

PeT Guide

by Simone Rota

Release 0.3.2

Contents

I. Using PeT	4
1. Introduction	5
1.1. What is PeT?	5
1.2. Why PeT?	5
1.3. History	5
1.4. Authors	6
1.5. Acknowledgements	6
1.6. License	6
1.7. Notes and disclaimer	6
1.8. Legal Notices	7
2. Installation	8
2.1. Obtaining PeT	8
2.2. System requirements	8
2.3. Installation and first run	8
3. User Interface	9
3.1. Organization of the Interface	9
3.2. File Operations	10
3.3. Drawing the net	10
3.3.1. Adding and connecting components	10
3.3.2. Undo actions	11
3.3.3. Editing values	12
3.3.4. View options	13
3.3.5. Colors	13
4. Simulation	14
4.1. Types of simulation	14
4.2. Starting the simulation	14
5. Plugins	15
5.1. What is a plugin?	15
5.2. General usage	15
5.3. Algebraic Representation	15
5.3.1. Show Matrices	15

5.4. Behavioural Analysis	17
5.4.1. Boundedness	17
5.4.2. Coverability / Reachability tree	17
5.4.3. Liveness	17
5.4.4. Reversibility	17
5.4.5. Summary of properties	17
5.4.6. Merge nets	17
5.5. Classification	17
5.5.1. Free choice, State Machine, Marked Graph	17
5.6. Control	17
5.6.1. Invariants based supervisory control	17
5.6.2. Siphons Control	17
5.7. Generation	17
5.7.1. Generate from matrix	17
5.7.2. Structural Analysis	17
5.7.3. Invariants, Siphons, Traps	17
5.7.4. Reachability / Coverability Graph	17
5.7.5. Traps and siphons	18
 II. PeT Developer's guide	 19
 6. PeT Architecture	 20
6.1. Required and optional software	20
6.2. The base packages	20
6.3. The plugin architecture	20
6.4. Additional files for distribution	20
6.5. Developement ideas	20
 7. A simple plugin example	 21
7.1. Preliminary notes	21
7.2. Setting up the environment	21
7.3. Extending the Plugin Interface	21
7.4. Rendering the Configuration	21
7.5. Working on the net	21
7.6. Rendering results	21
7.7. Exporting results	21
7.8. Summing it up	21

Part I.

Using PeT

1. Introduction

WARNING: this documentation is NOT complete and could be out of date.

1.1. What is PeT?

PeT, short for *Petri Tools*, is a Free and multiplatform program for layout and analysis of Petri Nets. It consists of a visual editor to create the net layout and a set of analysis tools that can be easily extended by third party developers.

1.2. Why PeT?

There is a good number of Petri Net tools available on the Web[1], and some of them are really good. Still I wanted to create a tool with the following characteristics:

- Multi-platform
- Free and Open Source
- Good looking
- Easily extensible

Hitting the first two targets was only a matter of choosing the appropriate programming language (Java) and license (GNU GPL). For the "good looking" aspect I decided to use the SWT (Standard Widget Toolkit) by IBM/Eclipse[2], obtaining a native aspect on different Operating Systems.

Last, I hope the Plugin Architecture of PeT would facilitate the development of addons.

1.3. History

PeT is the project for the final exam of my *Laurea in Ingegneria Informatica* at Politecnico di Milano. The relator for the project is Prof. Luca Ferrarini. This program is part of a collection of educational tools and studies related to the exams of "Automazione Industriale" and "Modellistica e controllo dei sistemi discreti" at Politecnico di Milano.

The idea of PeT came from a previous project for the *Automazione Industriale* exam with Prof. Ferrarini. The project, named poliJARP, consisted in some extension to the JARP tool[4].

1.4. Authors

- Simone Rota <sip@varlock.com> is the main author of PeT, the default plugins and the Documentation (including this file).
- Portions of code adapted from JARP[4] by Ricardo Sangoi Padilha, see the source files for details.
- Default Analysis plugins adapted from previous work by Francesco Cartella, Simone Civati, Giovanni Caria.
- Reachability and Liveness plugins adapted from Giuseppe Rizzo's work.

