information

MDPI

Article

The Adaptation of a Model of an Artifact-Centric Business Process Instance and Its Validation

Junbao Zhang * and Guohua Liu

School of Computer Science and Technology, Donghua University, Shanghai 201600, China; ghliu@dhu.edu.cn

* Correspondence: junbaozb0451@163.com

Received: 29 November 2018; Accepted: 30 January 2019; Published: 11 February 2019

Abstract: The adaptability of an in-progress business process is an essential requirement for any business process management system in dynamic business process environments. Over the last two decades, the artifact-centric approach for business process management has been evidenced to have higher level of flexibility. However, the adaptation of a model of an artifact-centric business process instance is still inevitable and pervasive due to the complex and ever-changing business environments. Almost all works of artifact-centric business process neglect this issue. To fill this gap, we propose a special business rule called adaptation rule to address the dynamic adaptation problem and describe the adaptation by a global adaptation model. Moreover, we provide a validation mechanism over our proposed adaptation rule of the global adaptation model to guarantee the behavior correctness of the adaptation. Through this validation approach, computing the lifecycle of the global adaptation model can be avoided.

Keywords: business process change; artifact-centric business process; process adaptation; adaptation rule; adaptation correctness

1. Introduction

A critical challenge for the competitive enterprise is its ability to quickly react to business process changes and adequately address them. The adaptability of in-progress business process is an essential requirement for any business process management system in dynamic business process environments [1]. Adaptation of a business process (also referred to as process adaptation) refers to the action of customizing a process instance to make it applicable to a particular situation [2].

In the past two decades, a new business process modeling paradigm has emerged, i.e., artifact-centric business process modeling [3,4]. It has been proved that this approach provides a higher level of flexibility of workflow enactment and evolution [5]. However, few works have developed artifact-centric business process paradigms for the adaptation of business processes. Wang et al. [6] proposed a taxonomy of changes of artifact-centric business processes based on a three-level workflow implementation and an analysis of the impact of changes. Yongchareon et al. [5] addressed the requirements of private artifact-centric business processes change and provided a verification mechanism to preserve the correctness of public business processes in interorganizational business process circumstances based on a view framework. Li and Yu [7] proposed three primitive changes (add, delete and modify) of activities of artifact-centric workflow model in the execution level and proposed guidelines to guarantee the execution of business process instances that are adapted. However, the change of artifact information model is not discussed and theoretical validation of the adaptation is not presented. Eshuis et al. [8] formally reasoned about properties of a process model that are preserved after the model is changed and defined several change operations that preserve certain properties based on Case Management Model and Notation (CMMN), which is an

artifact-centric business process model based on the Guard–Stage–Milestone (GSM) model [9,10]. Although flexibility is naturally considered by artifact-centric business process modeling approach, dynamic adaptation of a model of a business process instance is still inevitable because of the turbulent business process environments [11–13]. However, almost all works in artifact-centric business process community neglect this issue. It is difficult to address this issue, mainly for the following two reasons. First, both data and process should be considered during the adaptation of business process instances. Second, due to the flexible and the declarative properties of artifact-centric business processes, it is hard to guarantee the correctness of the adaptation, i.e., a business process instance that is adapted can still reach an acceptable business goal without deadlock.

Many efforts in activity-centric business process community have addressed this issue. Vasilecas et al. [14] proposed a rule- and context-based approach to dynamically define a business process model. The adaptations of business process instances are implemented by changing contexts, rules and activities. Hu et al. [15] proposed a rule-, context- and goal-based approach for the adaptation of business process instances. Ayora et al. [16] proposed nine change patterns to enable the modeling and evolution of process families at a high level of abstraction. They extended these change patterns to 10 change patterns in [17] and verified that these change patterns can guarantee the structure and the correct behavior of the variants by case study. Policies are used to decide whether a business process instance should be adapted. Nunes et al. [18] proposed the adaptation of business process instances based on context. Patiniotakis et al. [19] used a multi-criteria decision-making approach for the adaptation of event-driven business process instances. Oberhauser [20] proposed adaptation processes to adapt business process instances. Artifact-centric business processes focus on what can be done (goals and progresses) instead of what should be done (tasks or corresponding services) that activity-centric business processes focus on [21]. Nevertheless, due to this difference between activity-centric and artifact-centric modeling approaches, the adaptation methods in activity-centric modeling paradigm are not applicable for artifact-centric business processes.

To solve the issue, we present a taxonomy of changes of artifact-centric business processes and analysis characteristics of these changes. Considering these characteristics, we propose a special business rule called adaptation rule to implement the dynamic adaptation of a model of an artifact-centric business process instance and describe the adaptation by a global adaptation model. Although a global adaptation model can be converted into an artifact-centric business process model, we provide an incremental validation method to verify behavior properties of the business process instances that are adapted by an adaptation rule. As such, the traditional lifecycle composition process of model validation can be avoided. Note that our proposed solution partially solves the problem of adaptation of an artifact-centric business process. How to adapt a business process model to create a new business process model is not discussed in this paper.

2. Artifact-Centric Business Process Model

There are abundant artifact-centric business process models defined formally or informally, but they reach a consensus that business processes are dominated by business artifacts [22]. In this study, we adopt the ACP model with some modifications as it reflects the original idea of artifact-centric approach faithfully [22–24].

Definition 1 (modified from [23])**.** *An artifact-centric business process model is a triple $M = (Z, V, R)$, where Z is an artifact schema. An artifact schema is a set of artifact classes $Z = \{C_1, C_2, \ldots, C_n\}$, C_i $(1 \leq i \leq n)$ is an artifact class that defined by a tetrad (A, s_0, S, S_f), where A is a set of name-value paired attributes; s_0 is the default start state; S is a set of states; and $S_f \subset S$ is the set of final states. V is a set of services. R is a set of business rules over Z. A business rule is a triple (λ, β, v), where λ is the precondition; β is the postcondition; and $v \in V$ is a service that operates one or more artifacts $c_1, c_2, \ldots, c_n$ of artifact classes $C_1, C_2, \ldots, C_n$, respectively.*

Artifacts are business-relevant entities that are created and evolved through business processes. An artifact class (hereafter artifact) defines an information model and its lifecycle [3]. A business

rule associates service(s) with artifact(s) in a Condition–Action style. Each business rule describes which service is invoked and which state of an artifact is changed under what precondition and postcondition [23]. λ and β are defined by quantifier free first order logic formula. In the formula, there are two types of proposition over schema Z: (1) state proposition (the *instate* predicate); and (2) attribute proposition (the *defined* predicate and scalar comparison).

Example 1.

We use a purchase business process [23] to illustrate the concept of the artifact-centric business processes. Its artifact-centric representation is described in Figure 1.

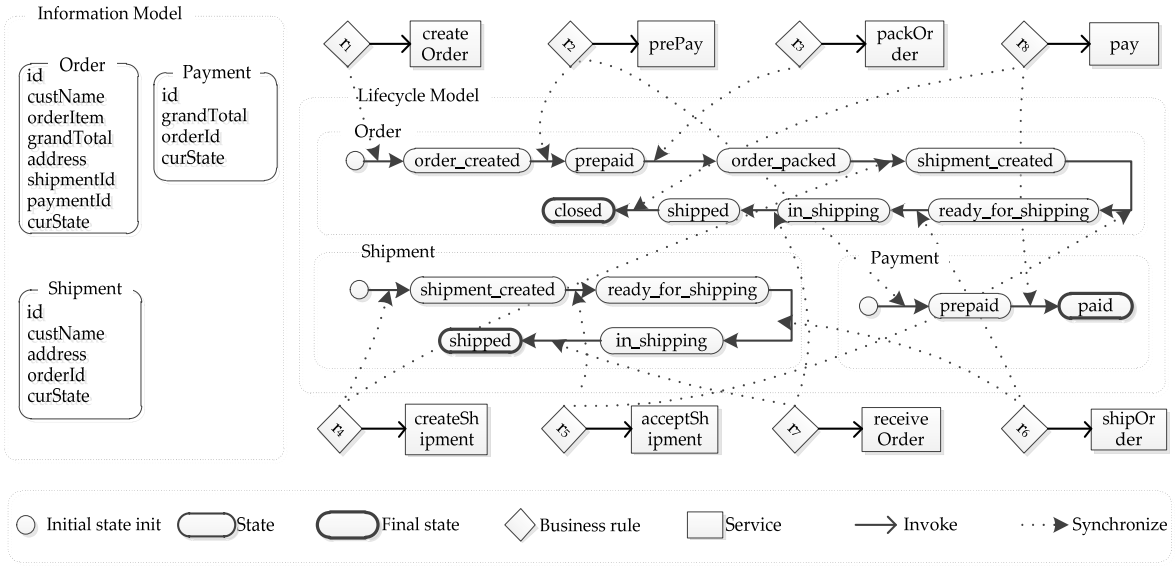

Figure 1. Artifact-centric representation of a purchase business process.

