Design Maintenance in Process Eco-systems

Tri A. Kurniawan, Aditya K. Ghose, Hoa Khanh Dam, Lam-Son Lê and Tiancheng Zhang
Decision Systems Lab., School of Computer Science and Software Engineering,
University of Wollongong, NSW 2522, Australia
Email: {tak976, aditya, hoa, lle, tz746}@uow.edu.au

***Abstract*—This paper addresses the problem of managing business process changes at the design level within a complex business process repository. We argue that a formal process eco-systems view can provide a particular useful solution to the problem in which their inter-process relationships can be properly described. Our intent is to build a modular infrastructure to support change management in process eco-systems by leveraging a change propagation approach that would maintain the inter-process relationships. This permits us to propagate the changes, made on a particular process, to the rest of the processes for maintaining the relationship equilibrium of such eco-system. Our experimental evaluation suggests that the propagation of change can be efficiently achieved, which suggests that the real source of complexity stems from the redesign of individual models.**

***Keywords*-process eco-system; business process; change propagation; semantic annotation;**

I. Introduction

Most medium to large organizations support large collections of process designs modeled using many business process modeling languages such as Business Process Model and Notation (BPMN). These are often stored in *business process repositories* and are typically characterized by the following features. First, the number, scale and complexity of the processes are large, consisting of hundreds or even thousands of process models. For example, the SAP R/3 reference model contains 600+ models and Suncorp's repository contains 6,000+ models [1]. Second, most of these processes are inter-dependent (in terms of both design and execution). Evidence of such dependency has been described in a number of published sources [2]–[5]. We can find a functional dependency, as one type of such dependency, where some process designs exist to realize component functionalities of other process designs. Third, changes to any one process are likely to impact several other processes. Approaches to analyze the impact of process changes, depending upon type of process dependency, have been described in [5]. Fourth, there are multiple alternative ways in which changes might be implemented [6]–[8]. Fifth, there are multiple design alternatives for determining a collection of process designs that realizes the required functionalities [7], [9]. Finally, these collections of processes must operate under complex constraints imposed by the domain (including compliance constraints) [10].

Dealing with the design, maintenance and full life-cycle management of such complex collections of processes is a non-trivial task. Due to this complexity, many issues arise during each process's life-cycle such as: (i) managing process variants [11]; (ii) maintaining relationship consistency, i.e. the relationship constraints between a pair of processes are satisfied, among inter-dependent processes due to any process change drivers (we can find these change drivers described in [12] as well as problems in process optimization introduced in [9]) and (iii) performing process impact analysis [5] if changes applied to any process. These issues are stemmed primarily from introducing process changes which may introduce relationship constraints violations.

We argue that a *formal process eco-systems* view can provide a particularly useful solution to the problem. Our intention is to leverage the eco-systems metaphor by using mathematical characterizations of such eco-systems - in particular of equilibria. In our conception of process eco-systems, process designs play a role analogous to that of biological entities in a biological eco-system. They are created (or discovered, using automated toolkits [13]), modified during their lifetimes, and eventually discarded. They undergo constant change, driven by changing requirements or changes in the operating context. Furthermore, perturbations in a process eco-system propagate across its constituent designs, driven by the need to maintain a range of critical inter-process relationship constraints. We will deem a process eco-system to be in an equilibrium if all inter-process relationships in the eco-system are mutually consistent, and there is no alternative equilibrium that further minimizes change to the prior state of the process eco-system.

Our intent is to build a modular infrastructure to support change management in process eco-systems by leveraging a change propagation approach that would maintain the inter-process relationships. It permits the individual process redesign module can be swapped without impacting the rest of the infrastructure (e.g. see [6], [7], [14] for business process redesign frameworks). The process redesign module might implement any of the approaches, or might be entirely manually executed. Our experimental evaluation suggests that the propagation of change can be efficiently achieved, which suggests that the real source of complexity stems from the redesign of individual models.

The rest of the paper is organized as follows. Section II introduces semantic effect-annotated process model and inter-process relationships. Section III describes the consistency management in process eco-systems. Section 9 describes the evaluation. Section V outlines the related work. Section VI draws the conclusion and suggests the future work.

II. Preliminaries

This section introduces basic notions of semantic effect-annotated process models and recalls inter-process relationships classification as presented in [15].

