Changyou ZHENG[1], Song HUANG[1], Xiaoming LIU1, Yi YAO[1]

Institute of Command Automation, PLA University science and technology, Nanjing, P. R. China (1)

A Modelling Approach for Workflow Constrained by Inputs and Outputs

***Abstract**. Workflow systems have been studied for years, but there is not so much research in workflow testing. The key point for workflow testing is to model a workflow. In this paper, a formalized definition of workflow constrained by inputs and outputs is presented first. Based on the definition and the traditional Petri Net, a kind of I/O_WF_NET model is proposed. In that model, the activities of the workflow are modelled as transitions and the inputs and outputs of an activity are modelled as places. The modelling approach for workflow constrained by inputs and outputs is also presented. And a case study is given to show how to apply this approach and the validity of the proposed modelling method.*

***Streszczenie.** W artykule przedstawiono model przepływu pracy (workflow), oparty na schemacie Petri Net. Wstępnie zdefiniowano sam model wraz z ograniczeniami wejść i wyjść. Na tej podstawie prowadzono dalsze prace. Opisano sposób modelowania przepływu informacji oraz elementów wejścia i wyjścia. Przeprowadzono także badanie warunkach rzeczywistych w celu weryfikacji proponowanego modelu. (**Modelowanie przepływu pracy (workflow) z ograniczeniami dla wejść i wyjść**).*

Keywords: workflow modelling, workflow testing, I/O Constrained workflow, Petri Net.
Słowa kluczowe: modelowanie przepływu pracy, testowanie przepływu pracy, ograniczenia I/O przepływu pracy, sieć Petriego.

1. Introduction

The key point of workflow testing is to model a workflow. Workflow modelling has been studied for years, and by now, there has been much research in this field [1-4]. However, because of the complexity of a workflow system, it is usually difficult to present a uniform modelling method for all kinds of workflows. Hwang [5] proposed a framework for the automatic dynamic testing of workflow management systems. As the syntax definitions are written in Backus–Naur Form, and the whole testing script is written in XML form, the testing script is too complicated for user to use, and the semantic is hard to understand. Karam [6] gave an abstract model to test the web applications which use development frameworks based on the MVC design pattern and the workflow paradigm. As they only described possible strategies for the testing of web applications, the detailed method to model the applications are not presented, and a detailed testing technique is not given. Quan, etc [7] developed an automatic and scalable testing tool to evaluate workflow systems' performance. Bartz [8] gave an automotive test data analysis method based on Petri Nets and stored by ASAM ODS.

Petri net is an effective process modelling technology. There are at least three good reasons for using Petri nets for workflow modelling and analysis [9]: 1) Petri net is a graphical language and its semantics have been defined formally; 2) Petri net is state based instead of event based, so the state of the case can be modelled explicitly in a Petri net; and 3) Petri nets are characterized by the availability of many analysis techniques. And based on the virtues described above, a lot of model analysis and verification techniques and business scheduling methods have been developed [10-14]. In this paper, we present a modelling approach based on a kind of I/O_WF_Net. In this model, the activities in a workflow are defined as transitions in a Petri net, and the inputs and outputs are modelled as places. After the components and structures are modelled, an algorithm for transferring the workflow constrained by inputs and outputs to I/O_WF_Net model is presented, and a case study is given to show how to apply this method.

2. An I/O constrained workflow model based on Petri net

2.1. Basic concepts of Petri nets

A detailed description for Petri net can be found in [15, 16, 17], here, we only present some essential terminologies and notations used in this paper.

Definition 1: A three-tuple $N = (P, T, F)$ is named as a Net iff P and T are finite sets of places and transitions respectively and the following are satisfied:

(1) $P \cap T = \varnothing$, $P \cup T \neq \varnothing$

(2) $F \subseteq (P \times T) \cup (T \times P)$

(3) $Dom(F) \cup Con(F) = P \cup T$

Definition 2: A Petri net is strongly connected iff for each point $x, y \in P \cup T$, there is a path from x to y.

