A Petri-net-based Tool to Analyze Workflows

W.M.P. van der Aalst   H.M.W. Verbeek   D. Hauschildt[1]

Eindhoven University of Technology
Department of Mathematics and Computing Science
P.O. Box 513, NL - 5600 MB Eindhoven
{*wsinwa,wsineric,dirk*}*@win.tue.nl*

Abstract
Workflow management systems facilitate the everyday operation of business processes by taking care of the logistic control of work. In contrast to traditional information systems, they attempt to support frequent changes of the workflows at hand. Therefore, the need for analysis methods to verify the correctness of workflows is becoming more prominent. In this paper we present a tool based on Petri nets: *Woflan*. Woflan (WOrkFLow ANalyzer) is an analysis tool which can be used to verify the correctness of a workflow procedure. The analysis tool uses state-of-the-art techniques to find potential errors in the definition of a workflow procedure.

Keywords: workflow management, Petri nets, workflow analysis, verification.

1 Introduction

Workflow management systems (WFMS) are used for the modeling, analysis, enactment, and coordination of structured business processes by groups of people. Business processes supported by a WFMS are *case-driven*, i.e., tasks are executed for specific cases. Approving loans, processing insurance claims, billing, processing tax declarations, handling traffic violations and mortgaging are typical case-driven processes which are often supported by a WFMS. These case-driven processes, also called *workflows*, are marked by three dimensions: (1) the process dimension, (2) the resource dimension, and (3) the case dimension (see Figure 1). The process dimension is concerned with the partial ordering of tasks. The tasks which need to be executed are identified and the routing of cases along these tasks is determined. Conditional, sequential, parallel and iterative routing are typical structures specified in the process dimension. Tasks are executed by resources. Resources are human (e.g. employee) and/or non-human (e.g. device, software, hardware). In the resource dimension these resources are classified by identifying roles (resource classes based on functional characteristics) and organizational units (groups, teams or departments). Both the process dimension and the resource dimension are generic, i.e., they are not tailored towards a specific case. The third dimension of a workflow is concerned with individual cases which are executed according to the process definition (first dimension) by the proper resources (second dimension).

Managing workflows is not a new idea. Workflow control techniques have existed for decades and many management concepts originating from production and logistics are also applicable in a workflow context. However, just recently, commercially available generic WFMS's have become a reality.

[1] On leave from University of Hamburg, Department of Computer Science, supported by the MATCH project, EC, Human Capital and Mobility, CHRX-CT94-0452.

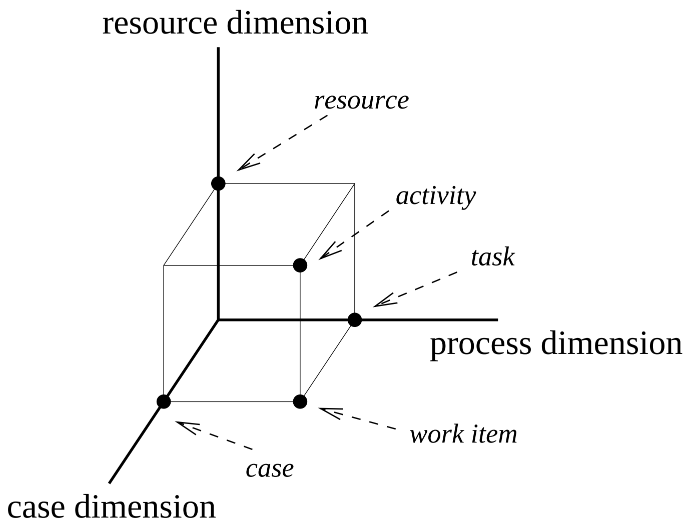


Figure 1: The three dimensions of workflow.

Although these systems have been applied successfully, contemporary WFMS's do not facilitate advanced analysis methods to determine the correctness of a workflow. Some WFMS's support simulation to validate a workflow definition before it becomes operational. Clearly, there is a need for tools to *verify* the correctness of a workflow definition. This is the reason we developed *Woflan*.

Woflan focuses on the process dimension (see Figure 1) and assumes a "fair" behavior of the environment (triggering and resources). Woflan also assumes that a workflow procedure (i.e. the process) is specified in terms of a Petri net. As many researchers have indicated ([EN93, MEM94, WR96]), Petri nets constitute a good starting point for a solid theoretical foundation for workflow management. Unfortunately, most of the more than 250 commercially available WFMS's use a vendor-specific ad-hoc modeling technique to design workflows. In spite of the efforts of the Workflow Management Coalition ([WFM96]) real standards are missing. The absence of formalized standards hinders the development of tool-independent analysis techniques. Fortunately, most WFMS's use a technique which can be translated into a Petri net, because they either use a subset or use high-level constructs which can be mapped onto Petri nets.