A. Semantic Effect-Annotated Process Model

Koliadis and Ghose [14] discuss the concept of semantic effects. An effect annotation relates to a specific result or outcome to an activity in a process model. An activity represents the work performed within a process. Activities are either atomic (called a *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 a *sub-process*, i.e. they can be broken down to see another level of process below) [16]. In an annotated BPMN model, each 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 [17] approach to semantic effect annotation. This framework permits us to determine, at design time, the answer to the following question which can be posed for any point in the process design: what would the effects of the process be if it were to be executed 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. This 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. The use of CNL permits the immediate effects of activities to be specified in natural language (but with a restricted set of sentence formats) and it 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.

Fig. 1 illustrates a semantic effect-annotated BPMN process model. The immediate effect e_i of each activity t_i is represented in a Conjunctive Normal Form (CNF) allowing us to describe such effect as a set of outcome clauses. Similarly, we annotate all activities in the remaining diagrams with their corresponding immediate effects. Let p be patient to be observed and treated. The activity t_{11} has an immediate effect $e_{11} = assessed(p)$ which depicts the outcomes of executing such activity. We also have annotated the other activities, e.g. activity t_{14} has immediate effects $e_{14} = to\_be\_operated(p) \wedge examined(p) \wedge operated(p) \wedge hospitalized(p) \wedge (recovered(p) \vee dead(p))$, and so forth. Obviously, we have four alternative paths for completing the process from the *start* to the *end* events due to two pairs of XOR gateways. The cumulative effects of such execution until t_{14} can be accumulated from t_{11} and t_{14}. Since there is no way to undo effects of the previous step, we would have the cumulative effects at t_{14} as $assessed(p) \wedge to\_be\_operated(p) \wedge examined(p) \wedge operated(p) \wedge hospitalized(p) \wedge (recovered(p) \vee dead(p))$. Similarly, we can also compute for the other activities.

B. Inter-process Relationships Classification

We classify the inter-process relationships into two categories: *functional dependencies* and *consistency links*. A functional dependency exists between a pair of processes when one process needs support from the other for realizing some of its functionalities. In contrast, a consistency link exists between a pair of processes when both of them have intersecting parts which represent the same functionality, i.e. the outcomes (e.g. effects) of these parts are exactly the same. They are functionally independent.

In such categories, we define the three different types of relationship that may 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 semantic effect analysis on process models [15]. We also consider other types of relationship (e.g. abstraction-refinement, resource-sharing, informational inter-operation and non-functional inter-operation) which may exist in a complex business process repository and leave them to be formalized in our future investigation. Our proposed framework focuses on these three types. We use $acc\,(P)$ to denote the cumulative end effects of process P; $CE\,(P, t_i)$

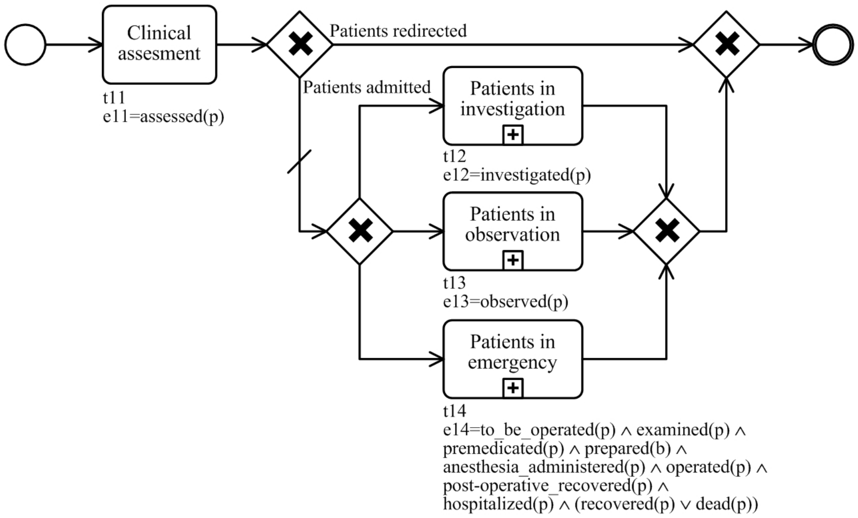


Figure 1. *Management of patients on arrival* process.

