# DALG: The Data Aware Event Log Generator

## Step-by-step Guide

Trier University
Department IV - Computer Science
Chair of Business Informatics II

German Research Center for Artificial Intelligence
Smart Data & Knowledge Services - Trier Branch
Experience-based learning systems

David Jilg
david.jilg@dfki.de
https://github.com/DavidJilg/

# Contents

# Figures

# 1. Introduction

Welcome to the step-by-step tutorial for DALG: The Data Aware Event Log Generator. DALG is developed by David Jilg, Joscha Grüger, and Tobias Geyer at the chair of Business Information Systems II[1] at the University of Trier in collaboration with the Trier Branch of the German Research Center for Artificial Intelligence[2].

DALG uses the SAMPLE [GGJB23] approach, which itself uses token-based simulation methods, to generate event logs that conform to the data and the control flow perspective of a provided data Petri net. The software is open-source, and access is provided via a GitHub repository[3]. This document provides a step-by-step guide to using DALG with an exemplary use-case. The guide is also available as a video showcase, which is also published in the GitHub repository[4]. However, the video and this guide will not cover all features and configuration options of the software. A full guide to DALG and all of its features can be found in DALGs user manual[5]. It should be noted that the users following this guide are expected to have basic knowledge of process mining and data Petri nets. The document is structured as follows. First, Section 2 provides a guide to acquiring, installing, and running DALG. Subsequently, Section 3 will provide a step-by-step tutorial on using DALG to generate an event log from an exemplary Petri net.

# 2. Acquiring and Running DALG

This section will guide you through acquiring, installing, and running DALG on your computer for the first time. Additionally, Section 2.1 will first provide information about the operating system compatibility of the software.

## 2.1 Operating system compatibility

DALG is generally compatible with Windows and Linux-based operating systems. However, the software relies on the QT6 framework[6] for the graphical user interface. The following operating systems are officially supported by QT6.6.

1. Red Hat 8.6, 8.8, 9.2
2. openSUSE 15.5

[1] https://www.wi2.uni-trier.de/
[2] https://www.dfki.de/en/web
[3] https://github.com/DavidJilg/DALG
[4] https://github.com/DavidJilg/DALG/blob/main/src/documentation/DALG%20Showcase.mp4
[5] https://github.com/DavidJilg/DALG/blob/main/src/documentation/manual.pdf
[6] https://www.qt.io/product/qt6

3. SUSE Linux Enterprise Server 15 SP5
4. Ubuntu 22.04
5. macOS 11, 12, 13
6. Windows 10, 11, Windows on ARM

Due to the limited operating system compatibility of the QT framework, it was decided to only officially support and test DALG for Windows 10, Windows 11, and Ubuntu 22.04. However, just because an operating system is not officially supported does not mean the program cannot work on it. It is also possible to run DALG without the graphical user interface. Therefore, it should be possible to run it on most Windows and Linux based operating systems by running it from the command line or the terminal. Information on how that can be done is provided in DALG's user manual.

## 2.2 Acquiring DALG

DALG is open-source, and access is provided via a GitHub repository[7]. You can download the software by navigating to the release section of the repository (https://github.com/DavidJilg/DALG/releases) and downloading the latest release. If you are using a machine running Windows, you can choose between an installer and a portable installation. If you are using Linux, you must use the portable installation since there is no installer available. We will be using the Windows version for this guide, but the process of running and operating DALG is very similar on Linux.

