Implementation of the Semantic Web in Business Process Modeling Using Petri Nets

Yutika Amelia Effendi, Riyanarto Sarno
Department of Informatics
Faculty of Information and Communication Technology, Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
e-mail: yutika.effendi@gmail.com, riyanarto@if.its.ac.id

***Abstract*—Nowadays, Petri net models have been used as a very promising modelling tool for systems and it presents randomness, concurrency, and synchronisation. Many researchers give attention to Petri nets because of its wide utilization. Therefore, reusability, sharing, and standardization are three functions that Petri nets must have. To actualized this concept, Semantic Web technologies can be implemented. One of the Semantic Web technologies is Ontology. In this paper, a formal approach for representing a business process in Petri nets into Ontologies is proposed. Firstly, a business process in Petri nets is presented. Then, formal definitions of Petri nets and OWL DL ontologies are introduced. Next step, converting a business process in Petri nets into OWL DL ontologies. Finally, using SPARQL Query in OWL DL ontologies to make sure that the relations of the business process in Petri nets are the same as the relations of the business process in OWL DL ontologies.**

Keywords—Petri nets; Ontology; SPARQL Query; business process; semantic web; protege

I. INTRODUCTION

A set of linked activities which is produced for specific service is the definition of the business process [1]. Business process has information about where and when the activities are executed, input and output of activity, initial condition before activity is executed and final condition after activity is executed. The characteristics of the business process itself are: have a specific purpose, have a specific input, have a specific output, utilize resource, have an activity that can be executed in a certain order [2], and it can involve more than one organization.

Business process model can represents business process correctly. There are a lot of ways to represent business process model, such as BPMN, UML, Causal Net, BPEL, EPC, PNML, etc [1]. Each type of model has different characteristic, such as Petri Net uses token to connect the activity in the business process model.

Petri nets is well-known as a mathematical and graphical language for the modeling of systems that present concurrency and resource sharing. They are one of the most studied process modeling language and used in many fields related to data processing, such as modeling of software engineer, data bases, and office automation [3]. However, in existing Petri net models, there are some inconsistent undestanding. These matters are caused by current Petri net interoperability which makes possible for model sharing [4]. In order to makes the researchers easy in constructing the models, the Petri net models must be standardized, shared, and reused well.

To actualized this concept, Semantic Web technologies can be implemented. One of the Semantic Web technologies is Ontology. The main advantage of ontology is easy to capture and represent the new knowledge from a certain domain in a formal way, i.e., classes, axioms, and properties.

Therefore, the concepts of Petri net and their relationship will be clear if there is a research about representing Petri nets into Ontology. This contribution will improve the common semantics and understandability among communities. In additional, because of its wide utilization, the needs of representing Petri nets into Ontologies are a must so that they can reuse on the Semantic Web effectively [5]. The current research only focus on developing a high-level mapping between Petri nets and Ontologies [6] as well as web services and ontologies.

In this paper, a formal approach for representing a business process in Petri nets into Ontologies is proposed. Firstly, a business process in Petri nets is presented. Then, formal definitions of Petri nets and OWL DL ontologies are introduced. Next step, converting a business process in Petri nets into OWL DL ontologies. Finally, using SPARQL Query in OWL DL ontologies to make sure that the relations of the business process in Petri nets are the same as the relations of the business process in OWL DL ontologies.

The remainder of this paper is organized as follows. In Section II, we review research which has been done in the representation of Petri Net into OWL DL Ontology. We introduce formal definition of Petri nets and OWL DL ontologies in Section III. The results and analysis are explained in Section IV. Last, this paper is concluded with conclusions and sketched future works in Section V.

II. RELATED WORK

Publishing Workflow Ontology (PWO) is developed to accommodate workflow requirements. The PWO is entirely based on the ontology patterns [7]. The description of the logical steps in a workflow is allowed in this ontology, such as the process of a document publication. Each step may require one or more activities (actions or events) which take place in a particular phase of the workflow. PWO has been implemented

based on the ontology patterns [8]. Table I explains patterns have been used in PWO.

