

A Time Workflow Model with Temporal Logic Constraints and Its Verification

Qiumei Yang

*Department of Computer Science and Engineering, College of Informatics,
South China Agricultural University, Guangzhou 510642, China
E-mail: yqmbegonia@163.com*

Yang Yu, Changsen Li

*Department of Computer Science,
Sun Yat-sen University,
Guangzhou 510275, China*

The traditional workflow model emphasizes on the control logic between activities. It cannot describe the schedule between workflow objects conveniently, especially the firing schedule between the distant activities. In this paper, a new workflow model TTL-TWF based on Time Petri net and Time Temporal Logic is proposed. Since temporal Logic is specialized in describing the properties and constraints of a system, and Petri net is proper to specify the behavioral structures of the system explicitly, the combination of them is a better approach to describe, analyze and verify the workflow process.

Keywords: Workflow; process model; temporal logic; time temporal logic.

1. Introduction

The purpose of workflow management system is to support the definition, execution and control of workflow processes. A workflow process defines a series of activities and the order of their execution so as to achieve a given goal. Workflow process modeling is a hot problem currently. Vander Aalst proposed Workflow-net [1], WF-net for short, which is based on Petri net. In WF-net, activities are represented as transitions, while its enabled conditions are denoted by places. This kind of modeling has been widely accepted since Petri net has a well-knit theory background. Petri net is proper to describe the operation structure of the system. Whereas, a workflow process model, even with the correct control logic, may contain conflicting temporal constraint. Some work has been done to solve this problem [2]. Temporal

logic has been introduced to be one of the useful tools in workflow modeling [3]. This paper presents a novel workflow model based on Time Petri net and Time Temporal Logic.

The remainder of this paper is organized as follows. Section 2 introduces the related work and technology, including workflow model and temporal logic. Section 3 proposes a new workflow model, i.e., Time Workflow Model with Temporal Logic Constraints (TTL-TWF). The verification of the new model is presented in section 4. At last, section 5 concludes the whole paper and gives view to the future work.

2. Basic theory and technology

2.1. WF-net and Time WF-net

WF-net [1] is used to describe the control-flow dimension of workflow and only can be used to specify the dynamic behavior of a single case in isolation.

A Petri net $PN = (P; T; F)$ is a WF-net (Workflow net) if and only if:

- PN has a source place i , $\bullet i = null$;
- PN has a sink place o , $o \bullet = null$;
- If a transition t^* which connects place o with i is added to PN (i.e. $\bullet t^* = o$ and $t^* \bullet = i$), then the resulting Petri net is strongly connected.

A TWF-net [4] (Time WF-net) is a WF-net in which a time interval that denotes the earliest and latest firing time is added to every transition.

A TWF-net is a tuple (P, T, F, FI) : such that: (P, T, F) is a WF-net and FI associates each transition $t \in T$ with a static firing interval: $FI : T \rightarrow INT$. The interval, $FI(t) = [mint, maxt]$, is a pair of real numbers referred to as the minimum static firing time and the maximum static firing time respectively.

2.2. Temporal Logic

Temporal Logic is the evolvement of Traditional Logic, to which temporal operator is added to express the temporal properties such as ‘eventually’, ‘until’ and ‘always’.

The goal of temporal logic is to investigate languages and “logical instruments” for propositions whose value depend on time and their temporal relationships. Z. Manna and A. Pnuelli developed a temporal logic system called PLTL [5], which is short for Proposition Linear Temporal logic. Logic connectors in PLTL are “negative” \neg , “conjunction” \wedge ,

“disjunction” \vee , “imply” \longrightarrow , while temporal operators includes: “always” \square , “eventually” \diamond , “next” \circ , “until” U . Its well-formed formulas are defined as follows:

Let AP be a set of atomic propositions. Then:

1. For all $p \in AP$, p is a formula.
2. If Φ is a formula, then $\neg\Phi, \square\Phi, \diamond\Phi, \circ\Phi$ are formulas.
3. If Φ and Ψ are formulas, then $\Phi \wedge \Psi, \Phi \vee \Psi, \Phi \longrightarrow \Psi, \Phi U \Psi$ are formulas.
4. All the formulas are formed by the rules mentioned above.