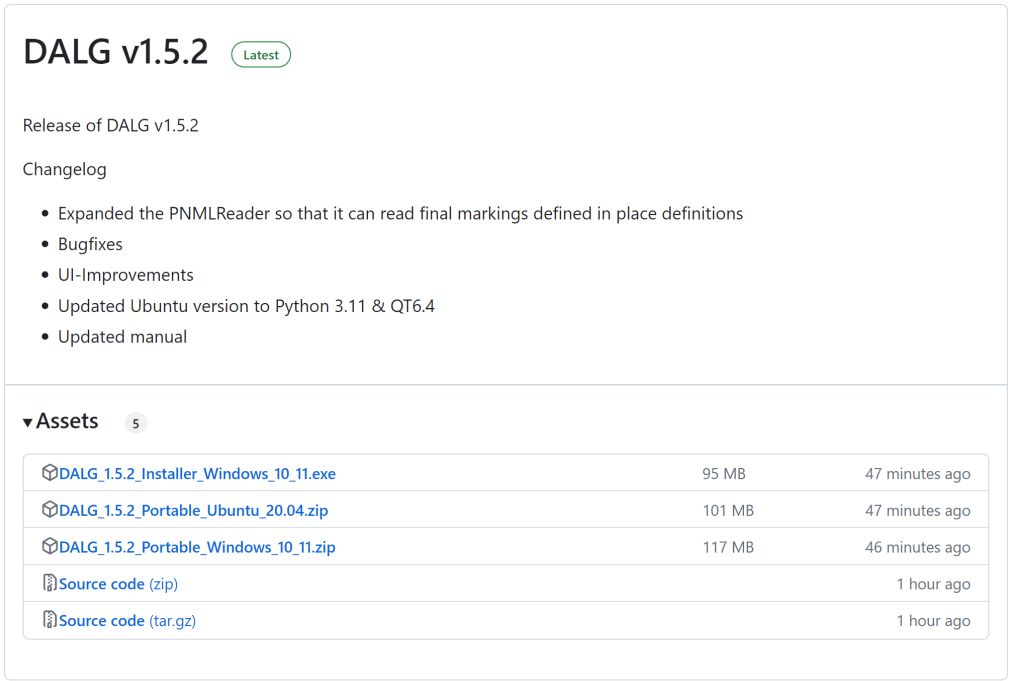


Figure 2.1: An exemplary DALG release on GitHub

## 2.3 Installing and Running DALG

If you have chosen a portable installation, you will only need to extract the `.zip` archive to the desired installation directory to install DALG. If you are using the installer, you need to run it and the installation-wizard will guide you through the installation (see 2.2).

