Detection of Dataflow Modeling Anomalies in Workflow system : Toward a Proactive Verification

Najat chadli, Mohamed Issam Kabbaj, Zohra Bakkoury

Abstract*: Business process modeling is a standard activity in the new technology of many organizations, because in order to model the process, the results are reflected in the execution, so any information system has to exchange data between activities by passing inputs and outputs. Workflow-net with data (WFD-net) has been used to verify data anomalies in a process such as missing, lost and redundant data and so on .Focusing on the approaches controls to read, write and destroy the operations of data and verifying the data at modeling. This verification begins until the system has completed the modeling . The correction starts after the end of a passive check. In this sense, the system using these approaches follow a certain malfunction, produce a significant flow over time during the modeling. In addition, after each end of correction, the modeler has to come back and start making further corrections for each detection of other errors in the process, which causes an infinite loop problem. The objective of this article is to reduce this infinite loop issue by identifying data flow anomalies and then correcting the failure of each data state in an activity, thus the proactive method that makes the modeler react in time in an ad hoc network that facilitates the application of proactive to intervene in time of data flow modeling anomalies in the workflow-net. In addition, the verification is done by applying the combination of active help checks and CTL* temporal logic in a model check with a data operation guard for each process fragment. In this verification in a workflow-net with o data operation, the Data-State concept is used to record the state of each last data operation with its activity as read ,write and destroy.

***Keywords* : Business Process Modeling, DataFlow, Anomalies, Verification, Validation.**

I. INTRODUCTION

The business process management provides an overview of the organization business process. To achieve a specific business goal, tasks need to be performed in the process by well-established means of a business process model [1].

Thus, it is needed to analyze the functionality of each activity of the company. Over the past fifteen years, many techniques have been developed to analyze workflow models [2]. In business process management, to model business process, workflow concept has become the standard reference of modeling [3].

In a previous, methodologies and tools in business process modeling were used to detect control flow errors (finding deadlocks and livelocks) [4]. Also, certain anomalies in data flow modeling such as missing data, contradictory data and redundant data [5,6,7]. However, these approaches raise many challenges during modelling and verification: 1) High complexity of the algorithm, 2) The verification time is extended until the system completes the processing, resulting in a passive method. 3) Not all anomalies are detected. 4) Some methods validate certain types of modeling cases but are unable to use loop modeling in conjunction with an xor-split.5) When checking, the system could not stop this infinite verification that is causes the infinite loop. The significance of data flow modelling in overall workflows specification and verification has been found to be justified [20]. Thus, data flow modeling with specification and validation is an important issue the criteria of workflow as in [23].

In the stage of data exchange in a system, every activity has an input data and an output data. For each state of data, the operation can be read, write or destroy in the activity [10]. Also, for each task are considered as state that is also used as matrix[5]. In addition, we also divided the model into five steps, considering each of these phases as a routing state by a model distinct from the other. The intersecting points are a logical connector or a place. Then, each state also has four connectors, for each state there are modeling and operating tasks. Eventually, each of the states is considered as a logical matrix. The data is checked at each fragment of the process, hence the verification of the data.

In this article, it is proposed to detect data flow modeling anomalies in business process modeling. Indeed, the fusion of two methodologies complements each other in the framework; the first has an active-help for control uses an ad hoc mesh network methodology when the second verifies the state for each fragment of a process where the activity must be considered as an element in a logical matrix . Data flows are protected by a Boolean predicate, (true or false), to workflow estimates with data operation, this protection is applied by mathematical formula.

Manuscript published on January 30, 2020.
* Correspondence Author

Najat chadli*, Mohammadia Engineering School, AMIPS Research TeamMohammed V University, Rabat,Morocco

Mohamed Issam Kabbaj, Mohammadia Engineering School, AMIPS Research TeamMohammed V University, Rabat,Morocco

Zohra Bakkoury, Mohammadia Engineering School, AMIPS Research TeamMohammed V University, Rabat,Morocco

We implemented the approach in a linear model using a split-xor connector with loop modeling.The model is validated in a network of workflows with data operations that required carefulness in each activity with logical predicates (true, false). Finally, in this case, the anomaly detection also suggested at this stage that the system should be locked until the anomalies have been corrected.

While this correction is being made, the processing state is monitored in the meantime to ensure that the new processing state of the data operations and its activities is recorded in a data state matrix concept that is available between times.The remainder of the paper is structured as follows. Section 2 reviews related work. Section 3 presents the proposed approach and the verification tools in the workflow system using a workflow-net with the data operation. Section 4 presents an The implementation of a motivated example .In section 5,presnts the interpretation of verification approach and rules validation. and in the last section we conclude our work and paves the way for future work.

