# Incremental verification of consistency properties of large-scale workflows from the perspectives of control flow and evidence life cycles *

Osamu Takaki, Izumi Takeuti, Takahiro Seino, Noriaki Izumi and Koichi Takahashi
National Institute of Advanced Industrial Science and Technology (AIST)
2-41-6 Aomi, Koto-ku, Tokyo 135-0064, Japan
{o-takaki, takeuti.i, seino-takahiro, n.izumi, k.takahashi}@aist.go.jp

## Abstract

*We investigate consistency properties of workflows from the perspectives of control flow and evidence life cycles for incremental verification for large-scale workflows. For modeling complicated business processes in developing large-scale information systems, it needs to develop large-scale workflows that consist of a lot of small workflows. As a workflow becomes larger and larger, it becomes harder and harder to verify the workflow. Therefore, it is useful to verify large-scale workflows "incrementally", that is, to verify small workflows before they are integrated to form the large-scale workflows. However, in order to verify a workflow incrementally, it is necessary to consider consistency properties of not only a whole workflow but also a subgraph of the whole workflow. Thus, we extend the correctness property of acyclic workflows to that of acyclic workflows with multiple starts and/or ends. Correctness of workflows is one of the most important consistency properties for improving workflow quality from the control flow perspective. Extended correctness is a natural extension of the original correctness property and is preserved in the vertical composition and vertical division of workflows. We also define a consistency property for evidence life cycles in workflows with multiple starts. Moreover, in order to validate the consistency properties above for incremental verification, we investigate real workflows and explain how to verify the consistency properties by using an example.*

Keywords: workflow, verification, correctness, evidence life cycle, incremental verification

## 1. Introduction

For developing large-scale information systems, it needs to model business processes that the systems support. A workflow, or a workflow diagram, is one of the most well known specifications for modeling of business processes. As business processes become more and more complicated, the workflows for modeling them become larger and larger. In the requirements analysis stage of developing large-scale information systems, for example, a number of engineers are needed for developing the workflows, which are divided into a number of smaller workflows. As a result, it has become harder for an engineer to verify the overall workflow in one operation. A method is thus needed for verifying large-scale workflows. One approach is to develop and verify workflows in parallel. We call such an approach "incremental verification". For incremental verification, small workflows should be verified before they are integrated to form a large scale workflow. However, in order to verify a workflow incrementally, it is necessary to consider consistency properties of not only a whole workflow but also a subgraph of the whole workflow, that can not completely satisfy the definition of a usual workflow. Thus, it needs to re-consider conventional consistency properties of a workflow from several perspectives.

Verifying the consistency of workflows from the control flow perspective is important, and several consistency properties have been defined and several verification methods have been developed. Correctness is one of the most standard consistency properties of acyclic workflows from the control flow perspective [10] (also [6] and [17]). However, these properties and methods can only be used to verify the overall workflow as a whole, not to incrementally verify workflows.

In this paper, we extend the correctness property to enable us to verify workflows incrementally. We consider workflows with multiple starts and/or multiple ends and extend the correctness of existing workflows to that of the extended workflows. A workflow in standard workflow lan-

*This work was supported by 'Service Research Center Infrastructure Development Program 2008' from METI and Grant-in-Aid for Scientific Research (C) 20500045.

guages, such as XPDL [23] and YAWL [19], has a single start and a single end. Verification of the consistency properties of workflow subgraphs requires consideration of workflows that may have multiple starts and/or multiple ends.

Extended correctness is a natural extension of the original correctness. Extended correctness is preserved in the vertical composition and division of the workflows. Extended correctness is a necessary and sufficient condition for obtaining a workflow with a single start and a single end. The workflow is correct in the sense of the original correctness property. It is obtained from a workflow satisfying extended correctness by appending appropriate workflows.

This paper is based on a previous one [13]. The main difference between them is the definition of a consistency property of evidence life cycles in a workflow with multiple starts and/or multiple ends. Here "evidence" means an annotation on a workflow, which denotes a document on which information is written, and/or with which something is approval, during the process of an operation. In [12] and [14], this property for a workflow with a single start is defined, based on "instances" of the workflow. We define the property for a workflow with multiple starts, by using "closed" subgraphs of the workflow. We define closed subgraphs of a workflow in this paper, while instances of a workflow are defined elsewhere [9]. Given this consistency property, one can incrementally check evidence life cycles in a workflow with multiple starts.

This paper also generalizes preservation theorems of vertical composition and workflow division (see Theorem 4.2 in this paper or ([13], Theorem 4.2)). This generalization, which is described in Theorem 4.4 in this paper, is easier to understand than that previously presented ([13], Appendix B).

The remainder of this paper is organized as follows. We define workflows with multiple starts and/or multiple ends and define vertical composition and workflow division in Section 2. We give a definition of an extended version of the original correctness property over acyclic workflows with a single start and a single end in Section 3. We refer to this correctness property as "extended correctness". We show the fundamental theorems of extended correctness in Section 4. We also give a definition of consistency of evidence life cycles in a workflow with multiple starts and/or multiple ends in Section 5. The definition is based on the previous one for a workflow with a single start [12]. We discuss the validity of extended correctness, consistency of evidence life cycles and incremental workflow verification based on the consistency properties in Section 6. Using an example, we investigate real workflows and explain how to incrementally verify control flow consistency for a large workflow. We discuss related work in Section 7 and summarize the key points in 8.

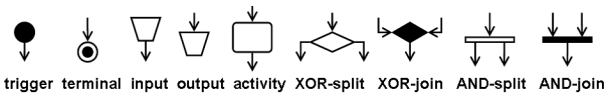


**Figure 1. Shapes of nodes in workflows**

## 2. Workflows

In this section, we define workflows. Moreover, we define certain composition and division of workflows. Workflows in this paper are essentially the same as those in previous studies such as [10], [6] and [17], except the point that a workflow in this paper may have multiple starts and ends. There are several languages of workflows with multiple starts and ends (see Section 7).

In this paper, we discuss workflows only on the control flow perspective. Therefore, we omit notions that are not relevant to control flow of workflows. For example, in this paper we do not consider data flow or actors in workflows.

**Definition 2.1 (Workflows)** A workflow denotes a directed graph $W := (\mathbf{node}, \mathbf{arc})$ that satisfies the following properties.

1. $\mathbf{node}$ is a non-empty finite set, whose element is called a node in $W$.

2. $\mathbf{arc}$ is a non-empty finite set, whose element is called an arc in $W$. Each arc $f$ is assigned to a node called a source of $f$ and another node called a target of $f$.

3. Each node is distinguished, as follows: trigger, terminal, input, output, activity, XOR-split, XOR-join, AND-split and AND-join.

   We employ the symbols in Figure 1 to describe nodes in a workflow in this paper.

4. Whenever an arc $f$ has a node $x$ as the target (or the source) of $f$, $x$ has $f$ as an incoming-arc (resp. an outgoing-arc) of $x$. The numbers of incoming-arcs and outgoing-arcs of a node are determined by the type of the node. We itemize them in the following table.

| | incoming-arcs | outgoing-arcs |
|---|---|---|
| trigger, input | 0 | 1 |
| terminal, output | 1 | 0 |
| activity | 1 | 1 |
| XOR-, AND-split | 1 | $\geqq 2$ |
| XOR-, AND-join | $\geqq 2$ | 1 |

**Table 1. Numbers of incoming- and outgoing-arcs of a node**

5. $W$ has at least one start and at least one end.

6. For a node $x$ in $W$, there exists a trigger or an input $s$ and a path on $W$ from $s$ to $x$, where a path $\pi$ from $s$ to $x$ denotes a sequence $\pi = (f_0, \ldots, f_n)$ of arcs in $W$ such that the source of $f_0$ is $s$, the target of $f_n$ is $x$ and that the target of $f_i$ is the source of $f_{i+1}$ for each $i < n$. Moreover, there exists a terminal or an output $e$ and another path on $W$ from $x$ to $e$.

**Remark 2.2** In the previous paper [13], a workflow $W$ is restricted to be a *connected* graph. That is, [13] assumes that, for each nodes $x$ and $y$ in $W$, there exists a sequence $(x_0, \ldots, x_n)$ consisting of nodes of $W$ such that $x_0 = x$, $x_n = y$ and that there exists an arc in $W$ between $x_i$ and $x_{i+1}$ for each $i < n$. However, in this paper, we also consider a workflow which is not connected. The reason why we consider some unconnected graphs as workflows is only because we have to consider unconnected workflows in Theorem 4.4 in Section 4. Actually, one can regard a workflow as a connected graph when they do not consider the theorem above.

**Remark 2.3** In what follows, triggers and inputs are called "start nodes" or "starts". Moreover, terminals and outputs are called "end nodes" or "ends".

$\mathbf{WF}(n, m)$ denotes the set of all workflows with $n$ starts and $m$ ends and $\mathbf{WF} := \bigcup_{n,m} \mathbf{WF}(n, m)$. For a subgraph $V$ of a workflow $W$, $\mathbf{arc}(V)$ denotes the set of all arcs in $V$, $\mathbf{start}(V)$ the set of all starts in $V$ and $\mathbf{end}(V)$ the set of all ends in $V$.

We next define vertical composition and division of workflows.

**Definition 2.4 (Vertical composition of workflows)** Let $W_1, W_2 \in \mathbf{WF}$, $E \subset \mathbf{end}(W_1)$ and $S \subset \mathbf{start}(W_2)$. Moreover, assume that there exists a bijection $f$ from $E$ to $S$. Then, $W_1 *_f W_2$ denotes the workflow obtained from $W_1$ and $W_2$ by executing the following procedures.

(1) Remove all ends of $E$ and their incoming-arcs.

(2) Remove all starts in $S$ and their outgoing-arcs.

(3) For the source $x$ of the incoming-arc of each end $e$ in $E$ and the target $y$ of the outgoing-arc of each start $f(e)$ in $S$, add the arc from $x$ to $y$.

