Autonomous process collaboration framework for C4ISR system interoperoperation

Lin Yang¹, Minggang Yu^{2,*}, and Qingchao Dong¹

1. Control Engineering Department, Naval Aeronautical and Astronautical University, Yantai 264000, China

2. Institute of Command Information System, PLA University of Science and Technology, Nanjing 210007, China

Abstract: Interoperability plays an important role in the joint command, control, communication, computer, intelligence, surveillance, reconnaissance (C4ISR) operations. Coordinating and integrating operational processes to fulfill a common mission are challenged by the ever-changing battlefield and hence requires a cross-organizational process management that produces an autonomous, flexible and adaptable architecture for collaborative process evolution. The traditional business process collaboration pattern is based on the predefined “public-view” perspective and cannot meet the requirement of the joint task operations. This paper proposes a flexible visibility control mechanism and a dynamic collaboration framework for modeling and generating collaborative processes. The mechanism allows collaborators to define a set of visibility rules to generate different views of the private processes for different collaborations, which gives a great flexibility for the collaboration initiator to decide on an appropriate collaboration pattern. The framework supports collaborators to dynamically and recursively add a new process or even a new organization to an existing collaboration. Moreover, a formal representation of the processes and a set of generation algorithms are provided to consolidate the proposed theory.

Keywords: cross-organizational collaboration, interoperability, collaborative process, joint operations.

DOI: 10.21629/JSEE.2017.06.08

1. Introduction

The U.S. Joint Vision 2020 states that system interoperability is the basis for conducting joint operations effectively [1]. The command, control, communication, computer, intelligence, surveillance, reconnaissance (C4ISR) system is the means by which military units collaborate with one another to fulfill joint operation tasks. The system is usually built networked and is operated cross organizations. Hence, the interoperability involves not only the technical

problems, such as system interfacing, but also cross-organizational coordination.

The joint operations, undertaken by various organizations across army, navy and airforce, are challenged by rapid change of battlefield environments where the reconfigurable collaboration architecture is required. The organizations have to adjust their operation behaviors dynamically in response to uncertain situations, which leads to a strong demand for operation flexibility and autonomy [2,3]. These military organizations merge their business processes together into an interconnected network for sharing their expertise and seeking common interests. As the networked processes are subject to a set of dynamic condition-adjusting-action rules, the participator organizations may join or leave the network at any time. This makes the network configuration and the collaborative business processes highly dynamic [4,5]. Such complex collaboration requires organizations to manage not only their own business processes but also cross-organization coordination. In addition, the diverse authority relationships between the organizations may result in different views of a same business process [6,7], which requires an appropriate process visibility control to prevent authority and privacy violations. These requirements bring challenges to collaborative business process modeling, execution and tracking [8,9].

Over the past two decades, many organizations and individuals have conducted in-depth studies on cross-organizational process interoperation and proposed a number of frameworks and standards. In the following section, we are about to review some typical approaches.

(i) Traditional approaches for process interoperation

Workflow Management Coalition (WfMC) [10] is the first international organization dedicated to the standardization of process interoperation. The interface 4 of the reference model provided by WfMC offers accessibility from one workflow engine to another; and the synchro-

Manuscript received October 14, 2016.

*Corresponding author.

This work was supported by the National Natural Science Foundation of China (61273210) and the National High Technology Research and Development Program of China (863 Program) (2007AA01Z126).

nization control between the processes is realized through inquiry/reply mechanism. Since then, some well-formed business models of cross-organizational interoperability, such as partners interface process (PIPs) [11], event-driven process chains (EPC) [12] and collaborative planning, forecasting, and replenishment (CPFR) [13,14], have been widely used to optimize workflow management. These approaches merely support the collaborators to track their process workflow at different authority levels, but fail to characterize the dynamics of collaborative business processes.

CrossFlow [15,16] is an European research project, aiming at supporting the dynamic cross-organizational workflows of virtual enterprises. The cooperation of these virtual enterprises is based on dynamic service outsourcing specified in electronic contracts. It supports a bi-directional matchmaking process where both customers and providers may describe what they offer and require to/from each other. However, the simple role division of providers and consumers cannot meet the challenges of large-scale collaboration with complex interoperations.

The project of Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Application (ATHENA) [17] puts forward a “top-down” workflow modeling scheme for view-based modeling. The scheme allows the collaboration organizations to make an agreement on a public process and then refine their own parts to generate the private processes.

Collaboration and Interoperability for Networked Enterprises (COIN) [18,19] proposes a “business rule-based Private2Public transformation”, a model-to-model transformation to generate public views of a specific collaboration scenario for specific partners. The corresponding visibility rules are specified with visibility parameters which are divided into data, activity and control flow.

Those workflow based approaches discussed above assume that a public view is predefined for all collaborators that launch processes according to the predefined responsibilities while neglecting some important requirements of cross-organizational process interoperation, such as flexibility and scalability.

(ii) Approaches for autonomous process collaboration

Recent research indicates that the cross-organizational process interoperation needs to adapt to the ever-increasing and volatile requirements of organization evolution, process reengineering and new technology innovation, which demands for a new kind of collaborative business processes. Such processes are not in the control of one single organization; instead, they are enacted by multiple organizations, where no participating party possesses full control over the entire process [20,21].

Breu et al. [22] summarized the challenge of cross-organizational process interoperability into four aspects: flexibility, correctness, traceability and scalability. The flexibility requires that the process interoperation is not fully pre-specified such that the processes can be handled differently depending on the collaboration context. The traceability requires that when and where the processes collaborate with one another can be monitored and recorded such that the states of interoperation are visible. The scalability requires that the architecture of process collaboration is scalable and is allowed to be adjusted to adapt changing requirements.

Zhao et al. [23–25] conducted a long and fruitful research on process interoperability. They proposed a set of important concepts about the relative workflow (RWF) that supports collaborators to build different collaborative processes from their own perspectives and dynamically add a process to an existing collaboration. Those concepts are referenced in our research. However, we find that the approach only supports the waterfall pattern of workflow and fails to consider the more complex activity model where there are bidirectional transitions between activities of a process or bidirectional messaging links between cross-organizational processes, while the activity model is one of the core products of the Department of Defence (DoD) architecture framework [26].

(iii) Approaches for military organization process interoperation

Unlike the research on e-business process collaboration, there are seemingly few literatures on the techniques for cross-organizational process interoperation in military domain and the related research rests on providing a high-level conceptual model.

Kingston et al. [27] pointed out that there have been significant changes in the nature, scale, scope and diversity of military operations. Unilateral operations are rare; joint, combined and coalition operations are now the rule. Coalitions are often fluid, with partners joining and leaving or scaling their commitments during the course of the collaboration or operation. The organizations should have ability to adapt to changing circumstances, i.e. adjust to constant changes in the environment. They focused on the development of an organizational interoperability agility model (OIAM) that aims to capture the dynamic aspects of working in coalitions including the ability of an organization to contribute to the rapid formation and reformation of coalitions. Unfortunately, the OIAM is just a high-level conceptual model and is hard to be refined and validated.

A number of hierarchical evaluation models have been proposed, including NATO C³ technical architecture (NC³TA) reference model for interoperability [28],

organization interoperability model (OIM) [29] and coalition interoperability model (CIM) [30]. In OIM, five levels of organizational maturity describing the ability to interoperate are defined and the non-technical or human-activity aspects of organization's ability to interoperate with one another have been widely recognized as being important contributors to the ability of organizations to work together effectively. However, those models concentrate more on evaluating the interoperability maturity of C4ISR systems rather than optimizing the mechanism of interoperation between the systems.

