An Approach to Workflow Modeling and Analysis

Hemant Kumar Meena hkmeena@cse.iitk.ac.in
Indradeep Saha indra@cse.iitk.ac.in
Koushik Kumar Mondal kkmondal@cse.iitk.ac.in
T.V. Prabhakar tvp@cse.iitk.ac.in
Department of Computer Science & Engineering,
IIT Kanpur,
India

***Abstract*— In this paper we present a new approach to workflow analysis. There are efforts to design and verify workflow models using both Activity diagrams and Petri nets. We model the workflow using Activity diagrams, convert the Activity diagrams to Petri nets and use the theoretical results in Petri nets to analyze the equivalent Petri nets and infer properties of the original workflow. We have demonstrated the possibility by developing an Eclipse plug-in which can be used to model workflows using Activity Diagrams and then analyze these workflow models using Petri nets.**

***Index Terms*— workflow, activity diagrams, Petri nets, eclipse, workflow analysis**

I. Introduction

WORKFLOW modeling needs a language that is intuitive and easy to use. Activity diagrams from UML 2.0 provide a good option. However, we need something more formal to analyze the workflow so modeled. Petri net offers that. In this paper we propose to use activity diagrams for workflow modeling and then use Petri net to analyze the workflow. Our work demonstrates that properties of workflow can be inferred from corresponding Petri net model. We have also built a plug-in on Eclipse [17] which provides an editor for workflow modeling using activity diagram. We then analyze this diagram by converting it into a corresponding Petri net model.

II. WORKFLOW

Workflow refers to automation of business processes, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules [5]. A workflow management system (WFMS) is a software package that is used to define, create and manage the execution of workflows. While designing a workflow, one describes which tasks have to be done and in what order. So process approach is given more importance. Hence it is important that a good modeling language is used to design a workflow.

III. Activity diagram for modeling Workflows

Activity diagram from UML 2.0 provides all the basic constructs needed. Major constructs for workflow modeling are sequence, parallel path, alternative path and iteration. Activity diagram constructs, start, end, fork, decision, and activity can be used for modeling all these constructs. Start can be used to indicate beginning of a process where as end can be used to indicate end of a process. Fork can be used for splitting a process into several parallel execution paths. Decision can be used for providing alternative paths. We can also model iteration by connecting two decisions.

IV. Petri Nets

A Petri net is a directed graph with two types of nodes called places and transitions. An arc connects between two nodes. A connection can be from a place to a transition or from a transition to a place.
Formal definition of Petri nets is as follows [2]:

A Petri Net is a 5-tuple, $PN = (P, T, F, W, M_0)$ where:
$P = \{p_1, p_2, p_{3\ldots}\ p_m\}$ *is a finite set of places,*
$T = \{t_1, t_2, t_3 \ldots t_n\}$ *is a finite set of transitions,*
$F \subseteq (P \times T) \cup (T \times P)$ *is a set of arcs,*
$W: F \rightarrow \{1, 2, 3\ldots\ldots\}$ *is a weight function,*
$M_0: P \rightarrow \{0, 1, 2, 3\ldots\ldots\}$ *is the initial marking,*
$P \cap T = \Phi$ *and* $P \cup T \neq \Phi$

Marking denotes initial distribution of tokens among places. A transition is said to be enabled if each of its input place contains at least that number of tokens which is equal to arc joining the place and transition. An enabled transition may fire i.e. tokens are removed from its input places and added to output places.
While representing graphically, places are drawn as circles, transitions are drawn as rectangles, tokens as black dots, and arcs as arrows.
Petri nets can also be used for modeling workflows [1]. However, it is not as intuitive as activity diagrams and hence it is not easy to model workflows with it. We instead can use Petri nets for analyzing workflows. Huge amount of research has been done in the Petri net domain and it has been explored to a great extent. So Petri net properties can be used to analyze workflows.

V. USEFUL PROPERTIES of Petri nets

Here we present some of the useful properties of Petri net.

Reachability: We start with some initial distribution of tokens among places which we call initial marking of the given Petri net. Now when an enabled transition fires, the distribution of tokens change. Starting with an initial marking we can construct a reachability tree which will produce all possible reachable markings. Since a marking represents a state in Petri net, from a reachability tree we can find out all possible reachable states of the given Petri net.

Coverability: Given a Petri net with initial marking M_0, a reachable marking M_1 is said to be coverable if there exist another marking M_2 whose distribution of tokens among places is either greater or equal to that of M_1.