PLTL is interpreted over models that abstract away from the actual times at which events occur, retaining only temporal ordering information about the states of a system. Some attempts have been made to introduce time explicitly in PLTL. Time Proposition Temporal Logic [6], TPTL for short, is one of the most popular temporal logics with time.

Freeze quantification was proposed, in which every variable was bound to the time of a particular state. Freeze quantification identifies, precisely the subclass of “intended” specifications, and it leads to a concise and readable notation. For instance, the typical time-bounded response requirement that event P is always followed by event Q within 9 time units, can be asserted by the formula:

$$\square x.(P \longrightarrow \diamond y.(Q \wedge (y < x + 9)))$$

The above formula means whenever there is an event P , and the variable x is frozen to the current time, it is followed by an event Q , at time y , such that y is at most $x + 9$.

3. A Time Workflow Model with Temporal Logic Constraints

When WF-net is used in workflow modeling, some of the constraints can't be expressed clearly. For instance, A and E are two independent activities in WF-net (see Fig. 1). It is required that if A fires then E will fire too and the time interval between the fire of A and E must not exceed 8 time units.

Therefore, a new modeling method to solve such problems must be considered.

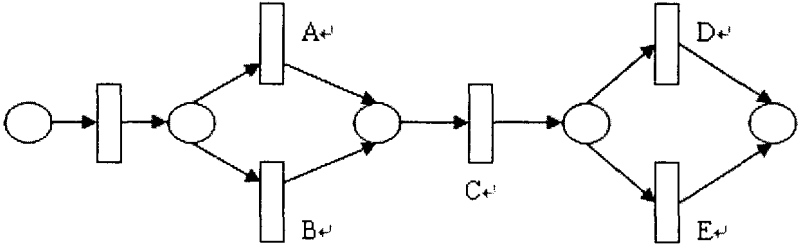


Fig. 1. A WF-net.

3.1. A Process Modeling Language Based on Petri Net and Temporal Logic

Process modeling means the computer presentation of business process. At present, there are three kinds of process modeling language [7].

- Operational modeling language: It supports the description of a system by means of an abstract model that simulates its behavior. Finite state machines and Petri nets are two of the important representatives of this species.
- Descriptive modeling language: These are mathematical notations suitable for describing system properties and constraints. Generally, they are used as requirement specification languages, and they are based mainly on algebra or mathematical logic.
- Dual-language approach: It is a fusion of the operational formalisms and descriptive formalisms. Dual-language approach not only can simulate system behavior but also can describe system properties and constraints. It is a language suitable for the verification of process model.

Petri net is appropriate to specify the behavioral structures of the system explicitly, while Temporal Logic is specialized in describing the properties and constraints of the system. Since one can complement the other, using a combination of Petri net and Temporal Logic is a highly promising approach to describe, analyze and verify the workflow systems.

The comparison between Petri net and temporal logic [8] is described in table 1.

There are several ways to combine a Petri net with temporal logic [8]. Since this method is applied to workflow process modeling, we only consider

Table 1. The comparison of Petri net and temporal logic.

Petri net	Temporal logic
Operational	Declarative
Suitable for structure description	Suitable for constraint description
Visual and textual representation	Textual representation

the situation that matches. The fusion of Petri net and temporal logic used in workflow process modeling can be seen in table 2.

Table 2. The combination of Petri net and temporal logic.

Temporal proposition formula	Petri net object
$mk(p)$	Place p has one token
$en(t)$	Transition t is enabled
$fi(t)$	Transition t fires

3.2. Temporal Logic Constraint Time Workflow

Because the workflow model existing is weak in describing the temporal properties and constraints among activities, we propose a new workflow model based on Time WF-net and Time Propositional Temporal Logic.

Definition 3.1. A Time Workflow Model with Temporal Logic Constraints ($TTL - TWF$) is a binary group (TWF, TS) , in which $TWF = (P, T, F, FI)$ denotes a Time WF-net. $TTL - TWF$ must satisfy:

- a) TWF has a source place i , $\bullet i = null$;
- b) TWF has a sink place o , $o \bullet = null$;
- c) If a transition t^* which connects place o with i is added to TWF (i.e. $\bullet t^* = o$ and $t^* \bullet = i$), then the resulting Petri net is strongly connected.
- d) $FI : T \rightarrow INT, FI(t) = [mint, maxt]$, is a real interval, is a pair of real numbers referred to as the minimum static firing time and the maximum static firing time respectively.
- e) TS is a TPPTL formula set, which is composed of a series of TPPTL formulas. Each TPPTL formula is connected by logic connector or temporal operator.