Definition 3: A Petri net $PN = (P, T, F, i)$ is a Workflow net (WF_ net) iff the followings are satisfied:

(1) There are to special places i and o , i is a start place, i.e. ${}^{\bullet}i = \varnothing$; o is an end place, i.e. $o^{\bullet} = \varnothing$.

(2) If a new transition t is added to PN such that ${}^{\bullet}t = \{o\}, t^{\bullet} = \{i\}$, then the new Petri net $\overline{PN}$ is strongly connected.

2.2. Workflow constrained by inputs and outputs

Definition 4: A workflow constrained by inputs and outputs (which can be short as I/O constrained workflow) can be defined as a six-tuple <*Activity, Input, Output, Relation,* f_{AI}, f_{AO} >, where

(1) $Activity = \{activity_1, activity_2, \cdots activity_k\}$ $(k \geq 1)$ is the activity set of a workflow.

(2) $Input = \{input_1, input_2, \cdots input_m\} (m \geq 1)$ is the input set of a workflow.

(3) $Output = \{output_1, output_2, \cdots output_n\} (n \geq 1)$ is the output set of a workflow.

(4) $Relation \subseteq (Activity \times Activity, Type)$ denotes the relation set of a workflow, where Type ⊆ {sequence, and-join, or-join, and-split, or-split}is the relation type between activities.

(5) $f_{AI} : Activity \rightarrow \rho(Input)$ is the input function of a workflow where $\rho(Input)$ is the power set of inputs.

(6) $f_{AO} : Activity \rightarrow \rho(Output)$ is the output function of a workflow where $\rho(Output)$ is the power set of outputs.

Definition 4 presents a formal definition of I/O constrained workflow, from which we can see that:

(1) The set $Activity$, $Input$ and $Output$ includes all the activities, input elements and output elements in the workflow respectively.

(2) The set $Relation$ defines the relations between activities and their types. $\forall activity_1, activity_2 \in Activity$, if $(activity_1, activity_2) \in Relation$, then $activity_2$ cannot started until $activity_1$ is finished, and $activity_1$ is called the pre-activity of $activity_2$, $activity_2$ is called the post-activity of $activity_1$. There are five kinds of types between two activities: *sequence* indicates a one-to-one relationship, which means that the pre-activity has only one post-activity and the post-activity has only one pre-activity; *and-join* indicates a many-to-one relationship, which means that more than one pre-activities has the same post-activity and the post-activity cannot execute until all the pre-activities are finished; *or-join* also indicates a many-to-one relationship, but the post-activity can execute immediately after one (or some) of the pre-activities is finished; *and-split* indicates a one-to-many relationship, which means that more than one post-activities has the same pre-activity and all the post-activities will be started after the pre-activity is finished; *or-split* also indicates a one-to-many relationship, but one (or some) of the post-activities can be selected to execute after the pre-activity is finished.

(3) For $activity_1 \in Activity$, $input_1 \subseteq Input$, if $f_{AI}(activity_1) = input_1$ then it means that the execution of $activity_1$ needs $input_1$; if $f_{AI}(activity_1) = \varnothing$ then the execution of $activity_1$ does not need any input.

(4) For $activity_1 \in Activity$, $output_1 \subseteq Output$, if $f_{AO}(activity_1) = Output_1$ then the implementation of $activity_1$ will generate $output_1$; if $f_{AO}(activity_1) = \varnothing$ then the implementation of $activity_1$ will not produce anything.

Table 1 presents an example of I/O constrained workflow which is composed of 9 activities.