Liu et al. [31,32] explored heterogeneous agent collaboration technologies to improve interoperability. An ontology-based approach is suggested to support dynamic adjustment of collaboration structure. According to the approach, collaborative agents may publish the additional functions that serve in the collaboration environment when some functions fail, and the agents may be automatically chosen in the collaboration through ontology-based reasoning and rule matching. Nevertheless, no visibility control mechanisms have been mentioned in the research and the approach for the interoperability issue stays on a high level without considering a concrete implementation.

The above approaches model a cross-organizational process as a workflow from public perspective, and there must be a third-party organization or a host organization that predefines a public collaboration architecture for the collaborators to launch their activities [33,34]. Such methods may suffer from the following problems: (i) the collaboration pattern, including the host organization, participants and their behaviors, must be predefined, which might lead to a great difficulty in reconfiguring collaboration architecture to fit an ever-changing operation environment; (ii) all collaborations are authorized by a host organization while other collaborators are passive to follow a certain arrangement and cannot carry out their strategies and operational rules to achieve their own purposes; and (iii) there is a lack of a flexible visibility control mechanism that allows to share the common knowledge of the collaborative processes among collaborators while at the same time hide the diversity and complexity of their internal behaviors.

Focusing on the above issues, we propose a flexible visibility control mechanism and a cross-organizational process collaboration framework for joint operations of C4ISR systems. This framework enables a peer-to-peer process collaboration that allows a collaboration initiator to determine the participants and their activities autonomously, sign collaboration contracts with the participants individually, and model an evolvable collaboration architecture and finally generate the collaborative processes automatically.

The remaining sections are organized as follows. Section 2 proposes a visibility control mechanism, providing the key concepts of visibility control and collaboration signature. Section 3 focuses on the process collaboration framework by discussing the main concepts and the principle of building a collaborative process. Section 4 presents a formal representation and automatic generation of the collaborative processes. Section 5 summarizes the main contributions and outlines the future work.

2. Visibility control mechanism

To support the relative workflow process, a flexible visibility control mechanism is necessary. This section will introduce the core concepts about visibility control, which provides a underlying mechanism for modeling and generating collaborative processes.

Collaborative organizations are treated as the autonomous systems that run their independent processes to execute sequences of activities. If an organization intends to collaborate with others, it needs to know some behavioral details of other organizations who, if willing to join in the collaboration, have to make some parts of their processes visible to the collaboration initiator. The visibility control provides a mechanism that turns the invisible parts to be visible ones and thereby transforms the private processes into visible ones. A number of definitions are given below to form the basic concepts of the mechanism.

Definition 1 (Private process, PP) A private process is a standalone process which is owned by a system and is visible to this system. It can be modeled as a directed graph $pp ::= \langle A, R \rangle$, where the node set $A ::= \{ \langle Activity \rangle \}$ represents a collection of activities and the edge set $R \subseteq A \times A$ represents a set of transitions that define the execution order of activities. A collection of private processes owned by a system S can be denoted as $\{pp^1, pp^2, \dots, pp^n\}$, and the i th process pp is denoted as $S.pp^i$.

Definition 2 (Visible process, VP) A visible process is a view of a private process of a system from the perspective of another system. It can be modeled as a directed graph $vp ::= \langle A', R' \rangle$, where A' denotes a collection of activities visible to the other system and R' is a set of the transitions between the visible activities. The j th visible process vp of the private process of the system S observed by the system S_0 is denoted as $S.vp_{S_0}^j$.

Table 1 defines three visibility attributes of a process for visibility control, covering the main constructs of a process model. Data and control flow have only values "visible" and "invisible", while the visibility of an activity is further divided into "invisible", "traceable" and "contactable". We include so far the visibility of activity in our visibility con-

trol mechanism, leaving other two attributes to be studied in the future.

Table 1 Types of private information and corresponding visibility value

Visibility attribute	Visibility value	Description
Activity	Invisible	An activity of a system is hidden from another system
	Traceable	The execution status of an activity of a system can be traced by another system through an inquiry/reply mechanism
	Contactable	An activity of a system can be inter-operated by an activity of another system through a messaging mechanism
Data	Visible	The data processed by a system can be accessed by another system
	Invisible	The data processed by a system are hidden from another system
Control flow	Visible	The conditions of the activity transitions of a system can be viewed by another system
	Invisible	The transition conditions are hidden from another system

The visibility control is subject to the following rules.

(i) For a private process, it is possible to generate different views of the process with a set of visibility rules.

(ii) It is possible to define a set of the generic rules which, if needed, can be refined into specific rules for a specific process.

(iii) The private processes may be independently transformed into visible processes.

(iv) For a private process, it is possible to change its visibility by modifying the related visibility rules and generating a new visible process.

(v) The visibility control may be conducted at the business level, which implies that the rule maker does not need to care about the technical details of the visibility control mechanism.

As shown in Fig. 1, the visibility control of the system S covers a number of private processes and a set of visibility rules that determine which activities of a private process are visible to or hidden from its collaborators. The generic rule “an activity of data processing is contactable to the command system”, for example, means that any data processing activity of other systems may be interoperated by the command system through sending/receiving messages to/from it. Such generic rules that define initial visibility constraints are published to all systems. Based on the rules, the other systems that intend to join a current collaboration process may refine their visibility constraints to define the detailed visibility of their activities. The views of visible processes will then be generated, subject to both the generic rules and the specified visibility constraints, to build a cross-organizational collaboration.

Fig. 1 Visibility control mechanism

Defining visibility rules plays an important role in visibility control mechanism. Among the existing rule specifications, semantic business vocabulary and business rules (SBVR) [35] is our favorite. With the advantages of well-known standard and representation in structured English, it allows business analysts to define structured and read-

able rules free from the embarrassment of technical background knowledge. The difficulty of applying SBVR is to create a suitable vocabulary for rule specification. We refer to the cross business process (CBP) vocabulary defined by Cyril [36] and extend it by adding specific concepts and relationships to meet the rule definition requirements of joint

C4ISR operation. The visibility rule definition vocabulary is modeled with a class diagram, which gives a graphical representation and helps clarifying the core concepts with

the associations. Those concepts provide a set of standard terms for specifying visibility rules. The main concepts and relations of visibility rules are listed in Fig. 2.

Fig. 2 Main concepts and relations in vocabulary

The rule specifications are conceptualized by the concepts “ProcessElement” and “System” and the association “is {visible/traceable/contactable}”, and those concepts are further broken into sub-concepts in order to specify a specific process.

A simplified case of regional air defense joint operation is illustrated in Fig. 3.

Fig. 3 shows the main processes of the regional air defense joint operation. The operation is cooperatively performed by four organizations (S_1 , S_2 , S_3 and S_4) that serve different functions in the cross-organizational collaboration. Each system in Fig. 3 defines visibility rules from its own perspective. Let us focus on the visibility rules of activities from the perspective of the command center S_2 , since it covers most key concepts of our approach on visibility control and collaboration management including collaboration signature, visible process, allied process, reach-

able process and collaborative process which will be further discussed in following sections.

According to the needs of the regional air defense joint operation, the systems that initiate collaborations may define their visibility requirements of collaboration activities and publish them to relevant systems (the collaboration participants), forming a set of activity visibility rules as shown as follows:

S_1 's activity visibility required by S_2 :

The activity that handles data collecting is contactable to the command system.

S_2 's activity visibility required by S_1 :

The activity that handles effect assessment is traceable to the intelligence system.

S_3 's activity visibility required by S_2 :

The activity that handles operation planning is contactable to the command system.

The activity that checks data collecting is contactable to the command system.

S_3 's activity visibility required by S_4 :

The activity that handles interception operation is traceable to the logistic system.

S_4 's activity visibility required by S_2 :

The activity that handles logistic panning is contactable to the command system.

.....

On the basis of activity visibility rules, the collaboration signature will be specified to formally define a process visibility.