II. RELATED WORK

In recent years, several papers have investigated business process modeling and detection of data flow modeling anomalies[9]. Many approaches are used to resolve this issue. Most of current approaches' use a passive help to detect the data flow anomalies in business process modeling or in workflow modeling. Therefore, the detection of anomalies in data flow modelling in business process modelling has spawned formal methodologies for formulating the modelling and verification of these data flows after the modeling. As a result, net workflows are extended types of Petri-net which uses data element with a data input and output [5]. Indeed, to formulate data perspective in business process management, data flow matrix which is an extension of a UML concept is used; this method has a high complexity algorithm to resolve the problems [6]. Moreover, for each workflow instance of a given workflow, there is an algorithm GtforDF for data flow verification to detect anomalies, such as lost data, missing data and redundant data [10]. The GtforDF makes use of the concept of corresponding pairs as in [12]. To execute tasks, the cases are manipulated in a specific order of the workflow management system is to support the definition, execution, recording and control of the process. Since processes are a major component part of workflow management, there is a requirement to revitalize an already well-established modeling and analytical framework for workflow processes [14]. Thus, for modeling and analyzing workflow process, it's required to a have proven framework based on Petri-Nets that are reliable process modeling techniques [15]. Some authors use workflow-net with data constraint to resolve the problem of the anomalies in data, but they don't specify whether the problems have corrected it or not [16]. Moreover, basic Petri-nets models have been used and proposed in many process representations by many researchers, and some use the wokflan tool (Workflow Analyze) to analyses the workflow process but without verifications [17]. Some of these approaches are based on workflow-net with data (WFD-Net) and apply in designing processing a model in parallel branches as a xor-parallel, xor-joint or xor-split connector, for example [7]. The various formalisms of data flow can be used in process modeling [18]. The natural candidate for the modeling and analysis of workflow has been formalized by the Petri net in Flow-oriented nature of workflows [19]. Few methods and tools exist to capture errors in the control-flow (finding, e.g., deadlocks and livelocks) [4], also data flow anomalies [9] (e.g., reading from an uninitialized element type of errors), and resource productivity type of problems in the information. Process modeling entangles methodologies for the conception of business process models. Some paper is intended principally to further extend existing efforts and provide a more general context for documenting and describing to understand the situations considered in them [21].Thus, before the business process is implemented as workflows, it must be correctly modeled as in [22]. The Petri Nets have been used with Data Operations (PN-DO), which extend contextual nets with write arcs and some other components to model the data operations of concurrent read and coverable write. Errors in data flow are concerned with inconsistent data and missing data. Thus, to check these errors an extended reachability graph (ERG) is constructed for a PN-DO, and a new detection method is put forward based on ERG [24]. Indeed, there is a method that can model both concurrent read and coverable write by a kind of Petri nets called Petri nets with data operation (PN-DO). A reachability graph with data operation for each PN-DO is reconstructed, and a detection algorithm which detects data-flow errors that can be identified rapidly is presented [25]. Using a temporal logic CTL* with a subset LTL. Thus, to find the data flow anomalies in a workflow system, that is choosing a workflow net with data and to verify the model checking using a Kripke structure, also, using the well-known stable, adaptable, and effective model-checking approach [19]. The objective of this documentation is to establish a formal unifying framework for finding data flow modeling errors in business process modeling and to develop a highly adaptive approach.work

III. PROBLEMATIC AND APPROACH OVERVIEW

A. Problematic and Approach Overview

Most of the company's business people and organizations have used the tax dematerialization system to produce and pay their taxes at any time. In a professional context, invoicing is the creation of a commercial and financial document on which a detailed and precise note of the products to be sold or the services rendered is issued. The adaptation of invoicing software in small and large companies saves time in processing their invoices.Thus, to minimize errors and automate processes and improve customer service. On the other hand, there are some problems when the process starts at the step to finish the invoicing and have a download of the receipt from the customer, so when paying these taxes via system, some problems are integrated into the modeling process performing verification and correction, this is caused by endless loop errors that prevent to finish the rest of the procedure and pass to the execution. Consequently, when a citizen begins to record this data in the system, there are some data that aren't learned in writing from the beginning of the operation.

These problems caused by a letter in Arabic that the system does not know learn it, but the data is read, it is a first case. In the second case, the system has written the data into the task, but when pursuing the payment processing, the system couldn't been read the data. Indeed, which may mean that there is a power interruption, or while the problem connected to the network the time be out is use up. Thus, this procedure should not continue until the end because the waiting time is over. In the third case, the data is written in several activities without being read, the problem frequently persists at the Asynchronous Transfer Mode (ATM) and we suggest that it be solved by our new approach.

B. Active help Method (Proactive)

Active help is a method that comes from several usable domains, in fact, this methodology facilitates verification in process models within the modeling process itself. It also provides a useful method to save time when checking and correcting according to an ad hoc approach, as described in this document. In this case, active help is a method applied in a business process model for each task or process activity. This method is suitable for use in a process instance [8]. .

C. Definition of Data Operation (DO)

The data operation is a data element state in each transition to the activity, this data operation can be read, written or destroyed. The status of this data operation is recorded in a status database concept. This recording of the last status of the data in the status database is normally done to protect this current status so as not to lose them as long as the system is locked to correct anomalies detected during the modelling process.

IV. THE APPROACH

As the system must be integrated data and other factors from outside, the flow from unprotected sources can also cause data anomalies during routing. Also, to manage them and prevent the system from falling into the infinite loop issue when the process is modeling. We propose an approach have been used two methods that are linked in each other. Consequently, an active help method in the ad hoc network and Guard with Boolean predicate (true, false) to control the state of data at every moment of a task in an activity. At the same time to ensure the integrity of data flows in the business process at each detection of anomalies when locking the system at the time of error correction in the task to be addressed, a database is used to save these data. As well as ,our objective is to analyze workflow-net system with data operations, which requires data operations to be stored in a Data-State with their last activity of each last operation, their benefit comes from the use Matrix. This approach is carried out in a CTL* temporal logic in a control model as in Figure 1.

