Verification of Inter-organizational Workflow Models in Supply Chain Management

Henry H. Bi  
Department of MIS  
University of Arizona  
Tucson, AZ 85721  
hbi@eller.arizona.edu

Arie Segev  
Haas School of Business, CITM  
University of California  
Berkeley, CA 94720  
segev@haas.berkeley.edu

J. Leon Zhao  
Department of MIS  
University of Arizona  
Tucson, AZ 85721  
lzhao@eller.arizona.edu

Abstract

Inter-organizational workflows play an important role in supply chain management. The deployment of correctly designed workflow models helps avoid higher costs of breakdown, debugging, and fixing during runtime. The issue of workflow verification is even more critical in supply chain management as its workflow is comprised of a global workflow and the private workflows of all organizations involved. Furthermore, the autonomous requirement makes it difficult to verify all parts of the supply chain workflow as a single step. That is, organizations in a supply chain usually have no information about the processes of other organizations, and each organization can change its processes so long as such changes do not violate the public processes. As a result, an inter-organizational workflow cannot be simply treated as a single workflow for the purpose of verification. In this paper, we extend our logic-based workflow verification approach to supply chain management. We propose a two-level technique in which we isolate private processes by splitting public activities, verify each private process, and then verify the global process. We demonstrate the validity of the two-level approach in the context of supply chain management.

Keywords: inter-organizational workflow verification, supply chain management, workflow verification

1. Introduction

The recent expansion in e-business and supply chain management has led to more complex business processes and has increased the need for more advanced workflow verification technologies to help workflow designers develop correct workflow models. Workflow verification is to examine whether there are any conflicts or errors in workflow models so as to avoid much higher costs of breakdown, debugging, and fixing during runtime. The existing research in workflow verification focuses on the situations in which the entire models are known to designers [1-3, 5-7, 9, 11]. These verification approaches are ineffective in supply chain management because its workflow spans multiple organizations where each organization usually does not know the private business processes of other organizations. Therefore, verification of supply chain workflows requires the extension of existing workflow verification techniques due to the inter-organizational nature.

In this paper, we propose a logic-based approach for verifying inter-organizational workflow models in the context of supply chains. As aforementioned, the autonomous requirement renders that it is unrealistic to verify public and private processes in one step. We propose a novel approach that consists of two levels of verification. At the first level, private workflows are separated from the public workflow at the interface between organizations. This is done by means of interfacing activities added to the original supply chain workflow. At the second level, the global workflow is verified with respect to the interfacing activities plus some other dummy activities resulting from the private workflow verification. We refer to this approach as a Partition by Interfacing Activities method, or simply PIA method. We demonstrate with an example that the PIA method is effective in verifying supply chain workflows.
The rest of the paper proceeds as follows. In Section 2 we give an overview of inter-organizational workflows and their characteristics that must be considered in verification. We review the logic-based verification approach in Section 3. In Section 4 we demonstrate how the logic-based approach can be extended to verifying inter-organizational workflows, resulting in the PIA method. We use a supply chain example to illustrate our approach. Finally, we conclude the paper in Section 5 by discussing related work, limitations and contributions, and future research directions.

2. Inter-organizational Workflow Verification

An inter-organizational workflow model comprises multiple interactive workflows. Each workflow runs inside an independent organization or unit. There are connections between activities of different workflows to link independent and interactive workflows as an integral inter-organizational workflow model. An inter-organizational workflow model consists of private processes and public processes that have been identified in previous research [4]. Each workflow that runs within the boundary of an organization is private business processes (or simply private processes). The connections between workflows cross the boundaries of organizations and constitute the public business processes (or simply public processes). Public processes are the interactive processes between related activities of different workflows. Intuitively, public processes are the links between workflows.

We use activity-based workflow modeling notation (Figure 1) to represent workflows [5]. In activity-based workflow modeling, there are two types of nodes, activity nodes that represent activities and control nodes that route the execution flows of activities. There are two special activity nodes. Start activity node represents the start point of the workflow and end activity node stands for the end point. There are three kinds of control nodes, AND node, XOR (exclusive or) node, and OR node. Activities linked to an AND node should all be executed. For activities that are linked to an XOR node, only one of them is executed. For activities that are linked to an OR node, one or more of them may be executed, depending on the scenarios. Finally, directed arcs are used to link nodes.