Fig. 3 Process collaboration of the regional air defense joint operation

Definition 3 (Collaboration signature, CS) A CS specifies the visibility of a private process of a collaboration participator from the perspective of the collaboration initiator, including the initiator, the participator, and the activities involved in the collaboration and the directions of the messaging links between processes. The signature specifies a collaboration pattern that requires some activities of the participator's private process to be made visible to the initiator.

The specification is formally defined as: $cs_{S_0}^{S, vp} ::= \{ \langle a_S, vv, a_{S_0}, dr \rangle \}$, where S_0 is the collaboration initiator, S is the collaboration participator, a_S and a_{S_0} specifies respectively the activities of the S 's process and the S_0 's process involved in the collaboration, and $vv \subseteq \{Trackable, Contactable\}$ and $dr \in \{in, out\}$ specifies respectively the visibility and the directions of the messaging links of the activities of a_S from the perspective of S_0 .

The followings give an example of CS specifications where the collaboration is initiated by S_2 and is participated by S_1 , S_3 and S_4 respectively.

$$cs_{S_2}^{S_1, vp_1} = \{ \langle a_1, traceable \rangle \langle a_2, invisible \rangle$$

$$\langle a_3, contactable, b_1, in \rangle \langle a_4, contactable, b_6, out \rangle \}$$

$$cs_{S_2}^{S_3, vp_3} = \{ \langle c_1, contactable, b_5, out \rangle \langle c_2, invisible \rangle$$

$$\langle c_3, contactable, b_6, in \rangle \}$$

$$cs_{S_2}^{S_4, vp_4} = \{ \langle d_1, traceable \rangle \langle d_2, invisible \rangle$$

$$\langle d_3, invisible \rangle \langle d_4, traceable \rangle \}$$

Based on the above specifications, the visible processes can be generated. Fig. 4 shows the models of the visible processes of S_1 and S_3 from the perspective of S_2 , where S_2 publishes its visibility rules to S_1 and S_3 which S_2 intends to collaborate with. To simplify the presentation, all activities are represented by circular symbols.

The visible process $S_1.vp_{S_2}$ in Fig. 4(a) is generated with the CS specification $cs_{S_2}^{S_1, vp_1}$, where the dashed circle represents an invisible activity of the private process and the rectangle box represents a merged unit, a newly produced visible activity where a number of invisible activities are merged into. The figure shows that as the visible process $S_1.vp_{S_2}$ is generated and is added to the collaborative process of S_2 , a_2 is merged into a_3 and is to be removed from the model, leaving only a_3 in the rectangular box, and the transition which is previously directed to a_2 will be re-directed to a_3 and it is also visible to S_2 . Similarly, the model of the visible process $S_3.vp_{S_2}$, shown

in Fig. 4(b), is generated with the specification $cs_{S_2}^{S_3.vp_3}$, where only c_1 and c_3 are visible to S_2 .

Fig. 4 Models of the visible processes of S_1 and S_3 from the perspective of S_2

3. Process collaboration framework

This section introduces several core concepts of process collaboration and gives a genetic procedure for generating and extending the cross-organization collaboration.

3.1 Core concepts

Definition 4 (Allied process, AP) AP defines a cross-organization process which is built cross neighboring systems that have direct messaging links between their private processes, based on the collaboration signatures between them. An AP may consist of one or more private processes belonging to the initiator S_0 and a number of visible processes belonging to the participators. It can be modeled as a directed graph $\langle A, R \rangle$, where

(i) A is a set of the collaborative activities visible to S_0 .

It is union of two parts:

i) $\cup_k S_0.pp^k.a$, the activities of the process of the initiator S_0 ; where $1 \leq k \leq m_0$ and m_0 is the number of the private processes belonging to S_0 .

ii) $\cup_{i,j} S_i.vp_{S_0}^j.a$, the activities of the visible processes of participators, all visible to S_0 ; where $1 \leq i \leq n$, n is the number of possible participators, and $1 \leq j \leq m_i$, m_i is the number of the visible processes of the i th participator.

(ii) R is a set of transitions or messaging links between the activities. It is union of four parts (i, j, k are the same as those in A):

i) $\cup_k S_0.pp^k.r$, the transitions between the internal activities of the private processes of S_0 .

ii) $\cup_{i,j} S_i.vp_{S_0}^j.r$, the transitions between the internal activities of the visible processes of participators.

iii) $\cup_{i,j} (S_0.pp^i.a \times S_0.pp^j.a)$, $i \neq j$, the messaging links between the boundary activities of the private processes of S_0 provided that there are more than one private processes involved in the collaboration.

iv) $\cup_{i,j,k} (S_0.pp^k.a \times S_i.vp_{S_0}^j.a \cup S_i.vp_{S_0}^j.a \times S_0.pp^k.a)$, the messaging links between the boundary activities of the private processes of S_0 and the boundary activities of the visible processes of participators, and they are bidirectional.

The above definition implies that in an AP specification, all processes of participators that are involved in the collaboration should be visible processes and should have messaging links with the private processes of S_0 . We call the systems that have such processes the neighboring systems of S_0 .

For example, both S_1 and S_3 are neighboring systems of S_2 since there are the messaging links r_{ab_1} (between $S_1.vp_{S_2}$ and $S_2.pp$) and r_{bc_1} (between $S_2.pp$ and $S_3.vp_{S_2}$), and therefore we can define the AP specification $ap_{S_2}^{S_1,S_3}$ from which the collaboration model can be built as shown in Fig 5.

Fig. 5 AP from the perspective of S_2

However, if we try to add a new system which is then not a neighboring system of S_0 , the AP specification is not available. Therefore, the collaboration management needs to be extended to include a non-neighboring system in the specification.

Definition 5 (Reachable process, RP) A visible process of a non-neighboring system of S_0 can be defined as a reachable process rp of S_0 if it has at least one messaging link with a participator of the allied process of S_0 and the relevant activities of the process involved in the collaboration are visible to S_0 .

To give an example, assume that the command system S_2 temporarily loses contact with the logistic system S_4 in the moment when S_2 issues operation orders to both S_3 and S_4 , and then S_4 fails to receive the orders. Since the attacking system S_3 keeps contact with S_4 , it might serve as a bridge between S_4 and S_2 so that the process of S_4 can be made reachable to S_2 . The model is built as shown in Fig. 6.

Before the process pp_4 of the non-neighboring system S_4 is added to the AP of S_2 , the visible process $S_4.vp_{S_2}$ has been described by the CS specification $cs_{S_2}^{S_4.vp_4}$. To define a reachable process, we can specify a messaging link r_{cd_1} between c_1 and d_1 , meaning that the attacking system S_3 has established a collaboration relationship with the logistic system S_4 and can pass the order of S_2 to S_4 . As c_1 and d_1 are visible to S_2 (c_1 is contactable according to $cs_{S_2}^{S_3.vp_3}$ and d_1 is traceable according to $cs_{S_2}^{S_4.vp_4}$), the visible process $S_4.vp_{S_2}$ becomes a reachable process of S_2 .

Fig. 6 Determining the reachable process

Based on the AP specifications, an allied process can be extended, and all reachable processes of non-neighboring systems can be added into a collaborative process.

Definition 6 (Collaborative process, CP) CP specifies a cross-organization process where there are a number of constituent processes that can collaborate with one another directly (they are all constituent processes of an allied process) or indirectly (if some of them are RPs). A collaborative process may consist of a private process of S_0 (the initiator) and a number of visible processes of neighboring or non-neighboring systems of S_0 (the participators). It is formally defined as a directed graph $\langle A, R \rangle$ as follows, where

(i) A is a set of activities visible to S_0 . It is union of two parts:

- i) $ap.a$, all activities of the allied process of S_0 .
- ii) $\cup_{i,j} S_i.rp_{S_0}^j.a$, all activities of the reachable processes of S_0 ; where $0 \leq i \leq n$ and n is the number of the non-neighboring systems of S_0 , $0 \leq j \leq m_i$ and m_i is the number of the RP of S_i .