Atomic Proposition: $true, false, fi(t)(t\text{ fires}), mk(p)(p \text{ contains one token})$;

Frozen Quantification: *e.g.* x, y ; every atomic proposition can match at most one frozen quantification;

Connector: $\longrightarrow, \neg, \wedge, \vee, >, <, \geq, \leq, =$;

Temporal Operator: \square (always), \diamond (eventually), U (until).

The well-formed formulas of TPTL are defined as follows:

Let AP be a set of atomic propositions. Then:

1. For all $p \in AP$, p is a formula;
2. If Φ is a formula, then $\neg\Phi, \square\Phi, \diamond\Phi, \circ\Phi$ are formulas;
3. If Φ and Ψ are formulas, then $\Phi \wedge \Psi, \Phi \vee \Psi, \Phi \longrightarrow \Psi, \Phi U \Psi, x \geq y, x \leq y, x > y, x < y, x = y$ are formulas;
4. All the formulas are formed by the rules mentioned above.

3.3. The Design of TTL-TWF

There are three steps for designing a TTL-TWF:

- Step1: First analyze system's business procedure and use Petri net to build a WF-net, which describes the control logic of the whole procedure;
- Step2: Add a firing time interval to each of the transitions on the WF-net built in step 1, then forms Time WF-net.
- Step3: Consider the temporal association among activities carefully and then append TPTL formulas to TS.

A simple example is given in order to make the design of TTL-TWF clearer. It is a procedure for the purchase management. The customer will order goods and pay. Manufacturer sends the goods after receiving the order form.

- a) First modeling by Petri net, show as Fig. 2.

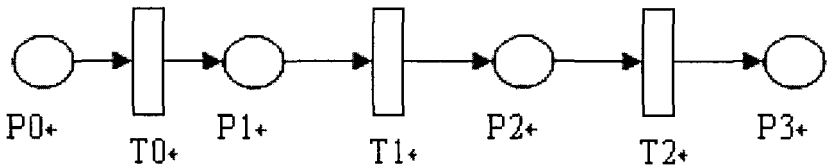


Fig. 2. Purchase procedure.

The meanings of the transition in Fig. 2 are introduced in table 3.

Table 3. The meanings of transitions.

Transition	T0	T1	T2
Meanings	Order goods and pay	Receive order form	Send goods

b) Add time intervals to every transition. See Fig. 3.

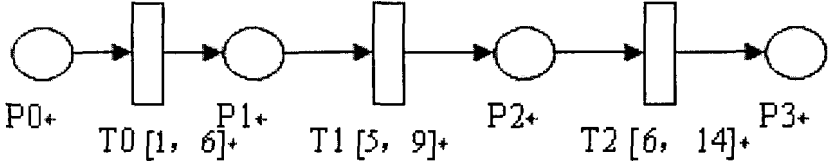


Fig. 3. The purchase procedure with time.

c) At last, analyze the temporal constraints in the system. For example, it is required that goods must be sent in 8 time units after receiving the order form, which will be described by the following TPTL formula:

$$\diamond x.[fi(T1) \wedge \diamond y.[fi(T2) \wedge (y \leq x + 8)]]$$

X denotes the time that $T1$ fires while y denotes $T2$'s firing time. $y \leq x + 8$ shows that $T2$ fires in 8 units of $T1$'s fire.

4. The Verification of TTL-TWF

4.1. The Soundness of TTL-TWF

Comparing to TWF, TTL-TWF has an extra temporal logic constraint set. Therefore, it is necessary to change the firing rule of the model according to the new constraint.

Definition 4.1. For every transition t in TTL-TWF, t can be fired if and only if t is enabled, $\text{mintclock}(t)$, maxt , $\text{clock}(t)$ denotes the delay after t is enabled, and the firing of t does not collide with TS.

Since the firing rule of the model has been changed, the firing sequence will be different accordingly. Here we import function $\text{elapsed}()$ representing the execution time from beginning to the current state, $\text{eval}(t)$ representing the actual firing time of transition t [9].