$W_1 *_f W_2$ is called the vertical composition of $W_1$ and $W_2$ by $f$, and the arcs made in (3) above are called connecting-arcs from $W_1$ to $W_2$ by $f$.

For simplicity, in the remainder of this paper, we omit "$f$" in $W_1 *_f W_2$ and identify each $e \in E$ with $f(e) \in S$.

**Example 2.5** The workflow in Figure 2 is the vertical composition of workflows $W1$ and $W2$, where the bijection function is expressed by two dot-lines in Figure 2, which maps $e_1$ to $s_1$ and $e_3$ to $s_2$.

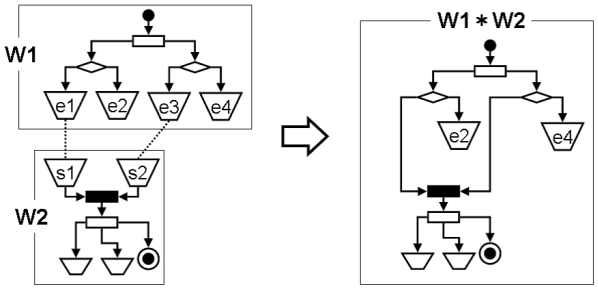


**Figure 2. Vertical composition of workflows**

**Remark 2.6** In fact, all elements of $E$ should be output nodes and those of $S$ input nodes. However, it is not important to distinguish terminal nodes and output nodes (or trigger nodes and input nodes). Therefore, for simplicity, in the remainder of this paper, we assume that a start node denotes an input node only and an end node denotes an output node only, respectively.

**Definition 2.7 (Vertical division of workflows)** For a workflow $W$, if there exist workflows $W_1$ and $W_2$ with $W = W_1 * W_2$, then $W$ is said to be vertically divided into $W_1$ and $W_2$.

## 3. Correctness and extended correctness

In this section, we explain correctness of workflows with a single start, which is defined in [10], and define an extended version of correctness, which we call extended correctness, and which is defined on workflows with multiple starts and multiple ends. Several basic theorems of extended correctness is shown in Appendix A.

In the remainder of this paper, we consider only acyclic workflows, which have no loop. In what follows, a workflow denotes an acyclic workflow.

**Definition 3.1** For a workflow $W$ and a start $s$ in $W$, an instance of $W$ from $s$ denotes a subgraph $V$ of $W$ that satisfies the following properties.

(1) $V$ contains $s$ but does not contain any start except $s$. Moreover, for each $x \in V$, there is a path on $V$ from $s$ to $x$.

(2) If $V$ contains an XOR-split $c$, then $V$ contains a single outgoing-arc of $c$.

(3) If $V$ contains a node $c$ other than XOR-split, then $V$ contains all outgoing-arcs of $c$.

For a workflow $W$, $\mathbf{INS}(W)$ denotes the set of all instances of $W$ and $\mathbf{INS}(W, s)$ the set of all instances of $W$ from $s$.

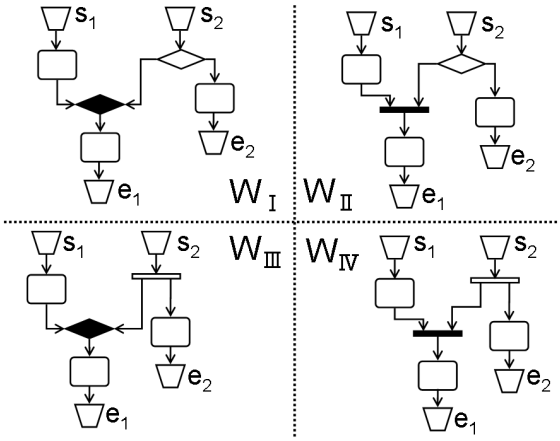


**Figure 3. Four workflows**

**Example 3.2** We explain instances of workflows, by using the four workflows in Figure 3.

(1) The workflow $W_{\rm I}$ has three instances $U_1^{\rm I}$, $U_2^{\rm I}$ and $U_3^{\rm I}$, where $U_1^{\rm I}$ is the path from the start $s_1$ to the end $e_1$, $U_2^{\rm I}$ is the path from $s_2$ to $e_1$ and $U_3^{\rm I}$ is the path from $s_2$ to $e_2$.

(2) The workflow $W_{\rm II}$ has similar instances $U_1^{\rm II}$, $U_2^{\rm II}$ and $U_3^{\rm II}$ to those in $W_{\rm I}$.

(3) The workflow $W_{\rm III}$ has two instances $U_1^{\rm III}$ and $U_2^{\rm III}$, where $U_1^{\rm III}$ is the path from $s_1$ to $e_1$, and $U_2^{\rm III}$ consists of the path from $s_2$ to $e_1$ and that from $s_2$ to $e_2$.

(4) The workflow $W_{\rm IV}$ has similar instances $U_1^{\rm IV}$ and $U_2^{\rm IV}$ to those in $W_{\rm III}$.

**Definition 3.3** Let $W$ be a workflow.

(1) A subgraph $V$ of $W$ is said to be deadlock free if, for every AND-join $r$ in $V$, $V$ contains all incoming-arcs of $r$.

(2) A subgraph $V$ of $W$ is said to be lack of synchronization free if, for every XOR-join $m$ in $V$, $V$ contains a single incoming-arc of $m$.

Correctness is a consistency property of workflows in the viewpoint of control flow of them (cf. [10] or [17]). This property was defined on workflows with a single start and a single end in [10] and [17]. One can easily extend correctness into that over workflows with a single start and multiple ends by using instances of $W$. So, we consider correctness as a consistency property of a workflow that has a single start but may have multiple ends.

**Definition 3.4** (Sadiq and Orlowska [10]) A workflow $W$ with a single start is said to be correct if every instance $V$ of $W$ is deadlock free and lack of synchronization free.

From now, we extend the correctness property above for workflows with multiple starts and/or multiple ends. In order to define the extended correctness, we introduce some basic concepts.

**Definition 3.5** For a workflow $W$ and a non-empty subgraph $V$ of $W$, $V$ is said to be closed if each node $x$ in $V$ satisfies the following properties.

(1) If $x$ is an XOR-split, then $V$ contains a single outgoing-arc of $x$ and the incoming-arc of $x$.

(2) If $x$ is an XOR-join, then $V$ contains a single incoming-arc of $x$ and the outgoing-arc of $x$.

(3) Otherwise, $V$ contains all incoming-arcs and all outgoing-arcs of $x$.

For a workflow $W$ and a set of starts in $W$, $\mathbf{CL}(W)$ denotes the set of all closed subgraphs of $W$ and $\mathbf{CL}(W, S)$ the set of all closed subgraphs $V$ of $W$ with $\mathbf{start}(V) = S$.

Note that, unlike instances of workflows, a closed subgraph of a workflow may not be connected as a graph.

**Example 3.6** We explain closed subgraphs of workflows, by using the previous four workflows in Figure 3.

(1) All instances $U_1^{\rm I}$, $U_2^{\rm I}$ and $U_3^{\rm I}$ of $W_{\rm I}$ are also closed subgraphs of $W_{\rm I}$. Moreover, $U_1^{\rm I} \cup U_3^{\rm I}$ is an unconnected closed subgraph of $W_{\rm I}$.

(2) The workflow $W_{\rm II}$ has two closed subgraphs $U_1^{\rm II} \cup U_2^{\rm II}$ and $U_3^{\rm II}$ in Example 3.2.2.

(3) All instances $U_1^{\rm III}$ and $U_2^{\rm III}$ of $W_{\rm III}$ are also closed subgraphs of $W_{\rm III}$.

(4) $W_{\rm IV}$ has a single closed subgraph, that is $W_{\rm IV}$ itself.

**Definition 3.7** Let $W$ be a workflow.

(1) For $U_1, U_2 \in \mathbf{INS}(W)$, $U_1$ and $U_2$ are said to conflict on an XOR-split $c$ if $U_1$ and $U_2$ share $c$ but the outgoing-arc of $c$ in $U_1$ differs from that in $U_2$.

(2) Let $\mathbf{U}$ be a set of some instances of $W$ and $c$ an XOR-split. Then, $\mathbf{U}$ is said to conflict on $c$ if there exists a pair $(U, U')$ on $\mathbf{U}$ that conflicts on $c$.

**Definition 3.8** Let $W$ be a workflow and $S$ be a non-empty subset $\{s_1, \ldots, s_n\}$ of $\mathbf{start}(W)$. Then, $S$ is called an inport of $W$ if $S$ satisfies the following properties: for each $U_i \in \mathbf{INS}(W, s_i)$ $(i = 1, \ldots, n)$, if $\{U_1, \ldots, U_n\}$ is not conflict on any XOR-split in $W$, then $U_1 \cup \cdots \cup U_n$ is closed.

**Example 3.9** We explain in-ports of workflows, by using the previous four workflows in Figure 3.

(1) There are three non-empty subsets of $\mathbf{start}(W_{\rm I})$: $\{s_1\}$, $\{s_2\}$ and $\{s_1, s_2\}$ $(= \mathbf{start}(W_{\rm I}))$. $\mathbf{INS}(W_{\rm I}, s_1) = \{U_1^{\rm I}\}$ and $U_1^{\rm I}$ is a closed subgraph by Example 3.6.1. So, $\{s_1\}$ is an in-port of $W_{\rm I}$. Similarly, $\{s_2\}$ is also an in-port of $W_{\rm I}$. However, $\{s_1, s_2\}$ is not an in-port of $W_{\rm I}$, since $\{U_1^{\rm I}, U_2^{\rm I}\}$ is not conflict on any XOR-split in $W_{\rm I}$, but $U_1^{\rm I} \cup U_2^{\rm I}$ is not closed (cf. Examples 3.2.1 and 3.6.1).

