Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/080224

A Novel Approach to Generate the Property for Web Service Verification from Threat-Driven Model

Yonghua Zhu , Honghao Gao[*]

Computing Center, Shanghai University, 200444 Shanghai, P.R. China

Received: 28 Mar. 2013, Revised: 29 Jul. 2013, Accepted: 31 Jul. 2013
Published online: 1 Mar. 2014

Abstract: Web service is considered as one of the most promising computing paradigms, which works as plugin mode to provide the value-added applications in Service-Oriented Computing (SOC) and Service-Oriented Architecture (SOA). The general Web service verification focuses on functionality and its termination, such as deadlock or livelock. However, it might not be able to help in accurately understanding service behaviours because it lacks interaction verifications, especially temporal behaviours. To this point, automatically generating proper temporal logic formulae of verification property is a primary and important task since the manual property generation is time-consuming and error-prone. Thus, this paper proposes an approach extending UML as service threat-driven model to generate the verification property, including the features of functionality, time constraints and probability during service interactions. First, it introduces a scenario description tool, mainly Probabilistic Timed Live Sequence Chart (PTLSC), on which kinds of implication threats are discussed, specifying the insecure behaviours which should be prohibited from occurring in Web service. Second, it gives corresponding transformation methods to extract the verification property from threat-driven model, in which the message coverage criterion and partial relation are employed. These formulae are in the form of Probabilistic Timed Computation Tree Logic (PTCTL), which afford an underlying guideline to guarantee the correctness and reliability of Web service since its threat-carried characteristics.

Keywords: The Verification Property, Service Behaviours, Threat-Driven Model, Formulae Generation.

1 Introduction

Web service is the new feature in constructing e-business applications of model service industry, which supports business agility, flexibility, and availability. The advantage of composting services via the workflow or interaction definition provides the value-added and platform-independent applications. Kinds of services composition approaches have been published by academia and industry, such as Orchestration and Choreography languages [1] (e.g., BPEL4WS, WS-Choreography, and WSCL). However, due to Web service perhaps be developed at different platforms, published by different people, and possibly designed with different implements, the correctness and reliability are hard to guarantee in the case of integrating existing services. Thus, the formal verification of Web service has been attracted more attentions.

Generally, the compatibility is to verify whether the functionality of business logics is correctly implemented, in which most of works study deadlock and livelock checking at the structural level. For each operation of a service requires interface that there is at least one service whose provides interface matches with respect to the correctness of number, sequence, and types of parameters [2,3]. It is interface comparing. Thus, once an assumption is made at the input interface of initial state, the structural verification is to guarantee the final state's output after invoking a service. However, only to consider the structural verification is insufficient because the service process of business logics may have complex dependencies.More functional and non-functional verifications should be performed. Consequently, the Web service verification requires the temporal behavior checking and probability behavior assessment when services support the same functionality at their interface.

Given two collaborated services are compatible with each other, such as booking tickle and tickle processing. But, the temporal behavior among services may influence the overall functionalities, if they are disorderly worked. To the best of our knowledge, the existing verification methods are qualitative. They transform the Orchestration and Choreography specification into Automata, Petri net

[*] Corresponding author e-mail: gaohonghao@shu.edu.cn

or Process Algebra [4,5,6]. After that, the temporal logic verification is employed to check the functional requirements, such as the behavior consistency, and give the true or false assertion. However running under open and ever-changing Internet environment, the non-functional requirement is another important evaluation index. It calls for quantitative verifications, such as performance estimation, which judges the quality of service when a service is invoked, mainly timing-response, cost and reliability. The non-functional requirement ensures that the composite service provides correct values, and the value is produced at the right time-points with lower cost and high reliability.

At present, how to specify and generate the verification property for Web service is still a big challenge. The first important issue is to give the proper property for checking service's behaviors. Undeniably, the manual property generation is time-consuming and error-prone. In addition, it is hard to give the complete correctness and reliability property since services are changeable in anytime and anywhere. However, we can describe threats to represent a scenario of not-allowed behaviors in reverse. If these behaviors occur, we confirm that the interaction of Web service is insecure.

In this paper, we are motivated to consider Web service interactions as a sequence of observed events, and then threat-driven model is proposed to describe these insecure behaviors for the final property generation. There are two novelties: (1) it extends the Live Sequence Chart (LSC) as threat-driven model in which time constrains and probability information are introduced, then gives kinds of threat scenarios; (2) it uses the message coverage criterion and partial relation to extract the verification property from the threat-driven model, by which the generated temporal logic formula carries not only the functional but also the non-functional requirements.

The remainder of this paper is organized as follows: Section 2 introduces preliminary concepts. Section 3 extends LSC for non-functional descriptions. Section 4 presents kinds of scenarios about threat-driven model. Section 5 gives a transformation method to generate the verification property. Section 6 designs a case study. Section 7 summarizes the related works. Section 8 discusses conclusions and future works.

2 Preliminaries

In following paragraphs, we will briefly review the basic concepts about LSC and PTCTL.

