

RuM Tutorial

ICPM 2020 Tool-Tutorial

Table of Contents

Introduction	2
Home Screen	2
Process discovery	3
Conformance Checking	5
MP-Declare Editor	7
Process Monitoring	9
Inventory	11
Log Generation	12

Introduction

This tutorial gives a short overview of RuM, a desktop application for rule mining. The tutorial follows the same main steps shown in: <https://youtu.be/nXFNDDBOcU0>.

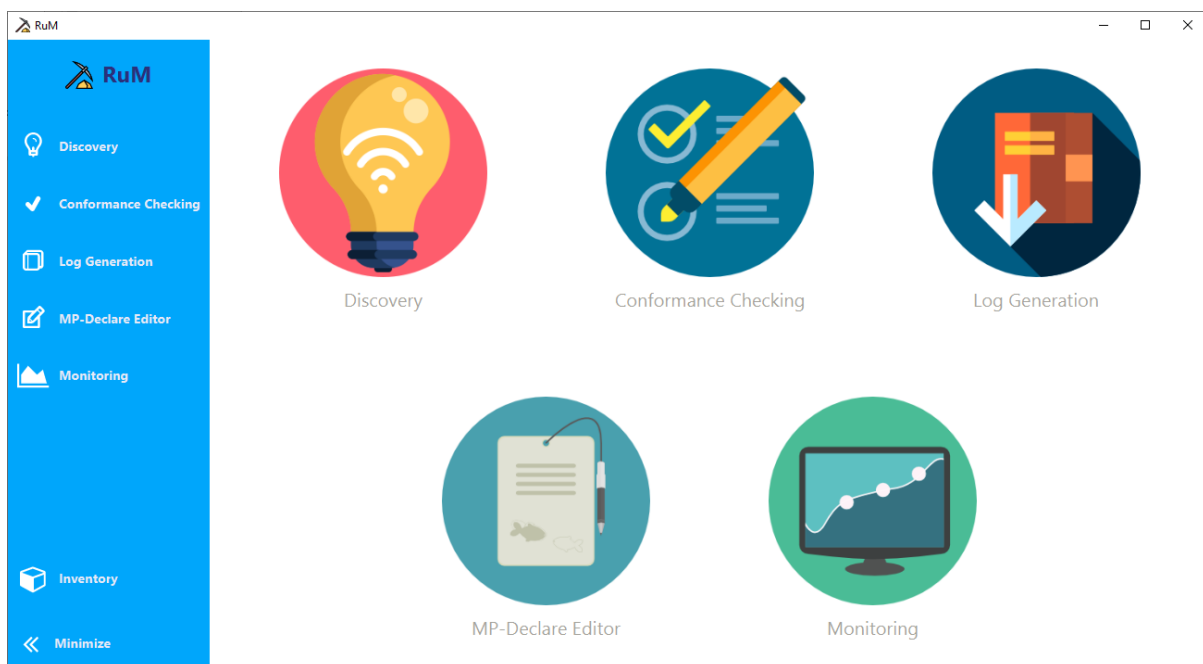
RuM is publicly available at: <https://rulemining.org/>. RuM requires installing Java 11 JDK which can be found at: <https://www.oracle.com/java/technologies/javase-jdk11-downloads.html>.

The event log used in this tutorial can be found at: <https://git.io/JUvav>.

Home Screen

After starting RuM, the home screen is shown that provides access to all the main sections of the application, which are Discovery, Conformance Checking, Log Generation, MP-Declare Editor, and Monitoring. These sections can also be accessed via the navigation menu on the left. Additionally, the navigation menu provides access to an inventory section that contains all the files imported from the file system or saved using the button “*Take Snapshot*” available in some of the sections of the tool. By clicking on the RuM icon, the home screen appears. The navigation menu can also be minimized so that it takes up less space.

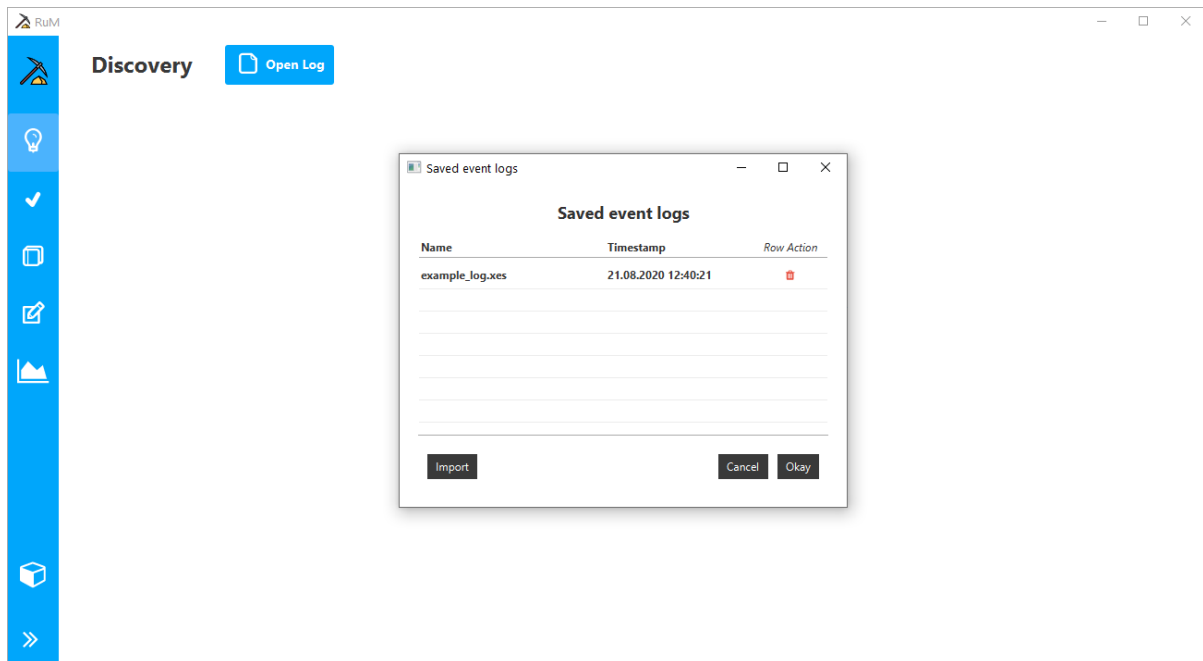
This tutorial covers all the main sections of RuM.



