American Journal of Operations Management and Information Systems
2019; 4(2): 48-56
http://www.sciencepublishinggroup.com/j/ajomis
doi: 10.11648/j.ajomis.20190402.11
ISSN: 2578-8302 (Print); ISSN: 2578-8310 (Online)

Detecting FNE in Sound Free-choice Petri Net with Data

Fang Zhao

Department of Computer Science, Tongji University, Shanghai, China

Email address:
1012377428@qq.com

To cite this article:
Fang Zhao. Detecting FNE in Sound Free-choice Petri Net with Data. *American Journal of Operations Management and Information Systems.* Vol. 4, No. 2, 2019, pp. 48-56. doi: 10.11648/j.ajomis.20190402.11

Received: April 24, 2019; **Accepted**: May 29, 2019; **Published**: June 12, 2019

Abstract: Nowadays, the development of a third-party service (Express industry) and a third-party payment (Alipay) are very fast in online shopping. Despite there are many technologies to detect control flow errors in business process, the soundness verification in data flow is very hard. To support the design of a workflow, we usually consider the correct control flow structure. However, information about data flow should also be ensured correct. The operation of the system may suffer some external attacks, which makes the task change the read and write operations, which result in changing of control flow structure which would lead to the emergence of unusual system. As a result, our approach provides a new technology to analysis the correctness of sound free-choice Petri net with data (SCDN). With the strong concealment of this attack, the system may suffer false-negative data flow errors (FNE), which would bring some loses to the participants. On the basis of behavioral profiles (BP), redundant data flow errors (RDE) and missing data flow errors (MDE), we provide the theory of FNE to demonstrate the stability, effectiveness and adaptation of our detection methods. Finally, a real E-commerce business system is used to illustrate the practicability of the method provided in this paper.

Keywords: SCDN, FNE, BP, RDE, MDE

1. Introduction

In recent years, the Internet technology has been widely used in stimulating the growth of E-commerce. I-Research Consulting statistics show that the transaction volume of Taobao reached 213.5 billion yuan in the Double Eleven of 2018. The E-commerce transaction keeps growing 30 percent to 40 percent every year. The advantages of tools such as laptop, mobile phone, iPad are low prices, more choices, never need to leave home for shopping make a contribution to the development of E-commerce.

When the design of the business process model can correctly reflect the system work without any control flow, data flow anomalies would get all participants' satisfaction, such as vendors, customers, express delivery, and trusted third party security interaction in order to achieve an electronic trading [1]. Designing a workflow model is also a big challenge and error-prone even for experienced process designers. For cost-efficient, rapid development of the design process, detecting errors in the design of system phase is more important than testing at run time. But some methods could not be understood easily by the system designers due to the lack of graphical language's description. Petri net as a suitable tool for modeling a real business process.

The detecting of control flow errors are basing on the analysis of reach-ability [2-4], live-ness [5], deadlock [6], lack of synchronization, the long cycle without reference model and so on [7-9]. The emergence of these anomalies caused by the incorrect link between the transitions in control flow. Verify the correctness of the control flow has become an important topic in today's research. In the last 23 years, many analysis techniques have been developed to analysis the process models. Languages like Business Process Modeling Notation (BPMN) [10-12], UML activity diagrams [5], [13-15] and extended Event-driven Process Chains (eEPCs) [16] et al.

Under the premise of sound free-choice Petri nets [17], existing approaches based on the notion of behavioral profile (BP) to analysis the model [18]. Such a profile is made up of three relations basing on the weak order between transitions. These relations could be used to detect the control flow errors. The BP theory has a strong application scope as it is not sensitive than trace equivalence and bi-simulation [19, 20].

In the process of workflow execution, the role of the data flow is becoming more and more important. With data in the system, we could make the choice of some important path.

The data flow anomalies of the system sometimes cause the change of control flow structure. Data flow stresses what kinds of data need to be as the input /output of transition. The limitation of data flow could be used to analyze the dependencies between different transitions. If the system lack of data flow information, it would be considered to be nondeterministic and unfair. In some scientific research, some scholars, such as the Taverna, Triana and Kepler think the data flow is more important than the control flow [21]. Some data play as an input of workflow task and only the workflow task should satisfy the input conditions could it be executed effectively and produces output. The transitions which are in exclusiveness relation need one of the tasks meets data input conditions and generate an execution path. Data flow could be shown using document-driven workflow and Meta-graphs [22, 23].

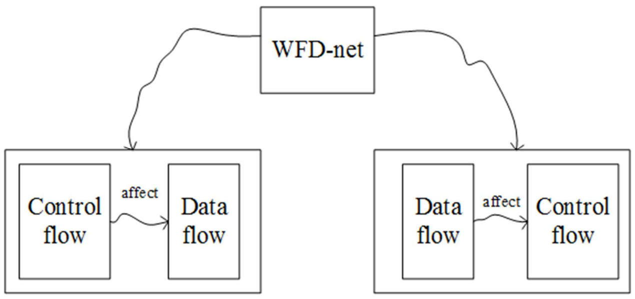

Figure 1. *The relations between control flow and data flow.*

Up till now, as I know, no techniques have yet been used to detect of control flow and data flow errors simultaneously. At the beginning of this paper, we focus on the data flow testing. Provide two kinds of data flow errors: redundant data flow errors (RDE) and missing data flow errors (MDE). On the basis of these errors, there comes the definition of false negative data flow errors (FNE). This error affects the control flow structure of the system. The control flow errors and data flow errors are interdependent and we analysis them without a break until the end.

In order to remove the errors in the system, in 2006, Sherry X. Sun et al provide three basic types of data flow errors, namely the missing data flow errors, conflicting data flow errors, and redundant data flow errors [13]. Then, in 2007, M Hema Sundari and others extend and generalize the study and define the missing data flow errors, redundant data flow errors, and lost data flow errors [24]. In 2008, basing on the input and output data of each task in a business process, Sherry X. Sun and J. Leon Zhao could analysis the dependencies among different transitions [25]. In 2009, Nikola Trcka et al put forward six kinds of data flow errors, namely missing data flow errors, strongly redundant data flow errors, weakly redundant data flow errors, strongly lost data flow errors, weakly lost data flow errors and inconsistent data flow errors [26, 27]. After then, in 2010, Hema S. Meda and Anup Kumar Sen provide missing data flow errors, inconsistent data flow errors, lost data flow errors and redundant data flow errors and present a graph traversal algorithm called GTforDF for detecting data flow errors in unstructured and nested workflow, also illustrate the operation on realistic examples [21]. This was different from the theory provided by Sherry X. Sun et al in 2006. A task makes use of existing data to generate a new data item. The sequencing of tasks is derived from the input/output data analysis of tasks. In 2014, Divya Sharma et al point out that there are some methods to classify the data flow errors and repair the data flow errors automatically [28]. In 2017, Mariusz Dramski indicates that the missing data flow errors in the process of transmission event logs and put forward some methods to restore the missing data [33]. However, they haven't considered the control flow and data flow could affect each other.

