Compatibility Analysis of Time Open Workflow Nets

Zohra Sbaï[1], Kamel Barkaoui[2], and Hanifa Boucheneb[3]

[1] Université de Tunis El Manar,
Ecole Nationale d'Ingénieurs de Tunis,
BP. 37 Le Belvédère, 1002 Tunis, Tunisia
`zohra.sbai@enit.rnu.tn`

[2] Conservatoire National des Arts et Métiers
292 rue Saint Martin, Paris Cedex 03, France
`kamel.barkaoui@cnam.fr`

[3] Ecole Polytechnique de Montréal,
P.O. Box 6079, Station Centre-ville, Montréal, Québec, Canada
`hanifa.boucheneb@polymtl.ca`

Abstract. Because of their expressive power, Petri nets are widely used in the context of concurrent and distributed systems. We study in this paper a sub class of time Petri nets, named time open workflow nets (ToWF-nets), used to interconnect time constrained business processes. To interact correctly with each other, these processes have to be compatible. This include not only composability of the involved processes but also the correct execution of the overall composite system. In this context, we suggest in this paper to study the compatibility of ToWF-nets in different aspects and to provide a formal approach to characterize and verify this property. This approach is based on qualitative and quantitative analysis ensured by TCTL model checking.

Keywords: Time open workflow nets, Reachability analysis, Compatibility, TCTL

1 Introduction

The workflow technology has shown a great interest to be adopted within organizations or even inter organizations. A workflow is the result of the automation of a business process, in whole or in part [35]. A business process consists of a number of tasks and ensure all the conditions that determine their order. A business process which involves different partner companies is said to be inter organizational. Indeed, it is a specific representation for which coordination mechanisms between activities, applications or participants can be managed by a workflow management system (WfMS). The success of workflow technology explains the fact that the number of emerging WfMS is growing fast. And therefore the need for effective mechanisms and tools for modeling and analysis of workflow processes is crucial.

Open workflow nets (oWF-nets) form a sub class of Petri nets which is successfully used to model workflow processes which communicate with other partners via interfaces. This class is promisingly used in the context of Web services orchestration and choreography. In fact each service in a composition is modeled by a workflow net augmented by interface places used to communicate with other services. In this way, one can guarantee the conversation between the processes interacting with each other. The conversation considered here is involved through the two well known behaviors: operational and control. An operational behavior is a behavior specific to each partner according to its business logic. A control behavior describes the general behavior of any process related to composite Web services. While we focused in a previous work [33] on the verification of oWF-nets, we propose in this paper to extend oWF-nets by modeling timing constraints and to study their analysis.

Several time Petri nets extensions were proposed in the literature which differ in their semantics and their analysis techniques. We propose to adopt in this work the time Petri nets, proposed by Merlin [31], in which transitions are labeled by intervals specifying the minimum and maximum delays of their firings. We extend, therefore oWF-nets by associating with each transition a minimum and maximum amount of time needed to its execution. The obtained model is said to be time open workflow net (ToWF-net). We define formally this model and present its semantics as well as the computation of its state space. The efficient construction of the state space leads to efficient techniques of ToWF-nets reachability analysis.

Dealing with time in inter organizational processes, we propose to study the compatibility of the processes communicating together. This property is not only related to the ability of processes to communicate (i.e. composability) but also to the correct and deadlock-free execution of the composite process. In this context, we propose to define compatibility classes of ToWF-nets and to emphasize a method of their verification.

To verify ToWF-nets compatibility, we propose to use formal methods due to their solid theoretical basis. More precisely, we present an analysis method based on model checking of the studied properties. In fact, Model checking is an automated verification technique for proving that a model satisfies a set of properties specified in temporal logic. Given a concurrent system Σ and a temporal logic formula φ, the model checking problem is to decide whether Σ satisfies φ. Hence, we have to formulate in temporal logic the properties to be verified. This kind of verification is situated at the design phase, allowing thus to find design bugs as early as possible and therefore to reduce the cost of failures. This, especially, permits us to check as early as possible if two or more processes are compatible before their composition. We express the proposed compatibility properties in Timed Computation Tree Logic (TCTL).

The rest of this paper is organized as follows. We propose in section 2 the ToWF-nets to model inter organizational workflow processes with timing delays. The same section presents the semantics of ToWF-nets in terms of states and their evolution and exposes a case study. Section 3 is dedicated to present some results of the reachability analysis of ToWF-nets. We focus in section 4 on the verification of ToWF-nets compatibility. We begin with expressing the properties in TCTL and then we present some experiments in Romeo model checker. Section 5 exhibits related work and finally section 6 concludes the paper and announces future work.

2 Time open WorkFlow nets

In this section, we propose a new sub class of time Petri nets modeling workflow processes with interface places used to communicate with other partners. To begin with, we present a Petri nets modeling of communicating processes and then we propose a time extension.

2.1 Petri nets modeling of communicating workflows

Nowadays, many organizations are implementing their business functionality and outsource their services on the internet. Thus, the selection as well as inter organizational and heterogeneous integration with efficiency and effectiveness of Web services during the execution has become an important step in Web services applications. In particular, if no service can meet the needs of the user, there should be a possibility to combine existing services to meet the demands required by the user. This trend has led to the notion of the composition of Web services.

In fact, the composition or aggregation of Web services is a process that involves building new services or aggregates called composite services by assembling existing services. The composite service is a value added service that can be the distribution of basic services or composite ones.

This composition can be modeled by means of a Petri net class named open workflow nets [25,33,29,30]. We model each involved process by an open workflow net possessing interface places used to communicate with other processes. Thus the conversation and interaction between the involved processes are guaranteed. The communication considered here is entangled through operational and control behaviors. The operational behavior is a behavior specific to each partner according to its business logic while the control behavior describes the general behavior of any process related to composite Web services.

As mentioned above, open workflow nets are mainly an extension of workflow nets (WF-nets) to model workflow processes which interact with other workflow processes via interface places. Simple WF-nets [7]is a result of Petri nets' application to workflow management. The choice of Petri nets is based on their formal semantics, expressiveness, graphical nature and the availability of Petri nets based analysis techniques and tools.

A Petri net is a 4-tuple $N = (P, T, F, W)$ 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 $W : (P \times T) \cup (T \times P) \rightarrow \mathbb{N}$ is the weight function of N satisfying $W(x, y) = 0 \Leftrightarrow (x, y) \notin F$. If $W(u) = 1$ $\forall u \in F$ then N is said to be ordinary net and it is denoted by $N = (P, T, F)$. For every node $x \in P \cup T$, the set of input nodes of x is defined by ${}^\bullet x = \{y|(y,x) \in F\}$ and the set of output nodes is denoted by $x^\bullet = \{y|(x,y) \in F\}$. We refer the reader to [8], for more Petri nets notations used in this paper.

A Petri net which models a workflow process is said to be a WF-net [3]. An ordinary Petri net $N = (P, T, F)$ is a WF-net iff N has one source place i named initial place (containing initially one token) and a sink place f named final place. In addition to this characteristic, in a WF-net, every node $n \in P \cup T$ is on a path from i to f.

For a composition, we propose to model each Web service by a WF-net specifying the set of tasks to be performed and their routing. The conversation between the different Web services is ensured by communication places used for messages sending. We are thus using open WF-nets (oWF-nets) which generalizes the classical WF-nets by introducing interface places for asynchronous communications with partners. Hence, we model a composition by a set of oWF-nets communicating via interface places. These places connect only transitions of different processes.

