Analyzing Temporal Constraints for Web Services Composition

Ruiqiang YU[1,2], Zhiqiu HUANG[1], Lin WANG[2] and Hongjie ZHANG[2]
[1]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China
[2]Yantai HaiYi Software Co.,Ltd., Yantai, China
Email: [1]ruiqiang.yu@gmail.com, [2]{ wanglin, zhanghj}@haiyisoft.com

***Abstract*—Web service composition has become the optimum technique for Service Oriented Architecture applications because it provides a way to obtain value-added services by combining several Web services. One key issue is that service composition must meet user's deadline requirements. In this paper we focus our attention on modeling and analyzing time-related properties in service composition. A model called extended time Petri net (ETPN) is introduced, in which the temporal constraints are across transitions. The formal definitions of atomic Web service and service composition are proposed based on ETPN. Timestamp state class method is used to analyze the temporal constraints. Some definitions of service composition are presented such as effective path, compatibility, etc. An algorithm is developed to analyze the weak compatibility of two Web services. Furthermore, the approach of service compatibility checking is provided. Finally, a real-life case is given to evaluate our proposal and to demonstrate the applicability of our approach.**

***Index Terms*—Web service; service composition; compatibility analysis; temporal constraint; state class**

I. INTRODUCTION

Nowadays, Web service has become the prominent paradigm for Service Oriented Architecture (SOA for short). More and more companies pack their business functions as Web services and deploy these services on the internet or intranet for the purpose of reuse and value-added service.

Atomic Web Service is too fine-grained to meet user's requirements. Typically many Web services are combined and collaborate with each other to match specific business process. It's a key issue to arrange and assemble Web services correctly and efficiently. Web service composition specifications, such as Web Service-Business Process Execution Language (WS-BPEL)[1] and Web Service Choreography Description Language (WS-CDL)[2], focused on the process and rules in service composition. Rachid Hamadi and Boualem Benatallah proposed Petri net-based algebra to model control flows [3]. They declared that the defined algebra catered for the creation of dynamic and transient relationships among services. Composition forms, such as sequence, choice, parallel and refinement were all discussed. Axel Martens regarded Web services as components of distributed business processes. He modeled Web services and service composition based on workflow Petri nets [4]. Usability, compatibility and environments were discussed in that paper. Petri net was also used in paper [5-6] to model Web service and service composition. The authors of [5-6] kept a careful watch on composition of partially compatible services, and their method was to supply a mediator.

However, the temporal constraints, as a critical non-functional property in service composition, are neglected in all above works. In fact, each component service in the composition has its own time constraints. The time which is spent on service composition must be treated seriously for the cost and safety concern because Web services are deployed in network environment. In some scenarios, we often expect that service composition satisfies some global temporal constraints. These constraints can be satisfied only if all the services participating in the composition are committed to respect their own local time constraints. The more time is consumed on service composition, the more cost and danger will follow. If the time consumed on service composition is too much to satisfy the global temporal constraints, the service composition action must be aborted. Therefore it is significant to study the temporal constraints for service composition.

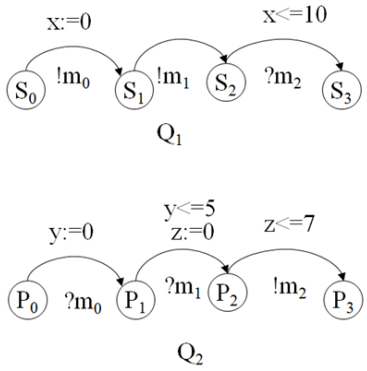

Fig.1 Service Composition Model by WSTTS

Corresponding author: Ruiqiang YU (ruiqiang.yu@gmail.com).

doi:10.4304/jsw.8.5.1051-1057

Nawal Guermouche and his team focused their interest on the compatibility analysis of Web services especially regarding the temporal requirements [7]. Web services timed transitions system (WSTTS) was intended for this use. Local consistency of transitions and external constraints inference were discussed in detail. It's a good job, but their research can't solve some issues just as Fig.1 shows. The reason is that their approach isn't suitable for the scenario where different zero-based clocks reside in Web service composition.

In our work, a preferable method is proposed for Web service compatibility analysis. Compared with the existing works, we make the following contributions:

1) We present an extended time Petri net (EPTN for short) model which can describe the time-related properties of service accurately. Clock-set is used to denote the cross-transition temporal constraints in service composition.