Woflan is designed as a WFMS-independent analysis tool. In principle it can interface with many WFMS's. At the moment, Woflan can interface with the WFMS COSA (Software Ley [SL96]) and the BPR-tool Protos (Pallas Athena [Pal97]). In the future we hope to extend the set of WFMS's which can interface with Woflan.

This paper describes the functionality, architecture and implementation of Woflan. The reader is assumed to have some basic knowledge of workflow management and more extensive knowledge of Petri nets. For more detailed information the reader is referred to [WFM96, EN93, Mur89, Aal96a, Aal96c, Aal95, DE95, Aal97b, Aal96b, AH97]. This paper is organized as follows. First, the architecture of Woflan is presented. Then a description of the internal data structure and the analysis routines are given. Finally, the user interface is illustrated by a number of screendumps.

2 Architecture of Woflan

As indicated in the introduction, Woflan (WOrkFLow ANalyzer) is a Petri-net-based tool to analyze the correctness of a workflow. The tool is workflow management system independent, i.e. a number of import functions to download workflow-scripts in Woflan (e.g. from COSA and Protos) are provided. Woflan uses standard Petri-net-based analysis techniques. However, the analysis results will be presented in such a manner that end-users will understand the output of Woflan. Moreover, Woflan guides the end-user in correcting an erroneous workflow. Figure 2 shows the architecture of Woflan.

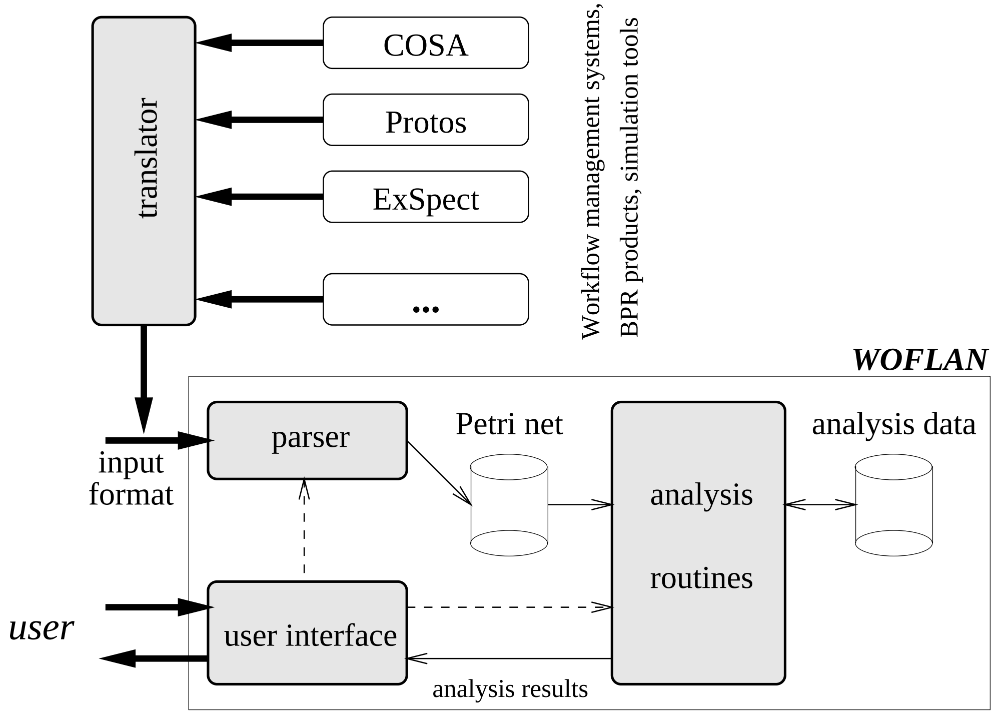


Figure 2: Architecture of Woflan.

Woflan consists of three main parts:

- *parser*
  To load a file representing a workflow process definition. The input format of Woflan is defined in [HVA97].

- *analysis routines*
  A large collection of standard Petri-net analysis routines.

- *user interface*
  To present the results to the end-user.

In addition there will be a module for each WFMS which can interface with Woflan. At the moment there is one such module which can convert COSA (Software Ley [SL96]) script files into the format used by Woflan. This way Woflan can analyze any workflow process definition constructed by using the COSA network editor (CONE). The same module can be used to import process definitions made with Protos (Pallas Athena [Pal97]). It is also possible to import process models made with ExSpect ([Bak96]). ExSpect is a general purpose simulation tool based on high-level Petri nets.