Table 1: a workflow constrained by inputs and outputs

Activity	Inputs	Outputs	Pre-activities	Relation type
A_1	p_1	p_2	$\varnothing$	$\varnothing$
A_2	p_2	p_3	A_1	*and-split*
A_3	p_2	p_4	A_1	*and-split*
A_4	p_5	p_8	A_2	*or-split*
A_5	p_5	p_9	A_2	*or-split*
A_6	p_6	p_{10}	A_3	*and-split*
A_7	p_7	p_{10}	A_3	*and-split*
A_8	p_{11}	p_{13}	A_4, A_5	*or-join*
A_9	p_{12}	p_{14}	A_6, A_7	*and-join*

2.3. I/O_WF_NET Model: a Petri net model for workflow constrained by inputs and outputs

Before introducing the modelling method, a formal definition for the I/O constrained workflow based on Petri net (I/O_WF_NET) is presented as follows:

Definition 5: $\sum = (P, T, F, M_0)$ is an I/O_WF_NET iff the following are satisfied:

(1) (P, T, F, M_0) is a Petri net;

(2) T represents the activity set of a workflow;

(3) $P = P_{in} \cup P_{out}$, $P_{in} \cap P_{out} = \varnothing$, where P_{in} represents the input places and P_{out} represents the output places;

(4) $\forall t \in T$, $p_{in} \in P_{in}$, the execution of t needs input p_{in} iff $p_{in} \subseteq {}^{\bullet}t$

(5) $\forall t \in T$, $p_{out} \in P_{out}$, the execution of t will output p_{out} iff $p_{out} \subseteq t^{\bullet}$

3. A modelling approach for I/O constrained workflow based on Petri nets

A workflow modelling approach based on I/O_WF_NET is presented in this section. It is assumed that in a workflow, every activity needs only one input and will produce one output.

(1) I/O_WF_NET model for single activity

As a workflow is composed of many activities, so a single activity should be modelled first.

(a) Single activity with input and output

A single activity with input and output is most common in a workflow. In this case, the activity is presented as tow places and one transition, as is shown in figure 1(a).

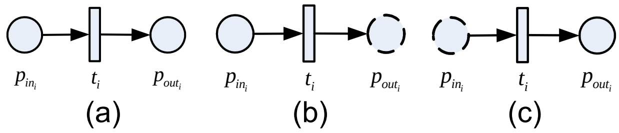

Fig.1. I/O_WF_NET model for single activity

In figure 1, t_i represents activity i, p_{in_i}, p_{out_i} are the input and output of activity i. If p_{in_i} contains tokens, it means that the input of activity i is satisfied and the activity can be implemented; if p_{out_i} contains tokens, then it means that the implementation of activity i is finished, and it produced p_{out_i}.

(b) The implementation of a single activity does not produce any outputs

If the implementation of t_i does not produce any output, then a virtual place p_{out_i} will be added to the model, and it does not mean anything, in figure 1(b), it is represented as a circle with broken line.

(c) The implementation of a single activity does not need any inputs

If the implementation of t_i does not need any input, then a virtual place p_{in_i} will be added to the model, and it does not mean anything, in figure 1(c), it is represented as a circle with broken line.

(2) I/O_WF_NET model for relationships

If $(t_i, t_j) \in Relation$, e.g. activity t_j is the post-activity of activity t_i, then the model will depend on the relation type of the two activities.

(a) The type is *sequence*

If the relation type between activity t_i and t_j is *sequence*, then a new virtual transition t_{ij} is added between them, such that ${}^{\bullet}t_{ij} = p_{out_i}$ and $t_{ij}^{\bullet} = p_{in_j}$, as figure 2. t_{ij} does not have any real meaning, it is just a bridge which connects t_i and t_j.

