Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/090451

A Novel Approach to Dynamically Construct User Navigation-Oriented Quantitative Model for Web Service

Honghao Gao [1,2,∗], *Huaikou Miao* [2,3] , *Hongwei Zeng* [2] *and Yonghua Zhu* [2]

[1] Computing Center, Shanghai University, 200444 Shanghai, P.R. China
[2] School of Computer Engineering and Science, Shanghai University, 200444 Shanghai, P.R. China
[3] Shanghai Key Laboratory of Computer Software Evaluating&Testing,201112 Shanghai, P.R. China

Received: 8 Jun. 2013, Revised: 5 Mar. 2014, Accepted: 6 Oct. 2014
Published online: 1 Jul. 2015

Abstract: Web service is considered as one of computational resources available on network, which has become a popular paradigm to develop platform-independent applications. However, due to the diverse-grained, non-determinism, and timeliness natures of Web service, it is not surprising that Web-based service application frequently experiences problems, such as unreachable pages and reduced responsiveness. To address this problem, the primary task is to verify and test Web engineering for trustworthy, in which the quantitative model generation of service behaviors plays a critical role during software life cycle, especially business processes verification. In this paper, navigation behaviors between Web application and users are formalized as service process. Then, the navigation model corresponding to service process is extended with non-functional specifications by using probability matrix and time constraint pair method, mainly random natures and time features. Finally, the comprehensive model and its parallel composition are introduced to cope with more complex service interactions modeling. In conclusion, our navigation-oriented quantitative model specified with time constrains and probability specifications gives a solution to dynamic features extraction for constructing hybrid model of Web service.

Keywords: User Navigations, Event Logs, Service Process, Probabilistic Timed Specifications.

1 Introduction

In global market, modern enterprise should have abilities to seize the fleeting business opportunity and make a quick response. However, traditional software is hard to meet these demands since its closed architecture can't handle the dynamic nature, which requires Web service and Service Oriented Architecture (SOA) to across organizational boundaries, supporting platform independent applications integration to promote more business transactions. First, Web service maximizes business benefits that companies can directly employ existing third-party services and public interfaces to realize inter-organizational cooperations. Second, business applications implemented by Web services enhances process much more agility, flexibility and availability, therefore, which have been widely used in multi-fields to deal with complex commercial logics, for instance, supply chain management and workflow management. Furthermore, many academic researches have been published about service software development and utilization, including services discovering, composition, and verification techniques. Consequently, applying Web service to develop the loosely-coupled and interoperable business software is regarded as the next-generation distributed computing model [1].

Due to the heterogeneous, open and collaborative natures of Internet, any failure occurred in software may immediately result in service interrupts. It will seriously impact the correctness and reliability of business process leading to huge economic losses. How to model Web service is a highly challenging task before verification. First of all, it needs to formalize Web service in the form of formal specification. Then, it needs to extend non-functional characteristics under uncertain execution environments. The expected service model should consider following new requirements: 1) Diverse-grained, service software dynamically invokes different grained Web services to achieve complicated logic composition goals. 2) Non-determinism, due to instable Internet, service components usually exhibit stochastic behaviors with random failure phenomenon. 3) Timeliness, each

∗ Corresponding author e-mail: gaohonghao@shu.edu.cn

service should accomplish its interactions within time limitation that is not only specified in functional requirements but also restricted in service time-out definitions. Given these premises, non-functional attributes are important features as well as behavioral modeling, such as time constrains and probabilistic behaviors.

Few works focused on non-functional formal modeling. The traditional service description languages, such as Orchestration and Choreography (e.g., BPEL4WS, WS-CDL, WSCL), Semantic Web specifications (e.g., OWL-S, SAML-S), and Workflow Management patterns (e.g., XPDL, XLANG), are proposed for high-level descriptions. Then, they are transformed into FSM model, Petri Net or Process Algebra for service modeling, respectively. However, these above methods have a narrow applicable scenario due to their functional modeling characteristics. The non-functional service attributes are dynamic features and the quality of service is user subjective feelings during service executions [2,3,4], such as time-response, cost and probability. The real-time property of Web service depends on the input/output and its timing [5]. The cost property of Web service refers to utilize Web service with little expense. The probability property of Web service guarantees the robustness when service-based software is invoked. In short, a response should provide correct values that are outputted at the right time-points with lower cost and high reliability. Critically, this new challenge has caused the issue of quantitative verification, such as ”file download service may be failure with probability 0.15 or greater”. Unfortunately, so far no effective and systematic approach can solve these quantitative requirements in non-functional specification aspect.

In this paper, as a base research object, Web application is researched on assumption that Web service used in each logic Web page is a functional component. Business processes of Web application are implemented by standard interfaces and SOAP protocols of Web services. Web user operations reflected to business logics are navigational behaviors when they interact with service-oriented software. For example, browsing Web, receiving and sending emails, uploading/downloading documents, and online payment. These behaviors are equal to service process. But, not all these operations are key behaviors. Support 5% users are loyal buyers, and 95% users are occasional buyers. Loyal customers are core and important. Therefore, modeling core buyer behaviors will help business to analyze and summarize hot commodity to attract more occasional buyers' attentions. For the former, if all loyal buyers (5% users) give up shopping, it will case huge enterprise economic benefits loss. For the latter, they only search or browse product information without ordering any things or triggering core business processes. Thus, there is no economic benefit.

However, dynamically constructing such user-oriented service model asks for technology solutions that comprehensively specify service quantitative features in order to give a formal model before further verifications. Service worked for different kinds of users will display with different QoS characteristics. Especially under different invocation time and consumption situations, user will be underwent different satisfactions. Evidently, the long time waiting and stochastic denial of service will impact on software services. For these reasons, the quantitative service model has been more and more important to verify the correctness and reliability of Web-based service application. We are motivated to research the quantitative model for Web service. Our main works are divided into following two steps: 1) it generates the functional model from user navigation behaviors; 2) it extracts non-functional features to extend the former model, including probabilistic behaviors and time constrains.

The remainder of this paper is organized as follows: Section 2 analyzes user group behaviors. Section 3 gives an approach to model functional behaviors from user navigation behaviors. Section 4 extends functional model with the probability and time specification. Section 5 introduces a navigation-oriented quantitative model and its parallel composition. Section 6 reviews the related works. Section 7 draws a conclusion and future works.

2 User Group Behaviors

Service processes of business logic are not active at the same time, which are worked under event-driven way that they are determined and invoked when users access to. Therefore, Web application gives a best option to service behaviors mining since navigation behaviors and user interactions recorded in Web servers. As Figure.1 shown, there is a relationship among Web application, service process and Web service. In service level, Web service publishes its functions via interface to service process level. While in Web application level, each service behavior is mapped to a service process during user activities. Each page link triggered from Web application has a corresponding service process and active service working in background. In the study of user group behaviors, Web users with similar interests will show similar behaviors on Web application that they have same service processes.

However, compared to most service modeling related literatures, little work has been done on taking care of user group behaviors. One of main problems we face is target user who will pay attentions to Web application and create commercial value to e-business.

Definition 1 User Group Classification. From the perspective of user order transactions, the user group of service software is divided into three categories: the valuable customers (*VeC*) with action set Act^v, the

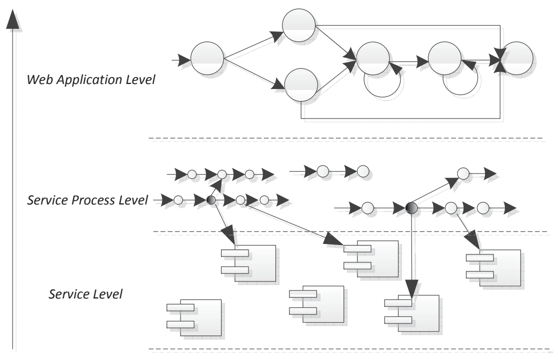

Fig. 1: The Service Software Architecture

potential customers (*PlC*) with action set Act^p and the valueless customers (*VsC*) with action set Act^e.

