Woflan 2.0
A Petri-net-based Workflow Diagnosis Tool

H.M.W. Verbeek and W.M.P. van der Aalst

Department of Technology Management, Eindhoven University of Technology, the Netherlands
{H.M.W.Verbeek, W.M.P.v.d.Aalst}@tm.tue.nl

Abstract. Workflow management technology promises a flexible solution facilitating the easy creation of new business processes and modification of existing ones. Unfortunately, most of today's workflow products allow for erroneous processes to be put in production: these products lack proper verification mechanisms in their process-definition tools for the created or modified processes. This paper presents the workflow diagnosis tool Woflan, which fills this gap. Using Petri-net based techniques, Woflan diagnoses process definitions before they are put into production. These process definitions can be imported from commercial workflow products. Furthermore, Woflan guides the modeler of a workflow process definition towards finding and correcting possible errors.

1 Introduction

Today's workflow management systems are ill suited to dealing with frequent changes: there are hardly any checks to assure some minimal level of correctness on the process [Aal98,AH00]. Even a simple change like adding a task can cause serious problems like deadlock or livelock. As a result, an erroneous process definition may be taken into production as a workflow, causing dramatic problems for the organization. Therefore, it is important to verify the correctness of a process definition before it becomes operational. The role of verification becomes even more important as many enterprises are making Total Quality Management (TQM) one of their focal points. For example, an ISO 9000 certification and compliance forces companies to document business processes and to meet self-imposed quality goals [IC96]. Clearly, verification of these process definitions can be used to ensure certain levels of quality.

The tool Woflan was built in response to the need for a workflow verification tool. Right from the start, three requirements have been imposed on the tool:

1. Woflan should be independent of the process definition tool used by the modeler.
2. Woflan should be able to handle complex process definitions (up to hundreds of tasks).
3. Woflan should give the modeler to-the-point diagnostic information to find and repair errors.

Based on these requirements, we decided to use Petri nets because they are a universal modeling language with a solid mathematical foundation, are close to diagramming techniques used in practice, and have efficient analysis techniques that are already available. The primary goal of the Woflan tool is to verify a process definition, i.e., to check whether a process definition is a workflow process definition that satisfies the so-called soundness property [Aal97,Aal98].

Workflow process definition A process definition is called a *workflow* process definition if it has a single start condition (indicating the arrival of a new case), a single end condition (indicating the completion of a case) and if all tasks contribute to completing a newly-arrived case.

Soundness property A workflow process definition is called sound if it is always possible to complete a case (i.e., if it is always possible to reach the end condition), if completion is always proper (i.e., if no references to the case are left behind when it reaches the end condition), and if every task can be executed in some way.

In case the process definition is not a workflow process definition that satisfies the soundness property, Woflan's diagnostic information guides the developer towards finding and correcting the errors.

First, we explain the terminology used in this paper. Second, we describe the architecture of the tool. Third, we discuss the properties used by the tool to decide whether it is a sound workflow process definition. Fourth, we introduce the diagnosis process that helps the developer in finding and correcting the errors. Fifth, we discuss a number of import- and export filters from third party (WFMS, BPR) tools that increase Woflan's usefulness, using a diagnosis process definition as example. Sixth, we diagnose and correct the flawed diagnosis process definition. Last, we conclude with conclusions and future work.

2 Terminology

The terminology used in this paper is based on the terminology used by the WfMC [WFM96]. However, to avoid confusion within the Petri-net community, we use the term *condition* instead of *transition* to describe places. Table 1 shows the mapping from the workflow terms [WFM96] used in this paper to Petri-net related terms. For some non-standard terms a brief explanation is given.

Short-circuited net In [Aal97] it has been shown that a workflow net is sound if and only if that net extended with an extra transition (called `EXTENSION` in Woflan) from the sink place(s) to the source place(s) is bounded and live. This extended net is called the short-circuited net.

In the remainder of this paper, we want to avoid mentioning the short-circuited net over and over again. For this reason, some properties of a process definition are defined (see Table 1) on the short-circuited P/T net, while others are defined on the original P/T net.

