TELKOMNIKA, Vol. 11, No. 8, August 2013, pp. 4385~4392
e-ISSN: 2087-278X

Formal Modeling of Trust Web Service Composition using Pi-calculus

Bensheng YUN
Department of Mathematics and Information Science, Zhejiang University of Science and Technology, Zhejiang 310023, P.R. China
e-mail: yunbsh@gmail.com

Abstract

To enhance the credibility of Web service composition, Pi-calculus based formal modeling of trust Web service composition is proposed. Trust Web service composition is firstly defined abstractly; then Pi-calculus is used to depict structure and internal interaction of Trust Web service composition, the mapping relation between trust entity and Pi-calculus is provided. Automatic reasoner MWB is adopted to analyze and reason the Trust Web service composition system, which is aimed at finding and correcting the faults before the implementation of trust authentication of Web service composition. It thus meets the users' demands on trust quality effectively.

Keywords*: formal model, trust web service composition, pi-calculus*

1. Introduction

Web service has become a most important computing resource, however, the network environment is dynamic, distributed, open, uncertain, and so on. These features may result in many uncertain factors, such as the uncertainty of behavior. Therefore, it is badly in need of a secure and reliable management tool. A valid method for the problems above is to evaluate trust value of the network entity and establish trust mechanism for Web service.

Trust is a concept that derives from sociology and has not yet formed a unified definition in the computer field [1]. Trust involves many factors, but it is generally acknowledged: Trust ≈ Security + Reliability [2]. Some computing methods of trust value have been proposed in previous works, such as probability and statistic based method [3, 4, 5], fuzz based method [6].

Web service composition is the main interaction system of Web service. During the composition process, interactive Web services need to evaluate trust value each other. When each Web service achieves satisfaction from trust value, the Web service composition can be implemented further. While with more complex function the users need, the scale of Web service composition is improved continuously. So, trust authentication of Web service composition is becoming more and more compex and error-prone. Thereby, it will affect the credibility of software which based on Web service composition technology [7].

In recent years, some formal methods are used to analyze and verify Web service composition, such as Pi-calculus [8], Petri Net [9], and so on. However, these methods are lack of describing and analyzing trust authentication. It is necessary to analyze and verify trust authentication of Web service composition, so that the errors can be found and corrected before the implementation of Web service composition.

Pi-calculus owns the powerful behavior equivalence theory, and Pi-calculus itself is also in constant development, such as Pi^{+}-calculus [10]. Therefore, Pi-calculus is used as a tool for modeling.

2. Pi-calculus

Pi-calculus is a process algebra for specifying and reasoning about concurrent systems. Although we refer to [11] for a detail description of Pi-calculus, a brief introduction to its syntax, transition relations and behavior equivalence theory is given as follows.

Definition 1 (Pi-calculus) The *processes* of the Pi-calculus are given respectively by

Received January 15, 2013; Revised May 12, 2013; Accepted May 22, 2013

$P ::= \mathbf{0} \mid \pi.P \mid P+Q \mid P|Q \mid \nu\tilde{z}\ P \mid !P \mid A(\tilde{x})$

$\pi ::= \bar{x}\tilde{y} \mid x(\tilde{y}) \mid \tau \mid [x = y]\ \pi.P$.

(1) $\mathbf{0}$ is inaction; it is a process that can do nothing.
(2) The *prefix* $\pi.P$ has a single capability, expressed by π; the process P cannot proceed until that capability has been exercised.

The *output prefix* $\bar{x}\tilde{y}.P$ can send the name tuple $\tilde{y}$ via the name x and continue as P.

The *input prefix* $x(\tilde{y}).P$ can receive any name tuple via x and continue as P with the received name substituted for $\tilde{y}$. The *unobservable prefix* $\tau . P$ can evolve invisibly to P. τ can be thought of as expressing an internal action of a process.

The *match prefix* $[x = y]\ \pi.P$ can evolve as $\pi.P$ if x and y are the same name, and can do nothing otherwise.

(3) The capabilities of the *sum* $P+Q$ are those of P together with those of Q. When a sum exercises one of its capabilities, the others are rendered void.
(4) In the *composition* $P|Q$, the components P and Q can proceed independently and can interact via shared names.
(5) In the restriction $\nu\tilde{z}\ P$, the scope of the name tuple $\tilde{z}$ is restricted to P.
(6) The *replication* $!P$ can be thought of as an infinite composiotion $P \mid P \mid \ldots$, replication is the operator that makes it possible to express infinite behaviours.
(7) The *process identifer* $A(\tilde{x})$, each process identifer can be defined as $A(\tilde{x}) = P$.