2.2 Time extension

When incorporating time constraints, different extensions of Petri nets were proposed. In general, when the time constraints are specified by constants(durations), the associated extension is said Timed Petri nets. This consider constant durations attached to places (P-Timed Petri nets) or transitions (T-Timed Petri nets). When these constraints are specified by intervals (delays) specifying the minimum and the maximum amounts of time needed for transitions' firing, the associated extension is called Time Petri nets. These intervals are attached to places (P-Time Petri nets), transitions (T-Time Petri nets) or arcs (A-Time Petri nets) leading thus to different extensions with variant semantics.

Petri nets form a powerful formalism for the expression of control flow in business processes [2,19,18,16]. In addition, several studies [1,6,27,22] have shown the importance of temporal reasoning in the specification of workflow systems.

In [27], the authors extend the simple WF-net presented by van der Aalst [3] by associating with each transition an amount of time representing the duration of the task it models. They propose a temporal extension of the WF-net, called Time WF-net and scored TWF-net. Timing discipline adopted in the proposed model announced that each enabled transition must start running immediately, otherwise it will be disabled, and once started, this transition can not be delayed, i.e. its duration should be respected. While this approach is strict in the fixed duration required, time Workflow nets (TWF-nets) incorporate time constraints of activities by associating to each transition an interval specifying its firing time [12,15,27].

Since this time consideration is flexible and given that we are interested by modeling the composition of workflow processes with time constraints, we propose the time open workflow net model (ToWF-net). This model associates a static time interval to each transition of an open workflow net to express the execution time or delay of corresponding activity. The formal definition of a ToWF-net model net is the following:

Definition 1 ***(ToWF-net)***
A Time Open Workflow Net N is a tuple (P, T, F, FI, I, O) with:

- *P is a set of places,*
- *T is a set of transitions,*
- *I is a set of places representing input interfaces which are responsible for receiving messages from other services: ${}^\bullet I = \emptyset$.*
- *O is a set of places representing output interfaces that are responsible for sending messages to other services: $O^\bullet = \emptyset$.*
- *I, O and P are disjoint. I and O connect transitions of different partners.*
- *$F \subseteq ((P \cup I) \times T) \cup (T \times (P \cup O))$ is the flow relation,*
- *$FI : T \to \mathbb{Q}^+ \times \mathbb{Q}^+ \cup \{\infty\}$ is the function that associates with each transition $t \in T$ a static firing interval, i.e. $FI(t) = [minFI(t), maxFI(t)]$ where $minFI(t)$ and $maxFI(t)$ are rational numbers representing respectively the minimal and maximal firing time,*

The marking of N is a vector of $\mathbb{N}^P$ such that for each place $p \in P$, $M(p)$ is the number of tokens in p. The initial marking of N is M_i knowing that M_p is used to denote a marking for which $M(p) = 1$ and $M(q) = 0 \; \forall q \in (P \cup I \cup O) \backslash \{p\}$.

A transition t is said to be enabled in a marking M if the required tokens are present in the input places of t. We denote by $En(M)$ the set of all the transitions enabled in the marking M. A transition t is said disabled by the firing of t' in M if it is enabled in M but it isn't in $M -^\bullet t'$. When focusing of newly enabled transitions after firing a transition t from M and leading to M', we denote by $NEn(M, t)$ the set of transitions enabled after this firing.

$NEn(M, t) = \{t' \in En(M') | t' = t \vee \neg M \geq^\bullet t +^\bullet t'\}$.

When a transition t becomes enabled, its firing interval is set to its static firing interval $FI(t)$. The lower and upper bounds of $FI(t)$ decrease synchronously with time, until t is fired or disabled by another firing. t can fire, if the lower bound of its firing interval reaches 0, but when upper bound of its firing interval reaches 0, t must be fired without any additional delay (strong semantic). The firing takes no time but may lead to another marking.

Let us define first the state of a ToWF-net and then the transition relation.

Definition 2 *A state in a ToWF-net represents the state of the process modeled with ToWF-net after the occurrence of an event. Formally, a state in a ToWF-net is a pair (M, Int) where:*

- *M is a marking,*
- *Int is a firing interval function, $Int : En(M) \to \mathbb{Q}^+ \times \mathbb{Q}^+ \cup \{\infty\}$. We denote $Int(t) = [minInt(t), maxInt(t)]$.*

The initial state of a ToWF-net is (M_0, Int_0) where $M_0 = M_i$ (since in a ToWF-net, only i contains initially one token) and $Int_0(t) = FI(t) \; \forall t \in En(M_0)$

Starting from the initial state (M_0, Int_0), the net evolves following the occurrence of events. An event corresponds to either a transition firing or a time progression. Hence, the transition relation between a state $s_1 = (M_1, Int_1)$ and $s_2 = (M_2, Int_2)$ is defined by $\overset{t}{\rightarrow}$ in case of a firing and by $\overset{d}{\rightarrow}$ in case of time progression. The conditions and the computation of the resulting state after an event occurrence are defined as follows:

1. $s_1 \overset{t}{\rightarrow} s_2$ if and only if s_2 is immediately reachable from s_1 by firing the transition t, i.e.
   $t \in En(M_1)$ and $minInt_1(t) = 0$,
   $M_2 = M_1 -^{\bullet} t + t^{\bullet}$, and
   $\forall t' \in En(M_2)$, $Int_2(t') = \begin{cases} FI(t') \; if t' \in NEn(M_1, t) \\ Int_1(t') \qquad otherwise \end{cases}$
2. $s_1 \overset{d}{\rightarrow} s_2$ $\forall d \in \mathbb{R}$ if and only if the state s_2 is reachable from s_1 by time progression with d time units, i.e.
   $minInt_1(t) + d \leq maxFI(t)$,
   $M_2 = M_1$, and
   $\forall t \in En(M_1)$, $Int_2(t) = [Max(0, minInt_1(t) - d), maxInt_1(t) - d]$

Therefore, the semantics of a ToWF-net N is defined by a transition system $(S, s_0, \rightarrow)$ where S is the set of all the states reachable from the initial state s_0 by the transition relation $\rightarrow$ defined above.

2.3 A case study

In order to illustrate the proposed ToWF-net, we propose to study the process of awarding of pensions to handicapped persons. This process requires the collaboration of three organizations:

- The prefecture which manages scholarships and grant of license to the disabled.
- A medical entity that is responsible for negotiating the date of appointments with patients and collecting the medical informations.
- The Town Hall which establishes certificates, births extracts, etc.

The allocation of pension process is seen as a collaboration between the services offered by these organizations.

In fact, citizens with disabilities ask a government scholarship. To start the process, citizens request the form corresponding to the prefecture. Once the citizen receives the form, he fills it and sends it to the prefecture. The latter seeks medical entity to consider disability that the citizen presents. Medical entity subsequently contacts the citizen to negotiate with him about a date of appointment. Once an appointment is fixed, and after reviewing the citizen, the entity establishes a medical examination report and forwards it to the prefecture. Meanwhile, the prefecture asked the town hall to establish a certificate of residence of the citizen. Once the certificate of residence and the medical report is received, the prefecture makes the final decision.

The figure 1 shows a screenshot of this composition involving the four processes relevant to the Applicant, the Prefecture, the Town Hall and the Medical Unit.