A purchase business process consists of an information model (Figure 1, top left) and a lifecycle model (Figure 1, top right). There are three artifacts, namely *Order*, *Payment* and *Shipment*, in the process. A customer can instantiate the process by selecting some products and creating an order artifact instance, and follows the process to prepay the order. At this moment, a payment artifact instance is created and move to the *prepaid* state. A seller can accept the order and get ready to ship the order after the order has been packed, and follow the process to ship the order. Now, a shipment artifact instance is created. After the seller ships the order, the customer can receive the order and follow the process to finish the payment. Eight business rules and eight services work together to update the information model of the process and move three artifacts from one state to another along their lifecycles.

There is an example of business rules in the purchase process (Table 1). In Table 1, state proposition *instate*(*O*, *order_created*) holds if state of *O* is *order_created*, i.e., *O.curState* = *order_created*; and attribute proposition *defined*(*O.custName*) holds if *custName* of *O* has been defined.

Table 1. An example of business rules in the purchase process (Figure 1).

r_1: Customer *C* requests to make an Order *O* with order items *OIS*.	
precondition	*instate*(*O*, *init*)
service	*create*(*O*, *C*, *OIS*)
postcondition	*instate*(*O*,*order_created*) ∧ *defined*(*O.custName*) ∧ *defined*(*O.address*) ∧ *defined*(*O.grandTotal*) ∧ *defined*(*O.orderItem*)
r_2: Customer *C* prepays an Order *O* through payment *P*.	
precondition	*instate*(*O*, *order_created*) ∧ *instate*(*P*, *init*) ∧ *O.grandTotal*>0
service	*prePay*(*O*, *P*, *C*)

postcondition	*instate*(*O*, *prepaid*) ∧ *instate*(*P*, *prepaid*) ∧ *defined*(*O.paymentId*) ∧ *defined*(*P.paymentId*) ∧ *O.paymentId* = *P.paymentId*
***r*3: A seller collects order items of an Order *O* and pack the Order.**	
precondition	*instate*(*O*, *prepaid*)
service	*packOrder*(*O*)
postcondition	*instate*(*O*, *order_packed*)
***r*4: Create Shipment *S* according to Order *O*.**	
precondition	*instate*(*O*, *order_packed*) ∧ *instate*(*S*, *init*)
service	*createShipment*(*O*, *S*)
postcondition	*instate*(*O*, *shipment_created*) ∧ *instate*(*S*, *shipment_created*) ∧ *defined*(*O.shipmentId*) ∧ *defined*(*S.shipmentId*) ∧ *O.shipmentId*=*S.shipmentId*

3. Adaptation of a Model of an Artifact-Centric Business Process Instance

In this section, we present a taxonomy of changes of an artifact-centric business process and discuss characteristics of these changes. Considering characteristics of these changes, we propose a global adaptation model for the adaptation of a model of an artifact-centric business process instance.

3.1. Taxonomy of Changes of an Artifact-Centric Business Process

According to key components of an artifact-centric business process, we present the taxonomy of changes of an artifact-centric business process in Figure 2. These changes are inspired from the work of Wang et al [6]. However, we focus on model level changes.

Artifact change contains two types of sub changes: attribute change and artifact existence change. Attribute change may add an attribute to an artifact, delete an attribute from an artifact and modify an attribute of an artifact. Artifact existence change includes add an artifact, delete an artifact, merge artifacts and split an artifact.

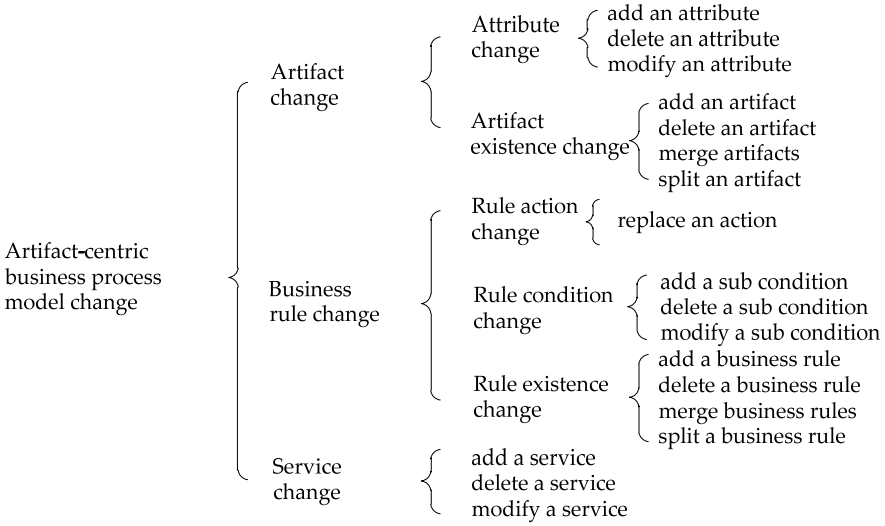

Figure 2. The taxonomy of changes of an artifact-centric business process.

Business rule change includes rule action change, rule condition change and rule existence change. Rule action change refers to replacement of an action. Rule condition change refers to modification of conditions. A sub condition could be added to λ or β of a business rule. A sub condition may be deleted from λ or β of a business rule. A sub condition of λ or β of a business rule may be modified. Business rule existence change includes add a business rule, delete a business rule, merge business rules and split a business rule.

Service change includes: add a service, delete a service and modify a service.

Suppose we want to adapt a business process instance from model M to model M'. According to the taxonomy, we can see that artifacts of M and M' may have following differences.

From view of information models of artifacts:

(1) An artifact of M may map to one artifact of M' if attribute changes are used alone. Nevertheless, the cardinality of attributes may be different if changes of add an attribute or delete an attribute are used.

(2) An artifact of M may map to multiple artifacts of M' if changes of split an artifact are used alone.

(3) Multiple artifacts of M may map to one artifact of M' if changes of merge artifacts are used alone.

(4) Multiple artifacts of M may map to multiple artifacts of M' if combination of changes is used.

(5) Artifacts of M may map partially to artifacts of M' if change of add an artifact or delete an artifact is used.

(6) Attributes of artifacts of M map partially to attributes of artifacts of M' while considering Characteristics (1)–(5).

Similarly, from view of lifecycles of artifacts, relationships of artifacts between M and M' may be one-to-one, one-to-many, many-to-one, many-to-many and partial mapping. Usually, mappings between states do not exist if changes of merge artifacts and split an artifact are used because both the cardinality of states and the names of states may be changed due to these changes. As a result, states of artifacts of M map partially to states of artifacts of M'.

From former analysis, we can see that relationships of artifacts between two models are complex from perspectives of both information models of artifacts and lifecycles of artifacts. Because data and states of artifacts of business process instances that should be adapted cannot be decided according to relationships of artifacts between M and M', it is hard to decide when and how to adapt a model of a business process instance automatically. This circumstance is common for most changes in Figure 2, for example, add an attribute, add an artifact, merge artifacts, and split an artifact.

Note that these changes are interleaving [6]; coordination of changes of artifacts, business rules and services should be done to generate a new version of business process that is modeled as M'. How to generate M' by these changes is omitted, as it is not the interests of this paper. Next, we present our method for the adaptation of a model of an artifact-centric business process instance semi-automatically.

3.2. The Proposed Global Adaptation Model

There are two operations while implementing the adaptation of a model of an artifact-centric business process instance when considering perspectives of data and process of an artifact-centric business process. Suppose the instance is adapted from model M to model M'. One operation is to convert the information models of artifacts of the instance from M to M'. The other operation is to set the converted artifact instances with proper data (attributes with proper values and artifact instances with proper states).

Considering complex relationships of artifacts of M and M' while adapting a business process instance from M to M', we propose special user-defined business rules called adaptation rule for deciding when and how to implement the adaptation. After an adaptation rule is defined and deployed to the business process system, a business process instance will be adapted dynamically by the adaptation rule if the precondition and the postcondition of the adaptation rule are satisfied. We use a global adaptation model to describe the adaptation.

Definition 2. *A global adaptation model is a tetrad $\Gamma = (M, M', v_a, r_a)$, where M and M' are two artifact-centric business process models; v_a is a service that implement the two operations while adapting a business process instance of M to M'; and r_a is a special business rule called adaptation rule that is a tuple $(\lambda_M, \lambda_{M'}, \beta_M, \beta_{M'}, v_a)$, where λ_M is the sub condition of r_a on M in the precondition of r_a; $\lambda_{M'}$ is the sub condition of r_a on M' in the precondition of r_a; β_M is the sub condition of r_a on M in the postcondition of r_a; and $\beta_{M'}$ is the sub condition of r_a on M' in the postcondition of r_a. $\lambda_M \wedge \lambda_{M'}$ formulates the precondition of r_a; and $\beta_M \wedge \beta_{M'}$ formulates the postcondition of r_a.*

In Definition 2, v_a is a service that implements a partial function f: $M.Z.C_1 \times \ldots \times M.Z.C_m \ldots \mapsto M'.Z.C_1 \times \ldots \times M'.Z.C_n$, $m = |M.Z|$ and $n = |M'.Z|$. $\lambda_{M'}$ takes formula $instate(M'.Z.C_1, s_0) \wedge \ldots \wedge instate(M'.Z.C_n, s_0)$ $(n = |M'.Z|)$ as default. Note that there is a special final state s_{gf} in $C_i.S_f$, where $C_i \in M.Z$. State proposition in β_M takes formula $instate(M.Z.C_1, s_{gf}) \wedge \ldots \wedge instate(M.Z.C_m, s_{gf})$ $(m = |M.Z|)$ as default. If state of an artifact instance was logged as s_{gf}, it means that the corresponding process instance of this artifact instance has been adapted.