(2) $W_{\rm II}$, too, has three non-empty subsets $\{s_1\}$, $\{s_2\}$ and $\{s_1, s_2\}$ of $\mathbf{start}(W_{\rm II})$. However, $\{s_1\}$ is not an in-port of $W_{\rm II}$, since $\{U_1^{\rm II}\}$ is not conflict on any XOR-split in $W_{\rm II}$, but $U_1^{\rm II}$ is not closed (cf. Examples 3.2.2 and 3.6.2). Similarly, $\{s_2\}$ is not an in-port of $W_{\rm II}$. Moreover, $\{s_1, s_2\}$ is not an in-port of $W_{\rm II}$, since $\{U_1^{\rm II}, U_3^{\rm II}\}$ is not conflict on any XOR-split in $W_{\rm II}$, but $U_1^{\rm II} \cup U_3^{\rm II}$ is not closed.

(3) $W_{\rm III}$ has two in-ports $\{s_1\}$ and $\{s_2\}$. However, $\{s_1, s_2\}$ is not an in-port of $W_{\rm III}$, since $\{U_1^{\rm III}, U_2^{\rm III}\}$ is not conflict on any XOR-split in $W_{\rm II}$, but $U_1^{\rm III} \cup U_2^{\rm III}$ is not closed (cf. Examples 3.2.3 and 3.6.3).

(4) $W_{\rm IV}$ has a single in-ports $\{s_1, s_2\}$ (cf. Examples 3.2.4 and 3.6.4).

**Definition 3.10** Let $W$ be a workflow and $\mathbb{I}$ a subset of the power set of $\mathbf{start}(W)$. Then, $W$ is said to satisfy extended correctness for $\mathbb{I}$ if the following properties hold.

(1) $\mathbb{I}$ is a set of some in-ports of $W$.

(2) $\mathbf{start}(W)$ is covered with $\mathbb{I}$, that is, every $s \in \mathbf{start}(W)$ is contained in some element of $\mathbb{I}$.

We call the $\mathbb{I}$ above a covering in-port family of $W$.

**Definition 3.11** A workflow $W$ is said to satisfy extended correctness if $W$ satisfies extended correctness for some covering in-port family.

**Definition 3.12** Let $W$ be a workflow.

(1) For an in-port $I$ of $W$, the set $\{\mathbf{end}(V) | V \in \mathbf{CL}(W, I)\}$ is called the out-port family of $W$ for $I$ and denoted by $\mathbb{O}(W, I)$.

(2) For an in-port family $\mathbb{I}$ of $W$, the set $\{(I, \mathbb{O}(W, I)) | I \in \mathbb{I}\}$ is called the out-port assignment of $W$ to $\mathbb{I}$ and denoted by $\mathbb{O}^*(W, \mathbb{I})$.

For the assignment $\mathbb{O}^*(W, \mathbb{I})$ of a workflow $W$ to an in-port family $\mathbb{I}$, $\bigcup \mathbb{O}^*(W, \mathbb{I})$ denotes $\bigcup_{I \in \mathbb{I}} \mathbb{O}(W, I)$, that is the set of all out-ports of $W$ for all in-ports in $\mathbb{I}$. That is, $\bigcup \mathbb{O}^*(W, \mathbb{I}) = \{\mathbf{end}(V) | V \in \mathbf{CL}(W, I)\ \&\ I \in \mathbb{I}\}$.

**Example 3.13** We explain extended correctness of workflows and out-port assignments of them, by using the previous four workflows in Figure 3.
(1) By Example 3.9.1, $W_{\rm I}$ satisfies extended correctness for $\{\{s_1\}, \{s_2\}\}$. Moreover,

$$\mathbb{O}^*(W_{\rm I}, \{\{s_1\}, \{s_2\}\}) = \{(\{s_1\}, \{\{e_1\}\}), (\{s_2\}, \{\{e_1\}, \{e_2\}\})\}.$$

(2) By Example 3.9.2, $W_{\rm II}$ does not satisfy extended correctness.
(3) In the same way as $W_{\rm I}$, $W_{\rm III}$ satisfies extended correctness for $\{\{s_1\}, \{s_2\}\}$. Moreover,

$$\mathbb{O}^*(W_{\rm III}, \{\{s_1\}, \{s_2\}\}) = \{(\{s_1\}, \{\{e_1\}\}), (\{s_2\}, \{\{e_1, e_2\}\})\}.$$

(4) By Example 3.9.4, $W_{\rm IV}$ satisfies extended correctness for $\{\{s_1, s_2\}\}$. Moreover,

$$\mathbb{O}^*(W_{\rm IV}, \{\{s_1, s_2\}\}) = \{(\{s_1, s_2\}, \{\{e_1, e_2\}\})\}.$$

## 4. Fundamental theorems of extended correctness

In this section, we show fundamental theorems of extended correctness. These theorems are utilized for incremental verification for large-scale workflows. Proofs of the theorems in this section are shown in Appendix A.

The first theorem shows that extended correctness is a conservative extension of original correctness.

**Theorem 4.1** For a workflow $W$ with a single start, $W$ is correct if and only if $W$ satisfies extended correctness.

Theorem 4.1 insures that extended correctness adequate property to be a natural extension of original correctness.

We next show that extended correctness is preserved by vertical composition and division of workflows. For simplicity, we fix workflows $W_1, W_2$, a non-empty subset $E_0$ of $\mathbf{end}(W_1)$, a non-empty subset $S_0$ of $\mathbf{start}(W_2)$, and assume that there exists a bijection $f : E_0 \to S_0$. We also identify $E_0$ with $S_0$ and abbreviate the vertical composition $W_1 *_f W_2$ to $W_1 * W_2$.

We first show the theorem above in a special case (Theorem 4.2), and then show that in the general case (Theorem 4.4).

**Theorem 4.2** Assume that $\mathbf{end}(W_1) = E_0 (= S_0) = \mathbf{start}(W_2)$ and let $\mathbb{I}$ be a covering family of $\mathbf{start}(W_1)$. Then, the vertical composition $W_1 * W_2$ satisfies extended correctness for $\mathbb{I}$ if and only if
(1) $W_1$ satisfies extended correctness for $\mathbb{I}$, and
(2) $W_2$ satisfies extended correctness for $\bigcup \mathbb{O}^*(W_1, \mathbb{I})$.

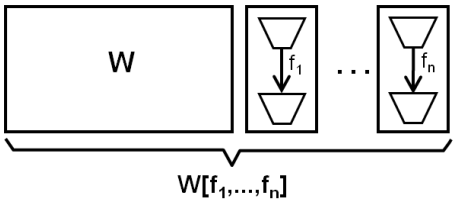


**Figure 4. Obvious extension of a workflow**

**Definition 4.3** Let $W$ be a workflow, and for $i = 1, \ldots, n$, let $f_i$ be an arc with source a start node and with target an end node. Then, the *obvious extension* of $W$ by $\{f_1, \ldots, f_n\}$, which is described by $W[f_1, \ldots, f_n]$, denotes the unconnected workflow obtained from $W$ by adding arcs $f_1, \ldots, f_n$.

We illustrate $W[f_1, \ldots, f_n]$ by Figure 4.

**Theorem 4.4** Let $S_2 := \mathbf{start}(W_2) - S_0$ [1] and $\mathbb{I}$ be a covering family of $\mathbf{start}(W_1) \cup S_2$. Then, the vertical composition $W_1 * W_2$ satisfies extended correctness for $\mathbb{I}$ if and only if
(1) $W_1$ satisfies extended correctness for

$$\{I \cap \mathbf{start}(W_1) | I \in \mathbb{I} \,\&\, I \cap \mathbf{start}(W_1) \neq \emptyset\}.$$

(2) $W_2$ satisfies extended correctness for

$$\{O \cap \mathbf{start}(W_2) \mid O \in \bigcup \mathbb{O}^*(W_1[f_1, \ldots, f_k], \mathbb{I}) \,\&\, O \cap \mathbf{start}(W_2) \neq \emptyset\},$$

where $f_1, \ldots, f_k$ denote arcs with source a start node and with target an end node (see Figure 5).

Theorems 4.2 and 4.4 claim that one can verify extended correctness of a workflow $W := W_1 * \cdots * W_n$ by calculating of the in-port families and the out-port assignments of $W_1, \ldots, W_n$. In the usual case, the calculation is not so complicated since most workflows have at most three start nodes (see Section 6).

**Example 4.5** Consider $W_{\mathrm{I}}$, $W_{\mathrm{III}}$ and $W_{\mathrm{IV}}$ in Figure 3. Then, by the function $f : \{e_1, e_2\} \to \{s_1, s_2\}$ with $f(e_1) = s_1$ and $f(e_2) = s_2$, one can consider nine vertical compositions $W_X * W_Y$, where $X$ and $Y$ are I, III or IV, respectively. By Example 3.13, $\bigcup \mathbb{O}^*(W_{\mathrm{I}}, \{\{s_1\}, \{s_2\}\}) = \{\{e_1\}, \{e_2\}\}$, $\bigcup \mathbb{O}^*(W_{\mathrm{III}}, \{\{s_1\}, \{s_2\}\}) = \{\{e_1\}, \{e_1, e_2\}\}$ and $\bigcup \mathbb{O}^*(W_{\mathrm{IV}}, \{\{s_1, s_2\}\}) = \{\{e_1, e_2\}\}$. Therefore, by Theorem 4.2, $W_{\mathrm{I}} * W_{\mathrm{I}}$, $W_{\mathrm{I}} * W_{\mathrm{III}}$ and $W_{\mathrm{IV}} * W_{\mathrm{IV}}$ satisfy extended correctness, but there is no other composition that satisfies extended correctness.

[1] For sets $X$ and $Y$, "$X - Y$" denotes the difference set $\{x \in X | x \notin Y\}$.