These processes are interconnected with available interfaces that facilitate communication and exchange of messages between them. These interfaces correspond to places denoted by I_{sn} (for input interfaces) and O_{sn} (for output ones), where s is the service number and n is the interface number in each category. Note that each input interface place of a service has an equivalent output interface of another service and this will guarantee the services communication. For sake of clarity, in this example, the interfaces are given names which explain the sequence of exchanged messages between partners.

The various services are forced to respect the different temporal properties of each service, in what follows , we mention a few of them:

- Once the medical entity proposes dates for appointment to the citizen, it must receive the confirmation within 24 hours.
- Once the application for the grant is received, the prefecture sends its final decision to the citizen, after at least 49 hours and not more than 180 hours.
- The medical report may be sent to the prefecture after at least 24 hours and up to 48 hours of sending the medical examination.
- The receipt of the result of the request is within 210 hours after sending the request of the purse.
- Two hours is the maximum time to review a citizen in medical entity.
- The time of receipt of the certificate of residence and review of citizen ratio is up to three hours.
- Negotiation of the appointment date between the citizen and the medical entity runs for up to one hour.

We present in the following section the analysis of reachability of the Web services composition modeled by ToWF-nets and we expose the case study reachability analysis in the tool Romeo [21].

3 Reachability analysis of ToWF-nets

After the formal definition of ToWF-nets, we focus now on their reachability analysis. This analysis is based on the efficient construction of the state space.

By analogy with the marking graph defined in the context of an ordinary Petri net, we define a state space by a graph containing all accessible states of a ToWF-net from the initial state. Therefore, to calculate the state space of a ToWF-net, we must be able to calculate the reachable states by activating the enabled transitions.

Definition 3 *The state space of a ToWF-net has the following structure: $SS = (S, \rightarrow, s_0)$; where S is the set of nodes in form (M, Int) representing the reachable states from the initial one $s_0 = (M_i, Int_0)$; $\rightarrow$ represents the transition relation which defines the evolution from one state to another.*

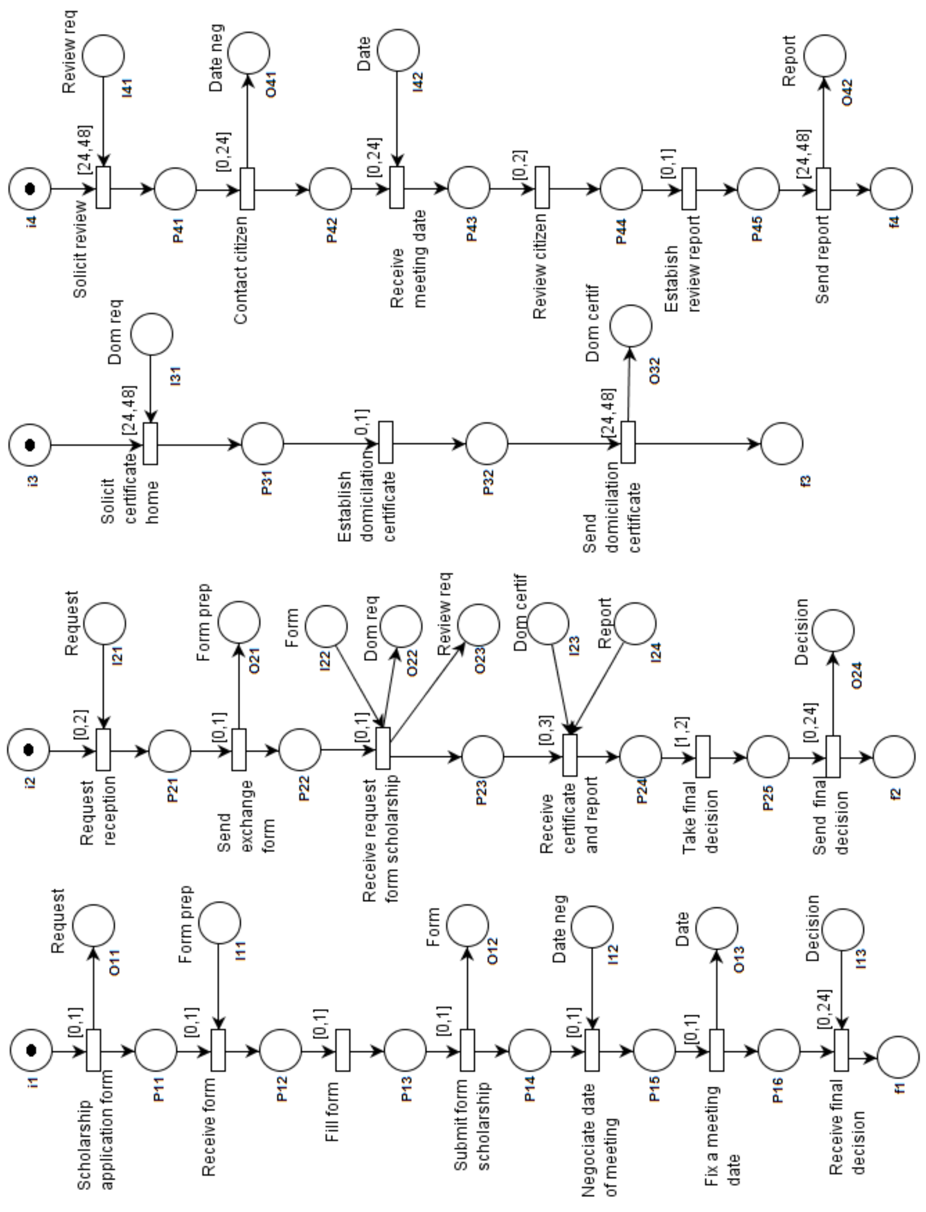


Fig. 1. Modeling of the case study with ToWF-nets

$S = \{s | s_0 \overset{*}{\rightarrow} s\}$ is the set of reachable states of the model, and $\overset{*}{\rightarrow}$ is the reflexive and transitive closure of $\rightarrow$.

The reachability analysis [12] in timed models (such as time extensions of Petri nets as well as timed automata) is based in general on abstraction, which preserves reachability properties. Such an abstraction for timed models, consists in considering only one node for all states reachable from the same firing sequence while abstracting from their firing times. The grouped states, known as state classes, are then considered modulo some equivalence relation preserving properties of interest.

In return, the state class method is intended to provide a finite representation of the infinite state space of any bounded time Petri net.

Technical classes produce for a large class of time nets a finite representation of their behavior states, which allows a reachability analysis similar to that permitted for Petri nets by the technique of marking graph.

The state classes can be represented by a marking and a firing domain. Formally, a state class is a couple (M, D) where M is a marking and D is characterized by a set of atomic constraints over the firing delays of enabled transitions: $minFI(t) \leq t \leq maxFI(t) \quad \forall t \in En(M)$.

Note that the initial class coincides with the initial state of the network. This initial class is (M_0, D_0) where $M_0 = M_i$ and D_0 corresponds to the firing domains of transitions enabled in M_0.