2.1 LSC Review

Live Sequence Charts (LSC) proposed by Damn and Harel [7] from Message Sequence Chart (MSC) is a UML-like scenario description language, which can visually express the interaction behavior among multiple processes/agents. It is one of the popular used software specifications. Literatures [7,15,16] about formally generating or transforming temporal logic from LSC scenarios have been successively published.

In LSC, there are two basic elements, mainly pre-chart and main-chart, which are surrounded by a dashed-line hexagon and solid-line rectangle respectively. LSC has two types of scenarios: the Universal scenario chart, and Existential scenario chart. The former is that whenever the pre-chart occurs in an execution, the main-chart shall follow and it must be executed at least once; while the latter describes a possible execution of main-chart in future. The temperature concept is important in location, condition and message of LSC elements, which consists of the hot and cold measurement. If hot is true, the element depicted as solid-line must be occurred.If cold is true, the element depicted as dashed-line may be occurred.

2.2 PTCTL Review

Timed Computation Tree Logic, abbreviated as PTCTL [8], is one of temporal logic. It mainly adds two important operators that the time and probability identifier. Let ζ be the time constraint [9] which is a set of relational expressions over clock X that $\zeta ::= x \sim c \,|x - y \sim c|\, \neg\zeta \,|\, \zeta \wedge \zeta$

For example, $\zeta = (x > 1) \wedge (y < 1)$,where $x, y \in \mathrm{X}$ are clocks and $c \in \mathrm{N}$ is natural number. Time value is Assignment that maps each clock in X to a real number $v : \mathrm{X} \rightarrow R$, i.e.,$\exists x \in \mathrm{X}$, $v(x) = 3$, states that the current time value of clock x is 3. If $v(x) \sim c$, it is called as time satisfaction that $v(x) \models c$, otherwise, it is called as time violation that $v(x) \not\models c$.

In this paper, we employ PTCTL to specify the temporal behavior of Web service. The general PTCTL formula φ is as follows.

$$\varphi ::= true\,|a|\,\zeta\,|z.\varphi|\,\varphi \wedge \varphi\,|\varphi \vee \varphi|\,\neg\varphi\,|\varphi \rightarrow \varphi|\,\mathrm{P}_{\sim p}[\mathrm{X}\varphi] \\ \left|\mathrm{P}_{\sim p}[F\varphi]\right|\mathrm{P}_{\sim p}[G\varphi]\left|\mathrm{P}_{\sim p}[\varphi U \varphi]\right|\mathrm{P}_{\sim p}\left[\varphi U^{\leq k}\varphi\right]$$

Where P is the probability operator,$\sim \in \{<, >, \leq, \geq\}$, a ranges over a set of atomic formulas,$p \in [0, 1]$, and step k and reward r are natural numbers. Freeze quantifier $z \in Z$ is encoded as global clock for time-bounded reachability or response, in which Z is a set of extra clock $X \cap Z = \emptyset$.The temporal operators X, F, G, and U describe functional requirements, which are state quantifiers, meaning neXt state, some Future state, all future states Globally, and Until state, respectively.

For instance, the following properties are probabilistic timed-related descriptions.

(1) $\mathrm{P}_{\geq 0.999}(F(data = delivered))$specifies that the data package will be successfully sent with the probability of more than 0.9999.

(2) $\mathrm{P}_{\geq 0.875}((\neg state = aborts)U(state = success))$ states that the service will not occur interrupts with the probability of more than 0.875.

(3) $\mathrm{P}_{\leq 0.01}(F(data = lost \wedge x \leq 3))$ defines the time constraint that if the time value is less than 3 the data package will be lost with the probability of less than 0.01.

(4) $z.\mathrm{P}_{\leq 0.7}(F(data = delivered \wedge z \leq 5)$) indicates that within 5 time units the data package will be delivered with the probability of less than 0.7.

3 Extend LSC with Time and Probability

In vertical directions of LSC, instance lines are considered as function modules of the target application. In horizontal directions of LSC, the line with an arrow shows the message exchange, simulating the service invocation. The $inst(c) = \{i_0, i_1, ..., i_m\}$ and $M = \{m_1, m_2, ...m_n\}$ denote total instances and messages respectively. Each instance line uses the location to track the message exchange. Each location is marked by digital number. The message between instance lines is triggered by a service through which the interaction comes true. In this paper, the message is extended as follows.

Definition 1 (Message Labeled with Sender and Receiver). For each message, it is a tuple $m = (i, l_s, s, i', l_r)$, that,

1) l_s is a sending location in instance line i.

2) l_r is a receiving location in instance line i'.

3) s is the service invocated between l_s and l_r.

The general LSC describes the functional behavior between interoperated components. However, it doesn't support corresponding non-functional descriptions. Thus, the extended LSC is proposed, called as PTLSC, since its time and probability extensions.

Definition 2 (Probabilistic Timed Live Sequence Chart). The LSC is extended with the time constraint and probability information. It is formally defined as follows:

1) To give the time constraint for each location in instance lines, the location is formalized as tuple $l^t = (l_i, v)$, where l_i is the location of the instance line, and $v \subseteq \Phi(X)$ is the time constraint which is called as location invariant giving the max duration of stay.