The complex links of control flow and data flow may bring a lot of problems in the process of electronic commerce, such as violation of fairness, so it is necessary to test them at the same time. Even if the formal methods of data flow analysis is presented in the literature [21, 26-30], but there is still a challenge in converting the formalized method to the design tools. The ADEPT flex tools supports a limited set of checks for the accuracy of the data flow, mainly focus on dynamic changes in workflow models [31].

In this article, we use the data flow analysis as a navigation of the right workflow design. In order to describe the problem in this article more clearly, the data flow could only be read and wrote, and could not be destroyed. The contribution of this paper is mainly embodied in the following aspects: Firstly, we define the sound free-choice Petri nets, weak order relation, behavioral profile; Secondly, we give some definitions about the data flow, like read and write operations, redundant data flow errors and missing data flow errors, we use them to detect false negative data flow errors. We haven't introduced lost data flow errors and inconsistent data flow errors, about these errors please refer to reference [26-30].

In order to get the correct workflow [32, 33], we put forward two kinds of algorithms used in model checking:

1. Base on the false negative data flow errors to detect the data flow anomalies.
2. Use the behavior profile to detect the control flow anomalies. Our method provides effective guarantee for the correctness of the system design, and through the real case to support the given algorithm.

For example, in Figure 7 and Figure 8, in order to make track of purchased product effectively, the transition C (sends the delivery information), which produces an output of the express company distributes, must precede the transition S receives the express arrival notice, which uses S as input. If claims S occurs before C, a data-flow error would occur, like redundant data flow errors and missing data flow errors. Obviously, this kind of error could be detected by existing system. Our method could be used to detect and eliminate some of the data flow errors, and puts forward a new kind of data flow error to detect model's anomaly more deeply.

In this article, we use real E-business process models to illustrate the FNE and control flow errors caused by FNE. The rest parts of this article are arranged as follows. Section 2 introduces some basic concepts. Section 3 presents the algorithm about detect the data flow anomaly based on FNE.

Section 4 gives a case study in E-commerce system and conclusions and future works are given in section 5.

2. Preliminaries

Petri nets are graphical languages for modeling concurrent and distributed systems. This part presents some basic concepts about Petri nets. Our concepts of Petri nets comprise of places, transitions and flow relations. Graphically, places are represented by circles, transitions are denoted as rectangles, as for flow relations we use arrows to denote them. For more details, please refer to these papers [5, 18].

1) Definition 1 (Sound free-choice Petri nets, SCN). Let $\mathrm{SCN} = (\mathrm{P}, \mathrm{T}, \mathrm{F}, M_0)$ be a sound free-choice Petri net if.
2) P is a limited non-empty set of places, T a limited non-empty set of transitions, $\mathrm{P} \cap \mathrm{T} = \emptyset$. M_0 is the initial identification of P.
3) $\mathrm{F} \subseteq (\mathrm{P} \times \mathrm{T}) \cup (\mathrm{T} \times \mathrm{P})$ is the flow relationship in SCN.
4) $\forall \mathrm{x} \in \mathrm{P} \cup \mathrm{T}$, ${}^{\cdot}x = \{y|(y \in P \cup T) \cap (y, x) \in F\}$ is the pre-set of x, $x^{\cdot} = \{y|(y \in P \cup T) \cap (x, y) \in F\}$ is the post-set of x.
5) If x is the source place, then ${}^{\cdot}x = \emptyset$; If x is the sink place, then $x^{\cdot} = \emptyset$, $\{t|t \in ({}^{\cdot}P/P_0\} \subset T$, $\{p|p \in ({}^{\cdot}T) \cup (T^{\cdot})\} \subset P$.
6) $\mathrm{dom(F)} \cup \mathrm{cod(F)} = \mathrm{P} \cup \mathrm{T}$, $\mathrm{dom(F)} = \{x \in P \cup T|\exists y \in P \cup T, (x, y) \in F\}$, $\mathrm{cod(F)} = \{x \in P \cup T|\exists y \in P \cup T, (y, x) \in F\}$.
7) Without circulation.
8) Without dead-lock.
9) Without live-lock.

When there is a marking in the sink place, there are no markings in the inner places.

Therefore, the criterion of soundness lies in liveness and boundedness, which requires a SCN always terminate and does not have dead transitions. In fact, if a SCN is sound, then the short-circuit net is live and bounded [30]. Example: Figure 2 is a sound free-choice Petri net (SCN).

Definition 2 (Weak Order Relation, WOR). [16] Let $\mathrm{SCN} = (\mathrm{P}, \mathrm{T}, \mathrm{F}, M_0)$ be a sound free-choice Petri net. The weak order relation $\lambda \in \mathrm{T} \times \mathrm{T}$ contains all pairs of transition (x, y), there exists a firing sequence $\sigma = t_1, \cdots, t_n$ with $(\mathrm{N}, [\mathrm{i}])[\sigma >$, $\mathrm{j} \in \{1, \cdots, \mathrm{n}-1\}$, $\mathrm{j} < \mathrm{k} \leq \mathrm{n}$, such that it holds $t_j = x, t_k = y$.

On the basis of sound free-choice Petri nets and weak order relation, there comes the following definition. We know the cyclic structures have a substantial impact on the behavioral relations. For example, two transitions which are in exclusiveness relations inside a cycle may be in interleaving relations. Therefore, our definition of behavior profile is based on sound free-choice Petri nets. (SCN)

Definition 3 (Behavioral profile, BP). [16] For a sound free-choice Petri net $\mathrm{SCN} = (\mathrm{P}, \mathrm{T}, \mathrm{F}, M_0)$, $T' \subset T$, $(\mathrm{x}, \mathrm{y}) \in (T' \times T')$ where there are three kinds of the relations:

1) Strict order relation $\rightarrow$, iff $\mathrm{x} \succ \mathrm{y}, \mathrm{y} \nsucc \mathrm{x}$.
2) Exclusiveness relation $+$, iff $\mathrm{x} \nsucc \mathrm{y}, \mathrm{y} \nsucc \mathrm{x}$.
3) Interleaving order relation $\parallel$, iff $\mathrm{x} \succ \mathrm{y}, \mathrm{y} \succ \mathrm{x}$.

