

## A New Method to Detect Useless Service Failure Model in SPN

Mingfeng Zhao<sup>\*1,2,3</sup>, Yajian Zhou<sup>\*2</sup>, Yixian Yang<sup>\*2,1</sup>, We Song<sup>\*3</sup>, Yajun Du<sup>\*3</sup>

<sup>\*1</sup>*School of Computer Science & Engineering, University of Electronic Science and Technology of China, Chengdu, China*

<sup>\*2</sup>*National Engineering Laboratory for Disaster Backup and Recovery, Beijing University of Posts and Telecommunications, Beijing, China*

<sup>\*3</sup>*School of Mathematics & Computer Science technology, Xihua University, Chengdu, China*  
zhaomingfeng81@gmail.com, yajian@bupt.edu.cn, yxy@bupt.edu.cn,  
syl1505@163.com, duyajun@mail.xhu.edu.cn

doi: 10.4156/jcit.vol5.issue3.18

### Abstract

*Service Failure model is described by Stochastic Petri nets. Some of the service failure precisely corresponds to the confusion. Confusion is one of very important concept in the Petri nets. Because it appears to be difficult to obtain a “correct” implementation when emerges confusion in the Petri net system, it is not a good model when exists confusion. When we build some service failure and recovery model based on Stochastic Petri nets, it is very important to avoid some structural design error such as useless service failure structure (siphon and confusion structure). In this paper, we discuss useless service failure structures which are precisely equivalent to structural confusion; an algorithm detection useless service failure structure is presented.*

**Keywords:** *Useless Service Failure, Petri nets, Confusion, Stochastic Petri nets*

## 1. Introduction

Stochastic Petri nets (SPN) which is based on classical Petri nets, has emerged over years as a favored approach for performance and dependability modeling and evaluation [1, 2]. From a system modeler viewpoint, SPN enable attention to be focused on the logic of the system, especially on the interactions and dependencies between classes of components, and to handle in a modular way components within each class. It is a good choice for applying SPN to analysis dependability of systems. The System which can provide reliable service capability is just called dependability. Dependability is a measure of a variety of systems for important target of quality of service, and greatly affects system performance. When using SPN to analyze system dependability, it must establish the dependability of the system model firstly. It can be obtained through the following three stages—first of all to establish the system performance model, which describes an ideal system situation of no failure occurrence; secondly, to build a failure-recovery model or fault-tolerant model, which describes a realistic system situation of possible failure of resources (mostly service failure); finally, the combination of these to establish a dependability system model. However, we often overlook the important structure theory of the Petri nets when make the establishment of the service failure-recovery or fault-tolerant model, it leads to the establishment of the dependability model is error or has nothing any meaning. Confusion plays an important role in building service failure model (usually useless service failure model).

When concurrency and conflict is mixed together, it is impossible from the same reachability graph to determine the final state whether there had been conflict, the system of this feature is called confusion. Confusion exists because the division system and the system environment are not correct [3]. So far, the discussion of the confusion is rarely seen. Rozenberg [4,5] first puts forward the concept of confusion in the Elementary Net Systems, Aalst[6,7,8] also takes into account the existence of confusion in the Workflow Nets(WF-nets) which

is based on classical Petri nets, and discuss some static structure of the WF-nets. For this purpose, they introduce three interesting subclasses of WF-nets: Free-choice WF-nets, Well-structured WF-nets, and S-coverable WF-nets to avoid the existence of confusion. Thiagarajan[9] gives a more detailed formal definition of confusion and divided into two categories—conflict-increasing confusion, conflict -decreasing confusion. Bolton [10, 11] gives a way to capture conflict and confusion when the Petri nets model is converted to Communication Sequential Process (CSP) in the Process Algebra. Zhao [12] have made a detailed discussion of confusion, and a detection confusion method has been presented. In summary, there is not a specific application in related areas for confusion.

This paper is precisely devoted to a (preliminary) exploration of how to take advantage of structure theory (structural confusion) of the Petri nets to detect useless service failure model of dependability models based on SPN.

