On Formalizing Inter-process Relationships

Tri A. Kurniawan*, Aditya K. Ghose, Lam-Son Lê, and Hoa Khanh Dam

Decision Systems Lab., School of Computer Science and Software Engineering,
University of Wollongong, NSW 2522, Australia
{tak976,aditya,lle,hoa}@uow.edu.au

Abstract. Most medium to large organizations support large collections of process designs, often stored in business process repositories. These processes are often inter-dependent. Managing such large collections of processes is not a trivial task. We argue that formalizing and establishing inter-process relationships play a critical role in that task leading to a machinery approach in the process repository management. We consider and propose three kinds of such relationships, namely *part-whole*, *inter-operation* and *generalization-specialization*, including their formal definitions, permitting us to develop a machinery approach. Analysis of the relationships relies on the semantically effects annotated process model in BPMN. This paper presents a rigorous approach to assist the designer to establish inter-process relationships in a process repository.

Keywords: business process, semantic effect annotation, process relationship.

1 Introduction

Most medium to large organizations support large collections of process designs modeled through many business process modeling languages such as Business Process Model and Notation (BPMN)[1], often stored in *business process repositories*. These are typically characterized by the following features. *First*, the number, scale and complexity of the processes are large, i.e. consisting hundreds or even thousands of business process models. For example, the SAP R/3 reference model contains 600+ process models and Suncorp's repository contains 6,000+ process models [4]. *Second*, most of these processes are inter-dependent (both in terms of design and execution). Some evidences of such dependency have been discussed in [2]. We can also found some dependencies among processes as shown in the MIT Process Handbook[2] [9], Map of Medicine[3] and the published literature (see, for example, the clinical process repository described in [1]). *Third*, changes to any one process are likely to impact several other processes. Approaches to analyze the impact of process changes, depending upon

* On leave from a lecturership at University of Brawijaya, East Java, Indonesia.
[1] BPMN homepage `http://www.bpmn.org/`
[2] MIT Process Handbook homepage `http://process.mit.edu/`
[3] Map of Medicine homepage `http://www.mapofmedicine.com/`

F. Daniel et al. (Eds.): BPM 2011 Workshops, Part II, LNBIP 100, pp. 75–86, 2012.

type of process dependency, have been described in [2]. Finally, some process designs exist to realize component functionalities of other process designs.

Dealing with such complex process repositories is not a trivial task. Due to this complexity, many issues come up along each process's life-cycle such as managing process variants [5], maintaining relationship consistency among inter-dependent processes due to any process change drivers [10](problem in process optimization, for example, introduced in [8]), performing process impact analysis [2] if changes applied to any process, finding a particular process in which the other processes depend on or extracting the structure of a process repository. We argue that formalizing and establishing process relationships play a critical role for building a machinery approach in the process repository management.

This paper makes three key contributions. *First*, we propose a taxonomy of inter-process relationships and provide formal definitions for each of them. We leverage semantically annotated process models, in the sense of [6] (or more loosely [13]). This allows us to perform deeper semantic analysis in establishing and checking these relationships than would be possible with simple (un-annotated) process models. *Second*, as the application of establishing such relationships, we outline a procedure for resolving relationship violations, in instance of one relationship type (similar procedures can be defined for other relationship types in our taxonomy, but are omitted due to space constraints). *Third*, for further such application, we show that the relationship types lead to partial orders, permitting us to structure the process repository in terms of *process lattices*. The process lattice view permits a range of formal analysis to support the identification and maintenance of inter-process relationships in a process repository including advanced process queries. We plan to further elaborate the aforementioned applications of process relationships establishment for our future work. In this paper, we only focus on presenting a novel approach for formally establishing relationships between processes modeled in BPMN. Relationship analysis will be performed based on the semantically effects annotated process model [6,7].

The remainder of the paper is organized as follows. Section 2 introduces semantic effect annotations for business process models as the basis for further formal definitions. Section 3 describes and formalizes all relationships between process models. Section 4 briefly surveys the related work. Finally, Section 5 draws some conclusions and outlines our future work.

2 Preliminaries

Koliadis and Ghose [7] discussed the concept of semantic effects. An effect annotation relates a specific result or outcome to an activity on a business process model. An activity represents the work performed within a business process. Activities are either atomic (called as *task* i.e. they are at the lowest level of detail presented in the diagram and can not be further broken down) or compound (called as *sub-process* i.e. they are decomposable to see another level of process below) [14]. In an annotated BPMN model, every activity has been annotated with its (immediate) effects. For a complete process, we also define a cumulative

effect annotation which is obtained from accumulating the immediate effects of all annotated activities based on all alternative paths (due to XOR gateways) to reach an activity being observed.