2) To describe the time constraint and probability during message exchanges, the message interaction is specified as $(< i, l_s^t >, g, p, m, \{Y\}, < i', l_r^t >)$

where $g \subseteq \Phi(X)$ is the guard condition, p is the probability, $Y \subseteq X$ is a set of reset clocks for the next message interaction.

The location set of LSC is $dom(c,i) = \{l_0^t, l_1^t, ...l_{\max(i)}^t\}$ which represents changes from the initial location l_0^t to the end location $l_{\max(i)}^t$ in instance line i. The set $dom(c) = \{< i, l^t > | i \in inst(c) \wedge l^t \in dom(c,i)\}$ gives the total instances and locations.

All instances initially stay at location l_0^t. With message exchanged, the PTLSC is moved forward with the location changed. The mapping set for the instance line and its location is called as Cut [7], such as $c = (l_{i1}^t, l_{i2}^t, ..., l_{in}^t)$, which is a state of main-chart.The initial state of PTLSC is Cut $c_0 = (l_0^t, l_0^t, ..., l_0^t)$ .For $1 \leq j \leq n$,Cut $c = (l_j^t, l_j^t, ..., l_j^t)$ is the success of Cut $c = (l_i^t, l_i^t, ..., l_i^t)$ if and only if they satisfy $\acute{l}_j = l_j + 1 \wedge \forall i \neq j \bullet \acute{l}_j = l_i$.

Definition 3 (Computing Probability for Message). Given $< m_1, m_2, ..., m_n >$ is a message sequence, where $1 \leq i \leq n, m_i \in M$.

1)If message $m_i = (< i, l >, g, p, e, \{Y\}, < i', l' >)$ is a hot interaction, the probability is $\mathrm{P}(m_i) = p$.

2)If message $m_i = (< i, l >, g, p, e, \{Y\}, < i', l' >)$ is a cold interaction, the probability is $\mathrm{P}(m_i) = p * 0.5$.

3)The total probability $\mathrm{P}(< m_1, m_2, ...m_n >)$ for the message sequence $< m_1, m_2, ...m_n >$ is $\prod_{i=1}^{n} \mathrm{P}(m_i)$.

4 Threat-Driven Model Based on PTLSC.

The security policy describes a set of rules which constrains limited behaviors. It gives a clear safety specification about the software system. The threat behavior is to violate the security policy. Thus, using the property generated from threat scenarios is called as threat-driven verification, and the model specifying threat scenarios is called as threat-driven model.

In the case of threat-driven model, it is required that these threats should not occur. Therefore, we give five categories about threat-driven model, mainly extending Message, Coregion, Simultaneous Region, Conditions and Sub-Chart of LSC.

4.1 Time Constraint Conflicts of Message Exchange

Definition 4 (Time Constraint Conflict between Message Exchange). There is a conflict during the message exchange, if the time for sender and sender is inconsistent. Give two time constrains $x_s = (t_1 \leq x) \wedge (x \leq t_2)$ and $x_1 = (t_3 \leq x) \wedge (x \leq t_4)$ under clock x. The message m between sending location $l_s^t = (l_i, x_s)$ and receiving location $l_r^t = (l_i, x_r)$ will be lost since its conflict.

If the interval $[t_1, t_2] \subseteq [t_3, t_4]$, then the produce of message exchange is safety. Otherwise, there exists security threat.

As Figure 1 shown, the message m_1 is sent when the time value $v(x) = 3$,while the receiver is notified that it should be taken an action to get that message between the time units [1,2]. Finally, the message m_1 is sent out but it can't be normally received after time value $x = 2$. Thus, it is a security threat.

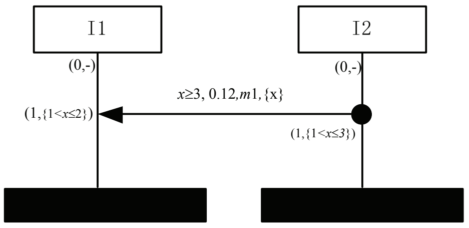

Fig. 1: Time Constraint Conflicts

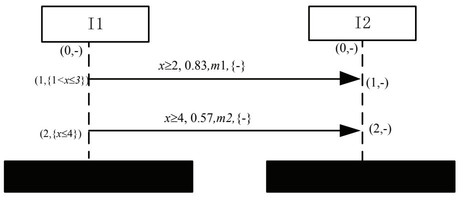

Fig. 2: Coregion Conflict

4.2 Coregion Conflicts

Coregion is the area between dotted vertical lines, in which the interaction behavior of message exchange can be executed without any order. After extending with the time and probability, the execution sequence is controlled by clock, which will lead to deadlock.

Definition 5 (Coregion Conflict). Suppose a set of messages $\{m_1, m_2, ...m_n\}$ under clock x is concentrated in coregion. After a message is selected to be executed, the time value will violate the location invariant of other messages. As a result, that message can't be invoked forever.

The time value $v(x)$ violates the message m_j's location invariant v_j after m_i message is finished its exchange, that $\exists i, j \bullet i \neq j \wedge m_i = \perp \wedge v(x) \notin v_j$.

