

## CorPN: managing instance correspondence in collaborative business processes

Xiaohui Zhao · Chengfei Liu · Yun Yang · Wasim Sadiq

Published online: 18 March 2011  
© Springer Science+Business Media, LLC 2011

**Abstract** Driven by the booming global business, organisations are required to align their business processes into an inter-connected network. The sophisticated nature of collaboration results in dynamic and complex interactions and correlations between participating business processes. This inevitably poses challenges to business process management in terms of recognising the instance correspondence and analysing the interaction behaviours of collaborative business processes. Nevertheless, this issue has received very limited attention from inter-organisational workflow research. In this paper, a novel correspondence Petri net model called CorPN is developed to specify instance correspondence with extensions to classical WF-Nets. In addition, a method is established to analyse the behavioural properties of CorPN nets for the purpose of process verification and examination.

**Keywords** Collaborative business process · Business process management · Process instance correspondence · Correspondence Petri net

---

Communicated by Athman Bouguettaya.

X. Zhao (✉)

Department of Computing, Faculty of Creative Industries and Business, Unitec Institute of Technology, Auckland, New Zealand  
e-mail: [xzhao@unitec.ac.nz](mailto:xzhao@unitec.ac.nz)

C. Liu · Y. Yang

Faculty of Information and Communication Technology, Swinburne University of Technology, Melbourne, Australia

C. Liu

e-mail: [cliu@swin.edu.au](mailto:cliu@swin.edu.au)

Y. Yang

e-mail: [yyang@swin.edu.au](mailto:yyang@swin.edu.au)

W. Sadiq

SAP Research, Brisbane, Australia  
e-mail: [wasim.sadiq@sap.com](mailto:wasim.sadiq@sap.com)

### 1 Introduction

Business globalisation drives organisations to create and manage collaborative business processes to quickly respond customers’ demands and grasp market opportunities [1–4]. From 1990s, organisations have been undergoing a thorough transformation towards highly flexible and agile collaborations with IT supports [5–9]. In such collaborations, the underlying collaborative business processes span over different business processes of participating organisations [10–12]. Such complex scenarios inevitably complicate the collaboration behaviours and relations, and pose challenges to process orchestration and choreography management.

Traditional process management approaches mainly focus on process interaction at build time, and facilitate it by composing related processes and linking the communication channels. Nevertheless, the instance-level collaboration among business processes always results in complex instance correspondence and correlations. Here, we use an example to illustrate this scenario. In the collaboration shown in Fig. 1, a retailer initiates a product-ordering process instance that orders products from a manufacturer. The manufacturer then uses a production process instance to receive orders from retailers. Once the manufacturer collects enough orders, the production instance may start making goods in bulk owing to the economies of scale.

During the production, the manufacturer may contact multiple shippers to deliver products to retailers, and these shippers may arrange different delivery routes and schedules according to their transport capability, delivery optimisation, etc. Finally, the shipping instances of different shippers deliver the goods to proper retailers according to these correlations. Figure 2 represents a possible instance correspondence situation, where two product-ordering process instances, one production process instance and three shipping process instances are involved. The dashed arrows denote the transferred messages between these instances.

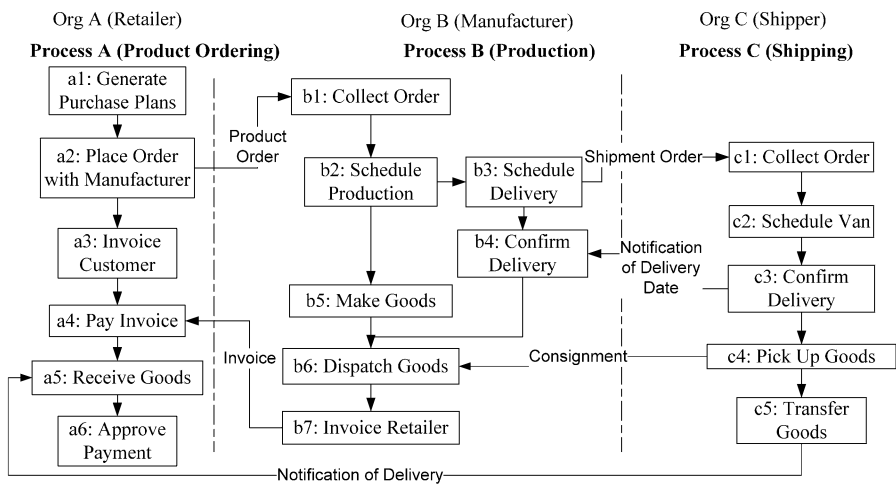
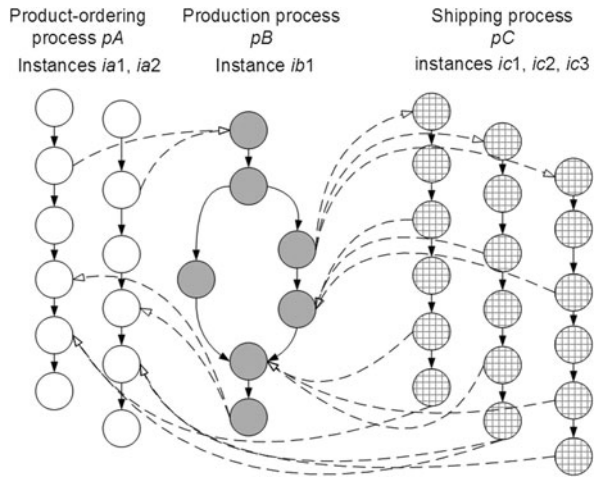


Fig. 1 A collaborative business process sample

**Fig. 2** Workflow cardinality of motivating example



This collaboration scenario clearly illustrates that a business process instance is likely to interact with multiple instances of other business processes. When collecting orders, a production instance may correspond to multiple product-ordering instances; and when delivering products, each shipping instance may handle the product deliveries ordered by multiple product-ordering instances. Such instance correspondence forms dynamically at run time, and evolves along with the collaboration. For example, when receiving orders from retailers, the manufacturer’s production instance is correlated with retailers’ product-ordering instances. Afterwards, the manufacturer contacts shippers to book deliveries and passes the order numbers to proper shippers. With these order numbers, shippers’ shipping instances are indirectly correlated with retailers’ product-ordering instances. Following these correlations, shippers have the knowledge of which products to pick up from the manufacturer and where to deliver them. In this way, instance correlations combine business interactions into a meaningful collaboration.

Though such instance correlations are important for process collaboration, traditional static modelling approaches can hardly capture these dynamics, as they simply assume a one-to-one relation between collaborating process instances. To tackle this issue, we propose a novel CorPN net model to characterise instance correspondences in terms of workflow cardinality and instance correlation. We also analyse the behavioural properties of CorPN nets to examine and verify the process interaction design.

The rest of this paper is organised as follows: Sect. 2 reviews the works related to instance correspondence. Section 3 presents our CorPN net model and its extensions to classical Petri nets. Section 4 introduces a methodology to analyse the behavioural properties of CorPN nets for the purpose of process verification and examination. To highlight our model’s advantages, Sect. 5 compares our CorPN net model with other approaches. Finally, the paper is concluded with an indication of our future work.

## 2 Related works

The issues of multiple process instances has been touched in WF-Net model [13] and workflow patterns, and some service orchestration languages, like WS-BPEL, etc.

Based on the classical Petri net, WF-Net model uses tokens to delegate the multiple process instances, and also provides a foundation for workflow validity and consistency checking. According to a given message sequence chart, several WF-Nets can be connected together to represent a collaborative business process. Yet, WF-Net does not specify instance correlations, nor analyses the interaction behavioural properties of a collaborative business process.

In regard to workflow patterns, van der Aalst, ter Hofstede, Kiepuszewski, and Barros deployed coloured Petri nets to represent six categories of workflow patterns [14]. Especially for “Patterns involving Multiple Instances”, particular mechanisms were proposed to track instance identities and synchronise them with high level Petri net model. These workflow patterns were supported by YAWL (Yet Another Workflow Language) and related workflow management system [15]. Instead of multiple instances of a workflow process, these patterns handle the multiple instances of a sub process belonging to a workflow process.

Multiple workflow instantiation was discussed by Dumas and ter Hofstede in their work [16]. The concept of multiple instances mainly denotes the multiple execution of one workflow activity, and therefore the proposed synchronisation techniques are for parallel activity instances, such as N-out-of-M join. Later, they extended their work into service interaction patterns [17], to cater for the emerging Web service technologies.

