An Artifact-Centric View-Based Approach to Modeling Inter-organizational Business Processes

Sira Yongchareon[1], Chengfei Liu[1], and Xiaohui Zhao[2]

[1] Faculty of Information and Communication Technologies
Swinburne University of Technology, Victoria, Australia
{syongchareon,cliu}@swin.edu.au
[2] Department of Computing, Faculty of Creative Industries and Business
Unitec Institute of Technology, Mount Albert, New Zealand
xzhao@unitec.ac.nz

Abstract. Over past several years, there have been increasing needs for more efficient approaches to the design and implementation of inter-organizational business processes. The process collaboration spanning organizational boundaries is deemed to keep the organization autonomy, which means each organization owns its freedom of modifying internal operations to meet their private goals while satisfying the mutual objective with its partners. To achieve these, we propose an artifact-centric view-based framework for inter-organizational process modeling consisting of an *artifact-centric collaboration model*, and a *conformance* mechanism between public view and private view to support the participating organization's customization and changes of their internal operations while ensuring the correctness of collaboration process.

1 Introduction

Over past years, service-oriented architecture (SOA) has been serving an increasing demand of IT for businesses to meet the challenges of the ever-changing market [7]. SOA particularly enables the business collaboration across organizations by the means of web service composition to achieve the mutual goal of collaboration as well as the individual goal of each participant. Service choreography is used to specify the interaction between business parties whereby providing the agreed behavioral contract between collaboration participants. Although service choreography technology intends to provide a global view of the collaboration and allow each party to design and implement its own portion, the existing choreography modeling approaches and languages mainly describe the collaboration from the procedural perspective, and focus on control-flow, message sequencing, etc. With this limited focus, the current service choreography approaches cannot well support the three major requirements of the collaboration: *compliance*, *flexibility*, and *autonomy*. The *compliance* refers to the expectation that all parties must provide the services as they have agreed in the contract. The *flexibility* means that each party has the freedom to change and implement its own part in the collaboration while remaining *autonomous*.

In recent years, a new approach for modeling business processes has emerged, namely *artifact-centric process* modeling, providing a higher level of robustness and flexibility for describing process specification than traditional process-centric approaches [1, 2, 3, 4, 12]. In the process coordination, we observed that at a particular state a message exchanged between organizations reflects the attribute changes of the artifacts used in an organization's internal process. Some states of the artifacts of each individual party link with the global constraints in the collaboration. Based on this observation and the three requirements, we propose a new paradigm of inter-organizational process modeling namely *artifact-centric collaboration model* together with the notion of *conformance* between public and private views to tackle the compliance, autonomy, and flexibility issues of business collaboration.

The remainder of this paper is organized as follows. Section 2 introduces an artifact-centric approach for modeling inter-organizational business processes. Section 3 discusses the construction of public and private views. Section 4 reviews the related works. Finally, the conclusion and future work are given in Section 5.

2 Modeling Inter-organizational Business Processes

In this section, we take supply chain collaboration as an example to illustrate and motivate the artifact-centric approach to modeling inter-organizational business collaboration, as shown Fig. 1. We initiate the discussion of this example by identifying the types of involved business artifacts and describing how they are modeled for this business collaboration. Here, we classify artifacts into two types: (1) *local artifact* and (2) *shared artifact*. A *local artifact* is used internally within one organization and can be used for supporting the coordination between intra-organizational processes and inter-organizational processes. A *shared artifact* serves as a contract between involved organizations, and it is used to indicate the agreed business stages towards the completion of the collaborative process. Note that in the implementation, shared artifacts act as messages sent and received between the organizations. In Fig. 1 (a), the Supplier will ship the order only if the state of the *Purchase Order* (*PO*) is in the *confirmed* state and the designated logistic company is arranged by using the *Shipping List* (*SL*) and *Shipping Order* (*SO*) artifacts. We draw the interrelation between artifacts as *synchronization dependency* (dashed-line).