(ii) R is a set of transitions or messaging links between activities. It is union of three parts:

- i) $ap.r$, the transitions between the activities of the processes of the AP.
- ii) $\cup_{i,j} S_i.rp_{S_0}^j.r$, the transitions between the activities of the RP.
- iii) $\cup_{i,j,k,m} (S_i.ap_{S_0}^j.a \times S_k.rp_{S_0}^m.a)$, the messaging

links between the boundary activities of the processes of the AP and the boundary activities of the RP.

Continuing the case study of the regional air defense joint operation, we add the visible process $S_4.vp_{S_2}$ to the AP $ap_{S_2}^{S_1,S_3}$ and thereby build the model of the collaborative process of S_0 . The model is shown in Fig. 7.

Fig. 7 View of the CP from the perspective of S_2

3.2 Collaboration construction

The collaboration framework proposed in this paper allows APs to combine the RPs of either existing systems or newly added systems to build CPs, in a recursive way, to form a large-scaled collaboration, which ensures the dynamic scalability as the structure of collaboration changes or a new system is added. The scheme is described by the dynamic extension pattern as shown in Fig. 8.

The pattern provides a genetic procedure by which a cross-organizational collaboration is step-by-step specified, dynamically built and recursively extended.

The first step is to define visibility rules. One of the organizations, S_2 for an instance, intends to initiate a collaboration with others. It needs to define the visibility rules that reflect the visibility requirements of the private processes of other systems, using the template mentioned in Section 4, and then publishes the rules to other systems. The other organizations, if hope to initiate collaborations, will do the similar things.

The second step is to specify collaboration signatures. Based on the rules, the collaboration initiator should specify collaboration signatures that define the views of the private processes of other organizations from the perspective of the initiator, for example, $cs_{S_2}^{S_1.vp_1}$, $cs_{S_2}^{S_3.vp_3}$ and $cs_{S_2}^{S_4.vp_4}$. The CS specifications mean that the collaboration to be initiated will possibly cross the three visible processes, vp_1 of S_1 , vp_3 of S_3 and vp_4 of S_4 , and therefore the view of these processes needs to be generated. The other organizations, if hope to initiate collaborations, will do the similar things.

Fig. 8 Dynamic extension pattern of collaborative processes

The third step is to build APs for the collaboration between neighboring systems. The initiator S_2 then combines the visible processes vp_1 , vp_3 and vp_4 with its private process pp_2 to construct an AP ap_2 . So far, an initial collaboration has been established. The other organizations, if hope to initiate collaborations, will do the similar things.

The final step is to extend the existing collaborations. The initiator may modify the architecture of an existing collaboration to adapt to the change of the operational environment. As mentioned in the case of the regional air defense joint operation, the communication between the command system and the logistic system is temporarily disconnected, and thus an RP rp_4 needs to be created subject to $cs_{S_2}^{S_4 \cdot vp_4}$. In the new collaboration architecture, vp_4 is removed from the AP ap_2 , and rp_4 is added to the collaboration and is combined with ap_2 to form the CP cp_2 . Thus, the cross-organizational collaboration model is finally built. The collaboration may evolve as more and more processes are recursively added.

In the course of collaboration evolution, it might involve a complex cognition process of acquiring and integrating human knowledge to define the visibility rules and find who to collaborate with, since there might be many com-

plicated factors to be taken into consideration [2,3]. However, we leave those perplexities to the decision makers of organizations to resolve while offer a flexible and scalable collaboration framework that does not exclude human intervention and that handles the underlying facilities such as process visibility control, dynamic process joining and collaboration architecture reconfiguration.

4. Formal description and CP generation

The above concepts need to be formalized for computation. This section will introduce matrices representation of processes and their boundaries, which provides a mathematical foundation for transforming the private processes into visible ones and then generating the CPs. A brief discussion on implementing a supporting tool will be also included.

4.1 Matrix representation of the processes

To facilitate a simple calculation for process transformation and generation, we use matrices to represent processes and their boundaries.

Definition 7 (Process matrix, M) Process matrix represents a process which is composed of activities and the

transitions or messaging links between the activities. A process p of a system S consists of n activities and is modeled as an $n \times n$ matrix $M^{S,p} = [m_{ij}]$,

$$m_{ij} = \begin{cases} 1, & \text{if there is a transition or a messaging link} \\ & \text{from } a_i \text{ to } a_j (i \neq j) \\ 0, & \text{otherwise} \end{cases}.$$

Any process, either private, visible, allied or collaborative, can be represented by the matrix from which a model or instance of the process can be generated. For example, $M_{S_0}^{S_1.vp}$ denotes a visible process of S_1 from the perspective of S_0 , and $M_{S_0}^{ap}$ denotes an AP of S_0 .

Definition 8 (Boundary matrix, B) Boundary matrix describes one side of the boundary between two processes of different systems, including the activities of the processes and the messaging links between the processes. The boundary between p_1 and p_2 on p_1 side can be modeled as an $m \times n$ matrix $B^{p_1/p_2} = [b_{ij}]$,

$$b_{ij} = \begin{cases} 1, & \text{if there is messaging link from } p_1.a_i \text{ to } p_2.a_j \\ 0, & \text{otherwise} \end{cases}$$

where m and n represent numbers of activities of the processes p_1 and p_2 respectively. The boundary on p_2 side is denoted as B^{p_2/p_1} . [0] stands for an empty boundary matrix, meaning that there is no boundary (or no any messaging link) between the processes.

Although both B^{p_1/p_2} and B^{p_2/p_1} describe the same boundary, they are usually different, because the message links are directed. Therefore, the boundary between p_1 and p_2 should be completely defined by both B^{p_1/p_2} and B^{p_2/p_1} .

Boundary matrix is used for generating a cross-organization process. To generate an AP, the boundary relationships between a private process of the initiator S_0 and the visible processes of the neighbors of S_0 , denoted as $B_{S_0}^{pp/S_{\text{neighboring}}.vp}$, need to be generated. To generate a CP cross the non-neighboring systems of S_0 , the boundary relationships between the visible processes of the neighbors and the RP of the non-neighbors, denoted as $B_{S_0}^{S_{\text{neighboring}}.vp/S_{\text{non-neighboring}}.rp}$, need to be generated, since S_0 has to know which process of its neighbors has a messaging link with the RP of the non-neighbor. The principle of generating a CP will be discussed in next section.

4.2 CP generation algorithm

4.2.1 Generating the visible process matrix

Since the initiator needs to know all the activities that involve in the collaboration, the views of the private processes should be generated first. It is not necessary to, and sometimes the collaborators do not want to, open all activities of their private processes, so that some activities need to be removed from the visible processes.

To generate a tailored visible process is to shrink the matrix of the private process of the participator by merging an invisible activity a_j into a visible one a_i and letting a_i inherit all transitions to or from a_j . The transformation may be conducted with the following operations:

(i) to inherit all transitions, add respectively the j th row and the j th column of the process matrix to the i th row and the i th column while keeping the diagonal elements unchanged, which is defined as two functions: `add_row` (M_S^p, j, i) and `add_column` (M_S^p, j, i);

(ii) to keep the matrix integrity, set the element $m_{ij} = 1$ if $m_{ij} > 1$;

(iii) `shrink_process_matrix` (M_S^p, i): removes the activity a_i (i.e., delete the i th row and the i th column) from the process matrix and returns the shrunk matrix.

