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.

Keywords: workflow modelling, workflow testing, I/O Constrained workflow, Petri Net.

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 if \( P \) and \( T \) are finite sets of places and transitions respectively and the following are satisfied:

1. \( P \cap T = \emptyset \), \( P \cup T \neq \emptyset \)
2. \( F \subseteq (P \times T) \cup (T \times P) \)
3. \( \text{Dom}(F) \cup \text{Con}(F) = P \cup T \)

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 following are satisfied:

1. There are to special places \( i \) and \( o^* \), \( i \) is a start place, i.e. \( *i = \emptyset \), \( o^* \) is an end place, i.e. \( o^* = \emptyset \).
2. If a new transition \( t \) is added to \( PN \) such that \( t = \{o^*\}, t^* = \{i\} \), 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 \(<\text{Activity}, \text{Input}, \text{Output}, \text{Relation}, f_{\text{IO}}, f_{\text{AO}}\>\), where

1. \( \text{Activity} = \{\text{activity}_1, \text{activity}_2, \cdots \text{activity}_k\} \) (\( k \geq 1 \)) is the activity set of a workflow.
2. \( \text{Input} = \{\text{input}_1, \text{input}_2, \cdots \text{input}_m\} \) (m \( \geq 1 \)) is the input set of a workflow.
3. \( \text{Output} = \{\text{output}_1, \text{output}_2, \cdots \text{output}_n\} \) (n \( \geq 1 \)) is the output set of a workflow.
4. \( \text{Relation} \subseteq \{\text{Activity} \times \text{Activity}, \text{Type}\} \) denotes the relation set of a workflow, where \( \text{Type} \subseteq \{\text{sequence, and-join, or-join, and-split, or-split}\} \) is the relation type between activities.
5. \( f_{\text{IO}} : \text{Activity} \rightarrow \rho(\text{Input}) \) is the input function of a workflow where \( \rho(\text{Input}) \) is the power set of inputs.
6. \( f_{\text{AO}} : \text{Activity} \rightarrow \rho(\text{Output}) \) is the output function of a workflow where \( \rho(\text{Output}) \) 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. \( \forall \text{activity}_1, \text{activity}_2 \in \text{Activity} \), if \((\text{activity}_1, \text{activity}_2) \in \text{Relation}\), then \(\text{activity}_2\) cannot started until \(\text{activity}_1\) is finished, and \(\text{activity}_1\) is called the pre-activity of \(\text{activity}_2\), \(\text{activity}_2\) is called the post-activity of \(\text{activity}_1\). 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 \(\text{activity}_i \in \text{Activity}\), \(\text{input}_i \subseteq \text{Input}\), if \(f_{\text{IO}}(\text{activity}_i) = \text{input}_i\), then it means the execution of \(\text{activity}_i\) needs \(\text{input}_i\); if \(f_{\text{IO}}(\text{activity}_i) = \emptyset\), then the execution of \(\text{activity}_i\) does not need any input.

(4) For \(\text{activity}_i \in \text{Activity}\), \(\text{output}_i \subseteq \text{Output}\), if \(f_{\text{IO}}(\text{activity}_i) = \text{Output}\), then the implementation of \(\text{activity}_i\) will generate \(\text{output}_i\); if \(f_{\text{IO}}(\text{activity}_i) = \emptyset\), then the implementation of \(\text{activity}_i\) will not produce anything.

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

<table>
<thead>
<tr>
<th>Activity</th>
<th>Inputs</th>
<th>Outputs</th>
<th>Pre-activities</th>
<th>Relation type</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>p1, p2</td>
<td></td>
<td>∅</td>
<td>∅</td>
</tr>
<tr>
<td>A2</td>
<td>p2, p3</td>
<td>p1, p3</td>
<td>A1</td>
<td>and-split</td>
</tr>
<tr>
<td>A3</td>
<td>p3, p4</td>
<td></td>
<td>A1</td>
<td>and-split</td>
</tr>
<tr>
<td>A4</td>
<td>p4, p5</td>
<td></td>
<td>A2</td>
<td>or-split</td>
</tr>
<tr>
<td>A5</td>
<td>p5, p6</td>
<td></td>
<td>A2</td>
<td>or-split</td>
</tr>
<tr>
<td>A6</td>
<td>p6, p7</td>
<td></td>
<td>A3</td>
<td>and-split</td>
</tr>
<tr>
<td>A7</td>
<td>p7, p8</td>
<td>p7, p8</td>
<td>A3</td>
<td>and-split</td>
</tr>
<tr>
<td>A8</td>
<td>p8, p9</td>
<td>p7, p8</td>
<td>A3</td>
<td>and-split</td>
</tr>
<tr>
<td>A9</td>
<td>p9, p10</td>
<td></td>
<td>A3, A4</td>
<td>or-join</td>
</tr>
<tr>
<td>A10</td>
<td>p10, p11</td>
<td></td>
<td>A3, A4</td>
<td>or-join</td>
</tr>
<tr>
<td>A11</td>
<td>p11, p12</td>
<td></td>
<td>A3, A4, A5</td>
<td>and-join</td>
</tr>
<tr>
<td>A12</td>
<td>p12, p13</td>
<td></td>
<td>A3, A4, A5</td>
<td>and-join</td>
</tr>
</tbody>
</table>

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. \(P = P_{\text{in}} \cup P_{\text{out}}\), \(P_{\text{in}} \cap P_{\text{out}} = \emptyset\), where \(P_{\text{in}}\) represents the input places and \(P_{\text{out}}\) represents the output places;
4. \(\forall t \in T, \ p_{\text{in}} \in P_{\text{in}}\), the execution of \(t\) needs \(p_{\text{in}}\) iff \(p_{\text{in}} \subseteq \ast t\)
5. \(\forall t \in T, \ p_{\text{out}} \in P_{\text{out}}\), the execution of \(t\) will output \(p_{\text{out}}\) iff \(p_{\text{out}} \subseteq \ast t\)

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 two places and one transition, as is shown in figure 1(a).

