
Originally published in S. Dustdar, J. L. Fiadeiro, & A. P. Sheth (eds.).

Available from: http://dx.doi.org/10.1007/11841760_4

Copyright © Springer-Verlag Berlin Heidelberg 2006.
This is the author's version of the work, posted here with the permission of the publisher for your personal use. No further distribution is permitted. You may also be able to access the published version from your library. The definitive version is available at http://www.springerlink.com/.
Tracking over Collaborative Business Processes

Xiaohui Zhao and Chengfei Liu

Centre for Information Technology Research
Faculty of Information and Communication Technologies
Swinburne University of Technology
Melbourne, Victoria, Australia
{xzhao, cliu}@it.swin.edu.au

Abstract. Workflow monitoring is a routine function of a workflow management system for tracking the progress of running workflow instances. To keep participating organisations as autonomous entities in an inter-organisational business collaboration environment, however, it brings challenges in generating workflow tracking structures and manipulating instance correspondences between different participating organisations. Aiming to tackle these problems, this paper proposed a matrix based framework on the basis of our relative workflow model. This framework enables a participating organisation to derive tracking structures over its relative workflows and the involved relevant workflows of its partner organisations, and to perform workflow tracking with the generated tracking structures.

1 Introduction

With the trend of booming global business collaborations, organisations are required to streamline their business processes into dynamic virtual organisations [1, 2]. A virtual organisation defines the trading community of a set of participating organisations for conducting collaborative business processes. Normally, the building blocks of a collaborative business process are the pre-existing business processes of participating organisations. Therefore, it is fundamental that a collaborative business process knows how the business process belonging to different organisations are linked together for cooperation [3, 4]. While this kind of cooperation is a prerequisite, organisations must act as autonomous entities during business collaboration. Besides, certain levels of privacy of participating organisations have to be guaranteed. Many existing inter-organisational workflow approaches streamline the related business processes of different organisations, into a public view workflow process [5-9]. This public view neutralises the diversity of the perception on collaborative business processes from different organisations, and fails to support business privacy sufficiently. We reckon that different organisations may see different pictures of a collaborative business process, and may need to know and be only allowed to know certain details of the collaboration with their partner organisations. To support this, we have proposed a new approach for collaborative business process modelling called relative workflow model [10]. In this model, different visibility constraints, perceptions, and relative workflow processes can be defined for different participating organisations.

Most traditional workflow monitoring approaches, such as WfMC Monitor and Audit specification [11, 12], BEA Weblogic Integration [13], IBM WebSphere MQ Workflow
the agent based workflow monitoring [15] and the customisable workflow monitoring [16], are mainly applicable either in an intra-organisational setting or in an environment where a public view of a collaborative business process is assumed without privacy concern. To our best knowledge, there is little discussion on workflow monitoring in an inter-organisational environment concerning privacy. This paper aims to fill this gap. Based on the relative workflow model, the tracking structure for a relative workflow process is defined and a matrix based framework is proposed to enable a participating organisation to derive tracking structures over its relative workflow processes and the involved relevant workflow processes of its partner organisations, and to perform tracking based on the generated tracking structures.

The remainder of this paper is organised as follows. Section 2 analyses requirements of workflow tracking in a privacy sensitive environment with a motivating example. In Section 3, we first review our relative workflow approach, then introduce some representation matrices, after that we define the tracking structure of a relative workflow process and discuss the fundamental rules for workflow tracking. Based on these rules, several matrix operations are presented in Section 4 for tracking structure generation, together with the algorithms for generating tracking structures and performing tracking. Conclusion remarks are given in Section 5.

2 Requirement Analysis with Motivating Example

Basically speaking, current public view approaches all rely on a single workflow model to support inter-organisational business collaboration. This means that once the workflow model for a collaborative business process is defined, it will be open to all participating organisations. If we follow a public view approach, a participating organisation may not be able to offer different visibilities to different organisations. As such, different partnerships between different collaborating organisations cannot be achieved. In our opinion, the visibility between participating organisations is inherently relative rather than absolute. Our relative workflow approach [10] was proposed based on this “relative perspective” philosophy. This approach discards the public view on the inter-organisational workflow process, and allows different organisations to create different views or relative workflow processes upon the same collaborative business process. These multiple relative workflow processes enable participating organisations behave as autonomous entities and enhance the flexibility and privacy control of business collaboration. In the same time, they bring challenges to inter-organisational workflow tracking.