If $\mathrm{x} \nsucc \mathrm{y}, \mathrm{y} \succ \mathrm{x}$, then the relation of x and y is the inverse strict order relation, denoted as $\mathrm{x} \rightarrow^{(-1)} y$. The relations above comprise the behavioral profile of a SCN, denoted as $\mathrm{BP} = \{\rightarrow, +, \parallel\}$.

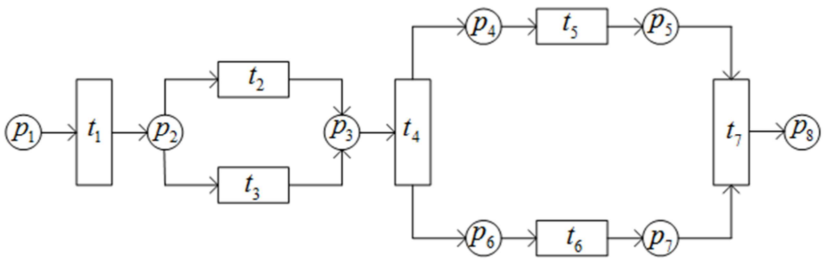

Figure 2. *The relations of transitions in SCN.*

In Figure 2, $t_1 \rightarrow t_j (j = 2,3,4,5,6,7)$, $t_i \rightarrow t_j (i = 2,3; j = 4,5,6,7)$, $t_4 \rightarrow t_j (j = 5,6,7)$, $t_i \rightarrow t_7 (j = 5,6)$, $t_2 + t_3$, $t_5 \parallel t_6$.

The execution of two transitions of a SCN either in strict order, exclusiveness, interleaving or in inverse strict order relations. These relations specify potential dependencies. The definition of the three relations are mutual exclusiveness. But in an ordinary net, there exists some pairs of transitions may belong to all these relations. (i.e. Figure 3) Sometimes, these relations in the ordinary net shows behavioral anomaly like dead-locks and unsoundness.

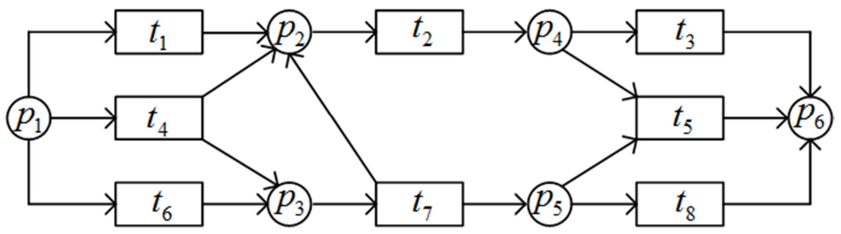

Figure 3. *The transitions with various kinds of relations.*

In Figure 3, we could see.

1) If the beginning firing sequence is $\{t_4\}$, then $t_2 \parallel t_7$.
2) If the beginning firing sequence is $\{t_1\}$, then $t_2 + t_7$.
3) If the beginning firing sequence is $\{t_6\}$, then $t_7 \rightarrow t_2$.

Therefore, the relations between t_2 and t_7 are differently and depend on the firing sequence.

Definition 4 (SCDN). Let $\mathrm{SCDN} = (\mathrm{P}, \mathrm{T}, \mathrm{F}, M_0, \mathrm{D}, R_T, W_T)$ be a sound free-choice Petri net with data operations if

1) $(\mathrm{P}, \mathrm{T}, \mathrm{F}, M_0)$ is a SCN.
2) D is the set of data items.
3) R_T denotes the "read" operation on T, W_T denotes the "write" operation on T.

Each transition should contain the "read" or "write" operations, if there are no element in "read" or "write" operation, we use $\emptyset$ to represent it. The "read" operations and the preset places are the preconditions of the transitions; correspondingly, the "write" operations and the post-set places are the post-conditions of the transitions. We use "R" denote the "read" operation and "W" denote the "write" operation. For example, in Figure 4:

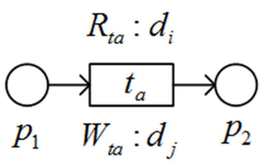

Figure 4. *The "read" "write" operations of* t_a.

As shown in Figure 4, d_i is the read operation of t_a, $d_i \in R_{t_a}$. d_j is the write operation of t_a, $d_j \in W_{t_a}$. Only occurring p_1 and $R_{t_a} = d_i$ could enable t_a and output p_2 with $W_{t_a} = d_j$.

Among the following definitions, in order to reduce the complexity of data flow analysis, we only consider those transitions that have the read or write operations in common, for others we ignore them.

Definition 5 (Relations of different data items). Let $\mathrm{SCDN} = (\mathrm{P}, \mathrm{T}, \mathrm{F}, M_0, \mathrm{D}, R_T, W_T)$ be a sound free-choice Petri net with data operations. $t_a \in T$, $d_i, d_j \in D$, $R_{t_a} \in R_T$, $W_{t_a} \in W_T$.

1) If $d_i \in R_{t_a}$ and $d_j \in R_{t_a}$, then $d_i \cap d_j \in R_{t_a}$.
2) If $d_i \in R_{t_a}$ or $d_j \in R_{t_a}$, then $d_i \cup d_j \in R_{t_a}$.
3) If $d_i \in W_{t_a}$ and $d_j \in W_{t_a}$, then $d_i \cap d_j \in W_{t_a}$.
4) If $d_i \in W_{t_a}$ or $d_j \in W_{t_a}$, then $d_i \cup d_j \in W_{t_a}$.

In this paper, we have ignored the delete operation and the guard function. Among the following definitions, in order to reduce the complexity of data flow analysis, we only consider those transitions that have the read or write operations in common, for others we ignore them.

Definition 6 (Redundant data flow errors, RDE). Let $\mathrm{SCDN} = (\mathrm{P}, \mathrm{T}, \mathrm{F}, M_0, \mathrm{D}, R_T, W_T)$ be a SCN with data operations, if a data element d_k satisfies with the following conditions in SCDN where there is a "write" operation but without a "read" operation later corresponding to it. That is to say, it may satisfy with one of the following conditions:

1) $\forall t_a, t_b \in T$, $t_a \rightarrow t_b$, $\exists d_k \in W_{t_a}$, s.t. $d_k \notin R_{t_b}$.
2) $\forall t_a, t_b, t_c \in T$, $(t_a + t_b) \cap (t_a \rightarrow t_c) \cap (t_b \rightarrow t_c)$, $d_k \in W_{t_a}$.
3) $\forall t_a, t_b, t_c \in T$, $(t_a \rightarrow t_b) \cap (t_a \rightarrow t_c) \cap (t_b + t_c)$, $d_k \in W_{t_a} \cap R_{t_b}$.