2) A refined enumerative approach, which is named timestamp state class method, is presented to analyze the temporal constraints.

3) We discuss and formally define the entire-path, valid-path and compatible degree for service composition.

4) We use a state-space based method to analyze the service compatibility. Temporal constraints are regarded as a critical factor.

5) We validate our approach through a real-life case.

The rest of the paper is structured as follows. Section 2 depicts the formal definition of extended time Petri net, and the transition firing rules are also discussed in detail. Timestamp State Class method for temporal constraints analysis is introduced in section 3. Section 4 formally defines the concepts of atomic Web service and service composition. Furthermore, entire-path, valid-path and compatible degree are discussed in this section. The approach of composition analysis is also presented. In section 5 a real-life case is provided to illustrate the feasibility of our approach. Conclusions and future work are presented in section 6.

II. Extended Time Petri Net

Time Petri net (TPN), which is thoroughly discussed by Merlin [8] has become one of the most powerful formal method to modeling system where time is a critical constraint. But a flaw in TPN is that each temporal constraint is local restriction, so TPN can't annotate the following case: when the system (Fig.1) runs from S_0 to S_3 via several transitions, the time cost should meet the constraint $x \le 10$. To get over this shortcoming, extended time Petri net is presented in our work.

Definition 1. Let X be the set of clock variables, and then clock constraint φ is defined as follows:

$$\varphi := x \le n \mid x < n \mid x > n \mid x \ge n \mid x = n \mid \varphi_1 \wedge \varphi_2$$

where x is a clock in $X, x \in X$ and $n \in N, N$ is the natural number set. This idea comes from Rajeev Alur [9].

Definition 2. Extended time Petri net is an eight-tuple $ETPN = (P,T,F,\alpha,\beta,X,E,M_0)$ (Fig.2), where:

1) P and T are the finite sets of places and transitions respectively, $P \cap T = \Phi \wedge P \cup T \ne \Phi$;

2) $F = P \times T \cup T \times P$, is the set of arcs (flow relation);

3) $\alpha : T \to N$ and $\beta : T \to (N \cup \infty)$ are the earliest firing time (*EFT*) and latest firing time (*LFT*) of transition t, respectively;

4) X is the clock set;

5) E is the clock constraint function. $\exists p \in P \wedge \exists x \in X$, $E(x,p) = (N \psi x) \vee (x \psi N)$, where $\psi \in \{>, \ge, =, <, \le\}$, $E(x,p)$ denotes the clock constraint of clock x for place p;

6) M_0 is the initial marking.

$[\alpha, \beta]$ is the firing interval of t and it will change when net runs[8]. To avoid confusion, we denote $[\alpha_t^s, \beta_t^s]$ for the firing interval of t in initial marking M_0 and $[\alpha_t, \beta_t]$ for other marking M.

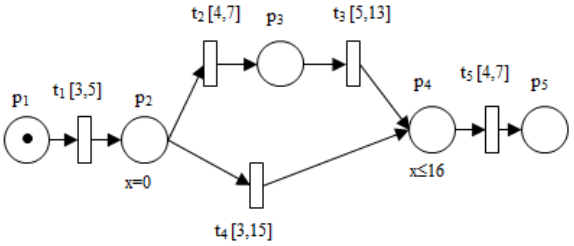

Fig.2 Extended Time Petri Net

In ETPN, places can be entitled with clock functions. The semantic of clock functions is when a place contains a token, the clock functions associated with that place must be satisfied. For the clock x in Fig.2, $E(x,p_2)$ is initialized to 0 at the moment in which p_2 owns a token. The operation of system is accompanied by marking change. When $M(p_4) > 0$, $E(x,p_4) \le 16$ must hold. The operating rules of ETPN are different from those of TPN because of the clock constraints. The transition firing rules of ETPN should be redefined.

Definition 3. For a reachable marking M in ETPN, transition t is enabled if and only if $\forall p \in {}^\bullet t, M(p) > 0$. The set of enabled transitions in marking M can be marked with $En(M)$.

Definition 4. In ETPN, if a transition t can be fired in marking M, we say it is firable, and the followings must hold:

1) t is enabled in M;

2) Let θ be the time span from the moment at which t is enabled to the moment when t is fired, then
$$EFT(t) \le \theta \le min(LFT(t_k))$$
should be met, where k ranges over the set of transitions enabled by marking M.