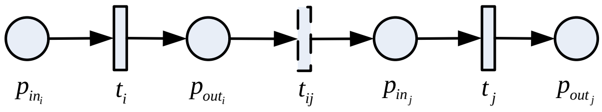

Fig.2. I/O_WF_NET model for *sequence* relation

(b) The type is *and-join*

If the relation type between activity t_i and t_j is *and-join*, then the model built here depends on whether ${}^{\bullet}p_{in_j} = \varnothing$. If ${}^{\bullet}p_{in_j} = \varnothing$, then a new virtual transition t_{ij} will be added between t_i and t_j, such that ${}^{\bullet}t_{ij} = p_{out_i}$, $t_{ij}^{\bullet} = p_{in_j}$. If ${}^{\bullet}p_{in_j} \neq \varnothing$, then another activity t_k has added the virtual transition t_{kj} when it was added to the model, so here the only work to do is to make $p_{out_i}^{\bullet} = {}^{\bullet}p_{in_j}$, that also means $p_{out_i}^{\bullet} = t_{kj}$, and t_i, t_k and t_j form an *and-join* relationship, as is shown in figure 3.

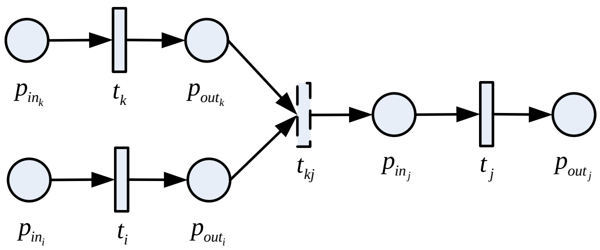

Fig 3: I/O_WF_NET model for *and-join* relation

(c) The type is *or-join*

If the relation type between activity t_i and t_j is *or-join*, then simply a new virtual transition t_{ij} will be added between t_i and t_j, such that ${}^{\bullet}t_{ij} = p_{out_i}, t_{ij}^{\bullet} = p_{in_j}$. The model will be like figure 4.

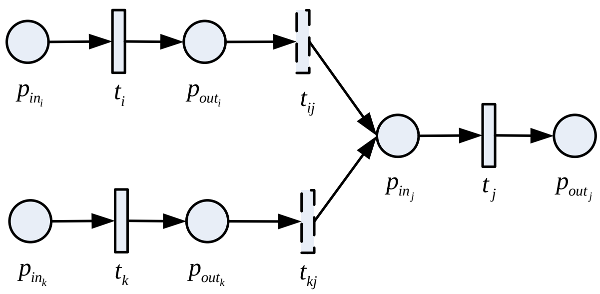

Fig 4: I/O_WF_NET model for *or-join* relation

(d) The type is *and-split*

If activities t_i and t_j form an *and-split* relation, then as *and-join* relation, the model built this time also depends on whether $p_{out_i}^{\bullet} = \varnothing$ or not. If $p_{out_i}^{\bullet} = \varnothing$ then a new transition t_{ij} will be added between t_i and t_j, such that ${}^{\bullet}t_{ij} = p_{out_i}$, $t_{ij}^{\bullet} = p_{in_j}$. If $p_{out_i}^{\bullet} \neq \varnothing$, then it means a virtual activity t_{ik} has been added while another activity t_k was added to the model. So here we simply let ${}^{\bullet}p_{in_j} = p_{out_i}^{\bullet}$, and the model is constructed, as figure 5.

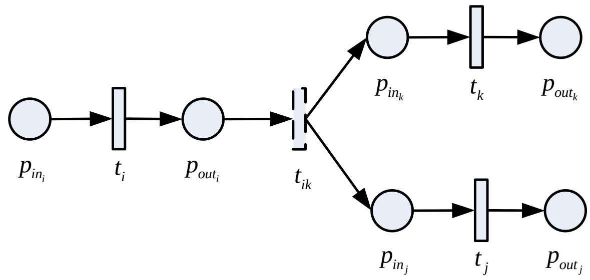

Fig 5: I/O_WF_NET model for *and-split* relation

(e) The type is *or-split*

If activities t_i and t_j form an *and-split* relation, a new transition will directly be added between t_i and t_j, such that ${}^{\bullet}t_{ij} = p_{out_i}$, $t_{ij}^{\bullet} = p_{in_j}$, the model built will be like figure 6.