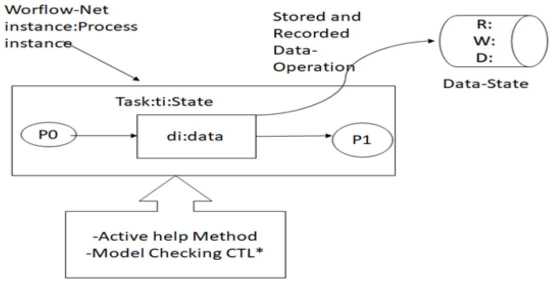


Fig. 1. Diagram of the proposed approach

A. The Workflow-Net with Data-Operation WFDO-Net

Our approach aims to verify the linear model with loop modeling using Xor-Split. Nevertheless, in some cases, active help cannot detect anomalies in the loop modeling[11]. For this reason, we provide a predicate check for each data operation state using data guard (i.e. blocking)[2]. Workflow system uses data in the process model, but this data must be routed through between transitions. For this reason, the workflow needs a network for data routing. That is why we suggest using Petri-net's wide-area workflow network[30] and having an enhanced read-write and destroy data operations. System uses Guard with Boolean predicate (true, false) for data in any activity of the process during the modeling process. Furthermore, both methods apply verification and control in parallel since the system locks once in each process fragmentation during modeling in case the error is detected. Meanwhile, data Operations and their latest state are record in Data-State with their activity.

B. Definition of Workflow-net with data operation (WFDO-Net).

1) A tuple $< P, T, F, DO, GD, DSt, f \ >$

2) $< T, P, F >$ is a WFDO-Net, with places P. transitions T and arcs F;

3) DO is a set of data operations;

4) DSt a Data-State of data with operations W, R, D.

5) GD is a set of guards over DO;

6) $f : D \rightarrow T \rightarrow \{R_{ti1}, W_{ti2}, D_{ti3}\}$

7)

$f = f_2 \ o \ f_1$ when $f_1 : D \rightarrow T \ and \ f_2 : T \rightarrow \{R_{ti1}, W_{ti2}, D_{ti3}\}$

$\{R_{ti}\}$ is a Read state of data operation in Data-State DSt.

8) $\{W_{ti}\}$ is a Write state of data operation in Data-State DSt

9) $\{D_{ti}\}$ is a Destroy state of data operation in Data-State DSt.

However, active help method has been used in a linear and a parallel model with a xor-split as in [9], but in loop modeling in this method isn't enough for missing data anomaly as in [10,11].

The proposed approach this paper aims to verify all types of process model condition used.

V. THE IMPLEMENTATION OF A MOTIVATED EXAMPLE

To construct that model, there is a linear model in which four xor-splits are used. One xor-split has a Decision Node including input data as in [11], while others are Connectors (Yes=true or No=false) for choosing the following routing. Next, activities are used through a set data in model during the processing as shown in figure 2.

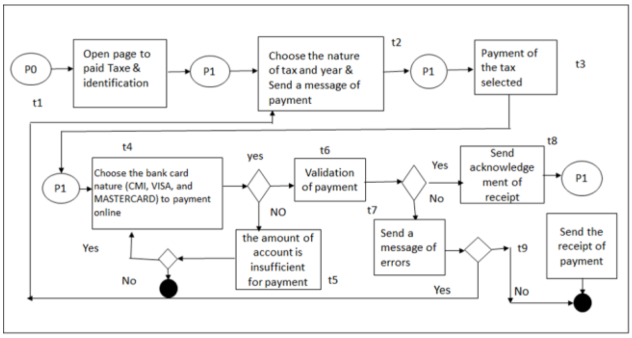

P0
Open page to paid Taxe & identification
t1
P1
Choose the nature of tax and year & Send a message of payment
t2
P1
Payment of the tax selected
t3
t4
P1
Choose the bank card nature (CMI, VISA, and MASTERCARD) to payment online
yes
NO
t6
Validation of payment
Yes
No
t8
Send acknowledge ment of receipt
P1
t7
Send a message of errors
Yes
No
the amount of account is insufficient for payment
t5
t9
Send the receipt of payment
Yes
No

Fig. 2.A linear model with three Xor-split.

The tasks ti in figure 2 are described in table 1 as activities.

Table- I: Description of Task Used.

Task	Description
t1	Open page to paid Taxe & identification
t2	Choose the nature of tax and year & Send a message of payment
t3	Payment of the tax selected (not clear)
t4	Choose the bank card nature (CMI, VISA, and MASTERCARD) to payment online
t5	Test whether the amount in the account is insufficient for the payment.
t6	Validation of payment
t7	Send an error message
t8	Send acknowledgement of receipt
t9	Send the payment receipt

Table I I: Description of data elements

Data	Description
d1	Identification
d2	Business name
d3	Business code
d4	Taxes-code (nature TVA, IS, VIGNETTE)
d5	The anterior years
d6	Nature and Account of Bank cards.
d7	security code of card.
d8	Record the sum of taxes selected
d9	Accept(yes/no)
d10	Bank card
d11	not valid (amount insufficient)
d12	Message error
d13	validation message
d14	Confirmation taxes receipt

Table III: Description of data elements

Operation	Description
R	Read
W	Write
D	Destroy

In the workflow system, various data flow anomalies may arise within processing, including missing data, conflicting data, and redundant data. Throughout , this paper proposes applying the approach to every workflow case. Furthermore, the guard manipulation in an activity controls a data operation which has read, written, and destroyed by using Boolean logic predicates either pred (True) or pred (False). Thereby approach consists on two methods, for any fragment from the model the first method applies in active help as in [9,10]. Nevertheless , this is insufficient if a loop modeling is being used. Accordingly , for every fragment in model we apply both verifications looking for anomalies. Such confirmation is confirmed by a CTL* temporal logic and a model check [13,14] as illustrated in Figure 3.