The user navigation of *VsC* users has an insignificant impact to business process. They often leave when they visit Website at the first time. In contrast, *VeC* users generate more commercial values. For example, they order goods via a serial of interactive operations, such as in trip arrangement and on-line shopping. While *PlC* users may cancel their orders due to poor software performance that goods in shopping cart will not be transferred into currency, or activity will be terminated in advance.

The core activity $C \subseteq T$ of important user behaviors is a key measurement,which contributes to construct a well-formed model for Web service, that,

1) Navigation behaviors included core activities will be considered as valuable customer actions. The action set is $Act^v ::= \{t_i|\forall a_i \in G^v, \exists t_i \in T \bullet a_i \rightarrow t_i\}$, if and only if each user belonged to valuable customer $a_i \in G^v$ has triggered a activity $a_i \rightarrow t_i$, and $\exists t_i \in Act^v \bullet t_i \in C$ is an core activity.

2) Navigation behaviors included general activities will be seamed as potential customer actions. The action set is $Act^p ::= \{t_i|\forall a_i \in G^p, \exists t_i \in T \bullet a_i \rightarrow t_i\}$, if and only if each user belonged to valuable customer $a_i \in G^p$ has triggered a activity $a_i \rightarrow t_i$, and $\forall t_i \in Act^p \bullet t_i \notin C$ are not core activity. But their post succeed behavior will be core activity.

3) Navigation behaviors included irrelevant activities will be treated as valueless customer actions. The action set is $Act^e ::= \{t_i|\forall a_i \in G^e, \exists t_i \in T \bullet a_i \rightarrow t_i\}$, if and only if each user belonged to valueless customer $a_i \in G^e$ has triggered a activity $a_i \rightarrow t_i$, and $\forall t_i \in Act^e \bullet t_i \notin C$ are not core activity. Moreover, their post succeed behavior will also not be core activity.

To characterize service process, experiments are carried out to explore impact factors. We build a simulation environment that a set of services are published using IBM XML generator in Online Survey System (OSS). Totally, more than 30 services are developed, which can be invoked by corresponding requests from Web pages during user interacts with Web browsers. More than 800 students are requested to fill in survey questions. From the user satisfaction of different viewpoints, the response time and reliability are considered as evaluation indicators. The stability of network is denoted as r and its value is figured out by unreachable pages and total access times. Then, the response time of each service denoted as t is controlled by its intrinsic session parameter settings. From Table.1, we find that with the stability of network reduced the active users are decreased. However, with the response time increased the active users are decreased. In summary, regarding probability and time constraint in OSS, the acceptable boundary for network stability r and response time t is 0.6 and 12s, respectively.

When r is less than 0.5 or t is greater than 12s, Web application will perform low QoS. The inactive users set includes valueless customers (*VsC*) and potential customers (*PlC*). When $r = 0.9$ and $t < 1s$, the user loss ratio is 0.17. The inactive users set consists of potential customers (*PlC*) because they will give up further operations and terminate current sessions. As important reference factor to enterprises application, the user loss ratio is defined as follow.

Definition 2 User Loss Ratio. Suppose that the valueless customers (*VsC*), the valuable customers (*VeC*), and the potential customers (*PlC*) are identified. To measure the user loss ratio of service, symbol η is to denote active user percent, that,

$$\eta = \frac{VsC}{VsC + VeC + PlC}$$

Definition 3 Success Order Ratio. The service process can be decomposed into sub-process since different structures will be used in different composition ways. The success order ration r is as follows.

1) If it is a sequence structure, success order ratio is $r = (1-\eta_1) * (1-\eta_2) * \ldots * (1-\eta_n)$. Each mode displays the success probability of $1-\eta_i$.

2) If it is a parallel structure, success order ratio is $r = 1 - \eta_1 * \eta_2 * \ldots * \eta_n$. The service process structure is synchronous communication behavior.

The good user experience and quality of service determine the life cycle of service software. Support Web user number is constant *C=VsC+VeC+PlC*. When user experiences problems with unreachable pages and reduced responsiveness, all potential customers *PlC* may be converted to valueless customers *VsC*, that is

$$VsC' = VsC + PlC$$

which makes active user percent $\eta' = \frac{VsC'}{VsC+VeC+PlC}$ increased and $\eta = \frac{VsC}{VsC+VeC+PlC}$ reduced. Finally, the success order ratio r will be low. In the worst scenario, all

Table 1: Active users of fill-in service in OSS

	$r=0.9$	$r=0.8$	$r=0.6$	$r=0.5$	$r=0.4$	$r=0.3$	$r=0.2$
t<1s	0.83	0.82	0.77	0.43	0.2	0.13	0.01
$t\in(1s,3s)$	0.83	0.82	0.77	0.4	0.2	0.13	<0.1
$t\in(3s,6s)$	0.82	0.81	0.74	0.4	0.2	<0.1	<0.1
$t\in(6s,9s)$	0.82	0.79	0.74	0.4	0.2	<0.1	<0.1
$t\in(9s,12s)$	0.7	0.67	0.51	0.4	0.1	<0.1	<0.1
$t\in(12s,15s)$	0.65	0.65	0.5	0.1	<0.1	<0.1	<0.1
t≥15s	0.45	0.4	0.4	0.1	<0.1	<0.1	<0.1

users are Website visitors *VsC*= *VsC*+*PlC*+*VeC* which makes user loss ratio η=1 that they are valueless customers (*VsC*). However, in the best scenario, all users are Website orders *VeC*=*VsC*+*PlC*+*VeC* which makes user loss ratio η=0 that there is no valueless customers (*VsC*) or potential customers (*PlC*).

Different customer groups generate different behavior models. It requires a suitable formal model to analyze why user gives up shopping or changes to access to competitors services. The tradeoff between valuable customers *VeC* and potential customers *PlC* should be considered. In practice, it will expense higher costs for converting a valuable customer *VeC* from potential customers *PlC* than retaining a valuable customer *VeC*. The behaviors of valuable customers *VeC* contributes to maximize enterprise beatifies. To this end, this paper mainly studies how to model *VeC* navigation behaviors.

3 Functional Behaviors Modeling

Web application works as the most intuitive manner of using Web service to provide the optimizations and desired functionality, which facilitates behaviors modeling via data mining and analysis from event logs. User navigation behavior of Web application is one possible alternative to construct service model. Thus, in our research, the user navigation model is taken in account to formalize service process, that user group behaviors from valuable customers.

3.1 Modeling User Navigations

Various link types between Web pages are relevant to navigational purposes, such as forms and frames. Hyperlinks linked by *page-to-page* paradigm connect different Web pages as an organized network. The next page is determined by the current page and its hyperlinks. When user visits Web-based service application, the navigation consists of possible sequences of Web pages. Web server records all behaviors as event logs, such as pages resources, click time and user information. Thus, utilizing these user navigations can reproduced corresponding services and service process supported in background.

Definition 4 Navigation Sequence Set (NSS). It is a finite set of navigation paths NSS=$\{\pi_1,\pi_2,\pi_3,\ldots\}$. Each navigation sequence π_i contains states of logic Web page $\pi_i =< s_1,s_2,..,s_k,s_{k+1} >$ satisfying that $\forall k \geq 1 \bullet (s_k,s_{k+1}) \in R$ is a page link. R is a set of navigations.

The navigation sequence set corresponds to Web page visit path. Each page state s_i is a functional service invoked by logic Web page, which user has interacted with. If $s_1 = s_0$ starting from index page, then the navigation path π_i is called as an execution sequence. To identify transition relation among page links, for instance, $\exists s,s' \bullet (s,s') \in R$, the $Pre/Post$ succeed states definition of each page state should be satisfied.

Definition 5 Pre/Post Succeed States. Given a state $s \in S$. The pre/post succeeds page states of state s are defined as follows:

Pre-Succeed States: $Post(s) = \{s' | \exists s' \in S \bullet (s,s') \in R\}$

Post-Succeed States: $Pre(s) = \{s' | \exists s' \in S \bullet (s',s) \in R\}$

The transition relation is miraculous. Let s be a page state. One miraculous is bad state transition. When s is not end state, it satisfies $Post(s) = \emptyset$ and $Pre(s) \neq \emptyset$. The other miraculous is dead-end state transition. When s is not start state, it satisfies $Post(s) \neq \emptyset$ but $Pre(s) = \emptyset$. If two navigation sequences have connection relation, the end state in one navigation sequence should have post succeed states set including the start state in the other navigation sequence. The navigation connection handling time-out implements service redirection. For instance, in online shopping system, if user is re-entered in case of disconnection, un-finished orders can be recommended to continue.

