BPMN Formalisation using Coloured Petri Nets

Mohamed Ramadan
Computer Science
Arab Academy for Science and Technology, Cairo, Egypt
mohamed.e.ramadan@gmail.com

Hicham G. Elmongui
Computer and Systems Engineering
Alexandria University
Alexandria, Egypt
elmongui@alexu.edu.eg

Riham Hassan
Computer Science
Virginia Tech
Blacksburg, VA, USA
rhabdel@cs.vt.edu

Abstract— **Business process modeling is an increasingly popular method for improving organizational efficiency and quality. The ability to automatically validate the process model became a significant feature of modeling tools with the increasing complexity of enterprise business processes and richness of modeling languages. This paper proposes formal semantics for process modeling by mapping Business Process Modeling Notations (BPMN) to Coloured Petri Nets (CPN). We automate the transformation process to allow for automatically validating the business process model. Formalizing BPMN using CPN enables simulating business process behavior to facilitate the early detection of flaws.**

Keywords- Business Process Modeling Notation; BPMN; Coloured Petri Nets; Formalisation; Semantics; Verification; Validation

I. INTRODUCTION

Business Process Management (BPM) is concerned with the methods, techniques and software developed to design, control, and analyze operational processes involving humans, organizations, applications, documents and other sources of information [1]. Organizations have realized the cost effectiveness of BPM that emerges from flaw detection and automation of business processes. [2].

Business process modeling is the visual representation of business processes. It is an effective method for designing, analyzing, simulating, and improving the current process [3].Business process modeling practice shows that syntactic and semantic inconsistencies often appear in process models [3]. Further, the complexity of modeling techniques and business processes is increasing recently, which influences the correctness of the resulting process models [3]. Consequently, vendors of business process modeling tools are urged to provide for model validation and consistency checking.

A formal semantics for the modeling languages assist in moving towards correct implementation of business processes. The existence of formal semantics enables modeling tool vendors to support automated model verification and validation [4].

In this paper, we propose formal semantics for business process models. The formal semantics is defined as a mapping between Business Process Modeling Notation (BPMN) and Colored Petri Nets (CPN). BPMN is a standard by Object Management Group (OMG) for business process modeling that provides the easy-to-use front-end to facilitate human understanding. Petri Nets formalism is a popular and powerful formal notation for the representation of processes, which exhibit concurrency, parallelism, synchronization, non-determinism, and mutual exclusion. The proposed mapping can be used in modeling tools to automate the process model validation. Our proposed formalization provides a comprehensive coverage of the various BPMN process model constructs along with their data specifications.

The rest of this paper is organized as follows. Section II provides an overview about BPMN and Coloured Petri Nets. Section III presents the mapping from BPMN to CPN. Section IV shows an example for the mapping. Finally Section V and Section VI discuss the related work and the conclusion.

II. BACKGROUND

This section gives background on the two related technologies. First, we summarize the BPMN 2.0 notation principles as it is the base for the work we propose in this paper. Next, we explain the concepts of the CPN as the formal mathematical modeling language.

A. Business Process Modeling Notation

Business Process Model and Notation (BPMN) is a standard for business process modeling by Object Management Group (OMG). BPMN provides a graphical notation for specifying business processes in a Business Process Diagram (BPD) [5]. The objective of BPMN is to support business process management for both technical users and business users by providing a notation that is intuitive to business users yet able to represent complex process semantics.

BPMN is designed to cover many types of modeling and allow the creation of end-to-end Business Processes [5].

There are three basic types of sub-models within an end-to-end BPMN model: Processes (Orchestration), Choreographies, and Collaborations.

The five basic categories of BPMN elements for Business Process Diagram (BPD) are: flow objects, data, connecting objects, swimlanes, and artifacts.

Flow Objects are the main graphical elements to define the behavior of a business process. There are three kinds of flow objects, which are event, activity, and gateway.

Data Items are the primary constructs for modeling data within the process flow**.** Data is represented with the four elements: data objects, data inputs, data outputs, and data stores.

Connection Objects are the graphical elements to connect the Flow Objects to each other. There are three kinds of Connecting Objects, which are Sequence Flow, Message Flow, and Association.

Swimlanes are the graphical elements to group the modeling elements. There are two ways of grouping the primary modeling elements, which are pools and lanes.

