A New Algorithm for Detecting the Correctness of Merging Operation in Workflow

[1]Homayun Motameni, [2]Somayeh gilani, [2]Fatemeh Zahra naderi and [3]Behnam Barzegar*

[1]Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran
[2]Science and Research branch, Islamic Azad university of Mazandaran, Iran
[3]Department of Computer Engineering, Nowshahr Branch, Islamic Azad University, Nowshahr, Iran
***Corresponding author**: Behnam Barzegar
E-mail: behnam.barzegar@yahoo.com or barzegar@iauns.ac.ir

Abstract: Business process in organization are defined as set of connected operations which have determined start and end, definitive purpose and created certain additive value for organization. Thus, modeling business process and developing it has significant importance on changing organization structure and turning them into the successful one. As Petri Nets are strong modeling tool and have graphical and formal base, we model workflow using Petri Nets. Furthermore, in this paper, we discuss workflows merging and offer a method for merging business process. This leads to decrease in cost and time in large organizations. However, if merge be correct, as there is no official investigation about correctness of merging operation, thus in this paper we aim to present an algorithm using vicinity matrix in order to determine correctness of merging operation.
[Homayun Motameni, Somayeh gilani, Fatemeh Zahra naderi and Behnam Barzegar. **A New Algorithm for Detecting the Correctness of Merging Operation in Workflow.** *Life Sci J* 2012;9(4):1759-1768] (ISSN:1097-8135). http://www.lifesciencesite.com. 268

Keywords: Workflow, Merging, Petri Net, Workflow.

1. Introduction

Processes are set of integration operations which follow definitive and unique purposes and provide facilities for organizations. In recent years, modeling business process and developing it has became one of important strategies for changing organization structures and turning them into the successful organizations with high competition potential.

Business process is set of horizontal processes which are joined to perform various basic actions through them output of related organizations are made. In fact, a business process is net of active connections and well-defined buffers and superior communication which use resources for turning inputs into outputs in order to satisfy customer demands and get their agreement.

A comprehensive approach in direction of business process should record a business process, provide required tools for identifying bottlenecks and bind points and analyzing them, and finally improve business process based on determined purposes. Successful modeling business process depends on choosing appropriate technique for analyzing workflow.

In this paper, we use Petri Nets as a modeling tool, because it displays a process in graphical and formal method and allow us to analyze with more details.

For agile business operation, modern corporations must make frequent business process changes as well as organizational changes through merges and acquisitions. For example, in 2001, Hewlett-Packard Company and Compad Computer Corporation were merged. The company merged offers set of production and services in IT industry and saved cost using merging operation has estimated 2.5 billion dollars in year [6]. Important problems have created for large companies through merging. One of these problems is recognizing correctness of merging operation, cause if merging be incorrect, not only leads to decrease in cost and time, but also increase them. Thus, in this paper we aim to present new algorithm for recognizing correctness of merging operation. Idea of this algorithm has derived from vicinity matrix, due to having mathematical basic, is easy implementations and understandable. We could evaluate correctness of operation before perform merging operation using this algorithm and save in cost and time. As this is new trend, many of merging operations don't save cost and time, and even increase them.

The organization of this paper as follows:

In section 2 we present related works. In section 3 we introduce basic workflow concepts. Then, in section 4 we state mapping workflow concepts onto Petri Nets. In section 5 we will discuss merging workflow and merging methods. In section 6 we analyze merging operation and present an algorithm for recognizing correctness of merging operation. And finally in section 7 we will conclude our text.

2. Related works

Successful modeling business process depends on available appropriate modeling method for analyzing workflow process. There are many analyzing methods, such as workflow diagram, dataflow

diagram, the integrated definitions of function modeling, the extendible markup language (XML), Petri Nets, object-oriented methods, which are used.

In [1] workflow has implemented with UML activity diagram, also investigate the expressiveness and the adequacy of activity diagrams for workflow specification, by systematically evaluating their ability to capture a collection of workflow patterns.

In [7], BPMN introduced as new standard which has developed for modeling business process. This process has formulated in consensus of BPMI Notation working group members, which include main part of business process modeling society. BPMN has several advantages than UML: first, BPMN presents technique for modeling process current which is closer to underused method for business process modeling. Second, UML has steady and complete mathematical foundation which has designed for executive business language, while UML doesn't have this advantage.

As business on Internet needs that business partners exchange information about their business processes in an automated manner, thus authors in [2] has proposed the design for an exchangeable routing language (XLR) using XML syntax. XML (extendible markup language) is a means for trading partners to exchange business data, electronically. Also, XML could support describe workflow process schemas, which through it we could analyze correctness and performance of workflows described in XRL.

As Petri Nets are an established tool for modeling and analyzing processes. One the one hand, they can be used as a design language for the specification of complex workflows. Also, Petri Nets theory provides powerful analysis techniques which can be used to for verify the correctness of workflow procedures.

In [3] a last has implemented workflow concepts using Petri Nets.

In [4] workflow modeling methods such as Petri Nets, WFMC, UML, ANSI and EPC have compared based on criteria such as formal basis, executability, ease visualization, etc. their Study showed that Petri Nets satisfied most of the criteria, and were therefore desirable.