TABLE I. A SUMMARY OF THE MAIN ENTITIES OF PWO WHICH HAVE RELATIONS TO SOME ENTITIES INTRODUCED IN THE PATTERNS USED

PWO entity	Pattern entity
Workflow	Plan (basic plan description, via basic plan)
WorkflowExecution	PlanExecution (basic plan execution, via basic plan)
Step	Task (task role, via control flow)
Action	Action (task execution and plan execution) TimeIndexedSituation (time-indexed situation)
hasStep	definesTask (basic plan description)
hasFirstStep	definesTask (basic plan description)
executes	satisfies (basic plan)
hasNextStep	directlyPrecedes (sequence)
hasPreviousStep	directlyFollows (sequence)
needs	isDefinedIn o describes (basic plan description)
produces	isExecutedIn (o hasParticipant) (task execution and participation)

III. REPRESENTATION A BUSINESS PROCESS IN PETRI NETS INTO ONTOLOGY

This subsection presents a formal definition of Petri net and OWL DL ontologies.

A. Petri net

A Petri net is a bipartite graph where the nodes are represented by *places* and *transitions* [9]. Each places can consist of *tokens* named positive integers. The goal of *tokens* is to simulate the ongoing *transitions* by flowing through the net. [9] calls the allocation of the *tokens* in the nets as *markings*. An example of a Petri Net is shown in Fig.1.

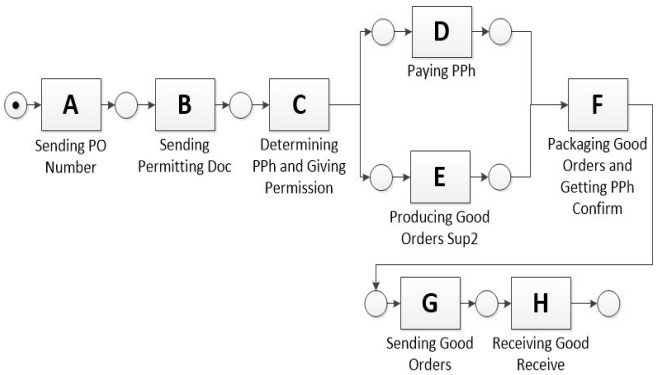


Fig.1 An example of Petri Net

Fig.1 represents the business process of purchasing goods production to a company. This process are modified from a business process in [10]. The process begins from Sending PO Number activity, goes to Sending Permitting Doc activity until reaching Receiving Good Receive activity. During the process, there are two activities that can be executed concurrently. The activities are Paying PPh and Producing Good Orders Sup2.

In Petri Net, the white and black rectangles represent *transitions* and the circles represent *places*. Every white *transition* contains an activity executing in the business process. Meanwhile, the black transitions represent *invisible transitions* or *invisible tasks***.** The example of white *transition* is *transition* A. Sending PO Number is an activity in *transition* A. In this research, we only focus on white transition or real executed activities in business process models.

The *places* is represented by circles, either front or back of *transitions*, in Petri Net. The *tokens* is represented by a black dot in the *places*. The example of a *place* that contains a *token* is the *place* before *transition* A in Fig.1.

B. Classical Petri net

The classical Petri net is the first formalism of Petri Net. As the formalism of Petri Net, *Classical Petri Net* have two node forms. It are *places* and *transitions*. The connectors of the nodes are *arcs*, represented by arrows. The rule of inputing *arcs* is the *arcs* cannot connect the same form of nodes. Specifically, the *arcs* cannot link the *places* and other *places*, and it also cannot link the *transitions* and other *transitions*. The definition of Petri Net as described in [11] is shown in Fig.2.

Places have symbols, $\bullet t$ and $t\bullet$. $\bullet t$ points of *input places* of *transition* t. Otherwise, $t\bullet$ points of *output places* of *transition* t. *Input places* are *places* which are linked to *transitions* by *arcs. Output places* are *places* which are linked from *transitions* by *arcs*. For example, the *place* between *transition* A and *transition* B in Fig.1 is the *output place* of *transition* A or the *input place* of *transition* B.