Artifacts are used to provide additional information about the Process. There are two standardized Artifacts, which are Group and Text Annotation, but modelers or modeling tools are free to add as many Artifacts as necessary.

This paper focuses on the formalization of BPMN process models (Orchestration).

B. Coloured Petri Nets

A Coloured Petri Net (CPN) is a graphical language for constructing models of concurrent systems and analyzing their properties. CPN is a discrete-event modeling language combining Petri Nets and the functional programming language CPN ML which is based on Standard ML. Standard ML provides the primitives for the definition of data types, describing data manipulation, and for creating compact and parameterisable models [6].

CPN is the most well-known kind of high-level Petri Nets. CPN incorporate both data structuring and hierarchical decomposition - without compromising the qualities of the original Petri Nets.

The choice of using CPN as a target for the mapping in this paper is motivated by the availability of more formal verification methods, state space analysis and invariant analysis. Also the existence of hierarchical CPN makes it possible to model large BPMN process structure and its sub-processes, and the coloured token concept allow the modeling of process data.

III. MAPPING BPMN ONTO COLOURED PETRI NETS

This section establishes a mapping scheme of the core BPMN models to Colored Petri Nets.

A. Common Elements

BPMN common elements might be used in more than one diagram type (e.g., Process, Collaboration, and Choreography). Common elements provide modelers with the capability of showing additional information about a Process such as operations, error definition, and resources that can be referenced by activities.

We produce a CPN ML declarations corresponding to the standard BPMN 2.0 constructs schema. These declarations allow one-to-one transformation of BPMN elements and their attributes.

Figure 1. shows the mapping of BPMN ItemDefinition schema definition onto a CPN ML fixed-length color set declaration. Also tItemKind enumeration, which defines the nature of the ItemDefinition with one of Information or Physical values, is mapped onto CPN ML enumeration color set.

```
colset tItemDefinition = record id: STRING *
                                isCollection: BOOL *
                                itemKind: tItemKind *
                                structureRef: STRING;

colset tItemKind =  with Information | Physical;
```

Figure 1. ItemDefinition BPMN XML schema

All other BPMN common elements schema are mapped onto CPN ML declarations using same idea of mapping the ItemDefinition which shown in last section. TABLE I. summarizes the mapping of Common Elements to CPN ML constructs mapping for the BPMN Common elements.

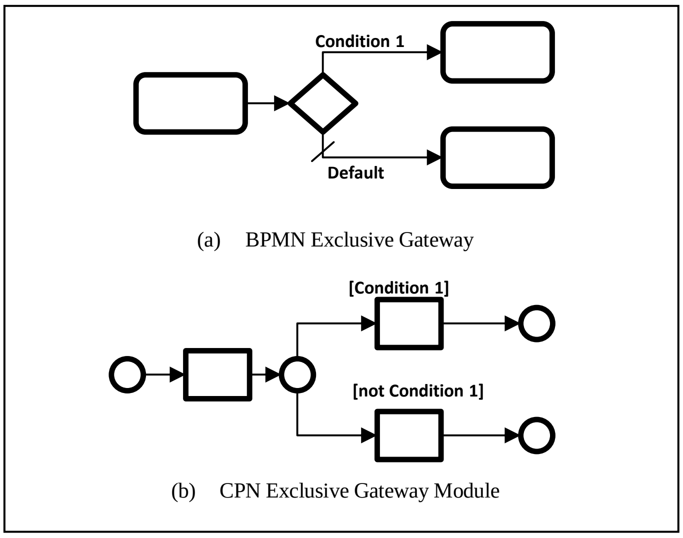
TABLE I. COMMON ELEMENTS MAPPING

BPMN Element	***CPN ML Declarations***
Message	colset **tOptionalItemDefinition** = list tItemDefinition with 0..1; colset **tMessage** = record id :STRING * name:STRING * itemRef: tOptionalItemDefinition;
Resource	colset **tResourceParameters** = record name: STRING* isRequired: BOOL* paramType: tItemDefinition; colset **tResource** = record id: STRING * name: STRING * resourceParameters: tResourceParametersList;
Error	colset **tOptionalItemDefinition** = list tItemDefinition with 0..1; colset **tError** = record id: STRING * name: STRING * errorCode: STRING * structureRef: tOptionalItemDefinition;
Interface	colset **tInterface** = record id: STRING * name:STRING * implementationRef: STRING;
Operation	colset **tOptionalMessageList** = list tMessage with 0..1; colset **tOperation** = record id: STRING * name:STRING * interfaceRef:tInterface * implementationRef: STRING * inMsg:tMessage * outMsg:tOptionalMessageList * errorRef: tErrorList ;

B. Gateways

Gateways are used to control how ssequence flows interact as they converge and diverge within a Process.