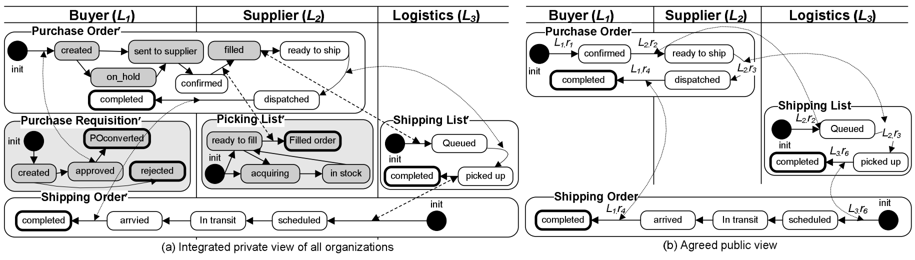

Fig. 1. Overall collaboration and its public view of supply chain business processes

In the view-based approach to modeling inter-organizational processes, e.g., [7, 10, 11], a *public view* has to be constructed at the first stage based on the integration of individual local process of each organization in the collaboration. For our artifact-centric model, only shared artifacts and their necessary (goal) states that are required to be globally visible to other parties appear in the public view. After the public view is built, the involved organizations should also have freedom to change their own parts (i.e., their responsible parts of shared artifacts and their local artifacts) later. As such, it is important to guarantee that such changes in the local process, or *private view*, do not influence the correctness of the overall collaboration. We propose the notion of *view conformance* as the validation mechanism of such changes in the private view w.r.t. the public view. Due to page limitation, we do not detail how public view is constructed in this paper; however, it can be achieved by adapting artifact-centric process abstraction presented in our previous work [4].

Now, we define *Artifact-Centric Collaboration Model* (*ACC model*) extended from the artifact-centric process model (*ACP model*) [3, 4, 16] to capture inter-organizational processes. It consists of four core constructs: *roles*, *artifacts*, *services*, and *business rules*. An *artifact* is either a *local-typed* or a *shared-typed* business entity or an object involved in inter-organizational processes. An artifact class C is a tuple (A, S, S^f) where A is set of a name-value pair attribute of scalar-typed value or nested attribute structure (array-typed), S is a set of states, and $S^f \subseteq S \backslash \{s^{init}\}$ is a set of *final* states where s^{init} denotes *initial* state. A *service* is a task owned by one organization in the collaboration and is used to perform (read/write) operations on artifact(s). A *business rule* is defined by a *condition-action* style, which describes what condition a service is invoked, and what (post) conditions attributes and state of artifacts must satisfy after the invocation. Business rule r is triple (λ, β, v) where λ and β are *pre-condition* and *post-condition*, respectively, containing two types of propositions: state proposition and attribute proposition; and v is a service to be performed. A business rule can be used to synchronize two or more artifacts, called *synchronization dependency*. We denote D for its set, e.g., in Fig. 1, $D(PO, SO) = \{L_1.r_4\}$ and $D(PO, SL) = \{L_2.r_2, L_2.r_3\}$. Note that the synchronization dependency is transitive.

Definition 1: (Artifact-Centric Collaboration Model or ACC model). Let Π denote an *ACC model*, and it is tuple (Z, V, R, L, ZL, VL) where,

- Z is a set of artifact classes, V is a set of services, and R is a set of business rules
- L is a set of organization roles participating in the collaboration
- $ZL \subseteq Z \times 2^L$ is a set of *artifact-role* relations between artifacts and their corresponding set of organization roles. Relation $(C_i, \{l_1, l_2, \ldots, l_x\}) \in ZL$ means that organization roles $l_1, l_2, \ldots, l_x$ involve in the changes of states of artifact C_i
- $VL \subseteq V \times L$ is a set of *service-role* relations between services and their corresponding organization roles. Relation $(v_i, l_x) \in VL$ means that service v_i is provided by organization role l_x.