Boundedness: A given Petri net with initial marking M_0 is said to be bounded if for any reachable marking, the number of tokens in each place does not exceed a finite value.

Safeness: A given Petri net with initial marking M_0 is safe, if it is bounded and maximum allowable token in each place is 1.

Liveness: A given Petri net with initial marking is said to be live, if from any reachable marking it is possible to fire any transition after some firing sequence. A transition t is said to be dead, if it can never be fired. If in a firing sequence we reach a point where a particular transition can not be fired, then the net is in a potential deadlock.

VI. Activity diagrams to Petri nets

An Activity diagram can be mapped to a Petri net which includes all kinds of control flow [4]. Here activity and fork nodes are mapped to Petri net transitions and start, end, and decision nodes are mapped to places. Connections are mapped in such a way that always there is an arc either from transition to place or place to transition. The converted Petri net model can be represented using Petri Net Markup Language (PNML) [3]. PNML is an XML based interchange format for Petri nets. This is useful for importing and exporting a Petri net model.

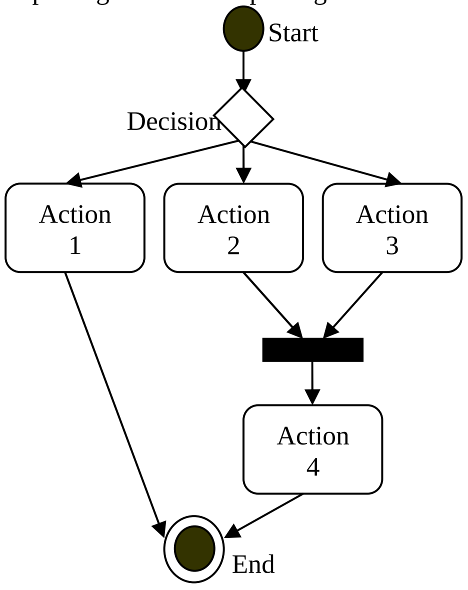


Fig. 1. An example of Activity diagram

Figure 1 shows an Activity diagram and Figure 2 shows the corresponding Petri net obtained after converting it.

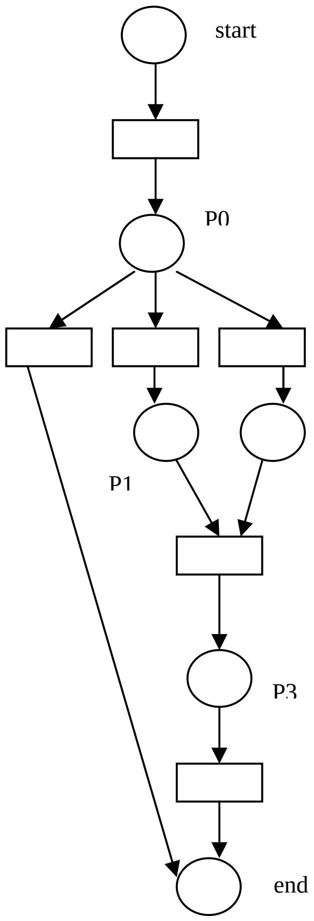


Fig. 2. Petri Net mapping of Fig. 1

VII. Analysis of workflows

Surely it is not enough to only design a workflow. It is also necessary to analyze it. As we have mentioned in an earlier section, we can use Petri nets to analyze workflows. A huge amount of work has been done on Petri nets so far and hence a huge number of results are available. One needs to find out a set of results which can help in analyzing workflows. Two such useful methods are coverability tree and incidence matrix.

A coverability tree is actually a reachability tree with some modification to take care of the case when the given Petri net is not bounded.

The coverability tree of a given Petri net with initial marking M0 is constructed using the following algorithm [2]:

- Label the initial marking M, as the root and tag it "new."
- While "new" markings exist, do the following:
    - Select a new marking M.

If M is identical to a marking on the path from the root to M, then tag M

- "old" and go to another new marking.
- If no transitions are enabled at M, tag M "dead-end."
- While there exist enabled transitions at M, do the following for each enabled transition t at M:
    - Obtain the marking M' that results from firing t at M.
    - On the path from the root to M if there exists a marking M" such that M'(p)>M"(p) for each place p and M'$\neq$M", i.e., M" is coverable, then replace M'(p) by ω for each p such that M'(p) > M"(p).
    - Introduce M' as a node, draw an arc with label t from M to M', and tag M' "new."