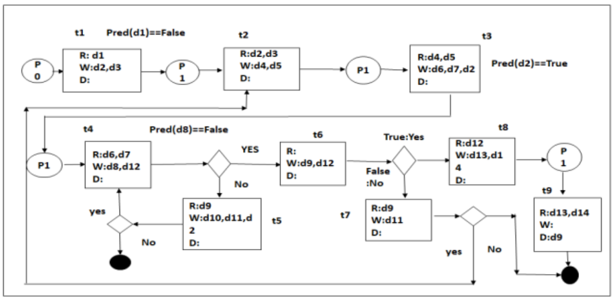


Fig. 3. WFDO-Net and the Boolean Predicate.

A.The Validation and verification of Data Flow Anomalies

The following anomalies are detected, among them Missing data ,Conflicting data and Redundant data. There are other anomalies, but in the present Paper we are dealing with these three anomalies. The validation and verification of data in relation to these anomalies are inferred by means of verification rules.

- **Missing Data Rule**

For certain data operations in this case example, if a data item is to be accessed, i.e. read or destroyed, but either it was never initialized, or it was deleted and not restarted, or it was destroyed without having been initialized again; this is called missing data. So if

f(d_ti) = {R_ti} ou f(d_ti) ==={D_ti} => Missing Data.

- **Conflicting data Rule**

The given data is created, and then rewritten again without being read before or in between, or even overwritten without being read first. These operations are in conflicting data situation: if f(d_ti)={W_ti } & f(d_ti)≠{R_ti} ,or if f(d_ti)={W_ti } & f(d_ti)={D_ti }=>f(d_ti)≠{R_ti}.

- **Redundant data Rule**

For each instance of the process, when data di is written to a task, and the data is never read at any time; St(ti)={Wti}, this implies that redundant data is implicated.

B. The Confirmation of Result of Rules

The active help in an ad-hoc method as in [9] is used as in figure 3.The gives results are :we showed the concept database " data-state" has recorded five phases of state 1, state 2, state 3, state 4, state 5. For each phase, the state is described the data operation in the current activity as shown in table 4 below.

Table IV: Description of data elements

State	Task
State 1	t1, t2, t3, t4, t5
State2	t5
State3	t6
State4	t7
State5	t8, t9

Below we show the analysis of data operation and the routing from a task (activity) to task (activity) in each record in data-state as shown in table 5, table 6 and table 7.

Table V: Description of data elements

Data	State 1			
	Task1	Task2	Task3	Task4
d1	(Rt1,0,0)			
d2	(0, Wt1,0)	(Rt2, Wt1,0)	(Rt2, Wt3,0)	
d3	(0, Wt1,0)	(Rt2, Wt1,0)		
d4		(0, Wt2,0)	(Rt3, Wt2,0)	
d5		(0, Wt2,0)	(Rt3, Wt2,0)	
d6			(0, Wt3,0)	(Rt4, Wt3,0)
d7			(0, Wt3,0)	(Rt4, Wt3,0)
d8				(0, Wt4,0)
d12				(0, Wt4,0)

Table VI: Verification Data Operation in the State 2, State3 and State 4.

Data	State 2	State3	State4
	Task5	Task6	Task7
d9	(Rt5, 0,0)	(0, Wt6,0)	(Rt7,0,0)
d10	(0, Wt5,0)		
d11	(0, Wt5,0)		(0, t7,0)
d12		(0, Wt6,0)	

Table VII: Verification data operation in the State 5.

Data	State 5	
	Task 8	Task9
d9		(0,0, Dt9)
d12	(Rt8, 0,0)	
d13	(0, Wt8,0)	(Rt9,Wt8,0)
D14	(0, Wt8,0)	(Rt9,Wt8,0)

Applying rule 1 of missing data, we notice that d1 has been detected in State 1 and d9 is detected in State 2, State 4, State 5 and d12 is detected in State 5. Similarly, when we applied the rule of conflicting data, we found that the data operation d10 is detected in state 2 and state 4 and also d11 is detected in State 2. Finally, when we apply the rule of redundant data, the data d8 is detected in State1.

During the design phase, it is better to check for the correctness of design in every state. When designing the system, the modeler locks the system to correct these anomalies, as in the table7. We choose to correct the data d1 by write and read in task1, the data d9 is written in the task5, and the data d8 is destroyed because it didn't read never for state. However, d10 and d11 are updated to be read in the same task. We always ensure that the model meets the specifications/requirements. Consequently, checking the correctness of specifications is a logical step. We may use some logical formalism to represent the specification and use the underlying theory of that coherent framework to reason it [13].

Table VIII: Verification and correction of anomalies.

Data	State 1	State2	State5
	Task1	Task4	Task5
d1	(Rt1, Wt1,0)		
d8		(0, Wt4, Dt4)	
d9			(Rt5, wt5,0)
d10			(Rt5, Wt5,0)
d11			(Rt5, Wt5,0)

VI. INTERPRETATION: THE VALIDATION AND VERIFICATION OF THE APPROACH

During the first stage of processing, input and output data of the activity are subject to such operations as reading, writing/updating, and destruction. The ability to assign data states in each instance of the process can be monitored and managed autonomously using the influence of the ad hoc method applied with active help. Consequently, the Workflow instance of workflow-Net providing a temporal modeling framework which is responsible for the interpreting a process instance and for controlling its transition from the base-state of a process instance initiated [29]. Hence, before running each process instance, it needs a modeling of the data flow. At this stage of the modeling, the system verifies process for detecting data anomalies. Accordingly, the activity can be considered as a state related with operation of the recorded data in Data state, that' s why the state of each activity can easily be analysed and without deviating from data analysis to checking through an active help technique. We have divided tasks (activities) in the matrix of Data Statement in five states. For each state, anomalies in data transactions are corrected. Nevertheless, there are certain anomalies which are not detected, as in the loop modeling as in [10].