Given ACC model Π, a *lifecycle model* of an artifact can be generated by deriving corresponding business rules that are used to induce state transitions of such artifact. A *lifecycle* of artifact class C_i, denoted as $\mathcal{L}_{C_i}$, can be defined as tuple (C_i, T), where $T \subseteq C_i.S \times \Pi.R \times (C_i.S \cup C_i.S^f)$ is a 3-ary transition relation. T^* is a reflexive transitive closure of T. Next, we define the *lifecycle occurrence* (*L-occurrence*) for a particular sequence of states occurring in the lifecycle. Based on *L-occurrences*, we define a *well-formed* behavior property of an individual lifecycle in which is used later for the discussion of a *soundness* property of the ACC model. A *L-occurrence* of lifecycle $\mathcal{L}_{C_i}$ is denoted as $\sigma = (s^{init}, .., s_f)$ such that for every state s in σ, there exists final state $s_f \in C_i.S^f$ and s_f can be reached from s^{init} through s by a particular firing sequence of some business rules in R. We also denote $\delta_{C_i} = \{\sigma_1, \sigma_2, .., \sigma_x\}$ for a pairwise disjoint set of all possible *L-occurrences* in $\mathcal{L}_{C_i}$. Now, given ACC model Π, Lifecycle $\mathcal{L}_{C_i}$ of $C_i \in \Pi.Z$ is *well-formed* iff (1) there exists business rule $r \in \Pi.R$ such that r induce one and only one transition in $\mathcal{L}_{C_i}$; and (2) for every non-final state $s \in S_i$, there exists s in some *L-occurrence* of $\mathcal{L}_{C_i}$; and (3) for every final state $s_f \in S_i{}^f$, there exists *L-occurrence* $\sigma \in \delta_{C_i}$ such that s_f is a last state of the sequence in σ.

Definition 2: (Public and Private views). Given ACC model Π, a *public view* of Π, denoted as $p(\Pi)$, is the abstraction of Π, such that it represents only agreed (abstracted) lifecycles of shared artifacts in Π. *Private view* Π^l is the local process of Π for organization role $l \in \Pi.L$ such that it represents only lifecycles of its own local artifacts and the lifecycles of shared artifacts in $p(\Pi)$.

As already discussed, when an agreed public view is constructed, involved organizations can have their own freedom to modify their own private view. Here, we define two behavioral-based modification operations, namely *lifecycle modification*: (1) *refine* (responsible parts of) the lifecycle of shared artifacts, and (2) *extend* shared artifacts with local artifact(s) that are required to coordinate with such shared artifacts. Note that refining existing local artifacts is also considered as extending them to the public view. To refine a shared artifact in a private view, we define *lifecycle fragment* representing a partial lifecycle that is embedded into its existing lifecycle.

Definition 3: (Lifecycle Fragment or *L*-fragment). Given ACC model Π, artifact $C_i \in \Pi.Z$, and a set of states $S = C_i.S \cup C_i.S^f$, we denote lf^{C_i} for a *L*-fragment which is a nonempty connected sub-lifecycle of lifecycle $\mathcal{L}_{C_i}$. Let lf^{C_i} be a 4-tuple $(S', T', T_{in}, T_{out})$ where $S' \subseteq C_i.S \backslash \{s^{init}\}$, $T' \subseteq S' \times \Pi.R \times S' \subseteq \mathcal{L}_{C_i}.T$, a set of entry transitions $T_{in} = \mathcal{L}^l_{C_i}.T \cap ((S \backslash S') \times \Pi.R \times S'))$, and a set of exit transitions $T_{out} = \mathcal{L}_{C_i}.T \cap (S' \times \Pi.R \times (S \backslash S'))$ such that for every state $s \in S'$, there exist s in *L-occurrences* δ of lf^{C_i}.

Next, we define *lifecycle modification* in a private view for the refinement (by *L*-fragments) of shared artifact and the extension of local artifacts.