Process discovery

The Discovery section can be opened by clicking the discovery button in the home screen or by using the navigation menu on the left panel of the application.

In the discovery section, it is possible to open an event log by clicking the “*Open Log*” button. This opens an inventory window which displays all the event logs imported so far. An event log can be imported from the file system using the “*Import*” button and then selecting the event log file. This adds the log to the inventory so that the log can be opened (from any section of the tool) either by double-clicking the row corresponding to the log or by selecting the row and clicking “*Okay*”.

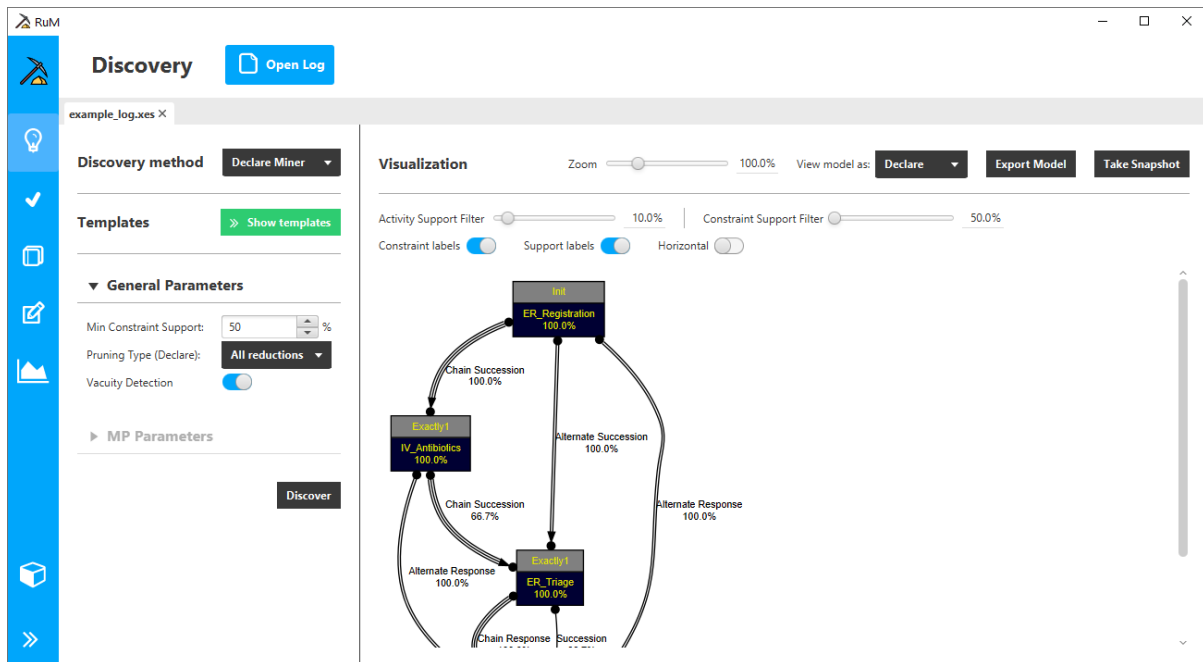


After opening an event log in the discovery section, a process model is automatically discovered using the default parameters for discovery. The parameters are shown in the left panel of the section and the resulting model is shown on the right. The model can be explored using three different views¹. The Declare view, which uses the standard Declare notation, the textual view which gives a description of each constraint in natural language and the automaton view that represents the model as an automaton. If the parameters in the left panel are changed, then the model must be rediscovered in order for the changes to take effect.

In this tutorial we will set “*Min Constraint Support*” to 50% and click “*Discover*” to rediscover the model and to obtain the constraints with a minimum support of 50% in the log which means that constraints satisfied in at least 50% of the traces in the log are discovered.²

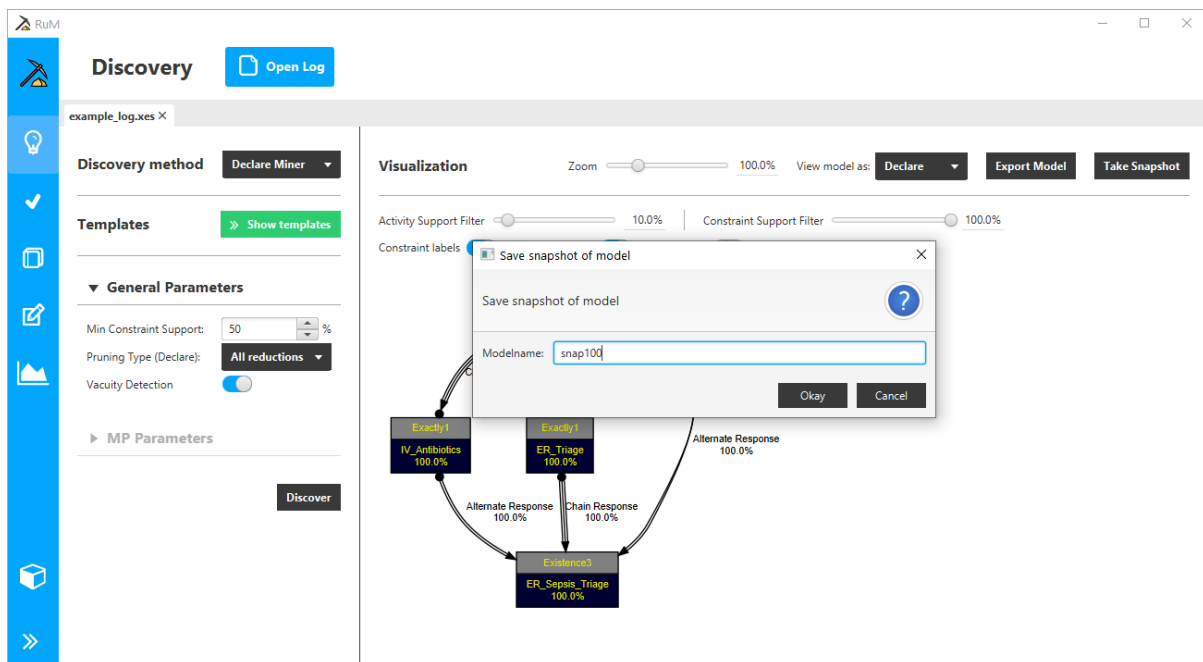
¹ The format of the exported model depends on the currently selected view. The export format for the Declare view is *decl*, for the Textual view *txt*, and for Automaton view *dot*.