In Figure 5, the data element d_k is belong to RDE. Transition t_a creates d_k, but it never read or may read by transitions in the SCDN.

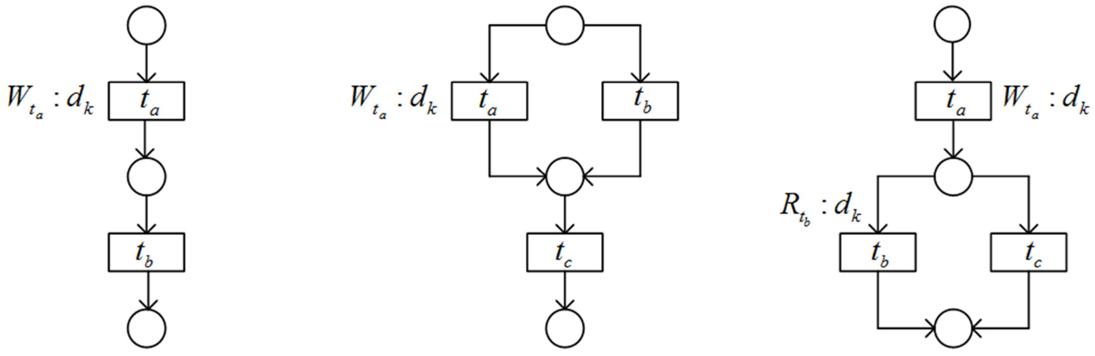

Figure 5. *Redundant data flow errors.*

Definition 7 (Missing data flow errors, MDE). Let $\mathrm{SCDN} = (\mathrm{P}, \mathrm{T}, \mathrm{F}, M_0, \mathrm{D}, R_T, W_T)$ be a SCN with data operations, if a data element d_k belongs to the "missing data flow errors" in SCDN where it may satisfy with one of the following conditions:

1) $\forall t_a, t_b \in T$, $t_a \rightarrow t_b$, $d_k \in R_{t_b}$.
2) $\forall t_a, t_b, t_c \in T$, $(t_a \rightarrow t_b) \cap (t_a \rightarrow t_c) \cap (t_b + t_c)$, $d_k \in R_{t_b} \cap R_{t_c}$.
3) $\forall t_a, t_b, t_c \in T$, $(t_a \rightarrow t_b) \cap (t_a \rightarrow t_c) \cap (t_b + t_c)$, $d_k \in R_{t_b}$.
4) $\forall t_a, t_b, t_c \in T$, $(t_a \rightarrow t_b) \cap (t_a \rightarrow t_c) \cap (t_b \parallel t_c)$, $(d_k \in R_{t_b}) \cap (d_k \in W_{t_c})$.
5) $\forall t_a, t_b, t_c \in T$, $(t_a + t_b) \cap (t_a \rightarrow t_c) \cap (t_b \rightarrow t_c)$, $(d_k \in W_{t_b}) \cap (d_k \in R_{t_c})$.

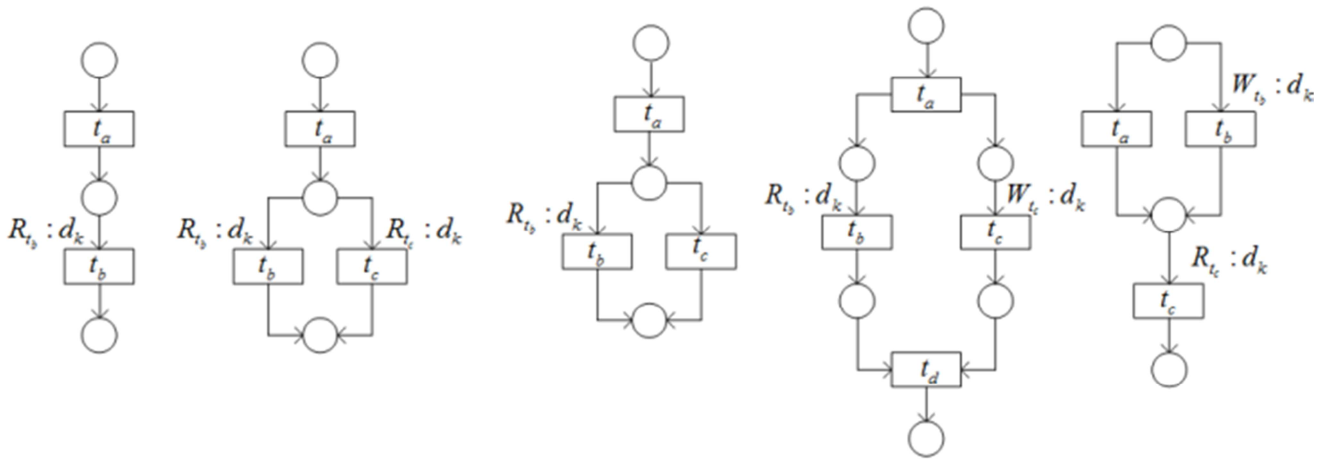

Figure 6. *Missing data flow errors.*

The data element d_k is in missing data flow errors. Note that d_k needs to be read immediately by t_b, but it has not been created yet on the left of Figure 5. On the right of Figure 6, data element d_k is created by t_c not by t_a, if the firing sequence is $\{t_a,t_b\}$, then $d_k \in MDE$. If d_k needs to be read immediately by t_b, but d_k has not been created if t_b firing in front of t_c result in $d_k \in MDE$. On the right of Figure 6, data element d_k is created by t_b not by t_a, but it should be read by t_c, if the firing sequence is $\{t_a,t_c\}$, then $d_k \in MDE$.

Definition 8 (False negative data flow errors, FNE). Let $\mathrm{SCDN}=(\mathrm{P},\mathrm{T},\mathrm{F},M_0,\mathrm{D},R_T,W_T)$, $SCDN'=(P',T',F',M_0',D',R_T',W_T')$, $\mathrm{SCDN}\subset SCDN'$.

1) $\exists \mathrm{t}\in \mathrm{T}\cap T'$, $\exists d_i \in R_{t\in T'}$, $d_i \notin W_{t\in T}$, then $d_i \in MDE$ in $SCDN'$.

2) $\exists d_j \in W_{t\in T'}$ and $d_j \notin R_{t\in T}$, then $d_j \in RDE$ in $SCDN'$.