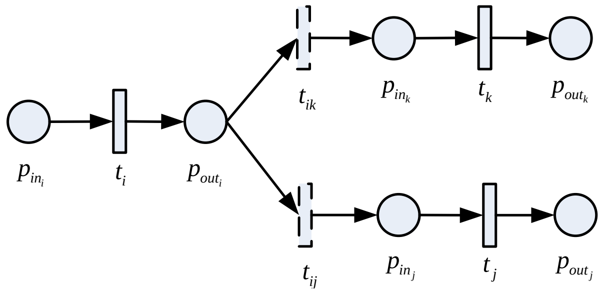

Fig 6: I/O_WF_NET model for *or-split* relation

(3) I/O_WF_NET model for start and end activities

(a) Start activity

For the activities t_i without pre-activities, (e.g. ${}^{\bullet}p_{in_i} = \varnothing$) a virtual transition t_s will be added before them, and is called the start activity, such that $p_{in_i} \subseteq t_s^{\bullet}$. In addition, to keep the characteristic of Petri net, a start place p_s is added before t_s such that ${}^{\bullet}t_s = p_s$, as shown in figure 7(a).

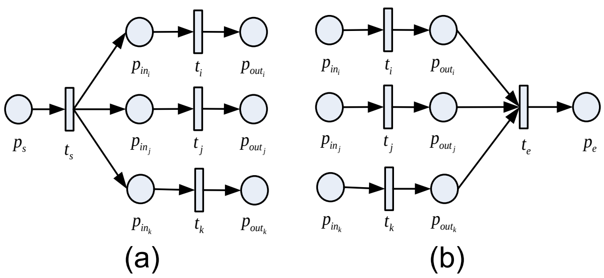

Fig 7: I/O_WF_NET model for start and end activity

(b) End activity

For the activities t_i without post-activities (e.g.

$p_{out_i}^{\bullet} = \varnothing$), a virtual transition t_e will be added after them, and is called the end activity, such that $p_{out_i} \subseteq {}^{\bullet}t_e$. In addition, to keep the characteristic of Petri net, an end place p_e is added after t_e such that $t_e^{\bullet} = p_e$, as shown in figure 7(b).

(4) Initial marking

The initial marking M_0 of I/O_WF_Net $\Sigma = (P, T, F, M_0)$ satisfies the following:

$$M_0(p) = \begin{cases} 1, & p = p_s \\ 0, & else \end{cases}$$

Based on the modelling details mentioned above, an algorithm for transforming a I/O constrained workflow into the I/O_WF_Net model is given as follows.

Algorithm 1：
INPUT：Workflow=<*Activity, Input, Output, Relation, f_{AI}, f_{AO}*>
OUTPUT：$\Sigma = (P, T, F, M_0)$
begin