3. Basic workflow concepts

Figure 1 shows that a workflow has three dimensions: (1) the case dimension, (2) the process dimension and (3) the resource dimension. Workflows are case-based, i.e. every piece of work is executed for a specific case. A case is the thing which needs to be processed by following the process definition. Examples of cases are a mortgage, an insurance claim, a tax declaration, an order, or a request for information. Each case has a unique identity and a limited lifetime [3].

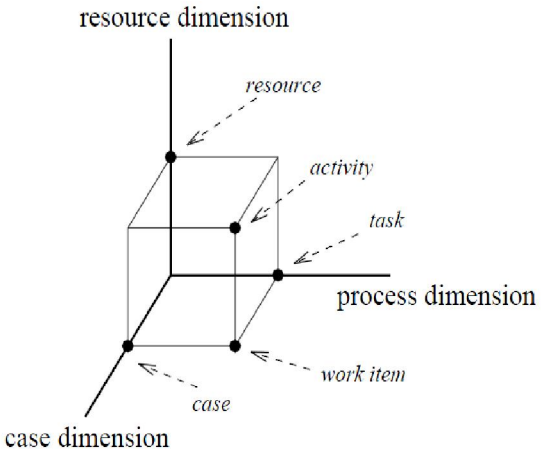

Figure 1: A three dimensional view of a workflow

In the process dimension, it is specified which tasks need to be executed and in what order. A task is a logical unit or a generic piece of a work.

In the resource dimension, the resources are grouped into roles and organizational units.

We can visualize a workflow as a number of dots in the three dimensional view shown in figure1. Each dot represents either a workitem (case+resource) or an activity (case+task+resource).

Since Petri Nets are a process modeling technique and the application is restricted to the first two dimensions, in this paper we focus on the first two dimensions.

4. Mapping workflow concepts onto Petri Nets

In this section, first we introduce Petri Nets and then using it we present a definition for workflow net. Next, we illustrate workflow modeling using Petri Nets with an example.

Petri Net is a mathematical method for modeling and evaluating software productions, which first was introduced by Carl Adam Petri in 1962. Petri Net offers obvious and precious concepts and understandable graphical notations and many analytical techniques. Petri Nets are based on graphs. In fact, idea of graphs caused that Adam Petri achieved to Petri Nets models.

All constructs in a Petri Net can be demonstrated mathematically and furthermore, the formallism can illustrate cases by use of tokens that move through the net (Petereson, 1981). The use of these tokens therefore makes the net executable and very well adapted to simulation (Reisig, 1985). Now, we offer a formal definition for Petri Nets.

Definition (1) Petri Net: Petri Net structure have consisted five components of C= (P, T, I, O, M_0) where:

- P denotes finite set of places;
- T denotes finite set of transitions;
- I denotes set of input functions for net transitions;
- O denotes set of output functions for net transitions; and
- M_0 denotes initial state of net which determine number of tokens in places. Graphicalnotat ions consist three components for business process and now is a formal model for workflow (Van 1998, Dar Adlst [4]).

Definition (2) Petri Net: a Petri Net is a triple (P, T, F):

- P is a finite set of places,
- T is a finite set of transitions (P∩T=Ø)
- $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs (flow relation)

  A place p is called an input place of a transition t if there exists a directed arc from p to t. Place p is called an output place of transition t if there exists a directed arc from t to p. We use •t (t•) to denote the set of input (output) places for a transition t. The notation •p (p•) have similar meanings. [3, 5]

Definition (3) workflow net (wf-net): a Petri Net P = (P, T, F) is a wf-net (workflow net) if and only if:

i. PN has two special places: i and o. place i is a source place: •i = Ø. Place o is a sink place: o• = Ø.
ii. If we add a transition t* to PN which connects place o with i (i.e. •t* = {o} and t*• = {i}), then the resulting Petri Net is strongly connected [3, 5].

A Petri Net which models a workflow process definition is called a workflow net (WF-net) [3, 5].

4.1. modeling workflow whith Petri Net

Tasks are modeled by transitions, conditions are modeled by places, and cases are modeled by tokens [3]. for example in figure 2 [3] models complaint handling process.

An incoming complaint is recorded, then the client and the department affected are contacted (can be done in parallel), afterwards, the data are gathered and a decision is taken either (1) a compensation payment is made, or (2) a letter is sent. Finally, the complaint is filed.