Given a global adaptation model Γ, it can be mapped to an artifact-centric business process model M_Γ, where $M_\Gamma.Z = \Gamma.M.Z \cup \Gamma.M'.Z$, $M_\Gamma.V = \Gamma.M.V \cup \Gamma.M'.V \cup \{\Gamma.v_a\}$ and $M_\Gamma.R = \Gamma.M.R \cup \Gamma.M'.R \cup \{\Gamma.r_a\}$. Compare M_Γ with M and M', we can see that only $\Gamma.r_a$ needs to be defined as we suppose $\Gamma.v_a$ has been provided. From definition of an adaptation rule, we can see that an adaptation rule dominates artifacts of two business process models, instead of artifacts of the same business process model as compared with general business rules. Another difference is that an adaptation rule synchronizes all artifacts of both business process models as contrast to part of artifacts within one business process model. In this paper, we suppose $\Gamma.M.Z \cap \Gamma.M'.Z = \emptyset$.

Example 2.

It is hard to find a real-world business process to illustrate all circumstances. In this example, we use two domain independent process models to illustrate the concept of our global adaptation model (Figure 3).

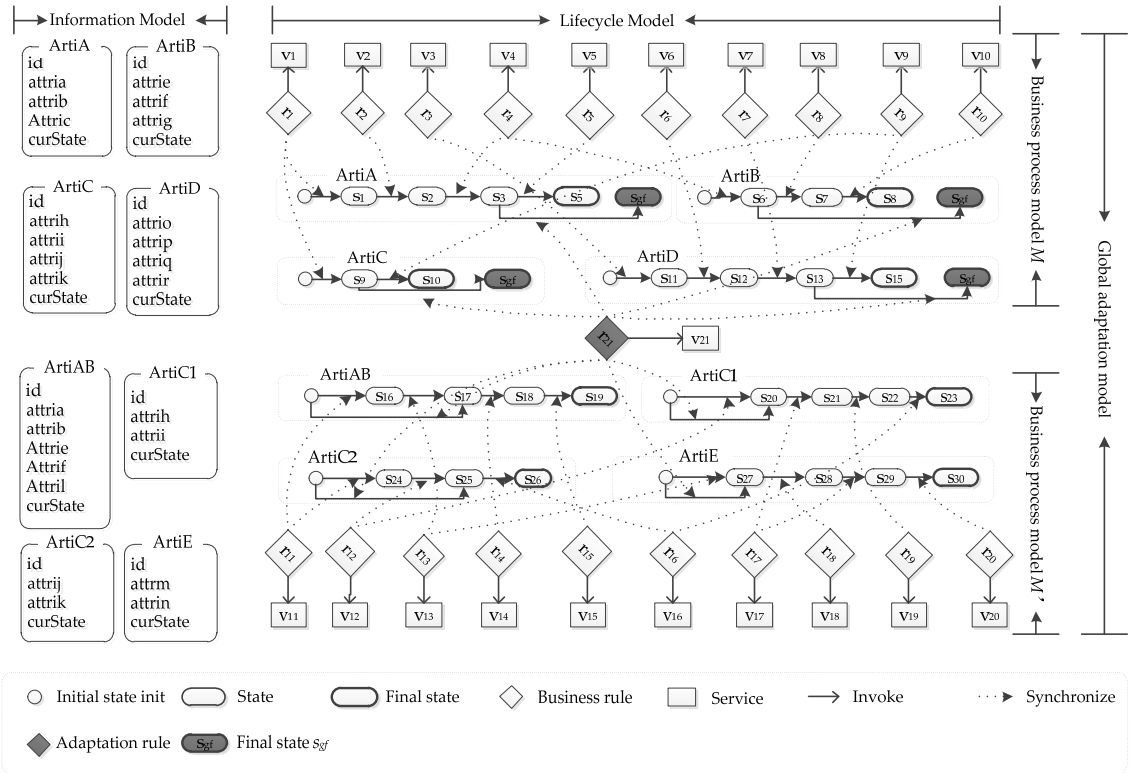

Figure 3. Artifact-centric representation of a global adaptation model.

Compare two business process models M and M' in Figure 3. two artifacts *ArtiA* and *ArtiB* are merged as artifact *ArtiAB*. Further attribute changes are used on *ArtiAB*, where the changes are delete attributes *Attric* and *Attrig* and add attribute *Attril*. Artifact *ArtiC* splits as two artifacts *ArtiC1* and *ArtiC2*. Artifact *ArtiD* is deleted and a new artifact *ArtiE* is added in model M'.

In Figure 3, we can see that, if adaptation rule r_{21} is evaluated, a business process instance of M will be adapted to model M' and continues to run according to definition of M'. Note that some complementary sub services may be incorporated in v_{21} to remedy losses and supplement gains of the business process instance due to the adaptation. For instance, a seller returns postage fee if type of

shipment of an order is changed from express to self-pick-up in a purchasing process. On the contrary, a customer pays postage fee if type of shipment of an order is changed from self-pick-up to express.

From Example 2, we can see that an adaptation rule combines two business process models as one brand new business process model M_Γ. Next, we validate a global adaptation model through its corresponding artifact-centric business process model.

The adaptation rule of Example 2 is defined in Table 2.

Table 2. The adaptation rule of Example 2.

r_{21}: **Adapt a business process instance of M to M'.**	
λ_M	$instate(ArtiA, s_3) \wedge instate(ArtiB, s_6) \wedge instate(ArtiC, s_9) \wedge instate(ArtiD, s_{13})\ldots$
$\lambda_{M'}$	$instate(ArtiAB, init) \wedge instate(ArtiC_1, init) \wedge instate(ArtiC_2, init) \wedge instate(ArtiE, init)$
v_a	$adapt(ArtiA\ a, ArtiB\ b, ArtiC\ c, ArtiD\ d, ArtiAB\ ab, ArtiC_1\ c_1, ArtiC_2\ c_2, ArtiE\ e, \ldots)$
β_M	$instate(ArtiA, s_{gf}) \wedge instate(ArtiB, s_{gf}) \wedge instate(ArtiC, s_{gf}) \wedge instate(ArtiD, s_{gf})\ldots$
$\beta_{M'}$	$instate(ArtiAB, s_{17}) \wedge instate(ArtiC_1, s_{20}) \wedge instate(ArtiC_2, s_{25}) \wedge instate(ArtiE, s_{27})\ldots$

4. Verification of a Global Adaptation Model

Verification of a global adaptation model Γ is crucial to guarantee the correctness of execution of a business process [24]. In this section, we use the soundness of a global adaptation model Γ to guarantee that business process instances of $\Gamma.M$ that are adapted by $\Gamma.r_a$ can reach business goals without deadlock. Considering flexible and declarative properties of artifact-centric business process, we use the correctness of an adaptation rule to guarantee that a business process instance that is adapted by the adaptation rule can reach an acceptable business goal.

4.1. Soundness of an Artifact-Centric Business Process Model

Before presenting the soundness of a global adaptation model, we introduce the soundness of an artifact-centric business process model as a foundation in this subsection.

It is known that verifying artifact behaviors is undecidable when the ability of creating new artifacts is considered [24]. However, the verification is decidable when artifact behaviors, i.e., the lifecycle of an artifact, are represented by finite states [5]. Differently, we generate the lifecycle of an artifact-centric business process directly as opposed to pair-wise composition of the lifecycles of two artifacts.

Definition 3 (modified from [5])**.** *Given an artifact-centric business process model M, the lifecycle of an artifact C ($C \in M.Z$), denoted as L_C is a triple (s_0, V_s, E_t), where s_0 is the default start state of C; $V_s = C.S \cup \{s_0\}$;* and *$E_t \subset V_s \times M.R \times V_s$ is the set of transitions.*

Given a transition $(s_x, r, s_y) \in E_t$, it means that state of an artifact will be updated from s_x to s_y if business rule r is evaluated.