Definition 6 Navigation Connection Operation. Given navigation connection operation symbol $^\cap$ and two navigation paths $t_1 =< s_1,s_2,\ldots,s_\mathrm{i} >$, $t_2 =< s_\mathrm{j},s_{\mathrm{j}+1},\ldots,s_\mathrm{n} >$. Their connection leads to following types.

1) If $s_\mathrm{j} \in Post(s_\mathrm{i})$ or $s_\mathrm{i} \in Pre(s_\mathrm{j})$, then the navigation connection operation is to joint these two path as one navigation sequence

$$t_1{}^\cap t_2 =< s_1,s_2,,\ldots,s_\mathrm{i},s_\mathrm{j},s_{\mathrm{j}+1},\ldots,s_\mathrm{n} > .$$

As result, new sequence t' replaces t_1 and t_2 in the navigation sequence set that $NSS' = \mathrm{NSS} \downarrow \{t_1,t_2\} \cup \{t'\}$ using projection symbol $\downarrow$.

2) If $s_j \in Post(s_i)$ and $s_i \notin Pre(s_j)$, then the navigation connection operation returns *null* that $t_1 {}^{\cap} t_2 = \emptyset$ since their intersection is empty. The empty connection shows that t_1 and t_2 are independent.

3) Otherwise $t_1 \cap t_2 \neq \emptyset$, the navigation connection operation generates a result that $\exists x \in t_1 \cap t_2$ combines two new navigations

$$< head(t_1,x), x, tail(t_2,x) >, and < head(t_2,x), x, tail(t_1,x) >$$

where truncation function *head(t,x)* and *tail(t,x)* are navigation sequences before or after at sharing state *x*, respectively.

For example, $t_1 =< login, order >$ and $t_2 =< payment, success >$ can be connected as a new navigation $t' = t_1 {}^{\cap} t_2 =< login, order, payment, success >$ where the condition is $payment \in Post(order)$ or $order \in Pre(payment)$. Given another new navigation sequence $< login, shoppingcart, payment, failed >$. The two new navigations are $< login, shoppingcart, payment, success >$ and $< login, order, payment, failed >$. Thus, more possible navigation sequences can be extracted when their same states are identified.

The behaviors of user group *VeC* will display as a same and unique model because of their similar business behaviors. Considering state dependency, the Greedy Based Model Constriction Algorithm (*GB-MCA*) is introduced to implement functional modeling. As Table.2 shown, algorithm input is navigation sequence set NSS=$\{\pi_1, \pi_2, \pi_3, \ldots\}$ and output is a user navigation model which is returned in the form of finite state machine UNM model. Initially, *S* is an empty set. When no suitable solution can be available, Definition 5 and 6 are applied to analyze navigation sequence set NSS, where function $NCO(NSS)$ is the navigation connection operation and function $PPSucceed(S,x)$ is the *yes* or *no* judgment based on Pre/Post succeed states. If no intersection exists in these navigation sequences, the construction produce is terminated.

Definition 7 User Navigation Model. User-oriented service process reflected by user navigation behaviors in Web application are formalized based on FSM model [6, 7]. It is formally defined as UNM ::= (S, s_0, R, AP, L), that,

1) *S* is a set of finite states. Each state represents a service entity which is worked in logic Web page that users have visited.

2) $s_0 \in S$ is initial state, mainly a start state corresponding to index page of Web application.

3) $R \subseteq S \times S$ is a set of finite page links in accord with user visited paths, which are user navigations during Web service invocation.

4) $L(S) \rightarrow 2^{AP}$ is a mapping function that labels each location with a power set of atomic propositions true. *AP* is a set of atomic propositions.

The UNM model is a directed graph where each state is denoted by dot, and each transition is a connection with a directed arrow showing user access intents. Since each state is mapped to a service and each transition is corresponded to a user interaction behavior, the user navigation model UNM is seemed as service process model at functionality level.

Table 2: Greedy Based Model Constriction Algorithm

```
Input: Navigation Sequence Set NSS={$\pi_1, \pi_2, \pi_3, \ldots$}
Output: formal model UNM based on FSM
GreedyModelConstriction (NSS)
{
    S=$\emptyset$
    while (not solution(S))
    {
        x=NCO(NSS);
        if feasible PPSucceed (S, x) {
            S = S + x;
            NSS = NSS -{x};
        }
    }
}
```

Definition 8 Navigation Trace. The navigation trace of UNM is a visit path that $\pi^t =< s_1, s_2, s_3, \ldots s_n >$ satisfying $\forall i \in [1, n-1] \bullet (s_i, s_{i+1}) \in R$. Given UNM model *m* and at least there two traces $|Trace(m)| > 2$ are connectable based on navigation connection operation.

1) If $TM = \{t | \exists t \in \text{Trace}(p), \forall s \in t \bullet (\forall t' \in \text{Trace}(\pi) \downarrow t, \forall s' \in t' \bullet L(s) \cap L(s') = \emptyset)\}$, then these traces are called as margins.

2) If $TC = \{t | \exists t \in \text{Trace}(\pi), \exists s \in t \bullet (\exists t' \in \text{Trace}(\pi) \downarrow t, \exists s' \in t' \bullet L(s) \cap L(s') \neq \emptyset)\}$, then these traces are called as cores.

Since user group behaviors include *VsC*, *VeC*, and *PlC*, UNM model optimization and refinement should prune margin traces generated by *VsC* and *PlC*. Their navigation behaviors are meaningless. Each page state should be identified for the core activity according to business logic for distinguishing cores traces. In fact, it is hard to model all possible navigation paths because the number of dynamic interactions and the personalized pages generated by different Web users may be huge or even infinite. Thus, the user navigation model we proposed is constructed on the fly when user surfs on Web. Based on navigation sequence set, navigation trace and connection operation, we proposed an On-The-Fly modeling method to user navigation model construction.

Definition 9 On-The-Fly Modeling for User Navigation Model. Given a UNM model $m = (S, s_0, R, AP, L)$ and new navigation trace $t =< s_1, s_2, s_3, \ldots s_n >$. Their synthesis model $UNM ::= (S', s'_0, R', AP', L')$ is built on the fly as follow

1) $S' = S \cup \{s_1, s_2, s_3, \ldots s_n\}$

2) $s'_0 = s_0$ if $s_0 \neq \emptyset$; $s'_0 = s'$ if $\exists s \in t \bullet s_0 = \emptyset \wedge s$ is the index page.

3) $R' = R \cup \{r | \exists s, s' \in t \bullet (s, s')$ is a page link in t and $t \in TC\}$

4) $L' = L \cup L(t)$, where $L(t)$ is set of mapping functions on navigation trace t.

5) $AP' = AP \cup AP(t)$, where $AP(t)$ is set of atomic propositions in navigation trace t.

3.2 Case Study

Service event logs in Web server consist of run-time information and access information. Service process mining from event logs for dynamic user navigations construction is way to comprehend Web service behaviors. Few literatures have studied Web log data for service process modeling, especially automata modeling. Ricca [8] developed a Web analysis tool ReWeb, which analyzes the web page and hyperlink of Website to generate UML representing Web page navigation and object state chart. Lucca and Tramontana [9] designed a tool WARE which uses reverse engineering on Web application for generating UML class chart and sequence chart, from two aspects that are coarse granularity and fine granularity level. These static methods aims to design-time verification, which is not suitable to model dynamic service behaviors.