Guabtini and Charoy classified the multiple-workflow instantiation into parallel and iterative instances [18]. Two special sets, for parallel instances and iterative instances, respectively, were designed to collect the activities of multiple executions. These sets can be nested or overlapped to handle complicated scenarios.

Mulyar, Aldred, and van der Aalst investigated the message multicasting patterns in service interactions [19]. Their research took into account the factors of message queuing, sorting and indexing, possibilities of non-responding parties and missing replies, etc.

Nevertheless, most of above research focus on interaction patterns, and sidestep the instance correspondence issue in collaborative business processes. As an initiate attempt to support instance correspondences, work on Proclets [20, 21] touched the issue of workflow cardinality. Following an object-oriented perspective, a Proclet class can define the cardinality for message multi casting at type level. Yet, Proclet assumes that the instance correlations are pre-defined before execution, and therefore cannot support dynamic instance correlations. Russel, Dew, and Djemame have created a service-based secure collaborative workflow system for the distributed aircraft maintenance environment (DAME) [22]. In their work, a dynamic workflow-team policy is created for each workflow instance, and used to resolve role mappings and define access to services instances (task instances of the workflow instance) for fine-grained access control. Basically, this approach only solves the intra-workflow correspondence between task instances and users, from a security perspective.

In BizAgi BPM suit [23], a database table relationship diagram was used to represent the cardinality between attributers of different entities. This diagram largely

focuses on the entities involved in processes, like clients and applied loans, a client's city and country information, etc., rather than processes themselves. Consequently, this diagram only indirectly represents the cardinality between business processes, and it is short of run-time correlation support. Based on the limited published documents, other commercial BPM softwares, such as IBM WebSphere [24], Tibco [25], etc., seem to adopt a pragmatic method, i.e., recording run-time instance correspondences in a backend database table. Yet, this simple recording mechanism lacks analysis capability, and therefore, few of these BPM softwares support build-time correspondence simulation or verification.

WS-BPEL (previously BPEL4WS) [26] used a set of explicit identifiers, a.k.a., correlation sets, to combine workflow instances. The identifiers with the same value act as the semantic clue for instance correlations. Some commercial BPEL engines, such as SAP NetWeaver Process Integration [27] and IBM WebSphere Process Server [28], realise this correlation by attaching extra identifiers, like message ID, message source and target, etc. Nevertheless, WS-BPEL defines a business process from the perspective of a pivot organisation, and therefore only represents the interaction behaviours between the pivot organisation and its neighbouring organisations. This feature limits its application for complex business collaborations, where exist interactions beyond neighbouring organisations.

Following WS-BPEL's correlation handling, a recent work by van der Aalst, Mooij, Stahl and Wolf discussed correlation patterns in service interactions with Petri net representations [29]. Particularly, this work introduced two dimensions for handling advanced correlations, viz., multilateral interaction and multiple instances. Correspondingly, we aim to tackle the instance correspondence issue in terms of these two dimensions. In work AO4BPEL, Charfi and Mezini adopted aspect-oriented programming paradigm into Web service composition by extending BPEL [30]. Though their work mainly focuses on crosscutting dynamics changes, the proposed dynamic weaving mechanism covers the mapping and delivery of requested aspects to proper BPEL instances. This mechanism however relies on pre-defined join points and does not address the correspondence between BPEL instances.

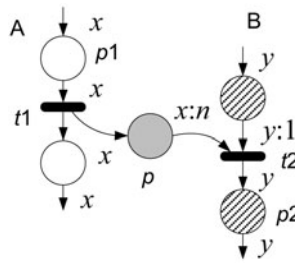
### 3 Correspondence representation methodology

#### 3.1 Overview of the methodology

This paper proposes a Petri net based model to represent the behaviours of a collaborative business process. The reported work is based on our previous research on relative workflows [31, 32] and instance correspondences [33] with significant extensions and improvements on model formalism and behavioural property analysis.

Technically, our CorPN net model follows the classical WF-Net formalism, which was proposed by van der Aalst and van Hee [13] for business process modelling. The detailed definition of WF-Net is given in the [Appendix](#). Nevertheless, WF-Net is originally designed to model single workflow processes, and it does not take into account workflow cardinality or instance correlations. According to the discussion on collaborative business processes in Sect. 1, the instance level representation requires the

**Fig. 3** Messaging representation



distinction of each single instance and the instances of constituent processes, communication behaviours, workflow cardinality, etc. In the WF-Net context, each WF-Net delegates a constituent business process, and its contained tokens represent the instances of the process. In the CorPN model, we combine these WF-Nets together using communication places to represent a collaborative business process. The communication behaviours are described as messaging actions between WF-Nets, which are always multi-casting actions due to workflow cardinality. Some special transitions are also needed to represent the process initiation triggered by messages. In addition, cardinality parameters and correlation structures are deployed to explicitly describe workflow cardinality and instance correlations. The next section is to introduce these extensions and the CorPN net model in details.

### 3.2 CorPN net model

To model instance correspondence in collaborative business processes, the following extensions are considered in CorPN net model based on WF-Nets.

- Token group and permission expressions

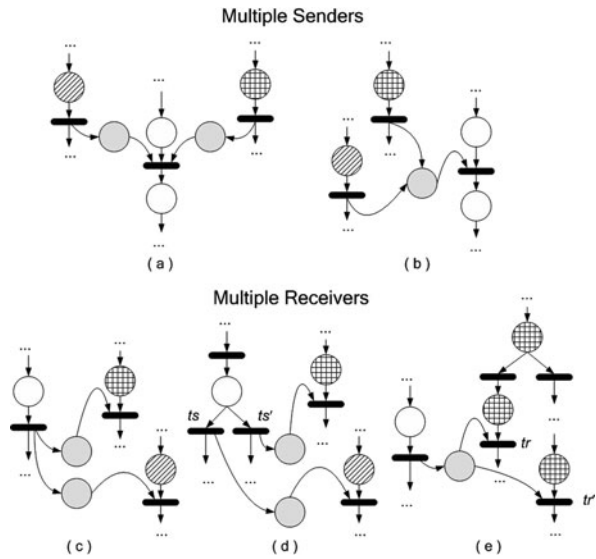
In WF-Net context, tokens delegate the instances of workflow processes, and transitions delegate the tasks of business processes. In our CorPN net model, we classify the tokens into different token groups according to their delegated workflow processes. Each token group represents the instances of an involved workflow process.

Token groups are also used to compose *permission expressions* for the purpose of restricting token flows. Take Fig. 3 as an example, letters ‘x’ and ‘y’ delegate two token groups, i.e., the token sets of WF-Nets A and B, respectively. When ‘x’ or ‘y’ is marked along the arcs between places and transitions, they become permission expressions, which restrict that only the tokens of that group can pass through corresponding arcs. For example, a token in WF-Net A can flow from place  $p_1$  to transition  $t_2$  via place  $p$ , yet as an individual token, it cannot flow into place  $p_2$  of WF-Net B, as indicated by the token group symbols. Symbols ‘:n’ and ‘:1’ are to be introduced later in this section.

- Messaging representation

To model the process collaboration, we have developed communication places to represent the synchronous messaging interactions between WF-Nets. Technically, a communication place links from or links to a WF-Net transition via an And-Split/Join structure, and such a WF-Net transition can serve to handle message sending/receiving. The rationale of this design is because that the message sending action

**Fig. 4** Interactions with multiple senders/receivers



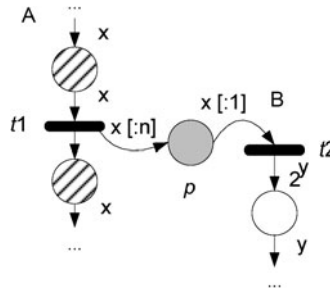
is considered as one consequence from the completion of a task, and another consequence is the change of the execution status. These two consequences always happen simultaneously, and therefore they are represented with an AND-Split structure. In regard to the synchronous message receiving action, it is subject to both the message arrival and the execution status of the message-sending process instance, i.e., the receiving action only occurs when the receiving task is in ready state and a message comes. Therefore, AND-Join structure fits into this scenario.

Figure 3 illustrates how messaging interactions are represented in CorPN net model. Communication place  $p$  connects WF-Net  $A$ 's transition  $t1$  and WF-Net  $B$ 's transition  $t2$ . A token flowing from  $t1$  to  $t2$  via  $p$  indicates an interaction that is requested by process  $A$  and responded by process  $B$ . Here,  $t1$  and  $t2$  are called *interaction requesting transition* and *interaction responding transition*, respectively.