We shall leverage the ProcessSEER [6] approach to semantic effect annotation. This framework permits us to determine, at design time, the answer to the following question that can be posed for any point in the process design: what would the effects of the process be if it were to execute up to this point? The answer is necessarily non-deterministic, since a process might have taken one of many possible alternative paths through a process design to get to that point. The non-determinism also arises from the fact that the effects of certain process steps might undo the effects of prior steps - the inconsistencies that result in the snapshot of the domain that we seek to maintain might be resolved in multiple alternative ways (a large body of work in the reasoning about action community addresses this problem). The answer to the question is therefore provided via a set of effect scenarios, any one of which might eventuate in a process instance. The approach simplifies the activity of semantic effect annotation by only requiring that activities (populating a capability library) be annotated with context-independent immediate effects. The tool then contextualizes these effects by propagating them through a process model (specified in BPMN in the current instance) to determine the cumulative effect scenarios at the end of each activity. It uses formal machinery (theorem-provers) to compute cumulative effects, but provides an analyst-friendly Controlled Natural Language (CNL) interface, coupled with a domain ontology, that permits the immediate effects of activities to be specified in natural language (but with a restricted set of sentence formats). The use of CNL permits us to translate these natural language specifications into underlying formal representation, which in turn makes the use of theorem-provers possible. In addition, the tool also makes provision for local (activity-specific) non-functional annotations to be propagated through a process design, so that we are able to determine the cumulative non-functional scenarios for each activity in a process design as well.

3 Inter-process Relationships

There are three main concepts to be described. *First*, the taxonomy of inter-process relationships will be identified and formalized. *Second*, we discuss partly (only takes part-whole relationship) the idea of resolving inconsistencies in inter-process relationships due to any changes on a particular process. *Third*, the idea of leveraging lattice theory in constructing process lattices based upon process relationships will be formalized. The last two concepts are derived from taking the advantages of formalizing inter-process relationships.

3.1 Relationships Taxonomy

We now propose a taxonomy of relationships that can be established between different processes which are classified into two categories: *functional dependencies* and *consistency links*. A functional dependency exists between a pair of

processes when one process depends on the other for realizing some of its functionalities. In other words, a process will not be able to achieve its goals without the support given by the others. In contrast, a consistency link exists between a pair of processes when both of them have intersecting parts represent the same functionality, i.e. the outcomes (e.g. effects) of these parts are exactly the same. They are functionally independent, i.e. one process is not supported by the other.

In such categories, we now define the three different types of relationship that can exist between processes, namely *part-whole*, *inter-operation*, and *generalization-specialization*. The first two fall in the functional dependencies category whereas the third is regarded as a consistency link. We formally define each of these relationship types using the semantic effect analysis on process models. We use $acc\,(P)$ to denote the cumulative end effects of process P; $CE\,(P, t_i)$ to describe cumulative effect at the point of activity t_i within process P; and es_j to denote an effect scenario *j-th*. It is noted that each of $acc\,(P)$ or $CE\,(P, t_i)$ is a set of effect scenarios. Each effect scenario is represented as a set of clauses and will be viewed, implicitly, as their conjunction.

Part-whole

Part-whole relationship exists between two processes when one process is required by the other process to fulfill some of its functionalities. More specifically, there must be an activity in the "whole" process representing the functionalities of the "part" process. The "part" process is also commonly referred to as a sub-process within the "whole" process. Intuitively, there is an insertion of the functionalities of the "part" into the "whole". We first define the insertion of a process in another process.

Definition 1. *The insertion of process $P2$ in process $P1$ at activity t, $P1 \uparrow^t P2$, is a process design obtained by viewing $P2$ as the sub-process expansion of activity t in $P1$.*

Literally, the insertion of $P2$ at an activity t in $P1$ simply involves connecting the path entering t with the starting event of $P2$ and connecting the path leaving t with the end event of $P2$. Semantic effects can be applied to in this situation as follows. Let $T1 = \{t_{11}, t_{12}, \ldots, t_{1i}\}$ and $T2 = \{t_{21}, t_{22}, \ldots, t_{2j}\}$ be the set of consecutive activities of process models $P1$ and $P2$ respectively. Let $CE\,(P1, t_{1s})$ be the cumulative effects of process model $P1$ at the point of activity t_{1s} where $1 \leq s \leq i$. Cumulative effects computation involves a left-to-right pass of evaluating the activities within a process until the defined point of activity t_{1s}. Then, $CE\,(P1 \uparrow^{t_{1s}} P2, t_{1s})$ would be computed by replacing activity $t_{1s} \in T1$ with a set of activities within $P2$ through the following procedures: (1) accumulate the effects from activity t_{11} until activities t_{1s-1} within $P1$, where t_{1s-1} denotes all activities immediately precede activity t_{1s}, might be in parallel; (2) continue the effects accumulation involving all activities within $P2$ through passing from the most left activity t_{21} to the most right one t_{2j}; (3) continue the accumulation through t_{1s+1} until t_{1i} within $P1$, where t_{1s+1} denotes all activities immediately succeed activity t_{1s}.