We can also use Incidence matrix [2] to calculate reachable marking from a given marking after firing a particular transition.

Using both coverability tree and incidence matrix we can study some properties of Petri nets which are helpful in analyzing corresponding workflow from which the Petri net has been constructed. Three such useful properties are boundedness, safeness, and deadlock. If we start with an initial marking where there is only one token in the start place and no token in other places, then absence of boundedness indicates that a particular place have infinite number of tokens. So this indicates that we can never reach the end place without having left some tokens in other places. In workflow domains this implies that we can never end an activity without leaving some reference to it.

Safeness property in workflow domain will ensure that we don't have more than reference to an object to be processed. This makes sense since there is no need of processing two same objects when one is needed.

Deadlock property is very useful from workflow point of view as it indicates that the corresponding workflow has some activity which can not be reached hence the design has some flaws.

From the above discussion it is clear that from Petri net analysis we can often comment on the properties of a workflow. We have demonstrated this by allowing a user to design workflow using Activity diagrams and then converting the Activity diagram to a corresponding Petri net [4] for analysis. More details are given in the next section.

VIII. Implementation on Eclipse

We have implemented a plug-in for Eclipse [16] with which we can model workflows using Activity diagrams and then analyze the models. While performing analysis, the activity diagrams are first converted into their Petri net representations which are analyzed and results reported back into the workflow domain. We generate the Petri nets in PNML format [3] which is a standardized XML based format for representing Petri Nets. By representing the Petri Nets in PNML we provide means for future extensions using new analysis methods for Petri Nets.

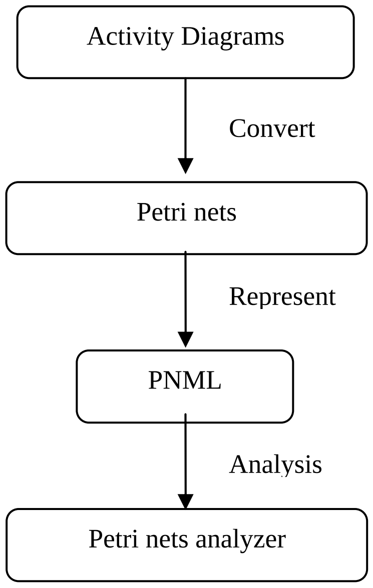


Fig. 3. Dataflow of our tool

Fig.3. gives the data flow of our tool. Fig.4. and Fig.5. are screenshots of our tool. Fig. 4 shows an Activity diagram drawn using the tool. The diagram is then converted into corresponding Petri net model which is then represented into PNML format. The analysis of the model is seen in Fig. 5.

IX. Related work

There is related work on workflows both in the Activity diagram domain as well as in the Petri net domain. Van Der Aalst et al has proposed Petri nets for both modeling and analyzing workflows in [11], [12], [13], [14]. Considering classical Petri nets are not powerful enough for modeling workflows, they have elevated it to high level Petri nets by adding time, color, and hierarchy [15]. The problem with this is that still Petri net is not an easy language for modeling workflows. Moreover, there are not many results available with high level Petri nets.

Activity diagram has been argued by many as an alternative for modeling workflows. After Van Der Aalst et al identified workflow patterns [9], it has been shown that they can be modeled using Activity diagrams [10]. There have been efforts for defining semantics for activity diagram, so that execution of the workflow models can be done ([6], [7], [8])