As Figure 2 shown, there are two messages in coregion. If message m_1 is prior to be executed, the message exchange m_2 will be still executable because the time value $v(x)$ may satisfy the location invariant in I1. l_2^t when message m_1 is finished. But, if message m_2 is prior to be exchanged, then the time value after finish is $v(x) \geq 4$.The interaction exchange of message m_2 will no long be triggered since the current time value exceeds the upper bound of location invariant in I1. l_2^t that $v(x) = 4$ is not in message m_1's interaction interval [2,3].Thus, it is a security threat.

4.3 Simultaneous Region Conflicts

All behaviors in simultaneous regions are synchronous, in which the black point represents the simultaneous region. The simultaneous region conflict destroys the concurrency.

Definition 6(Simultaneous Region Conflicts). Given a set of messages $(m_1, m_2, ...m_n)$ is located in simultaneous regions, in which l_s is their joint sending location. The time value for each message between the location invariant and guard condition is different. Some messages are executable but others are blocked forever.

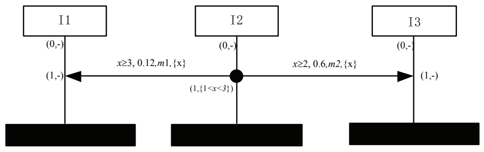

Fig. 3: Simultaneous Regions Conflict

The message m_j's guard condition first satisfy the time value $v(x)$ which make m_j remain unchanged, because the intersection of triggering times $(v_j \cap g_j)$ and $(v_i \cap g_i)$ is empty, that $\exists i, j \bullet i \neq j \wedge g_j \neq g_i \wedge (v_j \cap g_j) \cap (v_i \cap g_i) = \emptyset$

All guard conditions of message exchange must be equal. If this requirement is violated, the execution semantics will be destroyed.

As Figure 3 shown, when the time value $2 \leq v(x) \leq 3$, message m_2 is first selected to execute and message m_1 has to wait. As a result, the completeness of simultaneous can't be guaranteed. Thus, it is a security threat.

4.4 Time Condition Conflicts

The condition crossing multiple instance lines describes a set of predicates that the current locations should satisfy. Only the condition is true, the next message can be exchanged forward. It needs to check whether the predicate constrains are satisfied or not at their location.

Definition 7 (Time Condition Conflict). Given a condition $Cond \subseteq 2^{\zeta}$ under clock x. The time value should be $v(x) \models Cond$. The time condition conflict is in inverse. $\exists x \in X \bullet v(x) \not\models Cond$

As Figure 4 shown , when state $c = ((0,-),(0,-),(0,-))$ is moved to state $c' = ((1,-),(1,\{x \leq 3\}),(1,-))$,the message sequence can be $< m_2 >$ or $< m_1, m_2 >$. Thus, it might reach two possible states $c = ((0,-),(1,\{x \leq 3\}),(1,-))$ and $c = ((1,-),(1,\{x \leq 3\}),(1,-))$ before *Cond1*. In the first case, the messages m_2 will be immediately triggered due to message m_1 is a possible behavior. When this message is sent at the time value between $3 \geq v(x) > 2$, it is a security threat since the clock is not reset which makes the condition $Condl = \{1 < x \leq 2\}$ unsatisfied. In the second case, messages m_1 and m_2 are orderly executed. When messages m_1 and m_2 are sent at the time value $v(x) = 2$, it is a security threat.

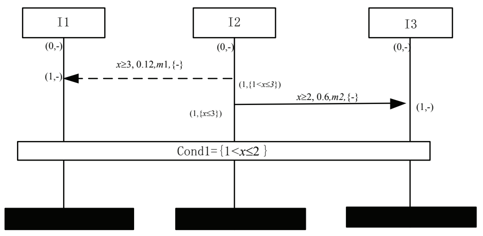

Fig. 4: Conditions Conflict

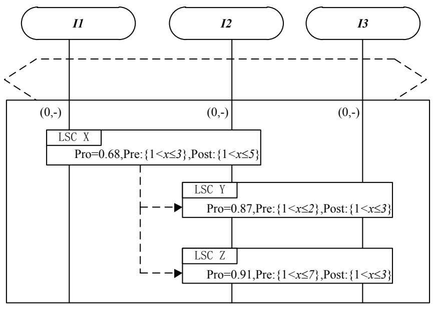
I1
I2
I3
(0,-)
(0,-)
(0,-)
LSC X
Pro=0.68,Pre:{1<x≤3},Post:{1<x≤5}
LSC Y
Pro=0.87,Pre:{1<x≤2},Post:{1<x≤3}
LSC Z
Pro=0.91,Pre:{1<x≤7},Post:{1<x≤3}

Fig. 5: Sub-Chart Composition Conflict

4.5 Timed Sub-Chart Composition Behaviors Conflicts

Definition 8(Non-functional Description for Sub-Chart description). The non-functional description for Sub-Chart is defined as tuple (Pre, Post, Pro).

1)The pre-condition Pre gives the start condition, which is determined by the time constrains of the first message and its location.