4.2. Routing of cases

Describes the lifecycle of a case, i.e., which tasks need to be performed and in which order. we have four types of routing:

- Sequential
- Parallel
- Conditional
- Iteration

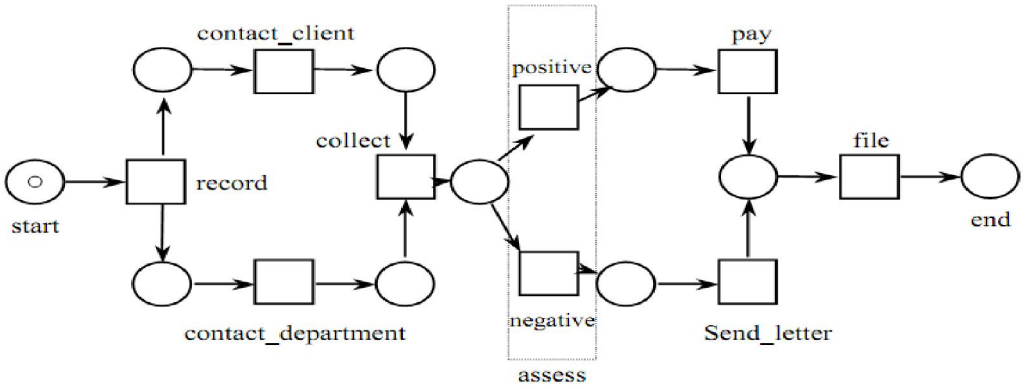

Figure2: modeling complaint handling as a Petri Net

4.2.1. Sequential routing

Sequential routing is used to deal with causal relationships between tasks. Consider two tasks A and B .if task B is executed after the completion of task A, then A and B are executed sequentially [3].

Figure 3: Sequential routing

4.2.2. Parallel routing

Parallel routing is used in situations where the order of execution is less strict. For example, two tasks B and C need to be executed but the order of execution is arbitrary. To model such a parallel routing , two building blocks are used : (1) the AND-split and (2) the AND-join [3].

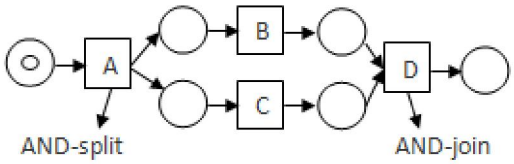

Figure 4: Parallel routing

4.2.3. Conditional routing

To model a choice between two or more alternatives we use two building blocks: (1) the OR-split and (2) the OR-join. Figure 5 [3] shows the situation where task A is followed by either task B or task C, i.e., a choice is made between B and C.

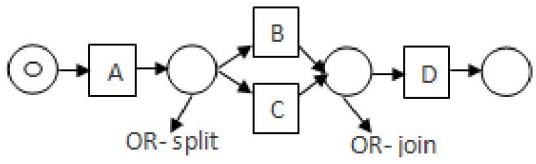

Figure 5: conditional routing

4.2.4. Iteration routing

In the figure 6[3] task B is executed one or more times.

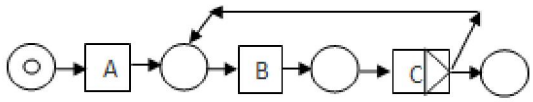

Figure 6: Iteration routing

5. Workflow merging

In this section, we introduce workflow merging concepts and type of merging methods.

5.1. Concepts of workflow merge

We define workflow merge as the process of combining one workflow schema into another and removing redundant steps but keeping all necessary ones [6].

Definition (4) Workflow Merge. When a WF-net PN′ = (P′, T′, F′) is combined with another WF-net PN = (P, T, F), we call the process a workflow merge, if and only if:

(i) The result is a new WF-net PN″ = (P″ ,T″ ,F″)

(ii) $T'' \subseteq T \cup T'$

(iii) $P'' \subseteq P \cup P' \cup P_m$ (where P_m are new merge points)

(iv) $F'' \subseteq F \cup F' \cup (T'' \times P_m) \cup (P_m \times T'')$

We call the merge function as Merge (PN, PN′), and we call PN the primary WF-net and PN′ the secondary WF-net.

According to condition (ii), the merged workflow should not involve any new tasks that are not in the merging workflows; condition (iii) ensures that only result merge points can include new conditions; condition (iv) states that dependencies in the merging workflows should be compliant with the ones in merged workflows [6].

Definition (5) Merge Point. When a primary WF-net, PN′= (P′, T′, F′), is merged with a secondary WF-net, PN= (P, T, F), and the merged workflow is a WF-net, PN″= (P″, T″, F″) a place node such as $p \in PN$, $p' \in PN'$ or $p_m \in PN''$ is called a merge point, if and only if:

(i) If $p \in PN \cap PN''$, $\exists t$ such that $t \in (\cdot p \cup p \cdot) \wedge t \in PN'$, or

(ii) If $p' \in PN' \cap PN''$, $\exists t$ such that $t \in (\cdot p' \cup p' \cdot) \wedge t \in PN$, or

(iii) $P_m \notin PN \cup PN'$

The place nodes where two merging workflows, say wf1 and wf2, are connected are called merge points. Merge points are always in pairs and they are noted as (p/p′) which means that p′ from wf2 will merge with p from wf1 [6].

For example, Fig. 7[6] depicts a merge function Merge (PN, PN′), and the merged workflow is PN″.

Because, in a Petri Net, two place nodes (such as p2 and p0′) cannot be connected directly without a transition node in between.

By eliminating redundant nodes (p2) and auxiliary node (tx) we can reduce PN″ into PN‴. This is a workflow simplification step. We will explain it in next section.