The paper is composed of four sections. In the first section, some concepts are presented for correct understanding some terminology. The second section addresses the relations of confusion, structural confusion and useless service failure, one lemma and corollary is presented and proofed. The third section, an algorithm detection useless service failure structure is presented based on second section. Finally, the fourth section summarizes the approach in the paper.

## 2. Concepts

### Definition 1<sup>[9]</sup>

Let  $(PN, c_{in}) = (B, E; F, c_{in})$  be an EN system,  $(B, E; F)$  is a Petri nets and  $c_{in} \subseteq B$ , is called the initial case of  $(PN, c_{in})$ . Let case  $c \in C = [c_{in}]$  and let  $e \in E$  be such that  $c [e]$ . The conflict set of  $e$  at  $c$ , denoted  $cfl(e, c)$ ,  $cfl(e, c) = \{e' \in E \mid c[e'] \wedge \text{not } c[\{e, e'\}]\}$ .

Thus the conflict set of  $e$  at  $c$  is the set of all events that are in conflict with  $e$  at  $c$ .

### Definition 2<sup>[9, 12]</sup>

Let  $c \in C = [c_{in}]$  and let  $e_1$  and  $e_2$  be two distinct events in  $E$  such that  $c [\{e_1, e_2\}]$ . The triplet  $(c, e_1, e_2)$  is a confusion (at  $c$ ) if  $cfl(e_1, c) \neq cfl(e_1, c_2)$ , where  $c [e_2] > c_2$ . We say that  $(PN, c_{in})$  exists confusion at  $c$  iff there is a confusion at  $c$ .

Let  $\gamma = (c, e_1, e_2)$  be a confusion and let  $c [e_1] > c_2$ .

$\gamma$  is a conflict-increasing confusion (abbreviated **ci** confusion) iff  $cfl(e_1, c) \subset cfl(e_1, c_2)$ .

$\gamma$  is a conflict-decreasing confusion (abbreviated **cd** confusion) iff  $cfl(e_1, c) \supset cfl(e_1, c_2)$ .

$\gamma$  is a neither conflict-increasing nor conflict-decreasing confusion (abbreviated **ncincd** confusion) iff  $cfl(e_1, c) \not\subset cfl(e_1, c_2) \wedge cfl(e_1, c_2) \not\subset cfl(e_1, c) \wedge cfl(e_1, c) \neq cfl(e_1, c_2)$ .

### Definition 3<sup>[12]</sup>

$(PN, c_{in})$  exists confusion at case  $c$ , let  $c \in C = [c_{in}]$ . when erase case  $c$ , just that have no any token in the  $(PN, c_{in})$ , in the case, PN exists structural confusion. As shown in Fig 1 and Fig 2(a).

### Definition 4<sup>[12]</sup>

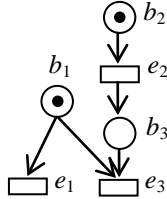
If PN exists structural confusion, then

It exists basic structural **ci** confusion iff  $\exists e_1, e_2 \in E, (e_1 \cup e_2) \cap (e_2 \cup e_1) = \emptyset, e_1 \cap (e_2) \neq \emptyset$ ;

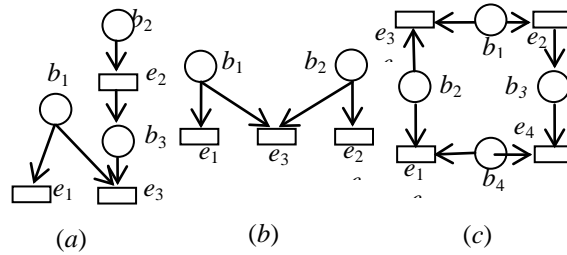
It exists basic structural *cd* confusion iff  $\exists e_1, e_2 \in E, (e_1 \cup e_1) \cap (e_2 \cup e_2) = \emptyset, (e_1) \cdot \cap (e_2) \cdot \neq \emptyset$ ;

It exists basic structural *ncinced* confusion iff  $\exists e_1, e_2 \in E, (e_1 \cup e_1) \cap (e_2 \cup e_2) = \emptyset, (e_1) \cdot \cap (e_2) \cdot \neq \emptyset \wedge (e_1) \cdot \cap (e_2) \cdot \neq \emptyset$ .