The temporal logical CTL* is thereby a formal check, like the one used for the last check, i.e. every state instance is controlled by CTL*, based on the function of the Activity-State Data operation. The formal checking uses a logical computation tree-method that allows to combine the quantifiers and the temporal operators. Additionally, this may be a useful feature in situations where the Active Help Method could not find errors while modeling for the loop, as CTL* verification is an option to complete in process verification.

The modeler has the possibility to use methods for any process fragment that cannot apply any of them for correction. Both methods are complimentary to each other methods for resolving data anomalies issues. For instance, the Formal verification by CTL* in Workflow-net is quite good, though it is not possible to record state operations in the database called data-state. The Garde of data is a means of formal verification in the model is used as an evaluation of Pred (True) or Pred (False) predicates for each data operation.

We suggest applying the CTL* compute tree logic verification rules for each anomaly, missing data, conflicting data and redundant data, this rules verification in the system as shown in Figure 3 and Table 9.

A.The Formalization of the Model

- **Definition: Validity of CTL* formulas in [17].**

On the other hand, we define when a CTL*, ϕ status formula is valid in an s (notation: s⊨¬ϕ). Moreover, when a CTL* path formula ψ is valid on a path π (notation: π⊨ψ) by simultaneous induction as follows:

1) $s \models \top$
2) $s \not\models \bot$
3) $s \models p \quad iff\ p \in L(s)$
4) $s \models \neg\phi \quad iff\ s \not\models \phi$
5) $s \models \phi \wedge \psi \quad iff\ s \models \phi\ and\ s \models \psi$
6) $s \models \phi \vee \psi \quad iff\ s \models \phi\ or\ s \models \psi$
7)$s \models \neg\phi \to \psi\ iff\ s \models \phi\ implies\ s \models \psi$

The state s = t; L is a label as in Definition St, grt.

- **Symbols and Signification**

1) Each temporal connect is a pair of a path quantifier: A - for all paths
2) E - There exists a path.
3) An LTL-like temporal operator X, F, U. Precedence(high-to-low): $(AX;\ EF;\ \neg)\ ;\ (\wedge,\vee)$
4) $AX\ \phi$: For every next state, ϕ holds
5) $EF\ \phi$: There exists a path with a future state where ϕ holds.
6) $E[\phi\ U\ \psi]$: There exists a path where eventually holds, and ϕ holds at all earlier states

We apply the Formalization of the three Rules in the following.

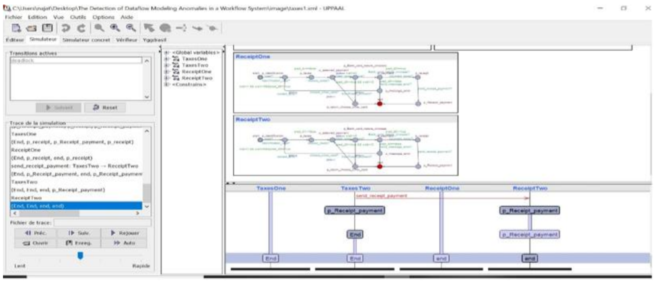
Table- IX: Rules Formalization.

Rules	Formalization Rules
Missing data Rule	$E[\neg St(Wti) \cup (St(Rti) \vee St(Dti))]$
Conflicting data Rule	$EF[St(Wti) \wedge AX[\neg(St(Rti) \vee St(Dti)) \cup (St(Wti) \wedge \neg St(Rti))]]$
Redundant data Rule	$EF[St(Wti) \wedge AX[\neg St(Rti) \cup (term \vee (St(Dti) \wedge St(Rti)))]]$

B. Experimentation Phase by Upaal tool.

The following describes detailed results from the experimentation with the Uppaal tool to validate and verify the proposed approach. The Uppaal tool is employed for both validation through graphical simulations. It is also used to verify through automatic model verification. The verification is a theoretical framework that is based on timed automata [26].

In Uppaal, system implementation of the approach is done through two models. First, Taxes() model with two instantiations "TaxesOne" and "TaxesTwo". The second template is the Receipt() Template that has both "ReceiptOne" and "ReceiptTwo" instantiations. These new templates instantiation of these new models has the same automaton structure as well as the local variables as being defined in them. The declarations in the template are global or local and may be clocks, bounded integers, channels. The templates are construct from a channel of localization's as well as transitions that are declared per channel (Chan) mentioned in:Open, choose_taxes,Identification_user,selected_taxes_nature,insufficient_account,payment_taxes,Send_message_error,choose_other_card,p_Identification,p_Bank_card_nature,p_receipt, p_payment,send_receipt_payment,Bank_card_nature_chooses,correct_error,p_message_error.In the case of an open.In this way, the transitions are linked together by their locations and their task-to-task synchronization, as well as by their number.In this manner Firstly, we built our Taxes model and declared them in java by another instantiation TaxesOne and TaxesTwo. In a dialing system, our model consists of two instances of the Receipt model called TaxesOne and TaxesTwo and the second Receipt model has the same two instances called ReceiptOne and ReceiptTwo. Consequently, our logic model has two instantaneous automata, operating in parallel. Thus, the global state of our model is entirely determined by the places with the states of each synchronization by the global transition as in figure 4.