Figure 1 illustrates a business collaboration scenario where a retailer collects orders from customers, and then purchases products from a manufacturer. The manufacturer may contact a shipper for booking product delivery while making goods with supplies from a supplier. In this scenario, a retailer may track the collaborative business process as follows: After placing an order with a manufacturer, the retailer may contact the manufacturer and enquire about the execution status of the production process by referring, say the order number. Furthermore, the retailer may also contact the shipper via the manufacturer and enquire about shipping information after the manufacturer organises product shipping for the retailer by a shipper.
However, the retailer may not be allowed to enquire about the goods supply information, because that could be confidential information of the manufacturer and is hidden from the retailer. For a manufacturer, it may track same collaborative business process differently. Besides the retailer and shipper, the manufacturer can also track the supplier for goods supply information.

From this scenario, we can see that (1) a participating organisation may require tracking other organisations for its involved part of a collaborative business process; (2) each participating organisation may track same collaborative business process differently.

The first point requires collaboration between participating organisations, which is fundamental to inter-organisational workflow tracking. The second point, however, requires that a participating organisation is treated as a fully autonomous entity and can provide different visibilities to different organisations. Obviously, the public view approaches cannot meet the second requirement. Our relative workflow approach can meet both requirements, as we can see from the following sections.

3 Relative Workflows and Tracking Structures

3.1 Relative Workflow Model

In this section, we briefly review the relative workflow model. Figure 2 shows the relative workflow meta model, which has been proposed in [10]. In this model, an organisation, say $g_1$, is considered as an entity holding its own workflow processes called local workflow processes. A local workflow process, $lp^1$, of organisation $g_1$ can be denoted as $g_1.lp^1$.

As the owner, an organisation naturally has an absolute view of its local workflow processes. On the contrary, the host organisation (the owner organisation) may only allow a restricted view of its local workflow processes to its partner organisations due
to the privacy concern. This restriction mechanism may hide some confidential workflow tasks and related links or set some tasks only observable rather than interactable to some partner organisations according to the partnership. The degree of task visibility are defined by visibility constraints, which currently contains three values, viz., “invisible”, “trackable” and “contactable”, as shown in Table 1.

**Table 1. Visibility values**

<table>
<thead>
<tr>
<th>Visibility value</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invisible</td>
<td>A task is said invisible to an organisation, if it is hidden from the organisation.</td>
</tr>
<tr>
<td>Trackable</td>
<td>A task is said trackable to an organisation, if this organisation is allowed to trace the execution status of the task.</td>
</tr>
<tr>
<td>Contactable</td>
<td>A task is said contactable to an organisation, if the task is trackable to the organisation and the task is also allowed to send/receive messages to/from this organisation for the purpose of business interaction.</td>
</tr>
</tbody>
</table>

Visibility constraints are used as a component in defining perceptions. A perception $p_{g_1}^{l_{p}}$ defines how organisation $g_1$ sees $g_2$’s local workflow process $l_{p}$. In the motivating example, the manufacturer may set up the following content in the set of visibility constraints, $\mathcal{V}_C$, of its perception $p_{Retailer}^{Manufacturer:Production}$.


These visibility constraints allow a partial view of the manufacturer’s production process for the retailer. This partial view is called perceivable workflow process. The perceivable workflow process of $g_2$’s local workflow process $g_2.l_{p}$ defined for organisation $g_1$ is denoted as $g_2.l_{p}^{g_1}$.

To represent the diverse partnerships, an organisation may generate a series of perceivable workflow processes of same local workflow process for different partner
organisations. And the inter-organisational business interactions are characterised as directed inter process links, such as $l_{ab1}$ and $l_{bc2}$ in Figure 1. In our relative workflow meta model, these inter process links are defined as message descriptions before being linked, and messaging links after being linked, as shown in Figure 2.

Finally, a relative workflow process can be created by combining the messaging links which connect “contactable” tasks of neighbouring organisations. As shown in Figure 2, a relative workflow process consists of three parts, viz. local workflow processes, perceivable workflow processes and relevant messaging links. Such a relative workflow process represents the collaborative business process perceivable from an organisation.