2)The post-condition Post gives the end condition, which is determined by the time constrains of the last message and its location.

3)The probability Pro is determined by the first message.

Definition 9 (Timed Sub-Chart Composition Behaviors Conflict). There are two compositions for using sub-charts, mainly sequence and choice. The former executes charts one by one. The latter executes one of components.

1) Given (Pre, Post, Pro) is a pioneer of (Pre', Post', Pro'). It is a conflict when Post $\nsubseteq$ Pre'.

2) Given (Pre, Post, Pro) is a pioneer of (Pre', Post', Pro') and (Pre", Post", Pro").It is a conflict when Post $\nsubseteq$ Pre' $\bigvee$ Post $\nsubseteq$ Pre".

As Figure 5 shown, it defines a choice behavior that the connection line is cold. PTLSC Y and Z follows PTLSC X. Due to $X.Post \nsubseteq Y.Pre$, the PTLSC Y will be ignored during each choice execution. This scenario is the process starvation problem. It is a security threat too.

5 Generate Temporal Logic Formulae from Threat-Driven Model.

The verification property generated from threat-driven model can be used as input to the verification supporting tool. If the property is satisfied, the implementation model hides bugs. In threat-driven model, the message exchange is considered as a service invocation process. Each message is mapped to a service. Thus, the message coverage criterion is first introduced.

Definition 10 (Message Coverage Criterion). For arbitrarily messages m_i and m_j, the rules based on partial relation $\prec$ are defined as follows [7]:

1), If the partial relation $m_i \prec m_j$ is true, the property $\phi_{m_i m_j} = \neg m_j U m_i$ specifies that message m_j shouldn't be invoked before message m_i.

2), If they are not partial relation $m_i \nprec m_j$, the property $\chi_{m_i m_j} = (\neg m_j \wedge \neg m_i) U (m_i \wedge X((\neg m_j \wedge \neg m_i) U m_i))$ specifies that message m_j occurs twice before message m_i.

3), The $\neg \chi_{m_i m_j}$ states that message m_i can't occur twice before invoking message m_j.

Definition 11 (Transformation Template). In order to give the non-functional requirement, the formulae $\phi_{m_i m_j}$ and $\chi_{m_i m_j}$ are extended as the transformation template, labeling with time and probability, that,

1),$PT(\phi_{m_i m_j}) = \mathrm{P}_{\leq p}(\neg m_j U m_i \wedge Tim(m_i))$ where the *symbol* $\leq p$ gives the message exchange probability of m_i,the time constrain $Tim(m_i)$ depends on the union set of the location invariance and the guard condition during message m_i is exchanged.

2),$PT(z.\phi_{m_i m_j}) = z.\mathrm{P}_{\leq p}(\neg m_j U m_i \wedge Tim(m_i) \wedge z \leq UT)$ is extended from formula 1) where the freeze predicate z is used as global time constrains, the parameter UT is specified for z instantiation.

3),$PT(\neg \chi_{m_i m_j}) = \mathrm{P}_{\leq p}(\neg((\neg m_j \wedge \neg m_i) U (m_i \wedge X((\neg m_j \wedge \neg m_i) U m_i \wedge Tim(m_i)))))$ states that message shouldn't be occurred twice. The *symbol* $\leq p$ gives message execution probability of m_1.

4), $PT(z.\neg \chi_{m_i m_j}) = z.\mathrm{P}_{\leq p}(\neg((\neg m_j \wedge \neg m_i) U (m_i \wedge X((\neg m_j \wedge \neg m_i) U m_i \wedge Tim(m_i) \wedge z \leq UT)))$ shows that the message m_1 shouldn't be occurred twice before message m_2,and the user specified global time constrains UT should be ensured.

The transformation template discussed above inserts non-functional descriptions into temporal behaviors. For example,$PT(\phi_{m_1 m_2}) = \mathrm{P}_{\leq 0.75}(\neg m_2 U m_1 \wedge 2 \leq x \wedge x \leq 3)$ shows that message m_1 occurs before message m_2 , in which the time constraint is interval [2,3] and the probability is less than 0.75. Furthermore, if the message exchange should be accomplished within 6 time units, then the property should be $PT(z.\phi_{m_1 m_2}) = \mathrm{P}_{\leq 0.75}(\neg m_2 U m_1 \wedge 2 \leq x \wedge x \leq 3 \wedge z \leq 6)$. For each scenario of threat-driven model, employing the transformation template can generate the corresponding threat-carried property. First, the symbol p and m corresponds to the message in pre-chart and main-chart, respectively, and the symbol e stands for all messages

$e = p \cup m$. After that, the property generation is processed at the pre-chart and main-chart level, considering using transformation templates for specifying non-functional requirements.

