Modelling and Validation with VipTool

Jörg Desel, Vesna Milijic
Department for Applied Computer Sciences
Catholic University Eichstätt-Ingolstadt, Germany
{joerg.desel,vesna.milijic}@ku-eichstaett.de

VipTool was originally developed at the University of Karlsruhe within the research project VIP[1] [2]. It is a tool for modelling, simulation, validation and verification of processes using Petri nets. VipTool allows to specify certain properties graphically. Examples are specific forms of forbidden behaviour (facts and causal chains) and desired behaviour (goals). VipTool generates and visualizes concurrent runs of a given Petri net model.

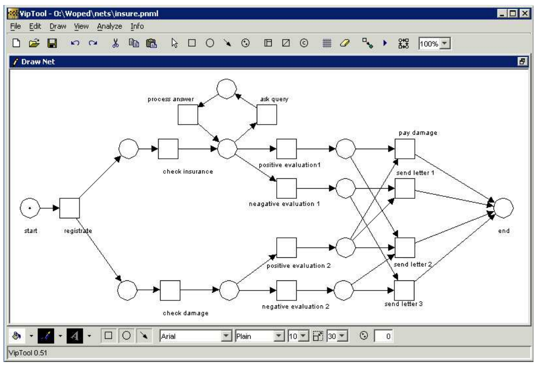

Figure 1: Screenshot of VipEditor, including an example net.

VipTool consists of VipEditor (see Figure 1) which enables the user to design the Petri net model. The modelling is very intuitive and drawing and painting features can be used analogously to standard Windows applications. Size, color, fonts, and other usual graphical parameters can be easily set by the user for all elements (places, transitions, arcs, labels, etc.). All standard editing features, such as select, move, copy, paste, undo, redo etc. are implemented in the usual way. The user-friendly environment is supported by many other features, for example by automatic alignment and by click-and-drag-points of net arcs. The usual token game simulation is also a part of the VipEditor.

VipTool supports exporting of pictures in Encapsulated Post Script (eps). To support the exchange of Petri nets between different tools, VipTool sup-

[1]Verification of Information Systems by evaluation of Partially ordered runs

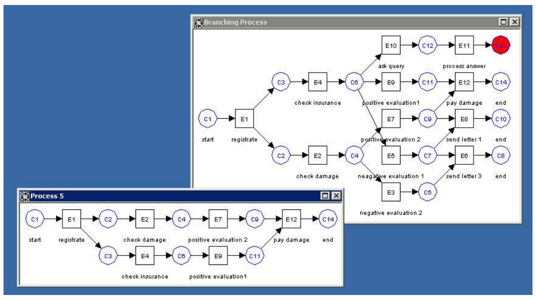

Figure 2: The branching process and a single run of the net in Figure 1

ports PNML (an XML exchange format developed in [4]; this format is now widely accepted as a standard exchange format for Petri nets).

The algorithmic core of VipTool is VipEngine. It computes the runs of the modelled Petri net. This computation is based on the construction of the complete prefix of the branching process of the net. If a Petri net is not too large and has only finite runs, all runs are generated. In the case the Petri net is too complex to generate the complete prefix, VipTool still generates a substantial set of runs because these runs are computed on the fly.

In addition to standard cut-off criteria for terminating potentially infinite runs (described in [3]), further termination criteria like bounds for the number of events or for the depth of the branching process (to be specified by the user) are implemented. The user can decide to compute the whole set of runs from the constructed branching process by applying an appropriate clique-algorithm. To guarantee that substantial runs are computed on the fly, priority criteria can be employed in the beginning.

The runs are visualized using VipVisualizer, which is based on the Sugiyama graph-drawing algorithm accommodated in [3].

References

[1] W.M.P. van der Aalst, J. Desel and A. Oberweis (Eds.). *Business Process Management.* Springer, LNCS 1806, 2000.

[2] J. Desel. Validation of Process Models by Construction of Process Nets. In [1], pp. 110–128.

[3] T. Freytag. *Softwarevaliedierung durch Auswertung von Petrinetz-Abläufen.* Dissertation, University of Karlsruhe 2001.

[4] M. Weber, E. Kindler. The Petri Net Markup Language. In H. Ehrig, W. Reisig, G. Rozenberg, H. Weber (Eds.). *Petri Net Technology for Communication Based Systems.* LNCS 2472, Springer 2003.