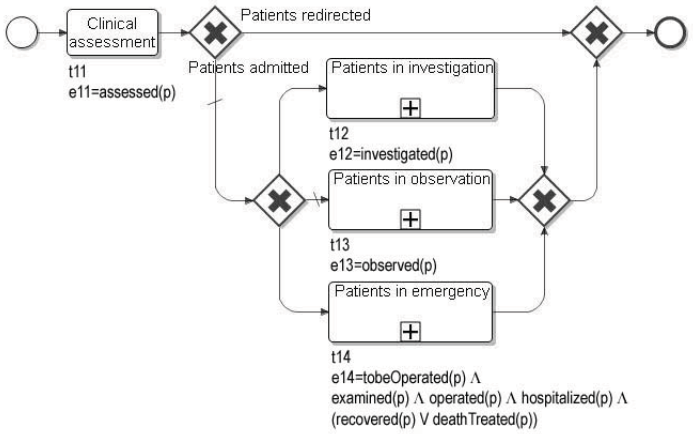

Fig. 1. BPMN model of *Management of patients on arrival* process, also showing the immediate effects e_i of each activity t_i

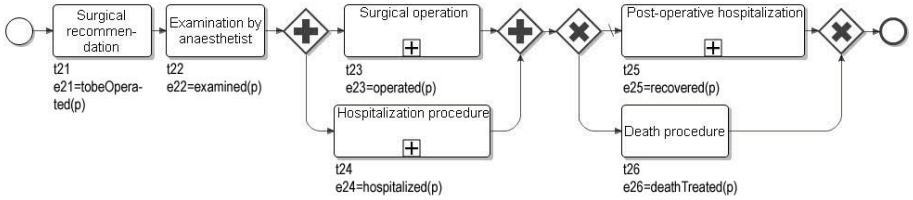

Fig. 2. BPMN model of *Patients in emergency* process. This is the sub-process expansion of the process in Fig. 1, also containing the immediate effects e_i of each activity t_i

Using the definition of process insertion, we formally define the part-whole relationship as Definition 2.

Definition 2. *Given process models* $P1$ *and* $P2$, $P2$ *is a direct part of* $P1$ *iff there exists an activity* t *in* $P1$ *such that* $CE\,(P1, t) = CE\,(P1 \uparrow^t P2, t)$. *If there is no insertion point at any activity* t *in* $P1$, *then* $P2$ *is an indirect part of* $P1$ *iff* $\forall es_i \in acc(P2)$, $\exists es_j \in CE\,(P1, t)$ *for any activity* t *in* $P1$ *such that* $es_j \models es_i$.

Let us consider an example of part-whole process relationship adopted from [1]. We transformed it, from originally represented in EPC, into BPMN. Fig. 1 (called $P1$) depicts the *Management of patients on arrival* process in the Neurosurgical Ward of Parma Hospital. As can be seen, the neurosurgeron makes a preliminary assessment of the patient's clinical condition and relies on such assessment result to recommend one of the following actions: keeping patients in observation (sub-process *Patients in observation*), patients in further investigation (sub-process *Patients in investigation*), patients in emergency (sub-process

Patients in emergency), or redirecting patients to other destinations. Fig. 2 (called $P2$) shows the *Patients in emergency* process in detail. Based on our definition, there exists a part-whole relationship between the processes described in Figures 2 and 1 in which the former is the "part" and the latter is the "whole". Such relationship is reflected by activity *Patients in emergency* (t_{14}) in $P1$ which is the abstract activity representing process $P2$. It means that the result of executing activity t_{14} in $P1$ is completely the result of executing process $P2$, and vise versa. The insertion point here is at activity t_{14} in $P1$. Let us compute the cumulative effects of $P1$ at such point, $CE(P1, t_{14}) = \{es_{14}\}$ where $es_{14} = assessed(p) \wedge tobeOperated(p) \wedge examined(p) \wedge operated(p) \wedge hospitalized(p) \wedge (recovered(p) \vee deathTreated(p))$. We only have one effect scenario i.e. es_{14} since there is only one path (no pair of branching-joining XOR) reaching activity t_{14} from the start event. Then, let us compute the cumulative effects by insertion, $CE(P1 \uparrow^{t_{14}} P2, t_{14}) = assessed(p) \wedge tobeOperated(p) \wedge examined(p) \wedge operated(p) \wedge hospitalized(p) \wedge (recovered(p) \vee deathTreated(p))$. We can infer that $P2$ is a part of $P1$, since $CE(P1, t_{14}) = CE(P1 \uparrow^{t_{14}} P2, t_{14})$.