$$P \leftarrow \varnothing, T \leftarrow \varnothing, F \leftarrow \varnothing, M_0 \leftarrow \varnothing$$
$$T \leftarrow Activity$$
for $i = 1$ to $|T|$ do
$$p_{in_i} \leftarrow F_{AI}(t_i), \quad p_{out_i} \leftarrow F_{AO}(t_i)$$
$$P \leftarrow P \cup \{p_{in_i}, p_{out_i}\}$$
$$F \leftarrow F \cup \{(p_{in_i}, t_i), (t_i, p_{out_i})\}$$
end for
$\forall t_i, t_j \in T (1 \le i, j \le |T|)$ if $t_i, t_j \in Relation$ then
Swich Relation.Type
Case and-join:
If ${}^{\bullet}p_{in_j} = \varnothing$ then
$$T \leftarrow T \cup t_{ij}$$
$$F \leftarrow F \cup \{(p_{out_i}, t_{ij}), (t_{ij}, p_{in_j})\}$$
Else
$$F \leftarrow F \cup \{(p_{out_i}, {}^{\bullet}p_{in_j})\}$$
End if
Case and-split:
If $p_{out_i}^{\bullet} = \varnothing$ then
$$T \leftarrow T \cup t_{ij}$$
$$F \leftarrow F \cup \{(p_{out_i}, t_{ij}), (t_{ij}, p_{in_j})\}$$
Else
$$F \leftarrow F \cup \{(p_{out_i}^{\bullet}, p_{in_j})\}$$
End if
Default:
$$T \leftarrow T \cup t_{ij}$$
$$F \leftarrow F \cup \{(p_{out_i}, t_{ij}), (t_{ij}, p_{in_j})\}$$
End Swich
$$T_s \leftarrow \{t_i \mid t_i \in T, (p_{in_i}, t_i) \in F \wedge {}^{\bullet}p_{in_i} = \varnothing\}$$
$$P \leftarrow P \cup p_s, \quad T \leftarrow T \cup t_s,$$
$$F \leftarrow F \cup \{(p_s, t_s)\}$$
for each $t_i \in T_s$ do
$$F \leftarrow F \cup \{(t_s, p_{in_i})\}$$
end for
$$T_e \leftarrow \{t_i \mid t_i \in T, (t_i, p_{out_i}) \in F \wedge p_{out_i} = \varnothing\}$$
$$P \leftarrow P \cup p_e, \quad T \leftarrow T \cup t_e,$$
$$F \leftarrow F \cup \{(t_e, p_e)\}$$
for each $t_i \in T_e$ do
$$F \leftarrow F \cup \{(p_{out_i}, t_e)\}$$
end for
$$M_0(p_s) \leftarrow 1$$
Output $\Sigma = (P, T, F, M_0)$

End

The time cost of algorithm 1 mainly depends on the loops in it. For the first one, its time cost is $O(|T|)$. For the second step that modelling the relationship, the time cost will be $O(|Relation|) = O(|T| \times |T|) = O(|T|^2)$. Since the total number of start activities and end activities is less than $|T|$, so the time cost for modelling the start and end activities is $O(|T|)$. The time spend on initial marking the net is $O(1)$. In conclusion, the time cost of algorithm 1 is $O(|T|^2)$. Taking the workflow in Table 1 as an example, its I/O_WF_Net model after transforming by algorithm 1 is shown in figure 8.

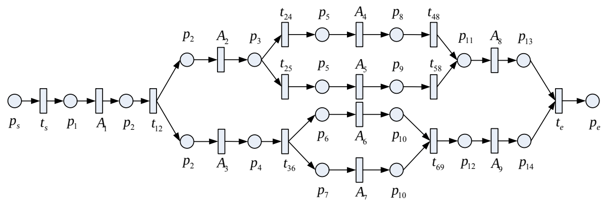

Fig 8: I/O_WF_NET model for the workflow shown in Table 1

4 . Conclusion and future work

Workflow systems have been developed and researched for years, the validity and its reliability should be focused on now. Based on the formal definition of workflow constrained by inputs and outputs, a testing oriented workflow modelling approach is presented in this paper.

The main contributions of this work include the following:

(1) A formal definition for I/O constrained workflow net is performed, and each component of the definition is analyzed and described in detailed.

(2) A I/O_WF_Net model and the modelling approach for I/O constrained workflow are given.

(3) An algorithm for transforming an I/O constrained workflow into I/O_WF_Net model is presented, and its time complexity is discussed.

In this paper, we only studied on the modelling approach for workflow. The future research will include following several subjects: first, the research of modelling and reducing method for workflows that have activities with multiple inputs and multiple outputs must be carry on. Second, based on the modelling approach, test case generation method should be studied. Third, test coverage oracle for completeness should be given. And forth, do some research on the optimal test strategy for the workflow constrained by shared resources.

Acknowledgements

This work is supported by National High Technology Research and Development Program of China (No: 2009AA01Z402), China Postdoctoral Science Foundation (No: 20110491843) and the Natural Science Foundation of Jiangsu Province, China (Grant No. BK2012059, BK2012060). Resources of the PLA Software Test and Evaluation Centre for Military Training are used in this research.