For example, we suppose that the involved organisations in the motivating example set up the following visibility constraints in proper perceptions, together with perception $p_{\text{Manufacturer.ProductOrdering}}$, which has been given before.

\[
\begin{align*}
\mathcal{V}_\text{Retailer:ProductOrdering}^M & = \{(\text{“raise order"}, \text{Invisible}), (\text{“place order with manufacturer"}, \text{Contactable}), (\text{“invoice customer"}, \text{Contactable}), (\text{“pay invoice"}, \text{Contactable})\}; \\
\mathcal{V}_\text{Shipper:Shipping}^M & = \{(\text{“collect order"}, \text{Contactable}), (\text{“preparation"}, \text{Invisible}), (\text{“delivery"}, \text{Trackable}), (\text{“confirm delivery"}, \text{Contactable})\}; \\
\mathcal{V}_\text{Shipper:Shipping}^R & = \{(\text{“collect order"}, \text{Invisible}), (\text{“preparation"}, \text{Trackable}), (\text{“delivery"}, \text{Trackable}), (\text{“confirm delivery"}, \text{Trackable})\}; \\
\mathcal{V}_\text{Supplier:Supplying}^M & = \{(\text{“collect order"}, \text{Contactable}), (\text{“preparation"}, \text{Invisible}), (\text{“delivery"}, \text{Contactable})\}.
\end{align*}
\]

Since the retailer and the supplier have no partner relationship in the collaborative business process, they do not define perceptions for each other.
According to these visibility constraints, the retailer and the manufacturer may generate corresponding relative workflow processes, as shown in Figure 3 (a) and (b), respectively. The tasks with dashed circles denote the invisible tasks. These two diagrams clearly illustrate that the relative workflow processes for same collaborative business process may be different from different organisations’ perspectives. This reflects the relativity characteristics of our relative workflow approach.

3.2 Representation Matrices

To accurately depict the proposed relative workflow model, we establish several matrices to formally represent key concepts of the relative workflow model.

Self Adjacency Matrix

An n-task workflow process $p$ of organisation $g$ is represented by a special matrix, called Self Adjacency Matrix (SAM), which is defined as,

$$D^g_{SAM} = [d_{ij}], \text{ where } d_{ij} = \begin{cases} r, & \text{if exists link } r \text{ linking task } t_i \text{ and task } t_j, \text{ where } i < j; \\ 0, & \text{otherwise.} \end{cases}$$

Each element of an SAM denotes an intra process link between tasks, such as $r_{a1}$ and $r_{b2}$ in Figure 1. As a link connecting tasks $t_i$ and $t_j$ is put in $d_{ij}$, not $d_{ji}$, where $i < j$, $D^g_{SAM}$ is always an upper triangular matrix. For example, process $a$ in Figure 1 can be represented by $\text{SAM } D^a_{SAM} = \begin{bmatrix} 0 & r_{a1} & 0 & 0 \\ 0 & 0 & r_{a2} & 0 \\ 0 & 0 & 0 & r_{a3} \\ 0 & 0 & 0 & 0 \end{bmatrix}$. Similarly, $D^b_{SAM} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$ and $D^c_{SAM} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$.

A self adjacency matrix can be used to represent not only a local workflow process but also a perceivable workflow process, a relative workflow process, or a tracking structure, which will be introduced later.

Transformation Matrix

When composing a local workflow process $p$ into a perceivable workflow process for organisation $g$, the composition is subject to the visibility constraints defined in proper perceptions. The details of this composition can be found in [10]. In this paper, we formalise the composition process as an $n \times n$ triangular 0-1 matrix, called Transformation Matrix (TM), which is defined as

$$T^g = [t_{ij}], \text{ where } t_{ij} = \begin{cases} 1, & \text{if task } t_j \text{ is composed into task } t_i (j \neq i), \text{ or not composed } (j = i); \\ 0, & \text{otherwise.} \end{cases}$$

This matrix can be directly derived from the visibility constraints defined in the corresponding perception, following the task composition algorithm discussed in [10]. Notice, each column has only one element with value “1”, because each task can be composed only once or may not be composed at all. For example, the procedure of composing local workflow process $b$ into a perceivable workflow process for
organisation $A$ can be described by $TM_{A}^k = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$. This composing procedure is conducted by the visibility constraints defined in perception $p_{A}^{R,b}$.

Likewise, we can calculate that $TM_{A}^k = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$.

**Boundary Adjacency Matrix**

Finally, we also represent the relevant messaging links in a matrix. The messaging links between two workflow processes, $p_1$ and $p_2$, from the perspective of organisation $g$, can be represented by an $m \times n$ matrix called boundary adjacency matrix (BAM), where $m$ is the number of tasks belonging to $p_1$, and $n$ is the number of tasks belonging to $p_2$. A BAM is defined as follows,

$$B_{A}^{p_1/p_2|_{nocb}} = [b_{ij}], \text{ where } b_{ij} = \begin{cases} l, & \text{if exists messaging link } l \text{ connecting } p_{1,t_i} \text{ and } p_{2,t_j} \\ 0, & \text{otherwise.} \end{cases}$$

For example, the interaction relationship between local workflow process $b$ and perceivable workflow process $c$ at the site of organisation $B$, can be represented by $B_{B}^{p_1/p_2} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$. Similarly, $B_{A}^{p_1/p_2} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$.