² For the Declare miner this means that constraints satisfied in at least 50% of the traces in the log are discovered (trace-based support), whereas for MINERful this means that constraints with at least 50% of the activations leading to a fulfilment are discovered (event-based support).



As already mentioned, it is possible to take snapshots of the models (and also of the event logs) obtained as outputs in any of the sections of the tool so that they become available to be used in the other sections of the application. A snapshot is basically a temporary image of a file that can be used in RuM as if it is an actual file imported from the file system. For example, starting from a discovered model, it is possible to take snapshots of the model by clicking the button “Take Snapshot”.

In this tutorial, two snapshots of the model just discovered are taken. The first one, “snap50”, is taken by setting the “Constraint Support Filter” to 50% and then clicking the button “Take Snapshot”. This adds a snapshot of the model just discovered to the inventory. The second one, “snap100”, is taken by setting the “Constraint Support Filter” to 100% to filter out all the constraints that have less than 100% support (*changing the filters does not require rediscovering the model*), and then clicking the button “Take Snapshot” again.

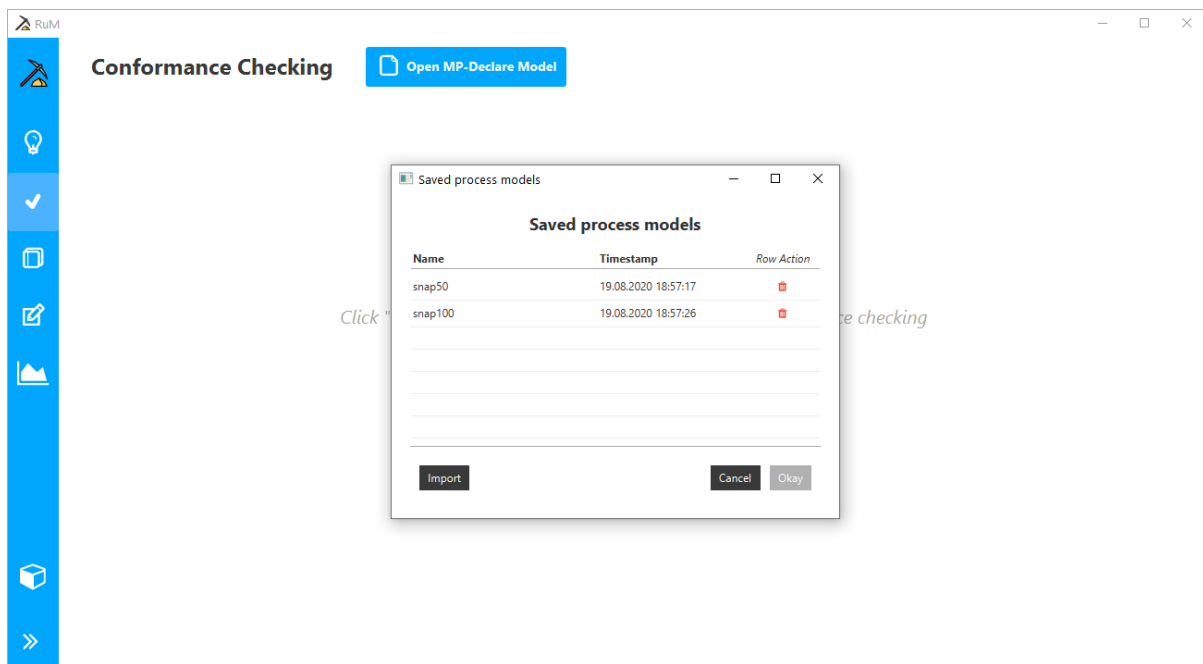


Now two independent versions of the same model are available in the inventory as snapshots. Both these snapshots can be used as inputs in the other sections of the tool, for example, for conformance checking.