3) $\forall p \in t^\bullet$, if there is a clock constraint $E(x)$ on the place, let $v(x)$ be the current figure of clock x, then $v(x) + \theta \le E(x,p)$ must be satisfied.

$Fir(M)$ is the set of transitions which are firable in marking M.

In Fig.2, when t_1 is fired, new marking is $M(p_2)=1$, then $En(M)=\{t_2,t_4\}$ and $v(x,p_2)=0$. From now on, let's increase the value of θ. When $4\le\theta\le7$, t_2 can fire because $v(x,p_2)+\theta=[4,7]<E(x,p_2)$. So t_4 is also firable. Firing t_2 in the interval $4\le\theta\le7$ leads to a new marking $M(p_3)=1\wedge v(x,p_3)=[4,7]\wedge En(M)=\{t_3\}$. For t_3, if $5\le\theta\le13$, $9\le v(x,p3)+\theta\le20$, so $E(x,p_4)\le16$ can't hold. This shows that transition sequence $\sigma=t_1\bullet t_2\bullet t_3$ can't ensure the clock functions be totally satisfied, and $\sigma=t_1\bullet t_4$ is a better choice for this reason.

III. State Class and State Space for ETPN

It's learned from definitions 3-4 that clock constraints X of places must be taken into account besides the firing interval $[\alpha_t,\beta_t]$. Because $E(x)$ can cover several transitions and can be superimposed (there will be one or more clock constraints on one place, for example $E(x,p_1)<16\wedge E(y,p_1)>3$), the method of calculating the transition firable interval is no longer applicable for operating analysis of EPTN.

Berthomieu presented state class method in [10], but his approach only considered the local temporal constraints for firable transitions, and can't satisfy the demand of analyzing the clock constraints which range over several transitions.

We can learn from definition 2 that, in ETPN, the clock constraints are the time span between a state-pair in the same trace during the course of system operating. So, if one can denote each state with a timestamp which is based on the same *ORIGINAL CLOCK*, it's easy to calculate whether each clock function is fulfilled. In our work, timestamp state class is introduced for this purpose and we can verify the satisfaction of clock constraints through the following steps:

1) Generate the state space of ETPN with timestamp state class;
2) For each trace of state space, verify the clock constraints satisfaction by value the difference of timestamp.

A Timestamp State Class

Definition 5. Timestamp state class (TSC for short) is a tuple $C=(M,Q,TS)$, where

1) M is the marking of EPTN;
2) Q is a time domain. It is identical to the Q of Berthomieu's strong state class. For any transition enabled in marking *M*, an vector *Q(t)* is presented. *Q(t)* stands for the time elapsed since that transition was last enabled. Let θ represent the time elapsed since current marking is entered, then we have $\forall t\in En(M), EFT_t\le Q(t)+\theta\le LFT_t$.
3) *TS* is the temporal interval based on the global original clock, and is used to represent the timestamp of state class. That is: *TS* is the time span from the initial state class to current state class. The *TS* of initial sate class is 0.

The state space of ETPN is a state transition system, C_0 denotes the TSC of initial state.

In this paper, the action that transition t is firable in state C at time θ and its firing leads to state C' will be denoted as:

$$C\xrightarrow{t@\theta}C'.$$

The method of computing TSC can be depicted as follows:

1) Start from the initial state class $C_0=(M_0,Q_0,TS_0)$;
   a) M_0 is the initial marking of net;
   b) For any transition t which is enabled in marking M_0, $Q_0(t)=0$;
   c) $TS_0=[0,0]$ is the initial timestamp state class. It is the origin of global clock;
2) If a transition t_i is firable in marking M, the following should hold:
   a) $t_i\in En(M)$;
   b) Let θ represent the time elapse in marking M and $\gamma=Q(t_i)$ (γ denotes the accumulated time elapsed since the transition t_i was last enabled), then the followings hold:

$$\begin{cases} 0\le\theta \\ \theta+\gamma\ge EFT(t_i) \\ \theta+\gamma\le\min(LFT(t)),\forall t\in En(M) \end{cases} \quad (1)$$