Definition 2 (Transition relations) The *transition relations* are defined by the rules in Table 1.

Definition 3 (Sequential composition) The *sequential composition* $P;Q$ means that 'when P finishes, Q starts'. Set $\bar{d}$ be the last action of process P,

$$P;Q = \nu d\ (\bar{d}.P \mid d().Q)$$

Definition 4 (Weak equivalence) Let $\Re$ be a binary relation over processes, then $\Re$ is said to be a *weak simulation* if, whenever $(P, Q) \in \Re$,

If $P \overset{e}{\Rightarrow} P'$, then $\exists Q'$ s.t. $Q \overset{e}{\Rightarrow} Q'$ and $(P', Q') \in \Re$.

Where $e = \alpha_1\alpha_2 \cdots \alpha_n$, $\overset{e}{\Rightarrow} = \overset{\tau}{\Rightarrow} \xrightarrow{\alpha_1} \overset{\tau}{\Rightarrow} \cdots \overset{\tau}{\Rightarrow} \xrightarrow{\alpha_n} \overset{\tau}{\Rightarrow}$, $\overset{\tau}{\Rightarrow} = \xrightarrow{\tau} \cdots \xrightarrow{\tau}$, the transitive reflexive closure of $\xrightarrow{\tau}$. $\Re$ is said to be a *weak bisimulation* if both $\Re$ and its converse are weak simulations. P and Q are called *weakly bisimilar*, *weakly equivalent* or *observation equivalent*, if there exists a weak bisimulation such that $(P, Q) \in \Re$, denoted by $P \approx Q$.

3. Trust Web Service Composition and Its Formal Model

3.1. Trust Web Service Composition

Trust is mutual, the identification of trust subject and trust object is relative, depending on their environment. In the service oriented network, the evaluation of trust value mainly depends on their own experience and the third party's recommendation. In theory, a trust relation can be established between any entities in the network, such as A and B, which is denoted by $TR(A,B)$.

Definition 5 (Trust Web Service Composition) In the service oriented network, *Trust Web Service Composition* can be defined as a three-tuple: $TWSC = <WS, CR, TR>$, where

(1) $WS = \{ WS_1, WS_2, \ldots, WS_n\}$ the set of trust entities, there are control relation and trust relation between entities;
(2) $CR = \{$*sequence*, *fork*, *parallel* $\}$ the set of basic control relations, as shown in Figure 1;
(3) TR the trust relation set between entities.

Table 1. The Transition Rules

$Act: \frac{-}{\pi.P \xrightarrow{\pi} P}$	$Match: \frac{\pi.P \xrightarrow{\pi} P}{[x=x]\pi.P \xrightarrow{\pi} P}$
$Sum_j: \frac{P_j \xrightarrow{\pi} P_j'}{\sum_{j\in I} P_i \xrightarrow{\pi} P_j'}$	$Res: \frac{P \xrightarrow{\pi} P'}{\nu\tilde{z}P \xrightarrow{\pi} \nu\tilde{z}P'}, \ (\tilde{z} \notin fn(P))$
$Par: \frac{P \xrightarrow{\pi} P'}{P \mid Q \xrightarrow{\pi} P' \mid Q}, \ (bn(\pi) \cap fn(Q) = \phi)$	
$Com: \frac{P \xrightarrow{\bar{x}\tilde{y}} P', Q \xrightarrow{x(\tilde{z})} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'\{\tilde{y}/\tilde{z}\}}$	$Id: \frac{P \xrightarrow{\pi} P'}{A \xrightarrow{\pi} P'}, \ (A \overset{def}{=} P)$

Table 2. Elements Mapping between TWSC and Pi-calculus

TWSC		Pi-calculus	
Web service		Process	
Operation		Action	
Message		Message	
Communication		Interaction(τ)	
CR	Sequence	;	Operator
	Fork	+	
	Parallel	|	

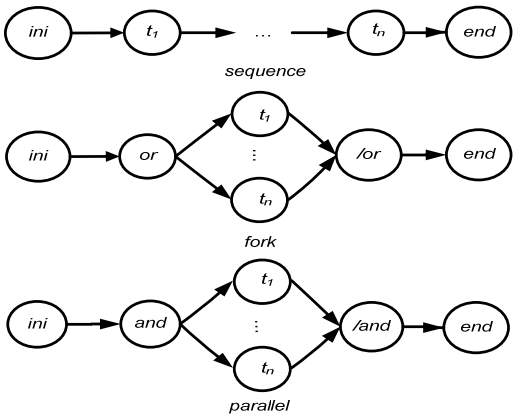

Figure 1. The Basic Control Relations (*CR*)