All states within the same node share the same untimed information and the union of their time domains is represented by a set of atomic constraints handled efficiently by means of a Difference Bound Matrix (DBM) [32]. A DBM form a system of linear inequalities which constrain single variables $(v_1 ... v_n)$ and their differences within limits identified by coefficients b_{ij}. This is formally expressed as:

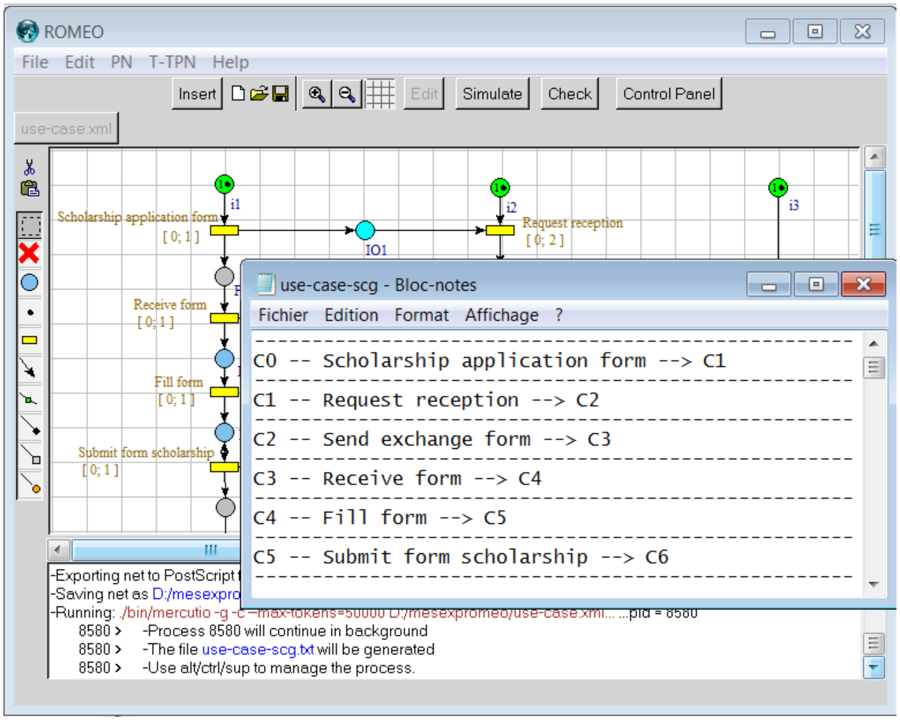
$$\begin{cases} v_i - v_j \leq b_{ij} \quad i, j \in [0..n], \; b_{ij} \in \mathbb{Q} \\ v_0 = 0 \end{cases}$$

In terms of behavior, this state classes group preserves highly the states traces, and thus the safety properties.

The computation of the state class graph is necessary at this point to perform the various reachability analysis. Among the abstractions proposed in the literature [9], [10,36], we consider here the state class graph method [9] for its advantage, over the others, which is the finiteness property for all bounded time Petri nets (while using some approximations).

Romeo [21] is a software studio dedicated to time Petri nets analysis. It is developed at IRCCyN by the Real-Time Systems Team. It performs analysis on T-Time Petri nets and on one of their extensions to scheduling. We chose Romeo because it performs, among other features, the computation of the State Class Graph (SCG) and a graphical simulation of a T-Time Petri net. It is also a model checker for a subclass of TCTL formulas.

Therefore, we used Romeo to generate the SCG of our case study. We begun with composing the different services by superposing the interface places which correspond to the same interface communication. We then simulated the overall obtained net in Romeo. Figure 2 presents a snapshot of the case study analysis conducted in Romeo and especially the beginning of the file generated by Romeo and which contains the SCG.



```
--------------------------------------------------------------
C0 -- Scholarship application form --> C1
--------------------------------------------------------------
C1 -- Request reception --> C2
--------------------------------------------------------------
C2 -- Send exchange form --> C3
--------------------------------------------------------------
C3 -- Receive form --> C4
--------------------------------------------------------------
C4 -- Fill form --> C5
--------------------------------------------------------------
C5 -- Submit form scholarship --> C6
--------------------------------------------------------------
```

```
-Exporting net to PostScript f
-Saving net as D:/mesexpro
-Running: ./bin/mercutio -g -c –max-tokens=50000 D:/mesexpromeo/use-case.xml... ...pid = 8580
     8580 >   -Process 8580 will continue in background
     8580 >   -The file use-case-scg.txt will be generated
     8580 >   -Use alt/ctrl/sup to manage the process.
```

Fig. 2. Analysis of the case study in Romeo

4 Compatibility analysis of ToWF-nets

The analysis of the state space is very significant to the extent that it can reveal important characteristics of the modeled system, about its structure and dynamic behavior. However, for a more accurate verification, we should not be limited to this type of checking rather another specific properties. Indeed, we focus in this section on the formal verification of compatibility properties of ToWF-nets. We propose to use model checking method to verify these properties since this method permits an exhaustive verification over all the possible executions. Given a concurrent system Σ and a temporal logic formula φ, the model checking problem is to decide whether Σ satisfies φ. Hence, we have to formulate in temporal logic the properties to be verified.

4.1 Model checking TPN-TCTL

Real systems often have behaviors that depend on time. The ability to manipulate and model the temporal dimension of the events that take place in the real world is fundamental in many applications. These applications may involve banking, medical, or multimedia applications. The variety of applications motivate many recent studies that aim to integrate all the features necessary to take into account the time during verification.

TCTL (Timed Computational Tree Logic) is a timed extension of the temporal logic CTL. TCTL added to CTL a quantitative information on the delays between actions. It is built from atomic propositions, logical connectors and temporal operators (U, F, G, X, etc.). The TCTL temporal logic can be used to check the properties of a time Petri net.

The syntax of TCTL formulas is inductively defined by:

$\varphi ::= \mathit{false} \mid \neg\varphi \mid \varphi \wedge \varphi \mid A(\varphi\ U_I\ \varphi) \mid E(\varphi\ U_I\ \varphi)$

where p denotes a proposition, φ denotes a formula and $I = [a, b]$ or $[a, \infty[$ with $a \in \mathbb{N}$ and $b \in \mathbb{N}$.

A and E are temporal quantifiers over the set of executions. $A\varphi$ announces that all the executions from the current state satisfy the property φ. $E\varphi$ states that from the current state, there exists an execution which satisfies φ. Finally $\varphi\ U_I\psi$ means that the property φ is true until ψ is true, and ψ will be true in the time interval I.

We can use other compositional temporal operators [5]: EF$_I$ φ = E(true U$_I$ φ) (Possibility), EG$_I$ φ = $\neg$ AF $_I$ $\neg\varphi$ (All locations along an execution), AF$_I$ φ = A(true U$_I$ φ) (Locations along all executions), AG$_I$ φ = $\neg$ EF $_I$ $\neg\varphi$ (All locations along all executions).

Semantically, TCTL formulas are interpreted on states and execution paths of a model $M = (S, V)$ where S is a transition system and V is a valuation function that associates with each state the set of atomic propositions it satisfies. [26]

To interpret a TCTL formula on an execution path, we introduce the notion of dense execution path. Let $s \in S$ be a state of S, $\pi(s)$ the set of all execution paths starting from s, and $\rho = s_0 \overset{d_0 t_0}{\to} s_1 \overset{d_1 t_1}{\to} s_2 ...$ an execution path of s. The dense path of the execution path ρ is the mapping $\hat{\rho} : R^+ \to S$ defined by: $\hat{\rho}(r) = s^i + \delta$ such that $r = \sum_{j=0}^{i-1} d_j + \delta$, $i \geq 0$ and $0 \leq \delta \leq d_i$.