It is convenient to describe workflow model's possible execution path after the introduction of function elapsed () and eval (t). Two of the execution path of the example mentioned above can be seen in Fig. 4 and Fig. 5.

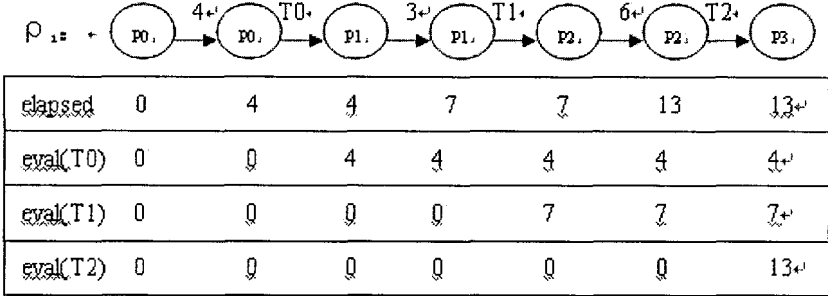


Fig. 4. One of the firing paths of the purchase procedure.

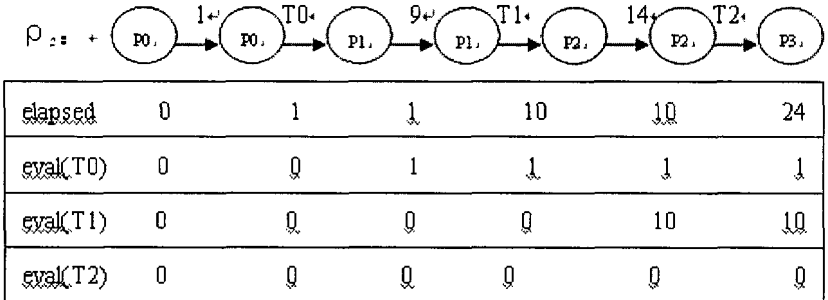


Fig. 5. Another firing path of the purchase procedure.

As mentioned before, the TS of the purchase procedure contains:

$$\diamond x.[fi(T1) \wedge \diamond y.[fi(T2) \wedge (y \leq x + 8)]]$$

X denotes the time that T1 fires while y denote T2's firing time. $y \leq x + 8$ shows that T2 fires in 8 units after T1 fires. Consequently, only ρ_1 satisfies the constraint because T2 fires after 6 time units of T1, while ρ_2 does not fulfil the requirement since T2 fires after 14 time units T1 fires. As we can

see, the execution path of TTL-TWF has been changed according to TWF.

Generally speaking, every workflow model, no matter with constraint or not, must have the possibility to terminate and have no dead transition.

Definition 4.2. A TTL-TWF is sound when:

- 1) For every state M reachable from state i, there exists a firing sequence leading from state M to state o.
- 2) Described by TPTL formulas as below.
 $\diamond(!mk(p_0) \wedge !mk(p_1) \dots \wedge mk(p_n))$, $p_0, p_1, \dots, p_n \in P$
P is a place set, p_0 is source place, p_n is sink place, $mk(p_i)$ denotes there is a token in place p_i . If the TPTL formula is satisfied by the model, state o with at only one token in place o can be eventually reached from state i.
- 3) $\forall ti \in T, \exists [\diamond fi(ti)](i = 1, 2, \dots, n)$ is not valid. T is a transition set, $fi(ti)$ denotes that t fires. The formula is in equivalence with the constraint that there is no dead transition.
- 4) M_0 is the initial state of the TWF in TTL-TWF, $\forall f \in TS, \exists (M_0, \rho) \models f$, that is, $L(TWF) \wedge L(TS)! = null$.

Theorem 4.1. *TTL-TWF is sound if the TWF-net in it is sound and $\exists (M_0, \rho) \models \Phi$, M_0 is the initial state of TWF, ρ is a firing sequence and $\Phi = f_1 \wedge f_2 \wedge \dots \wedge f_n, f_1, f_2 \dots f_n \in TS$.*

Proof. If TWF in TTL-TWF is sound, 1), 2), 3) in definition 2 will be met. So, only $\forall f \in TS, \exists (M_0, \rho) \models f$, that is, $L(TWF) \wedge L(TS)! = null$ is required. \square

Thereby, it is crucial to determine whether $L(TWF) \wedge L(TS)$ is null or not. Some of the work has been done. An algorithm [10] for solving this problem has been brought forward.