The algorithm given below is a function which is executed by a participator which is requested to join in the collaboration. To simplify the algorithm, three basic functions are further defined:

i) `get_the_pp` (participator): build and return the private process matrix of participator from the context of the participator's execution.

ii) `get_merge_set` ($cs_{\text{initiator}}^{\text{participator.vp}}$): find a set of activity number pairs that declare an invisible activity to be merged into a visible activity based on the signature $cs_{\text{initiator}}^{\text{participator.vp}}$ from the context of participator execution, and return the number pairs (i, j) of the activities to be merged, where the activity a_i is to be merged into a_j .

Algorithm `generate_vp`

Input

$cs_{\text{initiator}}^{\text{participator.vp}}$ //the collaboration signature between the initiator and the participator

Output

$M_{\text{initiator}}^{\text{participator.vp}}$ //the process matrix of the visible process of the participator

Begin

$M^{\text{participator.pp}} = \text{get_the_pp}(\text{participator});$

$\text{mergeSet} = \text{get_merge_set}(\text{participator});$

For each $(i, j) \in \text{mergeSet}$

$\text{add_row}(M_{\text{initiator}}^{\text{participator.vp}}, i, j);$

$\text{add_column}(M^{\text{participator.pp}}, i, j);$

$\text{shrink_process_matrix}(M^{\text{participator.pp}}, i)$

end for;

For each $m_{ij} \in M_{\text{initiator}}^{\text{participator.vp}}$

If $m_{ij} > 1$ **then** $m_{ij} = 1$ **end if**

end for

Return $M^{\text{participator.pp}}$

end

To give an example of constructing the matrix of vp_4 from pp_4 on the previous case (see Fig. 6 for the process

model), the transformation, shown in Fig.9, begins with the initial matrix of pp_4 , the activity d_2 is merged into d_1 by transforming the matrix into the second and then the third, d_3 is then merged into d_4 by transforming the matrix into the fourth and then the fifth, and thus the matrix of vp_4 is finally generated.

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

Fig. 9 Example of generating a visible process matrix

4.2.2 Generating the collaborative boundary matrix

Before generating the CP, the initiator needs to know the boundary of the intended cross-organization collaboration.

There are two kinds of cross-organizational boundaries to be generated: (i) the boundaries between the initiator's process and the processes of neighboring systems, named as allied boundaries, for building the APs; (ii) the boundaries between the processes of neighboring systems and the processes of non-neighboring systems, named as collaborative boundary matrices, for building the CP. The allied boundary matrices can be directly generated, while the

collaborative boundary matrices are more complicated to build and hence the following algorithm is provided.

The algorithm given below is a function which is executed by a neighboring system of the initiator, and four more atomic functions are defined:

(i) **build_allied_boundary** ($cs_{S_0}^{S_1.vp}$, $M_{S_0}^{pp}$, $M_{S_1}^{vp}$): build and return the matrix of the allied boundary between the process of the initiator system S_0 and the process of the participator system S_1 from the perspectives of S_0 by referencing the signature $cs_{S_0}^{S_1.vp}$;

(ii) **get_cs** (S_1 , $cs_{S_0}^{S_2.vp}$): find and return the existing signature between an initiator S_1 and S_2 (S_1 may play a role of a bridging neighbor to collaborate with S_2) from the context of the S_1 's execution, and the returned signature should include S_2 's activities declared in $cs_{S_0}^{S_2.vp}$;

(iii) **find_invisible** ($B_{S_{neighboring}}^{S_{neighboring}.pp/S_{non-neighboring}.pp}$, $cs_{initiator}^{S_{non-neighboring}.vp}$, $cs_{initiator}^{S_{neighboring}.vp}$): find invisible activities of either the non-neighboring process or the neighboring process by referencing the signatures $cs_{initiator}^{S_{non-neighboring}.vp}$ and $cs_{initiator}^{S_{neighboring}.vp}$ in the built boundary matrix $B_{S_{neighboring}}^{S_{neighboring}.pp/S_{non-neighboring}.pp}$, and return the set of the activities invisible to the initiator;

(iv) **shrink_boundary_matrix** ($B_{S_1}^{S_1.pp/S_2.pp}$, i , j): remove the invisible activity a_i ($i \neq 0, j = 0$) of $S_1.pp$ or a_j ($i = 0, j \neq 0$) of $S_2.pp$ from the boundary matrix and return the shrunk matrix.

Algorithm generate_boundary

Input

$cs_{initiator}^{S_{non-neighboring}.vp}$ //the collaboration signature between the initiator and the non-neighboring system

$cs_{initiator}^{S_{neighboring}.vp}$ //the collaboration signature between the initiator and the neighboring system

Output

$B_{initiator}^{S_{neighboring}.vp/S_{non-neighboring}.vp}$, $B_{S_{neighboring}}^{S_{non-neighboring}.pp/S_{neighboring}.vp}$ //the matrixes of both sides of the collaborative boundary

Begin

$M_{S_{neighboring}}^{S_{neighboring}.pp} = \text{get_the_pp}(S_{neighboring}.pp)$; $cs_{S_{neighboring}}^{S_{non-neighboring}.vp} = \text{get_cs}(S_{neighboring}, cs_{initiator}^{S_{non-neighboring}.vp})$;

$M_{S_{neighboring}}^{S_{non-neighboring}.vp} = \text{generate_vp}(cs_{S_{neighboring}}^{S_{non-neighboring}.vp})$;

//to build the boundary matrix of one side

$B_{S_{neighboring}}^{S_{neighboring}.pp/S_{non-neighboring}.vp} = \text{build_allied_boundary}(cs_{S_{neighboring}}^{S_{non-neighboring}.vp}, M_{S_{neighboring}}^{S_{neighboring}.pp}, M_{S_{neighboring}}^{S_{non-neighboring}.vp})$;

$invisibleActSet = \text{find_invisible}(B_{S_{neighboring}}^{S_{neighboring}.pp/S_{non-neighboring}.vp}, cs_{initiator}^{S_{non-neighboring}.vp}, cs_{initiator}^{S_{neighboring}.vp})$;

For each $b_{ij} \in B_{S_{neighboring}}^{S_{neighboring}.pp/S_{non-neighboring}.vp}$

If $a_i \in invisibleActSet$ **then** $j=0$; **shrink_boundary_matrix** ($B_{S_{neighboring}}^{S_{neighboring}.pp/S_{non-neighboring}.vp}$, i , j) **end if**;

If $a_j \in invisibleActSet$ **then** $i = 0$; **shrink_boundary_matrix** ($B_{S_{neighboring}}^{S_{neighboring}.pp/S_{non-neighboring}.vp}$, i , j) **end if**;

end for;

//to build the boundary matrix of the other side

$B_{S_{neighboring}}^{S_{non-neighboring}.pp/S_{neighboring}.vp} = \text{build_allied_boundary}(cs_{S_{neighboring}}^{S_{non-neighboring}.vp}, M_{S_{non-neighboring}}^{S_{non-neighboring}.vp}, M_{S_{neighboring}}^{S_{neighboring}.pp})$;

$invisibleActSet = \text{find_invisible}(B_{S_{neighboring}}^{S_{non-neighboring}.pp/S_{neighboring}.vp}, cs_{initiator}^{S_{neighboring}.vp}, cs_{initiator}^{S_{non-neighboring}.vp})$;

For each $b_{ij} \in B_{S_{\text{neighboring}}}^{S_{\text{non-neighboring}} \cdot pp / S_{\text{neighboring}} \cdot pp}$
If $a_i \in \text{invisibleActSe}$ **then** $j = 0$; **shrink_boundary_matrix** ($B_{S_{\text{neighboring}}}^{S_{\text{non-neighboring}} \cdot pp / S_{\text{neighboring}} \cdot vp}, i, j$) **end if**;
If $a_j \in \text{invisibleActSet}$ **then** $i = 0$; **shrink_boundary_matrix** ($B_{S_{\text{neighboring}}}^{S_{\text{non-neighboring}} \cdot pp / S_{\text{neighboring}} \cdot vp}, i, j$) **end if**;
end for;
Return ($B_{\text{initiator}}^{S_{\text{neighboring}} \cdot vp / S_{\text{non-neighboring}} \cdot vp}, B_{S_{\text{neighboring}}}^{S_{\text{non-neighboring}} \cdot pp / S_{\text{neighboring}} \cdot vp}$)
end begin