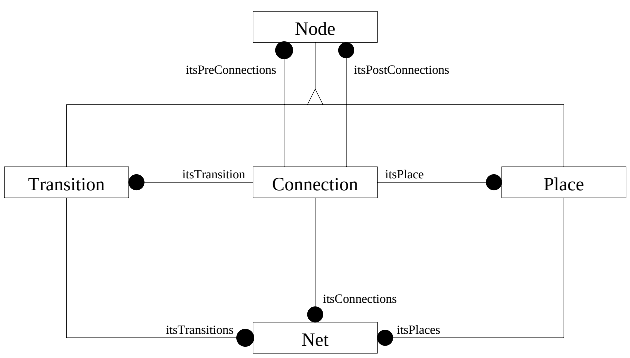
In the remainder of this paper, the internal data structure, the analysis routines and the user interface are described.

3 Data structure

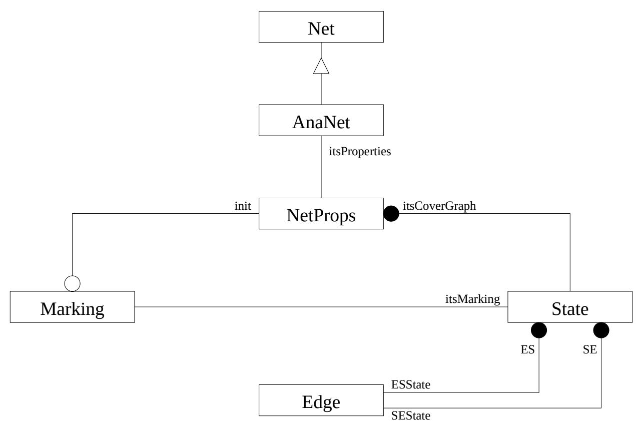
The data structure needed by Woflan can be divided into three parts:

1. Classes needed to hold one or more Petri nets. This structure is build up by Woflan's parser and used by its analysis routines.
2. Classes needed to hold necessary analysis data. At the moment this structure mainly holds the net's coverability graph.
3. Classes needed to hold the analysis results. When for instance the place invariants are requested by the user interface, the user interface and the analysis routine concerned have to agree on the way how to hold the place invariants.

The object model for a Petri net is rather straightforward: a Petri net consists of nodes (transitions and places) and connections (which connect one place with one transition and are directed). Figure 3(a) shows the object model for Petri nets.



(a)



(b)

Figure 3: Two object models.

For the analysis routines the class AnaNet is introduced, which is a derived class of Net. The class AnaNet contains all the analysis routines and the necessary analysis data like the coverability graph. Figure 3(b) shows the object model for analysis data.

Many analysis routines produce an answer which is built up from the Petri net's nodes, sets and pairs. The routine which returns the place invariants for example, has to produce an answer which is a set of set of weighted places. The template class `Array` is introduced to hold any set, for instance the class `P_Array` is defined as `Array<Place>`. Figure 4 shows the object model for Woflan's analysis results.

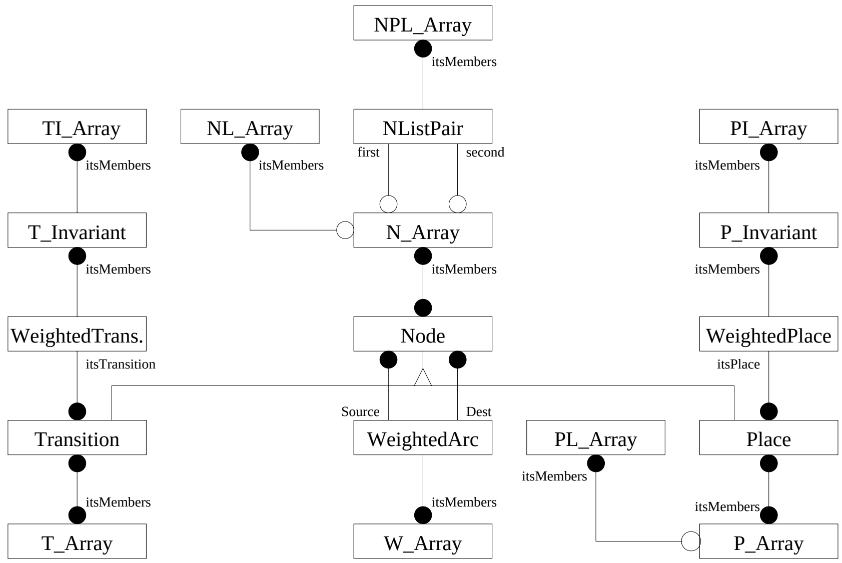


Figure 4: Object model for Woflan's analysis results.

4 Analysis routines

Woflan uses Petri-net-based analysis routines to analyze the workflows at hand. One of the central issues which is analyzed by Woflan is the so-called *soundness property*. A workflow process is sound if, for any case, the process terminates properly, i.e., termination is guaranteed, there are no dangling references, and deadlock and livelock are absent. A formal definition of the soundness property is given in [Aal97b, Aal96b]. Clearly, this property is the basic property any workflow should satisfy. In [Aal97b] it is shown that standard Petri-net-based analysis techniques to decide liveness and boundedness can be used to verify the soundness property. For many workflows this can be decided in polynomial time. However, Woflan uses a brute force approach by constructing the coverability graph to decide soundness. This turns out to be satisfactory from a practical point of view: even complex workflows have less than 100 tasks (< 200.000 states), this is no problem for Woflan.