Petri net

A Petri net is a triple (P; T; F)*:*

P : *a finite set of places,*

T : *a finite set of transitions (*$P \cap T = \emptyset$*),*

$F \subseteq (P \times T) \cup (T \times P)$: *a set of arcs (flow relation).*

Fig.2 Definition of Petri Net

Corresponding with the *places*, *transitions* have two symbols, $\bullet p$ and $p\bullet$. $\bullet p$ expresses the *transitions* which use *place* p as its *input place*. $p\bullet$ expresses the *transitions* which use *place* p as its *output place*. Using the same example as the example in paragraph before, *transition* A is $\bullet p$ and *transition* B is $p\bullet$ for the *place* between those transitions.

The *markings*, allocation of *tokens*, is $M \in P \rightarrow IN$. The example of the *marking* is $0p_1 + 1p_2 + 1p_3$ or it can also be represented $1p_2 + 1p_3$. It means that *place* p_1 has no *token*, *place* p_2 has 1 *token*, and *place* p_3 has 1 *token*. Because the *tokens* simulate the outgoing *transitions*, the number *tokens* in each *places* will change during the simulation [12].

C. Representation of Petri net into OWL DL Ontology

A formal approach for representing a business process in Petri nets into Ontologies [13, 14] is described in Definition 1.

Definition 1. Representation of Petri nets. Given $PN = (P, T, F, M_0)$ is a Petri net, $O = \phi(PN) = (ID_0, Axiom_0)$ is the OWL DL ontology [15], so they can be defined by transformation function ϕ as follows:

- The OWL DL identifier set ID_0 of $\phi(PN)$ contains following elements explained in Table II.

TABLE II. REPRESENTATION OF PETRI NET INTO OWL DL ONTOLOGY

Place (P) $p_i \in P$	**is mapped into**	**Class identifier**
Transition (T) $t_i \in T$	**is mapped into**	**Class identifier**
arc (from P to T) $f_{ptot} \in F$	**is mapped into**	**Property identifier**
token in P $tk_i \in M_0$	**is mapped into**	**An Individual Property identifier**

After defining each form of Petri Nets, then we explain the function of class identifier and property identifier of OWL DL Ontology. Class identifier in OWL DL Ontology denotes all Places (P) and Transition (T) in Petri Nets PN. Meanwhile, property identifier denotes all the arcs in Petri Nets PN from Places (P) and Transition (T).

IV. EVALUATION

In this subsection, a real-life data set of an organization which has a focus on procurement of goods and services is used to evaluate our proposed method and we present the final result in SPARQL in Protege 5.1.0.

A. Evaluation Methodology

There are five steps done in this research to show that the semantic web can present the business process model, including the event (usually known as case ID), activity before, place before and activity after.

Step 1. Get the business process model of an organization. The business process can be modeled by various tools, such as YAWL (Yet Another Workflow Language), WoPeD (Workflow Petri Net Designer). In this research, we present the model using YAWL tool
Step 2. Define the individual, object property and class in Protege
Step 3. Run the Pellet Reasoner to get the new knowledge which can be generated from this business process without have to input manually all information
Step 4. Create the SPARQL Query in Protege
Step 5. Execute the SPARQL Query to get the result of business process. The results will show the event, activity before, place before and activity after

B. Business Process Model

In this research, a real-life event log data which are taken from an organization which has a focus on procurement of goods and services. Business process model in Petri nets is shown in Fig.3.