Conformance Checking

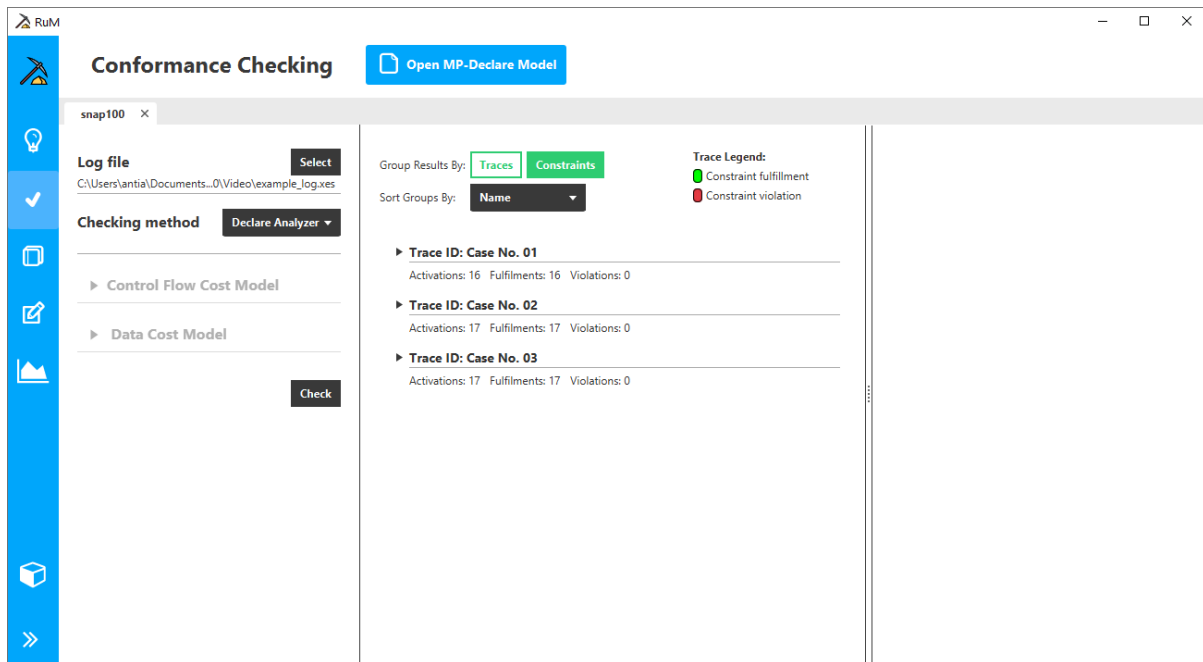
The Conformance Checking section can be opened by clicking the conformance checking button in the home screen or by using the navigation menu on the left panel of the application.

In the conformance checking section, it is possible to open a model by clicking the “*Open MP-Declare Model*” button. Opening the model is similar to opening an event log. In this case, when the inventory window is opened, both the snapshots taken previously are available in the inventory.



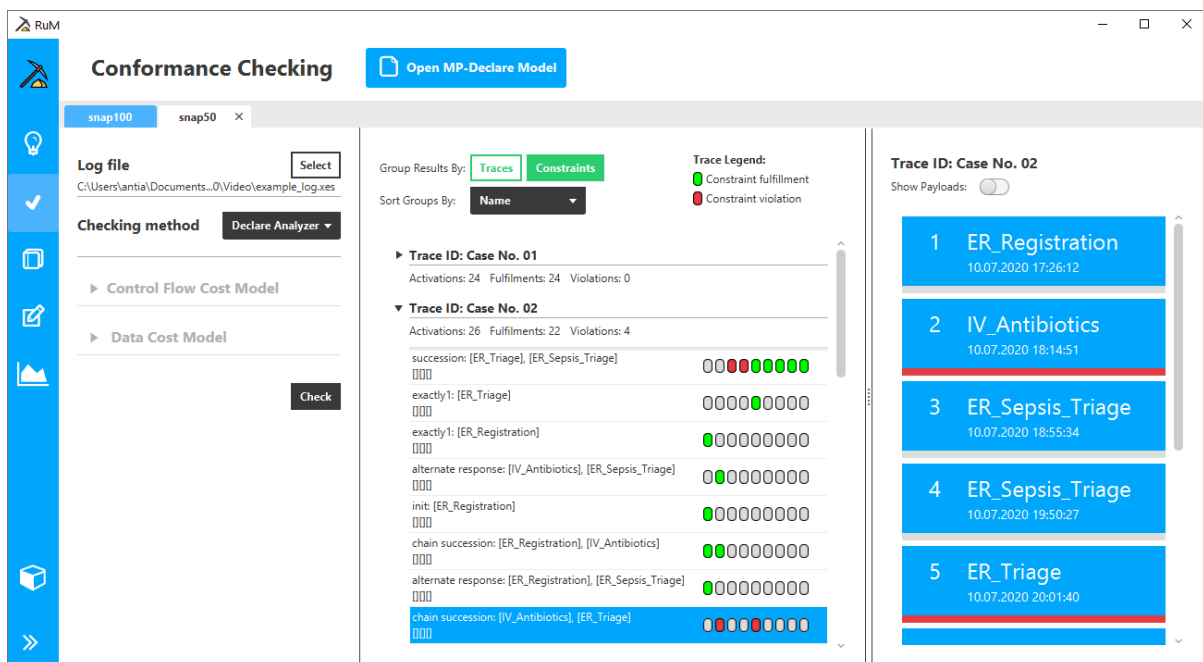
To use the snapshot “*snap100*” for conformance checking, it is possible to open it either by double-clicking the row corresponding to the snapshot or by selecting the row and clicking “*Okay*”.

By clicking “*Select*” in the parameters panel on the left (parameter “*Log file*”) the inventory window opens, and the previously imported event log can be selected without importing it again. Finally, the conformance checking can be started by clicking the “*Check*” button.



The results are displayed in two columns. The first column shows an overview of the results grouped either by trace or by constraint. Each group can be expanded to see detailed results about the group. If the results are grouped by trace, then the details of a group show how the corresponding trace is affected by each constraint of the model. If the results are grouped by constraint, then the details of a group show how this specific constraint affects each trace in the event log. By clicking on a trace in one of the groups, it is possible to see the details of the trace in the second column.

When using the snapshot “*snap100*”, there are no constraint violations. This is expected since all the constraints in this snapshot have 100% support (i.e., they are satisfied in all traces). However, when the same check is run using the snapshot “*snap50*” (following the same steps as before) then some violations should be detected.



Indeed, there are two constraints in this snapshot that cause violations in the second trace of the event log. Note that the violated constraints are the ones with support lower than 100% derived from the discovery.

Also notice that there are now two tabs in the conformance checking section. The first one contains the results for the snapshot “*snap100*” and the second one contains the results for the snapshot “*snap50*”. These tabs can be used independently from each other and it is possible to navigate from one tab to the other freely. In general, all the sections of the tool and all the tabs in each section run on parallel threads so that the execution of a task in a tab does not affect the other tabs and multiple process mining tasks can be executed simultaneously.

MP-Declare Editor

The MP-Declare Editor can be opened by clicking the editor button in the home screen or by using the navigation menu on the left panel of the application.