The formal semantics of TCTL is given by the satisfaction relation defined as follows:

- $M,\ s \nvDash \mathit{false}$,
- $M,\ s \vDash \phi$ iff $\phi \in V(s)$,
- $M,\ s \vDash \neg\varphi$ iff $M,\ s \nvDash \varphi$,
- $M,\ s \vDash \varphi \wedge \psi$ iff $M,\ s \vDash \varphi$ and $M,\ s \vDash \psi$,
- $M,\ s \vDash \forall(\varphi \cup_I \psi)$ iff $\forall \rho \in \pi(s)\ \exists r \in I,\ M,\ \hat{\rho}(r) \vDash \psi$ and $\forall 0 \leq r' \leq r\ M,\ \hat{\rho}(r') \vDash \varphi$,
- $M,\ s \vDash \exists(\varphi \cup_I \psi)$ iff $\exists \rho \in \pi(s)\ \exists r \in I,\ M,\ \hat{\rho}(r) \vDash \psi$ and $\forall 0 \leq r' \leq r\ M,\ \hat{\rho}(r') \vDash \varphi$,

When interval I is omitted, its value is by default $[0, \infty[$.

The Time Petri net model is said to satisfy a TCTL formula φ iff $M,\ s_0 \models \varphi$.

The logic TCTL allows writing temporal properties with a quantification of the time. We chose this approach because it is decidable and PSPACE-complete for bounded Petri nets [14].

The authors of [24] have gone further by defining a sub-class of TCTL for time Petri nets in dense time, called TPN-TCTL. They proved the decidability of model-checking of TPN-TCTL on Petri nets and showed that its complexity is PSPACE.

Definition 4 *The temporal logic TPN-TCTL is defined inductively by:*
TPN-TCTL $::= \mathit{false} \mid \varphi \mid \neg\varphi \mid \varphi \vee \psi \mid \varphi \wedge \psi \mid \varphi \Rightarrow \psi \mid E\varphi U_I \psi \mid A\varphi U_I \psi$
$\mid EG_I\varphi \mid AG_I\varphi \mid AF_I\varphi \mid EF_I\varphi \mid AG(\phi_1 \Rightarrow AF_{[0,d]}\phi_2)$.

Where φ and $\psi \in$ TPN-TCTL,
$I = [a, b]$ or $[a, b[$ with $a \in \mathbb{N}$ and $b \in \mathbb{N} \cup \{\infty\}$.
ϕ_1 and ϕ_2 are propositions on markings.
$\forall G(\phi_1 \Rightarrow \forall F_{[0,d]}\phi_2)$ means that from the current state, any occurrence of marking ϕ_1 is followed by an occurrence of marking ϕ_2 less of d units of time later.

Romeo permits a practical implementation of the verification of properties described in TPN-TCTL. It is therefore possible to model check on the fly temporal quantitative properties. That's why we investigate in the following section the TCTL expression of the compatibility property and hence its verification in Romeo.

Before this, let us recall the notation used by Romeo to implement a TPN-TCTL property:
$\mathit{TPN\text{-}TCTL}_{Romeo} = E(p)U[a, b](q) \mid A(p)U[a, b](q) \mid EF[a, b](p) \mid AF[a, b](p)$
$\mid EG[a, b](p) \mid AG[a, b](p) \mid EF[a, b](p) \mid (p) \rightarrow [0, b](q)$.

where p, q: GMEC; U: until; E: exists; A: forall; F: eventually; G: always; $\rightarrow$: response; a: integer; b integer or inf (to denote ∞).

$GMEC = a{*}M(i)\{+, -\}b{*}M(j)\{<, <=, >, >=, =\}k \mid \mathit{deadlock} \mid \mathit{bounded}(k)$
$\mid p\ \mathit{and}\ q \mid p\ \mathit{or}\ q \mid p \Rightarrow q \mid \mathit{not}\ p$.

M: keyword (marking); *deadlock*, *bounded*: keywords; i, j:place indexes; a, b, k :integers ; $*, +, -, \mathit{and}, \mathit{or}, \Rightarrow, \mathit{not}$: usual operators ; p, q: GMEC

The syntax $(p) \rightarrow [0, b](q)$ denotes a leads to property meaning $AG((p)$ imply $AE[0, b](q))$. E.g. $(p) \rightarrow [0, b](q)$ holds if and only if whenever p holds eventually q will hold as well in $[0, b]$ time units.

4.2 TCTL characterization of the compatibility property

From a behavioral point of view, two (or more) processes are said to be compatible if they can interact correctly: this means that they can exchange the same type of messages and the composite system does not suffer from the deadlock problem. This leads us to distinguish between a syntactic compatibility which concerns the verification of the interfaces conformance and a semantic compatibility which is related to check the absence of deadlocks. We investigate in this paper the analysis of the semantic compatibility.

But before this, let us define the composite system obtained from the superposition of a number of syntactically compatible ToWF-nets. The composed system N of nbX ToWF-nets $N_1 ... N_{nbX}$ consists of all ToWF-nets which share interface places, i.e. every place of N which is an input interface of a WF-net is also an output place of another WF-net in the composition. Trivially, N can be seen as a time Petri net with nbX input places and nbX output places. The initial marking of N is $M_0 = \sum_{s=1}^{nbX} i_s$.

According to [11,20,28], the compatibility is closely related to the absence of deadlock in the composite system. They considered that two oWF-nets are compatible if they can reach their final states. In addition to this condition, we characterize the compatibility in ToWF-nets by the timing constraints respect.

In this direction, we define three classes of compatibility:

- Partial compatibility: A composed system N is partially compatible if it is deadlock-free.
- Total Compatibility: A composed system N is compatible if N is already partially compatible and furthermore, it guarantees the proper termination.
- Perfect compatibility: A composed system is perfectly compatible if it verifies the total compatibility as well as the deadline constraints.

We focus here on formulating the three types of compatibility properties: partial compatibility, total compatibility and perfect compatibility. Let us consider the following:

- nbX: is the number of processes;
- nbp: is the number of places in a given process;
- nbi: is the number of interface places available in a composition;
- i_s: is the input place of the process number s.
- f_s: is the output place of the process number s.

– **Partial compatibility**

To assure its partial compatibility, we have to check the absence of deadlock in a composition. The process is deadlock-free if there is a transition allowed for any marking except the final marking M_{fin} in which all the final places f_s ($s = 1..nbX$) are marked. This property is expressed as follows:

$$\forall M \in [M_0\rangle,\ M_{fin} \in [M\rangle$$

In TCTL, the deadlock-freeness can be expressed as "for all the executions from the initial state, no deadlock will be encountered until the final state is reached". For the final state, it suffices to check if the final places are marked. Hence, the expression of the partial compatibility in TCTL is given as follows:

$$AG_{[0,\infty[}((not\ MF) \quad \Rightarrow \quad not\ deadlock)$$

where *deadlock* is a proposition which returns true iff there is no enabled transition from the current state; and MF is a proposition on the marking M_{fin} in which each final place contains at least one token.

$$MF = \overset{nbX}{\underset{s=1}{\wedge}} M(f_s) >= 1$$

Here we focus only on the arrival of tokens to final places and we don't care if the other places contains tokens or not.