**3.3 Tracking Structure**

From the discussion in the motivating example section, we see that an organisation’s tracking structure is its observable view upon the execution progress of one collaborative business process. Technically, a tracking structure is different from a relative workflow process, because the latter is created by messaging links connecting to contactable tasks of neighbouring organisations while the former may go beyond neighbouring organisations through trackable tasks.

Unlike the “contactable” visibility value defined in Table 1, the “trackable” value is designed for tracking purpose and can be set on the tasks of the workflow processes belonging to non-neighbouring organisations. We define a tracking structure for each relative workflow process and this tracking structure can be defined by including trackable tasks from its non-neighbouring organisations.

**Tracking Structure:** A tracking structure $ts$ for organisation $g$’s relative workflow process $rp$ consists of the following tasks and links.

- The tasks include: (i) the tasks of relative workflow process $rp$; (ii) the union of task sets of perceivable workflow processes that are reachable from $g$. These perceivable workflow processes may belong to $g$’s neighbouring and non-neighbouring organisations. The reachability of a perceivable workflow process from an organisation is to be discussed in next sub section.
The links include: (i) the links of relative workflow process \( rp \); (ii) the union of link sets of perceivable workflow processes that are reachable from \( g \); (iii) the set of messaging links between perceivable workflow processes that are visible from \( g \).

The visibility of a messaging link from an organisation is to be discussed in next sub section.

### 3.4 Rules

From the definition of a tracking structure, we need to first define the visibility of a messaging link and the reachability of a perceivable workflow process from an organisation. They all depend on the visibility of tasks. For this purpose, we establish the following rules that are used to generate a perceivable workflow process and to determine whether a perceivable workflow process is reachable via visible messaging links and therefore can be included in the tracking structure.

**Intra Process Visibility Rule:** If a task \( t \) in organisation \( g_1 \)’s local workflow process \( g_1.lp \) is set invisible to organisation \( g_2 \), then \( t \) is hidden by composing it into a visible (contactable or trackable) task of \( g_1.lp \). The links connecting \( t \) will be changed accordingly. The composition procedure will be discussed in the composition operation in next section. After composition, \( g_1.lp \) becomes a perceivable workflow process \( g_1.lpg \).

**Inter Process Visibility Rule:** A messaging link \( l \) connecting two perceivable workflow processes is said visible to organisation \( g \), if and only if both tasks connected by \( l \) are visible to \( g \).

**Expansion Rule:** Let \( ts \) be the tracking structure for a relative workflow process of organisation \( g \). A perceivable workflow process outside \( ts \) is said reachable and therefore can be included into \( ts \), if and only if it has at least one visible messaging link connecting a task inside \( ts \).

Following the Intra Process Visibility Rule, the original link \( r_{b_1} \) connecting tasks \( b_1 \) and \( b_2 \) of process \( b \) in Figure 1 becomes invisible in its perceivable form for organisation \( A \) in Figure 3 (c) because \( b_2 \) is invisible to organisation \( A \). Correspondingly, links \( r_{c_2} \) and \( r_{c_3} \), which connect \( b_2 \) and \( b_3 \), \( b_2 \) and \( b_4 \) in Figure 1 respectively, are now changed to connect \( b_1 \) and \( b_3 \), \( b_1 \) and \( b_4 \), in Figure 3 (c).

Following the Inter Process Visibility Rule, messaging link \( l_{bc} \) connecting task \( b_4 \) and task \( c_1 \) is not visible while messaging link \( l_{bc} \) connecting task \( b_3 \) and task \( c_2 \) is visible in Figure 1. Following the Expansion Rule, the perceivable workflow processes of process \( c \) is reachable because of the existence of the visible messaging link \( l_{bc} \). By applying all these rules, we can finally generate a tracking structure shown in Figure 3 (c) for \( A \)’s relative workflow process shown in Figure 3 (a).

### 4 Generating Tracking Structures

#### 4.1 Operations

According to the rules discussed in last section, we define three matrix operations for tracking structure derivation.
Operation 1. Composition Operation

As defined in the TM for a local workflow process, each element with value “1” in a non-diagonal position \((i, j)\) stands for a procedure of composing the composed task \(t_j\) to the composing task \(t_i\). Under the restriction of the Intra Process Visibility Rule, the following sub rules may apply to this composition:

1. A link connecting \(t_j\) and \(t_k\) \((k \neq i)\) is changed to a link connecting \(t_i\) and \(t_k\);
2. A link connecting \(t_i\) and \(t_k\) \((k \neq j)\) is unchanged;
3. A link connecting \(t_i\) and \(t_j\) is discarded.