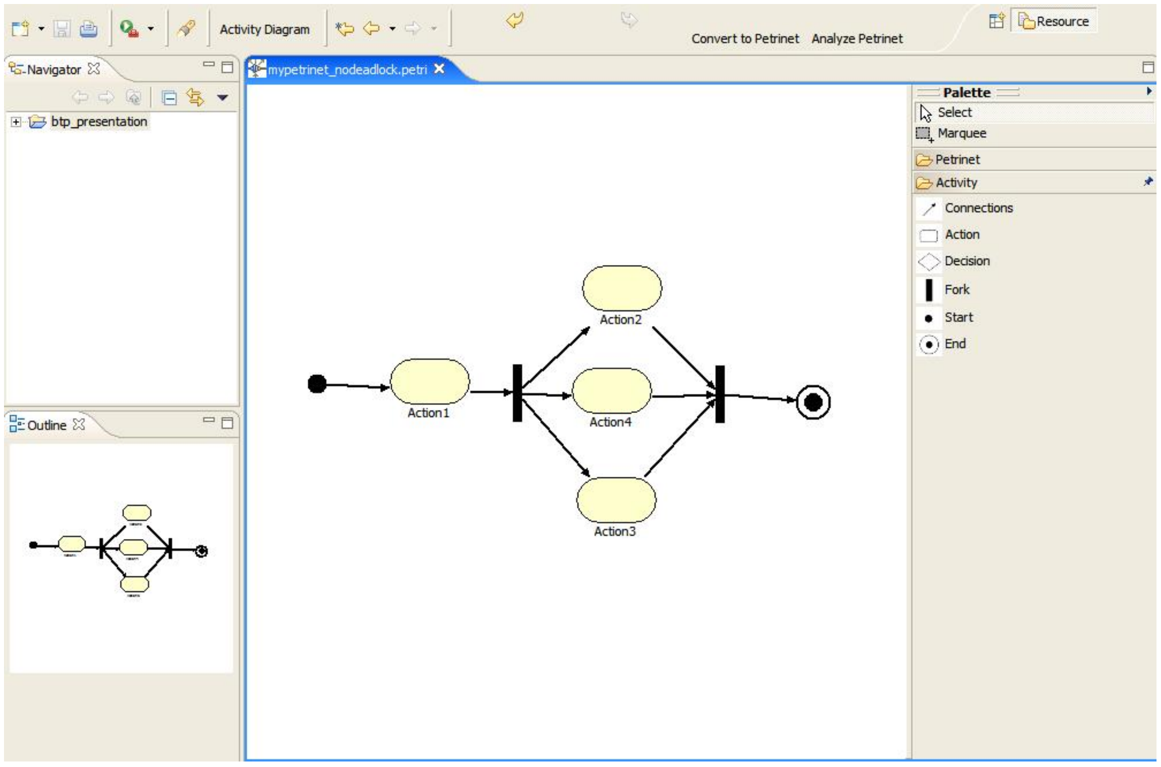
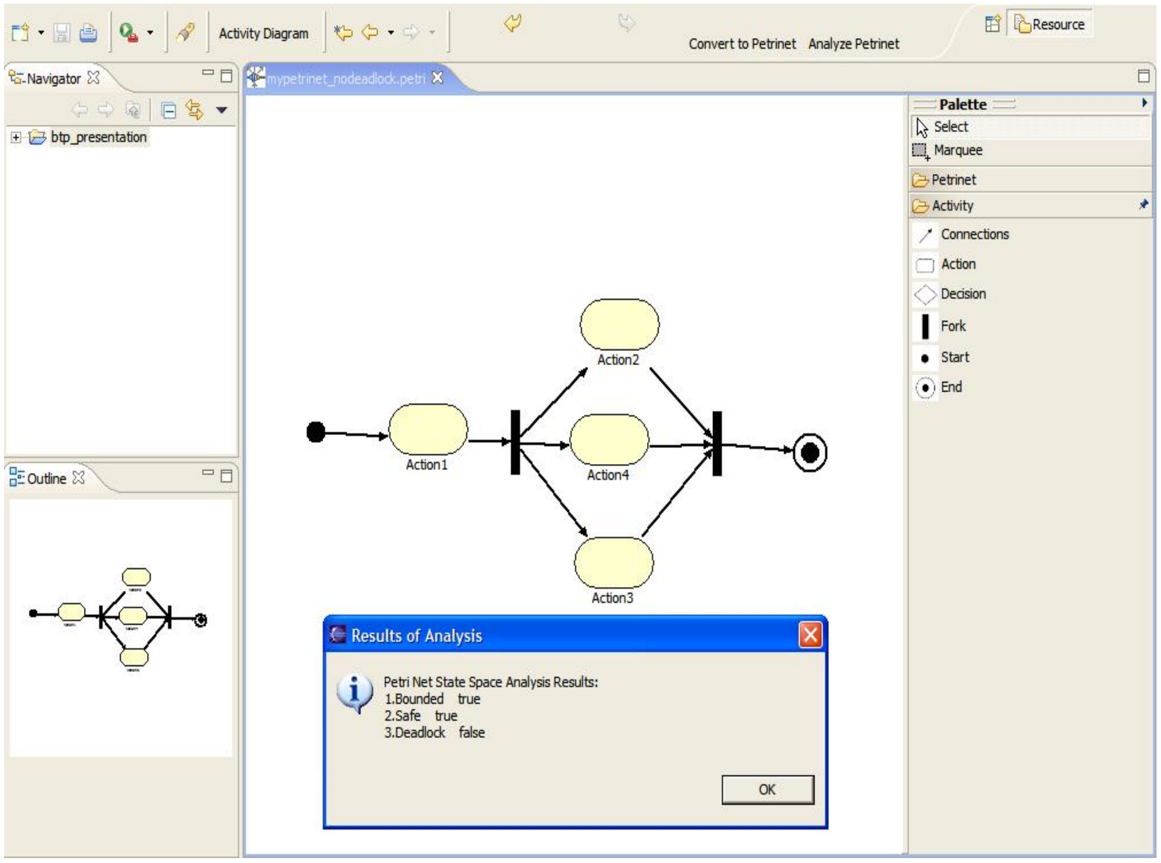