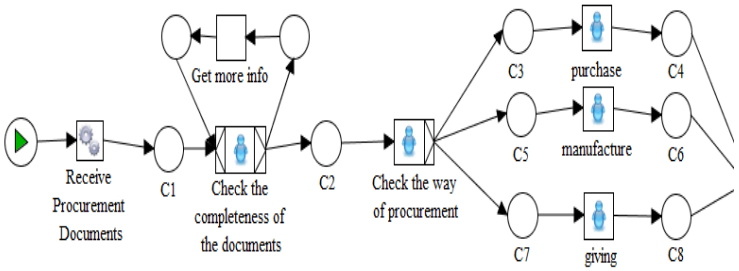


Fig.3 Part of business process model in Petri net

C. Protege

In order to evaluate our method, we use protege (an open source ontology editor and framework for building knowledge management system) as a tool to represent the Petri net into OWL DL ontology.

a. Individual

In protege, individual includes the transitions, cases, and places of the business process. Based on Fig.4, we generate some individuals:

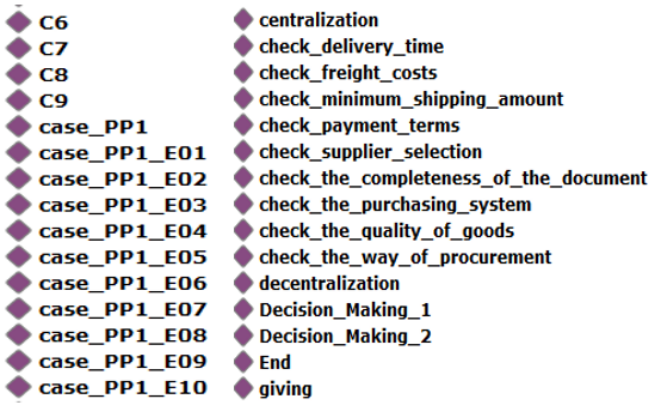
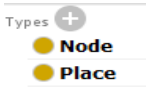


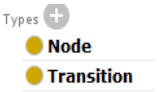
Fig.4 Individual of the business process in Protege

There are some types of invidual in protege. Based on this business process, we use four types of individual.

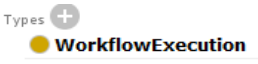
1. For all types of C, i.e. C1, C2,… : node and place



2. For all types of activities, i.e. 'receive procurement documents', 'check the completeness of the documents',… : node and transition



3. For all types of case_PP1,... : workflowexecution



4. For all types of case_PP1_E, i.e. case_PP1_E01, case_PP1_E02,… : action

b. Object Property

In this evaluation, we have nine object properties, namely:

1. arcPfromT : arc Place from Transition
2. arcPtoT : arc Place to Transition
3. arcTfromP : arc Transition from Place
4. arcTtoP : arc Transition to Place
5. has Activity
6. hasNextActivity
7. hasPreviousAction
8. involvesAction
9. isActionInvolvedIn

After we determine all the object properties, we need to assign each individual to object properties.

1. Object Properties of case_PP1_Exx

Based on Fig.3, we generate object properties of each Case ID, as shown in Fig.5 :

Fig.5 Object property of Case_PP1_Exx of the business process in Protege

2. Object Properties of Cx

Based on Fig.3, we generate object properties of each case and trace of activity, as shown in Fig.6:

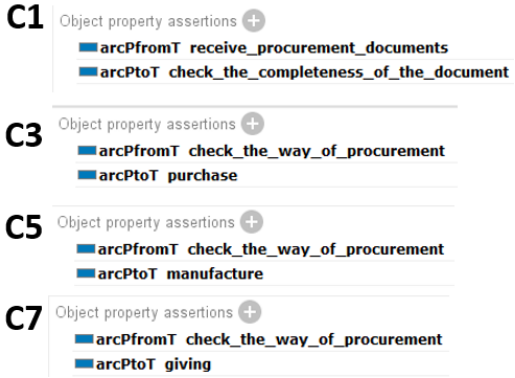


Fig.6 Object property of Cx of the business process in Protege

3. Object Properties of Activity

Based on Fig.3, we generate object properties of each activity, as shown in Fig.7:

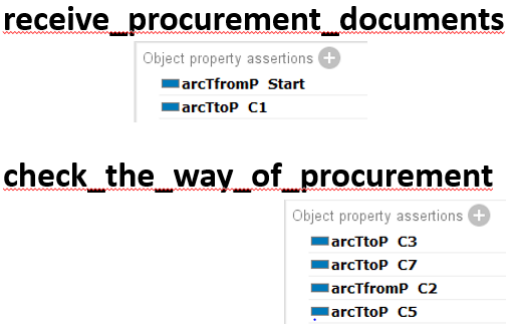


Fig.7 Object property of Activity of the business process in Protege

c. Class

In this evaluation, we have three main classes, namely:

1. Action
2. Node
   Place and Transition are in the class of node.
3. WokflowExecution

After we determine all classes, we need to assign each individual to each class, as shown in Fig.8.

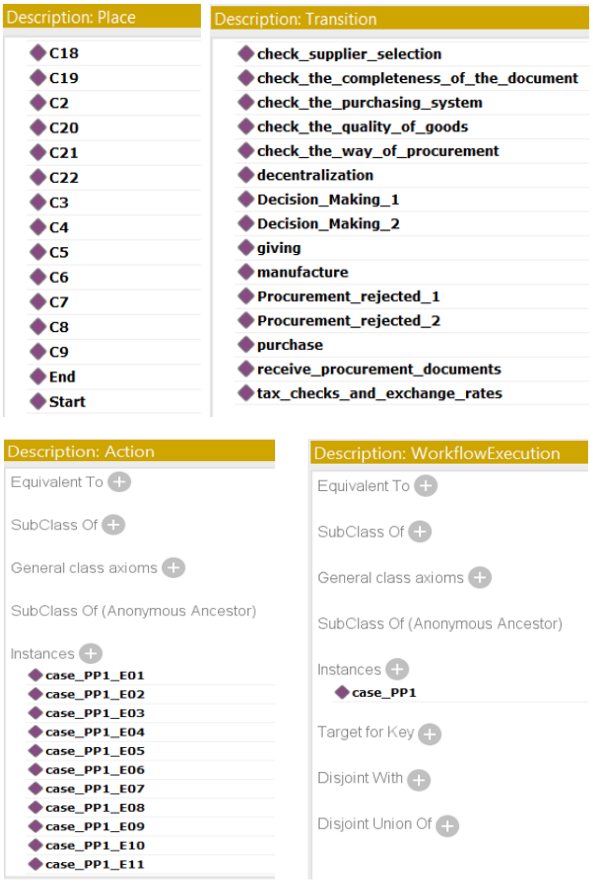


Fig.8 Classes of the business process in Protege

D. PELLET REASONER

A reasoner (semantic reasoner or reasoning engine or rules engine) is a free software which able to infer logical consequences from a set of asserted facts or axioms. Generalizing an inference engine and providing a richer set of mechanisms to work with are the notion of a reasoner. In this research, we use Pellet reasoner to run the rules.

From the reasoner, we get the new knowledge which can be generated from this business process without have to input manually all information. Based on Fig.9, we know that activity 'End' has arc from 'tax_checks_and_exchange_rates', 'procurement_rejected_2' and 'procurement_rejected_1'. For 'Start', it has arc place to transition 'receive_procurement_documents'.

E. SPARQL

SPARQL, which stands for The Simple Protocol and RDF Query Language, is a query language from the W3C. SPARQL is used for searching data defined in the RDF format. In this evaluation, we use SPARQL Query in OWL DL ontologies to make sure that the relations of the business process in Petri nets are the same as the relations of the business process in OWL DL ontologies. The SPARQL we use in this research is shown in Fig.10. Event, actbef, actaft mean activity, activity before and activity after in the process model. We use hasPreviousAction, hasActivity, arcTtoP and arcPtoT as object properties as explained in Section IV.B.

After we determine the SPARQL Query in protege, we execute the query to generate the result whether the activity, arc, and place in OWL DL ontology are the same as the activity, arc, and place in Petri net. From the Fig.11 the result of SPARQL Query is shown that the activity, arc, and place in OWL DL ontology are the same as the activity, arc, and place in Petri net. For case_PP1_E06, it shows that activity before is 'Decision_Making_1', place of activity before is 'C9', and activity after is 'check_the_purchasing_system.