Figure 1. Notation for Activity-based Workflow Modeling

![Activity node](image1)

![Start activity node](image2)

![End activity node](image3)

![AND node](image4)

![OR node](image5)

![XOR node](image6)

![Directed arc](image7)

Figure 2. A General Inter-organizational Workflow Model

![Organization A](image8)

![Organization B](image9)

![Organization N](image10)

Figure 2 demonstrates a general inter-organizational workflow model without showing control nodes. $a_1$ and $a_n$ represent the start activity and the end activity, respectively. An inter-organizational workflow model may have more than one start activity and/or end activity. Each private workflow process is displayed within the boundary of an organization. The directed arcs crossing organizational boundaries
and linking pairs of activity nodes from different workflows stand for the public processes.

In an inter-organizational workflow model, a workflow instance is an instantiation of a workflow definition and can be expressed as the executions or activations of activities and control nodes in a specified order. The correctness of an inter-organizational workflow model can be simply defined as that for each workflow instance, there is a “walk through” from the start nodes to all required end nodes, satisfying the ordering relations among activities. A workflow anomaly refers to an error or conflict in the workflow model such that the definition of model is violated or for some workflow instance there is no “walk through” from the start nodes to all required end nodes. The purpose of inter-organizational workflow verification is to examine whether there are any anomalies in an inter-organizational workflow model before the model is implemented.

An inter-organizational workflow has unique characteristics compared with a simple workflow. First, in an inter-organizational workflow model, an organization usually only knows its own business processes and the connection points from which its workflow is linked to other workflows. In other words, an organization does not know the private processes of other organizations. Second, because individual workflows are dependent on and interactive with one another through related activities, there are two types of control flows, control flows inside organizations and control flows between organizations. The correctness of an inter-organizational workflow model depends not only on the structures of all individual workflows but also on all links among them. Local correctness does not guarantee the global correctness of an inter-organizational workflow model.

3. Logic-based Verification Approach

3.1 Activity-based Workflow Modeling

Activity-based workflow modeling paradigm captures the workflow constructs found in commercial workflow systems. An activity-based workflow is composed of a set of activities $A = \{a_1, a_2, \ldots, a_n\}$, control nodes, and directed links. There are usually seven basic building constructs in workflows as shown in Figure 3, in which $a_i, a_j, a_k, \text{ and } a_m \in A$:

Figure 3. Seven Workflow Constructs

(a) **Sequence**: a construct in which an activity leads to another activity.
(b) **AND-Split**: a construct in which multiple threads are generated. These threads can be executed in parallel or in any order.
(c) **AND-Join**: a construct in which multiple parallel threads converge with synchronization.
(d) **XOR-Split**: a construct in which exactly one of multiple threads is to be executed.
(e) **XOR-Join**: a construct in which any one of multiple activities causes the following activity to be executed.
(f) **OR-Split**: a construct in which one or more of multiple activities are chosen to be executed.
(g) **OR-Join**: a construct in which one or more of multiple threads converge.

Five of these constructs, (a)–(e), have been defined as workflow primitives by the Workflow Management Coalition (WFMC) [12]. However, **OR-Split** and **OR-Join** have not been defined by WFMC. Because in some commercial workflow systems **OR-Split** and **OR-Join** have been implemented, we include these two constructs in the activity-based workflow modeling formalism.

With graphical notations in Figure 1 and Figure 3, we simply define a process graph as an activity-based workflow model that consists of activity nodes (including two special activity nodes, start activity and end activity), control nodes, and directed arcs. A standard process graph is a process graph in which there is no link from a split control node directly to a join control node.

### 3.2 Types of Structural Anomalies

**Deadlock.** There are two types of deadlocks, deterministic deadlock where only one of $a_1$ and $a_2$ is executed at the XOR-Split node, the AND-Join node will never execute (Figure 4(a)) and nondeterministic deadlock that happens similarly, but not in all executions (Figure 4(b)).

![Figure 4. Deadlocks](image1)

**Lack of synchronization.** Deterministic lack of synchronization occurs where AND node is followed immediately by XOR node, resulting in multiple executions of an activity (Figure 5(a)). It is nondeterministic when OR node is followed by XOR node (Figure 5(b)).