[7] https://github.com/DavidJilg/DALG

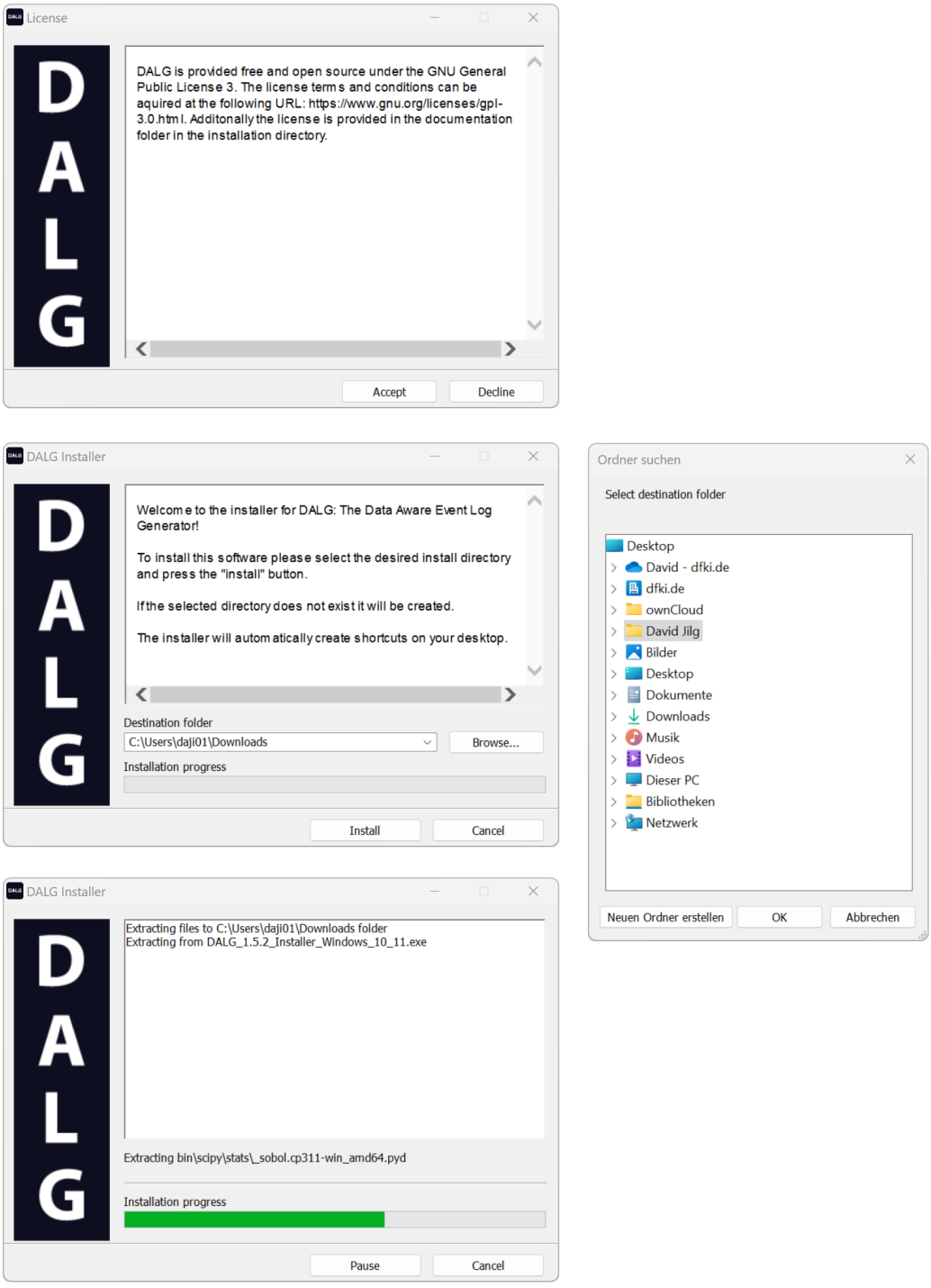


Figure 2.2: Screenshot of the installation-wizard of DALG

Before DALG can be installed through the installation wizard, you will have to read and accept the license agreement. DALG is licensed under the commonly used GNU General Public License v3.0[8]. After accepting the license agreement, you will need to choose an installation directory by typing a path into the text box or clicking the 'Browse...' button and selecting a folder. The program needs permission to read and write files from its installation folder and the specified output directory for the program to function. You need to make sure that the program has these permissions. For example, if you install the program in the 'C:/Program Files' folder on Windows 10 you might have to start the installer and the program itself with administrator rights or else it will not work correctly.

[8] https://www.gnu.org/licenses/gpl-3.0.en.html

After you have specified the installation directory, you can press the 'Install' button, and the wizard will install DALG, and it will also create a shortcut on your desktop. You can either run DALG by executing the shortcut on your desktop or by navigating to the installation directory and running the executable called 'DALG'. Alternatively, you could also run DALG in command line mode, but in this guide we will be using the user interface since it is the recommended way of using this software. The command line mode is covered in DALGs user manual.

# 3. Using DALG: An exemplary Use-Case

This section will guide you through using DALG to generate event logs based on an exemplary data Petri net. DALG can load Petri nets modeled in the Petri net Markup Language[9](`.pnml` files). For this guide, we will be using a publicly available data Petri net that models the management of road fines [10].