- Message multi-casting

To represent the messaging interactions between more than two processes, we have developed the following structures in our CorPN net model. For the case of multiple senders, Fig. 4 (a) shows an interaction of receiving messages from multiple senders, and Fig. 4 (b) shows an interaction of receiving one message from multiple senders. For the case of multiple receivers, Fig. 4 (c) shows an interaction of sending one message to multiple receivers, where one message from the left WF-Net is first split into two messages, and then these two messages flow into the two WF-Nets on the right side, respectively. For the case of sending the same message to one of multiple receivers, there are two representation scenarios, as shown in Fig. 4 (d) and (e), respectively. An Or-Split structure can be set to distribute the sending transition to each branch as shown in Fig. 4 (d); or an Or-Split structure can be set to distribute a receiving transition to each alternative branch of the receiver's WF-Net as shown in Fig. 4 (e).

**Fig. 5** Token producible transitions



- Cardinality parameters

The message multicasting representation only handles the interactions at process level. As discussed in Sect. 1, messaging interactions may involve multiple instances of the same process. To specify such workflow cardinality, we have designed two special parameters, i.e., ‘1’ (one) and ‘n’ (many), to represent the quantitative relation between participating tokens. Particularly, these cardinality parameters are used to compose permission expressions for interaction-responding transitions. Take transition  $t2$  in Fig. 3 for example, one  $y$ -type token and multiple  $x$ -type tokens are involved in this interaction, and the output of this interaction is a single  $y$ -type token.

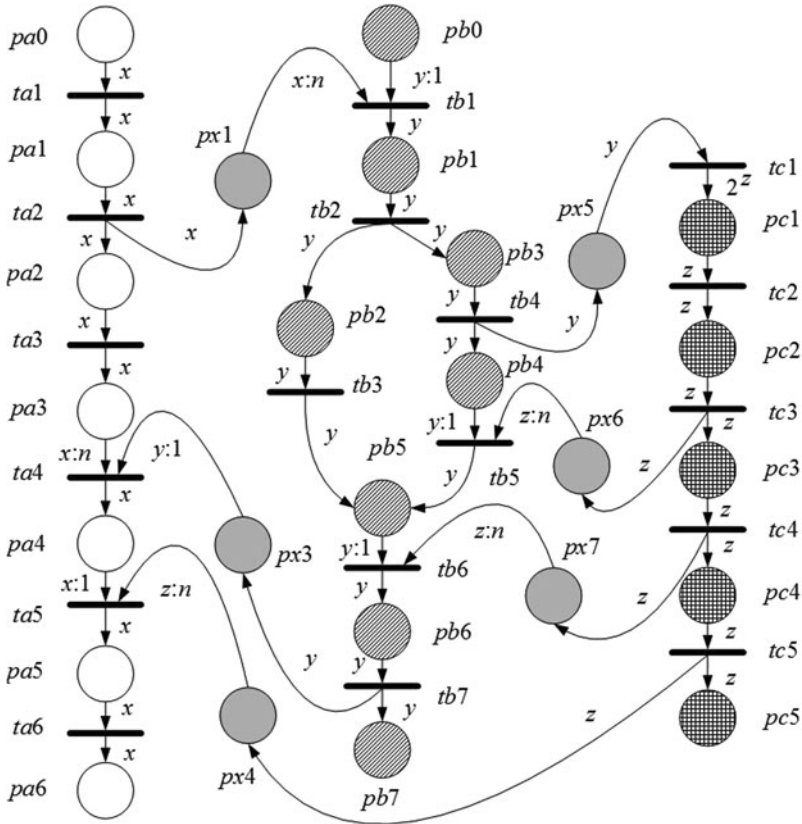
- Token producible transitions

In the collaboration example discussed in Sect. 1, the receipt of a product order from retailers may trigger the production instances of the manufacturer. In CorPN net context, this indicates that the token initialisation is subject to the receipt of foreign token(s) via a communication place. To model this behaviour, we have developed *token producible transitions* in our CorPN model. As shown in Fig. 5, transition  $t2$  is token producible, and therefore WF-Net  $B$  may generate several  $y$ -type tokens when an  $x$ -type token flows to  $t2$  via place  $p$ . Permission expression  $2^y$  indicates that any production of  $y$ -type tokens may go through that arc.

Based on these extensions, we give the definition of CorPN net, as follows.

**Definition 1** (Correspondence Petri net) For collaborative business process  $cbp$ , a CorPN net  $cpn$  is defined as tuple  $(WFN, P, T, F, P^\circ, F^\circ, V, Pem)$  to describe  $cbp$ , where

- set  $WFN$  contains the constituent WF-Nets, which represent all the participated processes in  $cbp$ ;
- sets  $P, T$  and  $F$  contain the places, transitions and arcs of WF-Nets in  $WFN$ , respectively, i.e.,  $P = \bigcup_{wfn \in WFN} wfn.P, T = \bigcup_{wfn \in WFN} wfn.T$  and  $F = \bigcup_{wfn \in WFN} wfn.F$ ;
- set  $P^\circ$  contains the communication places, which represent the messaging relations between  $cbp$ 's constituent processes;
- set  $F^\circ$  contains the arcs that connect communication places, i.e.,  $F^\circ \subseteq P^\circ \times T \cup T \times P^\circ$ . Each arc  $f^\circ \in F^\circ$  links from or links to a transition via an And-Split/Join structure, i.e.,  $\forall f^\circ = (t, p^\circ) \in F^\circ : \exists f = (t, p) \in F$  and  $\forall f^\circ = (p^\circ, t) \in F^\circ : \exists f = (p, t) \in F$ , where  $p^\circ \in P^\circ, p \in P$  and  $t \in T$ .



**Fig. 6** A CorPN net sample

- Further, we confine that each transition cannot handle message sending and receiving simultaneously, i.e.,  $\forall f1^\circ = (t, p1^\circ) \in F^\circ : \neg \exists f2^\circ = (p2^\circ, t) \in F^\circ$  and vice versa, where  $p1^\circ, p2^\circ \in P^\circ, p1^\circ \neq p2^\circ$  and  $t \in T$ .
- set  $V = \{v_1, v_2, \dots, v_n\}$  contains the token group IDs for the tokens of constituent WF-Nets, and  $n = |WFN|$ , i.e., the number of constituent WF-Nets;
- mapping  $Pem: F \rightarrow PE$  assigns each arc with a permission expression for the purpose of restricting token flows.  $PE$  is a set of permission expressions defined on  $V$  and cardinality parameters, i.e., ‘:1’ (one) and ‘:n’ (many), where ‘:1’ is the default parameter for arcs.

According to the definition of CorPN net, we can generate the CorPN net as shown in Fig. 6 for the collaboration scenario discussed in Sect. 1. In this figure, three constitute WF-Nets (distinguished by place patterns) represent the product ordering, production and shipping processes in Fig. 1, respectively, where the transitions and places are named with corresponding suffixes. These WF-Nets are assigned with different token group IDs, such as  $x, y$  and  $z$ , respectively, to distinguish between their tokens. These WF-Nets are connected via communication places,

such as  $px1, px2, \dots$ , into a CorPN net, and the connection structures represent the messaging interactions between these WF-Nets. Further, the cardinality parameters along related arcs describe the message multi-casting details. For example, the And-Join structure consisting of transition  $tb1$  and two input arcs with cardinality parameters  $x : n$  and  $y : 1$  indicates that the manufacturer’s one production instance may receive multiple orders from the product ordering instances of different retailers.

### 3.3 CorPN net system

As discussed in last section, CorPN net model targets at describing process collaboration at build time. Yet, at run time, the instances of different processes may interact dynamically, and result in diverse correlations. In the example introduced in Sect. 1, once the manufacturer accepts an order from a retailer, their production instance and product-ordering process instance are *directly correlated*. When the manufacturer outsources the product delivery to a shipper, the retailer’s product-ordering instance is therefore *indirectly correlated* with the shipper’s shipping instance via the manufacturer’s production instance. In CorPN net context, the process instances are delegated by tokens of corresponding WF-Nets, and thereby tokens may get directly or indirectly correlated correspondingly.

For each token  $tk$ , we develop a correlation structure  $cor^{tk}$  to record its directly and indirectly correlated tokens.