- 1) Construct the time reachable graph A of TWF, which is some kind of time automaton.
- 2) $\Phi = f_1 \wedge f_2 \wedge \dots \wedge f_n, f_1, f_2 \dots f_n \in TS, \Psi = \neg \Phi$, construct the Time automaton B for Ψ .
- 3) Get a time automaton C by compute the synchronized product of time automaton A and B.
- 4) If the accepted language of time automaton C is null, TTL-TWF satisfies that $L(TWF) \wedge L(TS)! = null$.

In TTL-TWF, the question that whether the whole procedure can be finished in T time units may be answered. Suppose T_n is the last transition of the procedure, the question can be translated into the following TPTL formula: $\diamond x.[fi(T_n) \wedge (x \leq T)]$.

4.2. The Advantage of TTL-TWF Model

First, TTL-TWF can boost up the description ability of the former model. When only Petri net is applied in workflow modeling, some of the constraint can't be explained. For example, the requirement that if A fires then B can't be fired can not be expressed in the old WF-net.

Second, TTL-TWF can reduce the redundancy of the nodes in the former model. The control logic which requires several nodes to indicate can be expressed by a TPTL formula.

Third, this kind of modeling language is more flexible. Sometimes when the requirement of the system changes, it is no need to modify the TWF-net but only re-defines its constraint formula, which is written in TPTL.

5. Conclusion

In this paper a novel workflow model, Time Workflow Model with Temporal Logic Constraints (TTL-TWF), has been introduced. TTL-TWF is a flexible and verification oriented process model. The soundness of TTL-TWF is redefined and how to verify its correctness is also proposed.

Although some of the model checkers such as Spin can be used to the verification of TTL-TWF, it is still a tough and complicated job when time is considered, in which the time automaton has to be constructed requiring the interference of users. An automatic verifier for the model is to be developed.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant No. 60573160; the Natural Science Foundation of Guangdong Province of China under Grant No. 04009746; the Research Foundation of Science and Technology Plan Project in Guangdong Province of China under Grant No. 2005B10101007; the Research Foundation of Science and Technology Plan Project in Guangzhou City of China under Grant No. 2006Z3-D0371.

References

1. WMP Van der Aalst. The Application of Petri Nets to Workflow Management. *JOURNAL OF CIRCUITS SYSTEMS AND COMPUTERS*, 1998.
2. Huadong Ma. Temporal Logic Based Workflow Service Modeling and Its Application. *CSCWD 2004, LNCS 3168*, pp. 349 – 358, 2005.
3. Huadong Ma. A Workflow Model Based on Temporal Logic. *Computer Supported Cooperative Work in Design*, 2004.
4. Ling, S., Schmidt, H. Time Petri nets for workflow modeling and analysis. In: *Proceedings of the IEEE International Conference on System, Man and Cybernetics*. 2000.4:3039–3044.
5. Z. Manna, A. Pnueli. *The Temporal Verification of Reactive and Concurrent Systems: Specification*. Springer, 1992.
6. R. Alur, T. Henzinger. A Really Temporal Logic. *Journal of the ACM*, 41(1):181–204, 1994.
7. Felder, M., Mandrioli, D., Morzenti, A. Proving Properties of Real-Time Systems Through Logical Specifications and Petri Net Models. *IEEE Transactions on Software Engineering*, Volume 20, Issue 2, pages 127–141.
8. Naoshi Uchihira. A Programming Environment for Reactive and Concurrent Systems Using Petri Nets and Temporal Logic. A thesis submitted to Tokyo Institute of Technology in partial fulfillment of the requirement for the degree of Doctor of Engineering, 1997.
9. T Yoneda, H Schlingloff, EM Clarke. Efficient timing verification based on one-safe time Petri nets and a real-time logic. Presented at *FTC27 workshop*, Toyama, Japan, July 1992.
10. Gerard J. Holzmann. The Model Checker Spin. *IEEE Trans. on Software Engineering*, Vol. 23, No. 5, May 1997, pp. 279–295.