We also consider another setting where there exists a process $P3$, e.g. a detailed process (not described in the diagram) of activity *Surgical operation* in Fig. 2, which is a sub-process of $P2$. Intuitively, we consider process $P3$ also be a part of process $P1$ though there is no activity in $P1$ which is completely represented by the functionalities of $P3$. On such setting, there is an activity in $P1$ entails the functionalities of $P3$. Then, we can say there is a *direct* part-whole relationship between $P2$ and $P1$ and an *indirect* one between $P3$ and $P1$.

Inter-operation

Inter-operation relationship exists between two processes when there is at least one message exchanged between them and there is no cumulative effects contradiction between tasks involved in exchanging messages. We formalize the definition of inter-operation relationship as Definition 3.

Definition 3. *Given process models $P1$ and $P2$, inter-operation relationship exists between these processes including activities t_i and t_j iff the following holds:*

- *$\exists t_i$ in $P1$ $\exists t_j$ in $P2$ such that $t_i \rightharpoonup t_j$ denotes t_i sends a message to t_j, or in the reverse direction $t_j \rightharpoonup t_i$;*
- *Let $E_i = \{es_{i1}, es_{i2}, \ldots, es_{im}\}$ be cumulative effects of process $P1$ at task t_i i.e. $CE(P1, t_i)$, and $E_j = \{es_{j1}, es_{j2}, \ldots, es_{jn}\}$ be cumulative effects of process $P2$ at task t_j i.e. $CE(P2, t_j)$. Then, there is no contradiction between E_i and E_j for all $es_{ip} \in E_i$ and $es_{jq} \in E_j$ s.t. $es_{ip} \cup es_{jq} \vdash \perp$ does not hold, where $1 \leq p \leq m$ and $1 \leq q \leq n$.*

We say there exists a **direct inter-operation** between processes $P1$ and $P2$ due to messange exchanged between them. However, we also consider another process $P3$ which has a direct inter-operation relationship with process $P2$. Intuitively, process $P3$ also has an inter-operation relationship with process $P1$ through process $P2$. We say process $P3$ is in an **indirect inter-operation** relationship

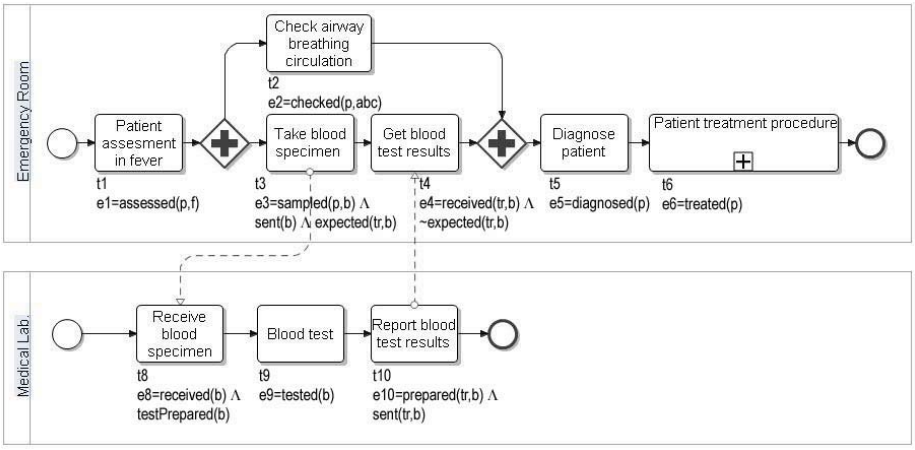

Fig. 3. BPMN model of inter-operation processes of *Handling of patient in fever in emergency room*, also containing the immediate effects e_i of each activity t_i

with process $P1$ iff there exists another process $P2$ such that $P3$ be in direct inter-operation with $P2$ as well as $P2$ be in direct inter-operation with $P1$. Effects contradiction exists if the expected effects differ from the given effects. If it is the case, we do not consider such relationship as inter-operation though there is a message exchanged between a pair of processes.