Deciding whether the workflow definition is sound is not sufficient. In many cases more requirements need to be satisfied. Moreover, if the workflow definition is not sound, then the user should be guided in detecting the source of the error and support should be given to repair the error. This is the reason Woflan offers a large selection of analysis methods:

- Syntactical checks, e.g., detection of tasks without input or output condition.
- Detection of potential errors by listing suspicious constructs, e.g., constructs violating the free-choice property, AND-split's complemented by OR-join's, OR-split's complemented by AND-join's (well-structuredness), and parts of the net which are not S-coverable.
- Detection of dynamic errors by listing unbounded places, non-safe places, dead transitions and non-live transitions.
- Place and transition invariants. The absence of certain invariants indicates the source of an error.
- Verification of the soundness property.

Woflan has an on-line help-facility which guides the user in using the tool and helps to understand the analysis results.
A detailed description of the analysis routines is outside the scope of this paper. A complete overview of the analysis routines is given in [HVA97]. The next section presents the user interface of Woflan and illustrates the analysis methods which can be used to detect and repair errors in a workflow design.

5 User interface

The user interface is built using XVT, a GUI builder which supports, among others, both MS Windows and Solaris.
For each workflow definition loaded by Woflan, a window is opened containing a button bar (on the lefthandside) and an area where the analysis results are displayed. Using the example shown in Figure 5, we will browse through some of the displays.

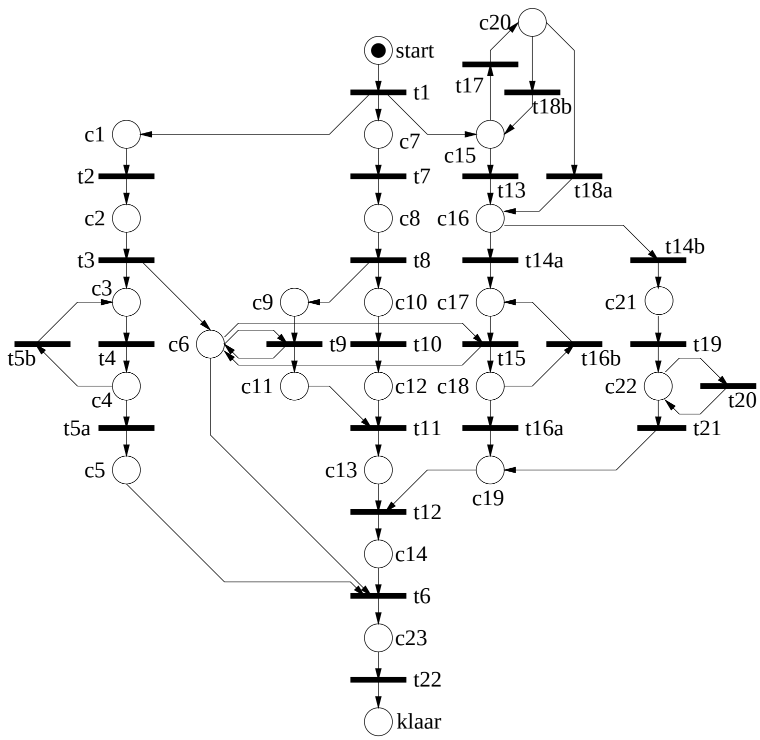


Figure 5: An example of a workflow specified in terms of a Petri net.

The default display is the summary display (see Figure 6). This display shows some basic characteristics of the net, like whether it can be covered by state machines (S-coverable). The coverability graph is not automatically generated, one needs to press a button to do so. Figure 6 shows the summary display after the graph has been generated so it also shows whether the net is bounded, etc.
The workflow display (Figure 7) shows whether the net is a workflow net (one source place, one sink place, no source nor sink transitions and no (strongly) unconnected nodes) and one can also see which nodes are not connected to any source or sink place. The example net is obviously a workflow net.
The free choice display (Figure 8) shows why the example net is not a free choice net: the cluster `c5 c6 c9 c14 c17 t6 t9 t15` is not free choice. When more than one non-free-choice cluster is found, one can browse through the clusters using the navigation buttons on the lefthandside.