3) If the firing of t_i results in $C\xrightarrow{t_i@\theta_i}C'$, the state class $C'=(M',Q',TS')$ can be calculated by
   a) Calculate marking M' with formula $M'=M-W(p,t_i)+W(t_i,p)$, where W is the function of token consumption. For simplicity, let us suppose $W=1$ in this paper.
   b) Clock domain Q' can be calculated as follows:
      - Calculate θ according formula (1);
      - For each transition *t* that is enabled at *M* and still enabled at $M''=M'-W(p,t_i)$, let $\gamma_t'=\gamma_t+\theta$ and $\gamma_t'\le\min(LFT(t_j)), j\ne i$ where t_j covers the transitions that can be fired in M';
      - For the transition which is newly enabled at M', $\gamma'=0$;
      - Eliminate transitions disabled in M'.
4) For the new TSC, the timestamp $TS'=TS+\theta$. This means that the global timestamp of *M* is TS. After that, temporal interval θ is elapsed, the firing of transition t_i leads to a new marking M'. TS' is the global timestamp of M'.
5) Repeat the above steps and get the whole timestamp state class space.

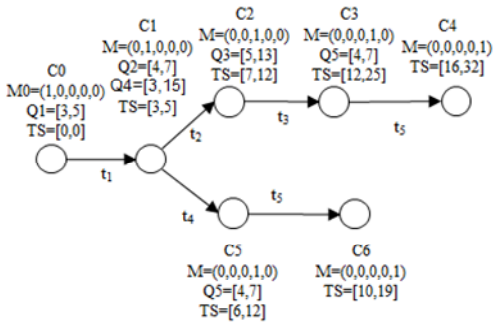

Fig.3 Timestamp State Class Space

By this means, one can get the state space of ETPN (in Fig.2) just as the result shown in Fig.3.

B Verifying the Clock Constraints

Definition 6: Let $TS_1=[a_1,b_1]$ and $TS_2=[a_2,b_2]$ be two clock domains, if $a_1 \le a_2$ and $b_1 \le b_2$ hold, $TS_1+TS_2=[a_1+a_2,b_1+b_2]$ and $TS_2-TS_1=[a_2-a_1,b_2-b_1]$ are also clock domains.

Theorem 1: If timestamp state class $C'=(M',Q',TS')$ can be reached from timestamp state class $C=(M,Q,TS)$ through a certain transition sequence, the time span between them is $[\downarrow TS'-\downarrow TS,\uparrow TS'-\uparrow TS]$, where $\downarrow$ and $\uparrow$ are the lower and upper bounds of TS respectively.

Proof. It is learned from the algorithm of TSC that a feasible transition sequence $C \xrightarrow{t_1@\theta_1} \cdots C_n \xrightarrow{t'@\theta'} C'$ must exist for any couple TSC in the same trace. For the sake of $TS_1=TS+\theta_1=[\downarrow TS+\downarrow\theta_1,\uparrow TS+\uparrow\theta_1]$, we can get $\theta_1=TS_1-TS=[\downarrow TS_1-\downarrow TS,\uparrow TS_1-\uparrow TS]$. The temporal interval from TS_1 to TS_2 is $\theta 2=TS_2-TS_1=[\downarrow TS_2-\downarrow TS_1,\uparrow TS_2-\uparrow TS_1]$. The rest may be deduced by analogy. At last $\theta'=TS'-TS_n=[\downarrow TS'-\downarrow TS_n,\uparrow TS'-\uparrow TS_n]$. The total time elapse $\theta=\theta_1+\theta_2+\cdots+\theta'=[\downarrow TS_1-\downarrow TS+\downarrow TS_2-\downarrow TS_2+\cdots \downarrow TS'-\downarrow TS_n,\uparrow TS_1-\uparrow TS+\uparrow TS_2-\uparrow TS_2+\cdots\uparrow TS'-\uparrow TS_n]=[\downarrow TS'-\downarrow TS,\uparrow TS'-\uparrow TS]$. The theorem is established. ■

lemma 1: The time span between timestamp state classes in same trace can be used to verify whether the clock constraints of ETPN are satisfied.

Proof. It's learned from definition 2 that the clock functions are used to record the time cost from one state to another. By TSC, one can get the value of original clock for any state in the same trace. Furthermore, one can calculate the time span between any TSC-pair in the same trace via theorem 1. So the difference of timestamp of two state classes can be used to verify the satisfaction of clock constraints. ■

Considering the transition sequence $\sigma=t_1\bullet t_2\bullet t_3$ in Fig.3, TS_1=[3,5] for $M_1=(0,1,0,0,0)$ and TS_2=[12,25] for $M_2=(0,0,0,1,0)$. In light of theorem 1 and definition of ETPN, the clock function $E(x,p_4)\le 16$ does not hold. But this clock constraint is guaranteed by transition sequence $\sigma=t_1\bullet t_4$. This result stands for the previous conclusion in section 2.