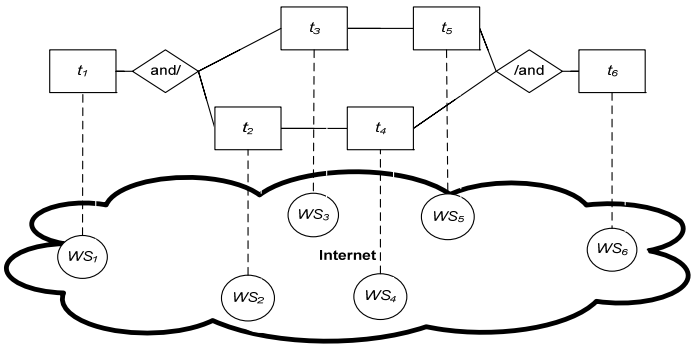

Figure 2. An Execution Plan of Trust Web Service Composition

An execution plan of trust Web service composition as shown in Figure 2, the set of trust entities in composition is $WS = \{WS_1, WS_2, WS_3, WS_4, WS_5, WS_6\}$, the set of basic control

relations is CR = {*sequence*, *and*}, and the trust relation set is TR = {$TR(WS_1,WS_2)$, $TR(WS_1, WS_3)$, $TR(WS_2, WS_4)$, $TR(WS_3, WS_5)$, $TR(WS_4, WS_6)$, $TR(WS_5, WS_6)$}.

3.2. Elements Mapping Between TWSC and Pi-calculus

According to the similarities between TWSC and Pi-calculus, the rule of correspondence from TWSC to the Pi- calculus is established, as shown in Table 2.

Each trust entity, i.e. basic Web service, is regarded as a process in Pi-calculus, the interaction between two trust entities is represented by τ action. The three control relations in composition: sequence, fork and parallel are mapping to ";", "+" and "|" respectively, the three operators in Pi-calculus. However, how to identify trust entity (process) contained in conposition, and how to identify the channel between the interactive trust entities. Therefore, three rules are proposed to identify process and channel.

Rule 1. In the trust Web service composition, one trust entity (atomic Web service) corresponds to one process.

Rule 2. In the trust Web service composition, two interactive trust entities share one channel at least logically.

Rule 3. In the trust Web service composition, allowing multiple small trust entities to be combined to form a bigger trust entity, and a bigger trust entity can be divided into several small trust entities.

3.3. Pi-calculus based Model of TWSC

Suppose that each task node in Figure 2 has two candidate Web services, as shown in Figure 3. According to Table 2, Rule 1 and 2, interaction diagram between processes in Figure 3 is presented in Figure 4.