According to the above definition, we can interpret it through the corresponding sub-net as shown in Fig 2.



**Fig.1** The subsystem which exists *ci* confusion

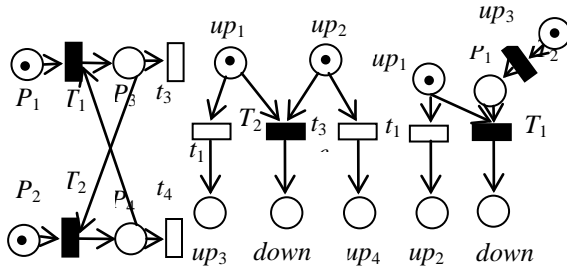


**Fig.2** The graph of basic structural confusion (a) Basic structural *ci* confusion; (b) Basic structural *cd* confusion; (c) Basic structural *ncinced* confusion.

**Definition 5**

Useless Service Failure is defined as the incorrect design of system architecture and system errors caused by the corresponding operation in the process of building dependability system model based on SPN, just as Fig 3~5.

We discuss two kinds of Useless Service Failure in the paper [1].



**Fig. 3** Siphon structure,  
**Fig.4** Conflict structure A  
**Fig.5** Conflict structure B (Sequentially)

The Siphon structure of Fig.3 leads to services  $T_1$  and  $T_2$  (called instantaneous transitions in the SPN) cannot be in an active state. Conflict structure A of Fig.4 describes the two services start at the same time (instantaneous transition  $T_2$  enabled) leads to two services  $t_1$  and  $t_3$  (called time transitions in the SPN) failure. Conflict structure B of Fig.5 describes the resource of the ongoing services  $t_1$ , is seized by other services  $T_1$  and  $T_2$ , and leads to services  $t_1$  failure.

### 3. Relation of Confusion and Useless Service Failure

Siphon is very important concept in the Petri nets. When Useless Service Failure is a Siphon, we can use Petri nets theory to eliminate it. For example, in Figure 3, we can add the corresponding token to Place  $P_3$  or  $P_4$  to remove the siphon between the two events  $T_1$  and  $T_2$ . So, the next section we only discuss Useless Service Failure with conflict structure A and B (abbreviated Useless Service Failure).

#### 3.1 Confusion and Useless Service Failure

According to the definition 2, 3, 4, and 5, we give the following proposition.

**Proposition 1.** Useless Service Failure is a subclass of Confusion.

Proof. Because Stochastic Petri nets (SPN) is built on classical Petri nets, for Useless Service Failure, when the instantaneous transition  $T_2$  of Conflict structure A and instantaneous transition  $T_1$  and  $T_2$  of Conflict structure B in the SPN are converted to normal transition in the Petri nets, Conflict structure A is precisely *ci* confusion and Conflict structure B is precisely *cd* confusion. If the combination of conflict structure A and conflict structure B, it is precisely *ncinced* confusion. Hence, Useless Service Failure is a subclass of Confusion.

#### 3.2 Structural Confusion and Useless Service Failure Structure

It is not a good model when exists confusion, but we cannot directly know the existence of confusion because it corresponds closely related to the initial case and reachability case. Therefore, we need to research the structural confusion and useless service failure structure. We give the following proposition and Lemma.

**Proposition 2.** Useless Service Failure Structure is equivalent to Structural Confusion.

Proof. When discuss the structure theory, it will not exist instantaneous transition, time transition and case. According to the definition 3 and 5, we can conclude that Useless Service Failure Structure is equivalent to Structural Confusion.

**Lemma 1**<sup>[12]</sup>. (PN,  $c_{in}$ ) exists confusion at case  $c$ , let  $c \in C = [c_{in} >$  iff PN at least exists basic structural *ci* confusion, basic structural *cd* confusion or basic structural *ncinced* confusion.

Therefore, we discuss the structural confusion just need to discuss the three categories of basic structural confusion.