If $\exists d_i \in MDE$ in $SCDN'$, then SCDN may not run to the end. If $\exists d_j \in RDE$ in $SCDN'$, then the $SCDN'$ could run to the end.

3. Data Flow Anomaly Detection Technology

Here we present our algorithms to analysis business process models. For ease of understanding, we illustrate the method in a usual way. The algorithms established upon detecting the data flow errors based on FNE.

Algorithm 1: Detect the data flow anomaly in SCDN based on FNE.

Input: $\mathrm{SCDN}=(\mathrm{P},\mathrm{T},\mathrm{F},M_0,\mathrm{D},R_T,W_T)$, $SCDN'=(P',T',F',M_0',D',R_T',W_T')$.

Output: The suspicious data flow elements in $SCDN'$,

```
1. If $\exists t\in T\cap T'$, $\exists d_i \in R_{t\in T'}$, $d_i \notin W_{t\in T}$ then,
2. $d_i \in MDE$ in $SCDN'$,
3. End if,
4. If $\exists d_j \in W_{t\in T'}$ and $d_j \notin R_{t\in T}$ then,
5. $d_j \in RDE$ in $SCDN'$,
6. End if,
7. Print the suspicious data flow elements $D = d_i \cup d_j$,
```

If the system could not detect MDE and RDE in $SCDN'$, the system may read the data from the transition outside as $SCDN'/SCDN$, result in changing the control flow structure which bring about control flow errors. In the following, we use algorithm 2 to analysis the control flow errors caused by FNE.

Algorithm 2: Detect the control flow errors based on BP relations.

Input: $\mathrm{SCDN}=(\mathrm{P},\mathrm{T},\mathrm{F},M_0,\mathrm{D},R_T,W_T)$, $SCDN'=(P',T',F',M_0',D',R_T',W_T')$, $\mathrm{BP}=\{\rightarrow,+,\parallel\}$, $\forall t_{a1},t_{b1},t_{c1},t_{d1},t_{e1},t_{f1}\in T_1$ correspond to $\forall t_{a2},t_{b2},t_{c2},t_{d2},t_{e2},t_{f2}\in T_1$

Output: The transitions with control flow errors T_1

```
1. If $t_{a1}\rightarrow t_{b1}$ in SCDN correspond to $t_{a2}\rightarrow t_{b2}$ in $SCDN'$ then,
2. $t_{a2}$ and $t_{b2}$ are normal,
3. Else,
4. $t_{a2}$ and $t_{b2}$ are abnormal,
5. End if,
6. If $t_{c1}+t_{d1}$ in SCDN correspond to $t_{c2}+t_{d2}$ in $SCDN'$ then,
7. $t_{c2}$ and $t_{d2}$ are normal,
8. Else,
9. $t_{c2}$ and $t_{d2}$ are abnormal,
10. End if,
11. If $t_{e1}+t_{f1}$ in SCDN correspond to $t_{e2}+t_{f2}$ in $SCDN'$ then,
12. $t_{e2}$ and $t_{f2}$ are normal,
13. Else,
14. $t_{e2}$ and $t_{f2}$ are abnormal,
15. End if,
16. Print the transitions with control flow errors $T_1 = t_{a2}\cup t_{b2}\cup t_{c2}\cup t_{d2}\cup t_{e2}\cup t_{f2}$.
```

The control flow errors would make the following transitions suffer from data flow errors. In the next section, we present some true examples to show the effectiveness of our methods.

4. Case Study

Table 1. *Control flow elements of Figure 7 and Figure 8.*

t_1	D waits for the supply of goods	t_{11}	M hasn't received the goods
t_2	The parcel waits for acceptation	t_{12}	M hasn't received the returned information
t_3	D accepts the goods	t_{13}	M sends a refund
t_4	D sends the delivery notice	t_{14}	D sends the delivery information
t_5	M receives the delivery notice	t_{15}	S receives the express arrival notice
t_6	S receives the delivery notice	t_{16}	S receives a refund
t_7	H receives the delivery notice	t_{17}	S receives the goods
t_8	H modifies the delivery notice	t_{18}	S closes the deal
t_9	M receives the notice of intercepting goods	t_{19}	M closes the deal
t_{10}	M receives the returned information	-	-
d_{11}	Cancel the order	d_{22}	M waits for the intercepting results