The first sub rule requires an operation that can be applied to the SAM defined for the local workflow process. This operation first adds the elements in row \(j\) to their corresponding elements in row \(i\), and then sets all elements in row \(j\) to zero. This can be achieved by applying a matrix multiplication to this TM and the SAM defined for the local workflow process. A function \(f_{\text{reshape}}\) is assigned to reshape the result matrix into an upper-triangular form.

For input matrix \(M_{n \times n}\), function \(f_{\text{reshape}}\) is defined as

\[
f_{\text{reshape}}(M_{n \times n}) = \left[ m_{ij} + m_{ji}, \right]_{i < j} \quad 0, \quad \text{otherwise.}
\]

The second sub rule identifies the case that needs no action. From the definition of a TM, we can see that the composing tasks of this case all have value “1” on the diagonal line, which takes no effect in the matrix multiplication.

Regarding the third sub rule, we need to check whether there exists a link connecting \(t_i\) and \(t_j\) in the corresponding TM. This can be easily achieved by checking whether there exists a row that has value “1” at both column \(i\) and column \(j\). We can represent the existence of such a link by a boolean expression, i.e. \(f_{\text{row}}(i) = f_{\text{row}}(j)\), where \(f_{\text{row}}(x)\) defines a function that returns the row where column \(x\) has the value “1”.

Finally, these three sub rules can be merged together as an operation \(\otimes\), which is defined on \(T_{n \times n} \otimes D_{n \times n} = [f_{\text{row}}(i) \neq f_{\text{row}}(j)] \sum_{k=1}^{n} a_{ij} d_{kj}\). Hence, organisation \(g_i\) may apply a Composition Operation on a local workflow process \(p\) to generate a perceivable workflow process for \(g_2\). This can be defined as

\[
D^p_{g_2} = f_{\text{reshape}}(T^p_{g_1} \otimes D^p_{g_1})
\]

Here \(D^p_{g_1}\) and \(T^p_{g_1}\) are the SAMs of \(g_1\’s\ local workflow process \(p\) and the corresponding TM for perception \(p^c\), respectively.

By applying this composition operation, organisations \(B\) and \(C\) can generate perceivable workflow processes \(b\) and \(c\) for organisation \(A\) in the form of

\[
D^b_A = f_{\text{reshape}}(T^b_A \otimes D^b_B) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\] and

\[
D^c_A = f_{\text{reshape}}(T^c_A \otimes D^c_C) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\]


Operation 2. Connection Operation
According to the Inter Process Visibility Rule, we need to identify the visible messaging links between perceivable workflow processes in order to include perceivable workflow processes of non-neighbouring organisations in the tracking structure for an organisation. For this purpose, we need to identify the visible tasks by simply checking elements valued “1” in diagonal positions of the corresponding TM. We use function \( f_{\text{diag}} \) to diagonalise TM \( T \) into a diagonal matrix \( T^\circ \). Function \( f_{\text{diag}} \) is defined as follows,
\[
f_{\text{diag}}(T[n \times n]) = T^\circ[n \times n],
\]
where \( o_{ij} = \begin{cases} 1, & \text{if } t_{ij} = 1 \text{ and } i = j; \\ 0, & \text{otherwise.} \end{cases} \)

The visible messaging link between two workflow processes, for example, \( g_1 \)'s \( p_1 \) and \( g_2 \)'s \( p_2 \), from the perspective of another organisation, say \( g_3 \), can be represented as \( \text{BAM}^{2211}_{3} \).

The Connection Operation connecting \( g_1.p_1 \) and \( g_2.p_2 \) for \( g_3 \) can be defined as
\[
B^{g_1.p_1,g_2.p_2}_{g_3} = (f_{\text{diag}}(T^{g_1.p_1}_{g_1})) \cdot (f_{\text{diag}}(T^{g_2.p_2}_{g_2})) \cdot B^{g_1.p_1,g_2.p_2}_{g_3} \cdot T^T
\]

This connection operation first requires \( g_1 \) to diagonalise TM \( T^{g_1.p_1}_{g_1} \), and then perform a matrix multiplication on the diagonalised \( T^{g_1.p_1}_{g_1} \) and BAM \( B^{g_1.p_1,g_2.p_2}_{g_3} \). \( g_2 \) will subsequently use the diagonalised matrix \( T^{g_2.p_2}_{g_2} \) to multiply the result matrix from \( g_1 \).

In the connection operation, proper transposition operations are needed to align the columns of the left hand matrix with the rows of the right hand matrix for matrix multiplication.