A diverging exclusive gateway (decision) is used to create alternative paths within a process flow. Figure 2. (b) shows the CPN ML mapping of the exclusive gateway in Figure 2. (a). As only one of the paths can be taken in the exclusive gateway, the mapping done using a place connected with arc to a transition for each path. BPMN condition expression that is associated with a gateway's outgoing sequence flows is mapped to a CPN transition with a ML guard inscription. The default path will be mapped as a path with inverse of all other conditions.



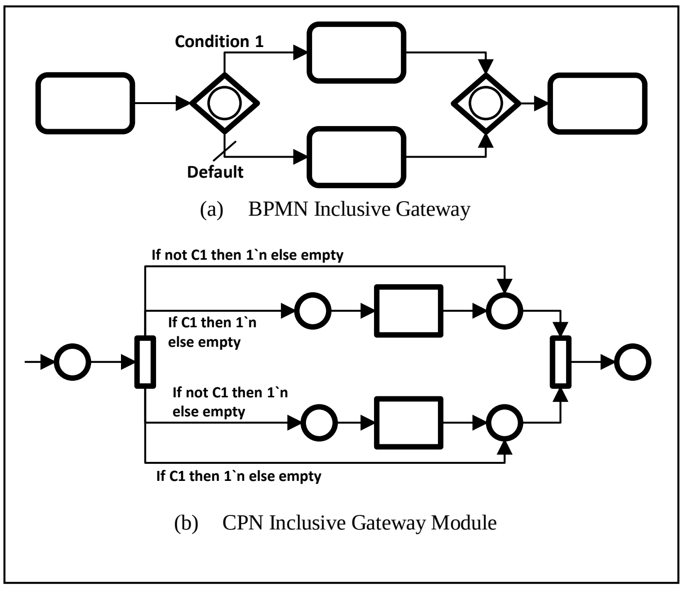
(a) BPMN Exclusive Gateway

(b) CPN Exclusive Gateway Module

Figure 2. Exclusive Gateway (Decision)

A diverging inclusive gateway (inclusive decision) can be used to create alternative but also parallel paths within a process flow.

Figure 3. (b) shows the CPN ML mapping of inclusive gateway in Figure 3. (a).



(a) BPMN Inclusive Gateway

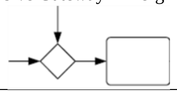
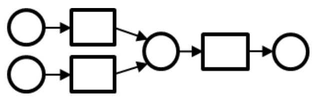
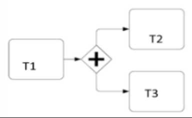
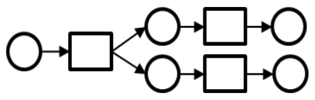
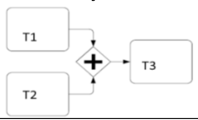
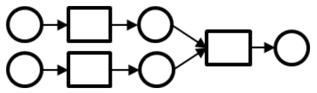
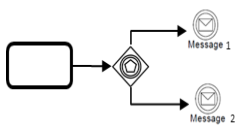
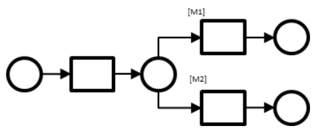
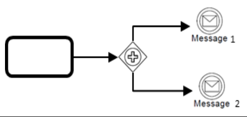
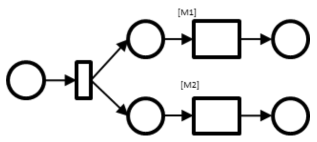
(b) CPN Inclusive Gateway Module

Figure 3. Inclusive Gateway (Decision/Merging)

Since each path is considered to be independent, all combinations of the paths may be taken, the mapping done using a Transition connected with arc to a place for each path. BPMN condition expression that is associated with a gateway's outgoing sequence flows is mapped to arc inscriptions.