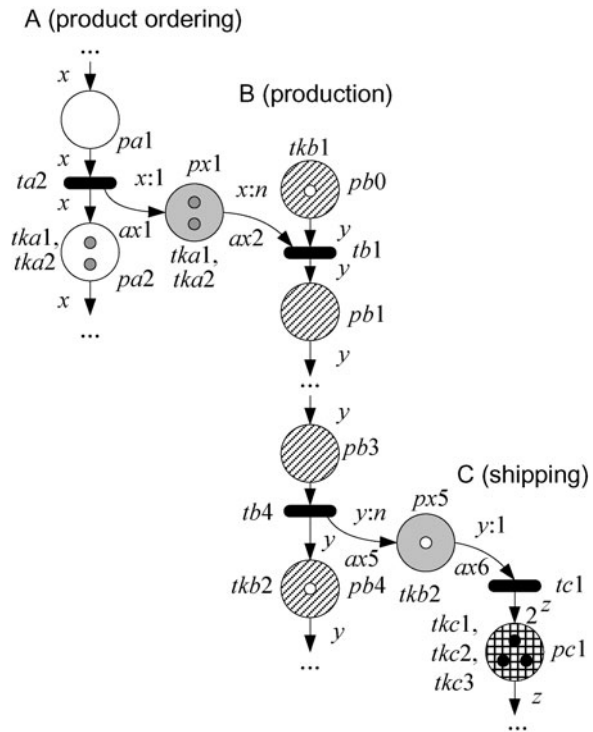
**Definition 2** (Correlation structure) For token  $tk$ , its correlation structure  $cor^{tk}$  is defined as tuple  $(tk, Tk_1, Tk_2, \dots, Tk_n, R^{tk})$ , where

- set  $Tk_i (1 \leq i \leq n)$  denotes the tokens belonging to a token group, which are correlated with  $tk$ .  $n$  is the number of underlying WF-Nets.
- $R^{tk}$  is a binary relation defined between tokens in  $\bigcup_{i=1}^n Tk_i$ . Here,  $tk_x R^{tk} tk_y (tk_x, tk_y \in \bigcup_i Tk_i)$ , denotes that the instances delegated by  $tk_x$  and  $tk_y$  are correlated via  $tk$ .

Token  $tk$  is called *base token* of this correlation structure. Token sets  $Tk_1, Tk_2, \dots, Tk_i$  may update dynamically during collaboration. Take the CorPN net segment in Fig. 7 as an example, the small dots in places represent tokens, and each token, such as  $tka1, tkb1$  and  $tkc1$ , delegates a running business process instance. When  $tka1$  and  $tka2$  flow to transition  $tb1$  via communication place  $px1$ , it indicates that the production instance accepts the orders from two retailers. Therefore, correlation structure  $r^{tkb1}$  turns to be  $\{tkb1, \{tka1, tka2\}, \emptyset\}$  at this moment. Tokens  $tka1$  and  $tka2$  may have correlation structures  $r^{tka1} = \{tka1, \{tkb1\}, \emptyset\}$  and  $r^{tka2} = \{tka2, \{tkb1\}, \emptyset\}$ , respectively. Correlation structures evolve along as their base tokens flow and interact with more tokens.

With the defined correlation structures, we can construct an executable CorPN net system to capture the dynamics of a collaborative business process. Below, we give the definition of CorPN net system.

**Fig. 7** Sample of a running CorPN net system



**Definition 3** (Correspondence Petri net system) For a CorPN net  $cpn$ , its executable system  $cpns$  is defined as tuple  $cpns = (cpn, TKG, CorS, Tk2Cor, IM)$ , where

- set  $TKG = \{Tkg_1, Tkg_2, \dots, Tkg_n\}$  contains the token groups for the constitute WF-Nets of  $cpn$ .  $Tkg_i \cap Tkg_j = \emptyset$ , where  $1 \leq i, j \leq n, i \neq j$ , and  $n$  is the number of token groups;
- set  $CorS$  contains the correlation structures for the tokens in  $STK$ , where  $STK = \bigcup_{i=1}^n Tkg_i$  and  $Tkg_i \in TKG$ ;
- mapping  $Tk2Cor: STK \rightarrow CorS$  assigns each token in  $STK$  with a correlations structure in  $CorS$ ;
- mapping  $IM: cpn.P \rightarrow 2^{STK}$  assigns the initial token distribution to the places of  $cpn$ , i.e., the initial marking of  $cpn$ .

In this CorPN net system, tokens act as a special semantic carrier. Inside a WF-Net, a token represents an instance of corresponding business process, and a token in a communication place represents a sent message.

For a CorPN net system, its execution is under certain enabling rule and firing rule, and its execution status is represented by its marking. In the following, we introduce these notions in details.

**Definition 4** (Marking of CorPN net system) For a CorPN net system  $cpns = (cpn, TKG, CorS, Tk2Cor, IM)$ , its marking at a given time is defined as mapping  $M : cpn.P \rightarrow 2^{STK}$ , where  $STK = \bigcup_{i=1}^n Tkg_i$  and  $Tkg_i \in TKG$ .

In business process context, such a marking indicates the run-time dynamics of the corresponding collaborative business process at that moment. Figure 7 represents a partial view of a running CorPN net system based on the CorPN net shown in Fig. 6. The marking at that moment indicates that the product ordering instances (delegated by tokens  $tka1$  and  $tka2$ ) have finished task ‘Place Order with Manufacturer’ (delegated by transition  $ta2$ ), and two messages (i.e., two product orders delegated by tokens  $tka1$  and  $tka2$  in communication place  $px1$ ) are sent to the manufacturer’s production process. Similarly, tokens  $tkb2$  in places  $pb4$  and  $px5$  indicate that the production instance has finished task ‘Schedule Delivery’ and a shipment order is sent to the shipper.

According to the extensions to classical Petri nets, we also extend the transition enabling rule and firing rule, which regulate the execution of the CorPN net, the following transition enabling rule and firing rule regulate the execution of our CorPN net system.

### Enabling rule

(a) For a non-interaction-responding transition  $t$  of CorPN net  $cpn = (WFN, P, T, F, P^\circ, F^\circ, V, Pem)$ , it is enabled at marking  $M$  if the token distribution of its prior place(s) satisfies the permission expression(s) of related arcs. This rule can be defined as the following condition.

At marking  $M, \forall p \in \bullet t, \text{arc } a = (p, t) : \exists Tk \subseteq [p]$  satisfies expression  $Pem(a)$ , where  $\bullet t$  denotes the set of  $t$ ’s prior places, and  $[p]$  denotes the set of tokens in place  $p$  at marking  $M$ . (1)

(b) For an interaction-responding transition  $t$  of CorPN net  $cpn = (WFN, P, T, F, P^\circ, F^\circ, V, Pem)$ , its enabling is subject to both *local tokens* (the tokens belonging to the token group for  $t$ ’s WF-Net) and the *foreign tokens* (the tokens belonging to other token groups) in the communication places linking to  $t$ . Here, we discuss the enabling of an interaction-responding transition in two following situations:

- Correlation Initiation Interaction: Transition  $t$  is enabled at marking  $M$  under condition (1), and the involved local tokens and foreign tokens will be correlated.
- Correlation-Constrained Interaction. Transition  $t$  is enabled at marking  $M$ , if the tokens are correlated with the foreign tokens, besides condition (1). This can be defined as below.

In addition to condition (1), at marking  $M, \forall p1 \in \bullet t \cap P^\circ$  and  $p2 \in \bullet t \cap P : \exists Tk1 \subseteq [p1]$  and  $Tk2 \subseteq [p2]$  that tokens in  $Tk1$  are correlated with tokens in  $Tk2$ , where  $\bullet t$  denotes the set of  $t$ ’s prior places, and  $[p]$  denotes the set of tokens in place  $p$  at marking  $M$ .

### Firing rule

For an enabled transition  $t$  of CorPN  $cpn = (WFN, P, T, F, P^\circ, F^\circ, V, Pem)$ , when  $t$  fires, it consumes the specified token(s) in its prior place(s) according to the Enabling Rule, and outputs token(s) to its posterior place(s) according to the corresponding permission expression(s) of its output arc(s). This rule can be defined as below.

When  $t$  is fired,  $\forall p \in t\bullet, \text{arc } a = (t, p)$ : the output tokens flowing via arc  $a$  to  $p$  satisfy expression  $Pem(a)$ , where  $t\bullet$  denotes the set of  $t$ ’s posterior places.

With these rules, the CorPN net model can well simulate the collaboration between business processes.

#### 4 Properties of correspondence Petri nets

The behavioural properties of a CorPN net are subject to both constituent WF-Nets and the communication behaviours between them. In this section, we present a methodology to analyse the behavioural properties of constituent WF-Nets and the interactions between them.