To give an example of generating the boundary matrix $B_{S_2}^{S_3 \cdot vp / S_4 \cdot vp}$ on the previous case (see Fig. 6), the matrix $B_{S_3}^{S_3 \cdot pp / S_4 \cdot vp}$ of the allied boundary between $S_3 \cdot pp$ and $S_4 \cdot vp$ is first built, the invisible activities c_2 , d_2 and d_3 are then removed and thus one side matrix of the collaborative boundary is finally generated as shown in Fig. 10(a), and so is the other side matrix $B_{S_2}^{S_4 \cdot vp / S_3 \cdot vp}$ shown in Fig. 10(b).

$$\begin{matrix} c_1 & d_1 & d_2 & d_3 & d_4 \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} & \Rightarrow & \begin{matrix} c_1 & d_1 & d_2 & d_3 & d_4 \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{matrix} & \Rightarrow & \begin{matrix} c_1 & d_1 & d_4 \\ \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \end{matrix} \end{matrix}$$

(a) One side matrix of the collaborative boundary

$$\begin{matrix} c_1 & c_2 & c_3 \\ \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} & \Rightarrow & \begin{matrix} c_1 & c_2 & c_3 \\ \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \end{matrix} & \Rightarrow & \begin{matrix} c_1 & c_3 \\ \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \end{matrix} \end{matrix}$$

(b) The other side matrix of the collaborative boundary

Fig. 10 Example of generating a pair of boundary matrices

$$\begin{aligned} (M_{S_0}^{p_0}) &\Rightarrow \begin{pmatrix} M_{S_0}^{p_0} & B_{S_0}^{S_0 \cdot p_0 / S_1 \cdot vp_1} \\ B_{S_0}^{S_1 \cdot vp_1 / S_0 \cdot p_0} & M_{S_0}^{S_1 \cdot vp_1} \end{pmatrix} \Rightarrow \begin{pmatrix} M_{S_0}^{p_0} & B_{S_0}^{S_0 \cdot p_0 / S_1 \cdot vp_1} & B_{S_0}^{S_0 \cdot p_0 / S_2 \cdot vp_2} \\ B_{S_0}^{S_1 \cdot vp_1 / S_0 \cdot p_0} & M_{S_0}^{S_1 \cdot vp_1} & B_{S_0}^{S_1 \cdot vp_1 / S_2 \cdot vp_2} \\ B_{S_0}^{S_2 \cdot vp_2 / S_0 \cdot p_0} & B_{S_0}^{S_2 \cdot vp_2 / S_1 \cdot vp_1} & M_{S_0}^{S_2 \cdot vp_2} \end{pmatrix} \\ &\dots \Rightarrow \begin{pmatrix} M_{S_0}^{p_0} & B_{S_0}^{S_0 \cdot p_0 / S_1 \cdot vp_1} & B_{S_0}^{S_0 \cdot p_0 / S_2 \cdot vp_2} & \dots & B_{S_0}^{S_0 \cdot p_0 / S_n \cdot vp_n} \\ B_{S_0}^{S_1 \cdot vp_1 / S_0 \cdot p_0} & M_{S_0}^{S_1 \cdot vp_1} & B_{S_0}^{S_1 \cdot vp_1 / S_2 \cdot vp_2} & \dots & B_{S_0}^{S_1 \cdot vp_1 / S_n \cdot vp_n} \\ B_{S_0}^{S_2 \cdot vp_2 / S_0 \cdot p_0} & B_{S_0}^{S_2 \cdot vp_2 / S_1 \cdot vp_1} & M_{S_0}^{S_2 \cdot vp_2} & \dots & B_{S_0}^{S_2 \cdot vp_2 / S_n \cdot vp_n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ B_{S_0}^{S_n \cdot vp_n / S_0 \cdot p_0} & B_{S_0}^{S_n \cdot vp_n / S_1 \cdot vp_1} & B_{S_0}^{S_n \cdot vp_n / S_2 \cdot vp_2} & \dots & M_{S_0}^{S_n \cdot vp_n} \end{pmatrix} \end{aligned}$$

Fig. 11 Matrix-based map of collaborative process evolution

The map shows there is only a single process p_0 of the initiator S_0 at the beginning. For the second process matrix, S_0 initiates a collaboration with a neighboring system S_1 to form an AP which is composed of the private process $M_{S_0}^{p_0}$ of S_0 , the visible process $M_{S_0}^{S_1 \cdot vp_1}$ of S_1 and the boundary $B_{S_0}^{S_0 \cdot p_0 / S_1 \cdot vp_1}$ between the two processes; then, for the third matrix, S_0 extends the collaboration by including the process p_2 of S_2 to form a collaborative process that may comprise one more visible process (or a reachable process) $M_{S_0}^{S_2 \cdot vp_2}$ and two more boundaries (note: S_2 may not be a neighbor of S_0 and in that case $B_{S_0}^{S_0 \cdot p_0 / S_2 \cdot vp_2}$

There might be a worry about missing the messaging link from d_3 to c_2 in the above transformation. However, it is normal from the perspective of S_2 , for S_2 does not care about the “internal collaboration” between S_3 and S_4 .

4.2.3 Generating the collaborative process matrix

Generating a collaborative process is a process of recursive extension. To build the collaborative process matrix, each reachable process joining in the collaboration would extend the process matrix by adding a set of new rows and columns where the newly added process matrix and the pair of boundary matrixes perfectly fill the empty spaces. Moreover, the new process matrix fits to describing the internal structure of the new process, and the boundary matrices fit to describing the messaging links between the new process and the other constituent processes of the collaborative process. Fig. 11 provides a matrix-based map of collaborative process evolution, where a single process evolves into the collaborative process.

should be an empty matrix [0]); and for the last matrix, the collaborative process would be extended recursively until no more reachable process may be added to the collaboration.

The algorithm given below is a function which is executed by the initiator, and two more atomic functions are defined:

(i) $\text{get_all_cs}(M_{\text{initiator}}^p)$: find and return a set of signatures which are possibly used for building the existing process $M_{\text{initiator}}^p$ from the context of the current initiator’s execution; if the process $M_{\text{initiator}}^p$ is neither an allied nor

a collaborative one, return an empty set [];

(ii) $\text{combine_proc}(M_{\text{initiator}}^p, M_{\text{initiator}}^{S, vp}, \text{boundarySet})$: combine the matrix of an existing process $M_{\text{initiator}}^p$ and the matrix of a newly added process $M_{\text{initiator}}^{S, vp}$ with a set

of the matrices of the boundaries between the new process and the constituent processes of the existing process, using the pattern shown in Fig. 11, and return the matrix of the CP.