TABLE II. summarize the mapping of other BPMN gateways onto CPN.

TABLE II. GATEWAYS MAPPING

BPMN Gateway	***CPN Mapping***
Exclusive Gateway – Merging 	
Parallel Gateway – Fork 	
Parallel Gateway – Join 	
Exclusive Event-based Gateway 	
Parallel Event-based Gateway 	

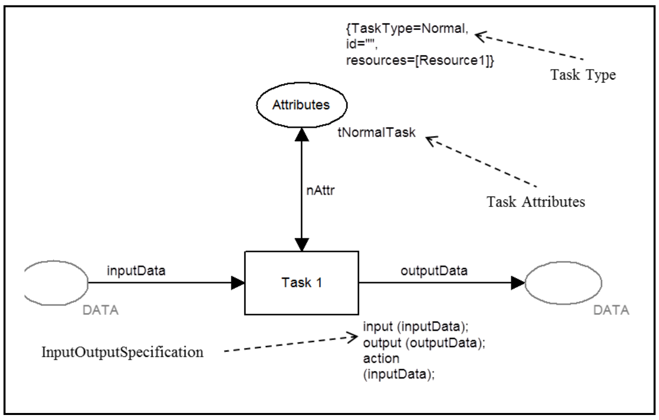


Figure 4. Task Transformation

- Complex Gateway

  Because complex gateway doesn't have a default semantic, which should be defined by the modeler, we don't provide a mapping for it to CPN. However the modeler should change it to other gateways to represent the gateway logic.

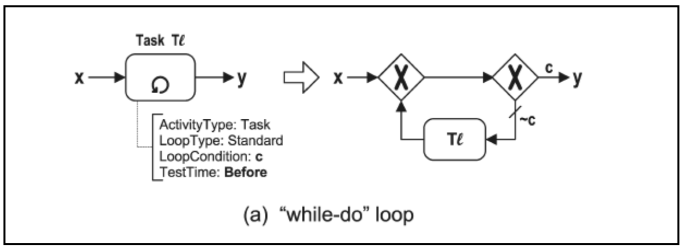
C. *Activities*

- Tasks

  Figure 4. gives a formal semantics for BPMN tasks in CPN. The CPN formal semantics have all task BPMN schema attributes mapped to CPN ML declarations. CPN ML colour set is created for each BPMN task type to cover all its additional attributes.

- Loop Activity

  Loop activity transformation idea is provided by [7] as shown in Figure 5.



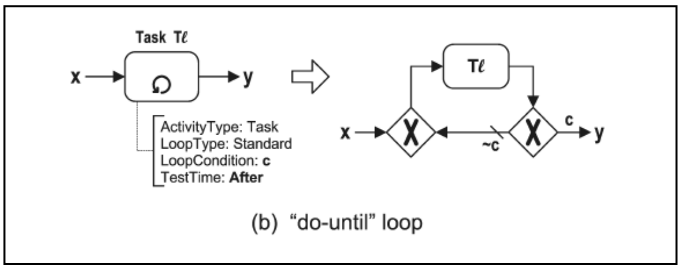


Figure 5. Loop Activity Transformation

- Multi-Instance Activity

  Multi-Instance activity can be set to be performed in sequential or parallel. Sequential mutli-instance activity will be transformed to while-do loop with n times as shown in Figure 5. (a). Parallel mutli-instance activity will be transformed to parallel gateway (fork) and parallel gateway (join) as provided by [7].

- Sub-Processes

  Sub-Process is transformed using the hierarchical concept of CPN. Each sub-process will be represented as a sub-page of the parent process page as shown in Figure 6.