5.2. Type of workflow merging

Whereas workflows have basic process patterns, such as sequential, parallel, conditional, and iterative. In basic merge situations, we assume that two merging workflows contain a single pattern in the merged workflow and two pairs of merge points: A pair of beginning merge points and a pair of ending merge points. within this condition we define sequential, parallel, conditional, and iterative workflow in. In more complex situations, a merged workflow may

contain multiple merge points. However, a complex merge can be represented by combining simple merge patterns.

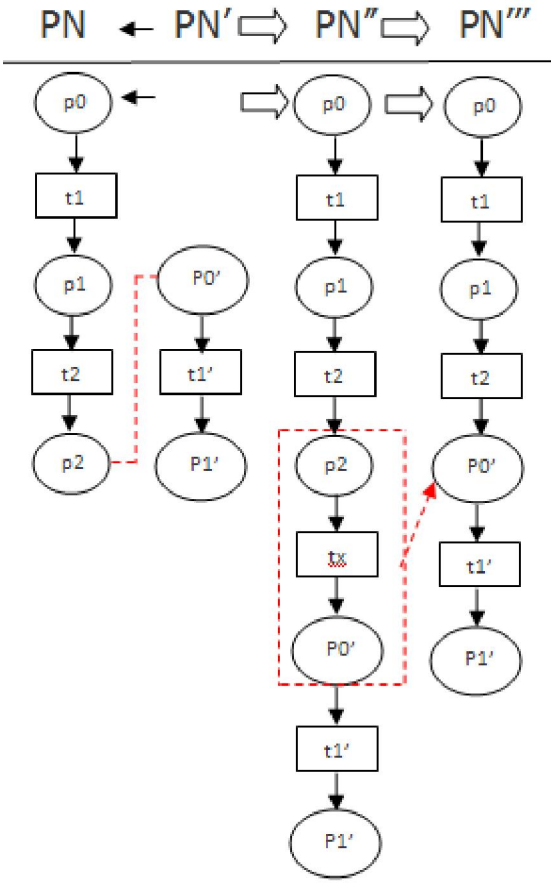

Fig.7. Workflow merge example.

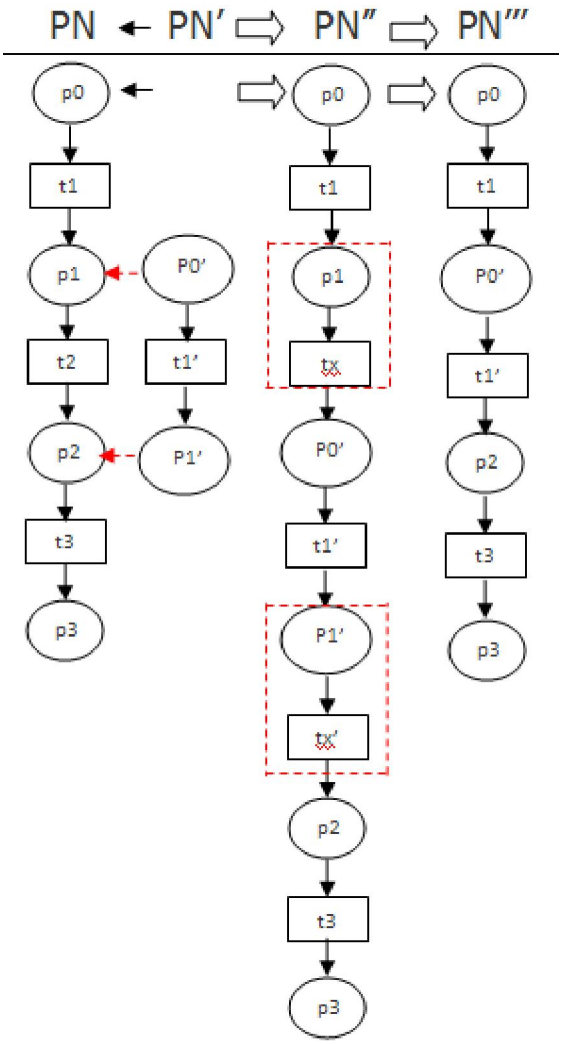

Fig.8.Replacement Merge.

Definition (6) Sequential merge. When two workflows, PN and PN′, merge at merge points (p1/p1′, p2/p2′), if p1 is replaced by p1′ and p2 by p2′, it is a sequential merge. The tasks or steps of PN between p1 and p2 are replaced by the steps in PN′ between p1′ and p2′. In general, no new place nodes are created in a sequential merge [6].

There are two types of sequential merges:

1. Replacement merge 2.Insertion merge.

Fig. 8[6] shows a replacement—Merge_Seq (PN, PN′, p1, p0′, p2, p1′) where t2 is replaced by t1′.

Fig. 9[6] shows an insertion—Merge_Seq (PN, PN′, p1, p0′, p1, p1′) where, in the primary workflow, place p1 is both the start place and the end place.

A sequential merge involves two steps: initial merge and simplification.

In the first step, the merging workflows are combined through two pairs of auxiliary-node sets at the merging points. In Fig. 8, an auxiliary node, tx, is connected to the merge point, p1, while another auxiliary node tx′ is connected to p2. As discussed above, the auxiliary transition nodes are required to connect two place nodes in a Petri net. Between the auxiliary node is the merge region of PN′. The use of the auxiliary nodes in the merge guarantees a sequential relation between the merging workflows.