Regarding the motivation example given in Section 2, organisations \( B \) and \( C \) can generate matrix \( B^{B_1,B_2}_{A} \) for organisation \( A \) to provide the visible messaging links between \( B \)'s process \( b \) and \( C \)'s process \( c \) in \( A \)'s view.

\[
B^{B_1,B_2}_{A} = (f_{\text{diag}}(T^{B_1}_{B_1})) \cdot (f_{\text{diag}}(T^{B_2}_{B_2})) \cdot B^{B_1,B_2}_{A} \cdot T^T
\]

Operation 3. Extension Operation
The Expansion Rule is used for extending the tracking structure to include perceivable workflow processes of both neighbouring and non-neighbouring organisations. Technically, an extension step can be represented as an Extension Operation. With a local workflow process \( p_1 \) in the tracking structure, organisation \( g_1 \) may apply the extension operation to include a local workflow process \( p_2 \) of organisation \( g_2 \) in the tracking structure. This can be defined as
\[
D^{g_1.p_1,g_2.p_2}_{g_2} = \begin{bmatrix}
D^{g_1.p_1}_{g_2} & B^{g_1.p_1,g_2.p_2}_{g_2} \\
0 & D^{g_2.p_2}_{g_2}
\end{bmatrix}
\]
For example, the tracking structure containing process \( a \) and \( b \) from the view of organisation \( A \), can be described by a composite SAM \( D_{ab}^\alpha = \begin{pmatrix} D_{ab}^\alpha \ & B_{ab}^\alpha \\ 0 \ & D_a^\alpha \end{pmatrix} \), which is obtainable through this extension operation.

### 4.2 Generation Algorithm

The tracking structure generation can be technically considered as a process of appending a new generated column each time a reachable workflow process is detected. This new generated column consists of a new SAM and a series of BAMs. The new SAM describes the inner structure of this detected workflow process, while the BAMs describe the interaction relationships between the detected workflow process and the processes already included in the structure.

As shown in Figure 4, at the starting point, the tracking structure contains only \( D_{g_1}^{\beta, \eta} \), which means that only \( g_1.p_1 \) is included. Afterwards, \( g_1 \) detects that perceivable workflow process \( g_2.p_2 \) is reachable from \( g_1.p_1 \), and then appends a column containing \( B_{g_1.p_1/g_2.p_2} \) and \( D_{g_2.p_2} \) to the tracking structure. Likewise, organisation \( g_2 \) may append a column containing \( B_{g_2.p_2/g_1.p_1} \), \( B_{g_2.p_2/g_3.p_3} \), and \( D_{g_2.p_2} \), when \( g_2 \) detects that process \( g_3.p_3 \) is reachable from \( g_1.p_1 \) via \( g_2.p_2 \). This appending process continues until all reachable perceivable workflow processes are detected.

Because the inter-process interaction relationships can only be identified by the organisation (context organisation) that owns the “bridging” workflow processes, by which the expansion proceeds, a propagation mechanism is adopted to spread this detection process over all involved organisations. The context organisation for an appending step may change from time to time. Organisation \( g_1 \) is called the original context organisation of this tracking structure.

We note that the process shown in Figure 4 starts from \( g_1 \)’s local workflow process \( g_1.p_1 \) instead of \( g_1 \)’s relative workflow process \( g_1.rp \). Actually, \( g_1.rp \) can be generated by the first step of the process when \( g_1 \) is the context organisation.

Algorithm 1 details the generation procedure. In algorithm 1, function \( relatedProc(p) \) returns a set of local workflow processes and perceivable workflow processes that have
direct interactions with process $p$. Function $\text{includedProc}(\text{trackStruc})$ returns all included workflow processes at that moment in tracking structure $\text{trackStruc}$, which initially contains an SAM defined on a local workflow process of the original context organisation. Function $\text{BAM}(p_1, p_2, g)$ returns the BAM between processes $p_1$ and $p_2$ from the view of organisation $g$, using the connection operation. Function $\text{SAM}(p, g)$ returns the SAM of process $p$ from the view of organisation $g$, using the composition operation. Function $\text{genOrg}(p)$ returns the organisation of process $p$.