Definition 4: (Lifecycle modification). Given public view $p(\Pi)$, private view Π^l can be modified based on $p(\Pi)$ with a *lifecycle modification*, denoted as Σ^l, and it is tuple (LM^l, LF^l), where LM^l is a set of extended lifecycles (of local artifacts) and LF^l is a set of *L*-fragments $\bigcup_{i=1}^{|Z^l|}\{lf_j{}^{C_i^l}\}$, $lf_j{}^{C_i^l} = \{lf_1{}^{C_i^l}, lf_2{}^{C_i^l}, ..., lf_k{}^{C_i^l}\}$, where $lf_n{}^{C_i^l}(1<n<k)$ is a *L*-fragment of $C_i^l \in \Pi^l.Z$ refining the shared lifecycle of its C_i in $p(\Pi)$.

Fig. 2 shows an example of public and a private view of organization L_1 in the collaboration where private view Π^{L_1} modify its public view with a lifecycle modification consisting of extended local lifecycles of artifacts $\{C_1, C_2, C_3\}$, refined *L*-fragments $\{lf_1{}^{C_A}, lf_2{}^{C_A}\}$ on artifact C_A, and *L*-fragment $lf_3{}^{C_B}$ on artifact C_B. The

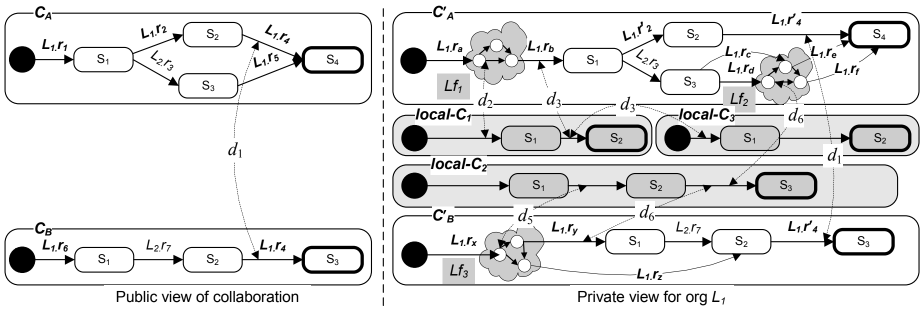

Fig. 2. Public view and its private view with *lifecycle modification*

organization also adds or modifies business rules used to synchronize local artifacts with shared artifacts (within the refined *L*-fragments) to complete the overall process.

3 Validating Changes to Private View of a Collaboration Model

In this section, we discuss about the validation of changes made to private views caused by *lifecycle modification*, i.e., how to ensure that the local changes to the private view of one organization do not interfere with the whole collaboration. Note that we confine our discussion only to the behavioural aspect of collaboration not the validation of artifacts' data. However, formal approaches to data verification can be found in [3]. Here, we use state transition system to describe the behavioral aspect of the ACC model, which can be generated by a composition technique presented in [6].

A state transition system of ACC model Π, namely *ACC machine* and denoted as M_Π, is tuple $(S,\ s^{init},\ T,\ S^f)$, where S and s^{init} are a set of ACC states and the initial state, respectively. $T \subseteq S \cup \{s^{init}\} \times \Pi.R \times G \times S$ is a set of transitions and G is a set of guards (state propositions). $S^f \subseteq S$ is a set of final states such that for every state s in S^f, s contains a combination of final state of every artifact in Π. M_Π is generated by iteratively composing a lifecycle of each artifact in $\prod$.

Definition 5: (Soundness). Given ACC model Π and its ACC machine M_Π, Π is *sound* iff (1) for every artifact $C_i \in \Pi.Z$, the lifecycle of C_i is *well-formed*; and (2) for every artifact $C_i \in \Pi.Z$, there exists a synchronization dependency between C_i and some other artifact; and (3) M_Π is *well-formed* and for every transition $t \in M_\Pi.T$, guard g of t contains no state referencing of any other artifact.