Definition 4. *Given an artifact-centric business process model M, the lifecycle of M, marked as L_M, is a triple (s_0, V_s, E_t), where $s_0 = (L_{C_1}.s_0, \ldots, L_{C_n}.s_0)$, $n = |M.Z|$ and $C_i \neq C_j$ $(i \neq j)$. $V_s \subset L_{C_1}.V_s \times , \ldots, \times L_{C_n}.V_s$. Given a state $s \in V_s$, s is a final state* of *L_M if $s[i]$ is a final state of L_{C_i} for i in $[1,n]$. $E_t \subset V_s \times R \times V_s$* is a set of transitions*, where R is a set of business rules that transformed from M.R.*

Given a lifecycle L_M and a state $s \in L_M.V_s$, we call s as a business goal of L_M if there is a business process instance of M that is completed, where $C_i \in M.Z$ and $instate(C_i, s[i])$ $(1 \leq i \leq |s|)$ holds for the process instance. If not specified clearly, the sequence of states in s are determined by lexicography order of artifact names, which take marks of artifacts as default. Note that a business goal of L_M is equivalent to a final state of L_M if M is a general artifact-centric business process model; and not equivalent if M is an artifact-centric business process model that derives from a global adaptation model. In the second circumstance, a final state of L_M must be a business goal of L_M, but a business goal of L_M may not be a final state. For instance, state (s_0, s_0, s_0, s_0, s_{19} s_{23} s_{26}, s_{30}) is a business goal, not

a final state of $L_{M\Gamma}$ (Γ is the global adaptation model of Figure 3). We call lifecycle $L_{M\Gamma}$ as the lifecycle of Γ.

Let $cv(r, s_x, s_y)$ be a transformation function. $cv(r, s_x, s_y)$ returns a new business rule denoted as $r[s_x,s_y]$, where sub condition of state proposition in the precondition of r is substituted by $instate(C_M, s_x)$ and sub condition of state proposition in the postcondition of r is substituted by $instate(C_M, s_y)$. C_M is a virtual artifact that composites all the artifacts of model M, where their states appear in s_x. Given an artifact-centric business process model M, L_M can be generated iteratively from $L_M.s_0$.

Example 3.

In this example, we use aforementioned purchase business process that is modeled as M_P to illustrate the process of generating the lifecycle of M_P, which is marked as L_{M_P}. The initial state of the lifecycle is (*init*, *init*, *init*) for there are three artifacts in the model. At present, only precondition of r_1 holds. Because r_1 induces *Order* from *init* to *order_created*, state of a process instance will change from (*init*, *init*, *init*) to (*order_created*, *init*, *init*) if r_1 is evaluated. To make r_1 suitable for the lifecycle, the sub condition of state propositions of the precondition and the postcondition of r_1 are substituted by $instate(C_{M_P}, (init, init, init))$ and $instate(C_{M_P}, (order\_created, init, init))$, respectively. As such, it generates a new business rule, i.e., $cv(r_1, (init, init, init), (order\_created, init, init))$. Consequently, transition $((init, init, init), r_1[(init, init, init), (order\_created, init, init)], (order\_created, init, init))$ belongs to $L_{M_P}.Et$ and state (*order_created*, *init*, *init*) belongs to $L_{M_P}.V_s$. Similarly, the lifecycle of M_P is derived iteratively until $L_{M_P}.V_s$ and $L_{M_P}.Et$ are fixed. Finally, L_{M_P} is generated as Figure 4.

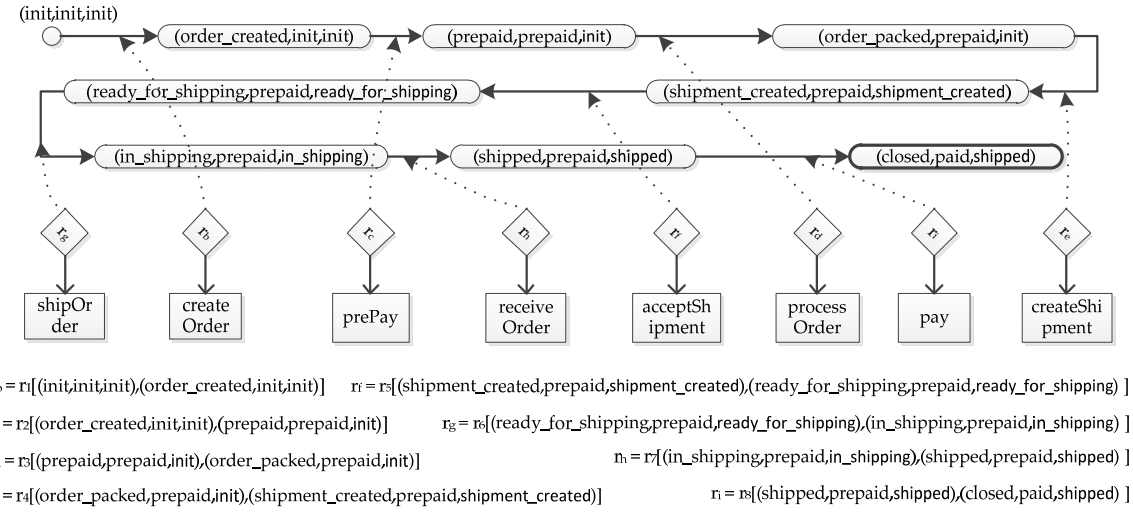

Figure 4. The lifecycle of the purchase business process model M_P (Figure 1).

Next, we introduce the soundness of a lifecycle with some modifications [5]. Given a lifecycle L, L is sound if and only if following two conditions are satisfied: (1) business rule $r \in rule(L)$ such that r induces one and only one transition, i.e.,

$$\forall r \in rule(L), (s_x, r, s_y) \in L.E_t \wedge (s_m, r, s_n) \notin L.E_t; \quad (1)$$

and (2) for every none final state s in $L.V_s$, s can be reached from $L.s_0$ and s can reach a final state, i.e.,

$$\forall s \in L.V_s \wedge \neg fi(s), \exists s_f, s_f \in L.V_s \wedge fi(s_f) \wedge L.s_0 \Rightarrow^* s \wedge s \Rightarrow^* s_f. \quad (2)$$

$rule(L)$ returns a set of business rules that appear in L. Function $fi(s)$ returns true if s is a final state of L. A transition (s_x, r, s_y) can be equivalently marked as $s_x \Rightarrow^r s_y$. The superscript of $\Rightarrow^r$ is omitted if it is not the concern. We write $s_x \Rightarrow^* s_y$ if state s_y can be reached from state s_x by some sequence of business rules in $rule(L)$. Condition (2) implies connectivity and goal-reachable of the lifecycle of an artifact or a business process model.

Definition 5. *Given an artifact-centric business process model M, M is sound if and only if* L_M *is sound.*

If an artifact-centric business process model M is sound, all business process instances that are regulated by M can reach some final states of L_M without deadlock [5].

4.2. Verification

In this sub section, we define the soundness of a global adaptation model Γ and verify it. We prove that the global adaptation model is sound if the adaptation rule in the model is sound. Here, we suppose $\Gamma.M$ and $\Gamma.M'$ are sound. Furthermore, we use an adaptation goal to define the acceptable business goals for business process instances that are adapted by an adaptation rule. Then, we use the correctness of the adaptation rule to guarantee that the business process instances can reach some acceptable business goals.

Definition 6. *Given a global adaptation model* Γ*, we say that* $\Gamma.r_a$ *is sound if and only if:* (1) s_x *is a reachable state from* $L_M.s_0$ *in the lifecycle of* $\Gamma.M$*; and* (2) s_y *is a reachable state from* $L_{M'}.s_0$ *in the lifecycle of* $\Gamma.M'$*. Given a global adaptation model* Γ*, let the artifact-centric business process model derived from* Γ *as* M_Γ*,* Γ *is sound if* M_Γ *is sound.*

s_x is a state of M according to sub condition $\Gamma.r_a.\lambda_M$, and s_y is a state of M' according to sub condition $\Gamma.r_a.\beta_{M'}$. For example, $s_x = (s_3, s_6, s_9, s_{13})$ and $s_y = (s_{17}, s_{20}, s_{25}, s_{27})$ in Example 2 (Figure 3).

Theorem 1: *Given a global adaptation model* Γ*,* Γ *is sound if* $\Gamma.M$*,* $\Gamma.M'$ *and* $\Gamma.ra$ *are sound.*

Proof. By definition of the soundness of a global adaptation model, proof of Theorem 1 is equivalent to prove that M_Γ is sound if $\Gamma.M$, $\Gamma.M'$ and $\Gamma.ra$ are sound. By Definition 5 and definition of the soundness of a lifecycle, M_Γ is sound if and only if Formulas (1) and (2) hold for the lifecycle of M_Γ, i.e.,

$$\forall r \in rule(L_{M\Gamma}), (s_x, r, s_y) \in L_{M\Gamma}.E_t \wedge (s_m, r, s_n) \notin L_{M\Gamma}.E_t, \quad (3)$$

$$\forall s \in L_{M\Gamma}.V_s \wedge \neg fi(s), \exists s_f, s_f \in L_{M\Gamma}.V_s \wedge fi(s_f) \wedge L_{M\Gamma}.s_0 \Rightarrow^* s \wedge s \Rightarrow^* s_f. \quad (4)$$

Therefore, Theorem 1 holds if we can confirm the following hypothesis: Formulas (3) and (4) hold if $\Gamma.M$, $\Gamma.M'$ and $\Gamma.ra$ are sound. This hypothesis can be divided as two parts: (I) Formula (3) holds if $\Gamma.M$, $\Gamma.M'$ and $\Gamma.ra$ are sound; and (II) Formula (4) holds if $\Gamma.M$, $\Gamma.M'$ and $\Gamma.ra$ are sound. Next, we prove these two parts separately. Suppose the lifecycle of $\Gamma.M$ is L_M; the lifecycle of $\Gamma.M'$ is $L_{M'}$; and $\Gamma.ra = (\lambda_M, \lambda_{M'}, \beta_M, \beta_{M'}, v_a)$. Note that the sequence of states of artifacts in a state of $L_{M\Gamma}$ may not be in lexicography order of artifact names, because we put states of artifacts of L_M before states of artifacts of $L_{M'}$ in a state of $L_{M\Gamma}$ for convenience of description.