1.5. Acknowledgements

I would like to thanks the following persons, which greatly contributed to PeT with ideas, corrections and support:

- Prof. Luca Ferrarini - relator of the project.
- Ing. Adamo Castelnuevo, Ing. Carlo Veber - co-relators, testing.
- *** students

1.6. License

PeT is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

See the LICENSE.txt file included in the source and binary package for the complete license text.

1.7. Notes and disclaimer

I'm not a professional developer, I cannot guarantee on the quality of the code contained in PeT. Corrections, suggestions and contributions are welcome. Furthermore, as stated in the license file:

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH

YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

1.8. Legal Notices

Linux is a Registered Trademark of Linus Torvalds; Java is a registered trademark of Sun Microsystems, Inc. Microsoft, Windows, Windows 95, Windows 98, Windows ME, Windows NT, Windows 2000, and Windows XP are registered trademarks of Microsoft, Inc.; FreeBSD is a registered trademark of Walnut Creek CDROM, Inc.; Mac OS is a registered trademark of Apple Computer, Inc.

All trademarks are property of their respective owners.

2. Installation

2.1. Obtaining PeT

At the moment I'm writing there is not an official website for the software, The files are hosted at the author's personal site <http://www.varlock.com>. Binaries for Windows and Linux and the sources are available.

2.2. System requirements

The software requirements for running PeT are essentially two:

- TODO

2.3. Installation and first run

There's really not much to do regarding the installation process: PeT files are self-contained in the distribution archives.

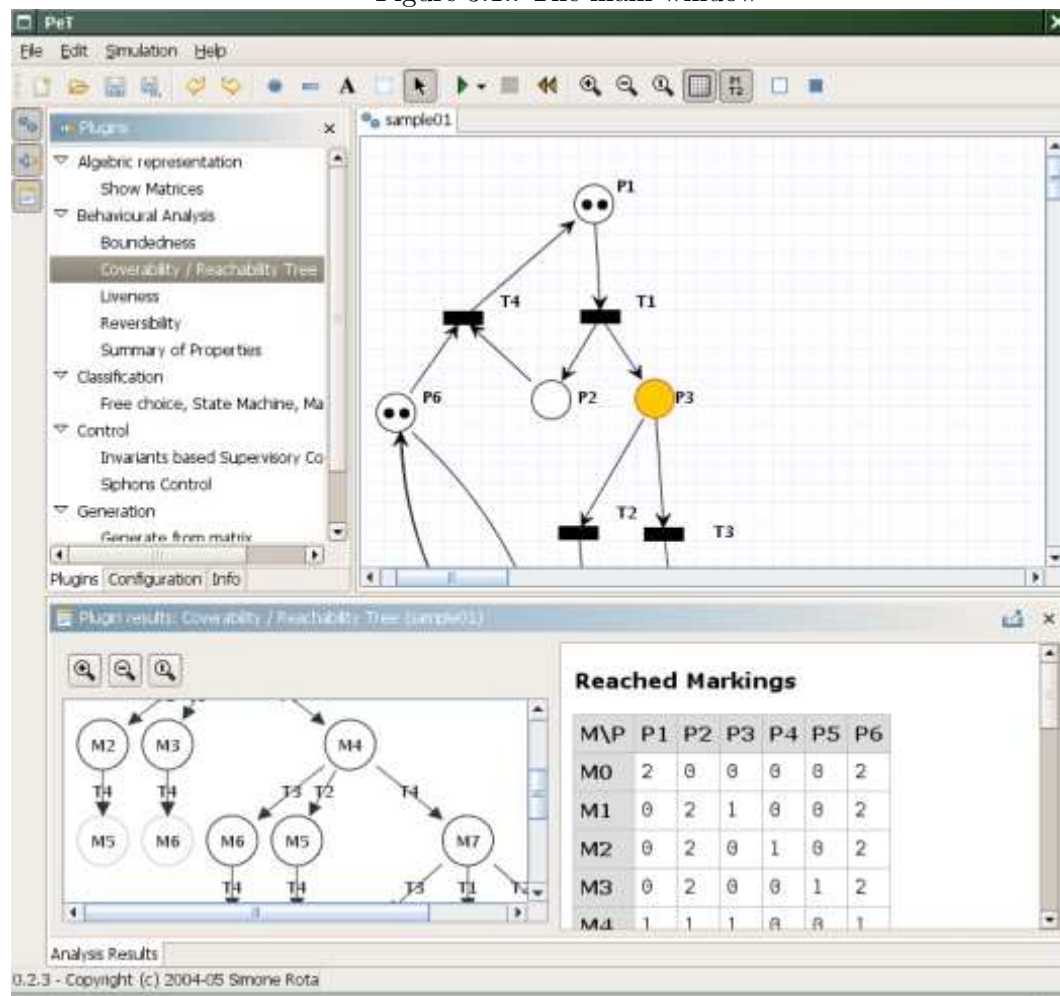
Generally you'll want to extract the archive to a folder on your hard drive, then run the PeT.exe file (on Windows) or PeT.sh (on Linux / other platforms).