**Lemma 2.** PN exists basic structural *ncinced* confusion iff  $\exists e_1, e_2 \in E$ , for  $e_1$  and  $e_2$ , PN exist both basic structural *ci* confusion and basic structural *cd* confusion.

Proof. If PN exists basic structural *ncinced* confusion, according to the definition 4,  $\exists e_1, e_2 \in E$ ,  $(e_1 \cup e_1)' \cap (e_2 \cup e_2)' = \emptyset$ ,  $(e_1)' \cap (e_2)' \neq \emptyset \wedge (e_1)' \cap (e_2)' \neq \emptyset$ . For one thing, from the definition of basic structural *ci* confusion, we can see that basic structural *ci* confusion is a subset of basic structural *ncinced* confusion, but basic structural *ci* confusion in the definition of  $(e_1)' \cap (e_2)' \neq \emptyset$  does not affect the definition of basic structural *ncinced* confusion. Therefore, there is basic structural *ncinced* confusion must have basic structural *ci* confusion. For another, let  $(e_1)' \neq \emptyset$ ,  $(e_2)' \neq \emptyset$ , so have  $e_1 \neq \emptyset$ ,  $(e_2)' \neq \emptyset$ ; let  $(e_1)' \cap (e_2)' \neq \emptyset$ . Clearly, there must be structural conflict between  $e_1$  and  $(e_2)'$ , that is to say,  $e_1 \cap (e_2)' \neq \emptyset$ , it is precisely the definition of basic structural *cd* confusion. Therefore, there is basic

structural **ncincd** confusion must have basic structural **cd** confusion. For the same transitions of  $e_1$  and  $e_2$ , so sufficiency is proved.

Let  $e_1, e_2 \in E$ , for  $e_1$  and  $e_2$ , PN exist both basic structural **ci** confusion and structural **cd** confusion. Then  $e_1 \cap (e_2) \neq \emptyset$  for basic structural **ci** confusion, clearly, there must be structural conflict between  $e_1$  and  $(e_2)$ , let  $e_3 \in (e_2)$ , make  $e_1 \cap e_3 \neq \emptyset$ ; for  $(e_1) \cap e_3 = e_3 \neq \emptyset$ , so have  $e_3 \in (e_2)$ ,  $(e_1) \cap (e_2) \neq \emptyset$ .  $(e_1) \cap (e_2) \neq \emptyset$  for basic structural **cd** confusion, and both have  $(e_1 \cup e_1) \cap (e_2 \cup e_2) = \emptyset$ , it is precisely the definition of basic structural **ncincd** confusion. Therefore, PN exist basic structural **ncincd** confusion.

According to Lemmas, propositions and definitions above, we can conclude following corollary.

**Corollary 1.** Let PN be a service failure model of dependability models based on SPN. If PN has basic structural **ci** confusion or basic structural **cd** confusion at least, then PN is a Useless Service Failure model.

#### 4. Detection Useless Service Failure Structure Algorithm

When we decide to detect useless service failure structure model, according to corollary1 and definition5, we only need to research whether exists the structure of Siphon, basic structural **ci** confusion or basic structural **ci** confusion in the service failure model of dependability models based on SPN. That is to say, we only need to research the sub-matrix in the correlation matrix **H** just as Fig.5. Considering the uncertainty of label sequence in the **H** about row and column, sub-matrix A1, A2 and A3 Rows and columns are interchangeable respectively. **H** is decided by the static structure of the Useless Service Failure model, which column represents Place (assumes  $m$ ) and row represents Transition (assumes  $n$ ). it is defined as follow, “1” indicates the existence of a directed adjacent path transition  $e$  to place  $b$ , “-1” indicates the existence of a directed adjacent path place  $b$  to transition  $e$ , “0” indicates the nonexistence of a directed adjacent path between place  $b$  and transition  $e$  in the SPN model.

$$A_1 = \begin{bmatrix} -1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix} \quad A_2 = \begin{bmatrix} -1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix} \quad A_3 = \begin{bmatrix} 0 & -1 \\ 0 & -1 \\ 1 & -1 \\ -1 & 1 \end{bmatrix}$$

**Fig.6** Three kinds of sub-matrix related with useless service failure structure

Now, an algorithm to detect Useless Service Failure Structure is proposed in the following part.