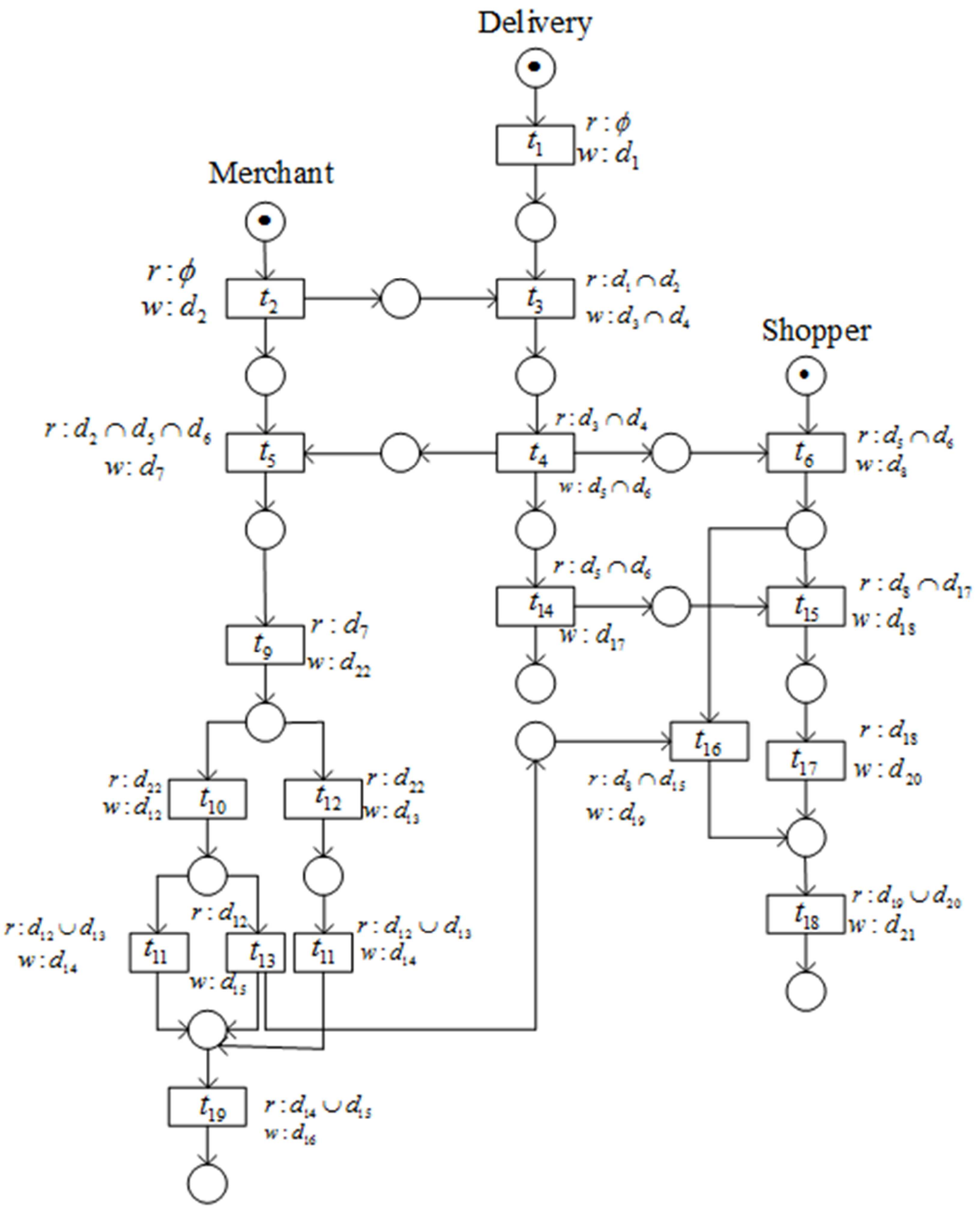
Delivery
Merchant
Shopper
$r:\phi$ $w:d_1$
$r:\phi$ $w:d_2$
$r:d_1\cap d_2$ $w:d_3\cap d_4$
$r:d_2\cap d_5\cap d_6$ $w:d_7$
$r:d_3\cap d_4$ $w:d_5\cap d_6$
$r:d_5\cap d_6$ $w:d_8$
$r:d_5\cap d_6$ $w:d_{17}$
$r:d_8\cap d_{17}$ $w:d_{18}$
$r:d_7$ $w:d_{22}$
$r:d_{18}$ $w:d_{20}$
$r:d_8\cap d_{15}$ $w:d_{19}$
$r:d_{22}$ $w:d_{12}$
$r:d_{22}$ $w:d_{13}$
$r:d_{19}\cup d_{20}$ $w:d_{21}$
$r:d_{12}\cup d_{13}$ $w:d_{14}$
$r:d_{12}$ $w:d_{15}$
$r:d_{12}\cup d_{13}$ $w:d_{14}$
$r:d_{14}\cup d_{15}$ $w:d_{16}$
t_1 t_2 t_3 t_4 t_5 t_6 t_9 t_{10} t_{11} t_{12} t_{13} t_{14} t_{15} t_{16} t_{17} t_{18} t_{19}

Figure 7. *The business process of a normal electronic trading.*

Table 2. *Data flow elements of Figure 7 and Figure 8.*

d_1	Bill of materials	d_{12}	Intercept the order succeed
d_2	Warehouse packing goods	d_{13}	Normal transport of goods
d_3	The type of goods	d_{14}	M waits for accepting goods
d_4	The quality of goods	d_{15}	M reduces account balance
d_5	Complete order sorting	d_{16}	M completes cancellation schedule
d_6	Packing success	d_{17}	The express company distributes D
d_7	M waits for goods being accepted	d_{18}	S searches express post
d_8	S waits for accepting goods	d_{19}	S increases account balance
d_9	S address	d_{20}	S completes the order
d_{10}	S ID	d_{21}	S completes the transaction

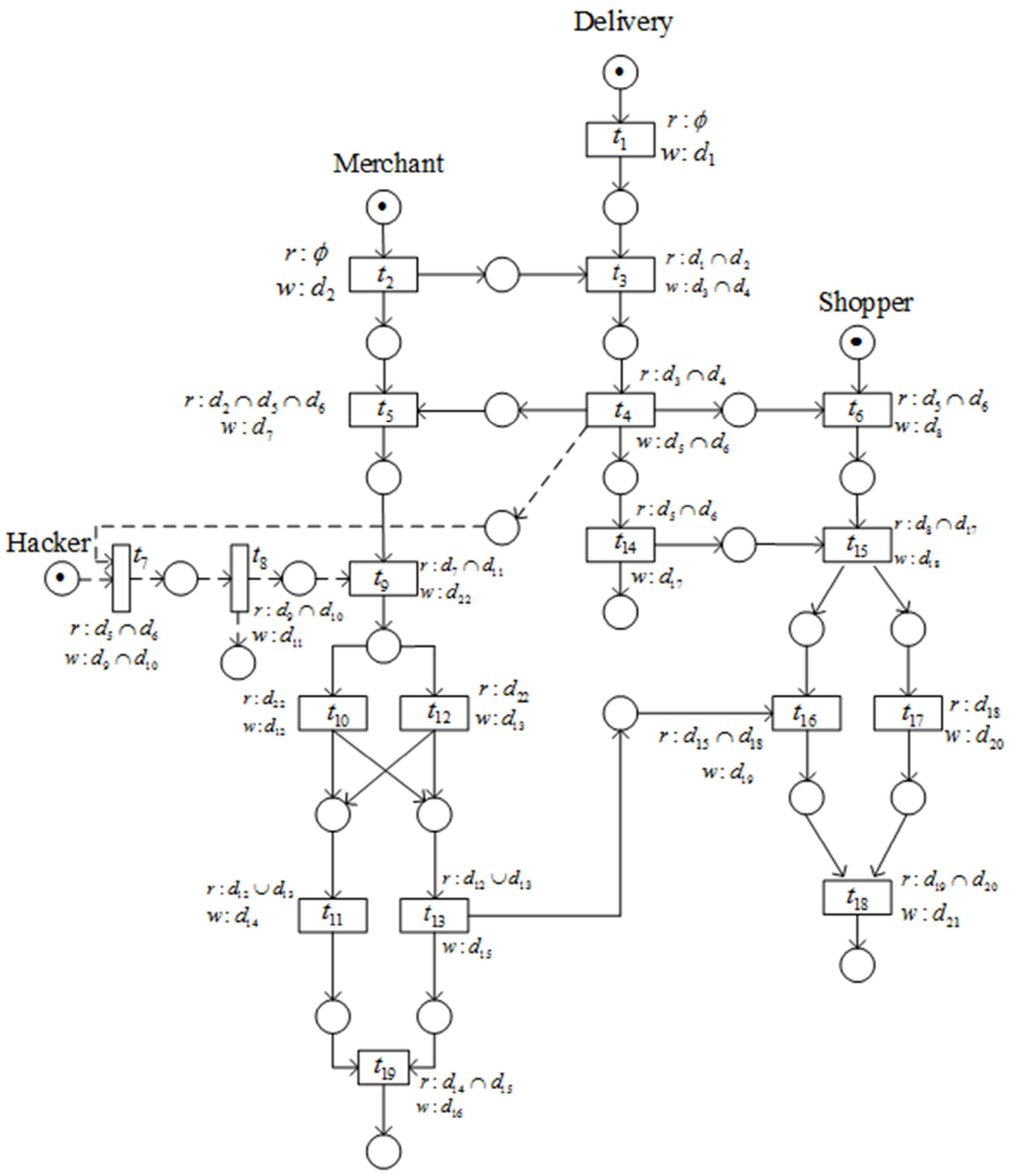
Delivery
Merchant
Shopper
Hacker
$r:\phi$ $w:d_1$
$r:\phi$ $w:d_2$
$r:d_1\cap d_2$ $w:d_3\cap d_4$
$r:d_3\cap d_4$ $w:d_5\cap d_6$
$r:d_2\cap d_5\cap d_6$ $w:d_7$
$r:d_5\cap d_6$ $w:d_8$
$r:d_5\cap d_6$ $w:d_{17}$
$r:d_8\cap d_{17}$ $w:d_{18}$
$r:d_5\cap d_6$ $w:d_9\cap d_{10}$
$r:d_9\cap d_{10}$ $w:d_{11}$
$r:d_7\cap d_{11}$ $w:d_{22}$
$r:d_{22}$ $w:d_{12}$
$r:d_{22}$ $w:d_{13}$
$r:d_{12}\cup d_{13}$ $w:d_{14}$
$r:d_{12}\cup d_{13}$ $w:d_{15}$
$r:d_{14}\cap d_{15}$ $w:d_{16}$
$r:d_{15}\cap d_{18}$ $w:d_{19}$
$r:d_{18}$ $w:d_{20}$
$r:d_{19}\cap d_{20}$ $w:d_{21}$
t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11} t_{12} t_{13} t_{14} t_{15} t_{16} t_{17} t_{18} t_{19}