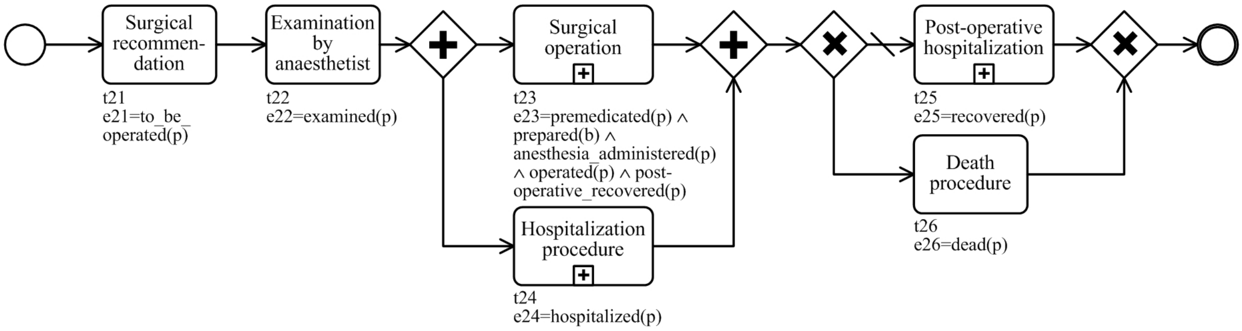


Figure 2. Expansion of *Patients in emergency* sub-process in Fig. 1.

to describe cumulative effect at the point of activity t_i within process P; and es_j to denote an effect scenario *j-th*. Noted, 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.

1) Part-whole: A part-whole relationship exists between two processes when one process is required by the other 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. Logically, there is an insertion of the functionalities of the 'part' into the 'whole'.

We define the insertion of process $P2$ in process $P1$ at activity t, $P1 \upharpoonleft^t P2$, is a process design obtained by viewing $P2$ as the sub-process expansion of t in $P1$. We then define the part-whole as follows. $P2$ is a direct part of $P1$ iff there exists an activity t in $P1$ such that $CE\,(P1, t) = CE\,(P1 \upharpoonleft^t P2, t)$. If there is no such insertion point 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 the part-whole process relationship, adopted from [3], which is illustrated in Figs. 1 and 2. We transformed them, from originally represented in EPC, into BPMN and annotated all activities with their corresponding semantic effects. Fig. 1 (called $P1$) depicts the *Management of patients on arrival* process in a hospital. As can be seen, the neurosurgeon 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*), keeping patients in further investigation (sub-process *Patients in investigation*), keeping 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 Figs. 2 and 1 in which the former is the 'part' and the latter is the 'whole'.

Such relationship is reflected by the activity *Patients in emergency* (t_{14}) in $P1$ which is the abstract activity representing $P2$. This means that the result of executing activity t_{14} in $P1$ is completely the result of executing $P2$, and vice-versa. The insertion point here is at t_{14} in $P1$. The cumulative effect of $P1$ at this point is $CE\,(P1, t_{14}) = \{es_{14}\}$; $es_{14} = assessed\,(p) \wedge to\_be\_operated\,(p) \wedge examined\,(p) \wedge operated\,(p) \wedge hospitalized\,(p) \wedge (recovered\,(p) \vee dead\,(p))$. We only have one effect scenario, i.e. es_{14}. Furthermore, the cumulative effect of $P1$ at t_{14} by inserting $P2$ at this activity is $CE\,(P1 \uparrow^{t_{14}} P2, t_{14}) = assessed\,(p) \wedge to\_be\_operated\,(p) \wedge examined\,(p) \wedge operated\,(p) \wedge hospitalized\,(p) \wedge (recovered\,(p) \vee dead\,(p))$. Finally, we can infer that $P2$ is a part of $P1$ since $CE\,(P1, t_{14}) = CE\,(P1 \uparrow^{t_{14}} P2, t_{14})$. Fig. 3 illustrates a process graph representing part-whole relationships between *Management of patients on arrival* process with the others.

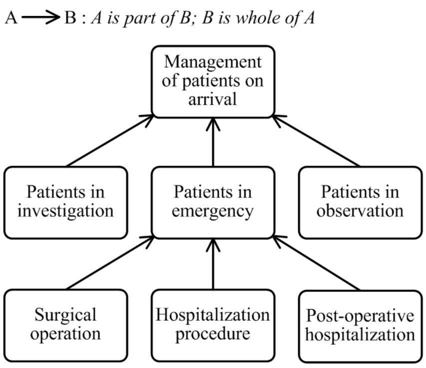


Figure 3. A process graph represents the part-whole relationships of *Management of patients on arrival* process with the others.