Algorithm generate_collaborative_process

Input

$M_{\text{initiator}}^p$; // the matrix of the existing process, either a private process, an AP or CP

$cs_{\text{initiator}}^{\text{participator.vp}}$; // the collaboration signature between the initiator and the participator which is requested to join

Output

$M_{\text{initiator}}^{cp}$; // the matrix of the CP to be generated, an empty matrix meaning that it cannot be generated

Begin

$M_{\text{initiator}}^{\text{participator.vp}} = \text{generate_vp}(cs_{\text{initiator}}^{\text{participator.vp}})$; // generate the matrix of the visible process of the participator

$M_{\text{initiator}}^{pp} = \text{get_the_pp}(\text{initiator})$; // get the matrix of the initiator's process

// try to build an AP between the initiator and the participator

$(B_{\text{initiator}}^{\text{initiator.pp/participator.vp}}, B_{\text{initiator}}^{\text{participator.vp/initiator.pp}}) = \text{build_allied_boundary}(cs_{\text{initiator}}^{\text{participator.vp}}, M_{\text{initiator}}^{pp}, M_{\text{participator.vp}}^{\text{participator.vp}})$;

$\text{successful} = 0$; // a flag that denotes the successfulness of generating the CP

$\text{boundarySet} = []$; // a set of the boundary matrices used for generating the CP

If $B_{\text{initiator}}^{\text{initiator.pp/participator.vp}} \neq [0]$ **or** $B_{\text{initiator}}^{\text{participator.vp/initiator.pp}} \neq [0]$ **then**

$\text{boundarySet} = \text{boundarySet} \cup B_{\text{initiator}}^{\text{initiator.pp/participator.vp}} \cup B_{\text{initiator}}^{\text{participator.vp/initiator.pp}}$;

$\text{successful} = 1$; // the AP will be generated, as the participator is a neighboring system of the initiator

else

// try to ask one of the collaborators of the initiator, a neighbor of the participator, to bridge the gap

$\text{constituentSet} = \text{get_all_cs}(M_{\text{initiator}}^p)$;

If $\text{constituentSet} \neq []$ **then**

For each $cs_{\text{initiator}}^{S_1, vp} \in \text{constituentSet}$

$(B_{\text{initiator}}^{S_1/\text{participator}}, B_{\text{initiator}}^{\text{participator}/S_1}) = \text{generate_boundary}(cs_{\text{initiator}}^{\text{participator.vp}}, cs_{\text{initiator}}^{S_1, vp})$;

If $B_{\text{initiator}}^{S_1/\text{participator}} \neq [0]$ **or** $B_{\text{initiator}}^{\text{participator}/S_1} \neq [0]$ **then**

$\text{successful} = 1$; // since a bridging neighbor has been found, the CP will be generated

$\text{boundarySet} = \text{boundarySet} \cup B_{\text{initiator}}^{S_1/\text{participator}} \cup B_{\text{initiator}}^{\text{participator}/S_1}$;

End if

End for

End if

End if;

If $\text{successful} = 1$ **then**

$M_{\text{initiator}}^{cp} = \text{combine_proc}(M_{\text{initiator}}^p, M_{\text{initiator}}^{\text{participator.vp}}, \text{boundarySet})$; // the collaborative process is successfully generated

else $M_{\text{initiator}}^{cp} = [0]$; // the collaborative process fails to be generated

End if;

Return $M_{\text{initiator}}^{cp}$;

End begin

4.3 Prototype implementation

The algorithms have been realized and integrated into our prototype OBCREAT 2.0 (ontology-based capability requirements elicitation and analysis tool) [37,38]. The core

part of the prototype is the process steering component that interacts with the workflow engine and collaborators for modeling and tracking CP. The architecture of the component is shown in Fig. 12, comprising six sub-components that function as follows.

Fig. 12 Architecture of the process steering component

(i) The *Visible Process Searcher* searches for a visible process of either neighboring or non-neighboring collaborators to get a view of it for modeling a collaborative process.

(ii) The *Collaborative Proc. Generator* transforms private processes into visible ones and generates the allied/collaborative processes by applying the generation algorithms. It responds to the invocation of the *Visible Process Searcher* by returning a set of the activities of a visible process.

(iii) The *Activity Status Monitor* is used for monitoring the execution status of an activity of a process. It may invoke the *Process Manager* to get an instance of the process and then invoke the *Activity Manager* to get an instance of the activity and finally invoke the *Status Manager* to get the execution status of the activity.

(iv) The *Process Manager* is used for getting or saving a specified process when modeling and generating a CP, and it may also access the database of the workflow engine to get an instance of a running process for monitoring the execution status of an activity.

(v) The *Activity Manger* is used for getting or saving the composition of a CP, and it may also help the *Process Manager* to locate the activity and get an instance of it.

(vi) The *Status Manager* is used for getting the execution status of an activity of a running process by accessing the database of the workflow engine.

We have made a qualitative comparison in Table 2 against the proposed framework, on those important features such as CBP construction manner, visibility control ability, perspective of CBP, and application field.

Table 2 Comparisons between the proposed framework and other framework

Framework	CBP construction	Process elements in visibility rules	Language for visibility rules	Perspective of CBP	Means of supporting cross-org collaboration	Application
CrossFlow	-	-	-	Public-view based on bi-directional matchmaking	CrossFlow platform	Commerce
CIM	-	-	-	Public-view	Public workflow	Military
RWF	Bottom-up	Task/activity	-	Org-oriented	Visible messaging link	Holistic
COIN	Bottom-up	Activity, data, controlflow	SBVR	Public-view	Coalition workflow	Holistic
The proposed framework	Bottom-up	Activity, data, controlflow	SBVR	Org-oriented	Messaging link and reachable process	Military

The proposed framework follows a bottom-up assembling fashion and, therefore, each individual organization can choose their partners and assemble the visible processes with its own process into final CP. Process elements contained in visibility rules largely determine the description ability of visibility rules, i.e. the more elements, the more fine description granularity. There are several description languages for business rules (e.g., RuleML, semantic web rule language (SWRL), resource description framework (RDF)). However, only SBVR offers the advantages to be a standard. In this paper we construct the vocabulary of SBVR to satisfy the rule definition requirement for joint operations in C4ISR. Org-oriented CBP perspective allows organizations to build their own collaborative process from their own view without being controlled by a third-party or a hosted organization. Different from the traditional method, the cross-org collaboration in the proposed framework is launched via messaging link and RP, which could assemble the processes of both neighboring and non-neighboring collaborators.

However, the proposed framework may bring some tradeoffs, although they may be outweighed by many advantages offered by our methodology.

(i) In our framework, different organizations deploy different collaborative processes for the same collaboration, which may result in the inconsistency between these CPs. This asks much higher requirement for process storage and coordination. In the prototype mentioned above, components such as the *Collaborative Proc. DB*, the *Private Proc. DB* are designed for business-process-definition storage. Components, such as the *Visible Proc. Searcher*, *workflow coordinator* are dedicatedly designed for process coordination.

(ii) We employ matrices to represent collaborative modeling framework. Yet, the matrices may contain quite a few zeros, which affect the space efficiency of data storage. This can be addressed by adopting some matrix compression techniques in our future work.

5. Conclusions and future work

This paper addresses the issues of cross-organizational process collaboration including process visibility control, process combination and adaptability for collaboration evolution. The main contributions are summarized as follows.

A flexible visibility control of a collaborative process and its constituent activities is proposed. It allows collaborators to define a set of visibility rules to generate different views of the private processes for different collaborations, which gives a great flexibility for the collaboration initiator

to decide on an appropriate collaboration pattern, including the participators, the activities involved and the messaging links between the activities, to suit a specific purpose. Moreover, the visible control may hide the complex structures of the private processes from the collaborators so that a possible change made to the structures, if it is inevitable, would have a least impact on the established collaboration.

An evolvable collaboration framework is presented. With the framework, the collaborators may define allied processes, reachable processes and neighborhoods of systems, and based on those concepts, build a collaborative process. As the framework supports dynamically and recursively add a new process or even a new organization to an existent collaboration, the adaptability to the change of an operational environment is guaranteed.

A formal representation of the collaborative process and its boundaries is suggested, and based on the representation, a set of generation algorithms are provided. It may help engineers to develop a computer-aided tool to automate the collaborative process modeling and construction based on the above theory.

In the future, we intend to extend the visibility control mechanism by including the data exchanged in the collaboration and the conditions of activity transitions to enrich the intension of the collaboration concepts so that the collaboration could be more specifically described and more precisely controlled. In addition, the mechanism of visibility publication and discovery is worth to be further studied, as it addresses the interesting issue of sharing the tacit knowledge to support the visionary cross-organizational process interoperability at the comprehensive level.