For run-time log, it saves exception during service execution. For access log, it saves access information, such as access time, IP address, visiting resources, and so on during user access. The widely used logs format can be divided into two types: One is NCSA format in Apache server, and the other is W3C format in IIS server. In this paper, Apache server integrated with AXIS 2.0 is employed to support service deployment, running and debugging since it is open source. As Table.3 shown, it is a fragment of event logs at Web server, which is outputted from Apache server. The *remotehost* column is labeled with IP address, by which it can distinguish user role. The *authuser* column records visitor identity, checking whether they are the same user or not. The *status* column is response returned from server, where 1xx is the continual messages, 2xx is the successful request, and 3xx is the request redirection, as well as the HTTP error code [10] that 4xx is the end user error and 5xx is the server error. The *bytes* column describes bytes for each requests transmission. The *referrer* column shows the pre-service. If current request has a pre-link, this information will not be blank. Otherwise, it is a start Web page which means a new service without any pre-link.

To output service event logs, the configuration file server.xml in Tomcat should be modified as follows

$$< Value \quad className = \text{"}org.apache.catalina.valves.AccessLogValue\text{"}$$
$$directory = \text{"}logs\text{"} \quad prefix = \text{"}localhost\_access\_log.\text{"} \quad Suffix = \text{"}.txt\text{"}$$
$$pattern = \text{"}commom\text{"} \quad resolveHosts = \text{"}false\text{"} >$$

The expression *common*="%h, %u, %t, %r, %s, %B, %U" is format setting, where %h corresponds to remotehost, %u corresponds to authuser, %t corresponds to time, %r corresponds to URL, %s corresponds to status, %B corresponds to bytes, %U corresponds to referrer. For instance, user Jean visits the service software at remote host 201.124.225.77. The login service is an initial state. Due to time out occurs at [2011-10-25 20:20:14], the redirection operation is executed (referring to state code 324). While at [2011-10-25 20:16:41], there exists service failures (referring to state code 404).

User in same group will have similar interesting behaviors. The valuable customers (*VeC*) behaviors are easy to model rather than construct all users' navigation behaviors because of state space explosion problem. The *referrer* column displays different URLs since different parameters make same service as different Web pages. To address this issue, the method implemented in tool Tansuo [11] is used to analyze *URL* and *referrer* parameter to model user navigation behaviors, which is a two-stage strategy, mainly the abstract model and combination strategy. Given two URLs $u_1 = \text{"}search.do?Cat = book\text{"}$ and $u_2 = \text{"}search.do?Cat = Tshirt\text{"}$. They are treated as an abstract service $u' = \text{"}search.do?Cat\text{"}$, representing the unique searching service. Thus, other parameters in searching service will be merged as a final abstract service. They behave like similar navigation structures.

From Table.3, following services are extracted that the login service *Login*, the book searching service *Search*, the shopping cart listing service *List*, Chinese book displaying service *ChineseIntroduce*, English book displaying service *EnglishIntroduce*, the order service *Order*, the online payment service *Online*, the credit card payment service *CreditCard*, and the notification service *Notify*. From parameters *URI* and *Referrer* in service logs, the *GB-MCA* algorithm and Navigation Trace method are used to extract service data to separate the margin traces from core traces. The navigation graph shown in Figure.2 is transformed from Table.3 where four effective navigation sequences are as follows,

$$< login, serach, list, chinsesIntroduce, order, creditCard, success >,$$
$$< login, serach, list, englishIntroduce, order, creditCard, success >,$$
$$< login, serach, list, chinsesIntroduce, order, online, success >,$$
$$< login, serach, list, englishIntroduce, order, online, success > .$$

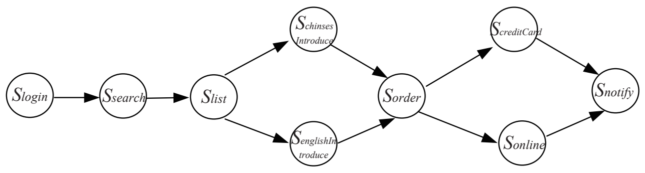

Fig. 2: The example of UNM model

According to formal definitions, the UNM model is as follows:

1) $S = \{S_{login}, S_{serach}, S_{list}, S_{chinsesIntroduce}, S_{englishIntroduce}, S_{online}, S_{creditCard}, S_{order}, S_{notify}\}$

2) $s_0 = login$

Table 3: Example of Service Log in Server

$< remotehost >, < authuser >, < time >, < URL >, < status >, < bytes >, < referrer >$
201.124.225.77, Jena , 2011-10-25 20:14:00, login.do?name=Jena&pass=31h12, 206, 56,
201.124.225.77, Jena , 2011-10-25 20:15:12, search.do?Cat=book, 206, 1342, login.do?name=Jena&pass=31h12
201.124.225.77, Jena , 2011-10-25 20:16:16, list.do?type=all, 206, 34212, search.do?Cat=book,
201.124.225.77, Jena , 2011-10-25 20:16:41, englishIntroduce.do?ID=123, 404, 1453, shoppingCart.do?select=all
201.124.225.77, Jena , 2011-10-25 20:16:44, chinsesIntroduce.do?ID=123, 206, 6532, list?type=all
201.124.225.77, Jena , 2011-10-25 20:20:14, login.do?ID=123, 324, 6532, chinsesIntroduce.do?ID=123
201.124.225.77, Jena , 2011-10-25 20:20:32, englishIntroduce.do?ID=123, 206, 1453, shoppingCart.do?select=all
201.124.225.77, Jena , 2011-10-25 20:22:03, order.do?ID=100001, 206, 1526, englishIntroduce?ID=123
201.124.225.77, Jena , 2011-10-25 20:23:43, order.do?ID=100242, 206, 451, chinsesIntroduce.do?ID=123
201.124.225.77, Jena , 2011-10-25 20:28:28, online.do?ID=100242, 206, 784, order.do?ID=100242
201.124.225.77, Jena , 2011-10-25 20:34:11, creditCard.do?ID=100001, 206, 1875, order.do?ID=100001
201.124.225.77, Jena , 2011-10-25 20:44:17, notify.do?state=succ, 206, 8344, creditCard.do?ID=100001
201.124.225.77, Jena , 2011-10-25 20:44:58, notify.do?state= fail, 206, 8764, online.do?ID=100242

3) R $= \{(S_{login}, S_{serach}), (S_{serach}, S_{list}), (S_{list}, S_{chinsesIntroduce}), (S_{list}, S_{englishIntroduce}), (S_{chinsesIntroduce}, S_{order}), (S_{englishIntroduce}, S_{order}), (S_{order}, S_{online}), (S_{order}, S_{creditCard}), (S_{creditCard}, S_{notify}), (S_{online}, S_{notify})\}$

4) $L(S_{login}) = \{login = true\}, L(S_{serach}) = \{serach = true\}, L(S_{list}) = \{list = true\}, L(S_{chinsesIntroduce}) = \{chinsesIntroduce = true\}, L(S_{englishIntroduce}) = \{englishIntroduce = true\}, L(S_{online}) = \{online = true\}, L(S_{creditCard}) = \{creditCard = true\}, L(S_{order}) = \{order = true\}, L(S_{notify}) = \{notify = true\}$

4 Non-functional Specification

The common service process modeling neglects the dynamic QoS characteristics. The functional behaviors model UNM is a so-called initial process model. Both the probability and time constraints of service are expected for model enhancement purpose. First, the transition between different states depends on users access frequency, where the transition probability of user navigation behaviors can be calculated by the frequency conversion method via filtering non-effective sessions. Thus, the transition probability matrix is proposed to figure out the transition probability. Second, service is a black-box component. Service interface specifies not only how the service interacts with its environment but also time constrains. It is called as guards of transition, which specifies the clock constraints for input and output interface. Only if the clock condition is fulfilled the related transition can be taken; otherwise, the transition is disabled. Therefore, In following paragraphs, the time constrains pair is proposed to describe the clock constraints. We will introduce how to add time constrains and probabilistic behaviors to UNM model.

4.1 Probability Extension

Definition 10 Transition Probability Matrix. The transition probability matrix is a (n+1)*(n+1) two-dimension matrix. $\forall i, j \leq n$, each cell M_{ij} gives a transition probability between service s_i and s_j where $M_{ij} \in [0,1]$. If M_{ij}=0, there is no transition between service s_i and s_j. If M_{ij} =1, the transition probability is 100% between service s_i and s_j.