On Linux / other you'll most likely have to edit the PeT.sh file to adjust the location of your Java installation.




3. User Interface

3.1. Organization of the Interface

Figure 3.1.: The main window







Apart from the "traditional" Menu and Toolbar elements (we'll discuss them in the following sections), it must be noted that PeT's window is divided in 3 main areas that can be shown or hidden using the toolbar on the left of the window:

-  *Net*: the area containing the drawing of the Petri Net
-  *Plugins*: a categorized tree containing the available Plugins
-  *Results*: this area will contain the results of the plugin launch.

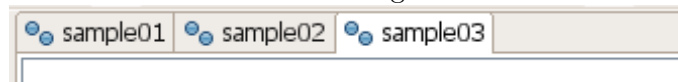
3.2. File Operations

The *File* menu (or in alternative the first set of buttons on the main toolbar) contains the usual file operations:

-  *New*: create a new file
-  *Open*: open a file
-  *Save*: save the current file
-  *Save as*: save the current file with the given filename

PeT's interface can handle opening multiple documents, which are accessible from the tabs at the top of the Net Area:

Figure 3.2.: The documents tab



PeT can also export the current net as an image (available formats: PNG, GIF and EPS); this function is available from the *File -> Export* menu.

The supported file format for saving and loading Petri Nets is PNML[7], a proposed standard for Petri Nets, based on XML[8].

3.3. Drawing the net

3.3.1. Adding and connecting components

After creating a new file, in order to add Places, Transitions and comments to the current net click on the toolbar buttons or on their equivalent on the *File -> Insert* menu and click on the point of the net you want the element to be inserted.






-  *Place*: insert a place
-  *Transition*: insert a transition
-  *Comment*: insert a comment
-  *Area*: insert a dotted-area (useful to highlight group of items)
-  *Select*: normal behaviour, for selecting already placed items

Figure 3.3.: Connecting two components

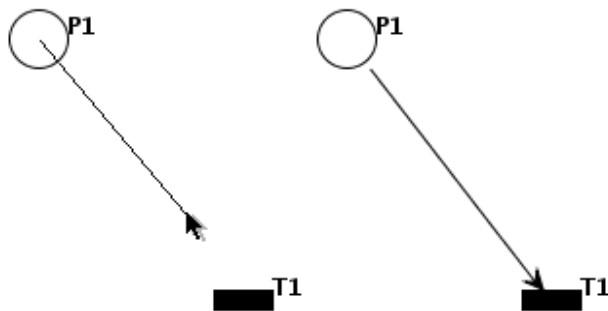
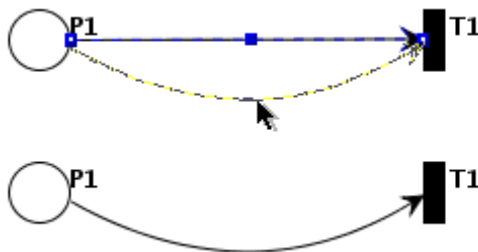


Figure 3.4.: Adding a point to an arc



Once you placed two or more components in the Net Area, you can drag them around with the mouse; by pressing the **SHIFT** key the movement is orthogonal and only one direction at a time is allowed (North-South or East/West).