Restricted coverability graph A restricted coverability graph (RCG) is a coverability graph (CG) except for the fact that infinite states are not expanded during construction of the RCG. Like a CG, an RCG is not uniquely defined if the net is unbounded. If no infinite states exist, an RCG equals the occurrence graph (OG) [VBA99].

Workflow	Petri net
Process definition	P/T net
Workflow process definition	Workflow (WF) net [Aal97,Aal98]
Condition	Place
Task	Transition
Start condition	Source place
End condition	Sink place
Useless task or condition	Strongly unconnected nodes in the short-circuited net
Thread of control	S-component in short-circuited net projected to places
Uniform invariant	P-invariant in short-circuited net containing only weights 0 and 1
Weighted invariant	P-invariant in the short-circuited net containing only semi-positive weights
Proper condition	Bounded place in minimal coverability graph (MCG, [Fin93]) of short-circuited net
Improper scenario	Unbounded sequence in restricted coverability graph (RCG, [VBA99]) of short-circuited net
Live task	Live transition in RCG of short-circuited net
Dead task	Dead transition in MCG of short-circuited net
Deadlock scenario	Non-live sequence in RCG of short-circuited net
Confusion	Non-free-choice cluster [DE95] in short-circuited net
AND-OR mismatch	TP-handle [EN94] in short-circuited net
OR-AND mismatch	PT-handle [EN94] in short-circuited net

Table 1. Mapping from workflow terms to Petri-net terms

Sequence A sequence is a firing sequence of minimal length (e.g., paths in the (R)CG) such that states with a given property become unavoidable (fairness etc. assumed). It is minimal in the sense that up to the last-but-one transition in the sequence (e.g., the last-but-one state in the path) the property is avoidable: the last transition in the sequence (e.g., the last edge in the RCG) makes the property unavoidable.

3 Architecture

The core of Woflan consists of Petri-net-based analysis routines. Using these routines, Woflan can verify the soundness of a given process definition. This soundness property is the minimal requirement any workflow process definition should satisfy. Because soundness is equivalent to the boundedness and liveness of the short-circuited WF net [Aal97], it can be verified using standard Petri-net techniques. Although it is possible to verify the soundness property for many process definitions in polynomial time, Woflan uses the general approach by constructing a minimal and/or restricted coverability graph. The diagnosis of the process definition is also partly based on these constructed CG's. The Woflan tool contains a number of modules:

1. One GUI module (`wofapp`),
2. One analysis module (`wofdll`) for loading, verifying and diagnosing process definitions, and

3. Three conversion modules (`scr2tpn`, `wil2tpn`, and `gwd2tpn`) for process definitions from commercial products (Cosa [SL98], Meteor [SKM], resp. Staffware [Sta97].

The GUI- and conversion modules are implemented in the main executable (called `wofapp.exe`). To support the use of Woflan as a back-end tool, the analysis module is implemented in a separate DLL (`wofdll.dll`).

4 Properties

Soundness of a workflow process definition is equivalent to that definition being proper and live, i.e., all conditions must be proper and all tasks must be live. Therefore, to decide soundness, Woflan computes whether all conditions are proper and all tasks are live. Preceding these two properties, Woflan has to decide whether or not the process definition is indeed a workflow process definition.

4.1 Workflow

The definition of a workflow process definition is straightforward: it should be a process definition with exactly one start condition, exactly one end condition, and no useless tasks or conditions. Because these properties are of a structural nature and do not require the construction of an MCG, RCG or OG, they are relative easy to check.

4.2 Properness

Properness of conditions can be decided using the conventional method, i.e., by generating the net's MCG etc. However, because of its complexity, we would like to avoid this if possible. Fortunately, there are alternatives that are less expensive from a computational point of view: all conditions covered by threads of control, uniform invariants or weighted invariants are proper. Because a thread of control is also a uniform invariant and a uniform invariant is also a weighted invariant, we have ordered these alternatives from more desirable to less desirable.