To generate transition probability matrix, an extra state called a failure service S_{fail} is introduced to user navigation model. It works as statistics for the non-effective session from run-time logs, mainly HTTP error code.

First, the access frequency for each transition is generated from access log. As Figure.3 shown, in service S_{list} , it is forwarded to $S_{chinsesIntroduce}$ and $S_{englishIntroduce}$ with frequency of 3 and 6 times, respectively. However, it will be failure with frequency of 4 times. The total frequency triggered at service S_{list} is 13. The transition $(S_{chinsesIntroduce}, S_{list})$ means in service $S_{chinsesIntroduce}$ the user can not order any goods and he or she is forced back to shopping cart service S_{list}. In this process, there has service failures with 2 times in total 9 times service access.

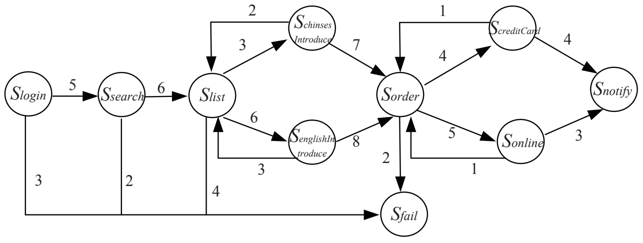

Fig. 3: A example of frequency-Extended UNM

According to above method, Table.4 shows a transition frequency matrix. There has a navigation behavior between service s_i and s_j. Each cell gives transition frequency. Each transition is annotated with a probability value, indicating the likelihood of its occurrence during service invocations. In addition, the failure frequency for service s_i is denoted in column s_{fail}, which is calculated by counting the number of unreachable page states. The HTTP error code, such as 400,401, 403, 404, 500, 501, and 502, will be used to analyze failures from access logs data.

Second, to obtain a transition probability matrix, the formula $c_{ij} = \frac{(s_i,s_j)}{(s_i,SUM)}$ converts the frequency matrix into probability matrix. The fraction (s_i,s_j) is transition frequency. The numerator (s_i, SUM) is total frequency including service failure phenomenon. Table.5 is the corresponding probability matrix transformed from Table.4. For example, the probability from service S_{list} to service $S_{chinsesIntroduce}$ and $S_{englishIntroduce}$ is 0.2307 and 0.4615, respectively. However, service S_{list} to service S_{fail} has probability of 0.3078.

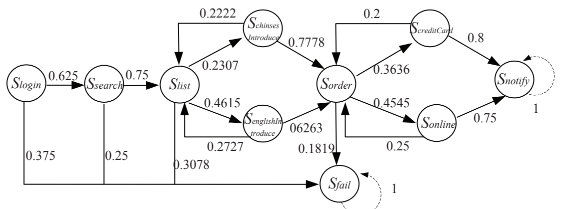

Fig. 4: A example of Probability-Extended UNM

When user number is increased, service access frequency will be different. As Figure 4 shown, probabilistic transitions are extended to UNM model using the conversion approach of probability matrix. Each transition is annotated with probability. The expression $Dist(s) = \{\mu_1,\mu_2,\ldots.,\mu_n\}$ where $\mu_1+\mu_2+\ldots.+\mu_n = 1$. Each state s is a set of all discrete probability distributions. For example, $Dist(s) = \{0.2307, 0.4615, 0.3078\}$ indicates service S_{list} to service $S_{chinsesIntroduce}$, $S_{englishIntroduce}$, and $S_{chinsesIntroduce}$ is 0.2307, 0.4615, and 0.3078, respectively.

4.2 Time Constrains Extension

Definition 11 Time Constrain Pair. Each transition under time constrain pair $[t_{min},t_{max}]$ describes that timed specifications t_{min} is the minimal time constrain which is used as guard condition of transition, and t_{max} is the maximal time constrain which is used as maximal duration of stay before changing state .

A finite set X is introduced to denote clocks. The element $x \in X$ is a clock variable mapping to a time domain $R_{\geq 0}$. Clock value is defined as $v : X \to R_{\geq 0}$. For example, given $X = \{x,y,z\}, x = 0.5, y = 7.2$ and $z = 11$, the clock value $v(x) = 0.5, v(y) = 7.2, v(z) = 11$. The expression $v+t$ is the time increment of $t > 0$ for v, which can be rewritten as $(v+t)x = v(x)+t$, $\forall x \in X$, such as, $v(x) \in [t_{min},t_{max}]$ and $v(x)+t \in [t_{min},t_{max}]$ are called as a satisfied time value.

Time constrain pair is clock constraints over X, which is a conjunction of atomic constraints that

$$\zeta ::= x \sim c | x-y \sim c | \neg\zeta | \zeta \wedge \zeta$$

where $x,y \in X$ and $\sim \in \{<,>,\leq,\geq\}, c \in N$, e.g. $(x > 1) \wedge (y < 1)$.

The time stamp of clock is a click time concept when user visits service, under which Web page link must be executed without delaying. The minimal time constrain t_{min} requires that the corresponding navigation needs to be executed after user hits Web page. Otherwise, it is service failure. Thus, the clock valuation can be figured out by time differences between user click times.

Definition 12 The Minimal Time Constrain. Support three services are sequentially triggered with order $< s_A, s_B, s_C >$ at clock HT_A, HT_B, HT_C. The t_{min} of service s_c $\mathrm{ST_c} = \mathrm{HT_B} - \mathrm{HT_c}$ requires page hyperlink taken. When user navigations are triggered from multi-persons, the different users in *Pre-succeed* states $Pre(s_c) = \{s_1, s_2, \ldots, s_n\}$ call for considerations. Given each state s_i in $Pre(s_c)$ has m users $User(s_i) = \{\mathrm{HT}_{u1}, \mathrm{HT}_{u2}, \ldots, \mathrm{HT}_{um}\}$, and each users click time at pre-state is denoted as $\mathrm{HT}^{si}{}_{uj}$, then the minimal time constrain is

$$t^{si}{}_{min} = Min(\{HT^{Si}_{U1} - HT_{Sc}, HT^{Si}_{U2} - HT_{Sc}, .., HT^{Si}_{Un} - HT_{Sc}\})$$

The time-out session mechanism controls the service session and interrupt response. There are two types of session constrains which should be handled. The first one is Web page session constrain. The value denoted as *nv* can be captured by name-value from Web session. The second one is service interface constrain. The value denoted as *si* can be analyzed from Web Services Description Language (WSDL),i.e.,$<$ *wsrm* : *InactivityTimeoutMilliseconds* $= 600000/>$. The Web session *nv* should be larger than service interface constrain *si*; Otherwise, error occurs when the request has been sent to service but Web session close its invocation since its session will be expired. As result, there no corresponding interaction matches it whenever the response is sent back or not.

Definition 13 The Maximal Time Constrain. The maximal time constrain t_{max} is obtained from Web page session value settings and its embedded services interface constrains.

1) If both *nv* and *si* exist, then $t_{max} = nv$. It compares *nv* with *si* to make sure that Web session *nv* is larger than service interface constrain *si*.