2) Inter-operation: An inter-operation relationship exists between two processes when there is at least one message exchanged between them and there is no cumulative effect contradiction between tasks involved in exchanging messages. Formally, given processes $P1$ and $P2$, an inter-operation relationship exists between them 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 that t_i sends a message to t_j, or $t_j \rightharpoonup t_i$, if the message is in the opposite direction;
- Let $E_i = \{es_{i1}, es_{i2}, \ldots, es_{im}\}$ be the 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 the 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 $es_{jq} \in E_j$ such that $es_{ip} \cup es_{jq} \vdash \bot$ does not hold, where $1 \leq p \leq m$ and $1 \leq q \leq n$.

We say there exists a **direct inter-operation** between $P1$ and $P2$ due to messages exchanged between them, however, we also consider another process model $P3$ which has a direct inter-operation relationship with $P2$. Logically, $P3$ also has an inter-operation relationship with $P1$ via $P2$. We say $P3$ is in an **indirect inter-operation** relationship with $P1$ 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 $P1$. Effect contradiction exists if the expected effects differ from the given effects. If this is the case, we do not consider such relationship as an inter-operation one even though there is a message between them.

3) Generalization-specialization: A generalization-specialization relationship exists between two processes when one process becomes the functional extension of the other. More specifically, the *specialized* process has the same functionalities as in the *generalized* one and also extends it with some additional functionalities. Our interpretation of it was inspired by the notion of sub-typing which 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, which is, in fact, a mechanism to achieve sub-typing. In essence, we do not apply a pairwise comparison of activities to both processes in question. Instead, as described below, we compare their cumulative effects to see if the *specialized* process can safely be used in a context where the *generalized* one is expected.

One way to extend the functionalities is by adding some additional activities so that the intended cumulative end 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 remains the same for both processes but the capabilities of the *specialized* process is extended. Noted, the *specialized* process inherits all functionalities of the *generalized* process, as formally defined as follows. 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 $es_i \models es_j$.

III. Consistency Management in Process Eco-systems

This section describes the proposed framework including basic notions of process changes, resolution of relationship violations and detailed example. By proposing this framework, we assume the following constraints: (i) each process model has been annotated with the semantic effect, (ii) any relationship which may exist between a pair of processes in a process repository has been carefully established and validated by the process analyst refers to the definitions (i.e. part-whole, inter-operation and generalization-specialization) accordingly, (iii) each validated relationship between a pair of processes has been stored as the relationships reference to be referred in resolving any relationship violation and (iv) there must be a single change in a particular process model to propagate until all violations are resolved (i.e. we are not allowed to make another change on that process while the previous change being propagated).

A. Process Changes as an Inconsistency Driver

Changes made on a particular process within a process eco-system may introduce inconsistencies on its relationships to the others, leading to perturbation in the equilibrium of the eco-system. The process graph shown in Fig. 3 can be used to illustrate the impacts of changes made on a particular process. Suppose that *Surgical operation* process is modified due to some requirement changes. As a result, its cumulative effects are changed. Consequently, its new cumulative effects would violate the part-whole relationship between this process and *Patients in emergency* process. In order to maintain this relationship, we need to modify *Patients in emergency* process in some ways. This modification may also violate the part-whole relationship between this process and *Management of patients on arrival* process. Therefore, we may need to modify *Management of patients on arrival* process as well.

The aforementioned change scenario illustrates the impacts of process changes to the equilibrium of a process eco-system, i.e. introducing inconsistencies in part-whole relationships. The changes on one process are propagated to the related processes in order to maintain such equilibrium. Depending on the configuration of process eco-system, we may have another inconsistency in generalization-specialization and inter-operation relationships. We also consider that changes on a process design, to satisfy the requirements, can be achieved in many ways. For example, achieving cumulative effects of two tasks A and B can be done by either sequencing or parallelizing both of them. The choice of such variation would be considered during redesigning a process model. Considering this variation would affect the equilibrium configuration of a process eco-system. Then, we would have many options to get a new consistent equilibrium of such process eco-system.

B. Resolving Relationship Violations

Resolving relationship violations in a process eco-system involves generating process variants after all change propagation paths have been observed. Generating a process variant from a process being evaluated can be performed either by means of machinery or with the help of an analyst[1]. This procedure is non-trivial and we need to develop algorithms for it. In designing this variant-generating process, we should be careful when modifying the immediate effect of tasks that are utilized in many processes. Modifications applied to such tasks from any of these processes (i.e. the ones that utilize them) will potentially introduce inconsistency to relationships between it and the other processes.

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 violations. Let $P1$ and $P2$ be the whole and the part, respectively. Let t_i in $P1$, with immediate effects e_{t_i}, be a sub-process representing $P2$ such that the condition $COND$ is satisfied, i.e. $CE\,(P1, t_i) = CE\,(P1 \uparrow^{t_i} P2, t_i)$.