![Figure 5. Lack of Synchronization](image2)

**Incomplete path.** When a path from an activity does not lead to the end, it is an activity without termination (Figure 6(a)). If an activity on a path that does not begin with the start, it is an activity without activation (Figure 6(b)). Both are referred to as incomplete paths.

![Figure 6. Incomplete Paths](image3)

### 3.3 Logic-based Workflow Verification

Proposed in [5], the theoretical foundation of logic-based workflow verification is based on an analogy between workflow models and logical deductive arguments. If there is no structural error in an activity-based workflow model, every workflow instance can walk its way from the start node of the model through some activities to the end node. This is similar to a deductive argument. If all premises of a deductive argument are true, the conclusion of the argument will be true [8].
In mathematical logic, an argument can be written in the form
Premise 1, Premise 2, …, Premise N \(\vdash\) Conclusion
(1)
where \(\vdash\) is called an assertion sign and is read as “therefore”. According to the aforementioned analogy, a workflow model can be expressed in the argument form
Formula 1, Formula 2, …, Formula N \(\vdash\) (Start \(\rightarrow\) End)
(2)
where logical formulas are obtained by translating constructs in the workflow model into logical formulas in which the “if … then” operator (\(\rightarrow\)) is the main logical operator, and “(Start \(\rightarrow\) End)” stands for from the start node of the model to the end node.

Formula (2) should be considered as a deductive argument whose conclusion (Start \(\rightarrow\) End) follows necessarily from its premises (logical formulas). In other words, it is impossible for (Start \(\rightarrow\) End) to be false while all logical formulas are true. If all logical formulas are true, the conclusion (Start \(\rightarrow\) End) will certainly be reached. If the conclusion (Start \(\rightarrow\) End) cannot be reached, it can be concluded that one or more logical formulas (i.e., one or more constructs in the model) are wrong.

It is necessary to mention that in classical propositional logic, if any logical formula in Formula (2) is false, no matter what the truth value of (Start \(\rightarrow\) End) is, Formula (2) is still a valid form. However, in workflow verification, we do not consider the situation that (Start \(\rightarrow\) End) is true while some constructs are wrong, because this situation will not occur in any real-world workflow models.

Table 1 lists logical operators and symbols from [8] and [10] used for logic-based workflow verification. In addition, each activity is assigned a truth value. For a workflow instance, if an activity is executed, its truth value is 1 (true); if an activity is not executed, its truth value is 0 (false). Furthermore, for any activities \(a_i\) and \(a_j\), \((a_i \rightarrow a_j)\) means if activity \(a_i\) is executed, then activity \(a_j\) will certainly be executed.

Table 1. Logical Operators and Symbols for Logic-based Workflow Verification

<table>
<thead>
<tr>
<th>Logical Operator</th>
<th>Logical Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>and</td>
<td>(\wedge)</td>
</tr>
<tr>
<td>or</td>
<td>(\vee)</td>
</tr>
<tr>
<td>exclusive or (XOR)</td>
<td>(\oplus)</td>
</tr>
<tr>
<td>if … then</td>
<td>(\rightarrow)</td>
</tr>
</tbody>
</table>

In order to convert workflow constructs into logical formulas, the corresponding relationships between them must be established. The conversion rules for seven workflow constructs are summarized in Table 2.

Table 2. Rules of Converting Workflow Constructs into Logical Formulas

<table>
<thead>
<tr>
<th>Workflow Construct</th>
<th>Graph Representation</th>
<th>Logical Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Sequence</td>
<td>Figure 3(a)</td>
<td>(a_i \rightarrow a_j)</td>
</tr>
<tr>
<td>2 AND-Split</td>
<td>Figure 3(b)</td>
<td>(a_i \rightarrow (a_j; a_s; a_t))</td>
</tr>
<tr>
<td>3 AND-Join</td>
<td>Figure 3(c)</td>
<td>((a_i; a_j; a_s) \wedge \rightarrow a_t)</td>
</tr>
<tr>
<td>4 XOR-Split</td>
<td>Figure 3(d)</td>
<td>(a_i \rightarrow \oplus(a_j; a_s; a_t))</td>
</tr>
<tr>
<td>5 XOR-Join</td>
<td>Figure 3(e)</td>
<td>((a_i; a_j; a_s) \oplus \rightarrow a_t)</td>
</tr>
<tr>
<td>6 OR-Split</td>
<td>Figure 3(f)</td>
<td>(a_i \rightarrow \lor(a_j; a_s; a_t))</td>
</tr>
<tr>
<td>7 OR-Join</td>
<td>Figure 3(g)</td>
<td>((a_i; a_j; a_s) \lor \rightarrow a_t)</td>
</tr>
</tbody>
</table>