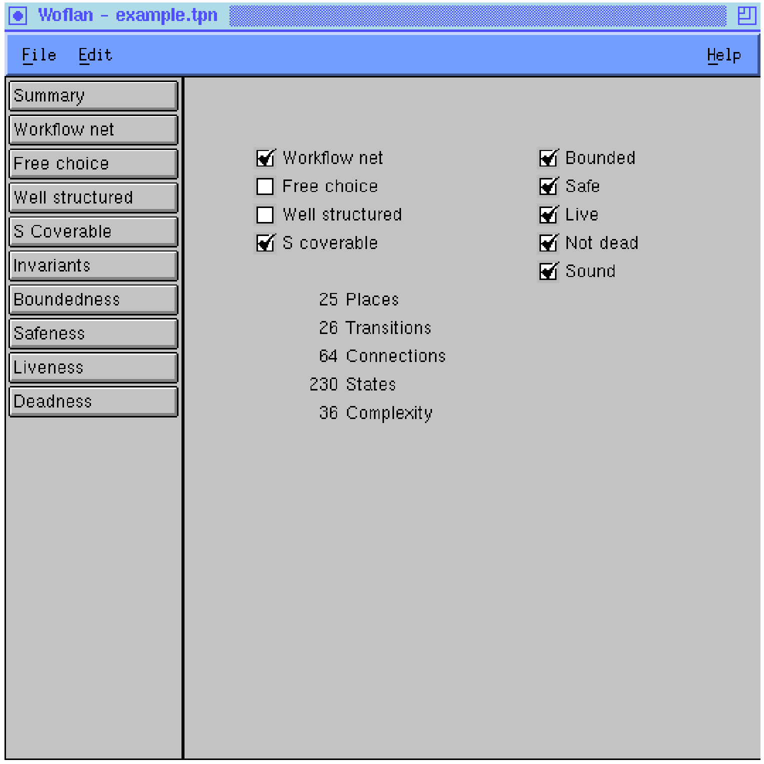


Figure 6: Screendump of Woflan's summary display.

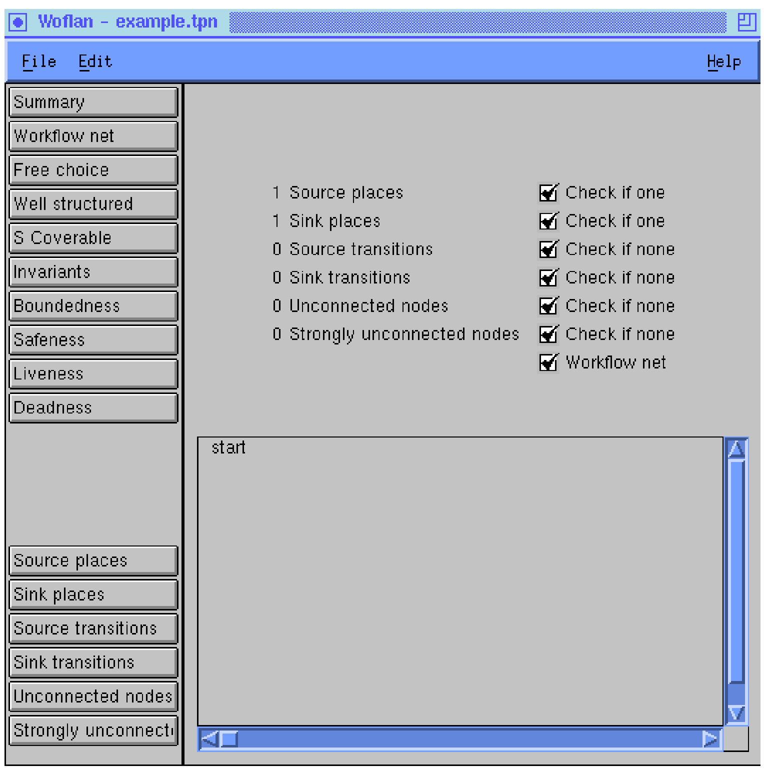


Figure 7: Screendump of Woflan's workflow display.