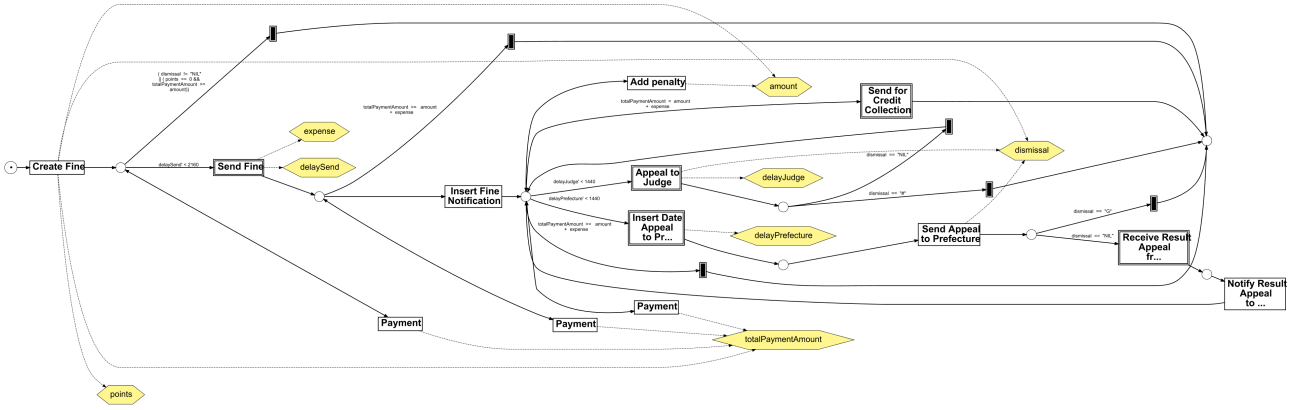


Figure 3.1: Data Petri Net for Road-Fine Management

When you open DALG for the first time, you will be greeted by a welcome screen that provides some basic information about the software (see 3.2). You will also find a button there that will open the user manual for you.

[9]https://www.pnml.org/
[10]https://github.com/bytekid/cocomot/blob/main/data/road_traffic_billing/RoadFines_WithData.pnml

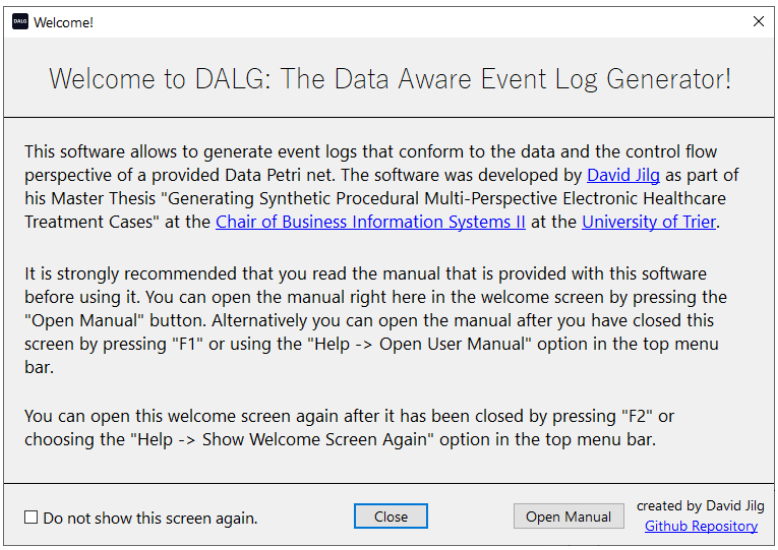


Figure 3.2: Welcome screen of DALG

Do not forget to check the box on the bottom left if you do not want this welcome screen to appear every time you start DALG. When you close the welcome screen, you will see the main user interface of DALG as shown in Figure 3.3. The interface is divided into six distinct parts, which are marked with the colored boxes in the figure.