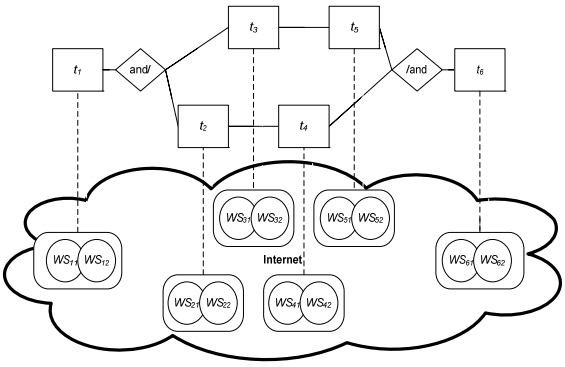

Figure 3. Trust Web Service Composition

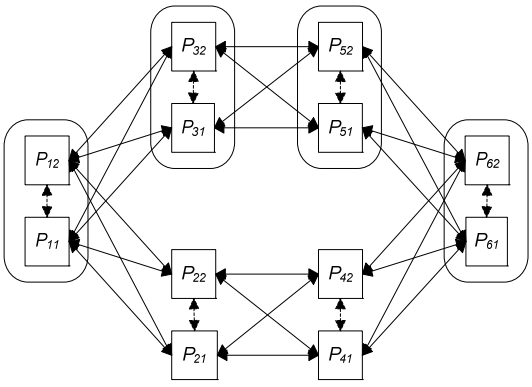

Figure 4. The Interaction of Trust Web Service Composition

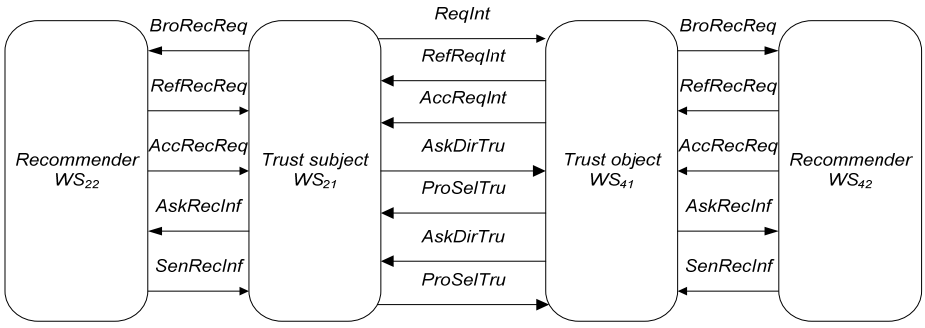

Figure 5. The Details of Trust Authentication

The trust certification between WS_{21} and WS_{41} in Figure 2 is taken for illustrate. Before implementation of WS_{21} and WS_{41}, it is need to evaluate each other's trust value. The evaluation process includes many steps of communication. In order to get accurate trust value, direct trust and recommendation trust are combined to evaluate trust value. The entity with the same functionality, such as WS_{22} and WS_{42}, can be used as recommender. WS_{22} has direct interaction experience with WS_{41} and evaluates the trust value of WS_{41} and recommends it to WS_{21}. As well as WS_{42} can recommends the trust value of WS_{21} to WS_{41}. When two trust entities satisfy mutual trust value, the two candidates will be chosen for further execution. The details of trust certification between WS_{21} and WS_{41} are illustrated in Figure 5.

(1) The messages in Figure 5 are interpreted as follows.
Messages from trust subject WS_{21}:
ReqInt: trust subject requests trust object to interact.
AskDirTru: trust subject asks the direct trust value of trust object.
ProSelTru: trust subject proposes its own direct trust value to trust object.
BroRecReq: trust subject broadcasts its request for recommendations of trust object.
AskRecInf: trust subject asks recommender trust value of trust object.
Messages from trust object WS_{41}:
AccReqInt: trust object accepts trust subject's request for interaction.
RefReqInt: trust object refuses trust subject's request for interaction.
AskDirTru: trust object asks the direct trust value of trust subject.
ProSelTru: trust object proposes its own direct trust value to trust subject.
BroRecReq: trust object broadcasts its request for recommendations of trust subject.
AskRecInf: trust object asks recommender trust value of trust subject.
Messages form recommender WS_{22} or WS_{42}:
AccRecReq: trust recommender accepts request for recommendation.
RefRecReq: trust recommender refuses request for recommendation.
SenRecInf: trust recommender provides recommendation.

(1) Pi-calculus based model of trust cetification in Figure 5.
According to Rule 1 and 2, Figure 5 can be coverted into the corresponding process graph, as shown in Figure 6. P, Q, R1 and R2 represent trust subject WS_{21}, trust object WS_{41}, recommender WS_{22} and WS_{42} respectively. P and Q share the channel x, P and R1 share the channel y, Q and R2 share the channel z.

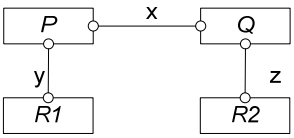

Figure 6. Process Graph

Trust subject P contains two concurrent processes P_1 and P_2.

$$P_1(a_1) = \overline{x} < ReqInt > .x(msg1).([msg1 = RefReqInt]P_1(a_1) + [msg1 = AccReqInt]$$
$$\overline{x} < AskDirTru > .x(msg2).[msg2 = ProSelTru]x(msg3).[msg3 =$$
$$AskDirTru]\overline{x} < ProSelTru > .P_1(a_1)).$$

$$P_2(a_2) = \overline{y} < BroRecReq > .y(msg4).([msg4 = RefRecReq]P_2(a_2) + [msg4 =$$
$$AccRecReq]\overline{y} < AskRecInf > .y(msg5).[msg5 = SenRecInf]P_2(a_2)).$$

$$P(a) = P_1(a_1) \mid P_2(a_2).$$

Where a_1 = {x, ReqInt, RefReqInt, AccReqInt, AskDirTru, ProSelTru}, a_2 = {AskRecInf, BroRecReq, RefRecReq, AccRecReq, SenReqInf, y}, $a = a_1 \cup a_2$.
Trust object Q contains two concurrent processes Q_1 and Q_2.