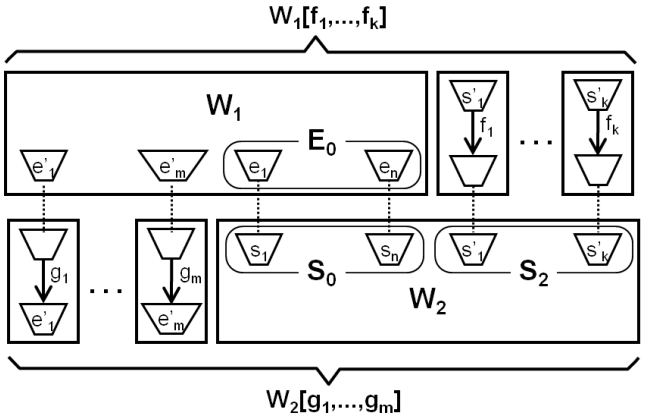


**Figure 5.** $W_1[f_1, \ldots, f_k] * W_2[g_1, \ldots, g_m]$ **(**$= W_1 * W_2$**)**

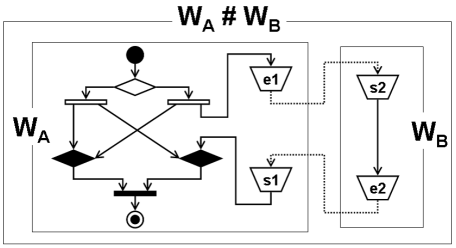


**Figure 6. Another type of composition**

On the other hand, since $W_{\mathrm{II}}$ does not satisfy extended correctness (see Example 3.13.2), one can obtain no workflow satisfying extended correctness by composing $W_{\mathrm{II}}$ and any workflow.

**Remark 4.6** It is not a trivial problem whether a consistency property of workflows is preserved in certain composition or division of them. As an example, we give another composition $W_A \sharp W_B$ of workflows $W_A$ and $W_B$ in the way of Figure 6. Note that there exists control flow between $W_A$ and $W_B$ in both directions. While $W_A$ does not satisfy extended correctness, $W_A \sharp W_B$ satisfies extended correctness. Therefore, the composition does not satisfy a property similar to Theorem 4.2 (or Theorem 4.4).

As the last part of this section, we define "extensible property" of workflows and show that the property is equivalent to extended correctness.

**Definition 4.7** For a workflow $W$, $W$ is said to be extensible if there exists a workflow $W_0$ such that $W_0 * W$ is correct.

**Theorem 4.8** For a workflow $W$, $W$ is extensible if and only if $W$ satisfies extended correctness.

If a workflow $W$ is extensible, it is possible that one can complete a correct workflow (with a single start) from $W$ by extending $W$ "vertically". On the other hand, Theorem 5.1 in [4] assures that one can modify a correct workflow $W$ with a single start and multiple ends to that with a single start and a single end, which is essentially equivalent to $W$. So, Theorem 4.8 assures that, if $W$ satisfies extended correctness, one can complete a correct workflow with a single start and a single end by extending $W$ vertically and modifying the extended workflow in the way in the proof of Theorem 5.1 in [4].

Theorem 4.8 also assures that, if a workflow $W$ does not satisfy extended correctness, one can not complete any correct workflow from $W$ by extending $W$ vertically. For example, since $W_{\rm II}$ in Figure 3 does not satisfy extended correctness, one can not complete any correct workflow from $W_{\rm II}$ by extending it vertically. This means that, if one likes to complete a correct workflow from $W_{\rm II}$, one has to modify $W_{\rm II}$ itself. So, it is useful to check extended correctness of an incomplete workflow (= a workflow with multiple starts and/or multiple ends) in the making of a correct workflow, since one may have an opportunity to modify structure of the incomplete workflow before it grows too large to modify the structure easily.

## 5. Consistency of evidence life cycles in a workflow with multiple starts

In the previous papers [12] and [14], we define a consistency property of life cycles of "evidences" in a workflow with a single start. We here define a similar consistency property for a workflow with multiple starts.

"evidence" is a technical term which means an annotation on a workflow, which denotes a document on which information is written, and/or with which something is approval, during the process of an operation. For simplicity, we often call such documents themselves "evidences". In large organizations such as large governments, evidences such as order forms, estimate sheets, invoices, and receipts play significant roles for purposes of feasibility, accountability, traceability, or transparency of business. Numerous actual operations are currently based on evidences even if they are carried out with information systems. Therefore, it is important to consider workflows in which one can concretely and precisely describe the life cycles of evidences to analyze requirements in developing large-scale information systems.

Roughly, the life cycles of evidences mean a series of states of the evidences, and consistency of evidence life cycles in a workflow means that the workflow has no inconsistent life cycles of evidences. In [12] and [14], we define a consistent property of evidence life cycles in a workflow with a single start, by using *instances* of the workflow. We here define that in a workflow with multiple starts, by using *closed* subgraphs of the workflow.

We precede the definition of the consistency property by that of evidences in a workflow.

### 5.1. Evidence

This subsection refers to [14]. We here regard an evidence as a paper document, which is composed, referred, re-written, judged, stored or dumped in some activities. Unlike data files, an evidence does not increase. Though one can make a copy of it, the copy is regarded not to be the same thing as the original evidence. Moreover, unlike data in a system multiple people can access simultaneously, an evidence can not be used by multiple people at the same time.

In the technical perspective, a list of evidences with length at least 0 is assigned to an activity, and an evidence $E$ is defined to be a triple $(e, created, removed)$, where $e$ is a label, and $created$ and $removed$ are boolean values. In what follows, we fix a non-empty set $\mathbb{E}$.

**Definition 5.1** *Evidence* is a triple $(e, created, removed)$, where $e$ is an element of $\mathbb{E}$ and $created$ and $removed$ are boolean values, that is, they are elements of $\{true, false\}$. For each evidence $E := (e, created, removed)$, we call $e$ the *evidence label* of $E$.

**Remark 5.2** For simplicity, we abbreviate evidences by the following ways.

(i) $(e, \mathrm{false}, \mathrm{false})$ is abbreviated to "$e$".

(ii) $(e, \mathrm{false}, \mathrm{true})$ is abbreviated to "$(-)e$".

(iii) $(e, \mathrm{true}, \mathrm{false})$ is abbreviated to "$(+)e$".

(iv) $(e, \mathrm{true}, \mathrm{true})$ is abbreviated to "$(+)(-)e$".

For a workflow $W$, we consider an allocation which assigns to each activity in $W$ a string of evidences. Note that such an allocation may assign to some activities the empty string, i.e., the string with length 0. By using workflows, one can express a lot of workflows. In order to explain evidences, we give an example of a workflow which explains how to submit a paper, as follows.

For each workflow $W$, each activity $A$ in $W$ and for each evidence $E$ in the string assigned to $A$, we call $E$ an evidence *on* $A$ and call $A$ an activity *having* $E$.

**Remark 5.3** In what follows, we assume that, for each workflow diagram $W$ and each activity $A$ in $W$, $A$ does not have multiple evidences sharing the same evidence label. We call the condition *the basic evidence condition*.

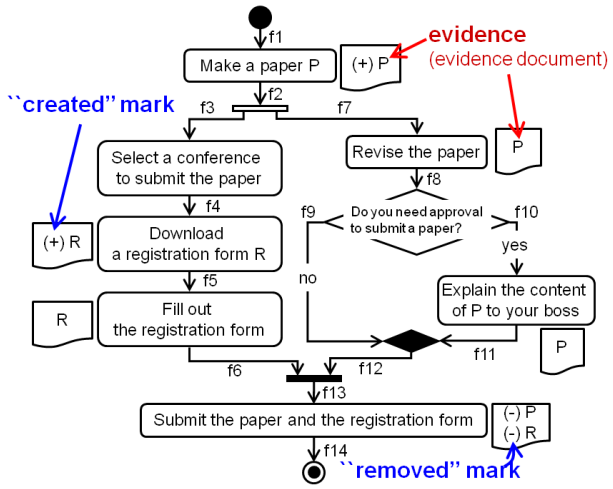


**Figure 7. Workflow of paper submission**

Since each workflow diagram $W$ is assumed to satisfy the basic evidence condition, if an activity $A$ in $W$ has an evidence label $e$, $A$ has just one evidence $E$ with label $e$. So, we often say that $e$ *is created (or removed) on* $A$ if $A$ has an evidence $E$ having the $(+)$-mark (or the $(-)$-mark, respectively).

**Example 5.4** In the workflow in Figure 7, a paper $P$ is created on the activity "Make a paper $P$", and it is removed on the activity "Submit the paper and the registration form". The evidence also appears on the activities "Revise the paper" and "Explain the content of $P$ to your boss".

## 5.2. Consistency property of evidence life cycles in a workflow with multiple starts

We here define a consistency of evidence life cycles in a workflow with multiple starts. This subsection also refers to [14].

Roughly, the "life cycle" of an evidence means that a series of states of the evidence. In order to define consistent life cycles of evidences in workflow in a rigorous manner, we introduce some new concepts.

**Definition 5.5** For a workflow $W$, a *line* in $W$ is a sequence of arcs in $W$

$$L = (A_1 \xrightarrow{f_1} A_2 \xrightarrow{f_2} \cdots \xrightarrow{f_{n-1}} A_n)$$

which satisfies the following properties.

(i) $A_1$ is an activity or the start in $W$.

(ii) $A_n$ is an activity or an end in $W$.

(iii) $A_2, \ldots, A_{n-1}$ are nodes in $W$, each of that is not any activity, the start, nor any end.

For a line $L$ above, $A_1$ is called the *source* of $L$, $A_n$ the *target* of $L$ and $f_{n-1}$ the *target arc* of $L$.

**Definition 5.6** A line $L$ is said to be *equivalent* to another line $L'$ if $L$ and $L'$ share the source and the target.

**Example 5.7** The workflow in Figure 7 has 10 lines, as follows: $(f1)$, $(f2f3)$, $(f2f7)$, $(f4)$, $(f5)$, $(f6f13)$, $(f8f9f12f13)$, $(f8f10)$, $(f11f12f13)$ and $(f14)$.

