (P, T, F, W, M_0)$ is a workflow net if and only if [4]:
- $PN$ has two special places: $i$ and $o$; place $i$ is a source place, i.e., $\text{pre}(i) = \emptyset$; place $o$ is a sink place, i.e., $\text{post}(o) = \emptyset$.
- If transition $t^*$ would be added to the set of transitions $T(t^* \notin T)$ and the arcs $(o, t^*)$ and $(t^*, i)$ would be added to the flow relation $F$ of $PN$, the resulting net is strongly connected.

Definition 5 (Open Workflow net, oWF-net): An oWF-net is a 6-tuple $= (P, T, F, IP, OP, F')$, where [5]
- $(P, T, F)$ is a WFN;
- $IP$ is the set of input message places, and $\forall z \in IP, z = \emptyset$;
- $OP$ is the set of output message places, and $\forall z \in OP, z = \emptyset, IP \cap OP = \emptyset$;
- $F' \subseteq (T \times OP) \cup (IP \times T)$ is the set of interface arcs.
III. THE PROPOSED DEFINITIONS OF HOTPN

Definition 6 (Higher-Order Open Workflow Net, HOWFN): An HOWFN = (P, T, F, IP, OP, F') is called a Higher-order Open Workflow Net (HOWFN), if
- \( F \subseteq (P \times T) \cup (P^2 \times T) \cup \ldots \cup (P^n \times T) \cup (T \times P) \) is a set of arcs;
- \{ P, T, F, IP, OP, F' \} is an oWF-Net,

Definition 7 (Firing Rule of HOWFN): A transition \( t \) is said to be enabled or firable if there exist at least one of its \( k \)-th order input arcs such that each of this arc’s places have at least as many tokens as the weight of this \( k \)-th order arc. Such arc is defined as an enabled arc.

Definition 8 (Conversion Procedure): This procedure consists of two phases:
**First phase:** we must ensure the validity of all of the following conditions:
- First: If there exist at least two of transitions that have the same output place(s).
- Second: If there exist one of the previous transitions that have at least two of the incoming arcs.
- Third: If all previous transitions have the same type (timed or immediate transitions).

**Second phase:** we apply the following steps:
1. For each transition contain at least two of incoming arcs. An arcs become a single arc, as in the follow figure:

![Fig.2 The process of integrating arcs](image)

2. Merge all transitions to a single transition, as in Fig.3.

![Fig.3 The process of integrating transitions](image)

After modeled by a HOWFN, web service composition is analyzed and verified by behaviour properties (or called dynamic properties).

Definition 9 (Reachability Property): Let (HOWFN, M0) Higher-Order Open Workflow Net, fire sequence \( \sigma = t_1, t_2, \ldots, t_n \), the sequence of states and transitions \( M_0, M_1, M_2, \ldots, M_n \) and from (HOWFN, M0) a set of all possible fire sequence \( L(HOWFN, M_0) \) or \( L(M_0) \) be given. The HOWFN model is reachable iff:
\[
\exists \sigma_i \in L(M_0) \Rightarrow M_0 \lfloor \sigma > M_n .
\]

Definition 10 (safeness Property): A place of a HOWFN is safe if the number of tokens in it never exceeds one for any reachable marking of the HOWFN. The HOWFN is safe if all its places are safe. i.e., iff:
\[
\forall p \in P, M_i \in R(M_0) \Rightarrow M(p) \leq 1 .
\]

Definition 11 (Boundedness Property): A place of a HOWFN is \( k \)-bounded if the number of tokens in it never exceeds \( k \geq 1 (k \in N) \). The HOWFN is \( k \)-bounded if all its places are \( k \)-bounded. i.e., iff:
\[
\forall p \in P, M_i \in R(M_0) \Rightarrow M(p) \leq k .
\]

Then the bound \( k \) for the HOWFN can be defined as the maximum bound over all its places.

Definition 12 (Liveness Property): (HOWFN, M0) is live; there is no transition that cannot fire by proper transition sequence. i.e., iff:
\[
\forall M_i \in R(M_0) \Rightarrow M_i(p) \neq 0, M_i \lfloor t > M_i, M_i \lfloor t > M_i ,
\]
\( \omega \) is the number of tokens presented at certain place is infinite.

IV. MODELING WEB SERVICES COMPOSITION WITH PETRI NET

Petri Net is one of tools to model discrete event systems [6]. It contains transition related events. In order that an event takes place, some conditions are to be first satisfied. Each of the information about the events and conditions are represented in transitions and places, respectively. The workflow process modeling with Petri Net approach is designed in three ways, i.e. 1) the processes are modeled with transitions, 2) the conditions are modeled with places and 3) cases are modeled with tokens.
We propose a Web portal of credit management to study. We dealt with only the case of a credit request by a client. The web services composition is described by HOWFN in Fig.4. It includes four web services, i.e., Client Service, Portal, Validation Service and Requests processing Service. Each web services can model by a WF-net. Then, we add interface places, which are distinguished from other places, to interconnect the different web services. After this modeling, we can model web services composition by the oWF-net. Finally, we convert oWF-net model to HOWFN model using Conversion Procedure (Definition 8). We note the emergence of higher-order arcs in one transition $t_{190}$- arcs is second-order, i.e., $t_{190}$ consists of the classical arcs and second-order arcs.