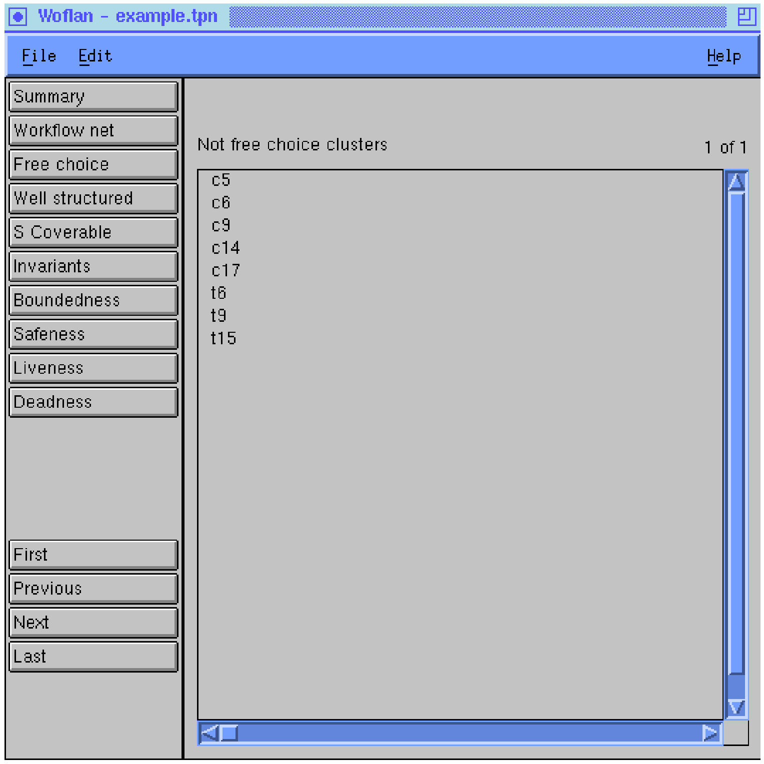


Figure 8: Screendump of Woflan's free choice display.

The example net is also not well structured, as is shown by the structure display (Figure 9). Apparently there are nine not-well-handled pairs (a place-transition pair with multiple paths strongly connecting the second to the first).
The example net however *can* be covered by state machines (five), as is shown by the S-cover display (Figure 10). Nodes which are no part of any S-cover are also listed.
The invariant display is shown in Figure 11. Various types of invariants can be reported: place/transition, semi-positive/base/extended. In Figure 11 the semi-positive place invariants found for the example workflow (Figure 5) are shown.

6 How to locate errors in the design of a workflow

The workflow shown in Figure 5 is a correct workflow. The only construct which endangers the soundness property is the place `c6`. This construct is not free choice and also results in a number of not-well-handled pairs (i.e. the workflow is not well-structured). Woflan warns for these constructs (see Figures 8 and 9).

To illustrate the use of Woflan, we briefly indicate the diagnostics that are given in case of a typical mistake.
Suppose the connection between `c19` and `t12` is forgotten. Clearly, the net is not sound. The summary display will indicate that the net is not a workflow net and the workflow display will indicate that there are multiple sink places (including `c19`). Therefore, it is easy to trace the error.
Suppose the places `c5` and `c14` are fused together. The resulting net is not sound. Woflan will report

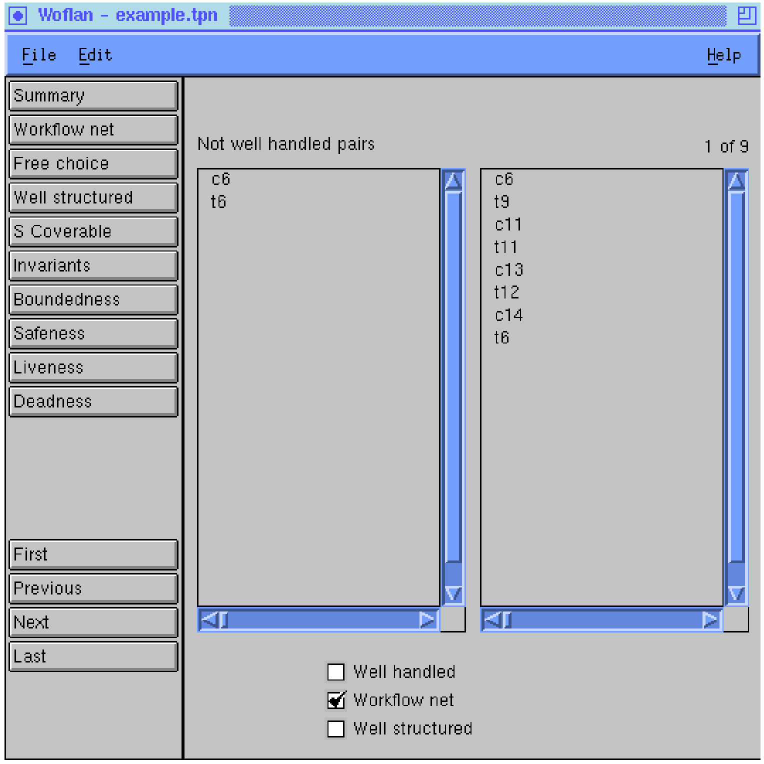


Figure 9: Screendump of Woflan's structure display.