Ⅳ Modeling Web Service and Service Composition

In real life, Web service is usually deployed in net environment. For the concern of safety and finance, Web service must be composed and finished within a certain time span. Because enterprises keep a careful watch on operating states and the time consumption between the states change, one can model Web service and service composition with ETPN.

A Modeling Web Service

Definition 7. Web service can be depicted by ETPN with a tuple $W=(PI,PM,,T,F,\alpha,\beta,\Psi,X,E,M_0)$, where

1) PI is the set of internal places, figured with solid line. It indicates the internal state of business. In PI, there is only one place p_e which holds $p_e^\bullet=\phi\wedge{}^\bullet p_e\neq\phi$. The execution of business is normally finished if and only if $M(p_e)>0$.
2) PM is the message places set, figured with dotted line. It indicates the message exchange when services interact with each other. $\forall p\in PM,({}^\bullet p=\phi\wedge|p^\bullet|=1)\vee(p^\bullet=\phi\wedge|{}^\bullet p|=1)$ hold.
3) Ψ is the message polarity, where $\Psi: PM\to\{-1,1\}$. When a place holds $\forall p\in PM\wedge p^\bullet=\phi,\Psi(p)=-1$, it is called output-message-place. Contrarily some places are called input-message-places if and only if $\forall p\in PM\wedge{}^\bullet p=\phi,\Psi(p)=1$.
4) Others are the same to definition 2.

B Modeling Web Service Composition

Definition 8 (Service Composition). Given a pair of Web services $W_i=(P_{Ii},P_{Mi},T_i,F_i,\alpha_i,\beta_i,\Psi_i,X_i,E_i,M_{0i}),i=1,2$, the composite Web service $CW=(P_I,T,F,\alpha,\beta,\Psi,X,E,M_0)$ is the fusion of W_1 and W_2 via message places, if and only if the following conditions are satisfied.

1) $\forall p_1\in P_{I1},\exists p_2\in P_{I2},\Psi(p_1)\bullet\Psi(p_2)=-1$. For any message place in W_1, W_2 must have a coupling message place. The coupling messages have converse polarity and same message type. That is, any output message of W_1 must be the input message of W_2, and vice versa.
2) For the composite Web service CW, the clock constraint function $E(X)=E(X_1)\cup E(X_2)$ still holds.
3) W_1 and W_2 will normally finish their business process via a transition sequence after all message are exchanged. i.e. $M(p_{e1})>0\wedge M(p_{e2})>0$ hold.

If so, W_1 and W_2 are called compatible.

4) For CW, $P_{I1}\cap P_{I2}=\phi;P_I=P_{I1}\cup P_{I2}\cup(P_{M1}\cap P_{M2})$ hold. That means:
   - There is no message place in composite Web service;

- The internal place set of CW is the union of internal place set and message set of W_1 and W_2.

Definition 9: If two Web services W_1 and W_2 can fuse into a composite Web service while their clock constraint functions are all neglected, they are called weak compatible, and the composite Web service is called weak-composition service.

Comparing the complete composition mentioned in definition 9 with definition 8, we can learn that weak composition only doesn't take into account the clock functions .

Definition 10: For a weak-composition Web service which is composed of W_1 and W_2, the transition sequence $\sigma = t_1 t_2 \cdots t_n$ can be named as complete path if and only if $M_0 \xrightarrow{t_1@\theta_1} M_1 \xrightarrow{t_2@\theta_2} M_2 \cdots \xrightarrow{t_n@\theta_n} M_n$ is satisfied, where $M_n(p_{e1}) > 0 \wedge M_n(p_{e2}) > 0$.

Let σ_1, σ_2 be the projection of σ in W_1 and W_2, then $(\sigma_1 \bigcap \sigma_2 = \phi) \wedge (\sigma_1 \bigcup \sigma_2 = \sigma)(\forall t \in \sigma_i, t \in T_i, i = 1,2)$. Moreover, σ_1, σ_2 are the complete path of W_1 and W_2 respectively.