In the MP-Declare Editor, it is possible either to open an existing model by clicking the “*Open MP-Declare Model*” button or to create an entirely new model by clicking the “*New MP-Declare Model*” button. In this tutorial, the snapshot “*snap100*” is opened and modified.

The screenshot shows the MP-Declare Editor interface. On the left is a navigation menu with icons for home, voice input, and activities/attributes. The main area is divided into three sections: Voice & Text Input, Activities & Attributes, and Constraints. The Visualization section shows a Petri net diagram with activities: ER_Registration, IV_Antibiotics, ER_Triage, and ER_Sepsis_Triage. The Constraints section contains a table with the following data:

Template	Activation (A)	Activation Condition	Target (B)	Correlation Condition	Time Condition	Row Actions
Init[A]	ER_Registration					
Alternate Successi...	ER_Registration		ER_Triage			
Chain Response[A...	ER_Triage		ER_Sepsis_Triage			

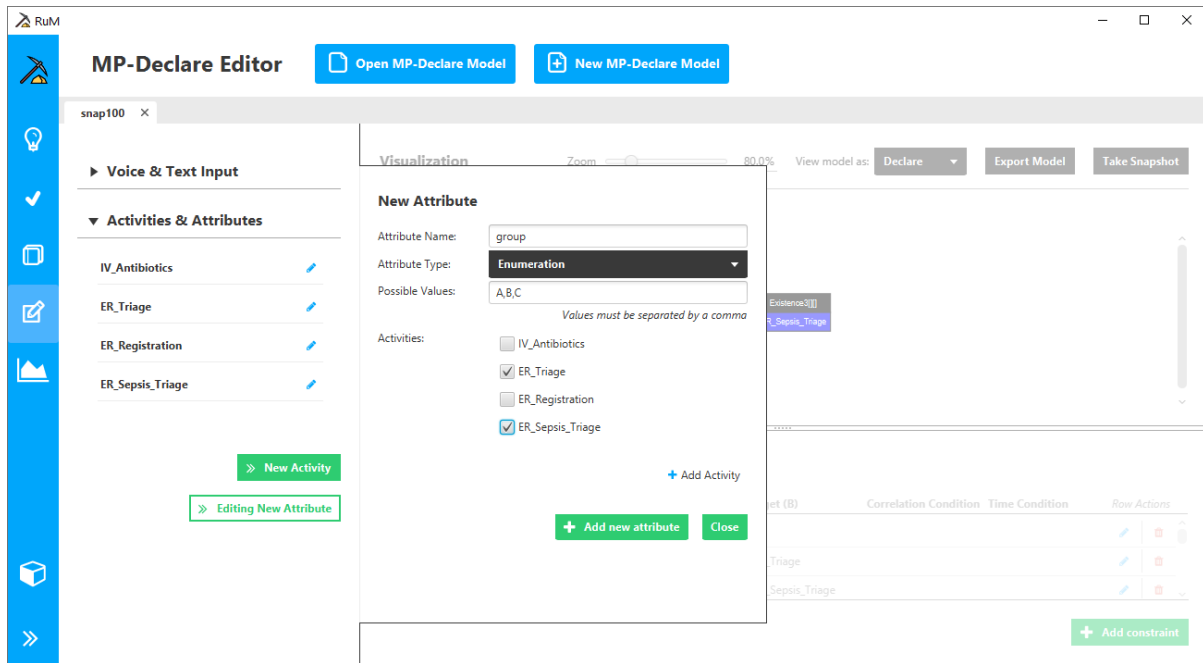
The MP-Declare Editor is divided into three areas. The panel on the left can be used to add and edit activities and attributes in the model. The top right area shows a visual representation of the model which is updated on the fly as the model is being edited and the same three views are available as in the Discovery section (Declare view, Textual view, and Automaton view). The bottom right area can be used to add and edit the constraints in the model.

Note that the visualisation in the figure above is slightly changed compared to the default one. The zoom level is set to 80% and a horizontal layout is used.

The editor is also equipped with a chatbot (Declo, presented in a different demo available https://www.youtube.com/watch?v=M_kBu9Bqso) supporting the user in designing MP-Declare models using natural language and accessible from the “*Voice & Text Input*” area in the top part of the left panel.

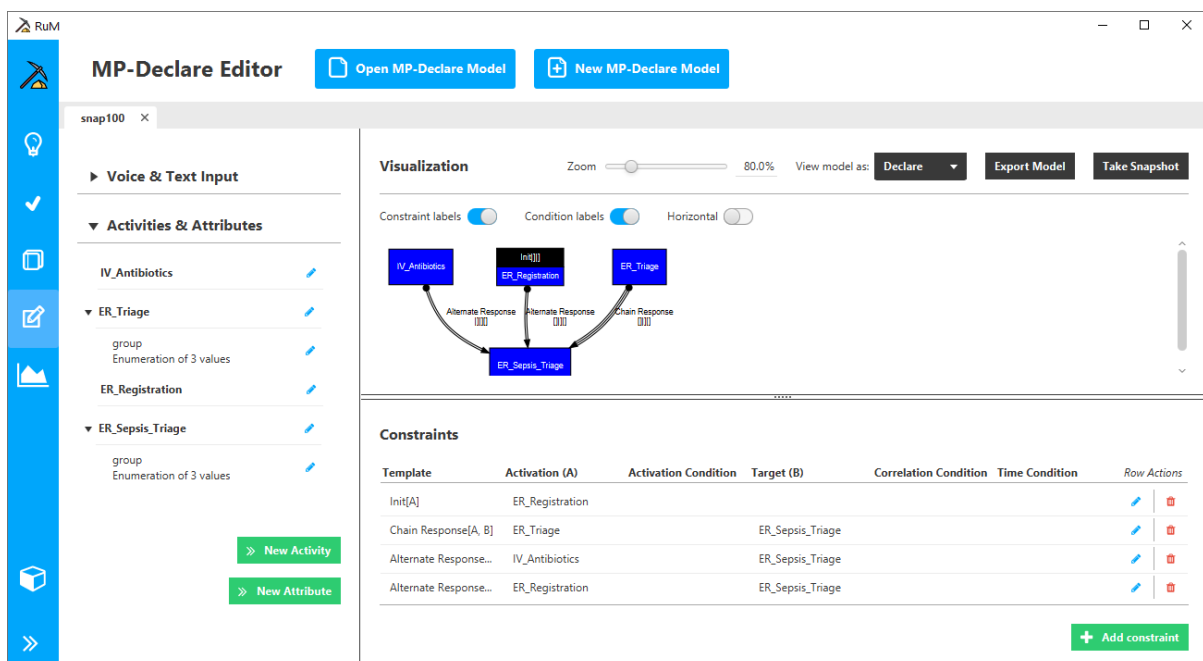
In this tutorial, a new attribute will be added to the model, some constraints will be removed from the model and an activation condition will be added to one of the constraints.