(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 \(P_{\text{out}}\) 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 \(P_{\text{in}}\) 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 \((t_i, t_j) \in \text{Relation}\), 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 \(t_y\) is added between them,
such that $t_{ij} = p \text{out}_j$ and $t_{ij}^* = p \text{in}_i$, as figure 2. $t_{ij}$ does not have any real meaning, it is just a bridge which connects $t_i$ and $t_j$.

$$p_{in} \quad t_j \quad p_{out} \quad t_i \quad p_{in} \quad t_j \quad p_{out}$$

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 $p_{in}^* = \emptyset$. If $p_{in}^* = \emptyset$, then a new virtual transition $t_{ij}$ will be added between $t_i$ and $t_j$, such that $t_{ij} = p \text{out}_j$, $t_{ij}^* = p \text{in}_i$. If $p_{in}^* \neq \emptyset$, 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 $p_{out}^* = p_{in}^*$, that also means $p_{out}^* = t_{ij}$, and $t_i$, $t_k$, and $t_j$ form an and-join relationship, as is shown in figure 3.

$$p_{in} \quad t_j \quad p_{out} \quad t_i \quad p_{in} \quad t_j \quad p_{out}$$

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 $t_{ij} = p \text{out}_j$, $t_{ij}^* = p \text{in}_i$. The model will be like figure 4.

$$p_{in} \quad t_j \quad p_{out} \quad t_i \quad p_{in} \quad t_j \quad p_{out}$$

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 $p_{in}^* = \emptyset$ or not. If $p_{in}^* = \emptyset$ then a new transition $t_{ij}$ will be added between $t_i$ and $t_j$, such that $t_{ij} = p \text{out}_j$, $t_{ij}^* = p \text{in}_i$. If $p_{in}^* \neq \emptyset$, 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 $p_{in}^* = p_{out}^*$, and the model is constructed, as figure 5.

$$p_{in} \quad t_j \quad p_{out} \quad t_i \quad p_{in} \quad t_j \quad p_{out}$$

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 $t_{ij} = p \text{out}_j$, $t_{ij}^* = p \text{in}_i$, the model built will be like figure 6.

$$p_{in} \quad t_j \quad p_{out} \quad t_i \quad p_{in} \quad t_j \quad p_{out}$$

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. $p_{in}^* = \emptyset$) a virtual transition $t_i$ will be added before them, and is called the start activity, such that $p_{in}^* = t_i^*$. In addition, to keep the characteristic of Petri net, a start place $p_s$ is added before $t_i$, such that $p_{in}^* = t_i^*$, as shown in figure 7(a).

$$p_{in} \quad t_i \quad p_{out} \quad t_i \quad p_{in} \quad t_i \quad p_{out}$$

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.
$p_{out}^e = \emptyset$), a virtual transition $t_e$ will be added after them, and is called the end activity, such that $p_{out} \subseteq t_e^e$. In addition, to keep the characteristic of Petri net, an end place $p_e$ is added after $t_e$ such that $t_e^e = p_e^e$, 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:

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

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

$P \leftarrow \emptyset, T \leftarrow \emptyset, F \leftarrow \emptyset, M_0 \leftarrow \emptyset$

$T \leftarrow \text{Activity}$

for $i = 1$ to $|T|$ do

$p_{in_i} \leftarrow F_{AI}(t_i)$, $p_{out_i} \leftarrow F_{AO}(t_i)$

$P \leftarrow P \cup \{p_{in_i}, p_{out_i}\}$

$F \leftarrow F \cup \{(p_{in_i}, t_i), (t_i, p_{out_i})\}$

end for

$\forall t_i, t_j \in T(1 \leq i, j \leq |T|)$ if $t_i, t_j \in \text{Relation}$ then

Switch Relation.Type

Case and-join:

if $^*p_{in_i} = \emptyset$ then

$T \leftarrow T \cup t_i$

$F \leftarrow F \cup \{(p_{out_i}, t_i), (t_i, p_{in_i})\}$

Else

$F \leftarrow F \cup \{(p_{out_i}, ^*p_{in_i})\}$

End if

Case and-split:

if $^*p_{out_i} = \emptyset$ then

$T \leftarrow T \cup t_i$

$F \leftarrow F \cup \{(p_{out_i}, ^*t_i), (t_i, p_{in_i})\}$

Else

$F \leftarrow F \cup \{(p_{out_i}, p_{in_i})\}$

End if

Default:

$T \leftarrow T \cup t_i$

$F \leftarrow F \cup \{(p_{out_i}, t_i), (t_i, p_{in_i})\}$

End Switch

end for

$T_s \leftarrow \{t_i | t_i \in T, (p_{in_i}, t_i) \in F \land ^*p_{in_i} = \emptyset\}$

$P \leftarrow P \cup p_e$, $T \leftarrow T \cup t_e$

$F \leftarrow F \cup \{(t_e, p_e)\}$

for each $t_i \in T_s$ do

$F \leftarrow F \cup \{(t_i, p_{in_i})\}$

end for

$T_e \leftarrow \{t_i | t_i \in T, (t_i, p_{out_i}) \in F \land p_{out_i} = \emptyset\}$

$P \leftarrow P \cup p_e$, $T \leftarrow T \cup t_e$

$F \leftarrow F \cup \{(p_{out_i}, t_e)\}$

for each $t_i \in T_e$ do

$F \leftarrow F \cup \{(p_{out_i}, p_{in_i})\}$

end for

$M_0(p_e) \leftarrow 1$

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


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


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.