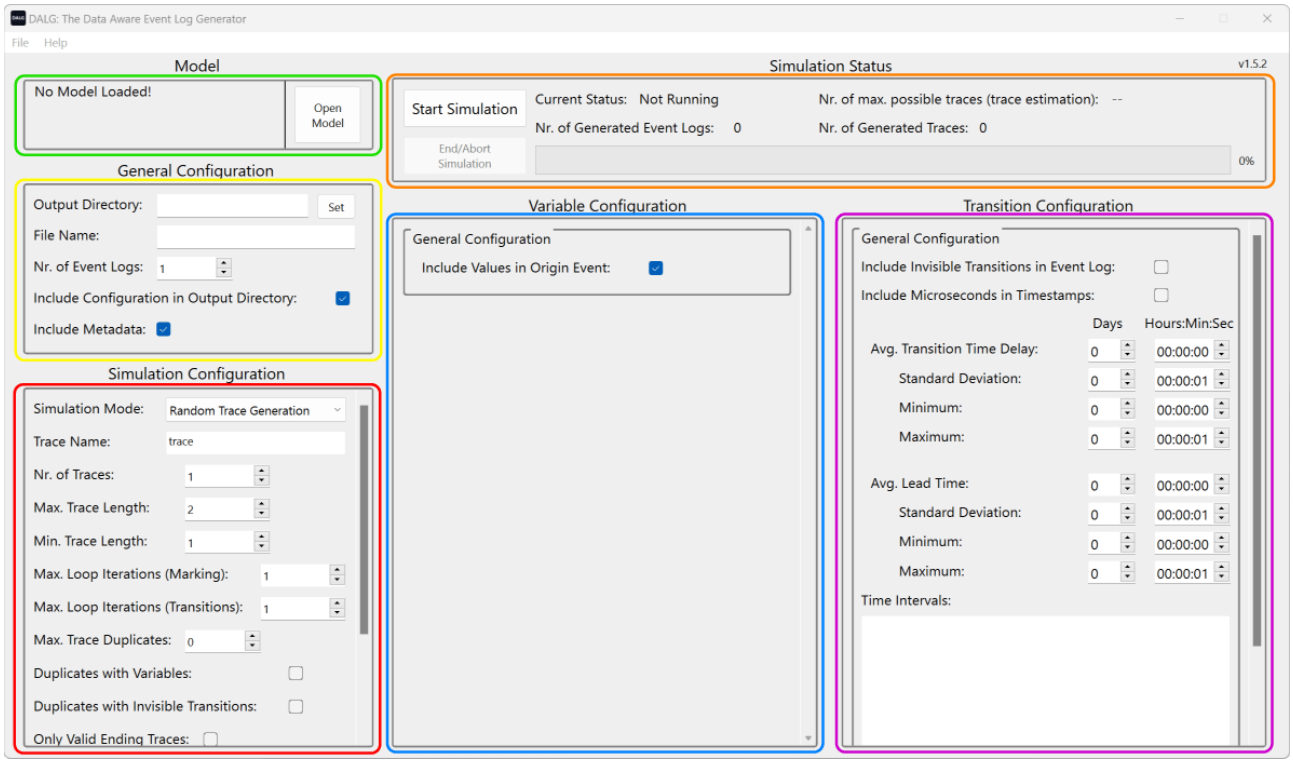


Figure 3.3: User Interface of DALG

To generate event logs with DALG you will have to complete three steps. First, you need to load a Petri net model. Subsequently, you need to configure DALG. This includes configuring the simulation and also configuring the additional semantic information necessary to generate more realistic event logs. Lastly, you will have to run the simulation to generate the desired event logs. The following paragraphs will guide you through these steps.

## 3.1 Loading a data Petri net

The first step towards generating synthetic event logs consists of opening a Petri net model. Go ahead and press the 'Open Model' button in the area surrounded by the green box in Figure 3.3. Alternatively, you can use the top menu (see Figure 3.4) or use the key combination `Ctrl+O`. This will open the standard file dialog of the operating system you are using, and you can select any `.pnml` file to load it into DALG.

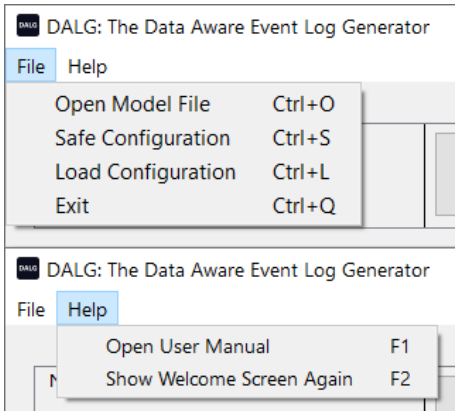


Figure 3.4: Top menu options of DALG

## 3.2 Configuring the general configuration and the simulation

DALG offers many configurations options that you can use to tailor the event logs to your liking. We will start by configuring the general configuration options. You can find these options in the area marked by the yellow box in Figure 3.3. Go ahead and choose an output directory and a filename for the `.xes` file that you want to generate. We will ignore the other options for now, since they are not needed to generate an event log.

Following up on the general configuration, we will now configure the token-based simulation that will be used to generate the event logs. Take a look at the area marked with the red box in Figure 3.3. In this area, you can, among other things, configure the simulation mode, the number of traces, i.e. number of simulation runs, and the minimum/maximum trace length. Go ahead and set the simulation mode to 'Random Trace Generation', set the number of traces to 50, and the maximum trace length to 15. Lastly, set both the loop iteration options to 3 (There is no perfect way of detecting loops when executing Petri net models, so DALG uses two different techniques to count loop iterations. This prevents infinite loops.), and tick the box for having only valid ending traces, when you only want traces that reach a valid final marking.