Condition (2) of the *soundness* property guarantees that the ACC model will consist of only desired artifacts that are used in the collaboration, and the condition (3) ensures the *reachability of goal states* of the collaboration and it also implies that a domain of artifacts and their states referenced in the model is bounded. Next, we define the *view conformance* between public and private views by checking whether the lifecycle of the later covers the former. We generalize *lifecycle coverage* for both artifact and ACC machine by reusing the *L*-occurrences definition of an individual artifact for a machine. Given two machines $m_x = (S_x, {s_x}^{init}, T_x, {S_x}^f)$ and

m_y = (S_y,$s_y{}^{init}$, T_y, $S_y{}^f$) and a set of states that exist in both $m_x.S$ and $m_y.S$, $S_{x\cap y}$ = $S_x \cap S_y$, *L*-occurrences of m_x is *covered* by *L*-occurrences of m_y iff $\forall s_i, s_j \in S_{x\cap y}$, $\exists(s_i, r, s_j) \in T_x$, $\forall s_k \in S_y \setminus S_{x\cap y}$, $s_i T_y{}^* s_k \wedge s_k T_y{}^* s_j$. Fig. 3 shows that lifecycle (a) is not *covered* by (b) as in the occurrences of (b) state *a* can reach state *c* without passing state *b*; in contrast, lifecycle (a) is *covered* by (c).

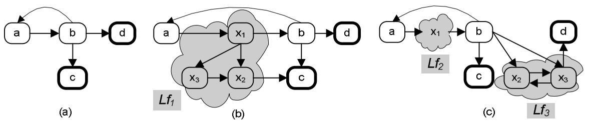

Fig. 3. Example of different refinement of *L*-fragments

Definition 6: (View conformance). Given private view Π^l modified based on public view $p(\Pi)$, Π^l *conforms* to $p(\Pi)$ iff *L*-occurrences of the ACC machine of $p(\Pi)$ is *covered* by *L*-occurrences of the ACC machine of Π^l.

Although the *view conformance* is defined based on the overall behavior of the model, it is important to discuss how we can validate the local modification. If such modification is valid and correctly applied, then we can guarantee that it will preserve the consistent behavior between the public and private views. Specifically, we apply the lifecycle coverage checking between public and private views of a shared artifact refined by *L*-fragments, and then we validate the whole modification with the extended local artifacts in the private view. Now, given private view Π^l modified based on *sound* public view $p(\Pi)$ with *lifecycle modification* Σ^l, Σ^l is said to be *valid* iff Σ^l can make Π^l *sound* and Π^l *conforms* to $p(\Pi)$. Let the lifecycle of C_i^l in private view Π^l refine its public lifecycle of corresponding artifact C_i in public view $p(\Pi)$ with a set of *L*-fragments $LF^{C_i^l}$. *L*-occurrences of C_i is *covered* by *L*-occurrences of C_i^l iff, for every $lf_j \in LF^{C_i^l}$, lf_j has a single entry state and a single exit state. We name this lf_j as *SESE L-fragment*. For example, in Fig. 3, lifecycle (c) consists of *SESE L-fragments* lf_2 and lf_3 refining lifecycle (a); thus, lifecycle (c) *covers* lifecycle (a).

Next, we discuss how the synchronization is taken into account for the validation of private views. First, we define the *locally-bound* property of the lifecycle of local artifact that synchronizes with a shared artifact. Then, we use this property to induce the coverage of all related artifacts in the private view. Let private view Π^l be modified based on *sound* public view $p(\Pi)$. For any local artifact $C_x^l \in \Pi^l.Z$ and any shared artifact $C_y^l \in \Pi^l.Z \cap p(\Pi).Z$ which is refined by a set of *L*-fragments $LF^{C_y^l}$, if every synchronization dependency between lifecycle $\mathcal{L}_{C_x}^l$ of C_x^l and lifecycle $\mathcal{L}_{C_y}^l$of C_y^l occurs within one and only one fragment $lf_j \in LF^{C_y^l}$ such that lf_j is *SESE L-fragment*, then $\mathcal{L}_{C_x}^l$ is *locally-bound* and $\mathcal{L}_{C_x}^l$ preserves the *soundness* of Π^l. The locally-bound property is transitive from one to another local artifact if a locally-bound artifact synchronizes with another local artifact such that they do not synchronize with any shared artifact, then such artifact is transitively *locally-bound.*