C:\Users\najat\Desktop\The Detection of Dataflow Modeling Anomalies in a Workflow System\image\taxes1.xml - UPPAAL

Transitions actives

deadlock

Trace de la simulation

TaxesOne
(End, p_receipt, p_Receipt_payment, p_receipt)
ReceiptOne
(End, p_receipt, end, p_receipt)
send_receipt_payment: TaxesTwo → ReceiptTwo
(End, p_Receipt_payment, end, p_Receipt_paymen
TaxesTwo
(End, End, end, p_Receipt_payment)
ReceiptTwo
(End, End, end, end)

Fig. 4. Template instantiations.

The list of composed processes into a system TaxesOne =Taxes ();

Taxestwo=Taxes ();

ReceiptOne=Receipt ();

ReceiptTwo=Receipt ();

system TaxesOne, TxaesTow, ReceiptOne, ReceiptTwo;
After the simulation triggered the platform, the tool has two boxes. One for active transitions, which the deadlock rules employed.

For the extended model simulation, the other model is in deadlock until the first simulation is finished. In this way, in the bellow box "Trace simulation", the chosen instances of receipt and payment for simulations tried to execute the rules of transition through until the deadlock happens.

As we will see, when the template drew, we start the next step of simulation for validation. At the same time, we check if there are any mistakes of the declaration. If the system is correct and the simulation is valid, we pass to drawing the second template payment and simulate it as in figure 3. For each transition (arc), the synchronization between the location and used the guard for Boolean predicate of data operation when the system detects anomalies as in figure 4.

- **The clock in Uppaal.**

During the design and analysis of real operating structures, the time issues may also contribute. They may sometimes deviate from such quantitative timing and only consider the sequencing of the events, but frequently it is also necessary to incorporate timing information into the model in order to allow us to answer some of these questions. For This reason, for example, they may have to determine both that a certain location can be accessed and at what speed. In their workshop, the same question may be asked. The Uppaal was more or less designed to answer such questions. [28].

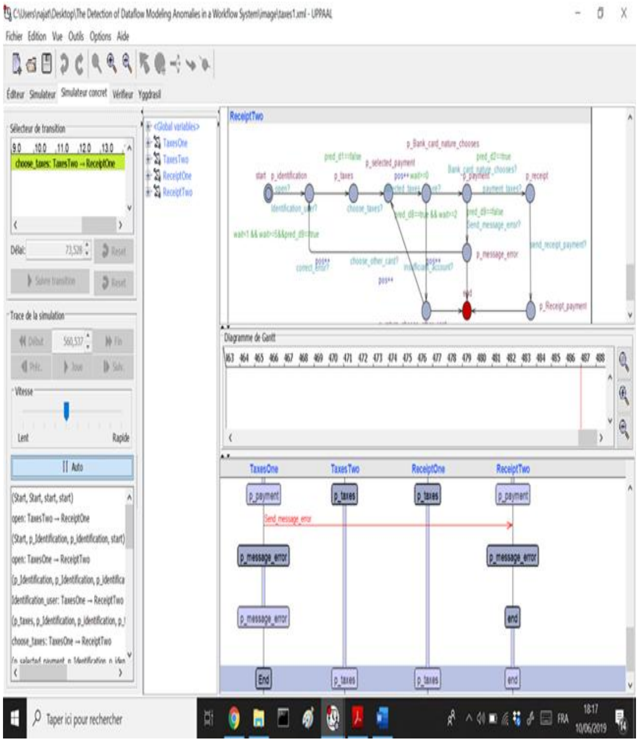
- **Validation by Simulation**

The simulation function supported by Uppaal tool is an interactive functionality for animating models [28]. Therefore, that simulation step can be placed in a workflow system that is designed upon running. Uppaal is a shake synchronization system: the two processes assume a simultaneously transition, one is a ! while the other is a ?, one of them being the synchronization channel. During a transition, two actions are possible: the allocation of variables or resetting time to zero [27].

a! = Emission

a? = Reception

Nevertheless, the simulation performance for time controlled automata changes whenever the variable declaration modification is made. Validation steps with Model in Uppaal conditions: After simulation triggered Platform, Uppaal tool contains two boxes. One for active transitions, which are used for blocking rules. For the extended model simulation, the other model is locked until the end of the first simulation. This way, in the box "Draw simulation", it is possible for the transition rules to be carried out until the deadlock occurs. At the top of this box, there are two: the "next" and "reset" buttons as shown in figure 5.



C:\Users\najat\Desktop\The Detection of Dataflow Modeling Anomalies in a Workflow System\image\taxes1.xml - UPPAAL

choose_taxes: TaxesTwo → ReceiptOne

Délai: 73,528

(Start, Start, start, start)
open: TaxesTwo → ReceiptOne
(Start, p_Identification, p_identification, start)
open: TaxesOne → ReceiptTwo
(p_Identification, p_Identification, p_Identifica
Identification_user: TaxesOne → ReceiptTwo
(p_taxes, p_Identification, p_identification, p_I
choose_taxes: TaxesOne → ReceiptTwo

Fig. 5. Screen dump of Concrete Simulator.

C.The Results of simulation

- **Rules implemented**

The Workflow-net with data operation with an active help is our approach that should be verified by Uppaal verifier. In the timed automata in Uppaal, this approach is able to verify it. This component is used by so-called Query property. This property may be allowed or not for a given model. For reachability states and for each global state in the two templates, the verifier is able to establish that the query is satisfied.Some symbols are used in Query as in logic temporal model CTL* of model checking, so it used the A [], E< >.

- A []: means “In all reachable states it is the case that”.
- E<>: taken from temporal logic, means “There exists a reachable state such that”.