– **Total compatibility**

Having expressed the partial compatibility, we focus here on the expression of the property of proper termination in TCTL. This property allows the process to complete its execution in any case, but at the time of termination, all places of ToWF-nets must be empty except for the final places which must have one token. Verifying the proper termination consists in checking the existence of a marking M for which all places are empty except the output ones. The expression of this property is given as follows:

$$\forall M \in [M_0\rangle : M(f_s) \geq 1 \ \forall s \in \{1, .., nbX\} \quad \Rightarrow \quad M = \textstyle\sum_{s=1}^{nbX} f_s$$

In TCTL, this property (proper termination) is formulated as follows:

$$AF_{[0,\infty[}\ StrictMF$$

Where $StrictMF$ is a proposition on the marking ensuring exactly one token in each final place f_s and no tokens in all the other places including the interface places.

$$StrictMF = \overset{nbX}{\underset{s=1}{\wedge}} \ (\overset{nbip}{\underset{p=1}{\wedge}}(M(p) = 0) \ \ \wedge \ \ (M(f_s) = 1)) \quad \wedge \quad (\overset{nbi}{\underset{i=1}{\wedge}} M(I_i) = 0)$$

In this definition, we used nbip to denote the number of places except the final place for a process.

– **Perfect compatibility**

Here, we have to check the deadlock-freeness and the proper termination taking into account the overall deadline constraint.

Let us consider that a process has to reach his final state in Tm time units. The proper termination within this delay is expressed as follows:

$$AF_{[0,Tm]}\ StrictMF$$

Hence, the perfect compatibility of a composition of ToWF-nets is ensured iff:

– $AG_{[0,Tm]}((not\ MF) \ \ \Rightarrow \ \ not\ deadlock)$
– $AF_{[0,Tm]}\ StrictMF$

4.3 On the fly model checking of ToWF-nets composition

We report in this section some results related to the verification of compatibility and soundness properties of the composition of ToWF-nets. This verification is ensured by Romeo since it implements an on the fly model checking algorithm of TPN-TCTL properties.

Let us study the simple composition of ToWF-nets of figure 3. One can easily see that no deadlock will be encountered until the final places will be marked. Hence the partial compatibility is satisfied as proven in figure 4. Nevertheless, the execution of transitions T_4 and T_5 of the second process leads to two tokens in the place f_2; which leads to violate the property of total compatibility. Figure 5 shows the negative result for this property and draws a trace.

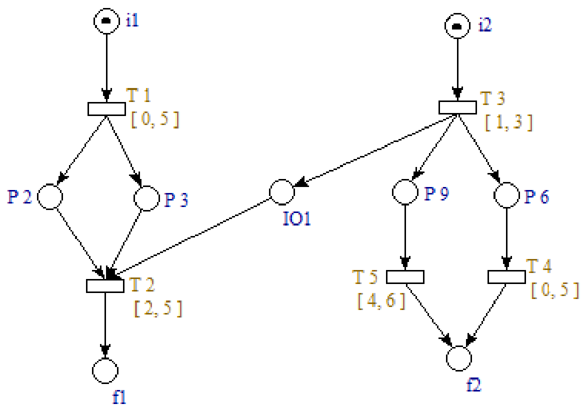


Fig. 3. A composition of ToWF-nets satisfying the partial compatibility but not the total compatibility

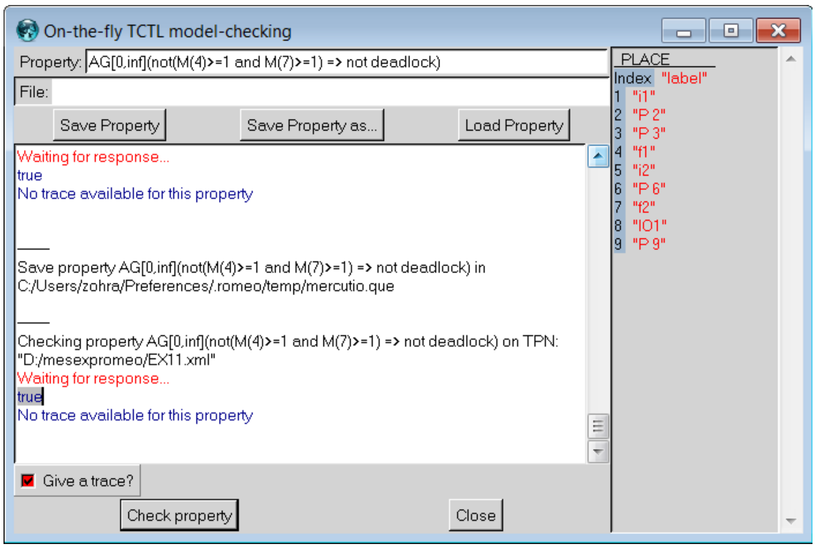
Let us now return to the case study given in figure 1. In order to check the partial compatibility of the involved processes, we formulate the correspondant TCTL formula as follows :

$$AG[0, inf]((not\ (M(30) >= 1 \text{ and } M(21) >= 1 \text{ and } M(23) >= 1 \text{ and } M(28) >= 1)) \quad \Rightarrow \quad not\ deadlock)$$

Where 30, 21, 23 and 28 are the indexes associated by Romeo to respectively the places f_1, f_4, f_3 and f_2.

Figure 6 draws a snapshot of a the verification of the partial compatibility of the four processes involved in the composition. As we can see in the figure, the result is *true* and hence the partial compatibility is ensured. The total compatibility characterized by the partial compatibility and the following formula is also verified for this example:

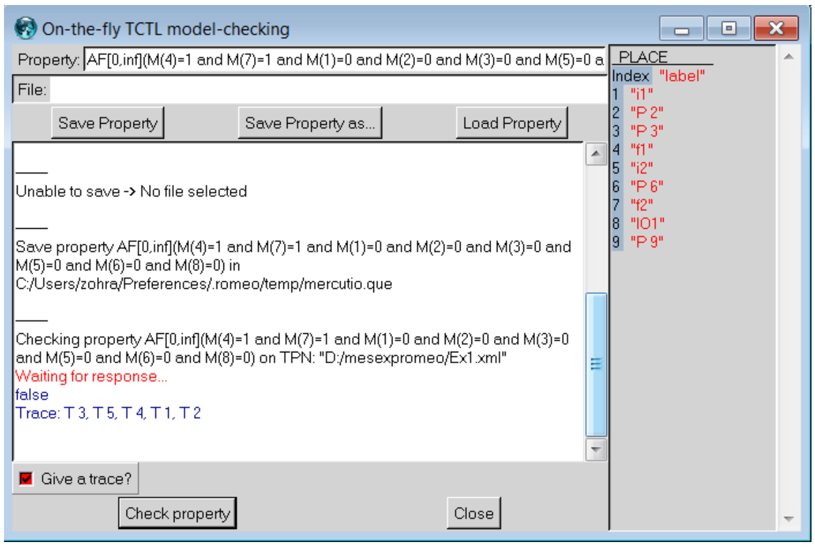
$$AF[0, inf](M(30) = 1 \text{ and } M(21) = 1 \text{ and } M(23) = 1 \text{ and } M(28) = 1 \text{ and } M(1) = 0 \text{ and } M(1) = 0 \text{ and } M(2) = 0 \text{ and } M(3) = 0 \text{ and .. and } M(36) = 0)$$



```
Property: AG[0,inf](not(M(4)>=1 and M(7)>=1) => not deadlock)
File:

Waiting for response...
true
No trace available for this property

____
Save property AG[0,inf](not(M(4)>=1 and M(7)>=1) => not deadlock) in
C:/Users/zohra/Preferences/.romeo/temp/mercutio.que

____
Checking property AG[0,inf](not(M(4)>=1 and M(7)>=1) => not deadlock) on TPN:
"D:/mesexpromeo/EX11.xml"
Waiting for response...
true
No trace available for this property
```