The transitivity is invalid if such artifact synchronizes with a *non locally-bound* artifact. For example in Fig. 2, lifecycle of C_1 is *locally-bound* since synchronizations d_2 and d_3 occur within *SESE fragment* lf_1 of C'_A, and the lifecycle of C_3 is transitively *locally-bound* via d_3. In contrast, with C_2, we can see that synchronizations d_5 and d_6 occur in non-*SESE L-fragment* $lf_3$of artifact C'_B (even though d_6 synchronizes with *SESE L-fragment* lf_2); thus, the lifecycle of C_2 is not *locally-bound*. Now, if organizations want to modify their private views using lifecycle modification based on their public view, they need to ensure that their local modifications do not violate the view conformance. Here, we show the conditions that restrict the *valid* modification. *Lifecycle modification* $\Sigma^l = (LM^l, LF^l)$ is *valid* for Π^l based on *sound* $p(\Pi)$, if, for every *L*-fragment $lf_j \in LF^l$, lf_j is *SESE L-fragment* and for every lifecycle $\mathcal{L}^l_{C_i} \in LM^l$, $\mathcal{L}^l_{C_i}$ is *locally-bound*, then *L*-occurrences of ACC machine of Π^l is *covered* by *L*-occurrences of ACC machine of $p(\Pi)$ (i.e., Π^l conforms to $p(\Pi)$) and Π^l is *sound*. As the result, we can assert the *soundness* property with the *conformance* by only checking the local lifecycle modification. Whatever changes made on a private view will not impact on the behavior of collaboration if they preserve *valid* modification.

4 Related Work and Discussion

In the area of artifact-centric process modeling, Nigam and Caswell [2] introduced the concept of modelling business artifacts with lifecycles. Bhattacharya et al. [3] used services to model activities and a set of business rules to capture and represent a process model with the study of necessary properties such as reachability of goal states, absence of deadlocks, and redundancy of data. Fritz et al. [12] studied problems of goal-directed artifact-centric workflow construction. Hull et al. [1] proposed an approach to interoperation of organizations hubs based on business artifacts providing a centralized and computerized rendezvous point. Kuster et al. [8] presented a notion of compliance of a business process model with object lifecycles and a technique for generating the model from such set of lifecycles. In [4], we proposed an artifact-centric process view framework to allow role-based process abstraction that can be used to construct public views. All the above works serve as a basis for this work. In the area of object-oriented design, Schrefl and Stumptner [5] studied the consistency criteria of the inheritance of object life cycles. While their work focused on the inheritance of single object, we adopt and extend their study of consistent refinement to multiple lifecycles where the multiple inheritances of artifacts may interact with each other in the collaboration.

In the area of cross-organizational workflow management, several approaches presented in [10, 11, 13, 14, 15] were proposed to define, construct, and validate public and private process views based on consistency rules and correctness-preserving constraints. Van der Aalst et al. [7] proposed a process-oriented contract agreed in the public view with a criterion for accordance between public and private views. Their work is done based on process-centric setting while our work is on artifact-centric setting. Lohmann and Wolf [9] presented an approach for artifact-centric choreographies with the concept of agents and location-aware artifacts using Petri-net model. They proposed a mechanism for automatic generation of an interaction model that serves as a contract between the agents ensuring that specified global goal states on the involved artifacts can be reached. Compared to our work, their work only focused on the interaction model and did not consider the model of local processes, while our approach presents the artifact-centric collaboration model together with the validation mechanism for local model modification where preserving global correctness.

5 Conclusions

This paper presents an artifact-centric view-based approach to modeling inter-organizational business processes. We define the ACC model with the notion of *conformance* of public and private views to address the choreography requirements. The view conformance checking serves as the main resolving mechanism. We also show that a modification on the private view of local organization can be theoretically validated against the *soundness* properties for both public and private views.