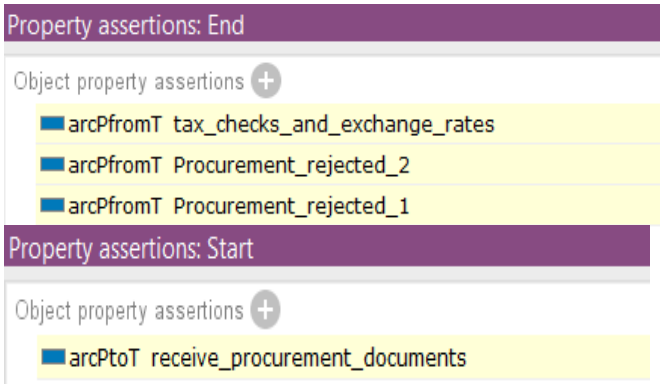


Fig.9 New knowledge generated using Pellet reasoner

```
SELECT ?event ?actbef ?p_actbef ?actaft
WHERE { ?event log:hasPreviousAction ?o. ?event
log:hasActivity ?actaft. ?o log:hasActivity ?actbef. ?actbef
log:arcTtoP ?p_actbef. ?p_actbef log:arcPtoT ?actaft}
```

Fig.10 SPARQL Query for analyzing the Petri nets model

Based on the results shown in Fig.11, we conclude that semantic web can present the business process in Petri net same as the business process model in Petri net modeled by YAWL tool. We explain the final results in Table III.

TABLE III. COMPARISON BETWEEN YAWL TOOL AND PROTEGE

Business process modeled by YAWL tool (based on Fig.3)	**Business process modeled by Protege (based on Fig.10)**
Activity now: Activity 'receive_procurement_documents' Place as link: C1 Activity after: Activity 'check_the_completeness_of_the_document'	Activity before: Activity 'receive_procurement_documents' Place before: C1 Activity after: Activity 'check_the_completeness_of_the_document'
Activity now: Activity 'check_the_completeness_of_the_document' Place as link: C2 Activity after: Activity 'check_the_way_of_procurement'	Activity before: Activity 'check_the_completeness_of_the_document' Place before: C2 Activity after: Activity check_the_way_of_procurement
..... until the process model is complete	..... until the process model is complete

V. CONCLUSION

Based on research, we proposed a formal approach for representing a business process in Petri nets into Ontologies in this paper. Before we define the formal approach, the formal definitions of OWL DL ontologies and Petri nets are introduced. The evaluation of this research were done using Protege and SPARQL Query.

The work was tested on a real-life event log data which are taken from an organization which has a focus on procurement of goods and services. The result shows that our method is able to represent a business process in Petri nets into Ontology using SPARQL Query. The relations of the business process in Petri nets are the same as the relations of the business process in OWL DL ontologies.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX log: <http://www.semanticweb.org/yutika/ontologyProcurement#>