Fig. 4. Screen shot showing an Activity diagram



Petri Net State Space Analysis Results:
1.Bounded true
2.Safe true
3.Deadlock false

Fig. 5 Screen shot showing the results of analysis

X. Conclusion and Further work

We have proposed a new way of looking at analysis of workflows. Modeling of workflows should be done in a language which is easy and more intuitive to work with like Activity diagram. Analysis has to be done in a more formal language like Petri nets. Identifying from large set of results, that would be useful for analysis of workflows, needs to be done. We demonstrate this by giving a tool which can model workflows using Activity diagrams and then analyze the model using Petri nets. We have so far mentioned three properties of Petri nets which are useful in commenting on workflow models.

References

[1] W.M.P van der Aalst and Kees van Hee, *Workflow Management, Models, Methods, and Systems*

[2] Tadao Murata, *Petri Nets: Properties, Analysis and Applications,* Proceedings of the IEEE, Vol. 77, No. 4

[3] Billington et al., *The Petri Net Markup Language: Concepts,Technology, and Tools* [Online]. Available: http://www.informatik.huberlin.de/top/pnml/download/about/PNML_CTT.pdf

[4] Harald Storrle, Semantics of UML 2.0 Activities

[5] Workflow management coalition [Online]. http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf

[6] Rik Eshuis, Roel Wieringa. *A formal semantics for UML Activity Diagrams – Formalising workflow models,* Technical Report CTIT-01-04, U. Twente, Dept. Of Computer Science, 2001.

[7] Rik Eshuis, Roel Wieringa. *Verification support for workflow design with UML activity graphs,* In Proc.24th Intl. Conf. on Software Engineering (ICSE'02), pages 166-176. IEEE, 2002.

[8] Rik Eshuis, Roel Wieringa. *A real time execution semantics for UML activity diagrams,* In H. Hussmann, editor, Proc. 4th Intl. Conf. Fundamental approaches to software engineering (FASE'01), number 2029 in LNCS, pages 76-90. Springer Verlag, 2001.

[9] W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros. *Workflow Patterns*, BETA Working Paper Series, WP 47, Eindhoven University of Technology, Eindhoven, 2000

[10] Stephen A white, *Process Modelling Notations and Workflow patterns.* [Online] http://www.omg.org/bp-corner/bp-files/Process_Modeling_Notations.pdf

[11] W.M.P. van der Aalst. *The application of Petri nets to workflow management*, The Journal of Circuits, Systems and Computers, 8(1):21-66, 1998.

[12] W.M.P. van der Aalst. Woflan: *A Petri-net-based Workflow Analyzer*, Systems Analysis - Modelling - Simulation, 35(3):345-357, 1999.

[13] W.M.P. van der Aalst. *Workflow Verification: Finding Control-Flow Errors using Petri-net-based Techniques*, In Business Process Management: Models, Techniques, and Empirical Studies, *volume 1806 of* Lecture Notes in Computer Science, pages 161-183. Springer-Verlag, Berlin, 2000.

[14] W.M.P. van der Aalst and A.H.M. ter Hofstede. *Verification of Workflow Task Structures: A Petri-net-based Approach*, Information Systems, 25(1):43-69, 2000.

[15] W.M.P. van der Aalst, K.M. van Hee, G.J. Houben, *Petri nets and related formalisms*, Proceedings of the second Workshop on Computer-Supported Cooperative Work.

[16] Shavor et al., *The Java Developer's Guide to Eclipse*, 3rd ed, Addison-Wesley, 2003.

[17] [Online] http://www.eclipse.org