## 3.3 Configuring the additional semantic information

To achieve the generation of event logs that are meaningful when viewed from a semantic perspective, DALG offers the user the ability to supply additional semantic information that is used together with the Petri net model to generate more realistic event logs. Take a look at the areas marked with the blue and magenta boxes in Figure 3.3. There is currently no model loaded in the screenshot that is used in this figure. The `Variable Configuration` and `Transition Configuration` areas on your screen should be populated with configuration options for the variables and transitions in the Petri net model we are using for this guide. Figure 3.5 shows DALG running a simulation on this model. Therefore, you can also see all configuration options. In these areas, you can provide additional semantic information, such as the minimum and maximum values for variables or the amount of time that an activity associated with a transition should accommodate. Go ahead and set the maximum value for the variable 'amount' to 1000 so that a person cannot get a fine that is higher than 1000 units of currency. We will skip over the rest of the configuration options in this guide, since they are not essential for generating event logs. Feel free to add as much semantic information as you like before you proceed to the last step, which is described in the next section.

## 3.4 Running the simulation

Now that you have configured the DALG, you can start the simulation and the event logs will be generated. Take a look at the area marked with the orange box in Figure 3.3. This area is used to control and monitor the simulation runs. For now, you only have to worry about the 'Start Simulation' button and the progress bar. When you hit this button, the token-based simulation will start and the progress bar will start to advance. Go ahead and press that button and wait for the simulation to finish. Figure 3.5 shows a screenshot of DALG running a simulation with the road fines model. Once the simulation is finished, DALG will save the generated event logs to the specified output directory and display a message summarizing the simulation. Figure 3.6 shows an exemplary trace generated by DALG based on the road fines model.