*Input:* correlation matrix **H** of service failure model

*Output:* Whether or not have Useless Service Failure Structure

*Process:* **Step1.** Search in the correlation matrix **H** and make some elementary transformation on the rows and columns respectively.

**Step2.** If search the isomorphic sub-matrix  $A_1$

Then output “It exists useless service failure structure because of structural **ci** confusion”

**Step3.** If search the isomorphic sub-matrix  $A_2$

Then output “It exists useless service failure structure because of structural **cd** confusion”

**Step4.** If search the isomorphic sub-matrix  $A_3$

Then output “It exists useless service failure structure because of structural Siphon”

else output “It has not useless service failure structure”.

As the matrix of rows and columns elementary transformation and search can be completed in polynomial time, so the algorithm is a polynomial time algorithm.

#### 5. Conclusion

In this paper, we discuss the relations of Confusion and Useless Service Failure, Structural Confusion and Useless Service Failure Structure. An algorithm to detect Useless Service Failure Structure by means of structure theory of Petri nets is presented. This will make a good reference and suggestion to establish a correct service failure model of dependability models based on SPN, and to avoid the existence of Useless Service Failure model.

## 6. Acknowledgement

The authors would like to thank National S&T Major Program (No.2009ZX03004-003-03) and Educational Commission of Sichuan Province of China (No.08ZA029)

## 7. References

- [1] Chuang Lin, Yuanzhou Wang, Yang Yang, et al. Research on network dependability analysis methods based on stochastic Petri net[J]. Acta Electronica Sinica,2006,34(2): 322-332(in Chinese)
- [2] Laprie J C, KaanicheM , Kanoun K Modeling computer systems evolutions: non-stationary processes and stochastic Petri nets application to dependability growth, Petri Nets and Performance Models [A ].1995, Proceedings of the Sixth International Workshop on 3-6 Oct[ C ]. 1995: 221~230
- [3] Chong-Yi YUAN. Petri net theory and application [M]. Publishing House of Electronics industry, 2005. 32~38 (in Chinese)
- [4] G. Rozenberg, P. S. Thiagarajan. Petri Nets: Basic Notions, Structure and Behavior[C]. LNCS224, 1986. 585~668
- [5] G. Rozenberg, J. Engelfriet: Elementary Net Systems[C]. In W. Reisig, G. Rozenberg (eds.):Lectures on Petri Nets I: Basic Models. LNCS 1491, 1998. 12~121
- [6] W. M. P. van der Aalst. Finding Control-Flow Errors Using Petri-Net-Based Techniques[M].Business Process Management: Models, Techniques, and Empirical Studies, volume1806 of LNCS, Springer-Verlag,Berlin, 2000.161~183
- [7] W. M. P. van der Aalst and Kees Van Hee. Workflow Management-Models, Methods, and Systems [M]. MIT Press, 2002. 210~226
- [8] W.M.P. van der Aalst. Verification of Workflow Nets[C].18thInternationalConference, ICATPN'97.volume1248 of LNCS, Springer-Verlag,Berlin,1997.407~426
- [9] P. S. Thiagarajan. Elementary Net Systems[C].LNCS, volume 254, Springer-Verlag, 1986. 26~59
- [10] Bolton, C: Adding conflict and confusion to CSP[C]. In: Fitzgerald, J.A., Hayes, I.J.,Tarlecki, A. (eds.) FM 2005. LNCS 3582, Springer, Heidelberg, 2005. 205~220
- [11] Bolton, C: Capturing Conflict and Confusion in CSP[C]. IFM 2007, LNCS 4591, Springer, 2007. 413~438
- [12] Ming-Feng ZHAO, Wen SONG, Yi-Xian YANG. Confusion detection based on Petri-net[J]. Computer Research and Development, volume 45, 10, 2008,1631~1637(in Chinese)