Fig. 3 represents an example of inter-operation between processes of *Handling of patient in fever in emergency room*. On this setting, there exist messages sent from task *Take blood specimen* t_3 in *Emergency Room* process (called $P1$) to task *Receive blood specimen* t_8 in *Medical Lab* process (called $P2$), and from task *Report blood test results* t_{10} in $P2$ to task *Get blood test results* t_4 in $P1$ in order to fulfill the functionalities of such processes. Semantically, we can compute $CE\,(P1, t_3) = \{es_{13}\}$ where $es_{13} = assessed\,(p, f) \wedge sampled\,(p, b) \wedge sent\,(b) \wedge expected\,(tr, b)$. Similarly, $CE\,(P2, t_8) = \{es_{28}\}$ where $es_{28} = received\,(b) \wedge testPrepared\,(b)$. We can observe that there is no contradiction between es_{13} and es_{28}. Dually, we can also compute $CE\,(P2, t_{10}) = \{es_{210}\}$ where $es_{210} = received\,(b) \wedge testPrepared\,(b) \wedge tested\,(b) \wedge prepared\,(tr, b) \wedge sent\,(tr, b)$. And, $CE\,(P1, t_4) = \{es_{14}\}$ where $es_{14} = assessed\,(p, f) \wedge sampled\,(p, b) \wedge sent\,(b) \wedge received\,(tr, b) \wedge \neg expected\,(tr, b)$. Again, it is obvious that there is no contradiction between es_{210} and es_{14}. We may consider effect contradiction in the following illustration. For example, see Fig. 3, if we include $labeled(b)$ as the expected effect in immediate effect $e8$ and $\neg labeled(b)$ as the given effect in immediate effect $e3$, then we fall into this contradiction since at t_8 we expect that the blood specimen has been labeled at the point of t_3.

Generalization-specialization

Generalization-specialization relationship exists between two processes when one process becomes the functional extension of the other. More specifically, the

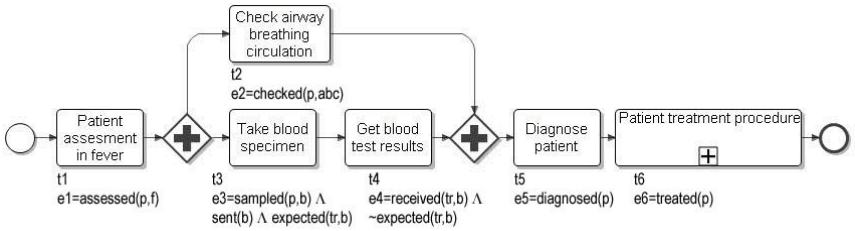

Fig. 4. BPMN model of *Handling of patient in fever in emergency room* process, also showing the immediate effects e_i of each activity t_i

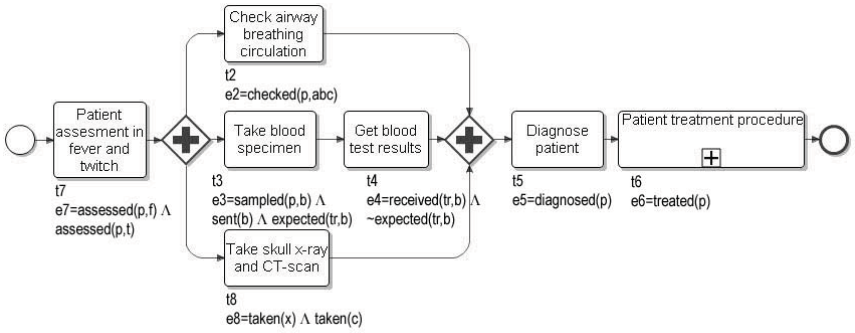

Fig. 5. BPMN model of *Handling of patient in fever and twitch in emergency room* process, also containing the immediate effects e_i of each activity t_i

specialized process has the same functionalities as in the *generalized* one and also extends it with some additional functionalities. Our interpretation of such relationship was inspired by the notion of subtyping that was first made popular in programming language theory and later extended to conceptual modeling. We do not directly link this interpretation to the definition of object-oriented inheritance or subclass, which is in fact a mechanism to achieve subtyping. In essence, we may not apply a pairwise comparison of tasks to the two process models in question. Instead, we compare their cumulative effects to see if the *specialized* process can safely be used in a context where the *generalized* one is expected, as described below. To the best of our knowledge, this interpretation is close to the *projection inheritance* defined in [12].

Using semantic effect analysis, the functionalities are represented as immediate effects (of individual activity) and cumulative effects (of the whole process). One way to extend the functionalities is adding some additional activities such that the intended cumulative effects of the process are consequently extended. Another way involves enriching the immediate effects of the existing activities. In this case, the number of activities remain the same for both processes but the capabilities of the *specialized* is extended. Noted, the *specialized* process inherits all functionalities of the *generalized* process, as formally defined in Definition 4.