References

1. Hull, R., Narendra, N.C., Nigam, A.: Facilitating Workflow Interoperation Using Artifact-Centric Hubs. In: Baresi, L., Chi, C.-H., Suzuki, J. (eds.) ICSOC-ServiceWave 2009. LNCS, vol. 5900, pp. 1–18. Springer, Heidelberg (2009)
2. Nigam, A., Caswell, N.S.: Business artifacts: An approach to operational specification. IBM Systems Journal 42(3), 428–445 (2003)
3. Bhattacharya, K., Gerede, C., Hull, R.: Towards Formal Analysis of Artifact-Centric Business Process Models. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 288–304. Springer, Heidelberg (2007)
4. Yongchareon, S., Liu, C.: Process View Framework for Artifact-Centric Business Processes. In: Meersman, R., Dillon, T.S., Herrero, P. (eds.) OTM 2010. LNCS, vol. 6426, pp. 26–43. Springer, Heidelberg (2010)
5. Schrefl, M., Stumptner, M.: Behavior-Consistent Specialization of Object Life Cycles. ACM TOSEM 11(1), 92–148 (2002)
6. Lind-Nielsen, J., Andersen, H., Hulgaard, H., Behrmann, G., Kristoffersen, K., Larsen, K.: Verification of Large State/Event Systems Using Compositionality and Dependency Analysis. Formal Methods in System Design 18(1), 5–23 (2001)
7. Van Der Aalst, W.M.P., Lohmann, N., Masuthe, P., Stahl, C., Wolf, K.: Multipart Contrats: Agreeing and Implementing Interorganizational Processes. The Computer Journal 53(1), 90–106 (2010)
8. Küster, J.M., Ryndina, K., Gall, H.C.: Generation of Business Process Models for Object Life Cycle Compliance. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 165–181. Springer, Heidelberg (2007)
9. Lohmann, N., Wolf, K.: Artifact-centric Choreographies. In: Maglio, P.P., Weske, M., Yang, J., Fantinato, M. (eds.) ICSOC 2010. LNCS, vol. 6470, pp. 32–46. Springer, Heidelberg (2010)
10. Chebbi, I., Dustdar, S., Tata, S.: The view-based approach to dynamic inter-organizational workflow cooperation. DKE 56, 139–173 (2006)

11. Van Der Aalst, W.M.P., Lohmann, N., Massuthe, P., Stahl, C., Wolf, K.: From Public Views to Private Views – Correctness-by-Design for Services. In: Dumas, M., Heckel, R. (eds.) WS-FM 2007. LNCS, vol. 4937, pp. 139–153. Springer, Heidelberg (2008)
12. Fritz, C., Hull, R., Su, J.: Automatic construction of simple artifact-based business processes. In: ICDT 2009, vol. 361, pp. 225–238. ACM, New York (2009)
13. Zhao, X., Liu, C., Sadiq, W., Kowalkiewicz, M., Yongchareon, S.: Implementing Process Views in the Web Service Environment. WWWJ 14(1), 27–52 (2011)
14. Zhao, X., Liu, C., Sadiq, W., Kowalkiewicz, M.: Process View Derivation and Composition in a Dynamic Collaboration Environment. In: Chung, S. (ed.) OTM 2008, Part I. LNCS, vol. 5331, pp. 82–99. Springer, Heidelberg (2008)
15. Zhao, X., Liu, C., Yang, Y., Sadiq, W.: Aligning Collaborative Business Processes - An Organisation-oriented Perspective. IEEE TSMC 39(6), 1152–1164 (2009)
16. Yongchareon, S., Liu, C., Zhao, X., Xu, J.: An Artifact-centric Approach to Generating Web-based Business Process Driven User Interfaces. In: Chen, L., Triantafillou, P., Suel, T. (eds.) WISE 2010. LNCS, vol. 6488, pp. 419–427. Springer, Heidelberg (2010)