- Transaction and Compensation

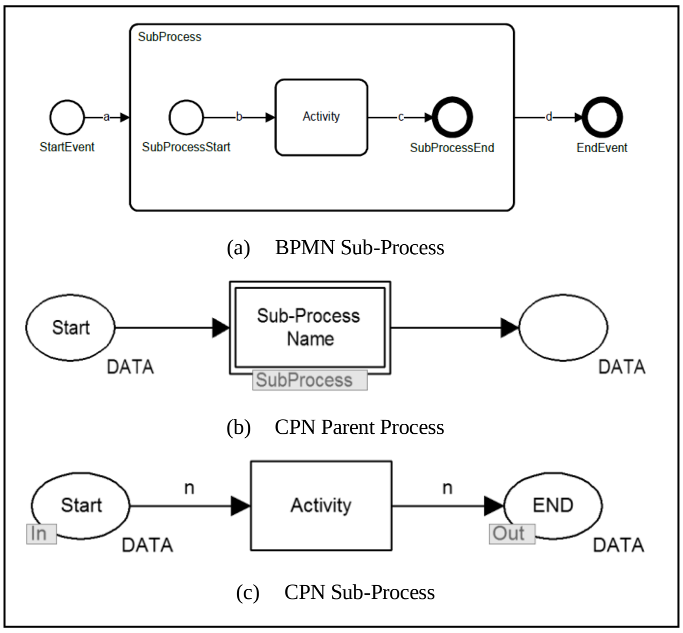
  Formal semantics for transaction and compensation activities is a descendent provided by [8].

- Global Task

  A Global Task is a reusable, atomic task definition that can be called from within any process by a call activity. Global task will be mapped to CPN subpage as a process with a one task activity as shown in Figure 6. (c).

- Call Activity

  The call activity acts as a 'wrapper' for the invocation of a global process or global task within the execution. Call activity will be mapped using same idea shown at Figure 6. but with CPN multiple instances subpage concept.



(a) BPMN Sub-Process

(b) CPN Parent Process

(c) CPN Sub-Process

Figure 6. Sub-Process Transformation

D. *Events*

An event is something that "happens" during the course of a process. These events affect the flow of the process and usually have a cause or an impact and in general require or allow for a reaction. There are three main types of events, which are start, end, and intermediate events. Figure 7. (a-b) show how start and end events will be mapped to CPN constructs. Figure 7. (c) shows how the event will be mapped in case of throw a result or catch a trigger (e.g. Message event) using a CPN transition with ML guards condition.

Start events for event sub-processes can be interrupting or non-interrupting.

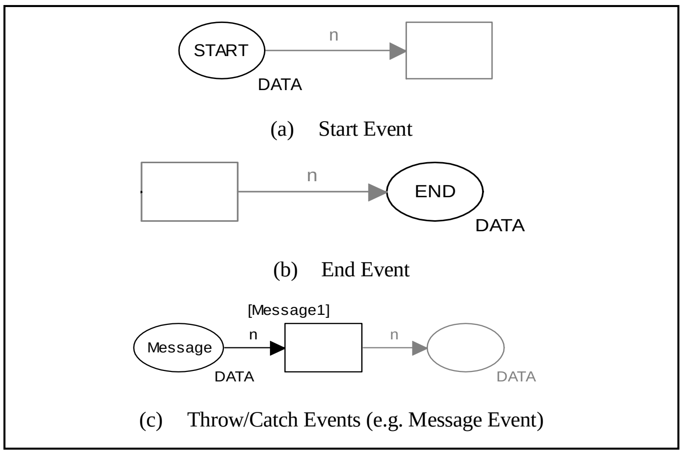
Non-interrupting events allow unlimited number of event sub-processes for the same event declaration can be modeled and executed in parallel. Non-interrupting start event for event sub-processes will be mapped as normal Start event mapping shown in Figure 7. (a). Interrupting events allow only one event sub-processes for the same event declaration. Whenever the event occurs, the associated activity is terminated. Figure 8. shows the mapping of BPMN interrupting events onto CPN.

E. *Data*

This paper represents a mapping for BPMN task InputOutputSpecification using CPN transition code segments inscription as shown in Figure 4. . InputOutputSpecification InputSet and OutputSet will be mapped onto CPN transition code segment input and output patterns respectively. BPMN event DataInput or DataOutput items definitions are mapped using the CPN place color inscription as shown in Figure 7. .

F. Transformation Process

This paper suggests a process for mapping BPMN process onto CPN as shown in Figure 9. (a). Transformation of BPMN common elements should be in order of the dependency shown in Figure 9. (b).