Algorithm 1. genTrackStruc - Tracking Structure Generation

Input:
\begin{itemize}
    \item $\text{trackStruc}$ - A tracking structure matrix
    \item $\text{cxtProc}$ - A local workflow process of the context organisation
    \item $\text{origCxtOrg}$ - The original context organisation that starts the generation
\end{itemize}

Output:
\begin{itemize}
    \item $\text{trackStruc}$ - The expanded tracking structure matrix
\end{itemize}

Step 1 \hspace{1em} Detect workflow processes
$\text{detectedProcSet} = \text{relatedProc}(\text{cxtProc})$;
$\text{includedProcSet} = \text{includedProc}(\text{trackStruc})$;
$\text{detectedProcSet} = \text{detectedProcSet} - \text{includedProcSet}$;

Step 2 \hspace{1em} Expand the tracking structure
$\text{appendedProcSet} = \emptyset$; for each process $p_i \in \text{detectedProcSet}$
    \begin{itemize}
        \item $tempB = \text{BAM}(\text{cxtProc}, p_i, \text{origCxtOrg})$;
        \item if $tempB$ is a non-zero matrix then
            \begin{itemize}
                \item $\text{newColumn} = \emptyset$;
                \item for each process $p_j \in \text{includedProcSet}$
                    \begin{itemize}
                        \item $B = \text{BAM}(p_j, p_i, \text{origCxtOrg})$;
                        \item Append $B$ to $\text{newColumn}$.
                    \end{itemize}
            \end{itemize}
        \item /* generate related boundary adjacency matrices of the new column*/
        \item end if
    \end{itemize}
    \item $D = \text{SAM}(p_i, \text{origCxtOrg})$;
    \item /* generate the self adjacency matrix of the new column */
    \begin{itemize}
        \item Append $\text{newColumn}$ and $D$ to $\text{trackStruc}$, using extension operation.
        \item $\text{includedProcSet} = \text{includedProcSet} \cup \{ p_i \}$;
        \item $\text{appendedProcSet} = \text{appendedProcSet} \cup \{ p_i \}$;
    \end{itemize}
end for

Step 3 \hspace{1em} Propagate the detection process
for each process $p_i \in \text{appendedProcSet}$
    \begin{itemize}
        \item $\text{targetOrg} = \text{genOrg}(p_i)$;
        \item /* Ask targetOrg to call genTrackStruc */
        \item $\text{trackStruc} = \text{targetOrg} \text{. genTrackStruc}(\text{trackStruc}, p_i, \text{origCxtOrg})$;
    \end{itemize}
end for

Step 4 \hspace{1em} Return the expanded tracking structure
\begin{itemize}
    \item return $\text{trackStruc}$
\end{itemize}

The tracking structure generation process starts from a local workflow process of the original context organisation, and then spreads to all reachable workflow processes of the involved organisations. When this generation process comes to an organisation, this organisation becomes the context organisation of the above algorithm.
For example, if we start from the retailer’s product ordering process, i.e., process $a$ in the motivating example, this algorithm first detects the workflow processes having direct interactions with process $a$. Then it checks for each detected workflow process whether it is reachable from organisation $A$, and if so, the detected process will be included to the tracking structure. In this step, organisation $B$’s process $b$ is included, and the tracking structure is expanded to $D_A^{D_B} = \begin{pmatrix} D_A & B_{A}^{B}\n\n 0 & D_A \end{pmatrix}$. After that, this generation process will be propagated to $B$, and $B$ repeats the above steps to extend the tracking structure. At this stage, $B$ may find process $c$ and process $d$, while only process $c$ is included. This is because that the retailer and the supplier do not set up perceptions for each other in this example, and hence no transformation matrix is defined for process $d$ from $A$. Therefore, the tracking structure is finally expanded to $D_A^{D_B} = \begin{pmatrix} D_A & B_{A}^{B}\n\n 0 & D_A \end{pmatrix}$, which equals to the diagram shown in Figure 3 (c).

Here, $B_{A}^{B}$ is a zero matrix because there is no direct interactions between processes $a$ and $c$, and the other sub matrices can be found from the former part of this paper.

### 4.3 Performing Workflow Tracking

In an inter-organisational workflow environment, there is another issue, i.e., how to keep the correct correspondence between collaborating local workflow instances. From the semantics of a collaborative business process, we can find the cardinality relationship between collaborating local processes, e.g., more than one instance of process $a$ may associate with a single instance of process $b$ for the purpose of batch production. While this kind of cardinality relationship can be determined at build time, the correlation between the particular instances of these processes has to be determined at run time, when they “shake hands”.

To perform workflow tracking, we design a data structure, as shown in Figure 5, to keep the necessary information for tracking. This data structure consists of a series of lists, each of which represents the set of instances belonging to a specific local workflow process. Each element of a list has several units to record the workflow execution status. The links connecting elements represent the correspondence between instances of different workflow processes.