For pre-chart [15], the formula is extended as $\phi_{pc} = \underset{p_i \prec p_j}{\Lambda} PT(\phi_{p_i p_j}) \wedge \underset{\forall p_i, m_j}{\Lambda} PT(\phi_{p_i m_j}) \wedge \underset{p_i \not\prec p_j}{\Lambda} PT(\neg\chi_{p_j p_i})$, where 1), $\underset{p_i \prec p_j}{\Lambda} PT(\phi_{p_i p_j})$ gives the partial relation formulae that the message p_j shouldn't be occurred before message p_i. 2), $\underset{\forall p_i, m_j}{\Lambda} PT(\phi_{p_i m_j})$ shows the message in pre-chart should be occurred before the message in main-chart. 3), $\underset{p_i \not\prec p_j}{\Lambda} PT(\neg\chi_{p_j p_i})$ states that in pre-chart the message p_i can't be occurred twice before invoking message p_j.

For main-chart [15], the formula is extended as$\phi_{mc} = \underset{m_i \prec m_j}{\Lambda} PT(\phi_{m_i m_j}) \wedge \underset{m_j\ is\ \max}{\Lambda} PT(F(m_j)) \wedge \underset{\forall e_i, m_j}{\Lambda} PT$ $(\neg\chi_{e_i m_j})$ where 1), $\underset{m_i \prec m_j}{\Lambda} PT(\phi_{m_i m_j})$ gives the partial relation formulae that the message m_j shouldn't be occurred before message m_i. 2), $\underset{m_j\ is\ \max}{\Lambda} PT(F(m_j))$ shows that if m_j is the final message it will be executed eventually. 3), $\underset{\forall e_i, m_j}{\Lambda} PT(\neg\chi_{e_i m_j})$ states that all events can't be occurred twice.

Definition 12 (Compositing Formulae). After obtaining ϕ_{pc} and ϕ_{mc}, the verification property, such as liveness and reachability [10], can be composited as following rules.

1), The Universal LSC is a mandatory scenario which describes the liveness property. Thus, in our threat-driven model, we introduce $P_{\leq 1}(G(\phi_{pc} \rightarrow F\phi_{mc}))$ for liveness that globally if ϕ_{pc} in pre-chart is satisfied then ϕ_{mc} in main-chart will be eventually occurred in future.

2), The Existential LSC is an optional scenario which describes the reachability property. Thus, in our threat-driven model, we introduce $P_{\leq 1}(F\phi_{mc})$ for reachability that ϕ_{mc} in main-chart will be occurred ultimately.

3), Based on above formulae,$P_{\leq 1}(G(\acute{\phi}_{pc} \rightarrow F\acute{\phi}_{mc}))$ and $P_{\leq 1}(F\acute{\phi}_{mc})$ are introduced for the special assign composition, where $\acute{\phi}_{pc}$ is a part of ϕ_{pc} and $\acute{\phi}_{mc}$ is a part of ϕ_{mc}.

6 Case Study

In order to show the feasibility of the proposed approach for verification property generation, we carry out a simple example in Figure.6 about how to generate PTCTL formulae. The original message sequence is $< m_2, m_1 >$. But in pre-chart, the threat-driven model gives an error sequence that message m_1 is sent before message m_2. Moreover, in main-chart, message m_4 is blocked after message m_3 is finished. Note that the time value after

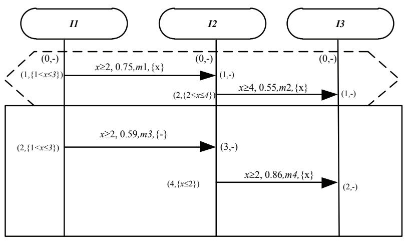

Fig. 6: Case Study

message m_3 occurs is greater than 2, while the location invariant of I2.l_4^t is less than 2. Thus, the scenario is a threat-driven model.

There are three instance lines {I1,I2,I3} with different locations. The instance lines I1, I2, and I3 are functional modules. The messages exchanged between instance lines are services. The partial relation of the message $\{m_1\} \prec \{m_2\} \prec \{m_3\} \prec \{m_4\}$. The clock is used for time constraints. According to the method proposed in section 5, we can get following formulae.

Part I: For pre-chart, the formulae are as follows:

1), $\underset{p_i \prec p_j}{\Lambda} PT(\phi_{p_i p_j}) = P_{\leq 0.75}(\neg m_2 \mathbf{U} m_1 \wedge 2 \leq x \wedge x \leq 3)$.

2),

$$\begin{aligned} \underset{\forall p_i, m_j}{\Lambda} PT(\phi_{p_i m_j}) = {} & P_{\leq 0.75}(\neg m_3 \mathbf{U} m_1 \wedge 2 \leq x \wedge x \leq 3) \\ & \wedge P_{\leq 0.75}(\neg m_4 \mathbf{U} m_1 \wedge 2 \leq x \wedge x \leq 3) \\ & \wedge P_{\leq 0.55}(\neg m_3 \mathbf{U} m_2 \wedge x = 4) \\ & \wedge P_{\leq 0.55}(\neg m_4 \mathbf{U} m_2 \wedge x = 4). \end{aligned}$$

3), $\underset{p_i \not\prec p_j}{\Lambda} PT(\neg\chi_{p_j p_i}) = PT(\neg\chi_{m_1, m_2})$.