##### 4.1 Preliminary

In WF-Net context, property analysis plays an important role in workflow verification and validation. Typically, soundness property has been proposed in [34, 35] to check task reachability, execution deadlock and proper termination of business processes. Particularly, these help process analysts design and verify the structure of single business processes. In regard to collaborative business processes, the task reachability requirement of soundness is too restrictive. The reason is that when an individual business process is designed, it is designed to be able to handle different interactions with diverse external processes using proper messages and tasks. Once it starts collaborating with a specific external process, the collaboration may only use part of its interaction capability, which indicates that the tasks and messages designed to support other interactions become redundant to this collaboration case. As a collaborative business process is only used to support the collaboration between specific business processes, the task reachability requirement is therefore not necessary. By relaxing the soundness property, Martens has proposed the weak soundness property [36] to describe the behavioural property of a composite WF-Net. Details of soundness and weak soundness are given in the [Appendix](#).

For a CorPN net, its ownership of multiple source places and sink places violates the standard format of Petri net. Therefore, we first need to transform a CorPN net and adapt it to standard Petri net format. The transformed CorPN net will be used to discuss its behavioural properties.

**Definition 5** (Transformed CorPN net) For CorPN net  $cpn = (WFN, P, T, F, P^\circ, F^\circ, V, Pem)$ , its transformed net  $\overline{cpn}$  is defined as tuple  $(WFN, \bar{P}, \bar{T}, \bar{F}, P^\circ, F^\circ, V, \overline{Pem})$ , where

- set  $\bar{P} = P \cup \{i, o\}$ , where  $i$  and  $o$  are a common source place and a common sink place, respectively;
- set  $\bar{T} = T \cup \{ti, to\}$ , where  $ti$  and  $to$  are two common transitions;
- set  $\bar{F} = F \cup \{fi, fo\} \cup \bigcup_{j=1}^n \{(ti, pi_j)\} \cup \bigcup_{j=1}^n \{(po_j, to)\}$ , where arcs  $fi = (i, ti)$ ,  $fo = (to, o)$ , places  $pi_j$  and  $po_j$  indicate the source place and the sink place of the  $j$ -th constituent WF-Net, and  $n = |WFN|$ , i.e., number of constituent WF-Nets,
- $\overline{Pem} = Pem \cup \{(fi, \sum_{j=1}^n 2^{v_j}), (fo, \sum_{j=1}^n 2^{v_j})\} \cup \bigcup_{j=1}^n \{(fi_j, 2^{v_j}), (fo_j, 2^{v_j}) \mid fi_j = (ti, pi_j), fo_j = (po_j, to)\}$ , where  $v_j \in V$  is the token group ID for the  $j$ -th WF-Net and  $n = |WFN|$ .

Basically, this transformation inserts a common source place and a common sink place into the original CorPN net via an AND-Split structure and an AND-Join structure with related arcs, transitions and permission expressions. For example, for the CorPN net shown in Fig. 8(a), Fig. 8(b) gives the transformed net. Permission expression  $2^x + 2^y$  ( $x, y \neq \text{null}$ ) denotes that any tokens of any token groups can flow from place  $i$  to transition  $ti$  and from transition  $to$  to place  $o$ .

To reduce the scale of analysis, we decompose the CorPN net into several modules, and analyse the communicating behaviours of each module first. Extended from Marten's work [36], in CorPN context we define a *workflow module* to be a constituent WF-net with its communication places and cardinality parameters. (The detailed definition is given in the Appendix.) This decomposition is subject to the condition that the cardinality parameters of corresponding communication-involving AND-Split/Join structures match each other. For example, the communication-involving AND-Split/Join structures in the CorPN net in Fig. 8 (a) all own the  $x : n$  and  $y : 1$  cardinality parameters. Therefore, this CorPN net can be decomposed into two workflow modules shown in Fig. 8 (c).

Another distinction of CorPN net is that it owns multiple tokens, as tokens of different token groups delegate the collaborating instances of different processes. Thus, property analysis should be conducted according to a set of correlated tokens rather than a single token. Therefore, we confine that the tokens must be with a correlation closure as a pre-condition. This pre-condition prevents the influence caused by an incomplete token set.

Given a CorPN net system  $cpns$  and the set of all tokens  $STK$  in the constituent WF-Nets of  $cpns$ , we define the token correlation closure as follows.

**Token Correlation Closure.** The tokens in set  $TK \subseteq STK$  are said to be with a close correlation, if any token in  $TK$  has its all correlated tokens in  $TK$  during the whole execution time of  $cpns$ . This condition can be formalised as below

$\forall tk \in TK \text{ and } tk \text{ cor } tk' : \exists tk' \in TK$ , where

- *cor* is a transitive and symmetric relation defined over tokens;
- $tk \text{ cor } tk'$  indicates that  $tk$  and  $tk'$  are correlated with each other, either directly or indirectly.

## 4.2 Analysing CorPN net properties

The analysis on CorPN net properties uses communication graph as an important tool. Conventional communication graph has been proposed to analyse the communication sequence of a single workflow module [36]. In the context of our CorPN net model, we adopt communication graphs to analyse the behavioural properties of CorPN nets by checking the compatibility between the communication sequences of involved workflow modules.

Figure 9 gives the communication graphs for the workflow modules shown in Fig. 8 (c). In each communication graph, a circle denotes a set of markings, and each black dot denotes a communication-involving transition. The symbols along its incoming and outgoing arcs indicate the input message and output message for the communication, respectively. The root represents the initial marking of the module.

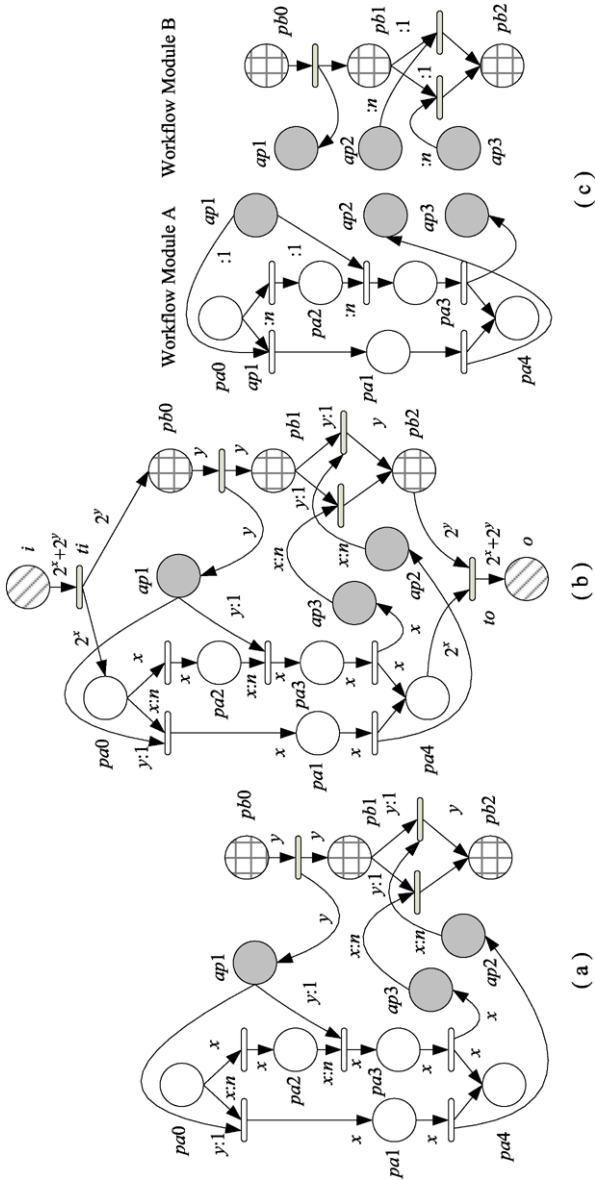
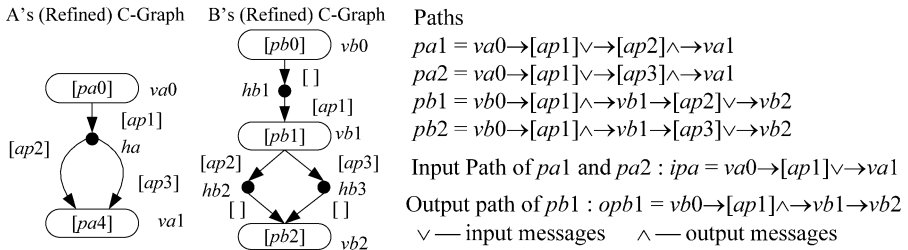


Fig. 8 Samples of transformed CopPN net and workflow modules



**Fig. 9** Communication graphs and paths

The definition and the generation algorithm of communication graph are given in the [Appendix](#).