A new attribute can be added by clicking the “*New Attribute*” button. This opens a slide-in panel that can be used to enter the details of the attribute. In this case, an attribute with name “*group*”, type “*Enumeration*”, and possible values “*A,B,C*” is added. This attribute is associated to activities “*ER_Triage*” and “*ER_Sepsis_Triage*” in the model.



The new attribute will be created by clicking “*Add new attribute*” and then “*Close*”.

It is possible to remove a constraint by using the constraints table. Each constraint is displayed on a separate row and removing a constraint is done by clicking trash can icon at the end of the row that corresponds to the constraint to be removed. All constraints except the ones where template is either “*Init*”, “*Chain Response*” or “*Alternate Response*” are removed in this demonstration.



Also, an activation condition “A.group is B” is added to constraint “Chain Response [ER_Triage, ER_Sepsis_Triage]”. This MP-Declare constraint means that when “ER_Triage” occurs with attribute “group” equals to “B” then “ER_Sepsis_Triage” must immediately follow. To add an activation condition to a constraint, the row corresponding to the constraint needs to be edited either by double-clicking on the row of by clicking the pencil icon at the end of the row. This changes the row to an editing state and allows the row to be changed. In this case, the activation condition “A.group is B” will be added in the “Activation Condition” column of constraint “Chain Response [ER_Triage, ER_Sepsis_Triage]”.

The screenshot shows the MP-Declare Editor interface. On the left is a navigation menu with icons for home, search, and other functions. The main area is divided into two sections: 'Visualization' and 'Constraints'.

Visualization: Shows a Petri net diagram with nodes for 'IV_Antibiotics', 'ER_Registration', 'ER_Triage', and 'ER_Sepsis_Triage'. Edges represent constraints: 'Alternate Response' from IV_Antibiotics to ER_Sepsis_Triage, 'Alternate Response' from ER_Registration to ER_Sepsis_Triage, and 'Chain Response' from ER_Triage to ER_Sepsis_Triage. Controls include zoom (80.0%), view model as (Declare), export model, and take snapshot buttons.

Constraints: A table listing constraints with columns for Template, Activation (A), Activation Condition, Target (B), Correlation Condition, Time Condition, and Row Actions.

Template	Activation (A)	Activation Condition	Target (B)	Correlation Condition	Time Condition	Row Actions
Init[A]	ER_Registration					
Chain Respo...	ER_Triage	A.group is B	ER_Sepsis_Tri...			
Alternate Response...	IV_Antibiotics		ER_Sepsis_Triage			
Alternate Response...	ER_Registration		ER_Sepsis_Triage			

Buttons for 'New Activity' and 'New Attribute' are visible at the bottom of the left panel. An 'Add constraint' button is at the bottom right of the constraints table.

Then, the row must be saved by clicking the checkmark icon at the end of the row. By clicking the “Take Snapshot” button, it is possible to take a snapshot (“snap100_modified”) of the modifications applied to the model.

Process Monitoring

The monitoring section can be opened by clicking the monitoring button in the home screen or by using the navigation menu on the left panel of the application.

In the monitoring section, it is possible to open a model by clicking the “Open MP-Declare Model” button. In this tutorial, the “snap100_modified” snapshot is used. In this section of the tool, it is possible to replay a log on the opened MP-Declare model. The event log can be opened by clicking “Select” in the left panel of the section (parameter “Log file”). For this tutorial, the same event log used in discovery and conformance checking sections will be used. The results in this section are displayed in two columns. The first column shows the visual representation of the model and allows the user to both replay a selected trace on the model and to step through the trace one event at a time. The second column allows the user to select the trace to be replayed, to jump to a specific event in the trace and to see the payloads (attributes) of the events.

For this tutorial, the Alloy-based monitoring “MP-Declare w Alloy” will be used that allows the user to monitor data-aware constraints (“snap100_modified” contains a data condition). The monitoring can be run by clicking the “Process Traces” button. After selecting a trace in the right panel, and by clicking the play button below the model, an animation of the selected trace will start. When each event of

the trace is processed the constraints change state accordingly. The states correspond to different colours. Green indicates that the constraint is temporarily satisfied meaning that the constraint is currently satisfied but can become violated in the future. Yellow indicates that the constraint is temporarily violated meaning that the constraint is currently violated but can become satisfied in the future. Red indicates that the constraint is permanently violated meaning that the constraint is currently violated and cannot become satisfied in the future. Blue indicates that the constraint is permanently satisfied meaning that the constraint is currently satisfied and cannot become violated in the future. Orange indicates that there is a conflict between some of the constraints, meaning that there is no possible sequence of future events that could satisfy all the conflicting constraints.