We only illustrate the first way of the violations, as follows. The possible change introduced in $P1$ that can cause violations is a change to t_i, i.e. either by: (i) changing e_{t_i} to be e'_{t_i} such that $e_{t_i} \neq e'_{t_i}$ or (ii) dropping t_i. In the first case, we need to change $P2$ to be $P2'$ by either adding or deleting some activities or reducing or extending some immediate effects of some particular activities such that: (a) $COND$ is satisfied with e'_{t_i} and (b) there exists no $P2''$ such that $COND$ is satisfied with e'_{t_i}. In contrast, we no longer need to maintain the relationship for the second case. Note, changing $P1$ by excluding t_i will not cause any violation.

C. Detailed Example

Let us use a small process eco-system to illustrate our approach. This setting includes three process models: (i) *Management of patients on arrival* process as shown in Fig. 1, denoted as $P1$, (ii) *Patients in emergency* process as shown in Fig. 2, denoted as $P2$ and (iii) *Surgical operation* process as shown in Fig. 4, denoted as $P3$. In this scenario, $P3$ is part of $P2$ and $P2$ is part of $P1$ as can be seen in Fig. 3. These part-whole relationships are established according to their formal definitions given in Section II.

Now, we need to change the process of Fig. 4 based on the understanding that not all operation surgeries require blood preparation. This activity would be performed only on the need. Hence, we have process in Fig. 5, denoted as $P3'$, as the result of such change. This change impacts on the cumulative end effects of processes in Figs. 4 and 5 whereas $acc\,(P3') \neq acc\,(P3)$. Let b be blood to be prepared in the operation surgeries. In process $P3'$, we compute the cumulative end effects as $premedicated(p) \wedge (\vee prepared(b)) \wedge anesthesia\_administered(p) \wedge operated(p) \wedge post\text{–}operative\_recovered(p)$, while the cumulative end effects of $P3$ is $premedicated(p) \wedge prepared(b) \wedge anesthesia\_administered(p) \wedge operated(p) \wedge post\text{–}operative\_recovered(p)$. Given this differences, the violation occurs in the part-whole relationship between processes in Figs. 5 and 2 since $CE\,(P2, t_{23}) \neq CE\,(P2 \uparrow^{t_{23}} P3', t_{23})$. In order to resolve this violation, we need to change the immediate effect of

[1]The techniques for generating all process variants are out of the scope of this paper. We leave them as our future investigations.

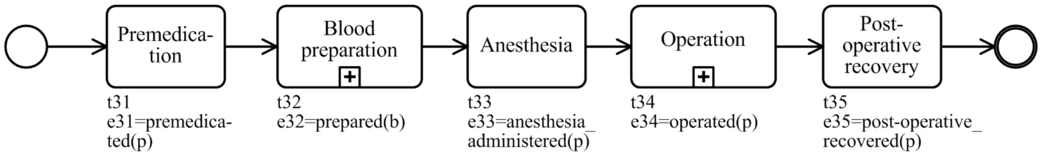


Figure 4. Original expansion of *Surgical operation* sub-process in Fig. 2.

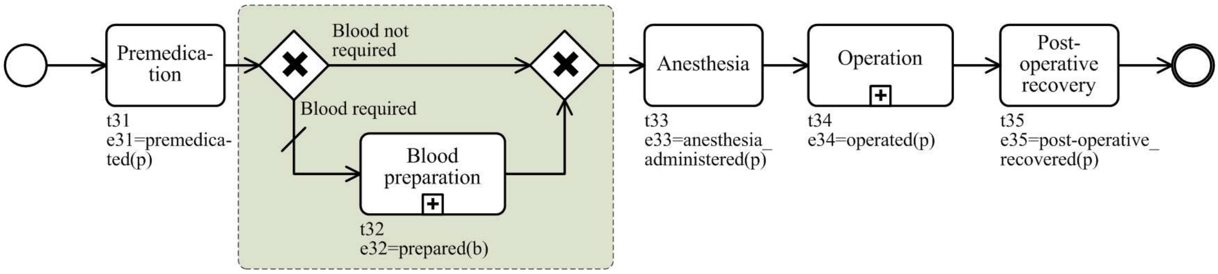


Figure 5. Modified expansion of *Surgical operation* sub-process in Fig. 2. The grey part depicts the changed section.

the corresponding sub-process in $P2$, i.e. $t23$, such that this effect represents the result of executing $P3'$, as illustrated partly in Fig. 6.