Threads of control From the workflow point of view, threads of control are very desirable. A workflow case typically consists of a number of documents. Each document has its own route through the workflow. A thread of control coincides with such a document route. So, if threads of control cover a workflow process definition, then each workflow case can be split into a number of documents such that each condition can be linked to some (possibly all) of these documents. If there is not such a cover, the uncovered conditions cannot be linked to any document. As a result, for a workflow process definition to be sound, both confusions and mismatches have to be present [VBA99]. Apparently, these constructions are vital to 'cure' the net from these uncovered conditions. This soundness-related property is called *interim soundness*: a process definition containing uncovered conditions is called interim sound if and only if it contains confusions and mismatches.

Diagnostic properties If a definition contains improper conditions, Woflan computes some additional properties that can help finding and correcting the properness problem: AND-OR mismatches (they endanger properness), confusions, and improper scenarios.

4.3 Liveness

Suppose we have a process definition that can be covered by invariants (i.e., that contains no improper conditions), that can not be covered by threads of control, and that has either no confusions or no mismatches. For such a definition we can conclude that it is unsound, i.e., it contains non-live tasks.

Likewise, suppose we have a process definition containing no improper conditions and for which we have detected substates during the construction of the MCG. A reachable markings M_1 is a substate of another reachable marking M_2 iff $M_1 < M_2$. At this point, we can conclude that from M_1 the extra task `EXTENSION` is dead.

Otherwise, Woflan has no method yet to decide liveness without generating the OG. Note that generating this OG is only possible if the process definition contains no improper conditions.

Diagnostic properties If a definition contains non-live tasks, Woflan computes some additional properties that can help finding and correcting the liveness problem: OR-AND mismatches (they endanger liveness), confusions, dead tasks, and deadlock scenarios.

5 Diagnosis

Based on practical experiences with earlier versions of Woflan we have developed a method for detecting errors in a workflow process. This method is supported by Woflan 2.0 and uses the diagnosis process shown in Figure 1. The diagnosis process can either be executed in-succession or step-by-step. In the latter case, dialogs are used to communicate with the user. First, we give the diagnosis process. Second, we explain one of the dialogs in detail, using the diagnosis process as shown before as example. Last, we explain the general view in detail, using again the diagnosis process itself. Please note that Figure 1 is used both as a meta-model describing the functionality of Woflan *and* as a concrete example of a workflow process.

5.1 Process

Because of the fact that we need the OG (and therefore a workflow process definition containing no improper conditions) for deciding liveness, it is obvious to decide it only if the workflow process definition has been proven to contain only proper conditions. Because the workflow properties are easy to check, Woflan starts with them. As a result, the main steps in the diagnostic process are:

1. Decide workflow,
2. Decide properness, and

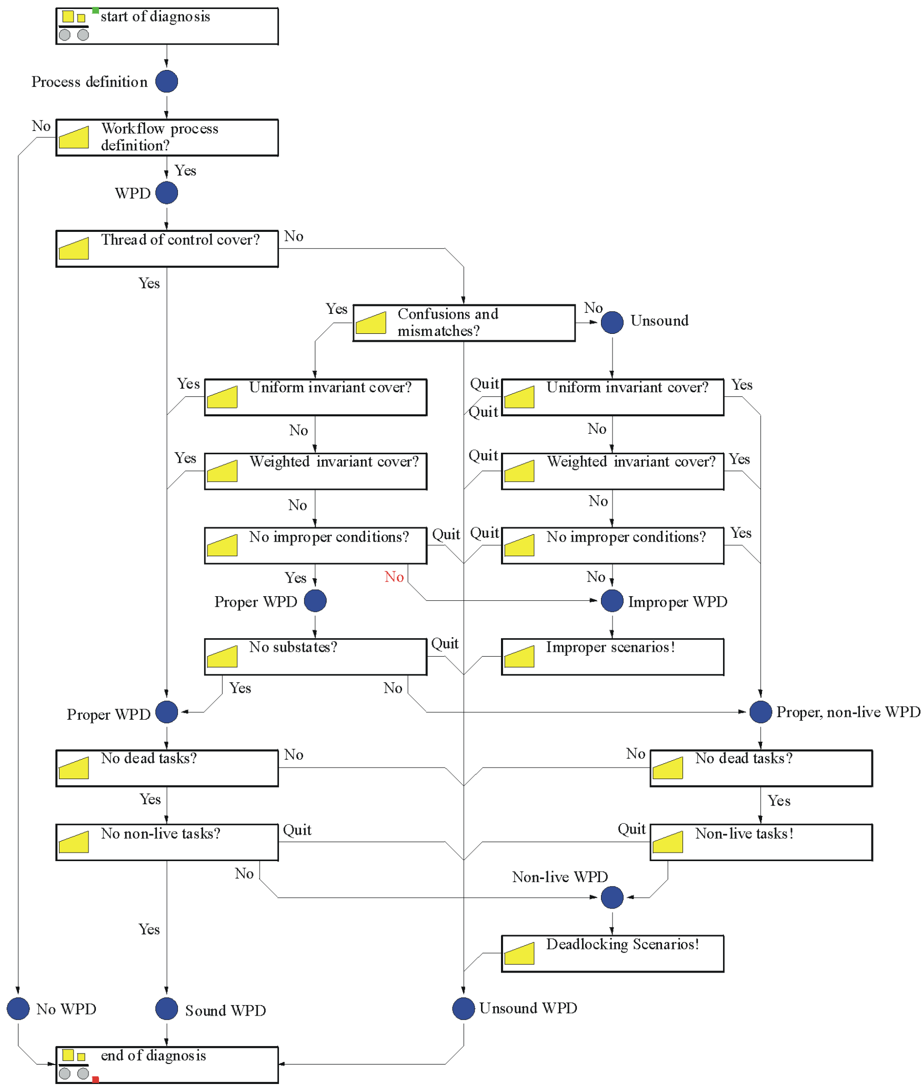

Fig. 1. Diagnosis process, modeled using Protos [Pal97]

3. Decide liveness.

If some step fails, there is not much use in continuing with the next step. The modeler of the process definition first has to correct the errors present.

Figure 1 shows a graphical representation of the diagnosis process definition. Note that in some cases the diagnosis process may be continued when unsoundness of the process definition has been detected. The reason for this is to collect more diagnostic information.

5.2 Dialogs

The diagnosis process uses a series of dialogs to guide the user step-by-step through the process. Depending on the diagnostic results, either a next dialog is presented or the process is finished (`end of diagnosis`). Properties that are likely to be of interest to the modeler are automatically unfolded. As running example, we take the diagnosis process definition as shown in Figure 1 and show the dialog concerning the thread of control cover.

Thread of control cover? The dialog as shown in Figure 2 shows that, using previous dialogs, Woflan has concluded that the diagnosis process definition is a workflow process definition, but that no threads of control exist. As a result, the 20 conditions of the diagnosis process definition are listed.

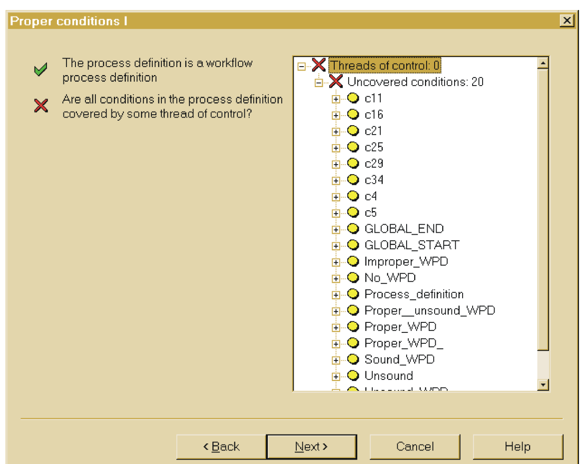

Fig. 2. Example "Thread of control cover?" dialog

5.3 Diagnosis view

The diagnosis view shows all properties of the process definition in a tree-like manner. At the root, the name of the process definition file is shown. This root node has two child nodes: the upper for the diagnosis results, the lower for the diagnostic properties. The diagnosis results node shows in brief the results on the main properties (workflow, safeness, liveness, soundness). The diagnostic properties node combines the diagnostic information from all dialogs.