Definition 4. *Given process models* $P1$ *and* $P2$, $P2$ *is a specialization of* $P1$ *iff* $\forall es_i \in acc(P1)$, $\exists es_j \in acc(P2)$ *such that* $es_j \models es_i$; *and* $\forall es_j \in acc(P2)$, $\exists es_i \in acc(P1)$ *such that* $es_i \models es_j$.

Figures 4 and 5 show an example of two processes describing how a patient in fever should be handled in an emergency room. As can be seen, the process described in Fig. 5 (called $P2$) has exactly the same functionalities as the one in Fig. 4 (called $P1$). The former however has some specific functionalities on tasks *Patient assessment in fever and twitch*, which is the extension of task *Patient assessment in fever*, and *Take skull x-ray and CT-scan*, which is the additional task. Both tasks together extend the functionalities of process in Fig. 4.

Furthermore, we can semantically observe such relationship overs Definition 4. Let us compute the cumulative effects of $P1$, $acc(P1) = \{es_1\}$ where $es_1 = assessed(p, f) \wedge checked(p, abc) \wedge sampled(p, b) \wedge sent(b) \wedge received(tr, b) \wedge \neg expected(tr, b) \wedge diagnosed(p) \wedge treated(p)$. Noted, we only have one effect scenario i.e. es_1 since there is only one path (no pair of branching-joining XOR) reaching the end event from the start event in $P1$. Dually, we can accumulate the effects of $P2$, $acc(P2) = \{es_2\}$ where $es_2 = assessed(p, f) \wedge assessed(p, t) \wedge checked(p, abc) \wedge sampled(p, b) \wedge sent(b) \wedge received(tr, b) \wedge \neg expected(tr, b) \wedge taken(x) \wedge taken(c) \wedge diagnosed(p) \wedge treated(p)$. It is evident that $e_7 \models e_1$ such that $es_2 \models es_1$. Obviously, we can also observe that $es_1 \models es_2$.

3.2 Process Changes

Now, let us discuss the first benefit of formalizing inter-process relationship in process changes. We consider three ways to look at relationship violations due to process changes between a pair of processes $P1$ and $P2$: (i) identify changes in $P1$ that can trigger violations and resolve them; (ii) identify changes in $P2$ that can trigger violations and resolve them; and (iii) identify resolutions to solve a given violated relationship of a pair of process with unknown changes trigger. Due to space constraint, we only describe the part-whole relationship. As in Definitions 1 and 2, let $P1$ be the whole process and $P2$ be the part one, and let t_i be a sub-process in $P1$ represents $P2$ with the corresponding immediate effects e_{t_i} such that the condition $COND$ is satisfied i.e. $CE(P1, t_i) = CE(P1 \uparrow^{t_i} P2, t_i)$.

First, the possible change introduced in $P1$ that can cause violations is changing on t_i, i.e. either by: (i) changing e_{t_i} to be e'_{t_i} s.t. $e_{t_i} \neq e'_{t_i}$, or (ii) dropping t_i. For the first case, we need to change $P2$ to be $P2'$ by either adding or deleting some activities such that: (a) $COND$ is satisfied with e'_{t_i}; and (b) there exists no $P2''$ s.t. $COND$ is satisfied with e'_{t_i}. In contrast, we no longer need to maintain the relationship for the second case. Noted, changing $P1$ excluding t_i will not cause any violation. *Second*, any changes in $P2$ which affect the $acc(P2)$ will cause an violation. Resolving such an violation, we need to replace e_{t_i} with e'_{t_i} such that: (a) $COND$ is satisfied with e'_{t_i}; and (b) there exists no e''_{t_i} s.t. $COND$ is satisfied with e''_{t_i} and $e_{t_i} \Delta e''_{t_i} \subset e_{t_i} \Delta e'_{t_i}$. There would be a complex case due to a fact that t_i might be being utilized in many other processes. Consequently, if we change e_{t_i}, we must propagate this change to the others as well. However,

note that changing e_{t_i} will change the cumulative effects of the process being evaluated. Other scenario would be possible such that we can avoid change propagation in a massive manner, i.e. establishing a new task with e'_{t_i} s.t. $COND$ is satisfied. *Third*, any given violated part-whole with unknown changes trigger can be resolved by using the aforementioned approaches after identifying a candidate of t_i which is approached by the closest $COND$ to be satisfied.