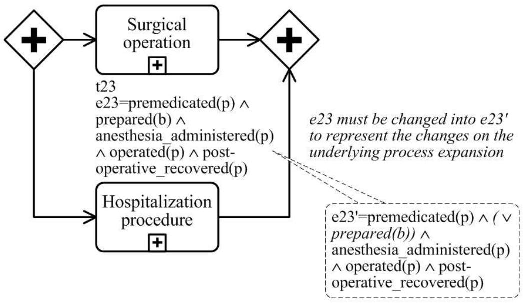


Figure 6. Process fragment of Fig. 2 showing changes on the immediate effects $e23$ as propagated from its process expansion in Fig. 5.

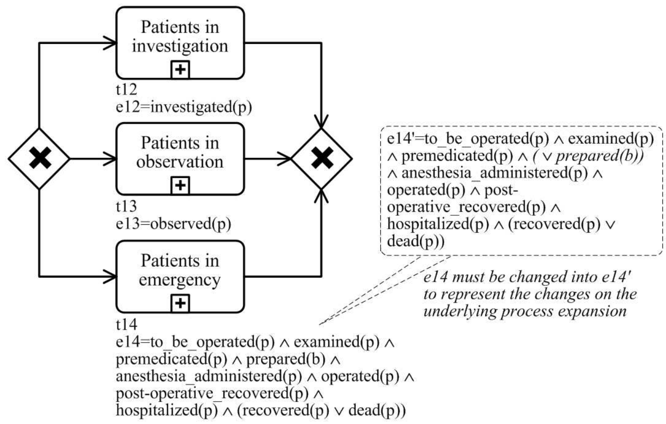


Figure 7. Process fragment of Fig. 1 showing changes on the immediate effects $e14$ as propagated from its process expansion in Fig. 2 illustrated in Fig. 6.

Furthermore, the above changes made on process in Fig. 2, i.e. to be $P2'$, will lead another violation between this process and process in Fig. 1. This violation can be similarly observed as in aforementioned analysis between $P3'$ and $P2$. Resolving this violation, the immediate effect of the corresponding sub-process in $P1$ to be $P1'$, i.e. $t14$, must be changed such that this effect represents the result of executing $P2'$, as illustrated partly in Fig. 7. Given such resolution, we would have $CE(P1', t_{14}) = CE(P1' \uparrow^{t_{14}} P2', t_{14})$. Hence, we have a new equilibrium of such process eco-system consisting of modified processes $P3'$, $P2'$ and $P1'$.

IV. Evaluation

We performed experiments on our change propagation approach, using a i3 Intel Core-2.27 GHz, RAM 2.85 Gb laptop. We established a process eco-system consisting of 50 dummy semantic effect-annotated BPMN process models. The number of activities in each model ranges from 5 to 20. We annotated immediate effects to each activity and computed the cumulative effects at the pointed node of each process on the fly by utilizing our tool. Our framework generated all possible relationships and suggested the constraint graph of such process eco-system as shown in Fig. 8.

Process $P1$ was selected as the change driver such that its changes trigger the violation on its relationships to the others. For evaluation purposes, we incrementally included those 50 processes in the eco-system in 5 steps, started by 10 processes, and made the same change on $P1$ on each step. All changes on processes, for satisfying the relationship constraints, were manually done by the analyst. We only measured the elapsed time for propagating the changes excluding the redesigning a process as follows: (i) switch the