Figure 8. *The business process of an abnormal electronic trading.*

In this section, we analysis the control flow errors and data flow errors based on the above algorithms. In the following tables, we use the alphabet M denotes "Merchant", S denotes "Shopper", D denotes "Delivery" and H denotes "Hacker".

In Figure 7, the firing sequence of have not read data item d_{11}. Cancel the order produced by Hacker, so the system does not suffer hacker's attacks which make the structure of transitions $t_{10}, t_{11}, t_{12}, t_{13}$ are as normal.

However, in Figure 8, the firing sequence of $t_{10}, t_{11}, t_{12}, t_{13}$ are affected by reading data item d_{11}. Cancel the order produced by Hacker not by the Delivery. For SCDN, the data item d_{11} belongs to MDE. For $SCDN'$, the data item d_{11} belongs to RDE for the system. If the system read the data item d_{11} wrote by Hacker, it would change the control flow structure and make changes in the data flow; multiple iterations of control flow and data flow error checking must be needed. Often the developed method could not defect this kind of error which bring about FNE.

According to algorithm 1 and algorithm 2, in $SCDN'$, data element $d_{11} \in RDE$ in t_8, $d_{11} \in MDE$ in t_9. The anomaly of control flow and data flow in source/target model caused by FNE are shown in Table 3.

Through Table 3, we know there are four different pairs of transitions' relations between the source model and the target model. At the time, there are five different read operations between them. The read operation's differences are just caused by the execution order change of control flow, which are all caused by FNE. We could draw the conclusion that the control flow and data flow could affect each other.

Table 4 shows the proposed algorithm compared with previous algorithms. It also shows that our algorithm could detect the drawbacks and logic errors which could not detect by others. The most important thing is that using our methods could detect these errors before the implementation of an e-commerce system which could avoid greater losses.

Table 3. *The anomalies of FNE.*

SCDN	**SCDN′**
$t_{11} + t_{13}$	$t_{11} \parallel t_{13}$
$t_{12} + t_{13}$	$t_{12} \rightarrow t_{13}$
$t_{15} + t_{16}$	$t_{15} \rightarrow t_{16}$
$t_{16} + t_{17}$	$t_{16} \parallel t_{17}$
$R_{t_9}: d_7$	$R_{t_9}: d_7 \cap d_{11}$
$R_{t_{13}}: d_{12}$	$R_{t_{13}}: d_{12} \cup d_{13}$
$R_{t_{16}}: d_8 \cap d_{15}$	$R_{t_{16}}: d_{18} \cap d_{15}$
$R_{t_{18}}: d_{19} \cup d_{20}$	$R_{t_{18}}: d_{19} \cap d_{20}$
$R_{t_{19}}: d_{14} \cup d_{15}$	$R_{t_{19}}: d_{14} \cap d_{15}$

Table 4. *Comparison of different algorithms.*

-	$\mathbf{d_{11}}$ **in** $\mathbf{W_{t_8}}$	$\mathbf{d_{11}}$ **in** $\mathbf{R_{t_9}}$
Reference [13, 25]	No problem	No problem
Reference [24]	No problem	No problem
Reference [26, 27]	No problem	No problem
Reference [28]	No problem	No problem
Our algorithm	RDE	MDE

5. Conclusions and Future Works

The formal techniques of fraud detection proposed in this paper based on the correctness analysis of the control flow and data flow. The advantages of the methods provided in this paper not only increase the detection rate but also could shorten the testing time and reduce the cost of attacks. The contribution of this paper is mainly manifested in the following aspects: Firstly, by analyzing the FNE caused by Hacker, we give a method to the system which could identify the original data flow errors. Secondly, based on the theory of BP, we could analysis the control flow errors and the data flow anomalies in the system. Through multiple iterations of control flow and data flow error checking, we could amend the model to avoid greater losses.

In the future, it is important for us to provide more methods on the detection of false negative data errors. In order to get rid of control flow anomaly and data flow anomaly, a series of iterations of the anomalies are needed in order to build an excellent model. At last, we should extend the method of model checking in irrationally free-choice Petri nets.

References

[1] Ying Huang, Wei Li, Zhengping Liang, Yu Xue, Xiuni Wang. Efficient business process consolidation: combining topic features with structure matching [J]. Soft Computing, 2018, 22 (2): 645-657.