Proof of Hypothesis Part (I). To prove Hypothesis Part (I), we divide transitions of $L_{M\Gamma}$ as three types. Rules appear in $L_{M\Gamma}$, i.e., $rule(L_{M\Gamma})$ can be divided as three sub sets, rules derive from $\Gamma.M.R$, marked as R_M; rules derive from $\Gamma.M'.R$, marked as $R_{M'}$ and $\{\Gamma.r_a\}$. According to construction process of the lifecycle of an artifact-centric business process model, there are three types of transitions in $L_{M\Gamma}$ that correspond to these three sub sets of rules. These transitions have the form of $(s_{m\_a} \circ s_{m'\_init}, r, s_{m\_b} \circ s_{m'\_init})$, $(s_{m\_init\_gf} \circ s_{m'\_a}, r', s_{m\_init\_gf} \circ s_{m'\_b})$ and $(s_{m\_a} \circ s_{m'\_init}, \Gamma.r_a, s_{m\_gf} \circ s_{m'\_b})$, respectively. s_{m_a} and s_{m_b} are two states of L_M and $s_{m'\_init}$ is the initial state $L_{M'}.s_0$. Notation $\circ$ represents concatenation operation. $s_{m\_init\_gf}$ is either the initial state or a state s_{m_gf} of L_M, where $s_{m\_gf}[i] = s_{gf}$ $(1 \le i \le |M.Z|)$. $s_{m'\_a}$ and $s_{m'\_b}$ are two states of $L_{M'}$. According to Definition 4 and the construction process of the lifecycle $L_{M\Gamma}$ (omitted in this paper), we have following two facts:

$$(s_{m\_a} \circ s_{m'\_init}, r, s_{m\_b} \circ s_{m'\_init}) \in L_{M\Gamma}.E_t \leftrightarrow (s_{m\_a}, r[s_{m\_a}, s_{m\_b}], s_{m\_b}) \in L_M.E_t, \quad (5)$$

$$(s_{m\_init\_gf} \circ s_{m'\_a}, r', s_{m\_init\_gf} \circ s_{m'\_b}) \in L_{M\Gamma}.E_t \leftrightarrow (s_{m\_a}, r'[s_{m\_a}, s_{m\_b}], s_{m\_b}) \in L_{M'}.E_t. \quad (6)$$

Since $\Gamma.M$ and $\Gamma.M'$ are sound, the following two formulas

$$\forall r \in R_M, (s_x, r, s_y) \in L_{M\Gamma}.E_t \wedge (s_m, r, s_n) \notin L_{M\Gamma}.E_t \quad (7)$$

$$\forall r \in R_{M'}, (s_x, r, s_y) \in L_{M\Gamma}.E_t \wedge (s_m, r, s_n) \notin L_{M\Gamma}.E_t \quad (8)$$

hold while considering Formulas (1), (5) and (6). Since $\Gamma.r_a$ is sound and transition ($s_{m\_x} \circ s_{m'\_init}$, $\Gamma.r_a$, $s_{m\_gf} \circ s_{m'\_y}$) is unique in $L_{M\Gamma}$, we have that

$$\forall r \in R_M, (s_x, \Gamma.r_a, s_y) \in L_{M\Gamma}.E_t \wedge (s_m, \Gamma.r_a, s_n) \notin L_{M\Gamma}.E_t. \quad (9)$$

s_{m_x} is an state of L_M according to $\Gamma.r_a.\lambda_M$, denoted as $prestate(\Gamma.r_a)$; $s_{m'\_y}$ is a state of $L_{M'}$ according to $\Gamma.r_a.\beta_{M'}$, denoted as $poststate(\Gamma.r_a)$. Here, we confirm Hypothesis Part (I) when transitions confine to the same type by Formulas (7)–(9). Next, we release this restriction. Since $\Gamma.M.Z \cap \Gamma.M'.Z = \emptyset$, we have $r_x \neq r_y$, $r_x \in \Gamma.M.R$ and $r_y \in \Gamma.M'.R$. Therefore, we can have $r \neq r'$, where r and r' belong to different rule sets R_M, $R_{M'}$ and $\{\Gamma.r_a\}$. Here, it is sufficient to show that Hypothesis Part (I) holds while considering Formulas (7)–(9). Proof of Hypothesis Part (I) is complete.

Proof of Hypothesis Part (II). To prove Hypothesis Part (II), we divide lifecycle $L_{M\Gamma}$ as two parts: lifecycle fragments that process instances are not adapted; and lifecycle fragments that process instances are adapted. Because M and M' are sound, by Formulas (2), (5) and (6), we can have that

$$\forall s_{m\_a} \circ s_{m'\_init} \in L_{M\Gamma}.V_s \wedge \neg fi(s_{m\_a} \circ s_{m'\_init}), \exists s_{m\_sf} \circ s_{m'\_init}, s_{m\_sf} \circ s_{m'\_init} \in L_{M\Gamma}.V_s \wedge fi(s_{m\_sf} \circ s_{m'\_init}) \wedge L_{M\Gamma}.s_0 \Rightarrow^* s_{m\_a} \circ s_{m'\_init} \wedge s_{m\_a} \circ s_{m'\_init} \Rightarrow^* s_{m\_sf} \circ s_{m'\_init} \quad (10)$$

and

$$\forall s_{m\_init} \circ s_{m'\_a} \in L_{M\Gamma}.V_s \wedge \neg fi(s_{m\_init} \circ s_{m'\_a}), \exists s_{m\_init} \circ s_{m'\_sf}, s_{m\_init} \circ s_{m'\_sf} \in L_{M\Gamma}.V_s \wedge fi(s_{m\_init} \circ s_{m'\_sf}) \wedge L_{M\Gamma}.s_0 \Rightarrow^* s_{m\_init} \circ s_{m'\_a} \wedge s_{m\_init} \circ s_{m'\_a} \Rightarrow^* s_{m\_init} \circ s_{m'\_sf}. \quad (11)$$

$s_{m\_init} = L_M.s_0$. $s_{m'\_sf}$ and s_{m_sf} are final states in $L_{M'}$ and L_M, respectively. Formulas (10) and (11) depict the lifecycle fragments that process instances are not adapted. Note that $s_{m\_sf} \circ s_{m'\_init}$ and $s_{m\_init} \circ s_{m'\_sf}$ are seen as final states in $L_{M\Gamma}$ for the fact that $s_{m\_sf} \circ s_{m'\_init}$ and $s_{m\_init} \circ s_{m'\_sf}$ are business goals of business process instances that are not adapted. Therefore, both $fi(s_{m\_sf} \circ s_{m'\_init})$ and $fi(s_{m\_init} \circ s_{m'\_sf})$ return true in Formulas (10) and (11). Next, we discuss the lifecycle fragment that business process instances are adapted. Because M' and $\Gamma.r_a$ are sound, by Formulas (2) and (6), we can have that:

$$\forall s_{m\_gf} \circ s_{m'\_c} \in L_{M\Gamma}.V_s \wedge \neg fi(s_{m\_gf} \circ s_{m'\_c}), \exists s_{m\_gf} \circ s_{m'\_sf}, s_{m\_gf} \circ s_{m'\_sf} \in L_{M\Gamma}.V_s \wedge fi(s_{m\_gf} \circ s_{m'\_sf}) \wedge s_{m\_gf} \circ s_{m'\_y} \Rightarrow^* s_{m\_gf} \circ s_{m'\_c} \wedge s_{m\_gf} \circ s_{m'\_c} \Rightarrow^* s_{m\_gf} \circ s_{m'\_sf}. \quad (12)$$

Because M is sound, by Formulas (2) and (5), we can have that $L_{M\Gamma}.s_0 \Rightarrow^* s_{m\_x} \circ s_{m'\_init}$. As ($s_{m\_x} \circ s_{m'\_init}$, $\Gamma.r_a$, $s_{m\_gf} \circ s_{m'\_y}$) and $s_{m\_gf} \circ s_{m'\_y} \Rightarrow^* s_{m\_gf} \circ s_{m'\_c}$, we can have that $L_{M\Gamma}.s_0 \Rightarrow^* s_{m\_gf} \circ s_{m'\_c}$. Since $L_{M\Gamma}.s_0 \Rightarrow^* s_{m\_gf} \circ s_{m'\_c}$ and Formula (12) hold, it is sufficient to show that following formula

$$\forall s_{m\_gf} \circ s_{m'\_c} \in L_{M\Gamma}.V_s \wedge \neg fi(s_{m\_gf} \circ s_{m'\_c}), \exists s_{m\_gf} \circ s_{m'\_sf}, s_{m\_gf} \circ s_{m'\_sf} \in L_{M\Gamma}.V_s \wedge fi(s_{m\_gf} \circ s_{m'\_sf}) \wedge L_{M\Gamma}.s_0 \Rightarrow^* s_{m\_gf} \circ s_{m'\_c} \wedge s_{m\_gf} \circ s_{m'\_c} \Rightarrow^* s_{m\_gf} \circ s_{m'\_sf} \quad (13)$$

holds. By Formulas (10), (11) and (13), it is sufficient to show that Hypothesis Part (II) holds. The proof of Theorem 1 is complete. □

It is known that the evolution of a business process instance is nondeterministic due to the flexibility and declarative properties of artifact-centric business processes [25]. A business process instance will reach a completely different final state even with the same business data. Furthermore, there exists unexpected business goals of M'. Therefore, we must guarantee that a business process instance that is adapted by an adaptation rule can reach an acceptable business goal, i.e., business goal that are acceptable for a customer. We provide the correctness of an adaptation rule to guarantee this constraint.