From table I-IV shows that the processes/tasks are modeled into transitions. While the places $(p_1, p_2, ..., p_{20})$ show the conditions of status in the credit management and an interface places $(ip_1, ip_2, ..., ip_{10})$ presented in table5.

Initially, the token is in $i_1, i_2, i_3$ and $i_4$. The client ($WS_1$) begins by entering his id (name and password). A service is needed to check the validity of this id by Portal ($WS_2$). If it is not valid, then the customer must enter his id again. Otherwise he can request a credit. Then, he enters his guarantee and balance. The validation service ($WS_3$) intervenes to check the availability of this balance. If it is not available, the process ends, else, the service suits the customer’s request. Here, the processing request service ($WS_4$) puts it in the list of tasks to be performed. A supervisor withdraws a request from this list and evaluates its private part (by checking the relevant information to the client) and its public part. In assessing the public part, the supervisor may either:

- Reject the request: then the supervisor sends a rejection message to the customer.
- Update the application: in this case, the supervisor sends a message to the client to ask him to update his application. This client can either accept or reject this proposal.
- Accept the request: in this case, a supervisor updates and evaluates the request. Then, he sends the customer an offer that he can accept or reject.

V. COMPOSITION ANALYSIS USING HOWFN

The process of composing service can be checked by analyzing corresponding Petri nets model and various well-known analysis methods including reachability graph, state equation and incidence matrix can be applied to Petri nets [7].

In this paper, we utilize reachability graph to perform the verification of a flow model. And the verification will be achieved by combining the construction of the reachability graph with the examination of the validity.

Reachability graph is used to describe all reachable status from the initial status. In a reachability graph, the nodes represent $M_0$ and its all reachable subsequences where: $M_0$ represents the start point; the marking correspond to the final status of the system; the arcs represent the relevant transitions. The path from the first node to a certain node represents an execution sequence.

In this section, we describe the model verification for the web services composition using HOWFN based on the Reachability graph. $i_1, i_2, i_3$ and $i_4$ places are the initial places for the different nodes; the black points in $i_1, i_2, i_3$ and $i_4$ places are the initial marking, each place transit into next place through a transition.

The dynamic behaviour of model can be represented using token in the state of the model. The initial state is $M_0 = \{ i_1, p_1, p_2, p_3, p_4, p_5, p_6, p_7, i_2, i_3, p_8, i_9, i_{10}, i_{11}, i_{12}, i_{13}, i_{14}, i_{15}, i_{16}, i_{17}, i_{18}, i_{19}, i_{20}\}$. If it is not valid, then the customer must enter his id again.

And is the set of all reachable marking of the net is $R(M_0) = \{ M_0, M_1, M_2, ..., M_{20}\}$.

The state space of an Higher-Order Workflow Net (HOWFN) is the set of all reachable states of the net is $L(M_0) = (\sigma_1, \sigma_2, ..., \sigma_{20})$, where

$\sigma_1 = \{ i_1, i_2, i_3, i_{10}\}$

$\sigma_2 = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_{10}\}$

$\sigma_3 = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_6, i_{10}\}$

$\sigma_4 = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_7, i_{10}\}$

$\sigma_5 = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_8, i_{10}\}$

$\sigma_6 = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_9, i_{10}\}$

$\sigma_7 = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_10\}$

$\sigma_8 = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_11\}$

$\sigma_9 = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_12\}$

$\sigma_{10} = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_13\}$

$\sigma_{11} = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_14\}$

$\sigma_{12} = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_15\}$

$\sigma_{13} = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_16\}$

$\sigma_{14} = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_17\}$

$\sigma_{15} = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_18\}$

$\sigma_{16} = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_19\}$

$\sigma_{17} = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_20\}$

$\sigma_{18} = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_21\}$

$\sigma_{19} = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_22\}$

$\sigma_{20} = \{ i_1, i_2, i_3, t_{12}, i_4, i_5, i_23\}$

Fig.5 shows the reachability graph for the validation of Higher-Order Workflow Net. We note follow:

© 2014, IJARCSSE All Rights Reserved
• Model is Boundedness and safe; marking count is always limited under 1.
  \[ \forall p \in P, M_i \in R(M_0) \Rightarrow M(p) \leq k, k = 1 \]
• Model is Reachability, i.e., all places are passable to transit from the initial place.
  \[ \exists \sigma_i \in L(M_0) \Rightarrow M_0[\sigma > M_n] \]
• Model is not liveness; the initial place never comes back, i.e., with a deadlock.
  \[ \neg [\forall M_i \in R(M_0) \Rightarrow M_i(p) \neq \omega, M_i[t > M_i, M_i[t > M_i]] \]