**Definition 5.8** A sequence $\pi$ of lines is said to be *equivalent* to another sequence $\pi'$ of lines if there exist lines $L_1, \ldots, L_n$ and $L'_1, \ldots, L'_n$ such that

$$\pi = (A_1 \xrightarrow{L_1} A_2 \xrightarrow{L_2} \cdots \xrightarrow{L_{n-1}} A_n)$$

$$\pi' = (A_1 \xrightarrow{L'_1} A_2 \xrightarrow{L'_2} \cdots \xrightarrow{L'_{n-1}} A_n)$$

and that, for each $i = 1, \ldots, n$, $L_i$ is equivalent to $L'_i$.

$L \sim L'$ (or $\pi \sim \pi'$) denotes that $L$ is equivalent to $L'$ ($\pi$ is equivalent to $\pi'$, respectively). Note that every line is equivalent to itself, and so is every sequence of lines.

**Definition 5.9** Let $W$ be a workflow, $V$ a closed subgraph of $W$ and let $e$ be an evidence in $W$. Then, the *consistent life cycle* of $e$ on $V$ is the sequence $\pi$ of lines in $V$

$$\pi := (A_0 \xrightarrow{L_0} A_1 \xrightarrow{L_1} \cdots \xrightarrow{L_{n-1}} A_n)$$

which satisfies the following properties.

(i) Every activity $A_i$ has $e$.

(ii) If $A_0$ is not the target of any line with source an input node, then $e$ is created on $A_0$.

(iii) $e$ is not created on $A_i$ for any $i$ with $0 < i \leqq n$.

(iv) If $A_n$ is not the source of any line with target an output node, then $e$ is removed on $A_n$.

(v) $e$ is not removed on $A_i$ for any $i$ with $i < n$.

**Definition 5.10** A workflow $W$ is said to *have consistent evidence life cycles* if, for each closed subgraph $V$ of $W$, each activity $A$ in $V$ and for each evidence $e$ on $A$, there is an essentially unique consistent life cycle $\pi$ of $e$ on $V$ which contains $A$.

The statement "there is an essentially unique consistent life cycle $\pi$ of $e$ on $V$ containing $A$" means that there is a consistent life cycle $\pi$ of $e$ on $V$ containing $A$ and that $\pi \sim \pi'$ for each consistent life cycle $\pi'$ of $e$ containing $A$.

**Example 5.11** The workflow in Figure 7 has two closed subgraph. The first closed subgraph (called $U$) consists of all nodes except the activity "Explain the content ..." and all arcs except $f10$ and $f11$. The second one (called $V$) consists of all nodes and all arcs except $f9$.

For an evidence $P$ in Figure 7, $U$ has just one consistent evidence life cycle of $P$: $((f2f7)(f8f9f12f13))$. $V$ also has just one consistent evidence life cycle of $P$: $((f2f7)(f8f10)(f11f12f13))$. For another evidence $R$, $U$ and $V$ share the same consistent evidence life cycle of $R$: $((f5)(f6f13))$. Moreover, they have no other consistent evidence life cycle of $R$. Therefore, the workflow in Figure 7 has consistent evidence life cycles.

For the consistency property of evidence life cycles in a workflow with multiple starts, one can also have similar theorems to those in Section 4. Actually, one can easily show the following theorems.

**Theorem 5.12** For a workflow with a single start, the original consistency property of evidence life cycles in the workflow (cf. [14], Definition 3.10) is equivalent to that in Definition 5.10.

**Theorem 5.13** Let $W_1$ and $W_2$ be workflows, $E_0 := \{A_1, \ldots, A_n\} \subset \mathbf{end}(W_1)$, $S_0 := \{B_1, \ldots, B_n\} \subset \mathbf{start}(W_2)$, and $f$ a bijection $E_0 \to S_0$ with $f(A_i) = B_i$. Then, $W_1 *_f W_2$ has consistent evidence life cycles if and only if so do $W_1$ and $W_2$ and, for any line $A \to A_i$ in $W_1$ and another line $B_i \to B$ in $W_2$, the following properties hold.

(i) For an evidence $E$ on $A$, if $E$ is not removed on $A$ and if $B$ is not any end node, then $B$ also has $E$ and $E$ is not created on $B$.

(ii) For an evidence $E$ on $B$, if $E$ is not created on $B$ and if $A$ is not any start node, then $A$ also has $E$ and $E$ is not removed on $B$.

The consistency of evidence life cycles in a workflow does not need extended correctness of the workflow. However, it is meaningless to define the consistency of evidence life cycles in a workflow which does not satisfy extended correctness. We show this claim by using a workflow $W_{\mathrm{II}}^*$ in Figure 8.

$W_{\mathrm{II}}^*$ has two closed subgraphs, both of which satisfies the conditions in Definition 5.10. So, $W_{\mathrm{II}}^*$ has consistent evidence life cycles. However, the structure of $W_{\mathrm{II}}^*$ is the same as that of $W_{\mathrm{II}}$ in Figure 3, and hence, $W_{\mathrm{II}}^*$ does not satisfy extended correctness. Actually, if the director returns the proposal $P$, the secretary can not receive it nor send it to the administration division. This example claims that, for a workflow which does not satisfy extended correctness, the semantics of the consistency of evidence life cycles in the workflow becomes ambiguous. Conversely, extended correctness of a workflow assures that the consistency of evidence life cycles in the workflow has the semantics we expect if one do not have to consider any set of start nodes which is not contained in the in-port family of the workflow.

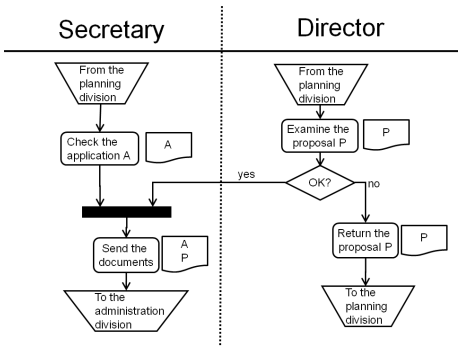


**Figure 8. Wrong workflow**

## 6. Discussion

In this section, in order to validate extended correctness, consistency of evidence life cycles and their fundamental theorems in Sections 4 and 5, we investigate real workflows and explain how to verify the consistency properties of a workflow by incremental verification.

### 6.1 Observations

We first investigate 154 workflows, which have been developed in requirement analysis for a real information system that helps one to manage personnel affairs. Each workflow has 10 to 30 nodes.

The observations are shown in the previous work [13].

**Observation 1** Among the 154 workflows above, there are 101 workflows that have connections to other workflows. For example, there exists a large workflow that consists of 12 small workflows. [2] We describe the large workflow in Figure 9, where $W1, \ldots, W12$ describe the small workflows in the large workflow. In this figure, we simplify the small workflows. Especially, we omit all activity nodes in the small workflows.

We also classify 154 workflows on the numbers of their start nodes. Then, we have the following result. The result claims that, in most cases, the maximal in-port family and its out-port assignment of a workflow are not very large.

[2] We often consider a "large workflow" to be a set of workflows that have connections to one another.

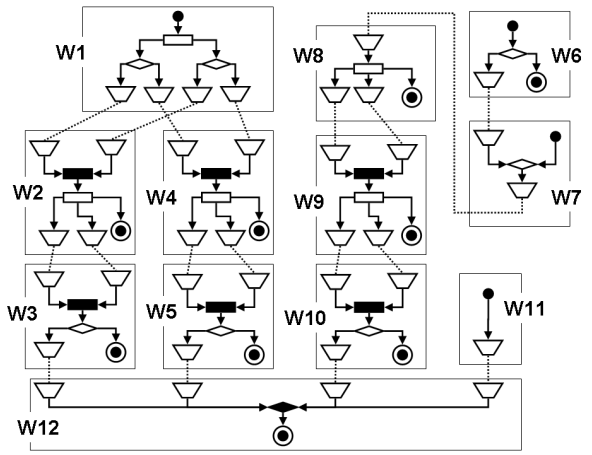


**Figure 9. Large workflow $W$**

| number of start nodes | 1 | 2 | 3 | 4 | 5 | 6 | 16 |
|---|---|---|---|---|---|---|---|
| number of workflows | 121 | 24 | 5 | 1 | 1 | 1 | 1 |

**Table 2. Classification of 154 workflows on the numbers of start nodes**

**Observation 2** In most cases, even if two workflows are connected to each other, there is only one-way control flow between the two workflows. For example, while there is control flow from $W2$ to $W3$ in Figure 9, there is no control flow from $W3$ to $W2$. As far as the 154 workflows above, there are about 80 connections of two workflows, but, there are only 2 connections that have control flow between workflows in both directions (one can see an example of such a connection in Figure 6). Therefore, at least as far as the 154 workflows, vertical composition sufficiently covers connections between workflows.

**Observation 3** As far as the 154 workflows, the order of development of the small workflows does not completely correspond to the direction of control flow of large workflows that consist of the small workflows. Moreover, there are some large workflows that plural engineers work together to develop. In fact, $W$ in Figure 9 has been developed by two system engineers. In a case like this, incremental verification is especially useful, since it is possible that one of the engineers can obtain only an incomplete set of workflows in $W$.

**Summary of the observations** The first observation claims that about two thirds of the 154 workflows are connected to other workflows and that about one fifth of the 154 workflows have multiple starts. The second observation claims that, while there are about 80 pairs that are vertically composed, only two pairs are composed but not vertically composed. This means that 97.5 percent of all pairs that are composed are vertically composed. The last observation claims that the order of development of small workflows does not completely correspond to the direction of control flow of a large workflow consisting of them. By the observations, one can claim that it is meaningful to consider incremental verification for a large-scale workflow that consists of small workflows with multiple starts or vertically composed.