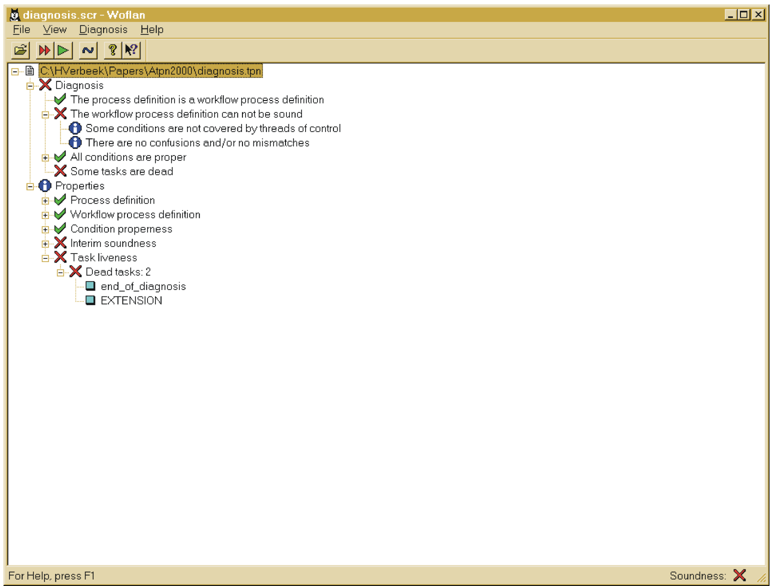

Fig. 3. Example diagnosis view

6 Links to third-party software

Woflan embeds three filters to import third party process definition files:

1. For Cosa [SL98] script files (`*.scr`),
2. For Meteor [SKM] workflow files (`*.wil`), and
3. For Staffware [Sta97] (`*.xfr`) files.

Furthermore, the BPR tool Protos [Pal97] comes with an additional Woflan export filter, which uses Cosa script files as an intermediate format. Each embedded filter shows the results (which could be error messages) of the import process in a dialog. For readability's sake the comments are colored gray, the keywords green, and error messages red.

The dialog as shown in Figure 4 results from the diagnosis process definition (see Figure 1) that was designed using Protos, exported to Woflan (i.e., to a Cosa script file) and imported by Woflan's Cosa import filter. Note that the Cosa import filter automatically added a start condition (GLOBAL_START), an end condition (GLOBAL_END), and a token in the start condition representing a newly-arrived case.

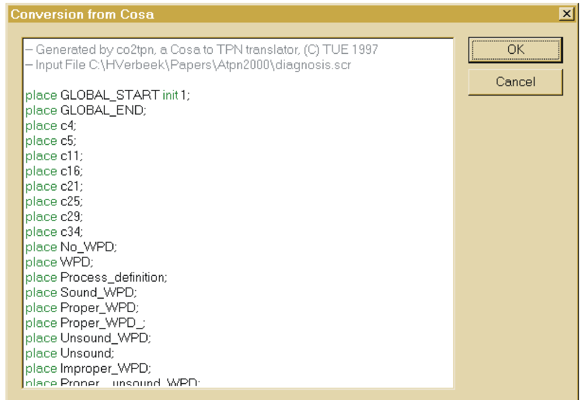

Fig. 4. Example import filter dialog

7 Example

Apparently, the diagnosis process definition from Figure 1 is unsound. In this case, particularly the dead tasks are of interest. Note that the short-circuiting task `EXTENSION` is dead. As a result, all tasks are non-live. The task `EXTENSION` can only be dead if task `end_of_diagnosis` is dead, which is dead because it acts as an AND-join instead of an OR-join. Protos' export filter and Woflan's import filter allow to have this property changed in Protos, both filters can handle a task which acts as an OR-join (split) instead of as an AND-join (split). After changing this property in Protos, exporting it to Woflan and importing the resulting Cosa script file, the diagnosis process appears to be a sound workflow process definition. Although this error seems trivial, taking a workflow with such a flawed process definition into production will result in much irritation and agony: prevention is better than cure. Also note that real world examples are not as straightforward as the workflow process definition shown in Figure 1.

