



## **Nuclear Py**

Dini, A.  
García Blesa, M.

February 15, 2019

### **Abstract**

nPy is a configurable graphical interface (GUI) written in Python 2.7 and PyQt4. Has been built as a new graphical platform for an easily running applications used in nuclear technology. Usually those programmes runs from a terminal and requires very specific and long string parameters. nPy enables the addition of any program or command into its menu and launch it subsequently, capturing the standard output produced and showing it at the main screen. A special Sequence Table tool allows the user to edit any list of system process –one after another– with or without parameters, edditing at the same time the main graphic menu and run that list with a simple button click.

# Contents

<b>1. Introduction</b>	<b>3</b>
1.1. nPy Elements . . . . .	3
1.2. Process . . . . .	4
1.3. Text Editor . . . . .	4
1.4. Application . . . . .	4
<b>2. Install Guide</b>	<b>5</b>
2.1. Generals . . . . .	5
2.2. On Linux system . . . . .	5
2.3. On Windows System . . . . .	5
2.4. Pip (Python Package Index) . . . . .	6
<b>3. Process Tab</b>	<b>6</b>
<b>4. Running nPy</b>	<b>6</b>
4.1. on Linux . . . . .	6
4.2. Running on Windows . . . . .	7
<b>5. Sequence Table</b>	<b>7</b>
5.1. nPy Sequence Table (ST) . . . . .	7
5.2. Sequence table: Properties . . . . .	7
5.3. Command String Syntax . . . . .	7
5.4. Create a Sequence of Processes . . . . .	10
5.5. Setting and ordering the main menu . . . . .	11
5.5.1. A Simple Example . . . . .	12
5.6. ST and the Main Menu . . . . .	12
5.7. A single execution process . . . . .	12
5.8. The Standard Output . . . . .	12
5.9. Customizing . . . . .	13
5.10. Customizing the ST . . . . .	13
5.11. Customizing directories . . . . .	13
<b>6. Windows Build</b>	<b>14</b>
6.1. Install wine, and Python under wine. . . . .	14
6.2. Install PyInstaller on wine . . . . .	14
6.3. Install PyQt4 on wine . . . . .	14
6.4. Build.exe . . . . .	15

# 1. Introduction

nPy is a configurable graphic interface written in Python 2.7 and PyQt4 that enables the addition of any program or command into its menu and launch it subsequently, capturing and showing the standard output produced in the main screen.

The aim of the project is to build a graphical environment to design any complex sequence of processes, and launch it as a list, enabling the change of any parameter for each system command or program.

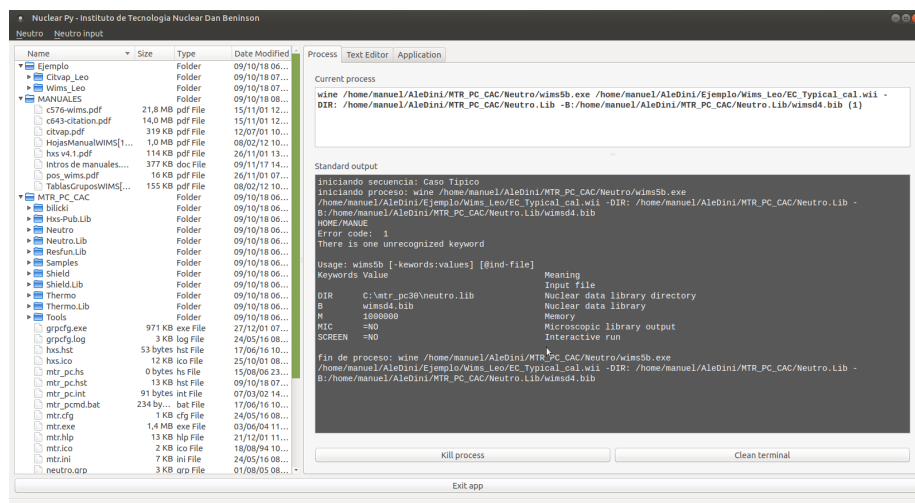
nPy provides a graphical solution. A dynamic table application will allow the user to create any list of processes, defining its order and parameters, enabling the edition and deletion of rows as needed.

It is possible to launch nPy in almost every operating system. It has been proved in Windows 8 and windows 10, and linux Ubuntu 18.04, linux Mint 17.2, linux Debian 8.x, even Raspbian for Raspberry Pi and others.

## 1.1. nPy Elements

At the beginning nPy presents a main window divided into two sections: a tree view and a tab section.

The tree view is a conventional application. It allows the user to navigate through the directories, and open a text file in the embedded text editor application.



The tab section includes three applications:

- Process
- Text editor
- Configuration

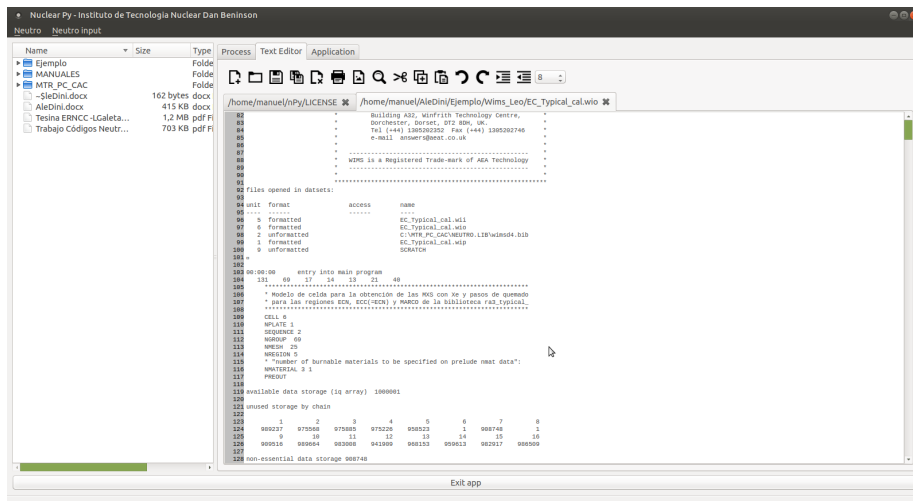
## 1.2. Process

This tab shows two fields configured as outputs. The Current Process indicates which process of the list is running. The Standard Output shows the process in real time as shown in a regular system terminal.