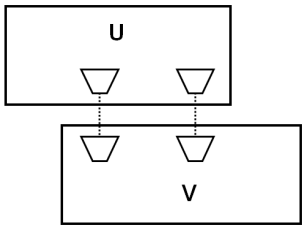


**Figure 10. Workflows $U$ and $V$**

### 6.2 Application

In order to validate extended correctness, consistency of evidence life cycles of a workflow and the fundamental theorems of them, we here explain how to verify the consistency properties by incremental verification, by using two workflows $U$ and $V$ in Figure 10, that are vertically composed.

As we explained in the third observation in the previous section, $U$ and $V$ have been developed regardless of the control flow of $U * V$. For example, assume that only $V$ has been developed. Dislike original correctness, one can verify control flow consistency of $V$ based on extended correctness. Let $V$ do not satisfy extended correctness. Then, by Theorems 4.2 and 4.4, whatever one develops $U$, $U * V$ will never satisfy extended correctness. This means that he/she should modify $V$ at this point when $V$ turn out not to satisfy extended correctness. Similarly, one can verify consistency of evidence life cycles of $V$ even if $V$ has multiple starts. He/She should also modify evidence life cycles of $V$ at this point when $V$ turn out not to have consistent evidence life cycles.

Moreover, assume that $V$ has been modified to satisfy extended correctness (and to have consistent evidence life cycles) and that $U$ has been developed additionally. If the workflow $U$ does not satisfy extended correctness or consistency of evidence life cycles, then neither does $U * V$. Therefore, he/she should modify $U$ at this point. If the workflow $U$ satisfies extended correctness, he/she can know whether or not $U * V$ satisfies extended correctness, by checking conditions of the in-port families of $V$ and the out-port assignments of $U$ (see the comment immediately after Theorem 4.2). Similarly, If the workflow $U$ has consistent evidence life cycles, he/she can know whether or not

$U * V$ has consistent evidence life cycles, by checking the conditions (i) and (ii) in Theorem 5.13.

As above, one can develop and verify the large workflow $U * V$ in parallel. Incremental verification is a useful approach, especially in the case of development of large-scale workflows.

## 7. Related work

The definition and fundamental theorems of extended correctness are based on the previous work [13]. In this paper, we show Theorem 4.4 by a simpler way than that of the similar theorem in [13].

There are a lot of researches of consistency properties of workflows in the viewpoint of control flow of them such as Aalst [15], Sadiq and Orlowska [9], Aalst [16], Sadiq and Orlowska [10], Verbeek et al. [21], Lin et al. [6], and Aalst et al. [17]. However, these researches deal with verification for a workflow with a single start and a single end as a complete workflow. In fact, a workflow in standard workflow languages such as XPDL [23] and YAWL [19] has a single start and a single end.

An open workflow net by Aalst et al. [18] can be considered to be a workflow with multiple starts and ends, and their "weak termination" essentially corresponds to soundness (correctness). Another well known workflow language EPC [3] has workflows with multiple starts and ends. By Mendling and Aalst [8], a semantics of EPC is given. Based on the notions above, one can obtain another extended correctness over (acyclic and cyclic) workflows with multiple starts and ends. However, the important point in this paper is that our extended version of correctness is a conservative extension of original correctness and it is preserved in vertical composition and division of workflows. These theorems are important for incremental verification based on the extended correctness. Since [18] and [8] have different purposes from ours, they do not show similar results about their extended correctness properties to our properties above. It is expected to show similar theorems based on correctness properties in [18] and [8].

By Dehnert and Aalst [2], Dongen et al. [20], and Mendling et al. [7], verification systems of consistency properties of workflows in EPC are developed. However, in order to verify consistency of workflows in EPC with the systems, users have to set start nodes which are fired at the initial point or the systems have to check all combinations of start nodes fired at the initial point. So, these approaches differ from ours.

A standard workflow model by Kiepuszewski et al. [4] also may have multiple starts and/or multiple ends. However, the semantics of the workflows in [4] is based on an assumption that a petri net modeling a workflow has a token on each initial place in every initial marking. Therefore, the correctness property of workflows defined in [4] is essentially the same as original correctness.

Kindler et al. [5] investigate "local soundness" for each sub-workflow in a workflow and "global soundness" for the whole workflow. The verification approach in [5] uses "scenario" that are used to verify global soundness of a workflow $W$ from verification of local soundness of sub-workflows constituting $W$. So, the approach verifies a workflow based on necessary data for the workflow instead of the set of all sub-workflows of the workflow. Moreover, the ways to divide or compose workflows differ from ours.

Siegeris and Zimmermann [11] also investigate correctness properties of workflows to verify consistency of a whole workflow based on verifications of that of sub-workflows of the workflow. The verification approach for a workflow is based on verification for all sub-workflows constituting the workflow. Moreover, the ways to divide or compose workflows in [11] differ from ours, too.

On the other hand, in the previous papers [12] and [14], we investigate consistency of evidence life cycles in a workflow with a single start. We extend the previous work for a workflow with multiple starts.

Wang and Kumar [22] investigate document-driven workflow systems, where "documents " are essentially the same concept as evidences. They propose a framework for designing and managing workflows based on structures and states of documents. While our framework manages control-flow based workflows with evidences, their framework manages document-driven workflows. Thus, the meaning of the verification for their workflows differs from that of consistency of evidence life cycles in control-flow based workflows.

## 8. Conclusion

In this paper, we extend the results in our previous work [13], by adding consistency property of evidence life cycles in a workflow with multiple start nodes. The consistency property of evidence life cycles is based on that in a workflow with a single start node in [12].

The purpose of this paper is to develop an incremental verification methodology for large-scale workflows. As a basis for the verification methodology, we have defined extended correctness of an acyclic workflow with multiple starts and multiple ends. Extended correctness is a conservative extension of original correctness property over an acyclic workflow with a single start (Theorem 4.1). We also consider vertical composition and division of workflows, and show that extended correctness is preserved in these operations on workflows (Theorems 4.2 and 4.4). We also characterize extended correctness of a workflow as extensible property (Theorem 4.8).

In Section 5, we define a consistency property of evidence life cycles in a workflow with multiple starts, and show two fundamental theorems of the consistency (Theorems 5.12 and 5.13).

In Section 6, we investigate real 154 workflows in order to validate incremental verification for a large-scale workflow that consists of small workflows with multiple starts and/or vertically composed. Moreover, in order to validate extended correctness, consistency of evidence life cycles of a workflow and the fundamental theorems of them, we explain how to verify the consistency properties by using an example.

Since the workflow language in this paper is simple and conventional, one can apply the definitions and the theorems of the consistency properties in this paper for incremental verification for acyclic workflows in other languages such as BPMN [1] and XPDL.

Extended correctness, consistency of evidence life cycles and their theorems in this paper enable us to develop a concrete method to the consistency properties of large-scale workflows incrementally. Our next challenge is to develop a tool that helps one to verify large-scale workflows by the incremental methodology.

## References

[1] Business Process Management Initiative (BPMI). *Business Process Modeling Notation (BPMN) Version 1.0*. Technical report, BPMI.org, 2004.

[2] J. Dehnert and W. M. P. van der Aalst. Bridging the gap between business models and workflow specifications. *International Journal of Cooperative Information Systems*, 13(3):289–332, 2004.

[3] G. Keller, M. Nuttgens, and A. W. Scheer. *Semantische Prozessmodellierung auf der Grundlage Ereignisgesteuerter Prozessketten (EPK)*. Technical Report 89, Institut fur Wirtschaftsinformatik Saarbrucken, Saarbrucken, Germany, 1992.

[4] B. Kiepuszewski, A. H. M. ter Hofstede, and W. M. P. van der Aalst. Fundamentals of control flow in workflows. *Acta Informatica*, 39(3):143–209, 2003.

[5] E. Kindler, A. Martens, and W. Reisig. Inter-operability of workflow applications: Local criteria for global soundness. In *Business Process Management: Models, Techniques, and Empirical Studies (BPM)*, LNCS 1806, pages 235–253. Springer, 2000.

[6] H. Lin, Z. Zhao, H. Li, and Z. Chen. A novel graph reduction algorithm to identify structural conflicts. In *Proceedings of the 35th Annual Hawaii International Conference on System Science (HICSS)*. IEEE Computer Society Press, 2002.

[7] J. Mendling, M. Moser, G. Neumann, H. M. W. Verbeek, B. F. van Dongen, and W. M. P. van der Aalst. Faulty epcs in the sap reference model. In *International Conference on Business Process Management (BPM)*, LNCS 4102, pages 451–457. Springer, 2006.

[8] J. Mendling and W. M. P. van der Aalst. Formalization and verification of epcs with or-joins based on state and context. In *Proceedings of the 19th International Conference on Advanced Information Systems Engineering (CAiSE)*, LNCS 4495, pages 493–453. Springer, 2007.

[9] W. Sadiq and M. E. Orlowska. On correctness issues in conceptual modeling of workflows. In *Proceedings of the 5th European Conference on Information Systems (ECIS)*, pages 943–964, 1997.

[10] W. Sadiq and M. E. Orlowska. Analyzing process models using graph reduction techniques. *Information Systems*, 25(2):117–134, 2000.

[11] J. Siegeris and A. Zimmermann. Workflow model compositions preserving relaxed soundness. In *Proceedings of 4th International Conference on Business Process Management (BPM)*, LNCS 4102, pages 177–192. Springer, 2006.

[12] O. Takaki, T. Seino, I. Takeuti, N. Izumi, and K. Takahashi. Verification algorithm of evidence life cycles in extended UML activity diagrams. In *Proceedings of The 2nd International Conference on Software Engineering Advances (ICSEA 2007)*. IEEE Computer Society Press, 2007.

[13] O. Takaki, T. Seino, I. Takeuti, N. Izumi, and K. Takahashi. Incremental verification of large scale workflows based on extended correctness. In *Proceedings of the 3rd International Conference on Software Engineering Advances (ICSEA 2008)*. IEEE Computer Society Press, 2008.