Table 4: Transition Matrix based on Frequency

	s_{login}	s_{serach}	s_{list}	$s_{chinsesIntroduce}$	$s_{englishIntroduce}$	s_{online}	$s_{creditCard}$	s_{order}	s_{notify}	s_{fail}	SUM
s_{login}	-	5	-	-	-	-	-	-	-	3	8
s_{serach}	-	-	6	-	-	-	-	-	-	2	8
s_{list}	-	-	-	3	6	-	-	-	-	4	13
$s_{chinsesIntroduce}$	-	-	2	-	-	-	-	7	-	-	9
$s_{englishIntroduce}$	-	-	3	-	-	-	-	8	-	-	11
s_{online}	-	-	-	-	-	-	-	1	3	-	4
$s_{creditCard}$	-	-	-	-	-	-	-	1	4	-	5
s_{order}	-	-	-	-	-	5	4	2	-	-	11
s_{notify}	-	-	-	-	-	-	-	-	-	-	-

Table 5: Probability Matrix

	s_{login}	s_{serach}	s_{list}	$s_{chinsesIntroduce}$	$s_{englishIntroduce}$	s_{online}	$s_{creditCard}$	s_{order}	s_{notify}	s_{fail}
s_{login}	-	0.625	-	-	-	-	-	-	-	0.375
s_{serach}	-	-	0.75	-	-	-	-	-	-	0.25
s_{list}	-	-	-	0.2307	0.4615	-	-	-	-	0.3078
$s_{chinsesIntroduce}$	-	-	0.2222	-	-	-	-	0.7778	-	-
$s_{englishIntroduce}$	-	-	0.2727	-	-	-	-	0.6263	-	-
s_{online}	-	-	-	-	-	-	-	0.25	0.75	-
$s_{creditCard}$	-	-	-	-	-	-	-	0.2	0.8	-
s_{order}	-	-	-	-	-	0.3636	0.4545	0.1819	-	-
s_{notify}	-	-	-	-	-	-	-	-	-	-

2) If *nv* exists, while *si* is absent, then t_{max}= *nv*. First, the object specified by system is identified from the *name-value* pair information. For example, *session("uslogin")* represents time constrains of object *uslogin*. Then, the property *session.timeOut* from session settings gives the times constrain value.

3) If *si* exists, while *nv* is absent, then t_{max}= *si*. This scenario releases its invocation time at interface descriptions.

4) Otherwise, if service s has *Post-succeed* states $Pre(s_c) = \{s_1, s_2, \ldots, s_n\}$, the maximal value $t^{si}{}_{max}$ is selected to denote the maximal time constrain of service s.

$$_{max}::=\begin{cases} t_{max} = nv & \exists nv, \exists si \bullet nv > si \\ t_{max} = nv & \exists nv \\ t_{max} = si & \exists si \\ Max(\{t^{s1}{}_{max}, t^{s2}{}_{max}, \ldots, t^{sn}{}_{max}\}) & Pre(s_c) = \{s_1, s_2, \ldots, s_n\} \end{cases}$$

Considering response time, the service process supporting Web page navigation is feasible if the time constrain pair $[t_{min}, t_{max}]$ is satisfied. When the transition is not executed within maximal time constrain t_{max}, it will never be executed any more. When the transition is occurred before the minimal time constrain t_{min}, it will be ignored. If t_{max} or t_{min} is empty, the time constrain pair is open interval. For example, $[0, t_{max}]$ denotes the minimal time constrain is 0, under which transition has no delay. $[t_{min}, \infty]$ denotes no maximal time constrain. $[0, \infty]$ denotes no maximal and minimal time constrain.

5 Navigation-Oriented Quantitative Model

In this section, the probability and time constrain introduced above paragraphs are integrated into user navigation model. It is a hybrid system, called as navigation-oriented quantitative model NoQM, which contains not only the functional requirement but also the non-functional requirement.

Definition 14 Navigation-Oriented Quantitative Model. The NUM model extended with probability and time is a tuple NoQMs ::=(Loc,l_{init},X,inv,$prob$,AP,L), that,

(1) *Loc* is a finite set of locations. Each location is a service which corresponds to UNM's state.

(2) $l_{init} \in Loc$ is an initial location, which considers the index page.

(3) X is a finite set of clocks, and *Zones* are the set of clock constraints over clocks X, denoted *Zones(X)* as

$$\zeta ::= x \sim c | x - y \sim c | \neg\zeta | \zeta \wedge \zeta$$

where $x, y \in X$, $\sim \in \{<, >, \leq, \geq\}$, and $c \in N$.

(4) $inv : Loc \rightarrow Zones(X)$ is a set of all invariant conditions, mapping to the minimal time constrain t_{min}

(5) $prob \subseteq Loc \times Zones(X) \times Dist\left(Loc \times 2^X\right)$ is the probabilistic relation. For example, $(l, g, p) \in prob$ that $l \in Loc$ is the source location, $g \in Zones(X)$ is the guard on the clock value mapping to the maximal time constrain t_{max}, and $p \in Dist\left(Loc \times 2^X\right)$ is the probability which corresponds to the probability matrix.

(6) $L : Loc \rightarrow 2^{AP}$ is a labeling function, where AP is a set of atomic propositions.

The navigation-oriented quantitative model contains the functional behaviors and non-functional requirement. For functional behaviors, it is user navigation behaviors. For the non-functional requirement, it is extended with probability and time. Expression (l,g,p,r,l') is a transition where $(l,g,p) \in prob$, the clock set $r \in X$ needs to be reset to zero after transition execution, and $l \in Loc$ is the target location. The transition means that if clock valuation satisfies the guard condition, denoted by $v \triangleright g$, the transition will be engaged immediately. Then the probability of moving to the location l and resetting all of the clocks in r to 0 is given by $p(l,r)$. Note that, $p(l,$ r)=$\mu((l,$ v[r:=0])) where $(l,v[r:=0])$ is a state. In Figure.5, $(S_{list},x \geq 3,0.2307,x,S_{chineseIntroduce})$ shows that: 1) at location S_{list} the user can stay for maximum 3 clock units; 2) the transition has a probability of 0.2307; 3) after transition is completed, the clock x needs to be reset to 0.

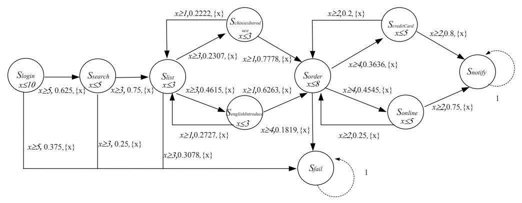

Fig. 5: A example of Navigation-Oriented Quantitative Model

Definition 15 (Semantics of NoQM). The semantics of a NoQM are formally defined as an infinite state DTMC (Discrete-time Markov Chains) [12, 13].

The DTMC::=$(S_p, s_{init}, \delta_p, L_p)$ is as follows,

1) Sates:$S_p = \{(l,v) \in Loc \times 2^X$ such that $v \triangleright inv(l)\}$, where a state is represented by a pair (l,v) that $l \in Loc, v \in \mathrm{V}(X), v \in inv(l)$.

(2) Initial state: $s_{init} = (l_{init}, \underline{0})$.

(3) Transition $\delta_\mathrm{p} \subseteq \mathrm{S_p} \times Dist\left(\mathrm{S_p}\right)$ maps each stare to discrete probability distributions $Dist\left(\mathrm{S_p}\right)$.

(4) Labelling: $L_\mathrm{p}(l,v) = L(l)$.

In semantics level of NoQM, the transition of link actions will be randomly selected to different reachable states under non-determinism choices. Here, probabilistic behaviors are introduced. The expression $Dist(S)$ is a set of all discrete probability distributions over S. $Dist(s) = \{\mu_1, \mu_2, \ldots, \mu_\mathrm{n}\}$, $\mu_1 + \mu_2 + \ldots + \mu_\mathrm{n} = 1$, where target state s is reached by making a probabilistic choice according to the distribution $\mu_i(s') > 0$. If probabilistic behaviors exist between particular states, then the probabilistic operator P, defined a probability space $\mathrm{Pr_s}(\mathrm{s}, \mathrm{s_1}, \mathrm{s_2}, \mathrm{s_3}, ..) = l_{init}(\mathrm{s}) \cdot \mathrm{P}(\mathrm{s}, \mathrm{s_1}, \mathrm{s_2}, \mathrm{s_3}, ..)$ over infinite paths (or for finite path, $\mathrm{Pr_s}(\mathrm{s}, \mathrm{s_1}, \mathrm{s_2}, \mathrm{s_3}, .., \mathrm{s_n}) = l_{init}(\mathrm{s}) \cdot \mathrm{P}(\mathrm{s}, \mathrm{s_1}, \mathrm{s_2}, \mathrm{s_3}, .., \mathrm{s_n})$), should be used to calculate the probability. Generally, $l_{init}(\mathrm{s}) = 1, \mathrm{P}(\mathrm{s_0}, \mathrm{s_1} \ldots) = \prod_{0 \leq i \leq n} \mathrm{P}(\mathrm{s_i}, \mathrm{s_{i+1}})$. Each probability of associated transition is multiplied as total probability, and $\mathrm{P}(\mathrm{s_i}, \mathrm{s_{i+1}})$ is probability of making a transition from state s_i to s_{i+1}. To measure the probability of desired property φ, $Prob(\mathrm{s_0}, \varphi) = \mathrm{Pr_{s0}}(\mathrm{S_{Path(s,\mathit{j})}})$ is defined on $\mathrm{Path}(\mathrm{s}, \varphi) = \{\omega \in \mathrm{Path}(\mathrm{s}) \, |\omega| = j\}$.