$$Q_1(b_1) = x(msg1).[msg1 = ReqInt](\bar{x} < RefReqInt > .Q_1(b_1) + \bar{x} < AccReqInt > .$$
$$x(msg2).[msg2 = AskDirTru]\bar{x} < ProSelTru > .\bar{x} < AskDirTru > .$$
$$x(msg3).[msg3 = ProSelTru]Q_1(b_1)).$$

$$Q_2(b_2) = \bar{z} < BroRecReq > .z(msg4).([msg4 = RefRecReq]Q_2(b_2) + [msg4 =$$
$$AccRecReq]\bar{z} < AskRecInf > .z(msg5).[msg5 = SenRecInf]Q_2(b_2))).$$

$$Q(b) = Q_1(b_1) \mid Q_2(b_2).$$

Where b1 = {x, ReqInt, RefReqInt, AccReqInt, AskDirTru, ProSelTru}, b2 = { BroRecReq, RefRecReq, AccRecReq, AskRecInf, SenReqInf, z}, $b = b_1 \cup b_2$.

Trust recommenders R1 and R2.

$$R1(c) = y(msg1).[msg1 = BroRecReq](\bar{y} < RefRecReq > .R1(c) +$$
$$\bar{y} < AccRecReq > .y(msg2).[msg2 = AskRecInf]\bar{y} < SenRecInf > .R1(c))$$

Where c = {y, BroRecReq, RefRecReq, AccRecReq, SenReqInf, AskRecInf}.

$$R2(d) = z(msg1).[msg1 = BroRecReq](\bar{z} < RefRecReq > .R2(d) +$$
$$\bar{z} < AccRecReq > .z(msg2).[msg2 = AskRecInf]\bar{z} < SenRecInf > .R2(d)).$$

Where d = {z, BroRecReq, RefRecReq, AccRecReq, SenReqInf, AskRecInf}.

According to Rule 3, Q, R1 and R2 can be combined to form a new process, as "Trust service somposition", denoted TSS, then $TSS(e) = \nu z(Q(b) \mid R1(c) \mid R2(d))$, $e = b \cup c \cup d$.

4. Simulation

The automatic reasoner MWBwhich is based on SML is chosen to analyze and reason the models above. MWB is an efficient model validation tool set [12]. It is capable of searching for deadlock state, testing for equivalence and checking whether a system has a given logical properties (e.g. safety or liveness).

How to judge whether the trust Web service composition is correct or not? The answer is that the system can help trust subject to evaluate the trust value of trust object, and can meet trust subject's demand on trust quality. Therefore, the trust subject's process is taken as design objective of trust service system, and to analyze whether the trust service system can satisfy trust subject's demand. If the trust service system can meet trust subject's demand, then the trust service system's behavior and trust subject's action are complementary. So, the trust service system's behavior is equivalent to trust subject's dual behavior.

The dual process of trust subject is defined as follows:

$$RP_1(g_1) = x(msg1).[msg1 = ReqInt](\bar{x} < RefReqInt > .RP_1(g_1) + \bar{x} < AccReqInt > .$$
$$x(msg2).[msg2 = AskDirTru]\bar{x} < ProSelTru > .\bar{x} < AskDirTru > .$$
$$x(msg3).[msg3 = ProSelTru]RP_1(g_1)).$$

$$RP_2(g_2) = y(msg1).[msg1 = BroRecReq](\bar{y} < RefRecReq > .RP_2(g_2) + \bar{y} < AccRecReq > .$$
$$y(msg2).[msg2 = AskRecInf]\bar{y} < SenRecInf > .RP_2(g_2)).$$

$$RP(g) = RP_1(g_1) \mid RP_2(g_2).$$

Where g_1 = {x, ReqInt, RefReqInt, AccReqInt, AskDirTru, ProSelTru}, g_2 = {y, BroRecReq, RefRecReq, AccRecReq, SenReqInf, AskRecInf}, $g = g_1 \cup g_2$.

The channel z in trust service system is an internal channel, the actions through this channel can not be observed by trust subject, which are equivalent to τ. The observable action set of trust service system is same to that of dual process RP exactly.