4. Logic-based Inter-organizational Workflow Verification

As an example, in an inter-organizational workflow model, in addition to aforementioned deadlocks that may arise inside each individual workflow, an inter-workflow deadlock may occur because the executions
of two or more workflows are interdependent on each other. Figure 7 shows an inter-workflow deadlock in which the execution of \( a_2 \) cannot start until \( b_2 \) is executed whereas \( b_2 \) cannot be executed until \( a_2 \) is completed, thus causing a deadlock.

![Figure 7. An Inter-workflow Deadlock](image)

The discussions on the inter-workflow deadlock indicate that the verification of an inter-organizational workflow model must consider the interactions among workflows. In other words, an inter-organizational workflow model must be verified as an entirety. This seems to be impossible because each workflow is a private business process inside an individual organization that is unknown to other organizations. The logic-based workflow verification approach [5] provides a feasible guideline for inter-organizational workflow verification.

### 4.1 Verification Formalism

On the one hand, under the assumption that in an inter-organizational workflow model each organization does not know the private processes of other organizations, it is impossible to verify the whole model altogether. A feasible way is to verify each individual workflow separately. On the other hand, because individual workflows are interactive with one another, it is impossible to verify each individual workflow separately. In order to overcome these two obstacles, we propose a two-level five-step inter-organizational workflow verification approach using logic.

**Step 1:** adding public activities to directed arcs that link different workflows. Given an inter-organizational workflow model in Figure 2, we first add a public activity to each link that connects two different workflows. This step assumes that every pair of organizations that are connected by directed link(s) agree on the public activities to be added. After adding public activities \( (p_1, p_2, ..., p_w) \) to Figure 2, we obtain a model in Figure 8. Like Figure 2, Figure 8 and the following Figure 9 do not include control nodes. Because these public activities do not actually exist and no organization will execute them, adding them to the model does not change the original model.

**Step 2:** splitting public activities. In order to make each individual workflow independent of all other workflows, we split each public activity into two copies, with one copy in each related workflow. Figure 9 is obtained after public activities in Figure 8 are split. After splitting, although individual workflows are still related to one another through the public activities, workflows become independent because there are no direct links among them.

**Step 3:** adding dummy activities to make each workflow a standard process graph. Because Figure 9 does not include control nodes, this step cannot be displayed with a figure here and will be illustrated in an example later.

**Step 4:** verifying each workflow independently. According to the verification approach proposed by Bi and Zhao (2003), each workflow obtained at Step 3 is converted into logical formulas. Then logical deduction is conducted based on four principles (due to the limited space, this paper does not explain logical rules used in logical deduction that are addressed in [5]):

1. Reserve the start and end activities of the inter-organizational workflow model
(2) Reserve public activities
(3) Remove as many other activities as possible
(4) Remove as many dummy nodes as possible

Figure 8. A General Inter-organizational Workflow Model after Adding Public Activities

Figure 9. A General Inter-organizational Workflow Model after Splitting Public Activities

Step 5: verifying the entire inter-organizational workflow model. The new formulas that are obtained at Step 4 and the formulas that have not been used through Step 4 are used altogether to conduct deduction to verify the global correctness of the entire inter-organizational workflow model.

4.2 A Supply Chain Workflow Verification Example

Figure 10 is an inter-organizational workflow example of supply chain management adapted from [2]. There are five organizations in this model, Customer, Supplier, Producer 1, Producer 2, and Producer 3. The Customer requests the Supplier to provide some goods. Depending on the Customer’s request, the Supplier asks one, two, or three Producers to make the requested goods. After receiving deliveries from Producer(s), the Supplier ships goods and sends invoice to the Customer. Finally, the Customer makes a payment to the Supplier.