(a) Start Event

(b) End Event

(c) Throw/Catch Events (e.g. Message Event)

Figure 7. Events Transformation

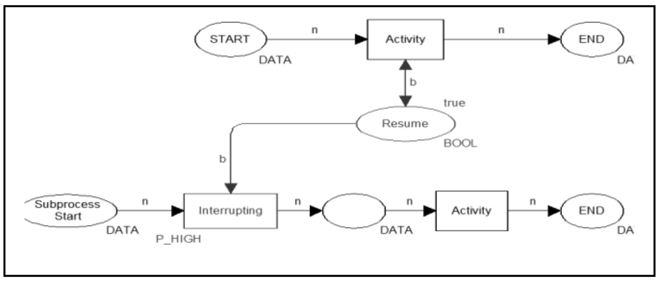


Figure 8. Intrrupting Events Transformation

IV. EXAMPLE

Figure 10. describes a business process for incident management process of a software manufacturer is triggered by a customer requesting help from his account manager because of a problem in the purchased product. The assignment of tickets to 1st and 2nd level support agents is automated by a trouble ticket system and modeled at the Trouble Ticket System process in Figure 10.

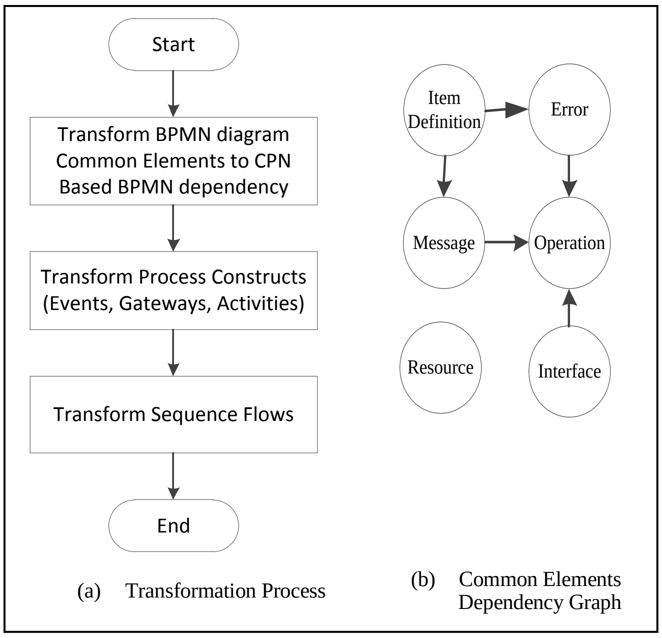
Using the mapping from BPMN onto CPN described in the previous section, the Trouble Ticket System process in Figure 10. is mapped onto the CPN in Figure 11.

V. RELATED WORK

The ability to statically check the semantics correctness of models is a desirable feature for modeling tools. Some approaches are developed to give formal semantics to model-based process languages such as Event-Condition-Action (ECA) business rules, UML activity diagram, and BPMN. Researches use several formalisms methods like π-calculus as a mathematical formalism, Petri Net as a mathematical modeling language, or Graph Rewrite Rules.

In [9] authors define ECA business rules for modeling processes and workflows. In [10] authors are using the π-Calculus for formalizing these workflow patterns. Using ECA business rule to model the business process provides gateways and events, but not exception handling or multiple instances of sub processes as in BPMN, thus making the formalization is easier.

In [11] authors provide a formalization of process modeling by UML Activity diagram using the π-Calculus. While In [12] author provides a similar formalization using the Petri Nets and Coloured Petri Nets. UML 2 activity diagram is more similar to BPMN, however BPMN have more business constructs to visualize business process.


Start
Transform BPMN diagram Common Elements to CPN Based BPMN dependency
Transform Process Constructs (Events, Gateways, Activities)
Transform Sequence Flows
End
Item Definition
Error
Message
Operation
Resource
Interface

