CPN IDE: An Extensible Replacement for CPN Tools That Uses Access/CPN (Extended Abstract)

1st Eric Verbeek
Mathematics and Computer Science
Eindhoven University of Technology
Eindhoven, The Netherlands
h.m.w.verbeek@tue.nl, 0000-0002-1658-9679

2nd Dirk Fahland
Mathematics and Computer Science
Eindhoven University of Technology
Eindhoven, The Netherlands
d.fahland@tue.nl, 0000-0002-1993-9363

***Abstract*—This extended abstract introduces CPN IDE, which replaces CPN Tools as a tool for editing and simulating (Coloured) Petri Net models. The main advantage of CPN IDE is that it is an extensible tool, which is needed to keep it running and to add new features which are of interest to the process mining community, like easily generating event logs.**

***Index Terms*—Coloured Petri Net (CPN), CPN Tools, Access/CPN, CPN IDE**

I. From CPN Tools via Access/CPN to CPN IDE

A. CPN Tools

CPN Tools [1] is a tool that is well-known in the Petri net community. CPN Tools provides a mature environment for constructing, simulating, and performing analysis of CPN (Coloured Petri Net) models [2]. CPN Tools consists of a CPN simulator (the back-end) based on ML (Meta Language), and a CPN editor (the front-end) that has been developed in the BETA programming language [3] using the Mjølner development system [4].

In the autumn of 2010, CPN Tools has been transferred from the CPN Group at Aarhus University to the Mathematics and Computer Science department of the Eindhoven University of Technology (TU/e), The Netherlands. In this department, CPN Tools has been used extensively in courses both as an editor and as a simulator for both coloured Petri nets [2] and (uncoloured) P/T nets [5] (like workflow nets [6]). These courses include our process mining [7] courses, where CPN Tools is used as a Petri net editor next to our process mining tool ProM [8]. As an example, we can first discover a Petri net from an event log with ProM, second we can edit the discovered Petri net using CPN Tools, and third we can replay the event log on the edited Petri net using again ProM.

Although the latest versions of CPN Tools are still working on recent operating systems like Windows 10, we foresee that at some point in the future changes have to be made to CPN Tools to keep it running, and that in particular changes to the CPN editor may be needed. The main problem with the current CPN editor is that the language and development system used for it (BETA and Mjølner) are rather uncommon and unlike other languages and systems. As a result, only a few people actually know how to change the current BETA-based CPN editor, and it is very hard to extend CPN Tools with new functionalities.

B. Access/CPN

This problem of using the BETA language and the Mjølner system has already been acknowledged in [9], which also introduced Access/CPN to alleviate it. Access/CPN wraps the CPN simulator in a Java-based implementation of CPN models. As a result of this, Java programs can now easily embed the ML-based simulator.

C. CPN IDE

As CPN Tools is used in courses on process mining at the TU/e, we did not want to wait until CPN Tools stops working due to new or updated operating systems. Therefore, we have developed a new tool called CPN IDE, that offers a new JavaScript-based CPN editor on top of Access/CPN. This CPN editor communicates through a REST[1] interface with a Java-based controller that embeds Access/CPN and that is running as a Spring Boot[2] server. When the CPN editor is started, the Spring Boot server with the controller is also started. As usual, Access/CPN will take care of starting an ML-based CPN simulator when needed.

As a result, CPN IDE can be extended either on the level of the CPN editor (using JavaScript) or on the level of the controller (using Java).

II. A brief glance

Figure 1 shows a screenshot of CPN IDE while it is simulating a workflow net discovered with ProM from an event log. In this screenshot, tokens are present in the places labeled “P 2”, “P 5”,“P 8” (green dots with numbers), the transitions labeled “g” are “h” ready to be fired (green auras) while the transition labeled “r” is being fired (red aura), which will result in new tokens in the places labeled “P 19” and “P 30” (red dots without numbers moving along the arcs). The console at the bottom shows information about which transitions were enabled, which transition is fired, and which tokens are created for which places. At the right-hand side, we see specific information on the selected node, which is now the transition labeled “r” which has the id “Trans141”.

[1]Representational state transfer, see https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm, accessed on September 1, 2021)