PLACE	
Index	"label"
1	"i1"
2	"P 2"
3	"P 3"
4	"f1"
5	"i2"
6	"P 6"
7	"f2"
8	"IO1"
9	"P 9"

Fig. 4. Test of partial compatibility

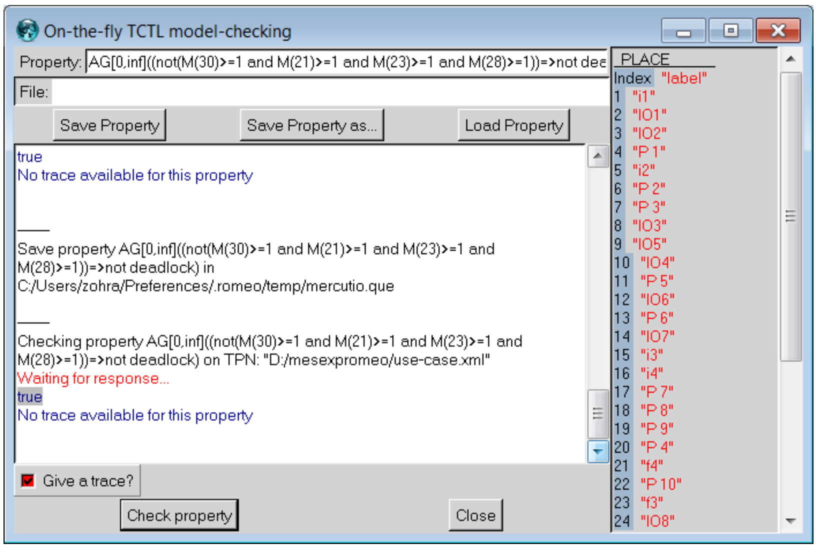


```
Property: AF[0,inf](M(4)=1 and M(7)=1 and M(1)=0 and M(2)=0 and M(3)=0 and M(5)=0 a
File:

____
Unable to save -> No file selected

____
Save property AF[0,inf](M(4)=1 and M(7)=1 and M(1)=0 and M(2)=0 and M(3)=0 and
M(5)=0 and M(6)=0 and M(8)=0) in
C:/Users/zohra/Preferences/.romeo/temp/mercutio.que

____
Checking property AF[0,inf](M(4)=1 and M(7)=1 and M(1)=0 and M(2)=0 and M(3)=0
and M(5)=0 and M(6)=0 and M(8)=0) on TPN: "D:/mesexpromeo/Ex1.xml"
Waiting for response...
false
Trace: T 3, T 5, T 4, T 1, T 2
```

PLACE	
Index	"label"
1	"i1"
2	"P 2"
3	"P 3"
4	"f1"
5	"i2"
6	"P 6"
7	"f2"
8	"IO1"
9	"P 9"

Fig. 5. Test of total compatibility

The perfect compatibility ensuring a proper termination with deadlock freeness within 210 hours is also verified for the example. However if we suggest a perfect compatibility in less than 210 hours, the result is "false".



Property: AG[0,inf]((not(M(30)>=1 and M(21)>=1 and M(23)>=1 and M(28)>=1))=>not dea

File:

```
true
No trace available for this property


____
Save property AG[0,inf]((not(M(30)>=1 and M(21)>=1 and M(23)>=1 and
M(28)>=1))=>not deadlock) in
C:/Users/zohra/Preferences/.romeo/temp/mercutio.que


____
Checking property AG[0,inf]((not(M(30)>=1 and M(21)>=1 and M(23)>=1 and
M(28)>=1))=>not deadlock) on TPN: "D:/mesexpromeo/use-case.xml"
Waiting for response...
true
No trace available for this property
```

PLACE

Index	"label"
1	"i1"
2	"IO1"
3	"IO2"
4	"P 1"
5	"i2"
6	"P 2"
7	"P 3"
8	"IO3"
9	"IO5"
10	"IO4"
11	"P 5"
12	"IO6"
13	"P 6"
14	"IO7"
15	"i3"
16	"i4"
17	"P 7"
18	"P 8"
19	"P 9"
20	"P 4"
21	"f4"
22	"P 10"
23	"f3"
24	"IO8"

Fig. 6. Model checking the partial compatibility of the case study with Romeo

5 Related work

Several works dealt with compatibility analysis of Web services modeled either by open workflow nets or other formalisms. Wil M. P. van der Aalst and al. [4] considered that two services are compatible if their interfaces are compatible and if in addition the composition does not suffer from a deadlock. They also formalized other concepts related to the compatibility as strategy and controllability.

Lucas Bordeaux and al. [11] studied the verification of compatibility of Web services assuming that the messages exchanged are semantically of the same type and have the same name. They based their work on labeled transition systems (LTS) for the modeling of Web services. Three types of compatibility have been defined: the opposite behavior, unspecified reception and absence of deadlock.

Marlon Dumas and al. [17] have classified the incompatibility of Web services into two types: 1) Incompatibility of signatures (it occurs when a service request an operation from another service which can't provide it) and 2) Protocol incompatibility which occurs when a service A engages in a series of interactions with a service B, but the order which undertakes the service A is not compatible with the service B. hence, they focused on the incompatibility of protocols in their article.

Wei Tan and al. [34] proposed an approach that checks interface compatibility of Web services described by BPEL, and corrects these services if they are not compatible. To do this, they modeled the composition by SWF-nets, a subclass of CPN (Colored Petri Nets). Then they checked the compatibility of interfaces.

These works dealt with non timed processes while we focus on those augmented by time information. Focusing on time constraints, Nawal Guermouche and al. [23] proposed an approach that allows the automatic verification of the compatibility taking into account their operations, the messages exchanged, the data associated with messages and time constraints. To check the compatibility of services using all of these properties, they proposed to extend the Web Services Timed Transition System (WSTTS), while we chose to extend oWF-nets with delays associated to activities. In addition, none of the approaches mentioned is based on the formal verification of compatibility while we have used this method in our approach. We mainly used the model checking formal method to check the compatibility classes of ToWF-nets, witch shows a counter example in case a property is violated allowing thus to recognize and correct the eventual errors as early as possible.

6 Conclusion

Open workflow nets form a sub class of Petri nets which has been widely and successfully used to model inter organizational business processes. In particular, they successfully form a solid theoretical basis for modeling and analysis of Web services composition. From a software engineering point of view, the construction of new services by composing existing ones raises a number of challenges. The most important is the challenge to guarantee a correct interaction of independent, communicating pieces of software. In deed, due to the message sending nature of service interaction, many delicate errors might take place when several services are put together (unreceived messages, deadlocks, contradictory behaviors, etc.). So far, it is necessary to ensure the proper functioning of each service involved in the composition as well as their ability to be composed, their good communication and the validity of their messages exchange.