3.3 Process Lattices

Let us leverage the lattice theory in constructing process lattices as the further benefit of formalizing inter-process relationship. We will show that the relationship types lead to partial orders which is the basis for constructing process lattices from a large collection of processes. We can then define *least upper bound* (lub) and *greatest lower bound* (glb), as described below, for each qualified type. The process lattice view permits us to perform formal analysis to support the identification and maintenance of inter-dependent processes in process repository, such as: (1) lub queries can tell us what the most specific generalization of a set of processes might be; (2) helps localizing change between glb and lub. If the glb and lub of a set of processes are not impacted, then change does not propagate past them; (3) we want to reason with the transitive closure, but explicitly representing it is expensive.

Definition 5. *[3] Let P be a set. A partial order on P is a binary relation $\leq$ on P such that, for all $x, y, z \in P$: (i) $x \leq x$, (ii) $x \leq y$ and $y \leq x$ imply $x = y$, (iii) $x \leq y$ and $y \leq z$ imply $x \leq z$.*

These conditions are referred to, respectively, as reflexivity, antisymmetry and transitivity. A set P equipped with an order relation $\leq$ is said to be an ordered set (or partially ordered set, called poset) [3]. A lattice is a poset in which any two elements have a unique supremum (the least upper bound *lub*; called their *join*) and an infimum (the greatest lower bound *glb*; called their *meet*). If $a \leq c$, $b \leq c$ in a partially ordered set $P = (X; \leq)$, we say that c is an upper bound of a and b. If $d \leq a$, $d \leq b$ we say d is a lower bound of a and b. We say an upper bound c of a and b is the lub if $c \leq c'$ for every upper bound c' of a and b. It is denoted $a \vee b$ and called the join of a and b. The glb is defined similarly and denoted $a \wedge b$ and called the meet of a and b.

Based on the given properties of a poset, we propose Theorems 1, 2, and 3 for the process relationship types to identify whether or not each type is a poset. Then, we may define a lattice for a relationship type if it qualifies a poset.

Theorem 1. *Part-whole is a reflexive, transitive and antisymmetric relationship.*

Proof. Let process $P2$ be a part of process $P1$ and their corresponding cumulative effects be $acc(P2)$ and $acc(P1)$ respectively. Let process $P3$, with cumulative effects $acc(P3)$, be a part of process $P2$. Based on Definitions 1 and 2, we have $acc\,(P1) = acc\,(P1 \uparrow^t P2)$ and $acc\,(P2) = acc\,(P2 \uparrow^t P3)$. Therefore,

$\forall es_k \in acc(P3)$, $\exists es_j \in acc(P1)$ such that $es_j \models es_k$. So part-whole is *transitive*. As for *reflexivity*, $\forall es_i \in acc(P1)$, $\exists es_j \in acc(P1)$ such that $es_j \models es_i$ whereas $i = j$. Finally, it is *antisymmetric* similar with the reflexivity proof.

Theorem 2. *Generalization-specialization is a reflexive, transitive and antisymmetric relationship.*

Proof. Let process $P2$ be a specialization of process $P1$ and their corresponding cumulative effects be $acc(P2)$ and $acc(P1)$ respectively. Let process $P3$, with cumulative effects $acc(P3)$, be a specialization of process $P2$. It is obviously *reflexive* because $\forall es_i \in acc\,(P1)$, $\exists es_j \in acc\,(P1)$ such that $es_j \models es_i$; and $\forall es_j \in acc\,(P1)$, $\exists es_i \in acc\,(P1)$ such that $es_i \models es_j$ whereas $i = j$. Similarly, we can analyze the rest processes. It is *transitive* since $\forall es_i \in acc\,(P1)$, $\exists es_j \in acc\,(P2)$ such that $es_j \models es_i$; and $\forall es_j \in acc\,(P2)$, $\exists es_i \in acc\,(P1)$ such that $es_i \models es_j$; furthermore $\forall es_j \in acc\,(P2)$, $\exists es_k \in acc\,(P3)$ such that $es_k \models es_j$; and $\forall es_k \in acc\,(P3)$, $\exists es_j \in acc\,(P2)$ such that $es_j \models es_k$. Then, we can summarize as follows: $es_k \models es_j \wedge es_j \models es_i \Rightarrow es_k \models es_i$; and $es_i \models es_j \wedge es_j \models es_k \Rightarrow es_i \models es_k$. It is *antisymmetric*. If $P2$ is specialization of $P1$ and $P1$ is specialization of $P2$, then $P1 = P2$. Since, $\forall es_i \in acc\,(P1)$, $\exists es_j \in acc\,(P2)$ such that $es_j \models es_i$; and $\forall es_j \in acc\,(P2)$, $\exists es_i \in acc\,(P1)$ such that $es_i \models es_j$; moreover $\forall es_j \in acc\,(P2)$, $\exists es_i \in acc\,(P1)$ such that $es_i \models es_j$; and $\forall es_i \in acc\,(P1)$, $\exists es_j \in acc\,(P2)$ such that $es_j \models es_i$. We can summarize as follows: $es_j \models es_i \wedge es_i \models es_j \Rightarrow es_j = es_i$.