To simplify and make the workflow nets concise, we need to eliminate the redundant nodes and auxiliary transitions.

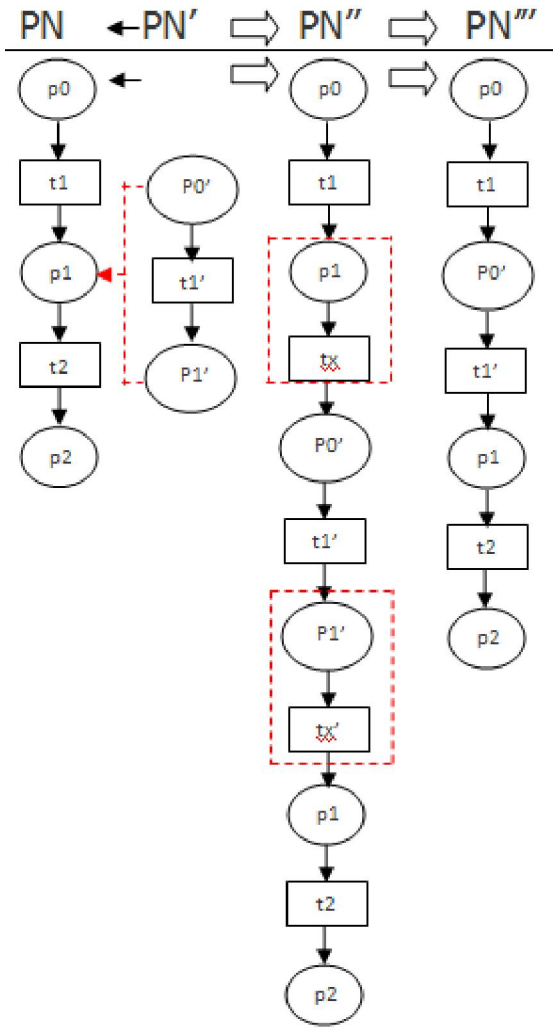

Fig. 9. Insertion Merge.

In both Figs. 8 and 9, PN″ represents the result of the initial merge and PN‴ is the result after simplification. It is easy to see that PN″ and PN‴ are equivalent. However, because the auxiliary nodes are not essential in the final merge result, we provide an algorithm that obtains the merged workflow without involving any auxiliary nodes.

Algorithm (1) Merge_Seq[6].

Algorithm Merge_Seq(PN,PN′,p1,p1′,p2,p2′)

1. Remove p1, the arcs after p1, and the arcs before p1′.
2. Connect •p1 to p1′ (i.e., connect the input transitions of p1 to p1′ with new arcs).
3. Remove p2•, arcs before p2, and arcs after p2′.
4. Connect •p2′ to p2 (i.e., connect all input transitions of place p2′ to the place p2 with new arcs).
5. The process that contains p1′ and p2 is the merged workflow.

Definition (7) Parallel Merge. When two workflows PN and PN′ merge at merge points (p1/p1′, p2/p2′), if, after the merge, p1 and p1′ construct an AND-split and p2 and p2′ construct an AND-join, it is a parallel merge, i.e., PN and PN′ have been connected at points p1/p1′ and p2/p2′, in parallel. In general, no new place nodes are created in a parallel merge.

Algorithm (2) Merge_Par[6].

Algorithm Merge_Par(PN, PN′, p1, p1′, p2, p2′)

1. Remove the arcs before p1′ and connect •p1 to p1′.
2. Remove the arcs after p2′ and connect p2′ to p2•.

Parallel merges are used when the causal order between the merging workflows is not relevant. Fig. 10 [6] illustrates a parallel merge where the order of tasks t2 and t2′ is not relevant.

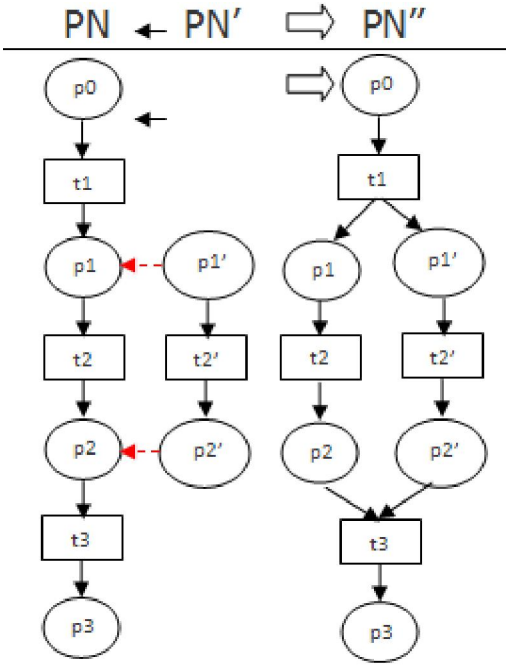

Fig.10.Parallel Merge.