We examine our implementation of deadlock:

E<> not deadlock

We start to verify our approach using the following rules:

Table- X: The Results of simulation

Missing data Rule	Conflicting data Rule	Redundant data Rule
$E <> ((ReceiptTwo.p\_identification == 1$ $and\ ReceiptTwo.p\_identification == true)$ $imply\ (pred_{d1} == false)$ $\&(ReceiptTwo.p\_Bank\_card\_nature\_chooses \leq 4$ $\&ReceiptTwo.p\_Bank\_card\_nature\_chooses$ $== true)imply(pred\_d9 == false))$	$E <> ((ReceiptTwo.p\_Bank\_card\_nature\_chooses$ $\leq 4 \& ReceiptTwo.p\_Bank\_card\_nature\_chooses =$ $= true \& (ReceiptTwo.p\_payment == 4$ $\& ReceiptTwo.p\_payment == false)$ $imply$ $(ReceiptTwo.p\_Bank\_card\_nature\_chooses == false)))$	$E <> ((ReceiptTwo.p\_Bank\_card\_nature\_chooses$ $\leq 4 \& ReceiptTwo.p\_Bank\_card\_nature\_chooses =$ $= true \& (ReceiptTwo.p\_payment == 4$ $\&ReceiptTwo.p\_payment == false)$ $imply$ $(ReceiptTwo.p\_Bank\_card\_nature\_chooses == false)))$

DOI:10.35940/ijrte.E6946.018520
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

D . The interpretation Results

The verification is approved by a check button in the right of platform of verifier. When the system verifies the query, it writes a message of satisfaction. Therefore, the verification used satisfies our approach in active help and WFDO-Net for checking our rules in the cases of missing data and conflicting data and redundant data. However, this checking didn't get to all data-state issues and the data operation is recorded when the system is locked and the operations of “ , is a set of {Rti,Wti,Dti} operation data and St is a labelling function” the operation data could not be handled. Consequently, for the future work, we attend to use a database synchronised with the tool to check the system in the business process modeling using data flow modeling for detection of anomalies.

VII. CONCLUSION

In this paper, we used two verifications in order to detect the data flow modeling anomalies in a certain precise and reduced time using the WFDO-Net and active help. The used example has been geared toward a dematerialization of taxation in the Ministry of Finance which validated our method. As the verification by a proactive couldn't detect some anomalies in different models with a loop. That's why ,it's proposed to add a guard for each data according the confirmation with active help, in an Ad Hoc mesh networking. Indeed , the (WFDO-net) with verification in a temporal logic CTL* and model checking and a guard for data in each activity when modeling. Furthermore, this approach is applied in an instance of process the workflow of WFDO-Net, in order to correct all anomalies to be detected . Consequently, we used a Tool for validation of the model checking to detect the data flow modeling anomalies, missing data, conflicting data and redundant data in the workflow-Net with the data operation. Consequently, for each instance in time the system is verified, and the timed automata is applied the CTL*. Nevertheless, this tool is not able to check all our issue that way. For future work, we intend to associate a database with a tool to verify each data operation.