REFERENCES

[1] Jin L.,Jian W.. A Linear Logic Representation for BPEL Process Protocol. Applied Mathematics & Information Sciences. 5(2011), No. 2, 25S-31S.

[2] Karniel, Y. Reich. Formalizing a Workflow-Net Implementation of Design-Structure-Matrix-Based Process Planning for New Product Development. IEEE Trans. Syst., Man, Cybern. A, Syst., Humans. 41(2011),No. 3, 476-491.

[3] Nichols, H. Demirkan, and M. Goul. Autonomic workflow executionin the grid. IEEE Trans. Syst., Man, Cybern. C, Appl. Rev. 36(2006),No 3, 353–364.

[4] M. P. van der Aalst, T. Weijters, and L. Maruster. Workflow mining: Discovering process models from event logs. IEEE Trans. Knowl. Data Eng. 16(2004). No. 9, 1128–1142.

[5] Hwang, C. Lin, etc. A Framework and Language Support for Automatic Dynamic Testing of Workflow Management Systems. 2009 Third IEEE International Symposium on Theoretical Aspects of Software Engineering. (2009), 139-146.

[6] Karam, W. Keirouz, R. Hage. An Abstract Model for Testing MVC and Workflow Based Web Applications. Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services. (2006).

[7] Quan, X. Lin, J. Wang. An Automatic and Scalable Testing Tool for Workflow Systems. The 3rd International Conference on Grid and Pervasive Computing – Workshops. (2008),75-80.

[8] Bartz. Workflow for Automotive Test Data Analysis Based on Petri Nets and Stored by ASAM ODS.

[9] Wang ,Q. Zeng. Modeling and Analysis for Workflow Constrained by Resources and Nondetermined Time-An Approach Based on Petri Nets. IEEE Trans. Syst., Man, Cybern. A, Syst., Humans. 38(2008), No. 4, 802-816.

[10]Pang, C. Jiang. Workflow Performance Analysis Based on Invariant Decomposition Algorithm. Chinese Journal of Computers. 33(2010), 5, 908-918.

[11]Y. Yu, Y. Tang, etc. Temporal Workflow Process Model and Its Soundness Verification. Journal of Software. 21(2010), No. 6, 1233−1253.

[12]Munir. Merdan, Thomas Moser, etc. Simulation of Workflow Scheduling Strategies Using the MAST Test Management System. 10th Intl. Conf. on Control, Automation, Robotics and Vision. (2008), 1172-1177.

[13]J. Wang, D Rosca, W Tepfenhart, etc. Dynamic Workflow Modeling and Analysis in Incident Command Systems. IEEE Trans. Syst., Man, Cybern. A, Syst., Humans. 38(2008), No. 5, 1041-1055.

[14]Y. Du, C. Jiang, M. Zhou. Modeling and Analysis of Real-Time Cooperative Systems Using Petri Nets. IEEE Trans. Syst., Man, Cybern. A, Syst., Humans. 37 (2007) 5, 643-654.

[15]C. A. Petri, Kommunikation mit automaten, Ph.D. dissertation, Inst. Instrumentelle Math., Bonn, Germany, 1962.

[16]W. Reisig, Petri nets: An introduction, in Monographs in Theoretical Computer Science: An EATCS Series, 4(1985). Berlin, Germany: Springer-Verlag,.

[17]T. Murata, Petri nets: Properties, analysis and applications, Proc. IEEE. 77(1989), No.4, 541–580.

***Authors**: C. Zheng, S. Huang, X. Liu and Y. Yao are with the Institute of Command Automation, PLA University of Science and Technology, Nanjing, China.*
E-mail: zheng_chy@163.com; hs0317@163.com; xml_nj@163.com; yaoyi226@yahoo.com.cn.