Before presenting the correctness of an adaptation rule, we provide a definition of an adaptation goal, which is used to define the acceptable business goals. An adaptation goal is defined by domain experts.

Definition 7. *Given a global adaptation model Γ, an adaptation goal of business process instances that are adapted by $\Gamma.r_a$ is a set of final states $S_{bg} = \{s_1, \ldots, s_n\}$, where s_i $(1 \leq i \leq n)$ is a final state of the lifecycle of $\Gamma.M'$.*

Definition 8. *Given a global adaptation model Γ and an adaptation goal S_{bg} of business process instances that are adapted by $\Gamma.r_a$, $\Gamma.r_a$ is correct if reach*(Γ) ⊆ S_{bg}.

In Definition 8, *reach*(Γ) returns a set of reachable final states from state s_{rp} in the lifecycle of $\Gamma.M'$, where s_{rp} = *poststate*($\Gamma.r_a$). For example, *reach*(Γ) = {(s_{19}, s_{23}, s_{26}, s_{30})}, s_{rp} = *poststate*(r_{21}) = (s_{17}, s_{20}, s_{25}, s_{27}) in Example 2 (Figure 3).

We provide an algorithm *iscorrect*() to check the correctness of an adaptation rule. In the algorithm, we suppose $\Gamma.M$, $\Gamma.M'$ and $\Gamma.r_a$ are sound. The algorithm is described as Algorithm 1.

Algorithm 1. Correctness checking algorithm

Input: a global adaptation model Γ and an adaptation goal S_{bg}.
Output: true or false.

```
0.  Begin
1.      s_rp ← poststate(Γ.ra)
2.      q.add(s_rp), S_f ← null, S ← S ∪ {s_rp}
3.      While q is not null
4.          s_x ← q.remove()
5.          M' ←Γ.M'
6.          For each (s_a, r, s_b) in L_M'.E_t
7.              If s_a = s_x and s_b not in S
8.                  q.add(s_b), S ← S ∪ {s_b}
9.                  If fi(s_b)
10.                     S_f ← S_f ∪ {s_b}
11.                 End
12.             End
13.         End
14.     End
15.     If S_f ⊆ S_f
16.         Return true
17.     Return False
18. End
```

In the algorithm, q is a queue. Lines 3–14 compute the set of reachable final states S_f from state s_{rp} in $L_{M'}$ transitively, i.e.,

$$\forall\, s_{m'\_sf} \in S_f,\ s_{rp} \Rightarrow^* s_{m'\_sf} \wedge fi(s_{m'\_sf}).$$

Since $\Gamma.M$, $\Gamma.M'$ and $\Gamma.r_a$ are sound, by Theorem 1, we have Γ is sound. From semantics of function *reach*(), we can see that S_f = *reach*(Γ). By Definition 8 and Lines 15–17, Algorithm 1 is correct. Let $N = max(|S|)$ and $K = |L_{M'}.E_t|$, the time complexity of Algorithm 1 is O(NK).

In Theorem 2, there is a function *adaptgoal*(). *adaptgoal*($\Gamma.r_a$) returns the adaptation goal of business process instances that are adapted by $\Gamma.r_a$.

Theorem 2. *Given a global adaptation model Γ, if Γ is sound and $\Gamma.r_a$ is correct, a business process instance that is adapted by $\Gamma.r_a$ can reach one business goal s of $L_{M'}$, where s ∈ adaptgoal*($\Gamma.r_a$).

Proof. Since Γ is sound, we have a set of reachable final states by *reach*(Γ). By Formulas (6) and (13), we have that state $s_{m\_gf} \circ s$ ($s \in$ *reach*(Γ)) can be reached from $s_{m\_gf} \circ$ *poststate*($\Gamma.r_a$) in $L_{M\Gamma}$. Reversely, by Formulas (13) and (6), we have that a process instance of $\Gamma.M$ that is adapted to $\Gamma.M'$ by $\Gamma.r_a$ can reach a final state s' of $L_{M'}$, where s' must be in *reach*(Γ). Since $\Gamma.r_a$ is correct, it is obvious that *reach*(Γ) ⊆ *adaptgoal*($\Gamma.r_a$) by Definition 8. Therefore, $s' \in$ *adaptgoal*($\Gamma.r_a$). The proof of Theorem 2 is complete. □

Example 4.

In this example, we use aforementioned purchase business process to illustrate our proposed global adaptation model. Suppose the purchase business process is improved by providing options

for a customer to pick up an order by him/herself or deliver the order by an express. The old model of the purchase business process is marked as M_P (Figure 1) and the new model of the purchase business process is marked as $M_{P'}$ (Figure 5).

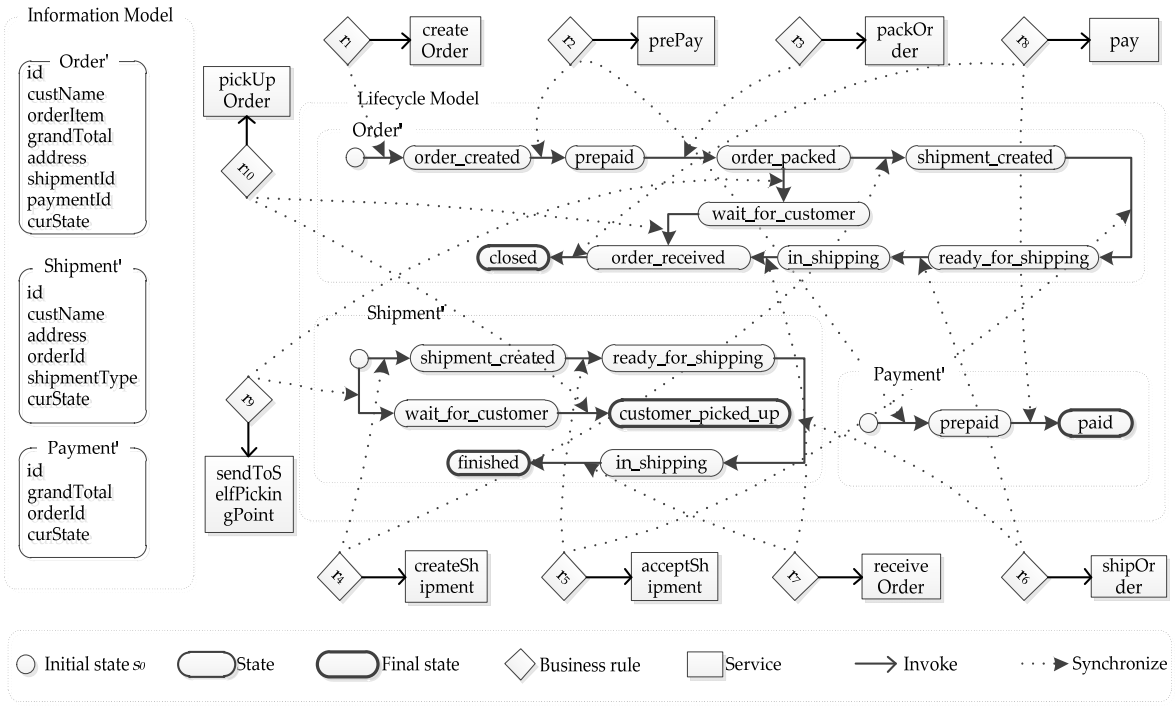

Figure 5. The purchase business process model $M_{P'}$.

Compare with M_P, a new attribute *shipmentType* is added in artifact Shipment'; two business rules r_9 and r_{10} are added in $M_{P'}$; and two states are added to artifacts Order' and Shipment'. After the new model is deployed, a business process instance can be adapted from M_P to $M_{P'}$ by our proposed adaptation rule. Here, we only discuss one circumstance. That is, a customer selects to change the type of shipment to self-pick-up and the order is at state *shipment_created*. Note that, after the new model has been deployed, there is a user interface for a customer to change the type of shipment, i.e., self-pick-up or express, if an order has not reached state *ready_for_shipping*. To implement the adaptation in this circumstance, we define an adaptation rule in Table 3.