[14] O. Takaki, T. Seino, I. Takeuti, N. Izumi, and K. Takahashi. Verification of evidence life cycles in workflow diagrams with passback flows. *International Journal On Advances in Software*, 1(1), 2008 (to appear).

[15] W. M. P. van der Aalst. Verification of workflow nets. In *Application and Theory of Petri Nets 1997*, LNCS 1248, pages 407–426. Springer, 1997.

[16] W. M. P. van der Aalst. The application of petri nets to workflow management. *The Journal of Circuits, Systems and Computers*, 8(1):21–66, 1998.

[17] W. M. P. van der Aalst, A. Hirnschall, and H. M. W. Verbeek. An alternative way to analyze workflow graphs. In *Proceedings of the 14th International Conference on Advanced Information Systems Engineering (CAiSE)*, LNCS 2348, pages 535–552. Springer, 2002.

[18] W. M. P. van der Aalst, N. Lohmann, P. Massuthe, C. Stahl, and K. Wolf. From public views to private views: Correctness-by-design for services. In *Informal Proceedings the 4th International Workshop on Web Services and Formal Methods (WS-FM)*, LNCS 4937, pages 139–153. Springer, 2007.

[19] W. M. P. van der Aalst and A. H. M. ter Hofstede. YAWL: Yet another workflow language. *Information Systems*, 30(4):245–275, 2005.

[20] B. F. van Dongen, W. M. P. van der Aalst, and H. M. W. Verbeek. Verification of epcs: Using reduction rules and petri nets. In *Proceedings of the 17th Conference on Advanced Information Systems Engineering (CAiSE)*, LNCS 3520, pages 372–386. Springer, 2005.

[21] H. M. W. Verbeek, T. Basten, and W. M. P. van der Aalst. Diagnosing workflow processes using woflan. *The Computer Journal*, 44(4):246–279, 2001.

[22] J. Wang and A. Kumar. A framework for document-driven workflow systems. In *Proceedings of 3rd International Conference on Business Process Management (BPM)*, LNCS 3649, pages 285–301. Springer, 2005.

[23] Workflow Management Coalition (WfMC). *Workflow Management Coalition Workflow Standard: Workflow Process Definition Interface - XML Process Definition Language (XPDL)*. (WfMC-TC-1025), Technical report, Workflow Management Coalition, Lighthouse Point, Florida, USA, 2002.

## A. Basic theorems and proofs of theorems

In this section, we show some basic theorems of extended correctness and theorems in Section 4.

**Definition A.1** Let $W$ be a workflow and $C$ the set of all XOR-splits on $W$. Then, a phenomenon on $W$ denotes a function $\psi : C \to \mathbf{arc}(W)$ satisfying that $\psi(c)$ is an outgoing-arc of $c$ for each $c \in C$.

**Lemma A.2** For a start $s$ and a phenomenon $\psi$, there exists a unique instance for $\psi$ from $s$, that is, there exists a unique instance $V$ from $s$ such that for every XOR-split $c$ in $V$ the outgoing-arc of $c$ in $V$ is $\psi(c)$.

**Proof**. Trivial. □

**Lemma A.3** Let $W$ be a workflow, $s$ a start in $W$, $\pi$ a path on $W$ from $s$ and $\psi$ a phenomenon on $W$. Moreover, assume that, for an XOR-split $c$ in $W$, if $c$ is the source of an arc in $\pi$, then $\psi(c)$ is contained in $\pi$. Then, the instance of $W$ for $\psi$ from $s$ contains $\pi$.

**Proof**. By induction on the length of $\pi$. □

**Lemma A.4** For a workflow $W$, $W$ is covered with $\mathbf{INS}(W)$, that is, every arc $f$ in $W$ is contained in some instance of $W$.

**Proof**. Let $f$ be an arc in $W$. Then, there exists a path $\pi$ with first element the outgoing-arc of some start $s$ of $W$ and last element $f$. So, by Lemma A.3, $\pi$ is contained in some instance in $\mathbf{INS}(W, s)$. □

**Lemma A.5** For a workflow $W$, every closed subgraph of $W$ is deadlock free and lack of synchronization free.

**Proof**. Trivial. □

**Proposition A.6** Let $W$ be a workflow and $\{U_1, \ldots, U_n\}$ a set of some instances in $W$ that is not conflict on any XOR-split in $W$. Then, $U := U_1 \cup \cdots U_2$ is closed if and only if $U$ is deadlock free and lack of synchronization free.

**Proof**. By Lemma A.5, $U$ is deadlock free and lack of synchronization free if $U$ is closed. So, we assume that $U$ is deadlock free and lack of synchronization free and show that $U$ is closed.
(1) Let $x$ be an XOR-split. Then, since $x$ is contained in some instance $U_i$, the incoming-arc of $x$ and some outgoing-arc(s) of $x$ are contained in $U$. Moreover, since $\{U_1, \ldots, U_n\}$ is not conflict on any XOR-split in $W$, the outgoing-arc of $x$ that is contained in $U$ is single.
(2) Let $x$ be an XOR-join. Then, since $x$ is contained in some instance $U_i$, the outgoing-arc of $x$ and some incoming-arc(s) of $x$ are contained in $U$. Moreover, since $U$ is lack of synchronization free, the incoming-arc of $x$ in $U$ is single.
(3) Let $x$ be an AND-join. Then, since $x$ is contained in some instance $U_i$ and $U$ is deadlock free, the outgoing-arc of $x$ and all incoming-arcs of $x$ are contained in $U$.
(4) Let $x$ be another type node. Then, since $x$ is contained in some instance $U_i$, all incoming-arcs and outgoing-arcs of $x$ are contained in $U$. □

**Lemma A.7** Let $W$ be a workflow and $I$ an in-port of $W$.
(1) For every $s \in I$ and $U \in \mathbf{INS}(W, s)$, there exists a closed subgraph $V$ with $V \supseteq U$.
(2) For every $s \in I$ and $U \in \mathbf{INS}(W, s)$, $U$ is lack of synchronization free.

**Proof**. (1) Let $\psi$ be a phenomenon such that $U$ is the instance for $\psi$ from $s$. Then, for each $s_i \in I$ there exists the instance $U_i$ for $\psi$ from $s_i$. Since $\{U_1, \ldots, U_n\}$ does not conflict on any XOR-split, $U_1 \cup \cdots \cup U_n$ is closed. Moreover, for some $i \leqq n$, $s = s_i$ and hence $U = U_i$ by Lemma A.2.
(2) By (1) above, there is a closed subgraph $V$ with $V \supset U$. Thus, we have the result since $V$ is lack of synchronization free by Lemma A.5. □

**Lemma A.8** For a workflow $W$ satisfying extended correctness and a covering in-port family $\mathbb{I}$ of $W$, $W$ is covered with $\bigcup_{I \in \mathbb{I}} \mathbf{CL}(W, I)$. Especially, $W$ is covered with $\mathbf{CL}(W)$.

**Proof**. By Lemmas A.4 and A.7.(1). □

**Lemma A.9** Let $W$ be a workflow satisfying extended correctness and $\mathbb{I}$ a covering in-port family of $W$. Then, $\mathbf{end}(W)$ is covered by $\bigcup \mathbb{O}^*(W, \mathbb{I}) := \bigcup_{I \in \mathbb{I}} \mathbb{O}(W, I)$.

**Proof**. By Lemma A.8. □

**Lemma A.10** For a workflow $W$ with a single start, if $W$ satisfies extended correctness, then $W$ is correct.

**Proof**. Let $W$ have only a single start $s$. Then, $W$ has a single in-port $\{s\}$. So, by Def.3.8, every instance is a closed subgraph. Thus, by Lemma A.5, we have the result. □

**Lemma A.11** Every correct workflow satisfies extended correctness.

**Proof**. Every instance of a correct workflow $W$ is a closed subgraph of $W$. So, for the start $s$ of $W$, $\{s\}$ is the in-port of $W$. □

**Proof of Theorem 4.1** By Lemmas A.10 and A.11. □

**Definition A.12** Let $W_1 \in \mathbf{WF}(n,m)$ and $W_2 \in \mathbf{WF}(m,l)$.
(1) Let $V$ be a subgraph of $W_1 * W_2$. Then, $V\lceil W_1$ denotes the subgraph of $W_1$ uniquely obtained from $V \cap W_1$ by adding all possible ends in $W_1$ that correspond to connecting-arcs contained in $V$. Similarly, $V\lceil W_2$ denotes the subgraph of $W_2$ uniquely obtained from $V \cap W_2$ by adding all possible starts in $W_2$ that correspond to connecting-arcs contained in $V$.
(2) For a subgraph $V_1$ of $W_1$ and a subgraph $V_2$ of $W_2$, if there exists a subgraph $V$ of $W_1 * W_2$ with $V\lceil W_1 = V_1$ and $V\lceil W_2 = V_2$, then $V_1$ and $W_2$ are said to be able to be composed, and $V_1 * V_2$ denotes the subgraph $V$ above.

**Lemma A.13** Let $W_1 \in \mathbf{WF}(n,m)$ and $W_2 \in \mathbf{WF}(m,l)$.
(1) For each $S \subset \mathbf{start}(W_1 * W_2)$ and each $V \in \mathbf{CL}(W_1 * W_2, S)$, $V\lceil W_1$ is a closed subgraph in $\mathbf{CL}(W_1, S)$ and $V\lceil W_2$ is that in $\mathbf{CL}(W_2, \mathbf{end}(V\lceil W_1))$.
(2) Conversely, for each $V_1 \in \mathbf{CL}(W_1)$ and for each $V_2 \in \mathbf{CL}(W_2, \mathbf{end}(V_1))$, if $V_1$ and $\mathbb{V}$ can be composed in $W_1 * W_2$, then $V_1 * V_2$ is a closed subgraph of $W_1 * W_2$.