$|\sigma|$ is the deepness and $|\sigma| = |\sigma_1| + |\sigma_2|$.

Definition 11: For a complete path, if the $E(X) = E(X_1) \bigcup E(X_2)$ of weak-composition Web service can be satisfied, it is called effective path, otherwise is called noneffective path.

Definition 12: For a weak-composition Web service composed of two atomic Web services W_1 and W_2, if each complete path is effective, W_1 and W_2 are called complete compatible, or else, if there is at least one complete path is effective, this Web service couple is called partial compatible. They are called incompatible when there is no effective path.

Definition 13: For a weak-composition service CW composed of two atomic Web services W_1 and W_2, let m, n denote the complete path and effective path of CW respectively, then the compatible degree ξ of W_1 and W_2 can be defined by:

$$\begin{cases} \xi = n/m; if & m > 0 \\ \xi = 0 \quad if & m = 0 \end{cases} \qquad (2)$$

C Analyzing the Compatibility of Web Services

It can be learned from definition 12 and definition 13 that the compatibility of Web services depends on the relation between complete path and effective path. If two Web services are compatible, the composite Web service fused by them must satisfy definition 8. We can analyze the compatibility of Web services with following steps:

1) Neglect the clock constraints ($E(X_1), E(X_2)$) of atomic Web services W_1 and W_2 , and then build the weak-composition Web service which satisfies definition 9;
2) If definition 9 does not hold, W_1 and W_2 must be incompatible, otherwise:
3) Compute the state space of weak-composition Web service using TSC, and compute the complete path;
4) Verify whether the clock functions $E(X_1), E(X_2)$ are all satisfied according to theorem 1 and lemma 1, and the effective path can be identified ;
5) Compute the compatible degree with definition 13.

How to compute the TSC and get the state space of EPTN are depicted in section 3, so we merely present the approach about how to get the weak-composition Web service from two atomic Web services here.

Let $W_1 = (P_{I1}, P_{M1}, T_1, F_1, \alpha_1, \beta_1, \Psi_1, X_1, E_1, M_{01})$ and $W_2 = (P_{I2}, P_{M2}, T_2, F_2, \alpha_2, \beta_2, \Psi_2, X_2, E_2, M_{02})$ be two atomic Web services. P_{M1}, P_{M2} are their message place sets, $Fir(M)$ denotes the firable transition set in marking M, then one can discern if W_1 and W_1 are weak compatible by following method:

Input : $W_i = (P_{Ii}, P_{Mi}, T_i, F_i, \alpha_i, \beta_i, \Psi_i, X_i, E_i, M_{0i}); i = 1,2$

Output: *bool;*

Algorithm:

$S_{Mi} = P_{Mi}$;

$M_i = M_{i0}; i = 1,2$;

bool v_Continue=true;
while(v_Continue)
{
v_Continue=false;
while($\exists t \in Fir(M_i) \wedge t^\bullet \notin S_{Mi}$ *)//i=1,2*
{

$M_i \xrightarrow{t_i@\theta} M_i'$;

$M_i = M_i'$

v_Continue=true;
}
If($(|t_i^\bullet| = |{}^\bullet t_j|) \,\&\&\, (\Psi(t_i^\bullet) \bullet \Psi({}^\bullet t_j) = -1)$ *){*

$t_i^\bullet = {}^\bullet t_j$

$S_{Mi} = S_{Mi} - t_i^\bullet; S_{Mj} = P_{Mj} - {}^\bullet t_j;$;

}
}
if ($S_{Mi} \neq \phi$ *)//i=1 or 2*
return false;
else
return true ;//be weak compatible

V. EXAMPLE

It is a real-life scene in Fig.4, (a) and (b) are two atomic Web services:

1) Customer service (Fig.4 (a)) depicts an E-Business process: customer orders some products and pay for the merchant via third-party checkout (TPC).

2) TPC service (Fig.4 (b)) is the process of third-party checkout. When receiving the request message, TPC asks some questions and then return the acknowledgement.

The semantics of all places and transitions in Fig.4 are shown in Table 1 and Table 2 respectively.