**TABLE I**

<table>
<thead>
<tr>
<th>Transitions</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(t_{1,1})</td>
<td>Enter id of the client</td>
</tr>
<tr>
<td>(t_{1,2})</td>
<td>Request credit</td>
</tr>
<tr>
<td>(t_{1,3})</td>
<td>Enter guarantee</td>
</tr>
<tr>
<td>(t_{1,4})</td>
<td>Enter balance</td>
</tr>
<tr>
<td>(t_{1,5})</td>
<td>Accept the request after validation</td>
</tr>
<tr>
<td>(t_{1,6})</td>
<td>Receive update Message</td>
</tr>
<tr>
<td>(t_{1,7})</td>
<td>Receive offer</td>
</tr>
<tr>
<td>(t_{1,8})</td>
<td>Accept the request</td>
</tr>
<tr>
<td>(t_{1,9})</td>
<td>Accept the request</td>
</tr>
<tr>
<td>(t_{10})</td>
<td>Receive Reject Message</td>
</tr>
</tbody>
</table>

**TABLE II**

<table>
<thead>
<tr>
<th>Transitions</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(t_{2,1})</td>
<td>Valid authentication</td>
</tr>
<tr>
<td>(t_{2,2})</td>
<td>Generate session id</td>
</tr>
<tr>
<td>(t_{2,3})</td>
<td>Ask the client to re-enter id</td>
</tr>
</tbody>
</table>

**TABLE III**

<table>
<thead>
<tr>
<th>Transitions</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(t_{3,1})</td>
<td>Check pay</td>
</tr>
<tr>
<td>(t_{3,2})</td>
<td>Put away Requests</td>
</tr>
<tr>
<td>(t_{3,3})</td>
<td>Ask the client to re-enter</td>
</tr>
</tbody>
</table>

**TABLE IV**

<table>
<thead>
<tr>
<th>Transitions</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(t_{4,1})</td>
<td>Put Request task list</td>
</tr>
<tr>
<td>(t_{4,2})</td>
<td>Withdraw request</td>
</tr>
<tr>
<td>(t_{4,3})</td>
<td>Evaluate (Public part)</td>
</tr>
<tr>
<td>(t_{4,4})</td>
<td>Evaluate (Private part)</td>
</tr>
<tr>
<td>(t_{4,5})</td>
<td>Update the application</td>
</tr>
<tr>
<td>(t_{4,6})</td>
<td>Reject the request</td>
</tr>
<tr>
<td>(t_{4,7})</td>
<td>Send update message</td>
</tr>
</tbody>
</table>
IV. RELATED WORK

In order to fast computation, many researchers prefer Petri nets, since they are well suited for capturing flows in web services, modeling the distributed nature of web services, representing methods in a web service and reasoning about the correctness of the flows.

In [2], Guerfel, et. al. develop a tool for formal verification of Web services composition. These services are modeled by oWF-nets composed of a set of workflow nets interconnected via interface places. These places ensure the communication between the different services. This model is generated to an SMV code that is sent to NuSMV Model Checker. NuSMV verifies certain properties formulated in CTL such as soundness properties.

In [5], Yanhua, Xitong and PengCheng present a Petri net approach to mediation-aided composition of Web services. First, services are modeled as open Workflow Nets (oWFNs) and are composed using mediation transitions (MTs). Second, the modular reachability graph (MRG) of composition is automatically constructed and the compatibility is analyzed.

In [8-10] Lohmann et. al. adopt open workflow nets (oWFN) for modeling BPEL processes and automatically analyze the interactional behavior of a given oWFN.

In [11], P. Massuthe, W. Reisig, and K. Schmidt propose oWFN as a formal model for services that use workflows as their internal control structure. They characterize well-behaving pairs of requester’s and provider’s services and suggest operating guidelines as a convenient and intuitive artifact to realize publish. Then, the find operation reduces to a matching problem between the requester’s service and the operating guideline. Binding of a requester’s and a provider’s service are therefore guaranteed to result in a well-behaving cooperating service.

In [12] K. Schmidt propose adequate model of Open workflow nets for services or parts of inter-organizational business processes. He investigates the problem of controllability, a natural counterpart of soundness in classical workflow nets (as studied by van der Aalst). He distinguishes centralized, distributed, and local controllability and provides solutions to all problems.

VII. CONCLUSION

We proposed in this paper a tool for modeling and analysis of Web services composition. First, these services are modeled by oWF-nets composed of a set of workflow nets interconnected via interface places. Then, web services composition is modeled by HOWFN.

This model reduces the number of transitions to one transition using Conversion Procedure, while maintaining the work of Model as is.

Finally, we analyze the behavioral properties of the model using reachability graph. Where model is Boundedness, reachability and not-liveness (The lack of repetition), i. e., the model is deadlock.
Fig. 4 A Higher-Order Workflow Net model of the credit management

Fig. 5 The Reachability Graph of HOWFN
REFERENCE