A path from the root node to a leaf node creates a *communication path*, which contains the passing transitions and associated input/output messages. Further, we separate a communication path into an *input path* and an *output path*, which correspond to the ordered lists of input/output message bags extracted from communication path, respectively. An output path *outp* is said to *match* an input path *inp* if *outp* provides sufficient messages to *inp* in corresponding order. For example, workflow module *A*'s communication graph has two paths *ptha1* and *ptha2*, and these two paths have the same input path *ipa*, as shown in Fig. 9. Workflow module *B*'s communication graph has two paths *pthb1* and *pthb2*. *pthb1*'s output path *opb1* matches *A*'s input path *ipa*.

The communication graph is originally proposed to represent all possible reachable markings and inputs/outputs of a workflow module. In CorPN net context, once the communicating workflow modules are specified, some possible markings and input/output messages turn to be redundant. To precisely describe the interaction behaviours between given workflow modules, we need to refine the communication graphs of collaborating workflow modules by removing unreachable markings.

The following algorithm details the procedure of refining communication graphs. In this algorithm, function *paths*(*cg*) returns all paths in graph *cg*; Functions *inPath*(*pth*) and *outPath*(*pth*) return the input path and output path of path *pth*, respectively. Function *matches*(*inp*, *outp*) returns whether output path *outp* matches input path *inp*; Function *unmatchedNode*(*inp*, *outp*) returns the node of output path *outp*, and this node is immediate after *outp*'s first unmatched message bag with input path *inp*; Function *adjustCG*(*inp*, *cx*, *cy*) returns the graph adjusted according to input path *inp*. This algorithm also gives the details of this function.

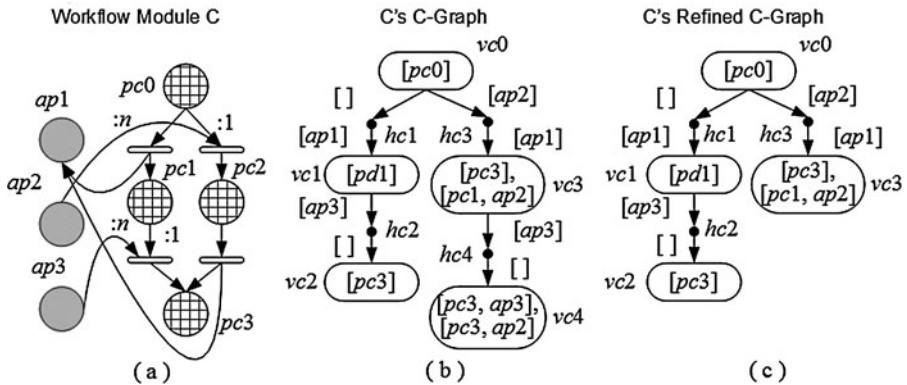
Figure 10 (a) and (b) show another workflow module *C* and its communication graph. If we combine modules *A* and *C*, we can find that *A*'s two output paths do not match the input path of *C*'s right-side communication path, i.e.,  $vc0 \rightarrow [ap2] \vee \rightarrow vc3 \rightarrow [ap3] \rightarrow vc4$ . Using above algorithm, *C*'s communication path can be refined into the graph shown in Fig. 10 (c). Module *A*'s refined communication graph remains the same.

After the refinement, a communication graph presents all the possible markings and input/output messages of a workflow module in the collaboration with a specific partner module. In such a refined communication graph, leaf nodes are classified into following types:

**Algorithm 1** Refining communication graphs

**Input:**  $cx$ : communication graph of a workflow module;  
 $cy$ : communication graph of the partner workflow module;  
**Output:**  $cx'$ : refined  $cx$ ;  
 $cy'$ : refined  $cy$ .

1. **do while** ( $\exists inp \in inPath(p), outp \in inPath(p'), pth \in paths(cx/cy), pth' \in paths(cy/cx) : \neg matches(inp, outp)$ )
2. **if**  $inp$  belongs to graph  $cx$  **then**  $cx = adjustCG(inp, cx, cy)$  **else**  $cy = adjustCG(inp, cy, cx)$
3. **end while**
4. **set**  $cx = cx'$  and  $cy = cy'$ ;  
*// function adjustCG is given below*  
*function adjustCG(ipth, cgx, cgy)*
  - u-1. **set**  $maxNode$  and  $tempNode$  be the root node of  $ipth$ ;
  - u-2. **for each** path  $pa \in paths(cgy)$
  - u-3.  $opth = outputPath(pa)$ ;
  - u-4. **if**  $\neg matches(ipth, opth)$  **then**
  - u-5.  $tempNode = unmatchedNode(ipth, opth)$ ;
  - u-6. **if**  $tempNode$  is beyond  $maxNode$  along  $ipth$  **then**  $maxNode = tempNode$
  - u-7. **endif**
  - u-8. **next**
  - u-9. **if**  $maxNode$  is not the root node of  $ipth$  **then** cut off the part of  $ipth$ 's belonged path from  $maxNode$  in graph  $cgx$
  - u-10. **return**  $cgx$



**Fig. 10** Communication graphs and a workflow module

*good leaf node*—The leaf node only contains the final marking of the module.

*ill leaf node*—Each element of the leaf node contains other places besides the sink place.

*bad leaf node*—Other leaf nodes.

For example, nodes  $vc2$  and  $vc3$  in Fig. 10 (c) are good leaf node and bad leaf node, respectively. Node  $vc4$  in Fig. 10 (b) shows an example of ill leaf node, though it is not in a refined communication graph. Under the assumption that all constituent WF-Nets are sound, these leaf node types provide the following clues for behavioural property analysis of underlying transformed CorPN net:

- (4-1) An ill leaf node indicates the improper termination case of transformed CorPN net. Because a communication place always links to a transition of a WF-Net via an And-Join structure (please refer to CorPN net definition), tokens in a communication place cannot fire any transition of the target WF-Net unless the transition itself is enabled. This implies that the improper termination cases of transformed CorPN net may only leave excessive tokens in communication places rather than any other places of constituent WF-Nets.
- (4-2) A bad leaf node indicates the non-termination case of corresponding constituent WF-Net, and thereby results in the non-termination of transformed CorPN net. In case that a bad leaf node contains the final marking as its element, the other contained markings hold the possibility of non-termination.
- (4-3) A path ends at a good leaf node indicates the communication pattern that each WF-Net can always terminate properly and leave no tokens in communication places.

According to above discussion, we have the following theorem to judge the weak soundness of a CorPN net from its communication behaviours.

**CorPN Weak Soundness Theorem** *If all constituent WF-Nets are sound and their refined communication graphs have no good leaf nodes, the corresponding corPN is weakly sound.*

*Proof* Good leaf nodes guarantee the Termination and Proper Termination conditions. □

## 5 Discussion

This paper looked into the instance correspondence issue in collaborative business process context. By establishing a CorPN net model, we propose a novel method to specify instance correspondences in a collaborating business process. This method captures the dynamics and diversity of business collaboration in terms of workflow cardinality and instance correlations. With this method, an organisation can explicitly define its perceivable view of a collaborative business process and track its execution. Table 1 compares our CorPN net model and other approaches in details. CorPN net model is specialised in handling inter-organisational instance correspondences and analysing the behavioural properties of collaborative business processes.

With cardinality parameters, the CorPN net model explicitly identifies the quantitative relation between collaborating business processes. The proposed message casting structures well describe the various interaction patterns between business processes. These structures can combine with each other to cater for the increasingly complex process interactions in business collaboration. With these features,

**Table 1** Comparison with other approaches

	<i>Facilitating technologies</i>	<i>Functioning Scope</i>	<i>Handling objects</i>	<i>Instance correlation</i>	<i>Behaviour analysis support</i>
<i>workflow pattern</i>	Patterns and conditions	Intra org	Activities or sub processes	N/A	Classical WF-Net analysis methodology
<i>Service interaction pattern</i>	Patterns and conditions	Intra/inter org	Activities or sub processes	N/A	N/A
<i>Multiple Instantiation</i>	Instance sets	Intra org	Activities or sub processes	N/A	N/A
<i>WS-BPEL</i>	Message correlation sets	Inter org	Business processes in a business collaboration	By messages	N/A
<i>CorPN</i>	Extended Petri net model	Inter org	Business processes in a business collaboration	By correlation structures	Comprehensive analysis methodology using extended communication graphs

the CorPN net model can sufficiently simulate the execution of a collaborative business process, including the formation, transfer and derivation of instance correlations. The CorPN net model and analysis method are applicable to the collaborative business processes of most collaboration scenarios, such as supply chain, cross-company manufacturing, virtual enterprises, and so on.