Definition (8) Conditional Merge. When two workflows PN and PN′ merge at merge points (p1/p1′, p2/p2′), if p1 and p1′ construct an OR-split and p2 and p2′ construct an OR-join, it is a conditional merge, i.e., PN and PN′ have been connected at points p1/p1′ and p2/p2′ with additional conditions. A new place called a condition place will be created in a conditional merge [6].

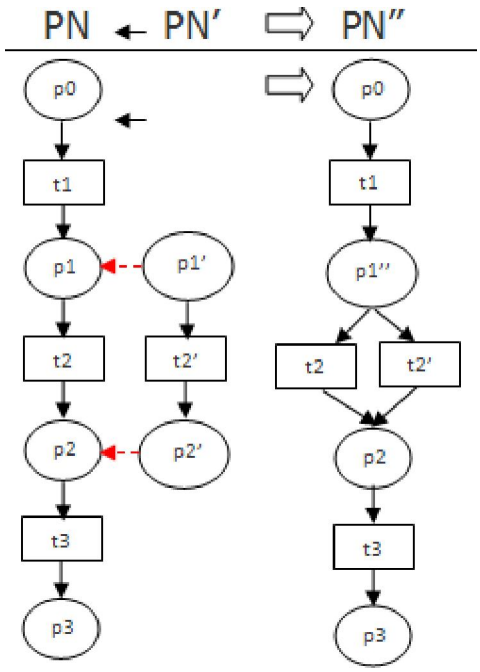

Fig.11.Conditional merge.

Fig. 11[6] is an example of a conditional merge, Merge_Cond(PN, PN′, p1, p1′, p2, p2′, C). In the merged workflow, p1 and p1′ are merged into a new place called p1″ that contains conditions for choosing between tasks t2 and t2′. Because the transitions after place p2′ are not included in the result, the conditions in p2′ are useless in the merged workflow. Therefore, the merge point of the secondary merging workflow is removed.

Algorithm (3) Merge_Cond[6].

Algorithm MergeCond(PN, PN′, p1, p1′,p2, p2′, C)

1. Remove the arcs before p1′, and connect p1′ to p1.
2. Remove the arcs after p2′, and connect p2′ to p2.
3. Modify the conditions in p1 according to new choice conditions C and p1′.

Definition (9) Iterative Merge. When two workflows PN and PN′ merge at merge points (p1/p2′, p2/p1′), if p1 and p2′ construct an OR-join and p2 and p1′ construct an OR-split, it is an iterative merge, i.e., PN and PN′ have been connected at point p1/p2′ and p2/p1′ with additional conditions. A new place will be created in an iterative merge [6].

Algorithm (4) Merge_Iterative[6].

Algorithm Merge_Iterative(PN, PN′, p1, p2′, p2, p1′, C)

1. Remove the arcs before p1′, and connect p2 to p1′.
2. Remove the arcs after p2′, and connect p2′ to p1.
3. Modify the conditions in p2 according to the new choice conditions C and original conditions in p2.

Fig. 12[6] is an example of iterative merge, Merge_Iterative (PN, PN′, p1, p2′, p2, p1′, C). In the merged workflow, p2″ is a new place that contains conditions for choosing between tasks t3 and t4.

Definition (10) Complex Merge. When two workflows PN and PN′ merge at more than two pairs of merge points, it is called a complex merge. A complex merge may involve multiple merge patterns [6].

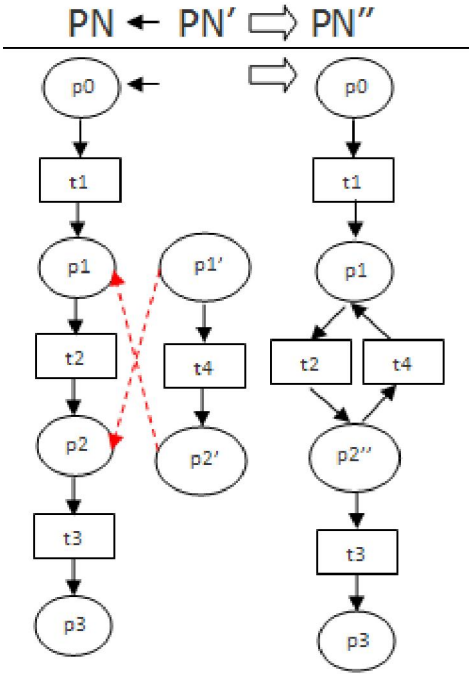

Fig.12.Iterative Merge.

6. Workflow Merge Analysis

Above we discussed various types of functions or operations for merging two workflows. However, without properly chosen merge points and merge functions, two merging workflows cannot yield a sound result, even a syntactically sound one.

Fig. 13[6], Merge_Seq(PN, PN′, p4, p6′, p7, p1′), gives an example of an unsound merged workflow because node p5, after the merge, becomes dangling, and the whole workflow, PN″, is ill structured.

If a workflow merge yields a correct result, we call it a sound merge. On the other hand, in some situations, two workflows cannot be merged correctly, and we call such merges unsound. Fig. 13 shows an unsound merge.