$PT(\neg\chi_{m_1, m_2}) = P_{\leq 0.75}(\neg((\neg m_1 \wedge \neg m_2) U m_2 \wedge X((\neg m_1 \wedge \neg m_2) U m_2 \wedge 2 \leq x \wedge x \leq 3)$

Part II: For main-chart, the formulae are as follows:

1), $\underset{m_i \prec m_j}{\Lambda} PT(\phi_{m_i m_j}) = P_{\leq 0.59}(\neg m_4 U m_3 \wedge 2 \leq x \wedge x \leq 3)$.

2), $\underset{m_j\ is\ \max}{\Lambda} PT(F(\mathrm{m}_j)) = P_{\leq 0.86}(F(m_4 \wedge x = 2))$.

3),

$$\begin{aligned} \underset{\forall e_i, m_j}{\Lambda} PT(\neg\chi_{e_i m_j}) = {} & PT(\neg\chi_{m_1, m_3}) \wedge PT(\neg\chi_{m_2, m_3}) \\ & \wedge PT(\neg\chi_{m_3, m_3}) \wedge PT(\neg\chi_{m_4, m_3}) \\ & \wedge PT(\neg\chi_{m_1, m_4}) \wedge PT(\neg\chi_{m_2, m_4}) \\ & \wedge PT(\neg\chi_{m_3, m_4}) \wedge PT(\neg\chi_{m_4, m_4}) \end{aligned}$$

$PT(\neg\chi_{m_1, m_3}) = P_{\leq 0.75}(\neg((\neg m_1 \wedge \neg m_3) U m_3 \wedge X((\neg m_1 \wedge \neg m_3) U m_3 \wedge 2 \leq x \wedge x \leq 3)$;
$PT(\neg\chi_{m_2, m_3}) = P_{\leq 0.55}(\neg((\neg m_2 \wedge \neg m_3) U m_3 \wedge X((\neg m_2 \wedge \neg m_3) U m_3 \wedge x = 4)$;
$PT(\neg\chi_{m_3, m_3}) = P_{\leq 0.59}(\neg((\neg m_3) U m_3 \wedge X((\neg m_3) U m_3 \wedge 2 \leq x \wedge x \leq 3)$;
$PT(\neg\chi_{m_4, m_3}) = P_{\leq 0.86}(\neg((\neg m_4 \wedge \neg m_3) U m_3 \wedge X((\neg m_4 \wedge \neg m_3) U m_3 \wedge x = 2)$;
$PT(\neg\chi_{m_1, m_4}) = P_{\leq 0.75}(\neg((\neg m_1 \wedge \neg m_4) U m_4 \wedge X((\neg m_1 \wedge$

$\neg m_4)Um_4 \wedge 2 \leq x \wedge x \leq 3)$;
$PT(\neg\chi_{m_2,m_4}) = P_{\leq 0.55}(\neg((\neg m_2 \wedge \neg m_4)Um_4 \wedge X((\neg m_2 \wedge \neg m_4)Um_4 \wedge x = 4)$;
$PT(\neg\chi_{m_3,m_4}) = P_{\leq 0.59}(\neg((\neg m_3 \wedge \neg m_4)Um_4 \wedge X((\neg m_3 \wedge \neg m_4)Um_4 \wedge 2 \leq x \wedge x \leq 3)$;
$PT(\neg\chi_{m_4,m_4}) = P_{\leq 0.86}(\neg((\neg m_4)Um_4 \wedge X((\neg m_4)Um_4 \wedge x = 2)$

Part III: Since the threat-driven model in Figure 7 is Universal, the formula $\mathrm{P}_{\leq 1}(G(\phi_{pc} \rightarrow F\phi_{mc}))$ gives the liveness property. The more formulae can be generated using $\mathrm{P}_{\leq 1}(G(\acute{\phi}_{pc} \rightarrow F\acute{\phi}_{mc}))$. For completeness, we give a simply introduction that formulae $P_{\leq 0.75}(\neg m_3 Um_1 \wedge 2 \leq x \wedge x \leq 3)$ of $\bigwedge_{\forall p_i m_j} PT(\phi_{p_i m_j})$, $P_{\leq 0.55}(\neg((\neg m_2 \wedge \neg m_3)Um_3 \wedge X((\neg m_2 \wedge \neg m_3)Um_3 \wedge x = 4)$ of $PT(\neg\chi_{m_2,m_3})$ can be integrated as $P_{\leq 1}(G(P_{\leq 0.75}(\neg m_3 Um_1 \wedge 2 \leq x \wedge)x \leq 3) \rightarrow F(P_{\leq 0.55}(\neg((\neg m_2 \wedge \neg m_3)Um_3 \wedge X((\neg m_2 \wedge \neg m_3)Um_3 \wedge x = 4))))$

Due to the formulae generated from threat-driven model represent a set of insecure temporal behaviors, if the service interaction is checked and no counterexample is output, we can make sure that the current Web service has bugs or may be failure in future.