Definition 16 (NoQM transition). There are two type transitions for $(s,\mu) \in \delta_\mathrm{p}$ which follow restraints that,

1)Discrete transition is $((l,v), p, (l',v'))$ iff there exits $(l,g,p) \in prob$ which make (l,g,p,r,l') satisfied that $v \triangleright g$ and clocks Y be reset, and $v \triangleright inv(l)$. Thus, when $(l,v) \in S_p$, the transition probability is

$$\mu(l,v) = \sum_{Y \subseteq X \wedge v[Y=0] \wedge v'=v[X \setminus Y]} p(l',Y)$$

2)Time transition is $((l,v), t, (l,v+t))$ iff for all $t \leq t$ satisfy $v+t \triangleright inv(l)$ and $\mu(q, v+t) = 1$.

When one or more services participate in composition, Web service needs to compose NoQMs to form more complex service software. The other issue is multi-thread access mechanisms. It is unrealistic approach to try all the possible combinations exhaustively. We thus introduce the parallel composition behaviors using synchronous product method as follows.

Definition 17 The Parallel Composition of NoQMs. Let $\mathrm{NoQM_i} = (LOC_i, l_{i.\mathrm{init}}, X_i, inv_i, prob_i, \mathrm{AP}_i, L_i)$ where $i \in \{1,2\}$ and $X_1 \cap X_2 = \emptyset$. The parallel composition is also a quantitative model that $NoQM_1 || NoQM_2 = (Loc, l_{init}, X, inv, prob, AP, L)$, where,

1) $Loc = Loc_1 \times Loc_2$

2) $l_{\mathrm{init}} = l_{1.\mathrm{init}} \times l_{2.\mathrm{init}}$

3) $X = X_1 \cup X_2$

4) $inv(l,l') = inv(l) \wedge inv(l')$ where $l \in Loc_1$and $l' \in Loc_2$

5) $AP = AP_1 \cup AP_2$

6) $L(l,l') = L(l) \cup L(l')$ for all $(l,l') \in L_1 \times L_2$

7) for $(l',g',p') \in prob_1$ and $(l'',g'',p'') \in prob_2$, the probabilistic relation is $(l,g,p) \in prob$ where $g = g'' \wedge g''$and $p = p'' \wedge p''$

The composited model gives possible concurrency behaviors in which they collaborate with each other. The Loc is a product of locations between $NoQM_1$ and $NoQM_2$. The X is a union set of clocks from $NoQM_1$ and $NoQM_2$. The AP and L is a union set from each component respectively. But the inv is a join set of locations between $NoQM_1$ and $NoQM_2$ which makes the location compostable, as well as the concept g and p. The new composited model NoQM is still a navigation-oriented quantitative model. Hence it can be verified by quantitative model checking tools, such as UPPAAL-PRO/TIGA and PRSIM.

6 Relate Works

To the best of our knowledge, few literatures have studied the Web-based service application modeling. We give reviews on the major techniques and researches that are most closely related to our works.

Facing a growing market of Web services, the first important work for service verification is to give a formal model. The conventional techniques widely adopt Petri Nets, Automata and Finite State Machines. Martens [14] considered Web service behaviors as workflow module composed of a local process and a serial of interfaces, and then used Petri Nets to model workflow modules and their compositions. Fu et.al., [15] proposed a top-down approach based on guarded mealy automata to analyze Web services composition. They applied model-checking techniques to verify the conversation between two services and implemented a tool for the analysis of Web service conversation [16]. But these works focused on the high level behavior description to model functional behaviors of business logic as their prior jobs. The limitation is in absent of non-functional requirements, such as user satisfactions and QoS characteristics.

At software application level, typical QoS metrics usually consists of throughput, response time, cost, reliability, fidelity, etc. Yu et.al., [17] designed a broker-based architecture to facilitate the QoS-based services selection. The objective of service selection is to maximize the application-specific utility function under end-to-end QoS constraints. Zeng et.al., [18] presented a middleware platform which addressed the issue of services selection for their composition purposes. User satisfactions were expressed as utility functions over QoS attributes. However, there is no introduction about where and how the QoS characteristic is extracted or generated in these literatures. New interesting topics about quantitative verification have attracted many researchers attentions. For timed verification, Elabd et.al., [19]added time of business process into atom service and model its work protocols using automata, which were used to verify correctness, compatibility and replaceability of service interaction. Mounir et.al., [20]proposed a novel language WS-TEFSM (Web Service Timed Extended Finite State Machine) to describe timed BPEL, and used time contains pair to test service composition. Diaz et.al., [21]transformed WSCI-WSCDL model into timed automata for timed property verifications. Nawal et.al., [22] given a time-based compatibility analysis approach to service composition, especially time conditions formalization for the internal and external constraints. However, time specification is not enough to model non-functional service behaviors in Internet environment. For probability verification, Arbab et.al., [23] labeled Reo channel with random delay probability, and proposed a method to transform Reo channel into CTMC to carry out performance analysis. Yang et.al., [24] used Web services choreography language *Chor* to service composition where each interaction behavior is labeled with transition probability for further verification using supporting tool PRISM. Zhao et.al., [25]extended Web service choreography language WS-CDL with QoS, mainly time, cost and probability information. But, these approaches are often used at design phase. The time and probability to service quantitative modeling are based on assumption value. When the number of Web services increases, this problem is more critical to service modeling, beacuse service processes are usually on changing. Research in this area is still immature.

A possible alternative method is to use user navigation model, which formalizes user navigations between Web pages and user interactions when users access to Web applications. Different from above QoS researches, our quantitative user navigation model is extended with probability and time statements. The main purpose is to give a base formal model to implement quantitative verification. Many published literatures have studied Web application modeling. For example, HMBS (Hypertext Model Based on StateCharts) [26] is a navigation-oriented model for hypertext based on UML StateCharts. FSM (Finite State Machine) is also suitable to model the behavior of Web applications [27,28]. In graph view, nodes represent the pages, and arcs represent the transition from one node to another. Donini et.al., [29] used Web Application Graph (WAG) to support for automated verification of the UML design of Web application, a proposed mathematical model that partitioned the usual Kripke structure into windows, links, pages and actions. Han et.al., [30] used StateCharts to formally model adaptive navigation, and shown how important properties of a navigation model were verified using existing model-checking tools. However, these models above give relative static models without taking dynamic features into account. Furthermore, they also do not involve the non-functional requirement modeling because time and probability are on the move. To this end, this paper proposes a novel approach to dynamically construct quantitative model for Web service. The navigation-oriented quantitative model is to formalize functional behaviors and non-functional specifications, including probabilistic behaviors and time constrains, which are generated from running data in event logs using probability matrix and time constraint pair method.

7 Conclusions

Web Service provides a flexible way to address interoperability issues [31] for software applications in e-business and scientific computing domains. However, it is important to monitor and analyze services behavior. The main contribution of this paper is to formalize user navigation-oriented behaviors to model Web service, which is suitable to implement quantitative verification. The quantitative information is characterized by time and probability. First, it introduces user navigations of Web application and discusses how to generate timed probabilistic specifications from event logs. Then, a formal hybrid model, called as navigation-oriented quantitative model NoQM, is proposed to describe Web service. After that, the parallel composition of NoQM models is introduced to specify concurrent behaviors. Our approach is a dynamic process. Thus, it has potential application prospects in formal verification of service-oriented software. One of our ongoing works is to consider turning our method into concrete business applications.