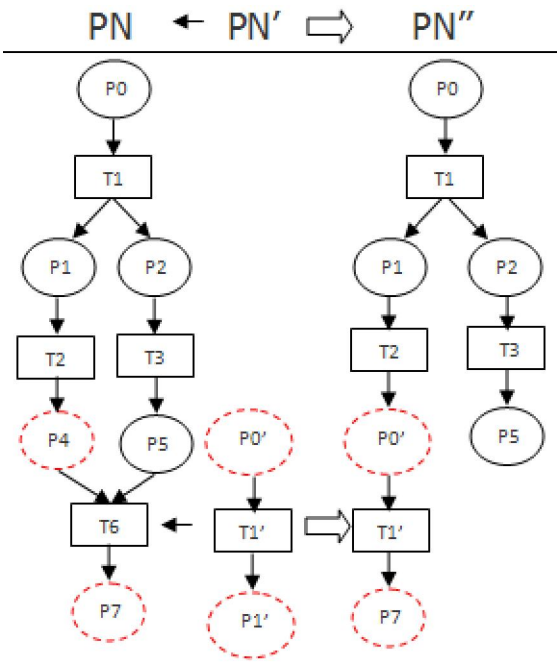

Fig.13.An unsound merged workflow.

Definition (11) Merge Region. When two workflows PN and PN′ are merged at merge points (p1/p1′, p2/p2′), by the operation merge (PN, PN′, p1, p1′, p2, p2′), the sub-process between p1 and p2 (or between p1′ and p2′) is called a merge region. The merge region for a merging workflow, say PN (or PN′), can be obtained through the following algorithm:

1. Remove •p1 and p2• (or •p1′ and p2′•).

2. The sub-process that contains merge points p1 and p2 (or p1′ and p2′) is the merge region for the merging workflow PN (or PN′).

Theorem. If the merge regions of two merging workflows are structured WF-nets, the merged workflow constructed with sequential, parallel, conditional, and iterative merge functions is structured too [6].

By choosing a proper merge point, we can change an unsound merge to a sound one. In Fig. 14, Merge_Seq(PN, PN′, p3, p1′, p5, p2′) results in PN″, which is not well-structured. If we change a merge point of PN from p5 to p4—thus, Merge_Seq(PN, PN′, p3, p1′,p4 ,p2′), the new result PN‴ is well-structured and sound.

Algorithm (5): Merging Correctness Detection.

Merging Correctness Detection Algorithm (PN, PN′, p1, p1′, p2, p2′)

1. t = •p1 ;
2. t′ = p2• ;
3. t″= •p1′ ;
4. t‴= p2′• ;
5. n1 = number of output arcs t;
6. n2 = number of input arcs t′;
7. n3 = number of output arcs t″;
8. n4 = number of input arcs t‴;
9. if (n1≠n2) then incorrect merge;
10. else if (n3≠n4) then unsound merge;
11. else correct merge;

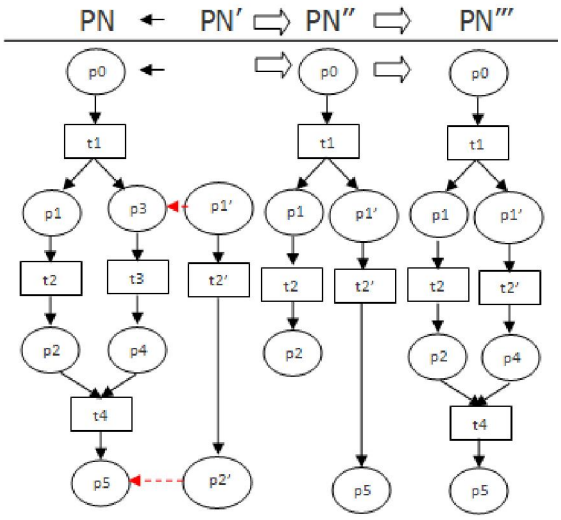

Fig.14. Example of choosing a proper merge point.

In this algorithm, with considering number of output transition arcs before merge start point and number of input transition arcs after merge end point in each merge workflow, we could find out that whether our merge region is well-structured or not, and if it leads to sound merge or not.

For example consider figure14. In PN workflow, number of output transition arcs before merge start point (t1) is equal 2, whereas number of input transition arcs after merge end point is equal zero (there is no transition). Thus, merge region is not well-structured. Now if merge region is changed to (P3/P4), as you can see, because number of output transition arcs before merge start point (t1) is equal to the number of input transition arcs (t4), therefore selected merge region well-structured and would be sound result merge (PN‴).

For convenience it could be used vicinity matrix, such that for highlighting output arc from each transition we could create a vicinity matrix, in which rows indicate transitions and columns show places. If there is an arc from one transition to a place,

corresponding element in vicinity matrix is equal one and otherwise is zero. Last column (SO) indicates sum of output arcs from each transition. Similar to above technique, we create a vicinity matrix for determining number of input arcs towards inside of each transition. If there is an arc from one place to an transition, corresponding element in vicinity matrix is equal one and otherwise is zero. Also, last column (SI) denotes sum of number of input arcs towards inside of each transition. For example consider figure15 shows vicinity matrix PN for figure 14. As SO value in t1 transition equal to SI value in t4 transition, then PN"' merge is correct.