7 Related Works

There is ongoing interest in translating specifications into the verification property. Researchers have devoted themself to the verification property generation. For example, Minmin [11] used StateCharts to formally model adaptive Web application navigations and shown how important properties of navigation model were verified. Rogin [12] proposed a new methodology to automatically generate complex properties for a given design, which described the abstract design behavior and improved design understanding. Hu [13] proposed a new general approach to property verification for access control models. The approach defined a standardized structure for access control models, providing for both property verification and automated generation of test cases. Soeken [14] developed an approach to assist the automatic generation of properties from the protocol specification for the formal verification of bus bridges. The technical contribution was that the final set of the verification suite was functionally complete in respect to the underlying verification tool which shows the absence of any verification holes.

In LSC, Hillel [15] proposed an approach to generate temporal logic formulae for scenario-based specifications. Based on his research, this paper introduces the threat-driven model concept because we can't give a clear description about how and what property the Web service should be reserved. The generated verification property is threat-carried formulae. Thus, the advantage of our method enables the adaptability and flexibility of Web service verification.

8 Conclusions

There is a growing demand to realize the complex business processes by combining and reusing available Web services over Internet. The formal verification plays as the core guarantee in SOC and SOA implementation before the service-based application is deployed. In this paper, we discuss an approach to extract the verification property from specifications. First, PTLSC is extended from LSC with time and probability. It is used as threat-driven model for specifying threat scenarios. Second, the transformation method is introduced to translate threat-driven model into temporal logic formulae for verifying Web service. Finally, a case study demonstrates the feasibility of our proposed approach. As for further research, we will focus on adding data-flow to threat-driven model, specifying the data intensive service. The main problems are that our approach may generate functionality-duplicated formulae and the sheer size is its limitation [15,16]. Thus, we will consider using the property rewritten technique to alleviate this problem.

References

[1] J. Rao and X. Su. A Survey of Automated Web Service Composition Methods. In Proc. the 1st International Workshop on Semantic Web Services and Web Process Composition. Lecture Notes in Computer Science,Springer-Verlag, Berlin, Heidelberg, 43-54 (2004).

[2] T. Bultan, J. Su, and X. Fu. Analyzing conversations of Web services. IEEE Internet Comput., **10**, 18-25 (2006).

[3] B. Daniela and C. Diego. Automatic service composition based on behavior descriptions. Int J Cooper Inf Syst., **14**, 333-376 (2005).

[4] HMW.Verbeek and van der Aalst WMP. Analyzing BPEL processes using Petri Nets. In Proc. international workshop on applications of Petri Nets to coordination, workflow and business process management, IEEE Computer Society, 59-78 (2005).

[5] R. Hull and J Su. Tools for composite Web services: a short overview. SIGMOD Record., **34**, 86-95 (2005).

[6] C. Salaun, L. Bordeaux, and M. Schaerf. Describing and reasoning on Web services using process algebra. Int J Bus Process Integr Manage, **1**, 116-128 (2006).

[7] W. Damm and D. Harel. LSCs: Breathing Life into Message Sequence Charts. Formal Methods in System Design, **19**, 45-80 (2001).

[8] M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Automatic Verification of Real-time Systems with Discrete Probability Distributions. Theoretical Computer Science, **282**, 101-150 (2002).

[9] R. Alur and D. L. Dill. A theory of timed automata, Theor. Comput. Sci., **126**, 183-235 (1994).

[10] B. Berard, M. Bidoit, Finkel A., et al. Systems and Software Verification: Model-Checking Techniques and Tools. Lecture Notes in Computer Science. Springer-Verlag, Berlin, Heidelberg, (2001).

[11] M. Han, and C. Hofmeister. Modeling and verification of adaptive navigation in web applications. In Proc. 6th international Conference on Web Engineering, ACM Press, 329-336 (2006).
[12] F. Rogin, T. Klotz, G. Fey, R. Drechsler, and S. Rulke. Advanced verification by automatic property generation. IET Comput. Digit. Tech., **3**, 338-353 (2009).
[13] C. Vincent, D. Hu, K. Richard, and X. Tao. Property Verification for Generic Access Control Models. In Proc. 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing - Volume 02 (EUC '08),IEEE Computer Society, **2**, 243-250 (2008).
[14] M. Soeken,; U. Kuhne, M.Freibothe, G. Fe. Automatic property generation for the formal verification of bus bridges, IEEE Computer Society, 417-422 (2011).
[15] K. Hillel, D. Harel, A. Pnueli, Y. Lu, and Y. Bontemps. Temporal logic for scenario-based specifications. In Proc. 11th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Heidelberg, **3440**, 445-460 (2005).
[16] J. Klose, T. Toben, B. Westphal, and H. Wittke. Check itout: On the efficient formal verification of live sequence charts. Computer Aided Verification. Lecture Notes in Computer Science, Springer-Verlag, Berlin, Heidelberg, **4144**, 219-233 (2006).

Yonghuaa Zhu received the Ph.D degree from the School of Communication and Information Engineering of Shanghai University, Shanghai, China, in 2006. His research interests include formal method, high performance computing, communication and information engineering.

Honghao Gao received the Ph.D degree in computer application technology from the School of Computer Engineering and Science of Shanghai University, Shanghai, China, in 2012. His research interests include Web service and model checking.