In this context, we investigated in this paper the verification of open workflow nets compatibility as a main feature to ensure a correct composition and to prevent eventual errors from occurring. In addition, we extended the oWF-nets by timing constraints specifying the activities delays. For the proposed model baptised Time oWF-net, we studied its semantics in terms of states evolution. Then, we defined compatibility classes relative to ToWF-nets and emphasized a formal method of their verification based on TCTL model checking. We finally studied a case study in which four services interact with each other to reach a common goal which is the awarding of pensions of handicapped persons. We conducted a reachability analysis of this example in conformance with the method we propose and we model checked some of the proposed properties with the time Petri net analyser Romeo. We presented, in addition, a simple example with a violated property in order to show the generation of a counter example.

As a perspective, we propose to study the parametric verification of ToWF-nets. In deed, this supposes to treat ToWF-nets modeling concurrent instances and thus the consistency of time properties is of great interest.

References

1. van der Aalst, W.: Interval timed coloured petri nets and their analysis. In: Proceedings of the 14th International Conference on Application and Theory of Petri Nets, London, Springer-Verlag. pp. 453–472 (1993)
2. van der Aalst, W.: Three good reasons for using a petri-net-based workflow management system. In: International Working Conference on Information and Process Integration in Enterprises (IPIC96). pp. 179–201 (1996)
3. van der Aalst, W.: Verification of workflow nets. In: ICATPN 97, LNCS, 1248 (1997)
4. van der Aalst, W., Arjan, J., Christian, S., Wolf, K.: Service interaction: Patterns, formalization, and analysis. In: 9th International School on Formal Methods for the design of Computer, Communication and Software Systems (2009)
5. Alur, R., Courchoubetis, C., Dill, D.: Model checking in dense real time. Information and computation. 104, 2–34 (1993)
6. Atluri, V., Huang, W.: An authorization model for workflows. In: Proceedings of the 4th European Symposium on Research in Computer Security, London, Springer-Verlag. pp. 44–64 (1996)
7. Barkaoui, K., Ben Ayed, R.: Uniform verification of workflow soundness. Transactions of the Institute of Measurement and Control Journal. 31, 1–16 (2010)
8. Barkaoui, K., Ben Ayed, R., Sbaï, Z.: Workflow soundness verification based on structure theory of petri nets. International Journal of Computing and Information Sciences (IJCIS). 5(1), 51–61 (2007)
9. Berthomieu, B., Diaz, M.: Modeling and verification of time dependent systems using time petri nets. IEEE Transactions on Software Engineering. 17(3) (1991)
10. Berthomieu, B., Vernadat, F.: State class constructions for branching analysis of time petri nets. In: TACAS 2003, volume 2619 of Lecture Notes in Computer Science. pp. 442–457 (2003)
11. Bordeaux, L., Salaun, G., Berardi, D., Mecella, M.: When are two web services compatible ? Sapienza University. 3324 (2005)
12. Boucheneb, H., Barkaoui, K.: Parametric verification of time workflow nets. In: 24th International Conference on Software Engineering (SEKE). pp. 375–380 (2012)
13. Boucheneb, H., Barkaoui, K.: Reducing interleaving semantics redundancy in reachability analysis of time petri nets. ACM Transactions in Embedded Computing Systems (TECS). 12(1), 1–24 (2013)
14. Boucheneb, H., Gardey, G., Roux, O.: Tctl model-checking of time petri nets. Journal of Logic and Computation. 19(6), 1509–1540 (2009)
15. Camerzan, I.: On soundness for time workflow nets. Computer Science Journal of Moldova. 15(1), 74–87 (2007)
16. De Michelis, G., Ellis, C., Memmi, G.: In: Proceedings of the second Workshop on Computer-Supported Cooperative Work, Petri nets and related formalisms, Zaragoza, Spain (1994)

17. Dumas, M., Benatallah, B., Motahari Nezhad, H.: Web service protocols : Compatibility and adaptation. Institute of Electrical and Electronics Engineers. (2008)
18. Ellis, C., Keddara, K., Rozenberg, G.: Dynamic change within workflow systems. In: Proceedings of conference on Organizational computing systems. pp. 10–21 (1995)
19. Esparza, J., Silva, M.: Circuits, handles, bridges and nets. In: Applications and Theory of Petri Nets. Lecture Notes in Computer Science, vol. 483, pp. 210–242. Springer (1989)
20. Foster, H., Uchitel, S., Magee, J., Kramer, J.: Compatibility verification for web service choreography. In: Proceedings of IEEE International Conference on Web Services. pp. 738–741 (2004)
21. Gardey, G., Lime, D., Magnin, M., Roux, O.: Romeo: A tool for time petri nets analysis. In: Proceeding of 17th International Conference on Computer Aided Verfication (CAV'05), volume 3576 of Lecture Notes in Computer Science. pp. 418–423 (2005)
22. Gou, H., Huang, B., Liu, W., Li, Y., Ren, S.: Modeling distributed business processes of virtual enterprises based on the object-oriented approach and petri nets. Systems Man and Cybernetics. 3 (2001)
23. Guermouche, N., Perrin, O., Ringeissen, C.: Timed specification for web services compatibility analysis. Theoretical Computer Science. (2008)
24. Hadjidj, R., Boucheneb, H.: On-the-fly tctl model-checking for time petri nets. Theoretical Computer Science. 410(42), 4241–4261 (2009)
25. Karsten, S.: Controllability of open workflow nets. In: EMISA. LNI, Bonner Köllen Verlag. pp. 236–249 (2005)
26. Konur, S., Fisher, M., Schewe, S.: Combined model checking for temporal, probabilistic, and real-time logics. Theoretical Computer Science. 503, 61–88 (2013)
27. Ling, S., Schmidt, H.: Time petri nets for workflow modelling and analysis. In: IEEE International Conference on Systems, Man, and Cybernetics. pp. 3039–3044 (2000)
28. Martens, A.: On compatibility of web services. In: Petri Net Newsletter. pp. 12–20 (2003)
29. Martens, A.: Analyzing web service based business processes. In: Proceeding of International Conference on Fundamental Approaches to Software Engineering, Part of the European Joint Conferences on Theory and Practice of Software, Lecture Notes in Computer Science vol. 3442, Springer-Verlag, (2005)
30. Massuthe, P., Reisig, W., Schmidt, K.: An operating guideline approach to the soa. Annals of Mathematics, Computing and Teleinformatics 1(3), 35–43 (2005)
31. Merlin, P.M.: A study of the recoverability of computing systems. University of California (1974)
32. Ridi, L., Torrini, J., Vicario, E.: Developing a scheduler with difference-bound matrices and the floyd-warshall algorithm. IEEE SOFTWARE (2012)
33. Sbaï, Z., Barkaoui, K.: Vérification formelle des processus workflow - extension aux workflows inter-organisationnels. Revue Ingénierie des Systèmes d'Information: Ingénierie des systèmes collaboratifs. 18(5), 33–57 (2013)
34. Tan, W., Fan, Y., Zhou, M.: A petri net-based method for compatibility analysis and composition of web services in business process execution language. IEEE T. Automation Science and Engineering. 6(1), 94–106 (2009)
35. WFMC: Workflow management coalition terminology and glossary (wfmc-tc-1011). Tech. Rep., Workflow Management Coalition, Brussals. (1999)
36. Yoneda, T., Ryuba, H.: Ctl model checking of time petri nets using geometric regions. IEICE Transactions on Information and Systems. pp. 297–396 (1998)