REFERENCES

1. Küster, J. M., Ryndina, K., & Gall, H. (2007, September). Generation of business process models for object life cycle compliance. In International Conference on Business Process Management (pp. 165-181). Springer, Berlin, DOI https://doi.org/10.1007/978-3-540-75183-0_13,
2. Van Der Aalst, W., Van Hee, K. M., & van Hee, K. (2004). Workflow management: models, methods, and systems. MIT press, ISBN: 0-262-72046-9.
3. Trcka, N., van der Aalst, W., & Sidorova, N. (2008). Analyzing control-flow and data-flow in workflow processes in a unified way. Computer science report, (08-31).
4. W.M.P. van der Aalst, K.M. van Hee, A.H.M. ter Hofstede, N. Sidorova, H.M.W.Verbeek, M. Voorhoeve, and M.T. Wynn. Soundness of Workflow Nets: Classification, Decidability, and Analysis. BPM Center Report BPM-08-02, BPMcenter.org,2008, ISSN: 0934-5043.
5. Cong, L. I. U., ZENG, Q., & Hua, D. U. A. N. (2014). Formulating the data-flow modeling and verification for workflow: A petri net based approach. International Journal of Science and Engineering Applications, 3, 107-112.
6. Sun, S. X., Zhao, J. L., Nunamaker, J. F., & Sheng, O. R. L. (2006). Formulating the data-flow perspective for business process management. Information Systems Research, 17(4), 374-391.ISSN: 1047-7047.
7. Trčka, N., Van der Aalst, W. M., & Sidorova, N. (2009, June). Data-flow anti-patterns: Discovering data-flow errors in workflows. In International Conference on Advanced Information Systems Engineering (pp. 425-439). Springer, Berlin, Heidelberg.
8. Kapuruge, M., Han, J., & Colman, A. (2014). Service Orchestration as Organization: Building Multi-tenant Service Applications in the Cloud. Elsevier. ISBN: 0-12-801097-5.
9. Kabbaj, M. I., Bétari, A., Bakkoury, Z., & Rharbi, A. (2015). Towards an active help on detecting data flow errors in business process models. IJCSA, 12(1), 16-25.
10. Chadli, N., Kabbaj, M. I., & Bakkoury, Z. (2018, October). Detection of Dataflow Anomalies in Business Process An Overview of Modeling Approaches. In Proceedings of the 12th International Conference on Intelligent Systems: Theories and Applications (p. 37). ACM. DOI:10.1145/3289402.3289537.
11. Chadli, N., Kabbaj, M. I., & Bakkoury, Z. (2020). An Enhanced Adhoc Approach Based on Active Help to Detect Data Flow Anomalies in a Loop of a Business Modeling. In Emerging Technologies for Connected Internet of Vehicles and Intelligent Transportation System Networks (pp. 127-147). Springer, Cham.ISSN: 3030227731.
12. Meda, H. S., Sen, A. K., & Bagchi, A. (2007, October). Detecting data flow errors in workflows: A systematic graph traversal approach. In 17th Annual Workshop on Information Technolgies & Systems (WITS) Paper.
13. Beyer, D., Gulwani, S., & Schmidt, D. A. (2018). Combining model checking and data-flow analysis. In Handbook of Model Checking (pp. 493-540). Springer, Cham.
14. Beyer, D., Gulwani, S., & Schmidt, D. A. (2018). Combining model checking and data-flow analysis. In Handbook of Model Checking (pp. 493-540). Springer, Cham.
15. Van der Aalst, W. M. (1998). The application of Petri nets to workflow management. Journal of circuits, systems, and computers, 8(01), 21-66.
16. He, Y., Liu, G., Xiang, D., Sun, J., Yan, C., & Jiang, C. (2018). Verifying the correctness of workflow systems based on workflow net with data constraints. IEEE Access, 6, 11412-11423. ISSN:2169-3536
17. Verbeek, H. M., Basten, T., & van der Aalst, W. M. (2001). Diagnosing workflow processes using Woflan. The computer journal, 44(4), 246-279.
18. Cappelli, C., Santoro, F. M., Cesar Sampaio do Prado Leite, J., Batista, T., Luisa Medeiros, A., & Romeiro, C. S. (2010). Reflections on the modularity of business process models: The case for introducing the aspect-oriented paradigm. Business Process Management Journal, 16(4), 662-687.
19. Van Der Aalst, W. M., Van Hee, K. M., Ter Hofstede, A. H., Sidorova, N., Verbeek, H. M. W., Voorhoeve, M., & Wynn, M. T. (2011). Soundness of workflow nets: classification, decidability, and analysis. Formal Aspects of Computing, 23(3), 333-363.ISSN: 0934-5043
20. Sadiq, S., Orlowska, M., Sadiq, W., & Foulger, C.[] (2004, January). Data flow and validation in workflow modelling. In Proceedings of the 15th Australasian database conference-Volume 27 (pp. 207-214). Australian Computer Society, Inc.
21. Liu, R., & Kumar, A. (2005, September). An analysis and taxonomy of unstructured workflows. In International Conference on Business Process Management (pp. 268-284). Springer, Berlin, Heidelberg.
22. Xiang, D., Liu, G., Yan, C., & Jiang, C. (2016, December). Checking the inconsistent data in concurrent systems by petri nets with data operations. In 2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS) (pp. 501-508). IEEE.ISBN: 1-5090-4457-4
23. Xiang, D., Liu, G., Yan, C., & Jiang, C. (2017). Detecting data-flow errors based on Petri nets with data operations. IEEE/CAA Journal of Automatica Sinica, 5(1), 251-260.ISSN: 2329-9266.
24. Behrmann, G., David, A., & Larsen, K. G. (2006). A tutorial on Uppaal 4.0. Department of computer science, Aalborg university.
25. David, A., Amnel, T., Stigge, M., & Ekberg, P. (2011). UPPAAL 4.0: Small Tutorial.
26. Mendoza Morales, L. E. (2014). Business Process Verification: The Application of Model Checking and Timed Automata. CLEI Electronic Journal, 17(2), 3-3.ISSN: 0717-5000.
27. Vaandrager, F. (2011). A First Introduction to uppaal. Deliverable no.: D5. 12 Title of Deliverable: Industrial Handbook, 18.
28. Kapuruge, M., Han, J., & Colman, A. (2014). Service Orchestration as Organization: Building Multi-tenant Service Applications in the Cloud. Elsevier.ISBN: 0-12-801097-5.
29. Jalali, A., Ouyang, C., Wohed, P., & Johannesson, P. (2017). Supporting aspect orientation in business process management. Software & Systems Modeling, 16(3), 903-925.ISSN: 1619-1366.

30. Murata, T. (1989). Petri nets: Properties, analysis and applications. Proceedings of the IEEE, 77(4), 541-580.

AUTHORS PROFILE

Najat Chadli, is a PhD Student in Mohammadia Engineering School, AMIPS Research Team Mohammed V University, Rabat ,Morocco .She has trees international Conference ACM, Springer, SCITEPRESS , and a chapter Books in Emerging Technologies for Connected Internet of Vehicles and Intelligent Transportation System Networks pp 127-147.

Mohammad. Issam Kabbaj, Holds A Ph.D Degree In Computer Science With Distinction In 2009; Habilitated Professor At Computer Science Department Of Ecole Mohammadia D'ingénieurs (Emi) In Morocco; His Interests Include Software Process, Business Process, Risk Management, Process Mining, Dynamic Adaptation, Mda.

Zohra Bakkoury, Holds A Phd In Engineering From Exeter. University In Uk And A Degree Of Engineer In Computer Science From Ecole Mohammadia (Emi) In Rabat, Morocco. She Is Professor At The Computer Science Department Of (Emi). Founder Of Amips Research Group. Founding Member And Head Of Masi Research Laboratory. Her Current Research Interests Are Focused On Business Process Management, Process Mining, Data Mining And Intelligent Systems. She Has Published A Book Chapter Beside Numerous Articles. Zohra Was The Co-Chair Of Sita Conference On Intelligent Systems In 2013 And 2018.