**Proof**. Trivial. □

**Lemma A.14** Let $W_1 \in \mathbf{WF}(n,m)$ and $W_2 \in \mathbf{WF}(m,l)$. If $W_1$ satisfies extended correctness for a covering in-port family $\mathbb{I}$ and if $W_2$ satisfies extended correctness for $\bigcup \mathbb{O}^*(W_1, \mathbb{I}) := \bigcup_{I \in \mathbb{I}} \mathbb{O}(W_1, I)$, then $W_1 * W_2$ satisfies extended correctness for $\mathbb{I}$.

**Proof**. We show that each $I \in \mathbb{I}$ is an image of $W_1 * W_2$. Let $I := \{s_1, \ldots, s_n\} \in \mathbb{I}$ and $U_i \in \mathbf{INS}(W_1 * W_2, s_i)$ $(i = 1, \ldots, n)$, and assume that $\{U_1 \ldots, U_n\}$ is not conflict on any XOR-split in $W_1 * W_2$. Then, $U_i\lceil W_1 \in \mathbf{INS}(W_1, s_i)$ for each $i \leqq n$, and $\{U_1\lceil W_1 \ldots, U_n\lceil W_1\}$ is not conflict on any XOR-split in $W_1$. Since $I$ is an image of $W_1$, $U^1 := U_1\lceil W_1 \cup \cdots \cup U_n\lceil W_1$ is a closed subgraph of $W_1$.

On the other hand, $U^2 := U_1\lceil W_2 \cup \cdots \cup U_n\lceil W_2$ consists of instances in $W_2$, that have elements of $\mathbf{end}(U^1)$ as the starts. Moreover, the set of all the instances in $W_2$ above is not conflict on any XOR-split in $W_2$. Therefore, since $\mathbf{end}(U^1) \in \mathbb{O}(W_2, I) \subset \bigcup \mathbb{O}^*(W_2, \mathbb{I})$, $U^2$ is a closed subgraph of $W_2$. Since $U = U_1 * U_2$, by Lemma A.13.2, $U$ is a closed subgraph of $W_1 * W_2$. So, we have the result. □

**Lemma A.15** Let $W_1 \in \mathbf{WF}(n,m)$ and $W_2 \in \mathbf{WF}(m,l)$. If $W_1 * W_2$ satisfies extended correctness for a covering in-port family $\mathbb{I}$, then so is $W_1$ and $W_2$ satisfies extended correctness for $\bigcup \mathbb{O}^*(W_1, \mathbb{I})$.

**Proof**. We first show that $W_1$ satisfies extended correctness for $\mathbb{I}$. Let $I := \{s_1, \ldots, s_n\} \in \mathbb{I}$ and $U_i \in \mathbf{INS}(W_1, s_i)$ for $i \leqq n$. Moreover, assume that $\{U_1, \ldots, U_n\}$ is not conflict on any XOR-split in $W_1$. Then, there exists a phenomenon $\psi_1$ on $W_1$ such that each $U$ is the instance for $\psi_1$ from $s_i$. So, we can have a phenomenon $\psi$ on $W_1 * W_2$ such that the restriction of $\psi$ to $W_1$ is $\psi_1$. Thus, for each $i \leqq n$, there exists the instance $U_i^*$ of $W_1 * W_2$ for $\psi$ from $s_i$. So, $U_i^*\lceil W_1 = U_i$ for each $i \leqq n$, and $\{U_1^*, \ldots, U_n^*\}$ is not conflict on any XOR-split on $W_1 * W_2$. So, since $I$ is an image of $W_1 * W_2$, $U_1^* \cup \cdots \cup U_n^*$ is a closed subgraph of $W_1 * W_2$. Therefore, by Lemma A.13.1, $U_1 \cup \cdots \cup U_n$ is a closed subgraph of $W_1$. So, $I$ is an image of $W_1$.

We next show that $W_2$ satisfies extended correctness for $\bigcup \mathbb{O}^*(W_1, \mathbb{I})$. Let $E$ is an element $\{e_1, \ldots, e_m\}$ of $\bigcup \mathbb{O}^*(W_1, \mathbb{I})$ and $U_i \in \mathbf{INS}(W_2, e_i)$ for $i \leqq m$. Moreover, assume that $\mathbf{U} := \{U_1, \ldots, U_m\}$ is not conflict on any XOR-split in $W_2$. Then, there exists a closed subgraph $V_1$ in $W_1$ such that $\mathbf{start}(V_1) \in \mathbb{I}$ and that $\mathbf{end}(V_1) = E$. So, we can have instances $U_1^1, \ldots, U_k^1$ in $W_1$ such that $V_1 = U_1^1 \cup \cdots \cup U_k^1$. For each $j \leqq k$, we have the subset $\mathbf{U}_j$ of $\mathbf{U}$ by $\mathbf{U}_j := \{U \in \mathbf{U} : \mathbf{start}(U) \in \mathbf{end}(U_j^1)\}$. Then, for each $j \leqq k$, $U_j^* := U_j^1 * (\bigcup \mathbf{U}_j)$ is an instance of $W_1 * W_2$, and $\{U_1^*, \ldots, U_k^*\}$ is not conflict on any XOR-split on $W_1 * W_2$. So, since $W_1 * W_2$ satisfies extended correctness for $\mathbb{I}$, $U_1^* \cup \cdots \cup U_k^*$ is a summation in $W_1 * W_2$. Therefore, by Lemma A.13.1, $U_1 \cup \cdots \cup U_m = (U_1^* \cup \cdots \cup U_k^*)\lceil W_2$ is a summation in $W_2$, and hence, $E$ is an image of $W_2$. □

**Proof of Theorem 4.2** By Lemmas A.14 and A.15. □

**Proof of Theorem 4.4** By the definition of a vertical composition (Definition 2.4), $W_1 * W_2$ is the same as $W_1[f_1, \ldots, f_k] * W_2[g_1, \ldots, g_m]$ in Figure 5 (see also Figure 2). So, the extended correctness of $W_1 * W_2$ for $\mathbb{I}$ is equivalent to that of $W_1[f_1, \ldots, f_k] * W_2[g_1, \ldots, g_m]$ for $\mathbb{I}$. So, by Theorem 4.2, it is equivalent to the following properties (i) and (ii).

(i) $W_1[f_1, \ldots, f_k]$ satisfies extended correctness for $\mathbb{I}$

(ii) $W_2[g_1, \ldots, g_m]$ satisfies extended correctness for $\bigcup \mathbb{O}^*(W_1[f_1, \ldots, f_k], \mathbb{I})$.

Now we first show that the property (i) above is equivalent to (1) in Theorem 4.4. For a subgraph $V$ of $W_1[f_1, \ldots, f_k]$, $V$ is closed in $W_1[f_1, \ldots, f_k]$ if and only if $V \cap W_1$ is closed in $W_1$. Thus, for an element $I$ of $\mathbb{I}$ with

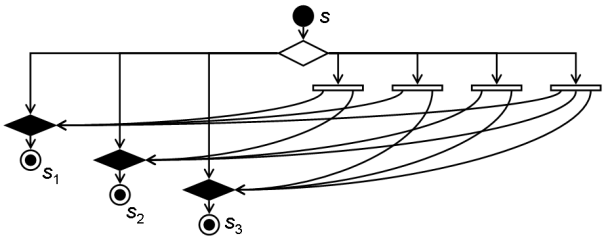


**Figure 11. Workflow** $W_0$

$I \cap \mathbf{start}(W_1) \neq \emptyset$, $I$ is an in-port of $W_1[f_1, \ldots, f_k]$ if and only if $I \cap \mathbf{start}(W_1)$ is that of $W_1$. Therefore, since $\mathbb{I}$ is a covering family of $\mathbf{start}(W_1[f_1, \ldots, f_k])$, the property (i) is equivalent to (1) in Theorem 4.4.

The property (i) implies that $\bigcup \mathbb{O}^*(W_1[f_1, \ldots, f_k], \mathbb{I})$ is a covering family of $\mathbf{start}(W_2[g_1, \ldots, g_m])$. Therefore, when (i) holds, one can show that the property (ii) is equivalent to (2) in Theorem 4.2 in the similar way to the case of (i) above. Thus, the extended correctness of $W_1 * W_2$ for $\mathbb{I}$ is equivalent to the properties (1) and (2) in Theorem 4.4. □.

**Lemma A.16** For a non-empty finite set $S$ and a subset $\mathbb{S}$ of the power set of $S$ with $S = \bigcup \mathbb{S}$, there exists a correct workflow $W$ that has a single start $s$ and that $\mathbb{S} = \mathbb{O}(W, \{s\})$.

**Sketch of the Proof**. Instead of showing this lemma directly, we give an example $S := \{s_1, s_2, s_3\}$ and $\mathbb{S} := \{\{s_1\}, \{s_2\}, \{s_3\}, \{s_1, s_2\}, \{s_1, s_3\}, \{s_2, s_3\}, S\}$ (=the power set of $S$), and illustrate a workflow $W_0$ satisfying the properties in this lemma for the $S$ and $\mathbb{S}$ above by Figure 11. (All workflows satisfying the properties in this lemma can be constructed in similar forms to $W_0$.)

Each outgoing-arc of the XOR-split $c$ in $W_0$ (see Figure 11) corresponds to an element of $\mathbb{S}$. Moreover, the numbers of outgoing-arcs of AND-splits $x_1 \sim x_4$ in $W_0$ correspond to the numbers of elements of $\{s_1, s_2\}$, $\{s_1, s_3\}$, $\{s_2, s_3\}$ and $S$, respectively. Note that outgoing-arcs corresponding to $\{s_1\}$, $\{s_2\}$ and $\{s_3\}$ do not have any AND-split, since $\{s_1\}$, $\{s_2\}$ and $\{s_3\}$ have a single element, respectively.

Obviously, $W_0$ is correct, and the ex-port family of $W_0$ for the in-port $\{s\}$ is $\mathbb{S}$. □

**Proof of Theorem 4.8** By Theorem 4.2 and Lemma A.16. □