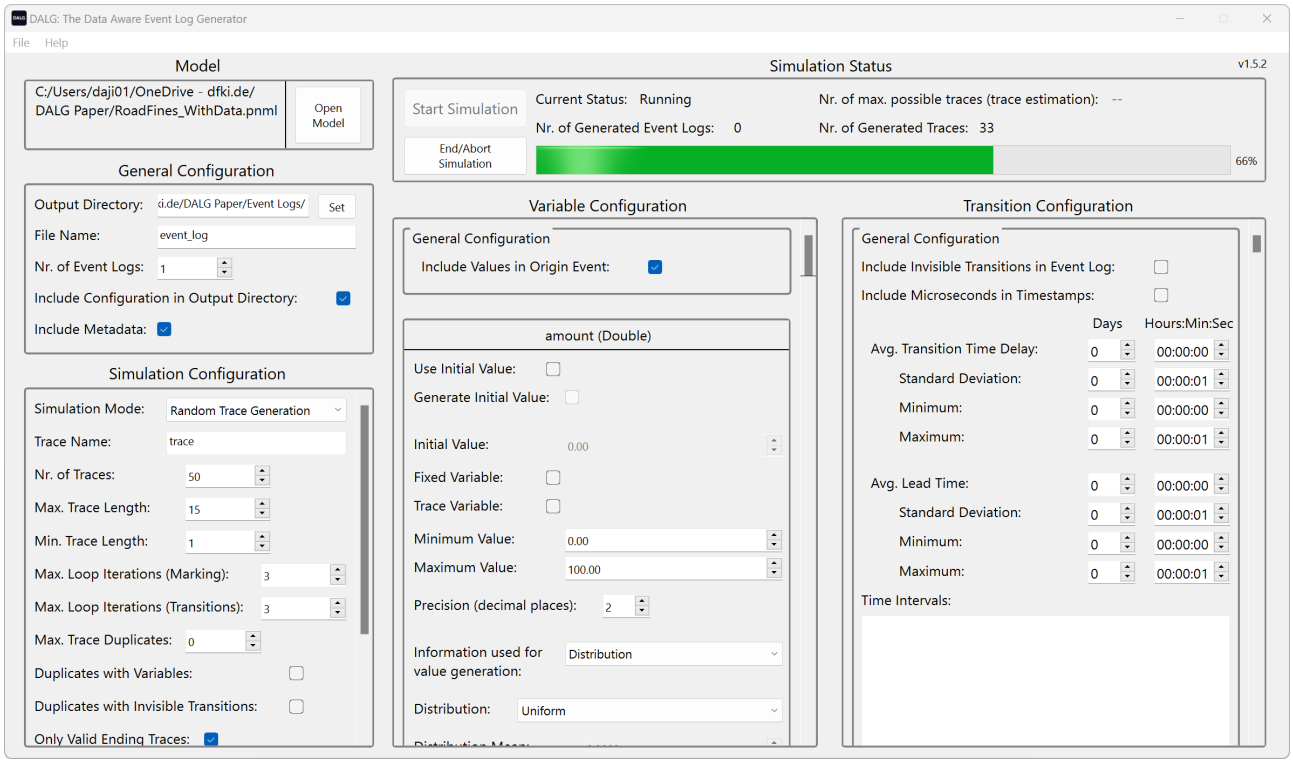


Figure 3.5: DALG Running

```
<trace>
    <string key="concept:name" value="trace1"/>
    <event>
      <string key="concept:name" value="Create Fine"/>
      <date key="time:timestamp" value="2000-01-01T00:00:00+00:00"/>
      <float key="amount" value="38.95"/>
      <float key="totalPaymentAmount" value="58.67"/>
      <int key="points" value="95"/>
      <string key="dismissal" value="G"/>
    </event>
    <event>
      <string key="concept:name" value="Send Fine"/>
      <date key="time:timestamp" value="2000-01-01T00:00:01+00:00"/>
      <float key="amount" value="38.95"/>
      <float key="totalPaymentAmount" value="58.67"/>
      <int key="points" value="95"/>
      <string key="dismissal" value="G"/>
      <int key="delaySend" value="26"/>
      <float key="expense" value="52.47"/>
    </event>
    <event>
      <string key="concept:name" value="Payment"/>
      <date key="time:timestamp" value="2000-01-01T00:00:01+00:00"/>
      <float key="amount" value="38.95"/>
      <float key="totalPaymentAmount" value="8.44"/>
      <int key="points" value="95"/>
      <string key="dismissal" value="G"/>
      <int key="delaySend" value="26"/>
      <float key="expense" value="52.47"/>
    </event>
    <event>
      <string key="concept:name" value="Insert Fine Notification"/>
      <date key="time:timestamp" value="2000-01-01T00:00:03+00:00"/>
      <float key="amount" value="38.95"/>
      <float key="totalPaymentAmount" value="8.44"/>
      <int key="points" value="95"/>
      <string key="dismissal" value="G"/>
      <int key="delaySend" value="26"/>
      <float key="expense" value="52.47"/>
    </event>
    <event>
      <string key="concept:name" value="Appeal to Judge"/>
      <date key="time:timestamp" value="2000-01-01T00:00:03+00:00"/>
      <float key="amount" value="38.95"/>
      <int key="delayJudge" value="42"/>
      <float key="totalPaymentAmount" value="8.44"/>
      <int key="points" value="95"/>
      <string key="dismissal" value="NIL"/>
      <int key="delaySend" value="26"/>
      <float key="expense" value="52.47"/>
    </event>
 </trace>
```

Figure 3.6: Example trace of an event log generated from the road fines model

Depending on your configuration, DALG can get stuck and may not be able to finish generating the desired number of traces. For example, DALG gets stuck when you have configured it to generate 50 unique traces and the provided model only allows for 30 unique traces. In such a case, you can press the 'End/Abort Simulation' button at any time. If you abort the simulation, DALG will ask you if

you want to save the traces generated so far. So, no traces generated will be lost even if you end the simulation early.

# 4. Conclusion

This brief guide only shows a basic use of DALG. The software offers many more features that were not discussed in this guide. The full scope of functions is described in the manual and the underlying approach in the paper SAMPLE: A Semantic Approach for Multi-perspective Event Log Generation [GGJB23]. You can access DALGs user manual by using the top menu or pressing `F2`. Alternatively, you can access it directly on GitHub[11].

[11] https://github.com/DavidJilg/DALG/blob/main/src/documentation/manual.pdf

# References

[GGJB23] J. Grüger, T. Geyer, D. Jilg und R. Bergmann. SAMPLE: A Semantic Approach for Multi-perspective Event Log Generation. In M. Montali, A. Senderovich und M. Weidlich (Hrsg.), *Process Mining Workshops*, Cham, 2023. Springer Nature Switzerland, S. 328–340.