Take the collaborative business process in Sect. 1 as an example, in the generated CorPN net shown in Figs. 6 and 7, the  $x$ -type tokens in place  $px1$  represent the orders from different product ordering instances. Once transition  $tb1$  is enabled and fired according to the enabling rule and firing rule, the correlation between product ordering instances and corresponding production instance are recorded. When the manufacture contacts shippers, the recorded correlation is transferred to the respective shipping instances via the binary relation in the correlation structure. With this knowledge, the shipping instances can automatically find the proper product ordering instances, and start the product delivery. In comparison, WF-Net can well simulate the execution of each single process. Though it can represent a collaborative business process by creating a transformed WF-Net, it can hardly support the multi-casting messaging interactions and trace correlation evolvments owing to the lack of cardinality and correlation support. Proclat basically targets at the interactions between intra-organisational processes, and requires the instance correlation to be predefined before messaging, which is totally not practicable in this scenario.

The proposed analysis method for behavioural properties enables the verification of collaborative business processes at build time, by checking the communication behaviours between constituent business processes. In comparison, WF-Net and other Petri net based approaches mainly check the classical soundness property of single workflows. Owing to this presupposition, soundness property requires the workflow to be strictly compliant with its three conditions. Nevertheless, these conditions are overstrict and even unnecessary in the business collaboration scenario, as discussed in Sect. 4. Upon this issue, our approach has relaxed the soundness condition to weak soundness by allowing the message redundancy in inter-process communications. This feature makes CorPN approach most practicable for communication behaviour checking over Petri net based workflows. Process-algebraic workflow checking approaches, such as [37], are specialised in analysing communication behaviours between concurrent processes. Yet, these approaches require the algebra-based formalisation on workflow process representation, which is very different from the popular graph-based workflow representation. This makes it difficult to be applicable to current workflows, compared with ours. Other workflow checking methods mainly focus on the aspect of structural correctness or temporal dependency, rather than interaction behaviours, and therefore are not referred in this paper.

A practical concern of applying CorPN net goes to the public process perception, because the design of a CorPN net requires the full knowledge of partners' business processes, which is not always possible in practice. This issue is actually very important, and has already attracted many research efforts [38–42]. Our previous work on relative workflow [31, 32] has also proposed a comprehensive solution for process view/perception control, which is applicable to solve the process privacy issue at CorPN design time. To keep the paper concentrated on a specific topic, we do not go into this issue in this paper. Interested readers may refer to the mentioned literatures.

The shift to the proposed instance correspondence method may bring some trade-offs, which can be potential limitations, although they may be outweighed by many advantages offered by our methodology. Compared to the representation of other workflow modelling approaches, such as WS-BPEL and Event-driven Process Chains (EPC), the Petri net representation of the CorPN net turns complicated when the collaborative business process is large. This is because that Petri nets rely on extra places and transitions to represent the control flow logic. Nevertheless, this shortcoming is compensated by the strong behaviour analysis capability of Petri nets.

## 6 Conclusion and future work

Our work has focused on the instance correspondence issue in the setting of collaborative business processes. In our method, instance correspondences have been characterised in terms of workflow cardinality and instance correlation. The CorPN net model has been proposed to represent instance correspondences with particular extensions for workflow cardinality and instance correlations. A methodology has also been proposed to analyse the behavioural properties of CorPN nets to examine and verify collaborative business processes. To prove the concept, we have implemented a prototype using SAP's Nehemiah and Maestro.

Our future work is to incorporate the proposed method into Business Process Management Notation (BPMN) or WS-BPEL languages. This future work is expected to provide a comprehensive solution for collaborative business process applications.

**Acknowledgement** The work reported in this paper is supported by the Australian Research Council linkage project, “An Organisation Oriented Framework for Collaborative Business Processes” (LP0669660), with industry partner SAP Research, Australia.

## Appendix

The definition of WF-Net is referenced from work [13].

**Definition (WF-Net)** A Petri net  $PN = (P, T, F)$  is a WF-Net if and only if:

- (i) There is one source place  $i \in P$  such that  $\bullet i = \emptyset$ .
- (ii) There is one sink place  $o \in P$  such that  $o \bullet = \emptyset$ .
- (iii) Every node  $x \in P \cup T$  is on a path from  $i$  to  $o$ .

where

- $P$  denotes the set of places.
- $T$  denotes the set of transitions.
- $F$  denotes the set of arcs that connect places and transitions.

The places, transitions and arcs of a WF-Net satisfy that  $P \cap T = \emptyset$ ,  $P \cup T \neq \emptyset$ ;  $F \subseteq P \times T \cup T \times P$ .

- $\bullet x$  denotes the set of  $x$ 's prior places if  $x$  is a transition, or the set of  $x$ 's prior transitions if  $x$  is a place.
- $y \bullet$  denotes the set of  $y$ 's posterior places if  $y$  is a transition, or the set of  $y$ 's posterior transitions if  $y$  is a place.

The following definition of soundness is referenced from work [34].

**Definition (Soundness)** A workflow system  $(PN, i)$  with a WF-Net  $PN = (P, T, F)$  is sound if and only if

- (*Termination*) For every marking  $M$  reachable from initial marking  $i$ , there exists a firing sequence leading from marking  $M$  to final marking  $o$ , i.e.,  $\forall M(i \xrightarrow{*} M) \Rightarrow (M \xrightarrow{*} o)$ .
- (*Proper termination*) Marking  $o$  is the only marking reachable from  $i$  with at least one token in the sink place, i.e.,  $\forall M(i \xrightarrow{*} M \wedge M \geq o) \Rightarrow (M = o)$ .
- (*No dead transitions*) There are no dead transitions in the WF-Net in marking  $i$ , i.e.,  $\forall t \in T, \exists M, M' : i \xrightarrow{*} M \xrightarrow{t} M'$ .

Here,  $i$  and  $o$  denote the initial marking (there is exactly one token in the source place and no token in any other place) and final marking (there is exactly one token in the sink place and no token in any other token), respectively.

The following definition and algorithm are from work [36].

**Definition** (Weak soundness) A workflow system  $(PN, i)$  is weakly sound if and only if the *termination* condition and the *proper termination* condition of soundness hold (please refer to the definition of soundness in the [Appendix](#) for the details of these conditions).

**Definition** (Workflow module) In CorPN context, a workflow module corresponds to a constituent WF-Net with related communication places. Formally, for a WF-Net  $PN1 = (P, T, F)$  belonging to CorPN net  $corPN$ , its workflow module can be defined as  $(P', T, F')$ , where

- $P' = P \cup Px$ ,  $Px$  denotes the set of communication places connected to  $PN1$  in  $corPN$ ;
- $F' = F \cup Fx$ ,  $Fx$  denotes the set of arcs that link  $PN1$  and places in  $Px$ .

**Definition** (Communication graph) A communication graph  $((V, H, E), m)$  is a directed, strongly connected, labelled, bipartite graph such that:

- The graph has two kinds of nodes: visible nodes  $V$  and hidden nodes  $H$ .
- Each arc  $e \in E$  connects two nodes of different kinds.
- The graph has a definite root node  $v_0 \in V$ , and each leaf is a visible node.
- The labelling  $m$  maps each visible node to a set of markings of the corresponding WF-Net, and each links to a bag of messages.

The following algorithm describes the procedure of generating the communication graph for a workflow module. For a given marking  $z$ , function  $INP(z)$  returns a set of minimal inputs such that all behaviour of the module is possible. For a given marking  $z$  and an input  $i$ , function  $OUT(z + i)$  returns a set of maximal outputs such that  $o \in OUT(z + i)$  is an output if there is a reachable marking  $(z' + o) \in OUT(z + i)$ . A communication step  $(z, i, o, z')$  denotes an interaction from marking  $z + i$  to marking  $z' + o$ , where  $i$  and  $o$  represent the input and output of the interaction.  $S(M)$  denotes the set of all communication steps.

This algorithm starts with the root node  $v_0$  labelled with the initial marking.