As future researches, how to generate the verification property will be studied focusing on UML-based threat model, such as safety and reliability formulae. State space in parallel composition may be huge when components taken part in services composition are rapidly increased. Thus, how to alleviate the state space explosion issues is another hot topic. We plan to use abstractions refinement technique in next step.

Acknowledgement

This paper is supported by The Natural Science Foundation of Shanghai Municipality of China under Grant No. 15ZR1415200, Foundation of Science and Technology Commission of Shanghai Municipality under Grant No. 14590500500, Young University Teachers Training Plan of Shanghai Municipality under Grant No. ZZSD13008, and National Natural Science Foundation of China under Grant No. 61170044, and 61073050.

References

[1] Papazoglou MP, Georgakopoulos D. Service-oriented computing. Communication ACM, 46(10), 25-28 (2003)

[2] Scott E. Sampson and Craig M. Froehle. Foundations and Implications of a Proposed Unified Services Theory. Poduction and Operations Management, 15(2), 329-343 (2006)

[3] Lin Chuang,Hu Jie, and Kong Xiang Zhen. Survey on models and Evaluation of Quality of Experience(in chinese). Chinese Journal of Computers, (35)1, 1-15 (2012).

[4] Deren Che, Minghua Shi, Zhihang Wu. The architecture of electronic commerce(in chinese). Higher Education Press, Beijing (2004)

[5] Mounir Lallali, Fatiha Za?di, Ana R. Cavalli. Timed Modeling of Web Services Composition for Automatic Testing. In Proceedings of the third International IEEE Conference on Signal-Image Technologies and Internet-Based System table of contents, pp.417-426, IEEE Computer Society (2007)

[6] Daniel A. Menasce and Virgilio A.F. Almeida. Scaling for E-Business Technologies, Models, Performance, and Capacity Planning. Prentice Hall Press (2000)

[7] Chen Shengbo, Miao Huaikou, Qian Zhongsheng. In Proceedings of the 15th Asia-Pacific Software Engineering Conference(APSEC). pp.351-358,IEEE Computer Society (2008)

[8] Filippo Ricca and Paolo Tonella. Web site analysis: structure and evolution. In Proceedings of the International Conference on Software Maintenance. pp.76-86, San Jose, California, USA(2000)

[9] Porfirio Tramontana. Reverse Engineering Web Applications. In Proceedings of the 21st IEEE International Conference on Software Maintenance, pp.705-708, Budapest, Hungary, IEEE Computer Society (2005)

[10] Jeff Tian, Sunita Rudraraju,Zhao Li. Evaluating Web software reliability based on workload and failure data extracted from server logs. IEEE Transactions on Software Engineering, 30(11), 754-769 (2004)

[11] Wenhua Wang, Yu Lei, Sampath S, Kacker R, Kuhn R and Lawrence J. A combinatorial approach to building navigation graphs for dynamic web applications. In Proceedings of the 25th IEEE International Conference on Software Maintenance, pp.211-220, Budapest, Hungary, IEEE Computer Society (2009)

[12] Marta Kwiatkowska, Gethin Norman, Roberto Segala, Jeremy Sproston. Automatic Verification of Real-time Systems with Discrete Probability Distributions. Theoretical Computer Science, 282,101-150(2002)

[13] small Pedro R. D Argenio, Bertrand Jeannet, Henrik Ejersbo Jensen, and Kim Guldstrand Larsen. Reduction and Refinement Strategies for Probabilistic Analysis. In Proceedings of the Second Joint International Workshop on Process Algebra and Probabilistic Methods, Performance Modeling and Verification, pp.57-76, Copenhagen,Denmark, IEEE Computer Society (2002)

[14] Martens A. On compatibility of Web services. Petri Net Newslett, 65, 12-20, (2003)

[15] Xiang Fu, Tevfik Bultan and Jianwen Su. Analysis of interacting BPELWeb services. In Proceedings of international World Wide Web conference, pp.621-630, IEEE Computer Society (2004)

[16] Richard Hull, Jianwen Su. Tools for composite Web services: a short overview. SIGMOD Record, 34(2), 86-95, (2005)

[17] Yu Tao, Zhang Yue, Lin Kwei-Jay. Efficient Algorithms for Web Services Selection with End-to-End QoS Constraints. ACM Transactions on the Web, 1(1), 1-26, (2007)

[18] Zeng Liangzhao, Benatallah Boualem, Ngu Anne H.H., Dumas Marlon, Kalagnanam Jayant, Chang Henry. QoS-aware middleware for Web services composition. IEEE Transactions on Software Engineering, 30(5), 311-327, (2004)

[19] Emad Elabd, Emmanuel Coquery, and Mohand-Said Hacid. Compatibility and Replaceability Analysis of Timed Web Services Protocols. In Proceedings of the Second International Conference on Computer and Electrical Engineering, Vol.2, pp.15-19 (2009)

[20] Mounir Lallali, Fatiha Zaidi, and Ana Cavalli. Timed Modeling of Web Services Composition for Automatic Testing. In Proceedings of the 3rd International IEEE Conference on Signal-Image Technologies and Internet-Based System, 417-426, Shanghai, China (2007)

[21] Gregorio Diaz, Juan-Jose Pardo, Maria-Emilia Cambronero, Valentin Valero, and Fernando Cuartero. Verification of Web Services with Timed Automata. Journal Electronic Notes in Theoretical Computer Science, 157, 19-34 (2005)

[22] Nawal Guermouche, Olivier Perrin and Christophe Ringeissen. Timed Specification For Web Services Compatibility Analysis. Electronic Notes in Theoretical Computer Science, 200(3), 155-170 (2008)

[23] Farhad Arbab, Tom Chothia, Rob van der Mei, Sun Meng, YoungJoo Moon, and Chrtien Verhoef. From Coordination to Stochastic Models of QoS Coordination Models and Languages, 5521, 268-287, Lecture Notes in Computer Science (2009)

[24] Hongli Yang, Liang Zhou, Kang He, Chen Deng, Xiangpeng Zhao, and Zongyan Qiu. A probabilistic QoS model-checking for dynamic routing protocol. In Proceedings of the 10th International Conference on Quality Software, pp.441-448, Zhangjiajie, China, IEEE Computer Society (2010)

[25] Zhao Xiangpeng, Cai Chao, Yang Hongli, and Qiu Zongyan. A QoS view of web service choreography. In Proceedings of the IEEE International Conference on e-Business Engineering, pp.607-611, Hong Kong, China, IEEE Computer Society (2007)

[26] Santos Turine Marcelo Augusto, Ferreira de Oliveira Maria Cristina, Masiero Paulo Cesar. A Navigation-oriented Hypertext Model Based on Statecharts. In Proceedings of the ACM Conference on Hypertext, pp.102-111, Southamption, UK, (1997)

[27] Anneliese A. Andrews, Jeff Offutt, and Roger T. Alexander. Testing Web Applications by Modeling with FSMs. Software and Systems Modeling, 4(3), 326-345 (2005).

[28] Joumana Dargham, and Sukaina Al-Nasrawi. FSM Behavioral Modeling Approach for Hypermedia Web Applications: FBM-HWA Approach. In Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services, pp.199-204, Washington, DC, USA, IEEE Computer Society (2006).

[29] Francesco Maria Doninia, Marina Mongiellob, Michele Rutac, and Rodolfo Totarod. A Model Checking-based Method for Verifying Web Application Design. Electronic Notes in Theoretical Computer Science, 151(2), 19-32 (2006).

[30] Minmin Han, and Christine Hofmeister. Modeling and verification of adaptive navigation in web applications. In Proceedings of the 6th international Conference on Web Engineering, Vol.263, pp.329-336. ACM, New York (2006)

[31] Qi Yu, Xumin Liu, Athman Bouguettaya, Brahim Medjahed. Deploying and managing web services: issues, solutions, and directions. The VLDB Journal, 17,537-572 (2008)

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.

Huaikou Miao is currently a professor in Computer Engineering and Science at Shanghai University, China. His research interests include formal methods and software engineering.

Hongwei Zeng Ph.D. is currently a senior research fellow in Computer Engineering and Science at Shanghai University,China. His research interests include formal methods and software testing.

Yonghua 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.