8 Conclusions and future work

For several commercial WFMS/BPR products, Woflan can be used to verify a process definition, checking both syntactic (cf. Section 4.1) and behavioral (cf. Sections 4.2 and 4.3) properties. By using Woflan and its state-of-the-art techniques it is possible to prevent that an unsound workflow is taken into production.

In the nearby future we hope to extend Woflan with two more features: transition invariants and visualization.

A *sound* workflow process definition is covered by non-negative transition invariants. If the process definition is safe, a task that is not covered by these invariants cannot be live. In a next version of Woflan we hope to use this property to avoid the use of the OG, if possible.

We also would like to visualize the diagnosis results in some intuitive way, using Petri nets of course. If possible, we even want to visualize these results in the

WFMS/BPR tool the modeler is using. To support this use of Woflan as a back-end tool, we separated the analysis techniques from the rest of the tool.

Woflan can be downloaded from [VA].

9 Acknowledgements

The authors wish to thank Twan Basten and Hajo Reijers for their fruitful remarks and suggestions.

References

Aal97. W.M.P. van der Aalst. Verification of Workflow Nets. In P. Azéma and G. Balbo, editors, *Application and Theory of Petri Nets 1997, Proceedings*, volume 1248 of *Lecture Notes in Computer Science*, pages 407–426, Toulouse, France, June 1997. Springer, Berlin, Germany, 1997.

Aal98. 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.

AH00. W.M.P. van der Aalst and A.H.M. ter Hofstede. Verification of Workflow Task Structures: A Petri-net-based Approach. *Information Systems*, To appear, 2000.

DE95. J. Desel and J. Esparza. *Free Choice Petri Nets*, volume 40 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, UK, 1995.

EN94. J. Esparza and M. Nielsen. Decidability Issues for Petri Nets - a Survey. *Journal of Information Processing and Cybernetics*, 30(3):210–242, 1994.

Fin93. A. Finkel. The Minimal Coverability Graph for Petri Nets. In G. Rozenberg, editor, *Advances in Petri Nets 1993*, volume 674 of *Lecture Notes in Computer Science*, pages 210–243. Springer, Berlin, Germany, 1993.

IC96. R.R.A. Issa and R.F. Cox. Using Process Modeling and Workflow Integration to gain (ISO 9000) Certification in Construction. In *CIB W89 Beijing International Conference on Construction, Modernization, and Education*, Beijing, China, 1996.

Pal97. Pallas Athena. *Protos User Manual*. Pallas Athena BV, Plasmolen, The Netherlands, 1997.

SKM. A. Sheth, K. Kochut, and J. Miller. Large Scale Distributed Information Systems (LSDIS) laboratory, METEOR project page. http://lsdis.cs.uga.edu/proj/meteor/meteor.html.

SL98. Software-Ley. *COSA User Manual*. Software-Ley GmbH, Pullheim, Germany, 1998.

Sta97. Staffware. *Staffware 97 / GWD User Manual*. Staffware Plc, Berkshire, UK, 1997.

VA. H.M.W. Verbeek and W.M.P. van der Aalst. Woflan Home Page. http://www.win.tue.nl/˜woflan.

VBA99. H.M.W. Verbeek, T. Basten, and W.M.P. van der Aalst. Diagnosing Workflow Processes using Woflan. Computing Science Report 99/02, Eindhoven University of Technology, Eindhoven, The Netherlands, 1999.

WFM96. WFMC. Workflow Management Coalition Terminology and Glossary (WFMC-TC-1011). Technical report, Workflow Management Coalition, Brussels, 1996.