The simulation results of models above as shown in Figure 7. The trust service system is grammatically correct, and without deadlock and circulation, namely system is active. Using step command to track both TSS and RP, it can be found that although they have different

internal structure, their external behaviors are same. Therefore, trust service system can meet trust subject's demand effectively.

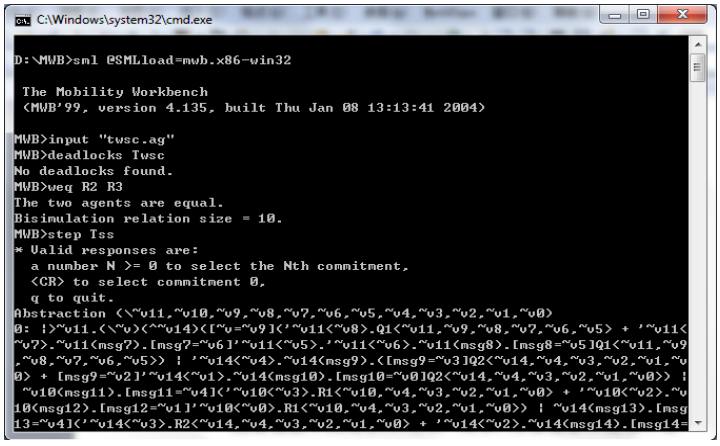

Figure 7. The Simulation Results

5. Conclusion

According to the characteristics of trust Web service composition, Pi-calculus based model method is proposed to analyze and reason it with MWB. The system behavior can be analyzed at design phase, errors can be found and corrected, then running time errors can be avoided. The results show that Pi-calculus based formal model is feasible and effective. The next work is to refine evaluation model of trust value and establish trust transfer mechanism for Web service.

Acknowledgement

The author would like to thank the anonymous referees for their valuable comments and suggestions of improvements. This work is partly funded by the National Natural Science Foundation of China (Grant No.11147114 & 11204272), and the Scientific Project of Zhejiang Provincial Science Technology Department (Grand No.2011C33012).

References

[1] Algirdas Avizienis, Jean-Claude Laprie, Brian Randell, et al. Basic Concepts and Taxonomy of Dependable and Secure Computing. *IEEE Transactions on Dependable and Secure Computing*. 2004; 1(1): 11-33.
[2] Shen Changxiang, Zhang Huanguo, Feng Dengguo, et al. Survey of Information Security. *Science China Information Sciences*. 2007; 50(3): 273-298.
[3] Wang Wei, Zeng Guosun. Bayesian Cognitive Trust Model Based Self-Clustering Algorithm for MANETs. *Science China Information Sciences*. 2010; 53(3): 494-505.
[4] Yun Ben-sheng, Yan Jun-wei, Liu Min. Method to Optimize Web Service Composition Based on Bayes Trust Model. *Computer Integrated Manufacturing System*. 2010; 16(5): 1104-1111.
[5] Bensheng Yun. A Trust Value Computing Approach for Web Service. *Applied Mechanics and Materials*. 2013; 263-266: 3151-3154.
[6] Ayman Tajeddine, Ayman Kayssi, Ali Chehab, et al. Fuzzy Reputation-based trust model. *Applied soft computing*. 2011; 11(1): 345-355.
[7] Yin Gang, Wang Huaimin, Yuan Lin, et al. Construction of Internet-Based Trustworthy Software Production Service System. *Journal of Frontiers of Computer Science and Technology*. 2011; 5(10): 880-890.
[8] Roberto Lucchi, Manuel Mazzara. A Pi-calculus based Semantics for WS-BPEL. *The Journal of Logic and Algebraic Programming*. 2007; 70(1): 96-118.
[9] Sofiane Chemaa, Faycal Bachtarzi, Allaoua Chaoui. A High-level Petri Net Based Approach for Modeling and Composition of Web Service. *Procedia Computer Science*. 2012; 9: 469-478.

[10] Hao Kegang, Guo Xiaoqun, Li Xiangning. The Pi^+ Calculus An Extension of the Pi Calculus for Expressing Petri Nets. *Chinese Journal of Computers*. 2011; 34(2): 193-202.
[11] Davide Sangiorgi, Davide Walker. The Pi-calculus: A Theory of Mobile Processes. Cambridge: Cambridge University Press. 2003.
[12] Victor B, Moller F. *The Mobility Workbench: A Tool for The Pi-calculus*. The University of Edinburgh. Report number: ECS-LFCS-94-285.1994.