Theorem 3. *Inter-operation is a non-reflexive, transitive and antisymmetric relationship.*

Proof. Let processes $P1$ and $P2$ have messages exchanged between them. So do processes $P2$ and $P3$. It is *non-reflexive* since there is no message sent to and received from the same process. It is *transitive*, i.e. $P1$ and $P3$ are in indirect inter-operation relationship through $P2$. It is *antisymmetric*, but it is not necessarily both processes are the same.

Theorems 1, 2 and 3 imply that part-whole and generalization-specialization qualify posets, thus they are considered in constructing process lattices.

4 Related Work

Malone et.al. [9] establish part-use and generalization-specialization to classify processes in the repository. van der Aalst [11] describes message sequence charts to specify the interaction between organizations. Dai et.al. [2] propose a lightweight query-based analysis for process impact analysis based upon process dependencies. van der Aalst and Basten [12] propose inheritance-preserving transformation rules to restrict changes in workflow process definitions. They introduce protocol and projection inheritances. Koliadis and Ghose [7] introduce an inter-operation business process in compliance checking. Different to the others, we specifically propose a framework for formalizing and establishing

inter-process relationships based on the semantically effects annotated model. However, we found similar ideas with the aforementioned researches i.e. part-whole in [9], generalization-specialization in [9,12] and inter-operation in [7,11].

5 Conclusion and Future Work

We have proposed a rigorous framework for establishing relationships between process models shedding light on further processing on process ecosystems (e.g. re-establishing equilibrium of a process ecosystem such that all inter-process relationship constraints are satisfied). Future works include: i) implementing this approach into a semi-automated system that assists the designer in establishing relationships between process models; ii) maintaining process relationships against changes made to any process model within an ecosystem; and iii) developing a machinery approach for querying processes based on process lattices.

References

1. Bevilacqua, M., Ciarapica, F.E., Giacchetta, G.: Business Process Re-engineering in Healthcare Management: A Case Study. BPM Journal 17(1), 42–66 (2011)
2. Dai, W., Covvey, D., Alencar, P., Cowan, D.: Lightweight Query-based Analysis of Workflow Process Dependencies. Journal of Syst. and Soft. 82(6), 915–931 (2009)
3. Davey, B.A., Priestley, H.A.: Introduction to Lattices and Order. Cambridge University Press (1990)
4. Ekanayake, C.C., La Rosa, M., ter Hofstede, A.H.M., Fauvet, M.-C.: Fragment-based Version Management for Repositories of Business Process Models. QUT Digital Repository (2011), `http://eprints.qut.edu.au/`
5. Hallerbach, A., Bauer, T., Reichert, M.: Managing Process Variants in the Process Life Cycle. In: ICEIS 2008, Barcelona, pp. 154–161 (2008)
6. Hinge, K., Ghose, A., Koliadis, G.: Process SEER: A Tool for Semantic Effect Annotation of Business Process Models. In: IEEE EDOC 2009, pp. 54–63 (2009)
7. Koliadis, G., Ghose, A.: Verifying Semantic Business Process Models in Inter-operation. In: IEEE SCC, pp. 731–738 (2007)
8. Kurniawan, T.A., Ghose, A.K., Lê, L.-S.: A Framework for Optimizing Inter-operating Business Process Portfolio. In: Proc. of the 19th International Conference on Information Systems Development, ISD-2010 (2010)
9. Malone, T.W., Crowston, K., Herman, G.A.: Organizing Business Knowledge: The MIT Process Handbook. The MIT Press (2003)
10. van der Aalst, W.M.P., Jablonski, S.: Dealing with Workflow Change: Identification of Issues and Solutions. Int. Journal of CSSE 15(5), 267–276 (2000)
11. van der Aalst, W.M.P.: Interorganizational Workflows: An Approach Based on Message Sequence Charts and Petri Nets. Systems Analysis-Modelling-Simulation 34(3), 335–367 (1999)
12. van der Aalst, W.M.P., Basten, T.: Inheritance of Workflows: An Approach to Tackling Problems Related to Change. Theo. Comp. Sci. 270(1-2), 125–203 (2002)
13. Weber, I., Hoffman, J., Mendling, J.: Semantic Business Process Validation. In: Proc. of the 3rd Int. Workshop on Semantic Business Process Management (2008)
14. White, S.A., Miers, D.: BPMN: Modeling and Reference Guide (2008)