Some components (i.e. the dotted areas) can be resized dragging the *handles* around them.

To draw an arc from a place to a transition (or vice-versa), right-click with the mouse on the source of the arc and drag the mouse to the target component. (Fig. 3.3).

Right-click on an arc and drag the mouse to add a point to its path, as in Fig.3.4 (useful for complex paths). Right-clicking on one of the points of the path will remove the point. The default path style is a Bezier line; you can easily change the line style to orthogonal by middle-clicking anywhere on the arc or by clicking on the arc while holding the **SHIFT** key. In Fig. 3.5 the two styles are compared.

3.3.2. Undo actions

Another set of well known commands is accessible from the *Edit* menu:

Figure 3.5.: Path Style (Orthogonal - Bezier)

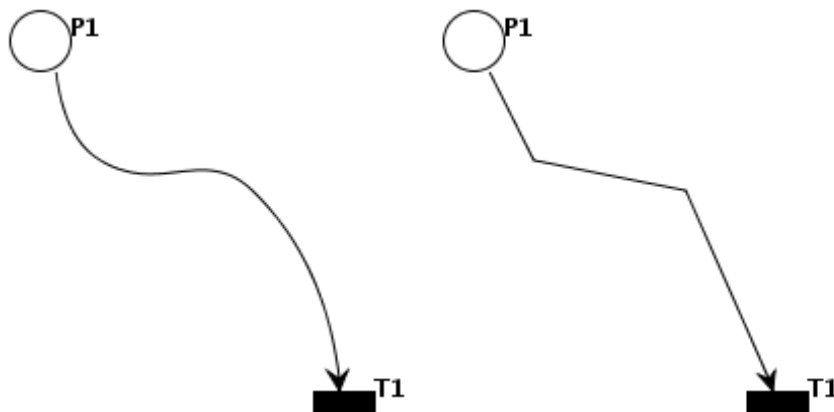
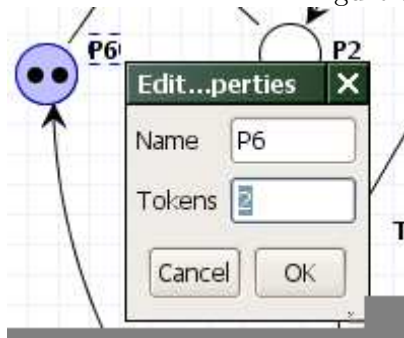


Figure 3.6.: Changing the token number



Undo: undo last action



Redo: redo last action



Cut: cut the seected components



Copy: copy the selected components into the clipboard



Paste: paste the content of the clipboard onto the current Net



Delete: delete the selected components

3.3.3. Editing values

To edit a Place/Transition name or a comment, just double click on the element, an input box will appear in which you can modify its properties, i.e. the number of tokens in a place (see Fig.3.6) or the weight of an arc.

When changing the properties of an arc, a "Create self-loop" button is available in the property editor; can be used to create a self-loop (automatic generation of an arc with

source and target inverted respect to the currently selected arc).

3.3.4. View options

While editing the net, the following additional tools are available in the main toolbar:



Zoom In: zoom in the net



Zoom Out: zoom out the net



Zoom Reset: Reset the zoom value to its original size



View Grid: show/hide the background grid



View labels: show/hide the places / transitions names

3.3.5. Colors

Border and background color for the items is customizable by selecting one or more elements and clicking on the following toolbar buttons:



Border: set the border color



Background: set the background (fill) color

4. Simulation

4.1. Types of simulation

With *Simulation* we intend letting the Petri Net evolve, keeping trace of the reached states and fired transitions. We can make the net evolve automatically (randomly firing active transitions) or manually choose the transition we want to fire clicking on it with the mouse.

The commands in the *Simulation* menu, also available from the toolbar, are the following:



Start: start the simulation. The automatic simulation is the default one.



Stop: stops the simulation.



Reset: bring the net to the initial state

4.2. Starting the simulation

When choosing the automatic simulation, a prompt dialog asks for the interval (in milliseconds) between the firing of two transitions. The default value is 500.