Table 3. The adaptation rule r_a that is used for the adaptation.

r_a: Adapt a purchase process instance of M_P to $M_{P'}$, where the state of the order is *shipment_created* and customer changed the type of the shipment to self-pick-up.	
λ_M	*instate*(*Order*, *shipment_created*) ∧ *instate*(*Payment*, *prepaid*) ∧ *instate*(*Shipment*, *shipment_created*) ∧ *shipmentType* = 1
$\lambda_{M'}$	*instate*(*Order'*, *init*) ∧ *instate*(*Payment'*, *init*) ∧ *instate*(*Shipment'*, *init*)
v_a	*adapt*(*Order o*, *Payment p*, *Shipment s*, *Order' o'*, *Payment' p'*, *Shipment' s'*, *Integer shipmentType*)
β_M	*instate*(*Order*, s_{gf}) ∧ *instate*(*Payment*, s_{gf}) ∧ *instate*(*Shipment*, s_{gf})
$\beta_{M'}$	*instate*(*Order'*, *wait_for_customer*) ∧ *instate*(*Payment'*, *prepaid*) ∧ *instate*(*Shipment'*, *wait_for_customer*)

In Table 3, the type of a shipment is self-pick-up if *shipmentType* equals to 1. According to the definition of the adaptation rule, we can have that *prestate*(r_a) = (*shipment_created*, *prepaid*, *shipment_created*) and *poststate*(r_a) = (*wait_for_customer*, *prepaid*, *wait_for_customer*). According to L_{M_P} (Figure 4), state (*shipment_created*, *prepaid*, *shipment_created*) is reachable in L_{M_P}. Similarly, state (*wait_for_customer*, *prepaid*, *wait_for_customer*) is reachable in $L_{M_{P'}}$. As such, r_a is sound by

Definition 6. Therefore, all purchase process instances that are adapted by the adaption rule can reach some business goals of $L_{M_{P'}}$.

In the proposed circumstance, a customer selects to pick up the order by him/herself, such that the order should be closed after the order is picked up and paid by the customer, i.e., the adaptation goal for a business process instance that is adapted by r_a is $adaptgoal(r_a)$ = {(*closed*, *paid*, *customer_picked_up*)}. The state of a purchase process instance is $poststate(r_a)$ after r_a is evaluated. According to Lines 3–14 of Algorithm 1 and $L_{M_{P'}}$, we can have that the set of reachable states from state (*wait_for_customer*, *prepaid*, *wait_for_customer*) in $L_{M_{P'}}$ is $reach(\Gamma)$ = {(*closed*, *paid*, *customer_picked_up*)}, where $\Gamma = (M_P, M_{P'}, r_a.v_a, r_a)$. Because $reach(\Gamma) \subseteq adaptgoal(r_a)$, r_a is correct by Definition 8. By Theorem 2, a purchase process instance of M_P that is adapted by r_a can finally reach an acceptable business goal that belongs to $adaptgoal(r_a)$. In this example, there is only one business goal (*closed*, *paid*, *customer_picked_up*), which is expected by customers.

5. Tool Prototype

From Example 4, we can see that it is relatively easy for a process designer to define an adaptation rule and an adaptation goal when implementing the adaptation of a model of an artifact-centric business process instance. However, it is a hard work for process designer to judge whether an adaptation rule is sound and correct. To solve this, we developed a tool prototype for helping a process designer to define a sound and correct adaptation rule.

In the tool, we implemented a function to compute the lifecycle of a process model according to lifecycles of artifacts of the process model. We also implemented a function to judge whether the state that corresponds to a condition of an adaptation rule is reachable. Based on these two functions, we implemented Algorithm 1 as one of the main features of the tool. The tool was developed based on Java 1.8, Java Universal Network/Graph Framework (JUNG) 2.1.1 and BeautyEye 3.7 [26,27]. Figure 6 demonstrates the screenshot of verification result of Example 4. Note that names of artifacts of M_P end with "*A*" and names of artifacts of $M_{P'}$ end with "*B*".

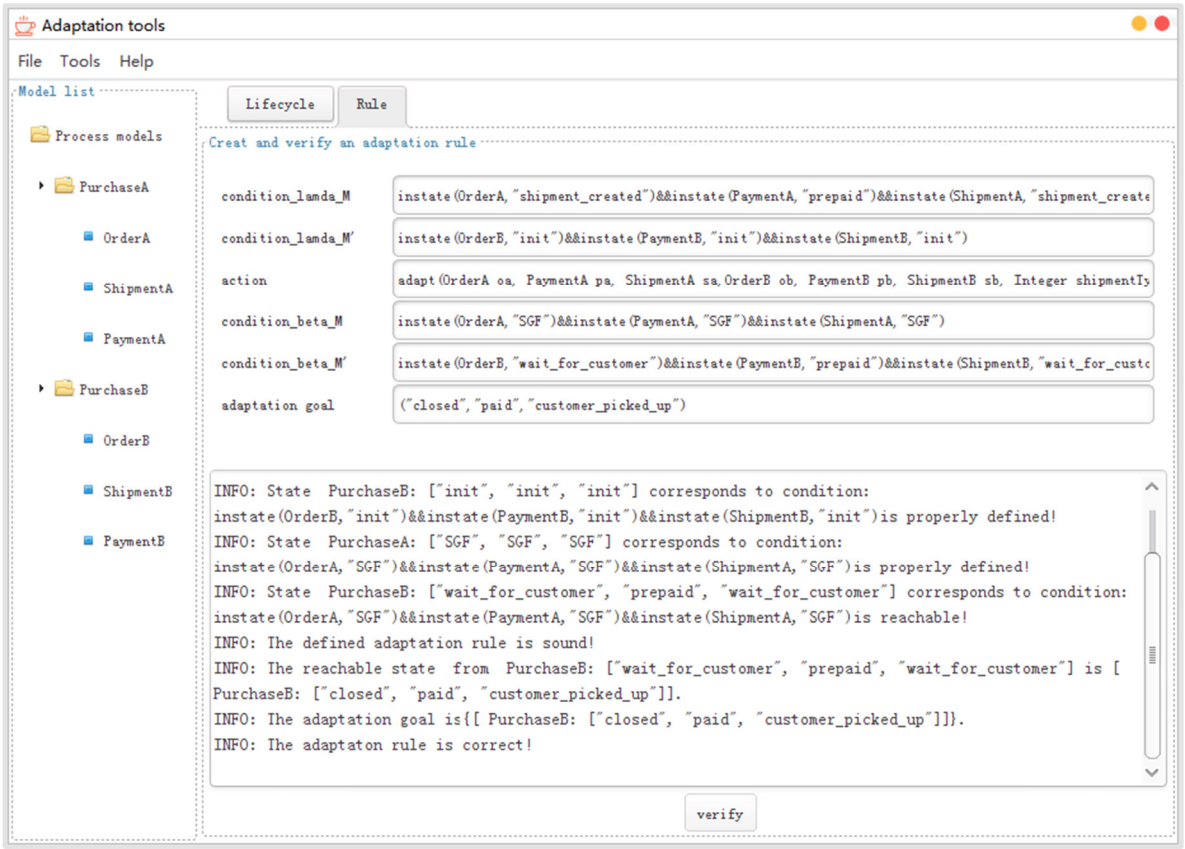

Figure 6. The screenshot of the verification results of Example 4 through the tool.

A process designer can import two or more artifact-centric business process models through File menu of the tool, where these models are stored as APMT (Artifact-Centric Process Model based on Label Transition System) document [5]. An APMT document, which is based on XML format, captures the behavior of an artifact-centric business process based on the definitions of lifecycles of artifacts and their synchronization. A process designer can click on the name of an artifact on the left of the main panel to check the lifecycle of the artifact, where the lifecycle is presented in the tabbed panel that is named as "Lifecycle".

A process designer can define an adaptation rule through Tools menu or click on the title of the tabbed panel that is named as "Rule". The defined adaptation rule and these imported business process models are used as input of the tool. Verification results for the defined adaptation rule are presented at the bottom of the tabbed panel that is named as "Rule".

6. Discussions

In this paper, we propose an adaptation rule for adapting a model of an artifact-centric business process instance dynamically and use a global adaptation model to describe the adaptation. We prove that a global adaptation model is sound if the adaptation rule and the two artifact-centric business process models in the global adaptation model are sound. Business process instances that are adapted by the adaptation rule can reach some acceptable business goals if the adaptation rule is correct and the global adaptation model is sound. During the adaptation, both changes of data (artifact information model) and process (lifecycles of artifacts) are considered.

Although we discuss the adaptation of a model of an artifact-centric business process instance theoretically, our theoretical results can be used directly to support the adaptation of business process instances in the execution level if business processes are running on the framework proposed in [25]. Because they claim that the framework supports execution of artifact-centric business process model (ACP model) directly without model transformation. Furthermore, our proposed global adaptation model can be seen as an artifact-centric business process model. Comparison of the framework with other works can be found in [25].