SELECT ?event ?actbef ?p_actbef ?actaft
WHERE { ?event log:hasPreviousAction ?o. ?event log:hasActivity ?actaft. ?o log:hasActivity ?actbef. ?actbef log:arcTtoP ?p_actbef. ?p_actbef log:arcPtoT ?actaft}
```

event	actbef	p_actbef	actaft
case_PP1_E13	check_freight_costs	C21	check_payment_terms
case_PP1_E06	Decision_Making_1	C9	check_the_purchasing_system
case_PP1_E10	check_delivery_time	C18	check_minimum_shipping_amount
case_PP1_E12	check_the_quality_of_goods	C20	check_freight_costs
case_PP1_E09	check_supplier_selection	C17	check_delivery_time
case_PP1_E05	manufacture	C6	Decision_Making_1
case_PP1_E11	check_minimum_shipping_amount	C19	check_the_quality_of_goods
case_PP1_E08	Decision_Making_2	C15	check_supplier_selection
case_PP1_E04	check_the_way_of_procurement	C5	manufacture
case_PP1_E03	check_the_completeness_of_the_document	C2	check_the_way_of_procurement
case_PP1_E14	check_payment_terms	C22	tax_checks_and_exchange_rates
case_PP1_E02	receive_procurement_documents	C1	check_the_completeness_of_the_document

Fig.11 The results in Protege after SPARQL Query is executed

In the future, we intend to extend the approach to represent more complex Petri nets and develop an automated tool. Moreover, we want to identify the short loop, invisible task, and non-free choice using OWL DL ontology based on event log ontology.

ACKNOWLEDGMENT

We would like to thank Department of Informatics, Faculty of Information and Communication Technology, Institut Teknologi Sepuluh Nopember for supporting the research.

REFERENCES

[1] Y. A. Effendi and R. Sarno, "SWRL Rules for Identifying Short Loops in Business Process Ontology Model," International Conference On Information & Communication Technology And System (ICTS), PP.209-214, 2017. DOI: 10.1109/ICTS.2017.8265672

[2] R. Sarno and Y. A. Effendi, "Hierarchy Process Mining from Multi-Source Logs," Telecommunication, Computing, Electronics and Control (TELKOMNIKA), Vol.15, No.4, 2017
DOI: http://dx.doi.org/10.12928/telkomnika.v15i4.6326

[3] Fu Zhang, Z. M. Ma, Slobodan Ribarić, "Representation of Petri Net with OWL DL Ontology", in 8th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), vol. 3, pp. 1396-1400, 2011

[4] Noy, N. "Semantic Integration: a Survey of Ontology-Based Approaches", ACM SIGMOD Record, Vol. 33 No. 4, pp. 65-70, 2004.

[5] Uschold, M. and M. Gruninger, "Ontologies and semantics for seamless connectivity", ACM SIGMOD Record, Vol. 33 No. 4, pp. 58-64, 2004.

[6] Gruninger, M. and J. Lee, "Ontology applications and design: Introduction", Communications of the ACM, Vol. 45 No. 2, pp. 39-41, 2002.

[7] S. Peroni, "Example of use of PWO : publishing domain. figshare", 2015 http://dx.doi.org/10.6084/m9.figshare.1449043

[8] A. Gangemia, S. Peronib, D. Shottonc, F. Vitalid, "The Publishing Workflow Ontology (PWO)", 2015

[9] W. M. P. van der Aalst, K M. van Hee: "Workflow Management: Models, Methods, and Systems", MIT Press 2002.

[10] Y. A. Effendi and R. Sarno, "Discovering optimized process model using rule discovery hybrid particle swarm optimization," 3rd International Conference on Science in Information Technology (ICSITech), pp. 97-103, 2017. DOI: 10.1109/ICSITech.2017.8257092

[11] Y. A. Effendi and R. Sarno, "Discovering process model from event logs by considering overlapping rules," 4th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), pp. 1-6, 2017. DOI: 10.1109/EECSI.2017.8239193

[12] K. D. Febriyanti, R. Sarno, and Y. A. Effendi, "Fraud Detection on Event Log Using Fuzzy Association Rule Learning," International Conference On Information & Communication Technology And System (ICTS), pp.149-154, 2017. DOI: 10.1109/ICTS.2017.8265661

[13] A. Brogi, S. Corfini, S. Iardella, "From OWL-S Descriptions to Petri Nets", Proc. of ICSOC'07 Workshops, 2007, pp. 427-438.

[14] D. Gasevic, V. Devedzic, "Interoperable Petri net models via ontology", International Journal of Web Engineering and Technology 3(4), 2007, pp. 374-396.

[15] P. Soffer, M. Kaner, Y. Wand, "Assigning Ontology-Based Semantics to Process Models: the Case of Petri Nets", In: Bellahsène, Z. (eds.) CAiSE 2008. 2008, pp. 16-31.