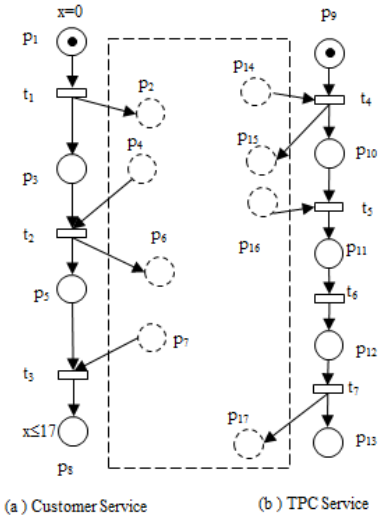

Fig.4 Atomic Web services

TABLE I.
SEMANTICS OF PLACES

Place	Explanation	Ψ
p1	Initial state of customer	
p2	Send order list message to TPC	-1
p3	Order list is submitted	
p4	TPC returns transaction code	1
p5	Wait for processing result	
p6	Send bill type to TPC	-1
p7	TPC returns the processing result	1
p8	Customer service is finished	
p9	TPC is initialized to ready	
p14	TPC receives the order list	1
p15	Send transaction code to customer	-1
p10	TPC waits for payment	
p16	Payment notice	1
p11	TPC gets payment notice	
p12	Process successfully	
p17	Result is returned to customer	-1
p13	TPC service is finished	

SEMANTICS OF TRANSITIONS

Transition	Explanation	[α,β](s)
t1	Call TPC service to pay	[1,2]
t2	Choice bill type	[2,4]
t3	Handle successfully, log out	[1,2]
t4	Calculate cost and supply transaction code	[4,9]
t5	Receive payment notice and deal information	[1,2]
t6	Process the transaction	[3,6]
t7	Return result to customer	[1,2]

With the method listed in subtitle *C* of section 4, we can get the weak-composition Web service just as Fig.5. Furthermore we can get its TSC spaces just as Table 3 shows.

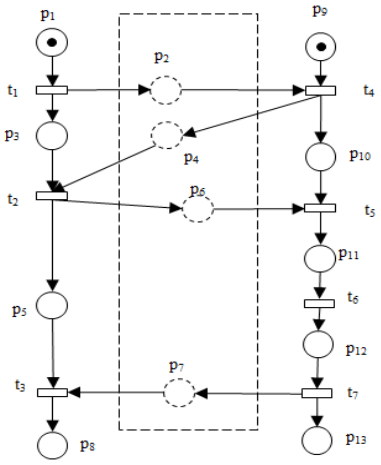

Fig.5 Service Composition

TABLE III.
THE TIMESTAMP STATE CLASS SPACES

State Class	M	TS	Firing Trans	Firing Domain(θ)	Successor State Class
C0	p0,p9	[0,0]	t1	1≤θ≤2	C1
C1	p2,p3,p9	[1,2]	t4	4≤θ≤9	C2
C2	p3,p4,p10	[5,11]	t2	2≤θ≤4	C3
C3	p5,p6,p10	[7,15]	t5	1≤θ≤2	C4
C4	p5,p11	[8,17]	t6	3≤θ≤6	C5
C5	p5,p12	[11,23]	t7	1≤θ≤2	C6
C6	p5,p7,p13	[12,25]	t3	1≤θ≤2	C7
C7	p8,p13	[13,27]			

Table 3 shows that the corresponding TSC of marking $M(p_1)>0$ is C_0, and C_7 is for marking $M(p_8)>0$. According to theorem 1 and lemma 1, the time span between C_0 and C_7 is $TS_7 - TS_0 = [13,27]$, but the clock function of place p_8 is $E(x, p_8) \le 17$. The TSC of C_7 shows: it is probable that the execution time of composition service would go beyond 17 although the lower bound of TSC is less than the clock constraint. So this composition is unreliable. For the weak-composition Web service in Fig.5, the following results could be derived by definition 10-13:

1) Complete path is 1;
2) Effective path is 0;
3) Compatible degree is 0

So, atomic Web services in Fig.4 are incompatible.

VI. CONCLUSIONS

Service composition software based on SOA is the accepted trend. When two Web services provide complementary functionality and could be linked together in principle, the temporal constraints across these Web services should be treated as a critical non-functional property. Few researches focus on temporal constraint for Web service composition. Once the temporal constraints can't be satisfied, the composition of Web services will

be regarded as failure. Existing research can't meet the scenario where different zero-based clocks reside in Web services composing. In this paper, we address this challenge based on Petri net and present a motivating scenario of E-Business to illustrate our approach.