[2] Natalia Sidorova, Christian Stahl and Nikola Trcka. Soundness verification for conceptual workflow nets with data: Early detection of errors with the most precision possible [J]. Information Systems, 2011, 36: 1026-1043.

[3] Eike Best and Harro Wimmel. Structure Theory of Petri Nets [J]. Transactions on Petri Nets and Other Models of Concurrency VII, 2013, 7480: 162-224.

[4] XuYa Cong, YuFeng Chen, ZhiWu Li, NaiQi Wu, Emad Abouel Nasr, and Abdulaziz Mohammed El-tamimi. Optimal Petri Net Supervisors of Discrete Event Systems via Weighted and Data Inhibitor Arcs [J]. IEEE Access, 2017, 6: 8245-8257.

[5] Matthias Weidlich, Artem Polyvyany, Jan Mendling and Mathias Weske. Causal Behavioral Profiles-Efficient Computation, Applications, and Evaluation [J]. Fundamental Informaticae, 2011, 113 (3-4): 399-435.

[6] Julio Clempner. Verifying soundness of business processes: A decision process Petri Nets approach [J]. Expert Systems with Applications, 2014, 41: 5030-5040.

[7] Matthias Weidlich, Tomer Sagi, Henrik Leopold, Avigdor Gal, and Jan Mendling. Predicting the Quality of Process Model Matching [C]. BPM, 2013, LNCS 8094: 203-210.

[8] Andreas Meyer, Luise Pufahl, Dirk Fahland, and Mathias Weske. Modeling and Enacting Complex Data Dependencies in Business Processes [C]. BPM, 2013, LNCS 8094: 171-186.

[9] Abdulelah Aldahami, Yuefeng Li, and Taizan Chan. Discovery of Dependency Relations in Sequential Data Flow [J]. Web Intelligence, 2017, 15 (1): 35-53.

[10] OMG. Business Process Modeling Notation Specification BPMN 1. 0, 2006.

[11] Nour Assy, Nguyen Ngoc Chan, and Walid Gaaloul. An Automated Approach for Assisting the Design of Configurable Process Models [J]. IEEE transactions on services computing, 2015, 8 (6): 874-888.

[12] Xinwei Zhu, Guobin Zhu, Seppe vanden Broucke, and Jan Recker. On Merging Business Process Management and Geographic Information Systems: Modeling and Execution of Ecological Concerns in Processes [C]. GRMSE, 2014, CCIS 482: 486-496.

[13] Sherry X. Sun, J. Leon Zhao, Jay F. Nunamaker and Olivia R. Liu Sheng. Formulating the Data-Flow Perspective for Business Process Management [J]. Information Systems Research, 2006, 17 (4): 374-391.

[14] Cristina Claudia Dolean, Razvan Petrusel. Data Flow Modeling: A Survey of Issues and Approaches [J]. Informatica Economica, 2012, 16 (4): 117-130.

[15] Booch G, Rumbaugh J, Jacobson I. The UML User Guide Addison Wesley, 1999.

[16] Wil M. P. van der Aalst and Hee K van. Workflow Management: Models, Methods and Systems, The MIT Press, 2002.

[17] Wil M. P. van der Aalst. Markings in Perpetual Free-Choice Nets are Fully Characterized by their Enabled Transitions [J]. Computer Science, Logic in Computer Science, 2018: 1-21.

[18] Matthias Weidlich, Jan Mendling and Mathias Weske. Efficient consistency Measurement Based on Behavioral Profiles of Process Models [J]. In 2011 IEEE Transactions on Software Engineer, 2011, 37 (3): 410-429.

[19] R. J. van Glabbeek and U. Goltz. Refinement of actions and equivalence notions for concurrent systems [J]. Acta Informatica, 2001, 37 (4/5): 229-327.

[20] J. Hidders, M. Dumas, W. M. P. van der Aalst, A. H. M. ter Hofstede, and J. Verelst. When are two Workflows the Same? [J] Australian Computer Society, 2005, 41: 3-11.

[21] Hema S. Meda, Anup Kumar Sen and Amitava Bagchi. On Detecting Data Flow Errors in Workflows [J]. ACM Journal of data and Information Quality, 2010, 2 (1): 1-31.

[22] Wang J and Kumar A. A framework for document-driven workflow systems [C]. BPM, 2005, LNCS 3649: 285-301.

[23] Basu A and Blanning R W. A formal approach to workflow analysis [J]. ISR, 2000, 11 (1): 17-36.

[24] M Hema Sundari, Anup K Sen, Amitava Bagchi. Detecting Data Flow Errors in Workflows: A Systematic Graph Traversal Approach [J]. In: Workshop on Information Technology and Systems, 2007.

[25] Sherry X. Sun and J. Leon Zhao. Developing a Workflow Design Framework Based on Dataflow Analysis [C]. Proceedings of the 41st Hawaii International Conference on System Sciences, 2008: 1-10.

[26] Nikola Trcka, Wil M. P. van der Aalst, and Natalia Sidorova. Analyzing Control Flow and Data Flow in Workflow Processes in a Unified Way [J]. Computer Science Reports, 2008, 0831: 1-23.

[27] Nikola Trcka, Wil M. P. van der Aalst, and Natalia Sidorova. Data-Flow Anti-patterns: Discovering Data-Flow Errors in Workflows [C]. In: International Conference on Advanced Information Systems Engineering, 2009, LNCS 5565: 425-439.

[28] Divya Sharma, Srujana Pinjala and Anup K Sen. Correction of Data-flow Errors in Workflows [C]. 25th Australasian Conference on Information Systems, Auckland, New Zealand, 8th -10th Dec 2014: 1-10.

[29] Silvia von Stackelberg, Susanne Putze, Jutta Mvlle, Klemens Bohm. Detecting Data Flow Errors in BPMN 2. 0 [J]. Open Journal of Information Systems, 2014, 1 (2): 1-19.

[30] Shazia Sadiq, Maria Orlowska, Wasim Sadiq and Cameron Foulger. Data Flow and Validation in Workflow Modeling [C]. Conferences in Research and Practice in Information Technology, 2003, 27: 207-214.

[31] Clarke, E. M., Grumberg, O., Peled, D. A. Model Checking [J]. The MIT Press, Cambridge, 1999.

[32] Win M. P. van der Aalst. Verification of workflow nets [J]. Lecture Notes in Computer Science, 1997, 1248: 407-426.

[33] Mariusz Dramski. Missing Data Problem in the Event Logs of Transport Processes [C]. TST 2017, 2017, CCIS 715: 110-120.