References

- [1] U.S. Government Printing Office. Joint Vision 2020. <http://www.dtic.mil/jointvision/jvpub2.htm>.
- [2] P. Darling, J. Lawlor. Fulfilling the promise: a joint corps for a joint military. *Military Review*, 2012, 92(3): 82–83.
- [3] C. R. Lawton, W. E. Beyeler, P. D. Finley, et al. A complexity science-based framework for global joint operations analysis to support force projection. LDRD Final Report, No. SAND2015-20804. Albuquerque: Sandia National Laboratories, 2015.
- [4] C. R. Allred, S. E. Fawcett, C. Wallin, et al. A dynamic collaboration capability as a source of competitive advantage. *Decision Sciences*, 2011, 42(1): 129–161.
- [5] M. Weske. *Business process management: concepts, languages, architectures*. New York: Springer Science & Business Media, 2012.
- [6] M. Rosemann, J. vom Brocke. The six core elements of business process management. *Handbook on Business Process Management*. Berlin Heidelberg: Springer, 2015.
- [7] P. Harmon. The scope and evolution of business process management. *Handbook on Business Process Management*. Berlin Heidelberg: Springer, 2015.

- [8] L. M. Camarinha-Matos, P. Macedo, F. Ferrada, et al. Collaborative business scenarios in a service-enhanced products ecosystem. *Collaborative Networks in the Internet of Services*. Berlin: Springer, 2012.
- [9] M. C. Branco, J. Troya, K. Czarniecki, et al. *Matching business process workflows across abstraction levels*. Berlin: Springer, 2012.
- [10] Workflow Management Coalition. The workflow reference model. <http://www.theartofbpm.com>, 1995.
- [11] Voluntary Interindustry Commerce Standards, Association. Collaborative planning, forecasting & replenishment (CPFR), version 2.0. <http://www.cpfr.org>, 2006.
- [12] University of Saarland. Modeling standards EPC/Aris, IDS sheer business process excellence. http://www.idsscheer.com/en/ARIS/Modeling_Standards/EPC/79740.html, 2005.
- [13] Voluntary Interindustry Commerce Standards (VICS), Association. Collaborative Planning, Forecasting & Replenishment (CPFR®), version 2.0. <http://www.cpfr.org>, 2006.
- [14] R. Kamalapur, D. Lyth, A. Houshyar. Benefits of cpfr and vmi collaboration strategies: a simulation study. *Journal of Operations & Supply Chain Management*, 2013, 6(2): 59–73.
- [15] P. Grefen, K. Aberer, Y. Hoffner. CrossFlow: cross-organizational workflow management in dynamic virtual enterprises. *Computer Systems Science & Engineering*, 2000, 2(1): 277–290.
- [16] P. W. P. J. Grefen. CrossFlow: cross-organizational workflow management for service outsourcing in dynamic virtual enterprises. *IEEE Data Engineering Bulletin*, 2001, 24(1): 52–57.
- [17] European Commission. ATHENA european integrated project. <http://www.athena-ip.org>, 2005.
- [18] EU-COIN Consortium. COIN enterprise collaboration & interoperability. <http://www.coin-ip.eu>, 2010.
- [19] A. P. Achilleos, G. M. Kapitsaki, G. Sielis, et al. *Enterprise Collaboration and Interoperability (COIN) platform: two case studies in the marine shipping domain*, *Web Information Systems Engineering-WISE*. Berlin Heidelberg: Springer, 2012.
- [20] M. Dumas, M. Rosa. La, J. Mendling. *Fundamentals of business process management*. Berlin: Springer, 2012.
- [21] M. Dumas, M. Rosa La, J. Mendling, et al. *Fundamentals of business process management*. Berlin: Springer, 2013.
- [22] R. Breu, S. Dustdar, J. Eder. Towards living inter-organizational processes. *Proc. of the 15th IEEE Conference on Business Informatics*, 2013: 363–366.
- [23] X. Zhao. On supporting collaborative business processes: an organisation-oriented perspective. *Proc of the 9th International Conference on Advanced Information Systems Engineering*, 2007: 254–267.
- [24] X. Zhao, C. Liu, Y. Yang. Aligning collaborative business processes: an organisation-oriented perspective. *IEEE Trans. on Systems, Man, and Cybernetics—Part A: Systems and Humans*, 2009, 39(6): 1152–1164.
- [25] X. Zhao, C. Liu, Y. Yang. Steering dynamic collaborations between business processes. *IEEE Trans. on Systems, Man and Cybernetics—Part A: Systems and Humans*, 2010, 40(4): 743–757.
- [26] DoD Architecture Framework Working Group, Department of Defense, US. *DoD architecture framework version 2.0*, 2009.
- [27] G. Kingston, S. Fewell, W. Richer. *An organisational interoperability agility model*. Canberra: Defence Science and Technology Organisation, 2005.
- [28] NATO Allied Data Publication. NATO C3 Technical Architecture (NC3TA), Versio 4.0. <http://www.nato.int/docu/standard.htm>, 2000.
- [29] T. Clark, R. Jones. Organizational interoperability maturity model for C2. *Proc of the 4th International Command and Control Research and Technology Symposium*, 1999: 137–143.
- [30] A. Tolk. Beyond technical interoperability-introducing a reference model for measures of merit for coalition interoperability. *International Command & Control Research & Technology Symposium*, 2003, 12(2): 11–35.
- [31] F. Liu, H. Liu, Y. Li, et al. Research on semantic Web based case representation and CBR system architecture. *Proc. of IEEE International Conference on Machine Learning and Cybernetics*, 2004: 2149–2153.
- [32] F. Liu, L. Yao, W. Zhang, et al. A conceptual model of agent mediated Web service. *Proc. IEEE International Conference on Services Computing*, 2004: 638–642.
- [33] W. Hur, H. Bae, S. H. Kang. Customizable workflow monitoring. *Concurrent Engineering*, 2003, 11(4): 313–325.
- [34] D. W. Van, H. K. M. Van. *Workflow management: models, methods, and systems*. Massachusetts: MIT Press, 2004.
- [35] Object Management Group. Semantics of business vocabulary and business rules (SBVR) V1.2. <http://www.omg.org/spec/SBVR/1.2/PDF>, 2013.
- [36] COIN Consortium. D5.4.1b–Business process interoperability services specification M29. <http://www.coin-ip.eu>, 2010.
- [37] Q. C. Dong, Z. X. Wang, G. Y. Chen. Domain-specific modeling and verification for C4ISR capability requirements. *Journal of Central South University*, 2012, 19(1): 1334–1340.
- [38] Q. C. Dong. Research on formal analysis framework of effectiveness concept of C4ISR systems. Nanjing, China: PLA University of Science and Technology, 2013. (in Chinese)

Biographies

Lin Yang was born in 1982. He is an instructor in Control Engineering Department at Naval Aeronautical and Astronautical University. He attended Nanjing Forestry University from 2000 to 2004, majoring in computer science and technology. He received his M.Eng degree in 2007 from Traffic Information Engineering and Control Department at Nanjing University of Aeronautics and Astronautics. His research interests concentrate on the fields of equipment support and information construction.
E-mail: yl_nuaa@126.com

Minggang Yu was born in 1986. He is a lecturer of Institute of Command Information Systems, PLA University of Science and Technology. His research interests are system engineering and the theory of command and control, focusing on system interoperability.
E-mail: yuminggang8989@163.com

Qingchao Dong was born in 1982. He is a lecturer of Naval Aeronautical and Astronautical University. He received his Ph.D. degree from PLA University of Science and Technology. His research interests are software engineering, requirements engineering and formal verification.
E-mail: dongqingchao001@163.com