The tracking process is similar to a graph traversal process, where the nodes represent the related workflow instances and the arcs represent their messaging links.
Algorithm 2. *trackProc* - Tracking Process

<table>
<thead>
<tr>
<th>Input:</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>trackStruct</td>
<td>The tracking structure to conduct the tracking</td>
</tr>
<tr>
<td>origInstance</td>
<td>An instance of the original context organisation’s initial local workflow process defined in <em>trackStruct</em></td>
</tr>
<tr>
<td>DS</td>
<td>The tracking data structure</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output:</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS</td>
<td>The updated tracking data structure</td>
</tr>
</tbody>
</table>

Step 1  **Initialisation**

- `trackInstanceSet = ∅;`
- `stack s = new stack();`
- `s.push( origInstance );`

Step 2  **Discover the participating workflow instances**

- While `s` is not empty do
  - `cxtInstance = s.pop();`
  - `foundInstanceSet = linkedInstances( *cxtInstance*, trackStruct ) – trackInstanceSet;`
  - For each `i` ∈ `foundInstanceSet`
    - `s.push( i );`
    - `cxtProc = genProc( *cxtInstance* );`
    - `BAMset = relatedBAMs( *cxtProc*, trackStruct );`
    - For each link `l` of each boundary adjacency matrix `B` ∈ `BAMset`
      - /* now, start discovering workflow instances by following each visible messaging link */
        - `partnProc = partnerProc( B, cxtProc );`
        - `partnOrg = genOrg( partnProc );`
        - If `cxtInstance` is newly fired then
          - `newInstanceSet = ∅;`
          - Ask `partnOrg` to check any new participating instances of *partnProc*, and set the instances to *newInstanceSet*.
          - `newInstanceSet = newInstanceSet – trackInstanceSet;`
          - /* filter the previous discovered instances */
            - For each `i` ∈ `newInstanceSet`
              - `addInstance( partnProc, i );`
              - `addLink( cxtInstance, i );`
          - /* update the tracking data structure */
            - `s.push( i );`
          - /* and add the newly discovered instance to the stack */
        - End if
    - End for
  - End for
  - `trackInstanceSet = trackInstanceSet ∪ { *cxtInstance* };`
    - /* the set of instances to track */
- End while

Step 3  **Update the execution status of participating workflow instances**

- For each instance `i` ∈ `trackInstanceSet`
  - `p = genProc( i );`
  - `targetOrg = genOrg( p );`
  - Enquire `targetOrg` for the execution status of `i`, and then update the status of `i` in *DS*.

end for
to be tracked. In addition, new participating workflow instances will be identified at the time when visible messaging links are fired.

Details can be found in Algorithm 2. In this algorithm, function addInstance(\(p, i\)) inserts instance \(i\) to the list of process \(p\) in the tracking data structure. Function addLink(\(i_1, i_2\)) creates a link between instances \(i_1\) and \(i_2\) in the tracking data structure. Function linkedInstances(\(i, trackStruc\)) returns the instances linked to instance \(i\) in the tracking data structure, according to the tracking structure \(trackStruc\). Function relatedBAMs(\(p, trackStruc\)) returns the set of BAMs related to process \(p\), defined in \(trackStruc\). Function partnerProc(\(B, p\)) returns the partner process of \(p\) defined in BAM \(B\). Function genOrg(\(p\)) returns the organisation of process \(p\). Function genProc(\(i\)) returns the process of instance \(i\).

This algorithm starts from a local workflow instance of the original context organisation. Following the corresponding tracking structure, this algorithm searches along visible messaging links and propagates the execution status queries to all reachable workflow instances with the cooperation of participating organisations. The corresponding tracking structure records the interaction relationship between the processes of these reachable workflow instances. When an inter-organisational interaction is fired, the algorithm will check whether any new workflow instances join the business collaboration. If so, the algorithm will add these workflow instances to the tracking data structure.

5 Conclusion

This paper contributed to the study of workflow tracking across organisational boundaries. Compared with other workflow tracking solutions, the approach proposed in the paper not only enables an organisation to track other organisations for its involved parts of collaborative business processes, but allows different organisations track same collaborative business process differently as well.

In the paper, we deployed a matrix based framework which includes three representation matrices and three matrix operations. Algorithms using these matrices and operations for generating tracking structures and performing workflow tracking are developed. The framework allows an organisation to generate its own tracking structure based on its visibility to other organisations, thus privacy can be protected. The framework also allows a tracking structure to be generated on the fly, thus enables flexibility in workflow tracking. Based on its own tracking structure, an organisation can proactively trace the execution progress of its involved part of a collaborative business process.

Acknowledgements

The work reported in this paper is partly supported by the Australian Research Council discovery project DP0557572.
References