In this model, node $a_1$ (start) is the start activity that initiates the inter-organizational workflows. Nodes $a_7$ (end) and $b_{11}$ (receive_payment) are the end activities that represent the finish of the inter-organizational workflows. The private processes are five workflows with each inside an organization. The public processes are the links among these five workflows. There are five links between the Customer workflow and the Producer workflow and six links between the Producer workflow and Suppliers’ workflows.
This inter-organizational workflow model can be verified using the five-step verification approach proposed above:

Step 1. Public activities \( p_1 \) through \( p_{11} \) are added to Figure 10.

Step 2. Public activities \( p_1 \) through \( p_{11} \) are split into two copies, with one copy in each related workflow. Figure 11 is the resulted model after adding and splitting public activities \( p_1 \) through \( p_{11} \).

Step 3. From Figure 11 to Figure 12, dummy activities \( t_1 \) through \( t_3 \) and \( t_4 \) through \( t_{11} \) are added to the workflows of Customer and Supplier, respectively, to make them standard process graphs.

Step 4. Each workflow in Figure 12 is converted into logical formulas. Due to limited space, Table 3 shows, as an example, only the logical formulas for the Customer workflow in Figure 12. Then logical deduction is conducted for all private workflows, respectively. For instance, the deduction of the logical formulas for the Customer workflow in Table 3 is illustrated in Table 4.

Step 5. The new formulas that are obtained at Step 4 and those that have not been used at Step 4 are listed
in Table 5. These formulas correspond to five reduced private workflows in Figure 13. These five reduced workflows form a global process as shown in Figure 14. To verify the global correctness of the entire inter-organizational workflow model, a deduction on formulas in Table 5 is conducted in Table 6.

![Inter-Organizational Workflow Model after Adding and Splitting Public Activities](image)

Figure 11. The Inter-Organizational Workflow Model after Adding and Splitting Public Activities

According to principle 3 in Step 4, all real activities except for the start and end in each private workflow have been removed during the deductions on individual private workflows at Step 4. The remaining formulas used to verify the entire inter-organizational workflow model at Step 5 do not contain much information about each private workflow. This is demonstrated in the inter-organizational supply chain workflow verification example. Table 5 of the example list the formulas that are used in the deduction at Step 5. These formulas do not contain any real activities of private workflows except the start activity $a_1$ and the end activities $a_7$ and $b_{11}$, as illustrated in Figure 14. Therefore, when the global correctness of the entire inter-organizational workflow model is verified at Step 5, there is no need to release the information about each private workflow to any other organizations.

The final result is the formula $a_1 \rightarrow \land (a_7, b_{11})$, which is accordant with the intuitive observation from Figure 10 that if the model in Figure 10 is correct, then after activity $a_1$ initiates the supply chain workflow, the end activities $a_7$ and $b_{11}$ will eventually be reached.
Figure 12. Adding Dummy Activities to Standardize the Private Workflows

Figure 13. Reduced Private Workflows after Logical Deductions
Table 3. Logical Formulas Converted from the Customer Workflow in Figure 12

<table>
<thead>
<tr>
<th>No.</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>a-1</td>
<td>(a_1 \rightarrow a_2)</td>
</tr>
<tr>
<td>a-2</td>
<td>(a_2 \rightarrow \land (p_1, t_1))</td>
</tr>
<tr>
<td>a-3</td>
<td>((p_2, t_1) \land a_3)</td>
</tr>
<tr>
<td>a-4</td>
<td>(a_4 \rightarrow \land (t_2, t_3))</td>
</tr>
<tr>
<td>a-5</td>
<td>((p_3, t_2) \land a_4)</td>
</tr>
<tr>
<td>a-6</td>
<td>((p_4, t_3) \land a_5)</td>
</tr>
<tr>
<td>a-7</td>
<td>((a_4, a_5) \land a_6)</td>
</tr>
<tr>
<td>a-8</td>
<td>(a_6 \rightarrow \land (a_7, p_3))</td>
</tr>
</tbody>
</table>