We extend time Petri net with ETPN. Clock functions are used to denote the time span between different states. A new timestamp state class approach is presented to analyze the clock functions in ETPN. Subsequently we model atomic Web service and service composition by ETPN. Furthermore we discuss several Web service compatibility questions such as weak-compatible, complete path, effective path and compatible degree. A method is presented to analyze whether two Web services can be fused into a weak-composition service. For the weak-composition Web service, one can analyze whether the clock constraints are satisfied with the TSC method which is thoroughly depicted in this paper. Finally the feasibility of our approach is illustrated by an example and the result supports our research.

Future work includes applying our research to more real-life cases. On the one hand the reachability-based methods are computationally expensive. On the other hand we just hypothesize that the output message of an atomic service would exactly be the input message of another Web service. In fact, there are too many partial-message-compatible scenarios. Additionally, according to "Six Degrees of Separation" theory, whatever complicated business process could be composed by less than six Web services. In the future, we will devote our efforts to constructing a proxy which would address partial-message-compatible issue and automatically choose less than six Web services for composition to meet business requirements. An automatic tool is also wanted for this use.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (61100034 and 61170043), China Postdoctoral Science Foundation (Grant 20110491411), Jiangsu Planned Projects for Postdoctoral Research Funds (Grant 1101092C); the IT Industry Foundation of Ministry of Industry and Information Technology of China under Grant No.[2008]97. The example was coded and examined by software engineers of Yantai HaiYi Software Co., Ltd.

REFERENCES

[1] OASIS. "Web Services Business Process Execution Language Version 2.0," 2007. *http://docs. oasis-open .org/ wsbpel/ 2.0/CS01/wsbpel-v2.0-CS01.html*

[2] W3C. Web Services Choreography Description Language Version 1.0, 2005. http://www.w3.org/TR/ 2005/CR-ws-cdl-10-20051109/

[3] Rachid Hamadi, Boualem Benatallah. A Petri Net-based Model for Web Service Composition. In *Proc of the 14th Australasian database conference, Australian Computer Society,* pages 191-200, 2003.

[4] Axel Martens. On Compatibility of Web Services. *Petri Net Newsletter*. Pages 65:12-20, 2003.

[5] Tan W, Fan YS, Zhou MC. A Petri Net-Based Method for Compatibility Analysis and Composition of Web Services in Business Process Execution Language. *IEEE Transactions on Automation Science and Engineering,* 2009,6(1): pages 94-106,2009.

[6] PengCheng Xiong, YuShun Fan and MengChu Zhou, Fellow. A Petri Net Approach to Analysis and Composition of Web Services. *IEEE Transactions on systems, man and cybernetics* 2010. 40(2), pages 376-387, 2010

[7] Nawal Guermouche, Olivier Perrin and Christophe Ringeissen. Timed Specification For Web Services Compatibility Analysis. *Electronic Notes in Theoretical Computer Science*, 2008,200: pages 155–170, 2008

[8] P. Merlin. A study of the recoverability of computing system. *Univ. California*, 1974.

[9] Rajeev Alur. Timed Automata. In: *N. Halbwachs and D. Peled (Eds.) Proc of CAV'99*, volume1633 of LNCS, pages 8-22, Springer 1999.

[10] Berthomieu B, Michel Dim. Modeling and verification of time dependent systems using time Petri nets. *IEEE Transactions on Software Engineering*, volume 17(3), pages 259-273, 1991.

Ruiqiang YU was born in 1973. He is currently a Ph.D student at Nanjing University of Aeronautics and Astronautics, Nanjing, China. He received the M.S. degree and B.S. degree in Hohai University, Nanjing, China, in 1996 and 1999 respectively. His research interests include software engineering, Web service and formal methods.

Zhiqiu HUANG was born in 1965. He received his Ph.D. degree in computer science from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1998. He is currently a professor of College of Computer Science and Technology of Nanjing University of Aeronautics and Astronautics. His main research interests include, but are not limited to, software engineering, Web service, formal methods, knowledge discovery and requirement engineering.

Lin WANG was born in 1966 and received his M.S. degree and B.S. degree in Jilin University, Changchun, China, in 1989 and 1992 respectively. He is currently a professor level senior engineer. His main research interests include cloud computing and SOA.

Hongjie ZHANG was born in 1968 and received his M.S. degree in Huazhong University of Science and Technology, Wuhan, China, in 2002. As a senior engineer, his research interests include Web service, E-Government and E-Business.