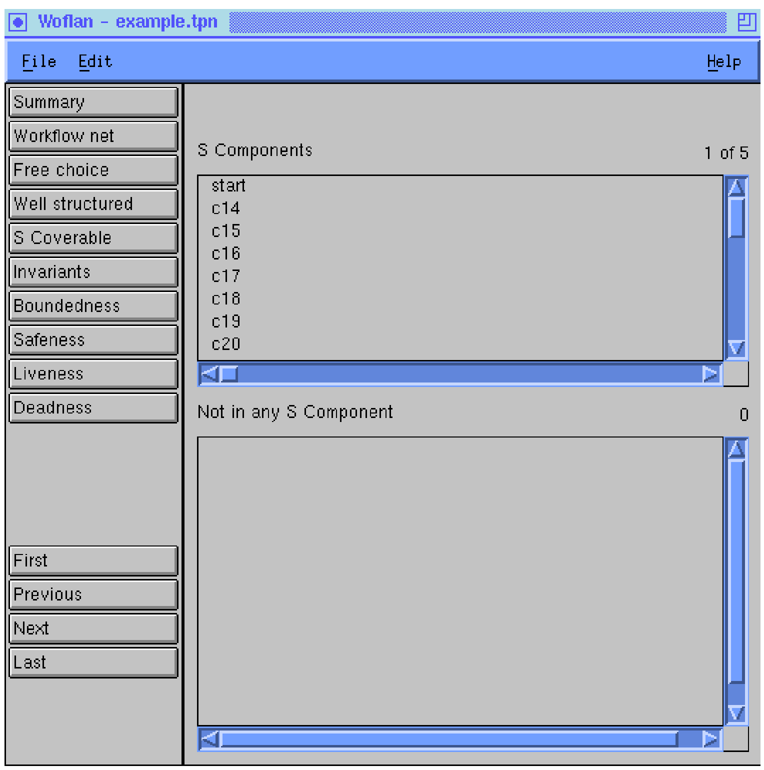


Figure 10: Screendump of Woflan's S-cover display.

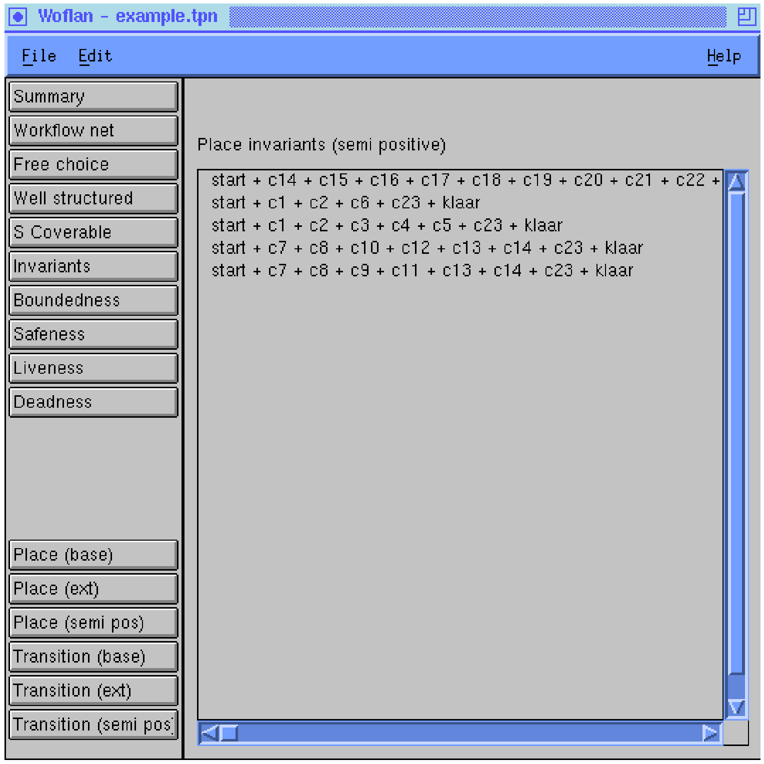


Figure 11: Screendump of Woflan's invariants display.

additional not-well-handled pairs which indicate that the AND-split `t1` is complemented by an OR-join (the new place which corresponds to the two fused places).

Suppose there is an additional place `c24` which connects `t16a` and `t2`, i.e., `t16a` has to occur before `t2`. The workflow net is not sound because of a potential deadlock. Woflan will indicate that the net is not live and lists the transitions which are dead.

Suppose the arrows connecting `t3`, `c6` and `t6` are reversed. Woflan will detect the deadlock and lists the dead transitions. Moreover, Woflan will report the place invariant `c6 + c3 + c4 + c5`. Every semi-positive place invariant which does not include `start` and `klaar` ('ready') corresponds to an error which results in a deadlock.

Suppose that `t5b` puts tokens in `c5`. Woflan will report that the net is not sound. Several diagnostics are given to detect the source of the error. First of all, Woflan reports that `c5` is unbounded. Secondly, the fact that the AND-split `t5b` is complemented by the OR-join `c5` is reported. Finally, `c6` is not covered by any S-component.

These five examples illustrate that Woflan guides the user in finding and correcting errors in the design of a workflow. For a relatively simple workflows, such as the one shown in Figure 5, these results may seem rather straightforward. However, for the more complex workflows encountered in practise, it is far from trivial to detect and repair errors.