Table 4. Logical Deduction Based on Formulas in Table 3

<table>
<thead>
<tr>
<th>No.</th>
<th>Transformation</th>
<th>Formulas Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>a-9</td>
<td>(a_1 \rightarrow \land (p_1, t_1))</td>
<td>a-1, a-2</td>
</tr>
<tr>
<td>a-10</td>
<td>((p_2, t_1) \land a_3)</td>
<td>a-3, a-4</td>
</tr>
<tr>
<td>a-11</td>
<td>((p_3, t_2) \land (p_4, t_3) \land \rightarrow a_6)</td>
<td>a-5, a-6, a-7</td>
</tr>
<tr>
<td>a-12</td>
<td>((p_3, t_2, p_4, t_3) \land \rightarrow a_6)</td>
<td>a-11</td>
</tr>
<tr>
<td>a-13</td>
<td>((p_3, p_4, p_2, t_1) \land \rightarrow a_6)</td>
<td>a-10, a-12</td>
</tr>
<tr>
<td>a-14</td>
<td>((p_3, p_4, p_2, t_1) \land \rightarrow \land (a_7, p_3))</td>
<td>a-8, a-13</td>
</tr>
</tbody>
</table>

Table 5. New Formulas Obtained from and Formulas Unused in Deduction at Step 4

<table>
<thead>
<tr>
<th>No.</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>a-9</td>
<td>(a_1 \rightarrow \land (p_1, t_1))</td>
</tr>
<tr>
<td>a-14</td>
<td>((p_3, p_4, p_2, t_1) \land \rightarrow \land (a_7, p_3))</td>
</tr>
<tr>
<td>b-19</td>
<td>(p_1 \rightarrow \land (\lor (p_6, p_7, p_8), \land (p_2, \lor (t_7, t_8, t_9))))</td>
</tr>
<tr>
<td>b-22</td>
<td>((p_3, p_10, p_11) \lor (t_7, t_8, t_9) \lor \land \rightarrow \land (p_1, p_10, p_11))</td>
</tr>
<tr>
<td>b-23</td>
<td>((t_{10}, t_{11}, p_3) \land \rightarrow b_{11})</td>
</tr>
<tr>
<td>c-12</td>
<td>(p_6 \rightarrow p_9)</td>
</tr>
<tr>
<td>d-12</td>
<td>(p_7 \rightarrow p_{10})</td>
</tr>
<tr>
<td>e-12</td>
<td>(p_8 \rightarrow p_{11})</td>
</tr>
</tbody>
</table>

Table 6. Logical Deduction Based on Formulas in Table 5

<table>
<thead>
<tr>
<th>No.</th>
<th>Transformation</th>
<th>Formulas Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>F-1</td>
<td>(a_1 \rightarrow \land (\lor (p_6, p_7, p_8), \land (p_2, \lor (t_7, t_8, t_9))))</td>
<td>a-9, b-19</td>
</tr>
<tr>
<td>F-2</td>
<td>(a_1 \rightarrow \land (\lor (p_3, p_{10}, p_{11}), \land (p_2, \lor (t_7, t_8, t_9))))</td>
<td>c-12, d-12, e-12, F-1</td>
</tr>
<tr>
<td>F-3</td>
<td>(a_1 \rightarrow \land (a_7, b_{11}))</td>
<td>a-14, b-22, b-23, F-2</td>
</tr>
</tbody>
</table>
5. Discussion and Conclusion
We have presented a novel workflow verification approach referred to as the PIA method in an inter-organizational setting, particularly in supply chain management. The PIA method, or Partition by Interfacing Activities, consists of two levels of verification – (1) verification of private workflows and (2) verification of the global workflow. In this paper, we developed the basic concepts of this new approach and demonstrated its effectiveness using an example of supply chain management.

Our work in this paper provides an alternative to Petri-nets-based verification of inter-organizational workflows. Petri nets have been applied to verifying workflows [1, 3] and inter-organizational workflows [2]. Petri-nets-based verification approach verifies the soundness of a workflow based on the reachability of states, assuming that each individual workflow is known to all organizations [2]. Several other workflow verification techniques have been introduced recently. Graph reduction techniques verify a workflow by removing the correct components that do not contain any structural error [9]. The matrix-based verification technique represents a workflow in adjacency matrices and verify the correctness of workflows using pattern-based rules and instance analysis [7]. The matrix-based verification and the graph reduction technique have been shown to be effective in verifying certain local workflows, but have not been investigated in the context of inter-organizational workflows.

There is still more work remaining in inter-organizational workflow verification. For instance, the PIA method does not take into account synchronization interactions between two organizations [2] although we think that the PIA method can accommodate it as well. In addition, we plan to implement the approach in a prototype system and investigate the completeness of the PIA method.

References