When the simulation ends (or the user presses the *Stop* button), a list of reached markings is shown in the Results Area. Click on a marking to assign it to the net.

The top-right button in the *Results Area* brings up a menu containing export functions (Matlab format).

5. Plugins

5.1. What is a plugin?

A plugin is a piece of code that can be "plugged" into PeT, making an additional set of functions available to the program.

A plugin could do almost everything with the current opened Petri Net(s): analyze its structure, make it evolve, etc. We included a good number of plugins covering the main aspects of Petri Net analysis, hoping to raise some interest into writing others. You can found more information on writing plugins in Part II of this book.

5.2. General usage

The *Plugin Area*, as described in Section 3.1, contains a tree of available plugins, divided into categories. Each plugin provides a quick online description page that can be activated by clicking on the "Info" tab at the bottom of the plugin area, as shown on Fig. 5.1.

To activate a plugin, double click on its name, the "Options" tab will appear to let the user configure the options¹ for the selected plugin. (Fig. 5.2)

Once the options are configured, press the start button to launch the plugin on the currently active net. When the plugin terminates its execution, the Results Area will be shown (if not already visible) with significant data acquired during plugin execution. (Fig. 5.3)

Many plugins provide an export function to permanently save the results to a file (i.e. Matlab); this function can be accessed with the "Export" button located at the top-right corner of the Results Area.

The following paragraphs contain a brief description of the plugins shipped with the current version of Pet.

5.3. Algebraic Representation

5.3.1. Show Matrices

This is a simple plugin (also used as developement example in Chapter 7) that displays the

¹Not every plugin needs options; plugins that does not require configuration will simply show a blank configuration area.