(a) Transformation Process

(b) Common Elements Dependency Graph

Figure 9. Transformation Process

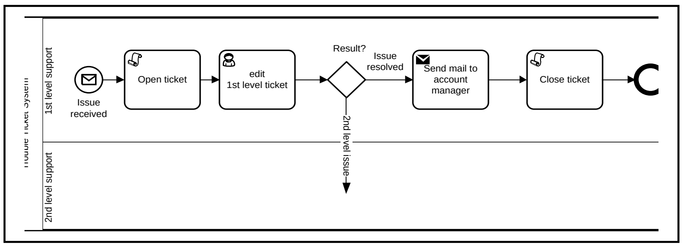
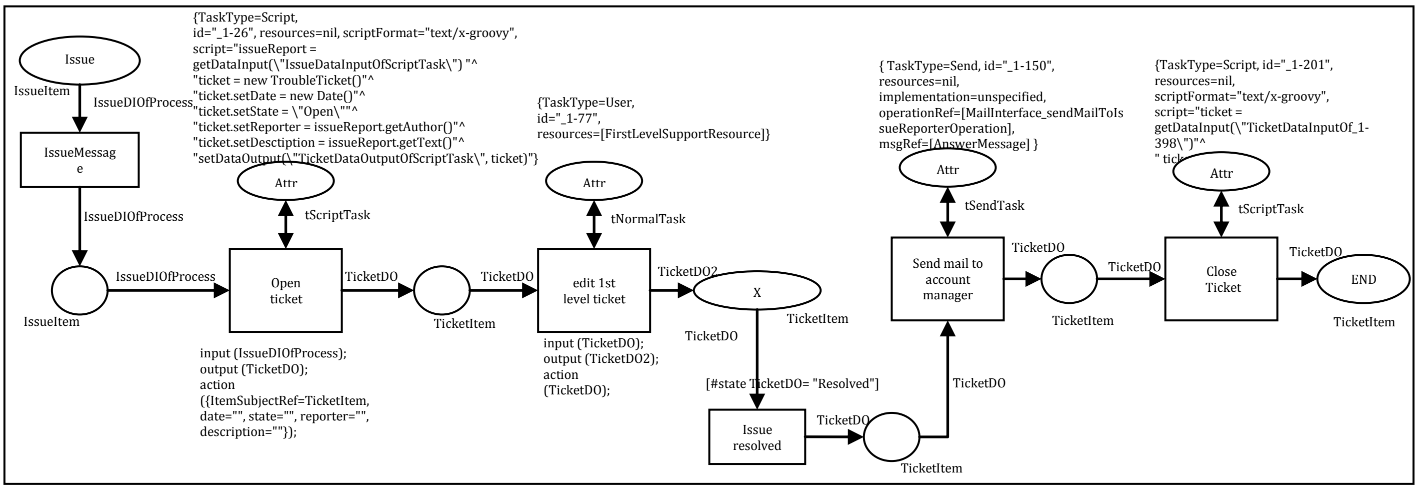


Figure 10. BPMN Incident Management Example

In [7] authors propose a mapping from core set of BPMN to Petri Nets. The paper focuses on the control-flow perspective of BPM, and does not deal with its process data and tasks input/output specifications. Paper also work based on earlier versions of the BPMN standard and raised up some issues; transaction transformation and compensation tasks, and OR-join gateways. For OR-join gateways issue was given formal semantics in [13] and [14]. [8]proposed a formal transformation for Transaction and Compensation activities.



```
{TaskType=Script,
id="_1-26", resources=nil, scriptFormat="text/x-groovy",
script="issueReport =
getDataInput(\"IssueDataInputOfScriptTask\") "^
"ticket = new TroubleTicket()"^
"ticket.setDate = new Date()"^
"ticket.setState = \"Open\""^
"ticket.setReporter = issueReport.getAuthor()"^
"ticket.setDesctiption = issueReport.getText()"^
"setDataOutput(\"TicketDataOutputOfScriptTask\", ticket)"}
```

```
input (IssueDIOfProcess);
output (TicketDO);
action
({ItemSubjectRef=TicketItem,
date="", state="", reporter="",
description=""});
```

```
{TaskType=User,
id="_1-77",
resources=[FirstLevelSupportResource]}
```

```
input (TicketDO);
output (TicketDO2);
action
(TicketDO);
```

```
{ TaskType=Send, id="_1-150",
resources=nil,
implementation=unspecified,
operationRef=[MailInterface_sendMailToIs
sueReporterOperation],
msgRef=[AnswerMessage] }
```

```
{TaskType=Script, id="_1-201",
resources=nil,
scriptFormat="text/x-groovy",
script="ticket =
getDataInput(\"TicketDataInputOf_1-
398\")"^
" tick
```

Figure 11. CPN Incident Management Example

In [15], authors presents a formalization of a subset of the BPMN 2.0 execution semantics in terms of graph rewrite rules. Paper provided semantics is less suitable for process correctness verification and also doesn't cover the data and resource aspects of BPMN. In [16] author proposes to reuse an existing discrete event simulator based on CPN as a simulator of BPMN process models. Paper focuses more on the process simulation and the current version of the meta-model is quite general and needs an improvement. Also there are lots of specific attributes of simulation which left out from the scope of simualtion.