In this paper, our proposed adaptation rule is used to adapt a model of a business process instance. From another perspective, our adaptation rule can also be used to combine lifecycle fragments of different business process models to construct complex business processes, because a global adaptation model can be seen as an equivalent artifact-centric business process model according to its definition. From this perspective, artifact-centric business process model is improved with the ability of modeling sub business processes, which, to our best knowledge, has not been addressed in the artifact-centric business process community.

However, in a global adaptation model, there is only one adaptation rule. In real-world, multiple adaptation rules should be defined to adapt business process instances of different circumstances. Moreover, transitive adaptation, i.e., a process instance is adapted more than one time, is not discussed. Nevertheless, the same results can also be derived when multiple adaptation rules are defined in one global adaptation model. Because a global adaptation model with one adaptation rule can be seen as an ordinary artifact-centric business process model, such that the soundness of the global adaptation model can be proved recursively. Furthermore, the adaptation goal of business process instances that are adapted by an adaptation rule is restricted locally to the adaptation rule, such that the correctness of an adaptation rule is independent of other adaptation rules.

There are also some limitations of our proposed method. First, an adaptation rule and the adaptation goal of business process instances that are adapted by the adaptation rule are defined by human. Second, our proposed method is closed related to ACP model [25]. There are some other similar artifact-centric business process modeling approaches, such as GSM and CMMN [3,9,10], where a milestone (state in ACP model) is reached if a guard (a condition for opening a stage) of a stage (action in ACP model) is satisfied and the stage is executed and condition of the sentry that corresponds to the milestone is satisfied. Comparing ACP model with GSM and CMMN models, they all depict artifact behaviors by lifecycles of artifacts. Differently, a lifecycle of an artifact in ACP model is driven by Condition–Action business rules, and a lifecycle of an artifact in GSM and

CMMN models is driven by variants of Event–Condition–Action rules. We try to adapt our proposed method to apply to GSM and CMMN models.

7. Conclusions

In this paper, we propose an adaptation rule for adapting a model of an artifact-centric business process instance and describe the adaptation by a global adaptation model. We prove that the business process instance can reach an acceptable business goal without deadlock if the adaptation rule is sound and correct. We developed a tool prototype for helping a process designer to define a sound and correct adaptation rule. Possible applications and limitations of our proposed method are discussed in the Discussion Section. Our future work will try to adapt our proposed approach to other models, like GSM and CMMN.

Author Contributions: Conceptualization, J.Z.; Methodology, J.Z.; Software, J.Z.; Writing-Original Draft Preparation, J.Z.; Writing-Review & Editing, J.Z. and G.L.; Supervision, G.L.

Funding: This research received no external funding.

Acknowledgments: The authors gratefully acknowledge the editors and the anonymous reviewers for their detailed and constructive criticisms, which have helped us to improve the quality and presentation of our manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rinderle, S.; Reichert, M.; Dadam, P. Correctness criteria for dynamic changes in workflow systems—A survey. *Data Knowl. Eng.* **2004**, *50*, 9–34.
2. Nunes, V.T.; Santoro, F.M.; Werner, C.M.; Ralha, C.G. Real-time process adaptation: A context-aware replanning approach. *IEEE Trans. Syst. Man Cybernetics Syst.* **2017**, *48*, 99–118.
3. Cohn, D.; Hull, R. Business artifacts: A data-centric approach to modeling business operations and processes. *IEEE Data Eng. Bull.* **2009**, *32*, 3–9.
4. Kunchala, J.; Yu, J.; Yongchareon, S. A survey on approaches to modeling artifact-centric business processes. In Proceedings of the International Conference on Web Information Systems Engineering Workshops (WISE), Thessaloniki, Greece, 12–14 October 2014; Volume 9051, pp. 117–132.
5. Yongchareon, S.; Liu, C.F.; Yu, J. Zhao, X.H. A view framework for modeling and change validation of artifact-centric inter-organizational business processes. *Inf. Syst.* **2015**, *47*, 51–81.
6. Wang, Y.; Wang, Y. Change analysis for artifact-centric business processes. In Proceedings of the International Conference on Business Information Systems (BIS), Larnaca, Cyprus, 22–23 May 2014, Volume 176, pp. 98–109.
7. Li, S.; Yu, S.J. Research on change adaptation technology of data-centric business process models. *Intell. Comput. Appl.* **2016**, *6*, 36–43.
8. Eshuis, R.; Hull, R.; Yi, M.F. Property preservation in adaptive case management. In Proceedings of the International Conference on Service-Oriented Computing (ICSOC), Goa, India, 16–19 November 2015; Volume 9435, pp. 285–302.
9. Damaggio, E.; Hull, R.; Vaculín, R. On the equivalence of incremental and fixpoint semantics for business artifacts with Guard-Stage-Milestone lifecycles. In Proceedings of the International Conference on Business Process Management (BPM), Clermont-Ferrand, France, 30 August–2 September 2011; Volume 6896, pp. 396–412.
10. About the Case Management Model and Notation specification version 1.1. Available online: https://www.omg.org/spec/cmmn/1.1/ (accessed on 15 October 2018).
11. Cognini, R.; Corradini, F.; Gnesi, S.; Polini, A.; Re, B. Business process flexibility—A systematic literature review with a software systems perspective. *Inf. Syst. Front.* **2016**, *20*, 343–371.
12. Song, W.; Jacobsen, H.A. Static and dynamic process change. *IEEE Trans. Serv. Comput.* **2018**, *11*, 215–231.
13. Rosa, M.L.; Aalast, W.M.; Dumas, M.; Milani, F.P. Business process variability modeling: A survey. *ACM Comput. Surv.* **2017**, *50*, 1–45.

14. Vasilecas, O.; Kalibatiene, D.; Lavbič, D. Rule- and context-based dynamic business process modelling and simulation. *J. Syst. Softw.* **2016**, *122*, 1–15.
15. Hu, G.C.; Wu, B.D.; Chen, J.L. Dynamic adaptation of business process based on context changes: A rule-oriented approach. In Proceedings of the International Conference on Service-Oriented Computing Workshops (ICSOC), Berlin, Germany, 2–5 December 2013; Volume 8377, pp. 492–504.
16. Ayora, C.; Torres, V.; Weber, B.; Reichert, M.; Pelechano, V. Enhancing modeling and change support for process families through change patterns. In Proceedings of the International Workshop on Business Process Modeling, Development and Support (BPMDS), Valencia, Spain, 17–18 June 2013; Volume 147, pp. 246–260.
17. Ayora, C.; Torres, V.; Vara, J.L.; Pelechanoa, V. Variability management in process families through change patterns. *Inf. Softw. Technol.* **2016**, *74*, 86–104.
18. Patiniotakis, I.; Apostolou, D.; Verginadis, Y.; Papageorgiou, N.; Mentzas, G. Assessing flexibility in event-driven process adaptation. *Inf. Syst.* **2017**, 1–19, doi:10.1016/j.is.2017.10.009.
19. Oberhauser, R. Adapting processes via adaptation processes: A flexible and cloud-capable adaptation approach for dynamic business process management. In Proceedings of the International Symposium on Business Modeling and Software Design (BMSD), Milan, Italy, 6–8 July 2015; pp. 9–18.
20. Bhattacharya, K.; Gerede, C.; Hull, R.; Liu, R.; Su, J.W. Towards formal analysis of artifact-centric business process models. In Proceedings of the International Conference on Business Process Management (BPM), Brisbane, Australia, 24–28 September 2007; Volume 4714, pp. 24–28.
21. Lei, J.K.; Bai, R.F.; Guo, L.P.; Zhang, L. Towards a scalable framework for artifact-centric business process management systems. In Proceedings of International Conference on Web Information Systems Engineering (WISE), Shanghai, China, 8–10 November 2016; Volume 10042, pp. 309–323.
22. Yongchareon, S.; Liu, C.F. A process view framework for artifact-centric business processes. In Proceedings of the International Conference on "On the Move to Meaningful Internet Systems" (OTM), Crete, Greece, 25–29 October 2010; Volume 6426, pp. 26–43.
23. Nigam, A. Business artifacts: An approach to operational specification. *IBM Syst. J.* **2003**, *42*, 428–445.
24. Gerede, C.E.; Bhattacharya, K.; Su, J.W. Static analysis of business artifact-centric operational models. In Proceedings of IEEE International Conference on Service-oriented Computing & Applications (SOCA), Newport Beach, CA, USA ,19–20 June 2007; pp. 133–140.
25. Ngamakeur, K.; Yongchareon, S.; Liu, C.F. A framework for realizing artifact-centric business processes in service-oriented architecture. In Proceedings of International Conference on Database Systems for Advanced Applications (DASFAA), Busan, South Korea, 15–19 April 2012; Volume 7238, pp. 63–78.
26. Java Universal Network/Graph Framework version 2.1.1. Available online: https://github.com/jrtom/jung/releases/tag/jung-2.1.1 (accessed on 11 July 2018).
27. BeautyEye version 3.7. Available online: https://github.com/JackJiang2011/beautyeye (accessed on 11 July 2018).

© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).