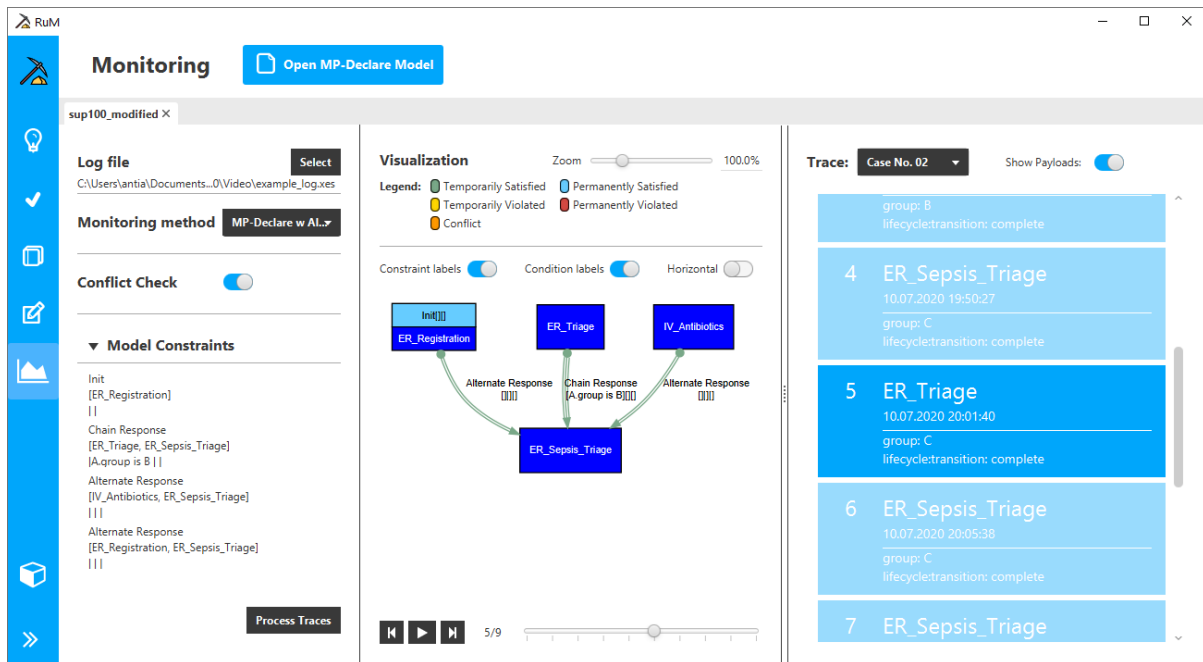
The user can also manually process the events in the trace one by one or jump to a specific event in the trace by using either the timeline or by clicking on an event in the right panel.

The screenshot displays the RuM Monitoring interface. On the left, there is a sidebar with navigation icons. The main area is divided into three panels:

- Log file:** Shows the selected log file path: `C:\Users\antia\Documents...0\Video\example_log.xes`. The monitoring method is set to "MP-Declare w AI...".
- Visualization:** Contains a legend for constraint states: Temporarily Satisfied (green), Permanently Satisfied (blue), Temporarily Violated (yellow), Permanently Violated (red), and Conflict (orange). Below the legend is a constraint network diagram with nodes: `Init()`, `ER_Registration`, `ER_Triage`, `IV_Antibiotics`, and `ER_Sepsis_Triage`. Edges represent constraints: "Alternate Response" between `ER_Registration` and `ER_Sepsis_Triage`; "Chain Response [A.group is B]" between `ER_Triage` and `ER_Sepsis_Triage`; and "Alternate Response" between `IV_Antibiotics` and `ER_Sepsis_Triage`. The diagram is zoomed to 100.0%.
- Traces:** A list of six events for "Case No. 01":
 - 1 ER_Registration (10.07.2020 13:35:03)
 - 2 IV_Antibiotics (10.07.2020 13:40:47)
 - 3 ER_Triage (10.07.2020 14:11:57)
 - 4 ER_Sepsis_Triage (10.07.2020 14:21:58)
 - 5 ER_Sepsis_Triage (10.07.2020 14:42:01)
 - 6 ER_Sepsis_Triage (10.07.2020 14:47:58)

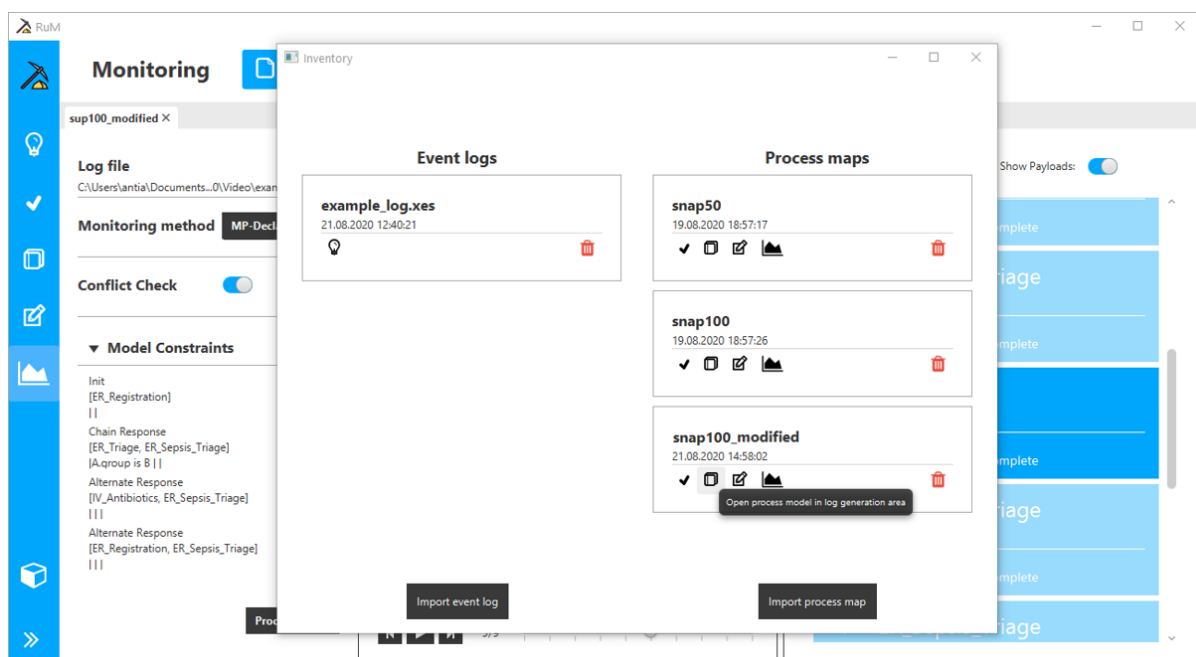
In the case under analysis, in trace number 1, the execution of event “*ER_Triage*” causes a temporary violation of constraint “*Chain Response [ER_Triage, ER_Sepsis_Triage]*”. Indeed, this event has a group value of “*B*” (that can be seen by using the “*Show Payloads*” toggle in the right panel) so that the activation condition (“*A.group is B*”) is satisfied. Hence, the constraint is currently violated but can become satisfied with the occurrence of an “*ER_Sepsis_Triage*” event immediately after “*ER_Triage*”. This is the case for this trace (in fact, all constraints become permanently satisfied at the end of the animation).