VI. CONCLUSION

This paper presents a formalization of the business process modeling notation (BPMN) as a standard process modeling language using Coloured Petri Nets (CPN). Not only does our mapping span BPMN 2.0 orchestration constructs, but also cover aspects such as the data and activity attributes. Moreover, a CPN ML declaration schema is built for each BPMN XML schema definition. To connect the dots, we presented the mechanism of mapping a full BPMN model onto its corresponding CPN. Defining formal semantics for business process modeling languages helps the modeling tools vendors to automatically validate the business model. Further, it helps business analysts simulate the business process behavior to enable detection of flaws.

Our ongoing work is directed towards building an automated tool for the transformation. Such automation should help tool vendors verify the correctness of process models. Business analysts could also employ our automation to use CPN simulation tools for process behavior analysis.

REFERENCES

[1] Ryan K.L. Ko, Stephen S.G. Lee, and Eng Wah Lee, "Business process management (BPM) standards: a survey," *Business Process Management Journal*, vol. 15, pp. 744--791, 2009.

[2] Marc Fasbinder, Why model business processes?, 2007.

[3] L. J. Hommes, "The Evaluation of Business Process Modeling Techniques," Delft University of Technology, Ph.D. thesis 90-9017698-5, 2004.

[4] Peter Y.H. Wong and Jeremy Gibbons, "A process semantics for BPMN," *Formal Methods and Software Engineering*, pp. 355--374, 2008.

[5] Object Management Group, *Business Process Model and Notation (BPMN)*, 20th ed.: Object Management Group, 2011.

[6] L. M. Kristensen, L. Wells, and K. Jensen, "Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems," *STTT*, vol. 9, pp. 213--254, 2007.

[7] R.M. Dijkman, M. Dumas , and C. Ouyang, "Formal Semantics and Analysis of BPMN Process Models using Petri Nets," *Queensland University of Technology, Tech. Rep*, 2007.

[8] Tsukasa Takemura, "Formal Semantics and Verification of BPMN Transaction and Compensation," in *APSCC'08. IEEE*, 2008, pp. 284--290.

[9] G. Knolmayer, R. Endl, and M. Pfahrer, "Modeling Processes and Workflows by Business Rules," *Business Process Management*, pp. 201--245, 2000.

[10] F. Puhlmann and M. Weske, "Using the pi-Calculus for Formalizing Workflow Patterns," *Business Process Management*, pp. 153--168, 2005.

[11] Yang Dong and Zhang ShenSheng, "Using pi-calculus to Formalize UML Activity Diagram for Business Process Modeling," in *ECBS`03, IEEE*, 2003, pp. 47-54.

[12] T.S. Staines, "Intuitive mapping of UML 2 activity diagrams into fundamental modeling concept Petri net diagrams and colored Petri nets," in *ECBS`08, IEEE*, 2008, pp. 191--200.

[13] M. Dumas, A. Grosskopf, T. Hettel, and M. Wynn, "Semantics of Standard Process Models with OR-Joins," *OTM`07*, pp. 41--58, 2007.

[14] D. Christiansen, M. Carbone, and T. Hildebrandt, "Formal Semantics and Implementation of BPMN 2.0 Inclusive Gateways," *Web Services and Formal Methods*, pp. 146--160, 2011.

[15] Remco Dijkman and Pieter Gorp, "BPMN 2.0 Execution Semantics Formalized as Graph Rewrite Rules," in *Second International Workshop, BPMN 2010*, Potsdam, 2011, p. 16.

[16] Marek Zäuram, "Business Process Simulation Using Coloured Petri Nets," Institute of Computer Science, University of Tartu, Tartu, Master's Thesis 2010.