[2]See https://spring.io/projects/spring-boot, accessed on September 1, 2021

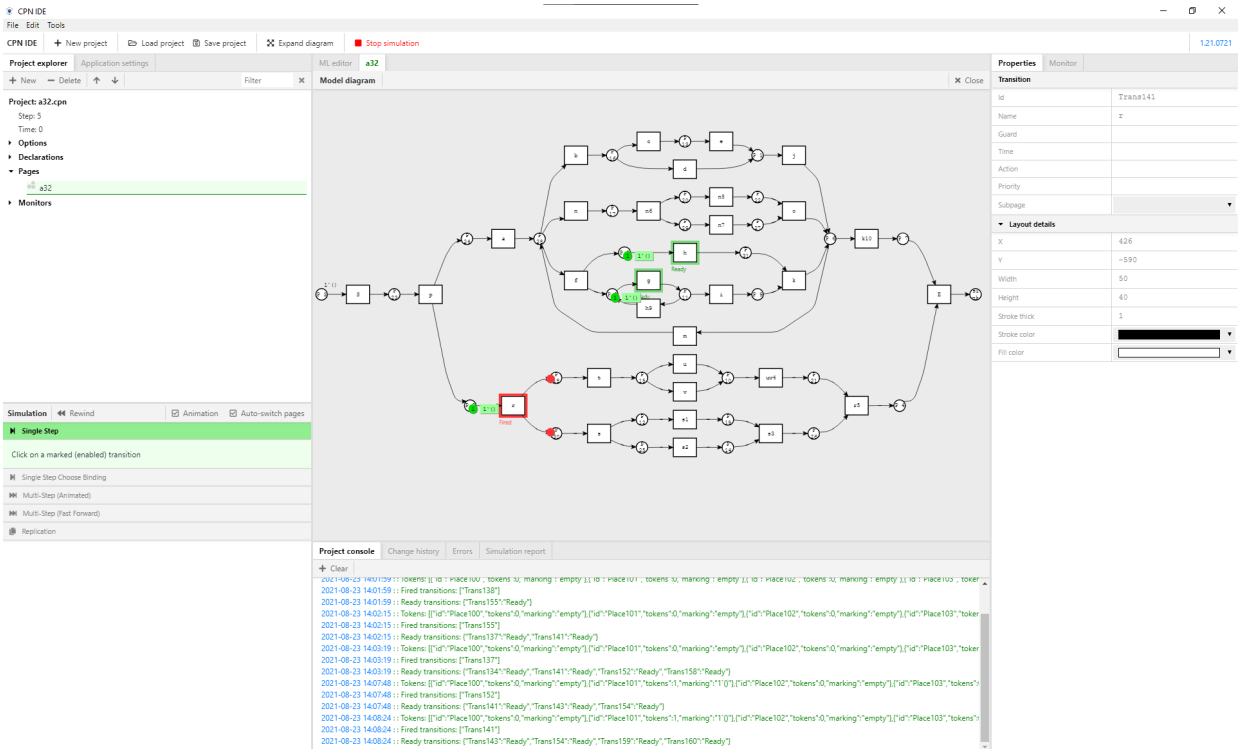

Fig. 1. Simulation of a model discovered with ProM from an event log.

III. Links

A. Downloads

From https://cpntools.org/cpn-ide/ you can download the latest releases for CPN IDE. The first release is CPN IDE 1.21.

B. Screencast

From https://cpntools.org/cpn-ide/ you can also access a screencast on CPN IDE. In this screencast, we create a workflow net for the running example from [7] handling compensation requests, and perform a simulation on it.

C. Sources

On https://github.com/cpn-io/cpn-js/ you will find the sources for CPN IDE. CPN IDE will be Open Source.

IV. Future work

- Export to XES [10] At the moment, there is no easy way to generate an event log in XES format from a running simulation (like in the screencast). We would like to add this functionality into CPN IDE.
- Import/export from/to PNML [11]. At the moment, CPN IDE (and CPN Tools as well) can be used together with ProM because ProM can import/export P/T nets from/to CPNXML files[3], which is the native file format for both CPN IDE and CPN Tools. However, if CPN IDE could import/export workflow nets from/to PNML, then other tools could also be used together with CPN IDE.

[3]As from ProM 6.10, the "CPNXML export (Petri net)" plugin exports a P/T net to the CPNXML format and the "Import Petri net from CPNXML file" plugin imports a P/T net from a CPNXML file.

References

[1] M. Westergaard and H. M. W. Verbeek. (2018) CPN Tools. [Online]. Available: https://cpntools.org/

[2] K. Jensen and L. M. Kristensen, *Coloured Petri Nets Modeling and Validation of Concurrent Systems*. Springer-Verlag, 2009.

[3] O. L. Madsen, B. Møller-Pedersen, and K. Nygaard, *Object-Oriented Programming in the BETA Programming Language*. Addison-Wesley, 1993.

[4] J. L. Knudsen, M. Lofgren, O. L. Madsen, and B. Magnusson, *Object-Oriented Environments - The Mjølner Approach*. Prentice Hall, 1993.

[5] W. Reisig, *Petri Nets: An Introduction*, ser. EATCS Monographs in Theoretical Computer Science. Springer-Verlag, Berlin, 1985, vol. 4.

[6] W. M. P. v. d. Aalst, "The application of Petri nets to workflow management," *The Journal of Circuits, Systems and Computers*, vol. 8, no. 1, pp. 21–66, 1998.

[7] ——, *Process Mining: Data Science in Action*, 2nd ed. Springer-Verlag, 2016.

[8] H. M. W. Verbeek, J. C. A. M. Buijs, B. F. v. Dongen, and W. M. P. v. d. Aalst, "Xes, xesame, and prom 6," in *Information System Evolution*, ser. Lecture Notes in Business Information Processing (LNBIP), P. Soffer and E. Proper, Eds. Hammamet, Tunisia: Springer, June 7-9 2011, vol. 72, pp. 60–75.

[9] M. Westergaard and L. M. Kristensen, "The access/cpn framework: A tool for interacting with the cpn tools simulator," in *Applications and Theory of Petri Nets*, G. Franceschinis and K. Wolf, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 313–322.

[10] G. Acampora, A. Vitiello, B. Di Stefano, W. M. P. v. d. Aalst, C. W. Günther, and H. M. W. Verbeek, "IEEE 1849TM: The XES standard: The second IEEE standard sponsored by IEEE Computational Intelligence Society," *IEEE Computational Intelligence Magazine*, pp. 4–8, May 2017.

[11] J. Billington and et. al., "The Petri Net Markup Language: Concepts, Technology, and Tools," in *Application and Theory of Petri Nets 2003*, W. Aalst and E. Best, Eds., vol. 2679, 2003, pp. 483–506.