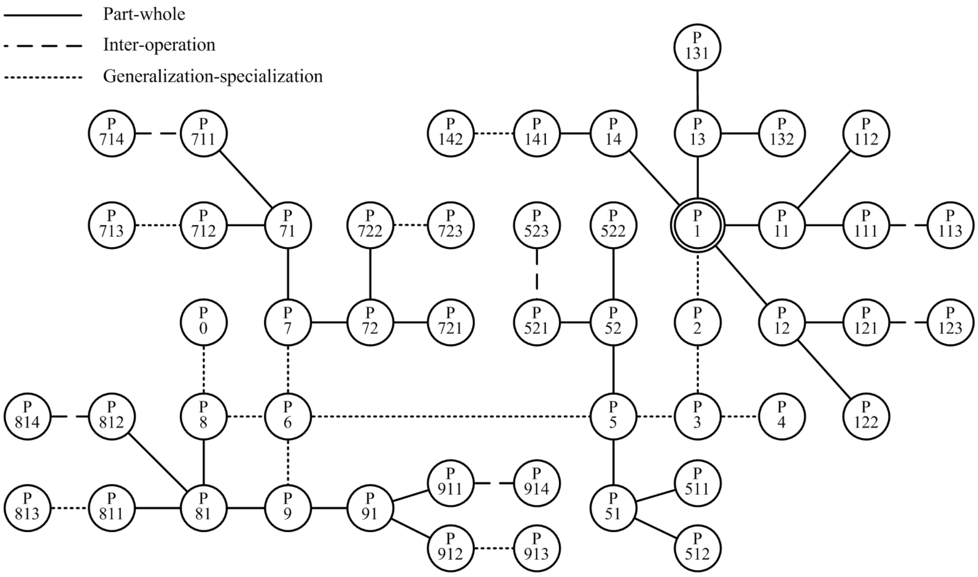


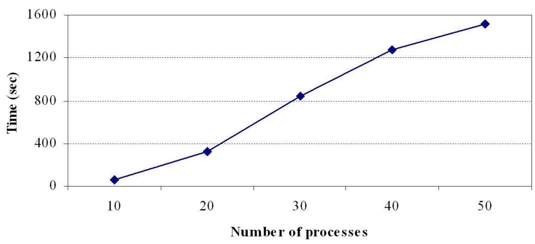
Figure 8. A complete constraint process graph represents a process eco-system consisting of 50 processes : $P1$ is the change driver.

timer ON to search the next process to be evaluated based on the current changed process and check the constraint satisfaction, (ii) if not satisfied, switch the timer OFF, and start to redesign such process, otherwise continue to get the other next processes (iii) after finishing such redesigning, switch the timer ON to check the constraint satisfaction between such redesigned process with the corresponding processes, and so forth. Let us assume that each redesigning process takes t **sec** time.

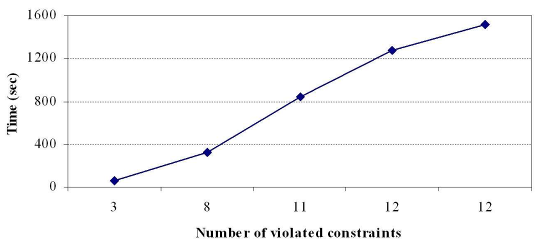
Table I
ELAPSED TIME FOR SEARCHING THE PROCESS ORDERING AND CHECKING THE CONSTRAINTS OF PROCESS ECO-SYSTEM IN FIG. 8

No. of processes	**No. of violated constraints**	**No. of redesigned processes *n***	**Elapsed time (sec)**
10	3	3	63
20	8	8	329
30	11	11	844
40	12	12	1,272
50	12	12	1,512

Table 1 describes number of processes, number violated constraints, number redesigned processes on each step and elapsed time for establishing a new equilibrium process eco-system using our change propagation approach. Each row represents a single step. The total elapsed time required to establish a restored-equilibrium process eco-system on each step should be calculated in *sec* by formula $(s + nt)$ where s, n and t represent elapsed time measured in experiment, number of redesigned processes and assumption time required for redesigning each process, respectively. It is not necessary the number of constraints to be equal with the number of redesigned process n since there might be a case that given m constraints, we only need to redesign n processes to satisfy all constraints where $n \leq m$. In addition, Fig. 9 describes these results into charts by evaluating the number of processes and violated constraints, respectively.



(a) Elapsed time for restoring the equilibrium according to number of processes



(b) Elapsed time for restoring the equilibrium according to number of violated constraints

Figure 9. Elapsed time by number of processes and violated constraints.

Our experiment results suggest that the proposed approach is efficient (excluding the complexity of redesigning an individual process) and scalable in propagating the changes to maintain the equilibrium of a given process eco-system involving 50 process models.

V. Related Work

Weidlich *et al.* [18] introduce an approach to determine a change region in another model by exploiting the behavioral profile of corresponding activities due to a model change. Their work based on related process models at different levels of abstraction. Their behavioral profile relies on three relations, which are based on the notion of weak order, between nodes in a process graph. If change propagation seems to be appropriate, the change region spots the position where extend the model. Dai *et al.* [5] propose a simple query-based analysis based on process dependencies for identifying the potentially affected entities if changes occur to processes. They develop a set of query rules that can be applied to the well-defined knowledge base at both activity and process levels to retrieve the potentially affected entities. Sadiq and Orlowska [8] propose process model transformation in order to identify the relationship between processes due to process changes. Ghose and Koliadis [19] present a framework for managing model change by considering each change as a perturbation that leads to inconsistencies and incompleteness which needs to be resolved. The authors illustrate the framework with consistency and completeness theories for the graphical BPMN, as well as mapping and completeness constraints for BPMN and Unified Modelling Language (UML) Sequence diagrams.