Fig.15.vicinity matrix.

As another example, we explain sample of sound merge in hotel management system in figure 16.

Suppose that workflow of room delivery in at a hotel is such that a customer receive his key room in return for paying stay cost. Here we have two workflows for paying stay cost. In first workflow, customer has his travel expenditures with himself and pay cost of his room directly. In second workflow, customer has credit card for the security reasons and as there is no ATM system in the hotel, customer has to receive money from an outside ATM and then deliver it to hotelier and receive his key room. After opinion sampling from hotel customers, hotel managers found out that for those customers who highly care about security problems create new workflow, by which instead of receiving money from an outdoor ATM, place an ATM in the inside of hotel through it payments directly deposit to hotel account.

In figure 16 we illustrate example of replacement workflow merge, in which merge points should be selected correctly. If our merge be as:

Merge_Seq(PN, PN′, "statement payment", "statement payment" ,"statement has paid", "statement has paid")

Merge is sound, since "statement delivery" transition has two outputs and "key room delivery" has two inputs. According to the algorithm these two are equal and so merge is sound.

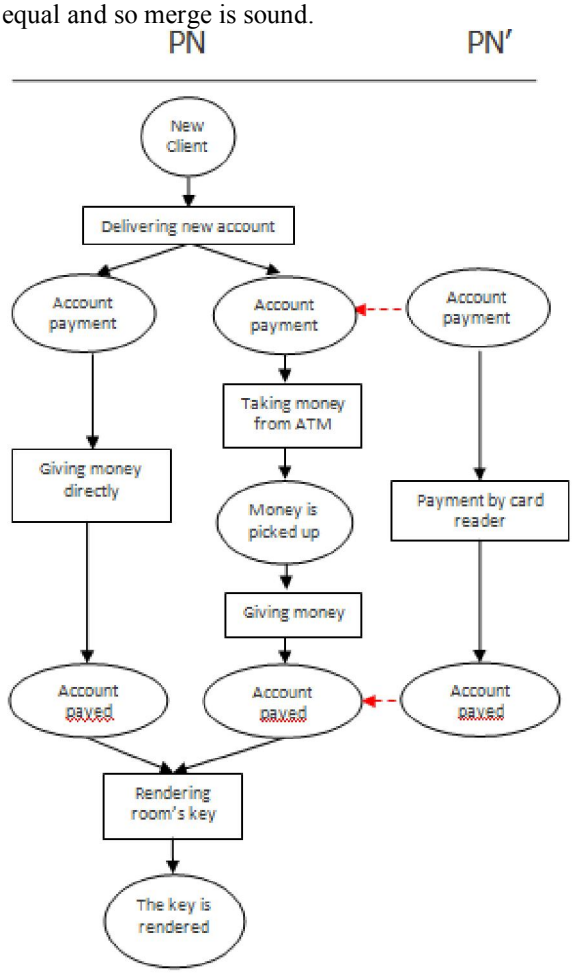

Fig.16.Example of sound merge in hotel management system.

7. Conclusion

In this paper we used Petri Net for modeling workflow, because: (1) a workflow process specified in terms of a Petri Net has a clear and precise definition; (2) Petri Net are a graphical language. As a result, it could be learned easily and intuitively; (3) Petri Net support all the primitives needed to model a workflow process; (4) Petri Net has rigid mathematical basic; (5) Petri Nets have marked using many analyzing techniques. These techniques can be

used to prove properties (safety properties, invariance properties, deadlock and etc). (6) Petri Net provide an independent framework for modeling and analyzing processes. Moreover, we discussed about workflow merge and all types of merge methods, which lead to decreases in cost, time and increase in throughput of huge organizations in the case that merge operation be sound. Merge operation would be sound if two selected merge region be well-structured. Merge points are effective in determining being well-structure of merge region, so by selecting suitable merge points created merge region would be well-structured and thus our merge is sound.

References

[1] M. Dumas, A.H.M.t. Hofstede, "UML activity diagrams as a workflow specification language ", International Conference on the Unified Modeling Language (UML), springer Verlage,Toronto, Canada, 2001.

[2] W.M.P.v.d. Aalst, A. Kumar, "XML based schema definition for support of inter-organizitional workflow", Information systems Research 14 (1) (2003) 23-47.

[3] W.M.P.v.d. Aalst, "The application of Petri nets to workflow management", Journal of Circuits, Systems and Computers 8(1) (1998) 21–66.

[4] A. Dussart,B.A. Aubert, M. Patry, "An evaluation of interorganizational workflow modeling formalisms", Journal of Database Management 15 (2) (2004) 74–104.

[5] Dongsheng Liu , Jianmin Wang, Stephen C.F.Chan,Jiaguang Sun,Li Zhang,"Modeling workflow processes with colored Petri nets",computers in Industry 49(2002) 267-281.

[6] Shuang Sun, Akhil Kumar, John Yen, "Merging workflows: A new perspective on connecting business processes", Decision Support Systems 42 (2006) 844– 858.

[7] Reza samizadeh ,marzieh Chapardar , "Standard introduced BPMN. The modeling of business processes"

8/15/2012