Figure 5.1.: Online Help

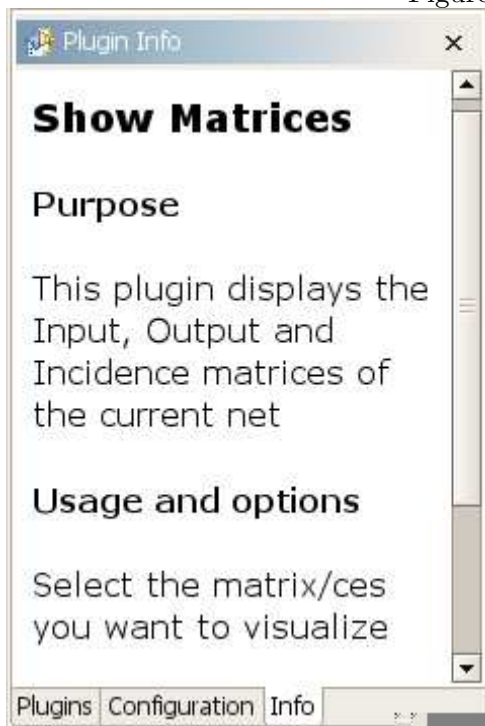


Figure 5.2.: Online Help

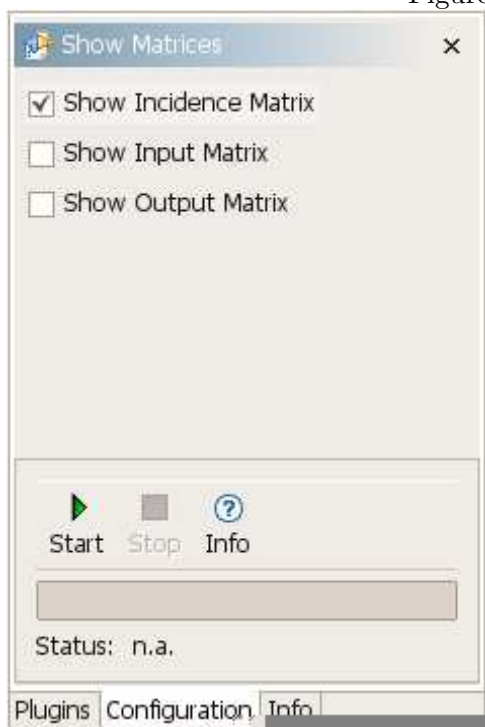
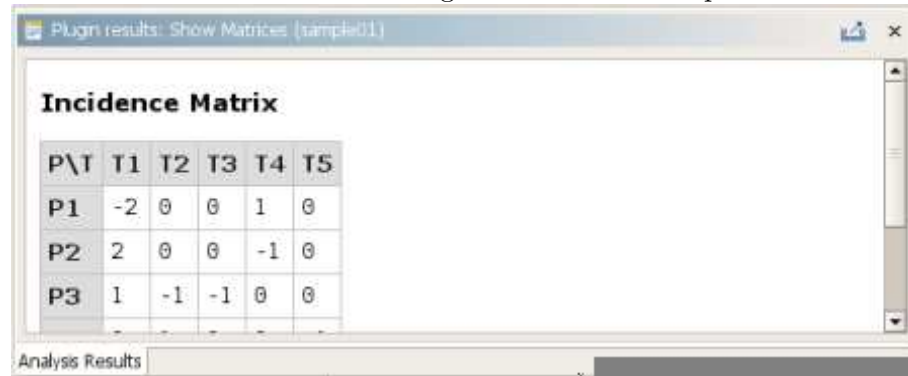


Figure 5.3.: Online Help



Incidence Matrix

P\T	T1	T2	T3	T4	T5
P1	-2	0	0	1	0
P2	2	0	0	-1	0
P3	1	-1	-1	0	0

Analysis Results

5.4. Behavioural Analysis

5.4.1. Boundedness

5.4.2. Coverability / Reachability tree

5.4.3. Liveness

5.4.4. Reversibility

5.4.5. Summary of properties

5.4.6. Merge nets

5.5. Classification

5.5.1. Free choice, State Machine, Marked Graph

5.6. Control

5.6.1. Invariants based supervisory control

5.6.2. Siphons Control

5.7. Generation

5.7.1. Generate from matrix

5.7.2. Structural Analysis

5.7.3. Invariants, Siphons, Traps

5.7.4. Reachability / Coverability Graph

wefrwer uwekrlwerjelkrkewr wre werlrewkrwe ewl

5.7.5. Traps and siphons

Part II.

PeT Developer's guide

6. PeT Architecture

6.1. Required and optional software

Here's a brief list of programs and libraries required to work on PeT and PeT plugins.

- A Java Development Kit, we suggest using the latest release available from Sun[6]. Required.
- The PeT sources archive; always be sure to use the latest version. Required.
- Apache Ant[9]. Suggested for plugin development, required for PeT development.
- ROXES Ant Tasks. Required for PeT development.
- L^AT_EX, required if you intend to modify PeT Documentation (i.e., this Guide). Note that only the Linux version has been tested so far.
- IBM's Eclipse[2]: suggested for general Java development.

6.2. The base packages

PeT is organized

6.3. The plugin architecture

How plugins work

6.4. Additional files for distribution

Ant, docs

6.5. Development ideas

Parts that need work

7. A simple plugin example

7.1. Preliminary notes

Interfaces not definitive, notes on threads

7.2. Setting up the environment

Brief notes on setting up Eclipse

7.3. Extending the Plugin Interface

Interface, Loader

7.4. Rendering the Configuration

Render config

7.5. Working on the net

run() method

7.6. Rendering results

Results

7.7. Exporting results

Exporting

7.8. Summing it up

Complete code

Bibliography

- [1] Petri Net Worlds - <http://www.ppp.com>
- [2] The Eclipse Project - <http://www.eclipse.org>
- [3] Politecnico di Milano - <http://www.polimi.it>
- [4] JARP - <http://jarp.sourceforge.net>
- [5] Simone Rota's personal site - <http://www.varlock.com>
- [6] Sun's Java: <http://java.sun.com>
- [7] <http://www.informatik.hu-berlin.de/top/pnml/about.html>
- [8] <http://www.w3.org/XML/>
- [9] <http://ant.apache.org>
- [10] <http://www.roxes.com/produkte/rat.html>
- [11] <http://www.lyx.org>