Different to the other related work, we specifically propose a novel approach for propagating changes in business process eco-systems by focusing on the three kind of inter-process relationships i.e. part-whole, inter-operation and generalization-specialization. In our approach, we consider business processes represented in semantic effect-annotated BPMN. We focus on how changes on one process can be properly propagated to the related processes which may be in relation. By doing so, eventually we can maintain the consistencies of relationships in process eco-systems.

VI. Conclusion and Future Work

We have proposed a novel framework, by leveraging change propagation approach, for managing process changes in process eco-systems. We have described and illustrated problem in process changes in order to maintain relationship consistency between processes. This framework can assist the process analyst in maintaining process relationship consistency in process eco-systems. Future work includes developing a framework of process variants generation for a given business process to extend the capabilities of the proposed framework. In addition, we aim to do experiments on our framework using case-studies taken from the industry.

References

[1] C. C. Ekanayake, M. L. Rosa, A. H. M. ter Hofstede, and M. C. Fauvet. (2011) Fragment-based version management for repositories of business process models. QUT Digital Repository. [Online]. Available: http://eprints.qut.edu.au/

[2] Map of Medicine. (2012). [Online]. Available: http://www.mapofmedicine.com/

[3] M. Bevilacqua, F. E. Ciarapica, and G. Giacchetta, “Business process re-engineering in healthcare management: a case study,” *BPM Journal*, vol. 17, no. 1, pp. 42–66, 2011.

[4] T. W. Malone, K. Crowston, and G. A. Herman, *Organizing Business Knowledge: The MIT Process Handbook*. The MIT Press, 2003.

[5] W. Dai, D. Covvey, P. Alencar, and D. Cowan, “Lightweight query-based analysis of workflow process dependencies,” *Journal of Syst. and Soft.*, vol. 82, no. 6, pp. 915–931, 2009.

[6] B. Weber, S. Rinderle, and M. Reichert, “Change patterns and change support features in process-aware information systems,” *LNCS*, vol. 4495, pp. 574–588, 2007.

[7] G. Koliadis and A. Ghose, “A conceptual framework for business process redesign,” *LNBIP*, vol. 29, pp. 14–26, 2009.

[8] W. Sadiq and M. E. Orlowska, “On business process model transformations,” *LNCS*, vol. 1920, pp. 267–280, 2000.

[9] T. A. Kurniawan, A. K. Ghose, and L. S. Lê, “A framework for optimizing inter-operating business process portfolio,” in *Proceedings of the ISD 2010*. Springer-Verlag Berlin Heidelberg, 2010, pp. 383–396.

[10] A. Ghose and G. Koliadis, “Auditing business process compliance,” *ICSOC 2007,*, vol. LNCS 4749, pp. 169–180, 2007.

[11] A. Hallerbach, T. Bauer, and M. Reichert, “Managing process variants in the process life cycle,” *ICEIS08*, pp. 154–161, 2008.

[12] W. M. P. van der Aalst and S. Jablonski, “Dealing with workflow change: Identification of issues and solutions,” *Int. Journal of CSSE*, vol. 15, no. 5, pp. 267–276, 2000.

[13] A. K. Ghose, G. Koliadis, and A. Cheung, “Rapid business process discovery,” *LNCS*, vol. 4801, pp. 391–406, 2007.

[14] G. Koliadis and A. Ghose, “Verifying semantic business process models in inter-operation,” *IEEE SCC*, pp. 731–738, 2007.

[15] T. A. Kurniawan, A. K. Ghose, L. S. Lê, and H. K. Dam, “On formalizing inter-process relationships,” in *Proceedings of the BPM 2011 Workshop*. Springer-Verlag Berlin Heidelberg, 2011, pp. 75–86.

[16] S. A. White and D. Miers, *BPMN: Modeling and Reference Guide*. Future Strategies Inc., 2008.

[17] K. Hinge, A. Ghose, and G. Koliadis, “Process seer: A tool for semantic effect annotation of business process models,” *IEEE EDOC*, pp. 54–63, 2009.

[18] M. Weidlich, M. Weske, and J. Mendling, “Change propagation in process models using behavioral profiles,” *IEEE SCC*, pp. 33–40, 2009.

[19] A. Ghose and G. Koliadis, “Model eco-systems: Preliminary work,” *APCCM 2008*, vol. 79, pp. 19–26, 2008.