7 Conclusion

The tool presented in this paper gives workflow designers a handle to construct correct workflows. Woflan is both from a theoretical and a practical point of view an interesting tool. On the one hand, Woflan uses advanced analysis techniques. On the other hand, it interfaces with some of leading workflow tools on the Dutch market (COSA/Protos). Woflan clearly shows that the workflow market is a challenging application domain for Petri-net-based technology.

References

[Aal95] W.M.P. van der Aalst. A class of Petri net for modeling and analyzing business processes. Computing Science Reports 95/26, Eindhoven University of Technology, Eindhoven, 1995.

[Aal96a] W.M.P. van der Aalst. Petri-net-based Workflow Management Software. In A. Sheth, editor, *Proceedings of the NFS Workshop on Workflow and Process Automation in Information Systems*, pages 114–118, Athens, Georgia, May 1996.

[Aal96b] W.M.P. van der Aalst. Structural Characterizations of Sound Workflow Nets. Computing Science Reports 96/23, Eindhoven University of Technology, Eindhoven, 1996.

[Aal96c] W.M.P. van der Aalst. Three Good reasons for Using a Petri-net-based Workflow Management System. In S. Navathe and T. Wakayama, editors, *Proceedings of the International Working Conference on Information and Process Integration in Enterprises (IPIC'96)*, pages 179–201, Camebridge, Massachusetts, Nov 1996.

[Aal97a] W.M.P. van der Aalst. Exploring the Process Dimension of Workflow Management. Computing Science Reports 97/13, Eindhoven University of Technology, Eindhoven, 1997.

[Aal97b] W.M.P. van der Aalst. Verification of Workflow Nets. In P. Azema and G. Balbo, editors, *Application and Theory of Petri Nets 1997*, volume 1248 of *Lecture Notes in Computer Science*, pages 407–426. Springer-Verlag, Berlin, 1997.

[AH97] W.M.P. van der Aalst and K.M. van Hee. *Workflow Management: Modellen, Methoden en Systemen (in Dutch)*. Academic Service, Schoonhoven, 1997.

[Bak96] Bakkenist Management Consultants. *ExSpect 5.0 User Manual*, 1996.

[BCD95] K. Barkaoui, J.M. Couvreur, and C. Dutheillet. On liveness in Extended Non Self-Controlling Nets. In G. De Michelis and M. Diaz, editors, *Application and Theory of Petri Nets 1995*, volume 935 of *Lecture Notes in Computer Science*, pages 25–44. Springer-Verlag, Berlin, 1995.

[DE95] J. Desel and J. Esparza. *Free choice Petri nets*, volume 40 of *Cambridge tracts in theoretical computer science*. Cambridge University Press, Cambridge, 1995.

[EKR95] C. Ellis, K. Keddara, and G. Rozenberg. Dynamic change within workflow systems. In N. Comstock and C. Ellis, editors, *Conf. on Organizational Computing Systems*, pages 10 – 21. ACM SIGOIS, ACM, Aug 1995. Milpitas, CA.

[EN93] C.A. Ellis and G.J. Nutt. Modelling and Enactment of Workflow Systems. In M. Ajmone Marsan, editor, *Application and Theory of Petri Nets 1993*, volume 691 of *Lecture Notes in Computer Science*, pages 1–16. Springer-Verlag, Berlin, 1993.

[ES90] J. Esparza and M. Silva. Circuits, Handles, Bridges and Nets. In G. Rozenberg, editor, *Advances in Petri Nets 1990*, volume 483 of *Lecture Notes in Computer Science*, pages 210–242. Springer-Verlag, Berlin, 1990.

[HVA97] D. Hauschildt, H.M.W. Verbeek, and W.M.P. van der Aalst. WOFLAN: a Petri-net-based Workflow Analyzer. Computing Science Reports 97/12, Eindhoven University of Technology, Eindhoven, 1997.

[Law97] P. Lawrence, editor. *Workflow Handbook 1997, Workflow Management Coalition*. John Wiley and Sons, New York, 1997.

[MEM94] G. De Michelis, C. Ellis, and G. Memmi, editors. *Proceedings of the second Workshop on Computer-Supported Cooperative Work, Petri nets and related formalisms*, Zaragoza, Spain, June 1994.

[Mur89] T. Murata. Petri Nets: Properties, Analysis and Applications. *Proceedings of the IEEE*, 77(4):541–580, April 1989.

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

[SL96] Software-Ley. *COSA User Manual*. Software-Ley GmbH, Pullheim, Germany, 1996.

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

[WR96] M. Wolf and U. Reimer, editors. *Proceedings of the International Conference on Practical Aspects of Knowledge Management (PAKM'96), Workshop on Adaptive Workflow*, Basel, Switzerland, Oct 1996.