1. For each marking within the label of  $v_k$  calculate the set of activated inputs:  $\bigcup_{z \in m(v_k)} INP(z)$ .
2. For each activated input  $i$  within this set:
  - (a) Add a hidden node  $h$ , add a new edge  $(v_k, h)$  with the label  $i$ .
  - (b) For each marking within the label of  $v_k$  calculate the set of possible outputs:  $\bigcup_{z \in m(v_k)} OUT(z + i)$ .
  - (c) For each possible output  $o$  within this set:
    - i. Add a visible node  $v_{k+1}$ , add a new edge  $(h, v_{k+1})$  with the label  $o$ .
    - ii. For each marking  $z \in m(v_k)$  and for each communication step  $(z, i, o, z') \in S(M)$  add  $z'$  to the label of  $v_{k+1}$ .
    - iii. If there exists a visible node  $v$  such that  $m(v_{k+1}) = m(v)$  then merge  $v$  and  $v_{k+1}$ . Otherwise, go to step 1 with node  $v_{k+1}$ .

## References

1. Ferguson, D.F., Stockton, M.L.: Enterprise business process management—architecture, technology and standards. In: The 4th International Conference on Business Process Management, Vienna, Austria, pp. 1–15 (2006)
2. Khoshafian, S.: Service Oriented Enterprises. Auerbach Publications, New York (2007)
3. Liu, C., Li, Q., Zhao, X.: Challenges and opportunities in collaborative business process management. *Inf. Syst. Front.* **11**, 201–209 (2009)
4. Weske, M., van der Aalst, W.M.P., Verbeek, H.M.W.: Advances in business process management. *Data Knowl. Eng.* **50**, 1–8 (2004)
5. Samtani, G., Healey, M.J., Samtani, S.: B2B Integration: A Practical Guide to Collaborative e-Commerce. Imperial College Press, London (2002)
6. Bussler, C.: B2B Integration. Springer, New York (2003)
7. Smith, H., Fingar, P.: Business Process Management—The Third Wave. Meghan-Kiffer Press, Tampa (2003)
8. Krogstie, J., Opdahl, A.L., Brinkkemper, S.: Conceptual Modelling in Information Systems Engineering. Springer, Berlin (2007)
9. Azadeh, A., Haghnevis, M., Khodadadegan, Y.: Design of the integrated information system, business, and production process by simulation. *J. Am. Soc. Inf. Sci. Technol.* **59**, 216–234 (2007)
10. Mehandjiev, N., Grefen, P.: Dynamic Business Process Formation for Instant Virtual Enterprises. Springer, Berlin (2010)
11. Chiu, D.K.W., Karlapalem, K., Li, Q., Kafeza, E.: Workflow view based e-contracts in a cross-organizational e-services environment. *Distrib. Parallel Databases* **12**, 193–216 (2002)
12. Atluri, V., Chun, S.A., Mukkamala, R., Mazzoleni, P.: A decentralized execution model for inter-organizational workflows. *Distrib. Parallel Databases* **22**, 55–83 (2007)
13. van der Aalst, W.M.P., van Hee, K.: Workflow Management Models, Methods, and Systems. MIT Press, Cambridge (2002)
14. van der Aalst, W.M.P., ter Hofstede, A.H.M., Kiepuszewski, B., Barros, A.P.: Workflow patterns. *Distrib. Parallel Databases* **14**, 5–51 (2003)
15. van der Aalst, W.M.P., ter Hofstede, A.H.M.: YAWL: Yet Another Workflow Language. *Inf. Sci.* **30**, 245–275 (2005)
16. Dumas, M., ter Hofstede, A.H.M.: UML activity diagrams as a workflow specification language. In: The 4th International Conference on the Unified Modeling Language, Modeling Languages, Concepts, and Tools, Toronto, Canada, pp. 76–90 (2001)
17. Barros, A.P., Dumas, M., ter Hofstede, A.H.M.: Service interaction patterns. In: The 3rd International Conference on Business Process Management (BPM 2005), Nancy, France, pp. 302–318 (2005)
18. Guabtni, A., Charoy, F.: Multiple instantiation in a dynamic workflow environment. In: The 16th International Conference on Advanced Information Systems Engineering (CAiSE 2004), Riga, Latvia, pp. 175–188 (2004)
19. Mulyar, N., Aldred, L., van der Aalst, W.M.P.: The conceptualization of a configurable multi-party multi-message request-reply conversation. In: The 9th International Symposium on Distributed Objects and Applications, pp. 735–753 (2007)
20. van der Aalst, W.M.P., Barthelmeß, P., Ellis, C.A., Wainer, J.: Proclets: a framework for lightweight interacting workflow processes. *Int. J. Coop. Inf. Syst.* **10**, 443–481 (2001)
21. van der Aalst, W.M.P., Barthelmeß, P., Ellis, C.A., Wainer, J.: Workflow modeling using proclets. In: The 7th International Conference on Cooperative Information Systems, Eilat, Israel, pp. 198–209 (2000)
22. Russell, D., Dew, P., Djemame, K.: Service-based collaborative workflow for DAME. In: IEEE International Conference on Service Computing, Orlando, Florida, USA, pp. 139–146 (2005)
23. BizAgi, Wiki.BizAgi: Process Data. [http://wiki.bizagi.com/en/index.php?title=Process\\_Data](http://wiki.bizagi.com/en/index.php?title=Process_Data)
24. IBM, WebSphere BPM. <http://www-01.ibm.com/software/au/websphere/>
25. Tibco, ActiveMatrix@BPM. <http://www.tibco.com/solutions/bpm/resource-library/>
26. Andrews, T., Curbera, F., Dholakia, H., Golland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I., Weerawarana, S.: Business Process Execution Language for Web Services
27. SAP, NetWeaver Process Integration. <http://www.sdn.sap.com/irj/sdn/nw-pi71>
28. IBM, WebSphere Process Server. <http://www-01.ibm.com/software/au/websphere/>
29. van der Aalst, W.M.P., Mooij, A.J., Stahl, C., Wolf, K.: Service interaction patterns, formalization, and analysis. In: The 9th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, Bertinoro, Italy, pp. 42–88 (2009)

30. Charfi, A., Mezini, M.: AO4BPEL: an aspect-oriented extension to BPEL. *World Wide Web J.* **10**, 309–344 (2007)
31. Zhao, X., Liu, C., Yang, Y., Sadiq, W.: Aligning collaborative business processes: an organisation-oriented perspective. *IEEE Trans. Syst. Man Cybern., Part A, Syst. Hum.* **39**, 1152–1164 (2009)
32. Zhao, X., Liu, C.: Steering dynamic collaborations between business processes. *IEEE Trans. Syst. Man Cybern., Part A, Syst. Hum.* **40**, 743–757 (2010)
33. Zhao, X., Liu, C., Yang, Y., Sadiq, W.: Handling instance correspondence in inter-organisational workflows. In: *The 19th International Conference on Advanced Information Systems Engineering*, Trondheim, Norway, pp. 51–65 (2007)
34. van der Aalst, W.M.P.: Verification of workflow nets. In: *The 18th International Conference on Application and Theory of Petri Nets*, pp. 407–426 (1997)
35. Russell, N.C., ter Hofstede, A.H.M.: The language: rationale and fundamentals. In: ter Hofstede, A.H.M., van der Aalst, W.M.P., Adams, M., Russell, N.C. (eds.) *Modern Business Process Automation: YAWL and Its Support Environment*, pp. 23–97. Springer, Berlin (2010)
36. Martens, A.: Analyzing web service based business processes. In: *The 8th International Conference on Fundamental Approaches to Software Engineering*, Edinburgh, UK, pp. 19–33 (2005)
37. Wong, P.Y.H., Gibbons, J.: A process-algebraic approach to workflow specification and refinement. In: *6th International Symposium on Software Composition*, Braga, Portugal, pp. 51–65 (2007)
38. Issam, C., Shahram, D., Samir, T.: The view-based approach to dynamic inter-organizational workflow cooperation. *Data Knowl. Eng.* **56**, 139–173 (2006)
39. Chiu, D.K.W., Karlapalem, K., Li, Q., Kafeza, E.: Workflow view based e-contracts in a cross-organizational e-services environment. *Distrib. Parallel Databases* **12**(2–3), 193–216 (2002)
40. Liu, D.R., Shen, M.: Workflow modeling for virtual processes: an order-preserving process-view approach. *Inf. Sci.* **28**, 505–532 (2003)
41. Schulz, K.A., Orłowska, M.E.: Facilitating cross-organisational workflows with a workflow view approach. *Data Knowl. Eng.* **51**, 109–147 (2004)
42. Eshuis, R., Grefen, P.: Constructing customized process views. *Data Knowl. Eng.* **64**, 419–438 (2008)