However, in trace number 2, the execution of event “*ER_Triage*” does not change the state of constraint “*Chain Response [ER_Triage, ER_Sepsis_Triage]*”. Indeed, this event has a group value of “*C*” so that the activation condition (“*A.group is B*”) is not satisfied and the constraint is not activated at all.



Inventory

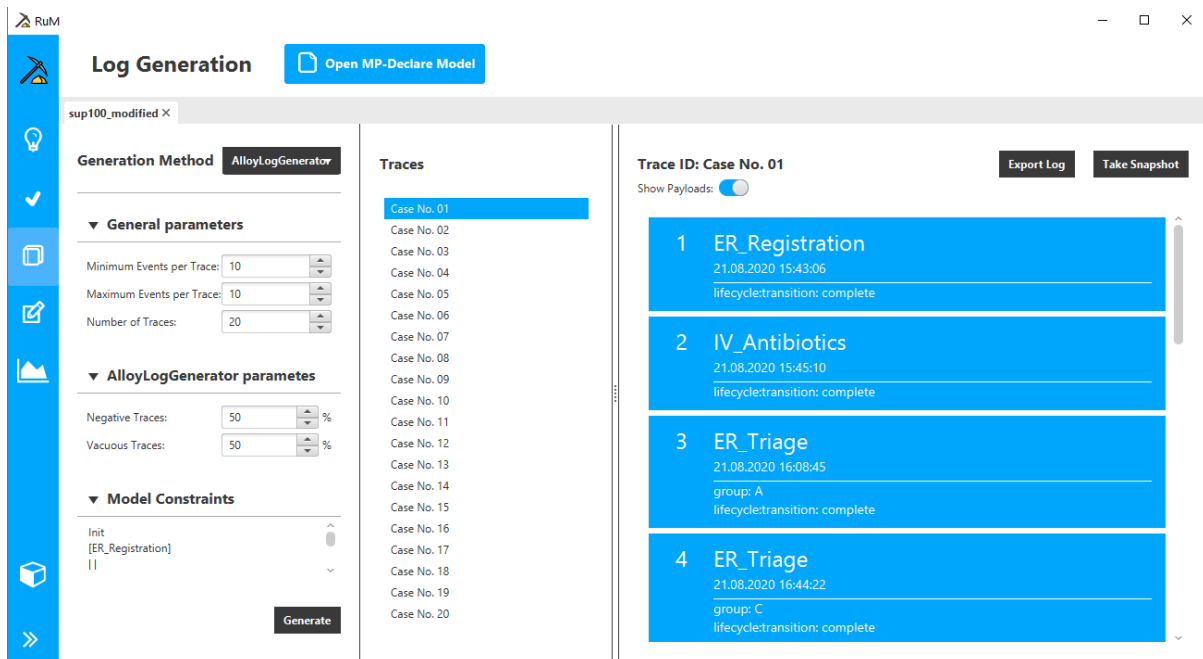
Up to this point of this tutorial, the procedure to run a task in the tool has been to first open a section and then open the input file to be used in that section. However, it is also possible to use a dedicated inventory section which can be opened from the navigation menu (*second icon from the bottom*).



This window shows all the input files used so far (*including the snapshots*). From here, it is possible to click on different icons attached to each file that automatically start the corresponding sections by using that file as input. For example, by clicking on the Log Generation icon of "snap100_modified", the Log Generation section will be opened and the model will be automatically loaded in a tab using that snapshot as input.

Log Generation

After opening a model in the log generation, to generate an artificial event log based on the constraints in the model the “*Generate*” button has to be clicked.



The screenshot displays the RuM Log Generation interface. The window title is "sup100_modified x". The main area is divided into three panels:

- Generation Method:** AlloyLogGenerator
- General parameters:**
 - Minimum Events per Trace: 10
 - Maximum Events per Trace: 10
 - Number of Traces: 20
- AlloyLogGenerator parameters:**
 - Negative Traces: 50 %
 - Vacuous Traces: 50 %
- Model Constraints:**
 - Init: [ER_Registration]
 - II

The **Traces** panel lists 20 cases, with Case No. 01 selected. The **Trace ID: Case No. 01** panel shows the following events:

- ER_Registration**
21.08.2020 15:43:06
lifecycle:transition: complete
- IV_Antibiotics**
21.08.2020 15:45:10
lifecycle:transition: complete
- ER_Triage**
21.08.2020 16:08:45
group: A
lifecycle:transition: complete
- ER_Triage**
21.08.2020 16:44:22
group: C
lifecycle:transition: complete

Buttons for "Export Log" and "Take Snapshot" are visible in the top right of the Trace ID panel. A "Show Payloads" toggle is also present.

The generated log will be displayed in two columns. The first column shows the generated traces that can be selected to show their details. The second column shows the selected trace event by event. It is also possible to show the generated attributes by using the “*Show Payloads*” toggle.

The generated model can be exported to the file system by clicking “*Export Log*”. It is also possible to take a snapshot of the log by clicking “*Take Snapshot*” and use it in other sections of the tool.