It is possible to stop any process at any time and continue with the next in the list using the Kill process button, as well as to clean the terminal to start a new list of process by simply pressing the Clean terminal button.

## 1.3. Text Editor

nPy was created to provide a flexible environment to execute complex command lists with any parameters.



In order to provide such environment we include a simple but powerful text editor. Its purpose is to work as an assistant to view and edit the in-files and out-files in every process defined. This application is not a regular text editor; it was conceived as a small programming application with a highlighter font.

This embedded text editor supports python syntax highlighting and multiple tabs, so you can edit your files without needing anything else.

## 1.4. Application

The third tab is a special section which, in turn, presents two tabs: Sequence Table and Environment.

The first of these two tabs, the Sequence Table, shows the sequence table application which helps the user to set the list of processes.

The second tab, Environment, allows the user to change the root path and the initial path required to set the initial position in the tree view application.

Pages 4

- Home
- Process Tab

- Sequence Table User manual
- Text-Code editor

## 2. Install Guide

### 2.1. Generals

You can download **nPy** from github:

<https://github.com/alete89/nPy/releases>

nPy runs over Windows and Linux operative systems (OS). To complete the installation is needed to follow instructions depending on the OS in use. nPy needs Python 2.7 and PyQt4 framework to run. All those are free software and can be downloaded from every respective official site.

Last Python release:

<https://www.python.org/download/releases/2.7/>

Last PyQt4 release:

<https://www.riverbankcomputing.com/software/pyqt/download>

### 2.2. On Linux system

Almost every linux system comes with python 2.7 preinstalled. In other case you can install it using a system terminal:

```
apt-get install python2.7
```

PyQT4 usually doesn't comes preinstalled so we must to install. We can use the same way before:

```
apt-get install python-pyqt4
```

A compact way to install both at the same time:

```
apt-get install python2.7 python-pyqt4
```

### 2.3. On Windows System

Usually windows doesn't comes with python preinstalled. So you need to download it from the official site and run the simple install.

The same occurs with the PyQt4 libraries. You can download it from

<https://riverbankcomputing.com/software/pyqt/download>

and run the windows installer.

## 2.4. Pip (Python Package Index)

Optionally pip to install the packages. Pip is a dedicated python package system to get python libraries and others you can use to install all python's needs to run nPy. It is simple to use. Just run in a system terminal as follows:

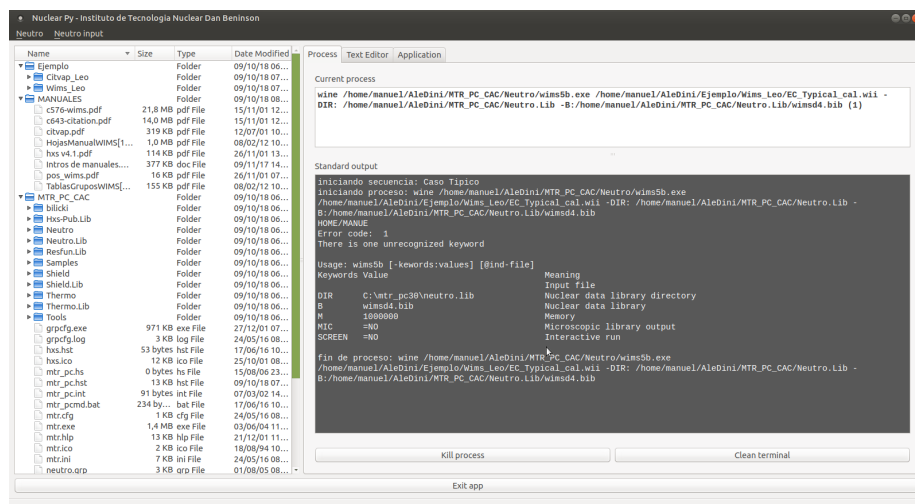
```
pip install python2.7 pyqt4
```

Under Windows systems, pip wont find PyQt4 to install, .whl are provided here:

<https://www.lfd.uci.edu/~gohlke/pythonlibs/#pyqt4>

## 3. Process Tab

The first screen that you see when you run nPy is the Process tab.



A side you have a TreeView that persist through all the screens, it allows to browse your file system. The main tab features two outputs. Current process (experimental) shows the sequence of commands as they're being executed. Standar output shows exactly that, of every command executed by nPy.

If eventually, a command running were to hang, you can press Kill process in order to terminate the current command in execution.

Clean terminal button just clears the Standard output box.

## 4. Running nPy

### 4.1. on Linux

Unpack the nPy.tar. Open a terminal, place in the nPy directory and simply run

```
python nPy.py
```

## 4.2. Running on Windows

Just execute the nPy.exe

## 5. Sequence Table

### 5.1. nPy Sequence Table (ST)

The ST represents the core of the nPy application. The table structure is a simple way to store information to configure system processes (it means is possible to put in order any process that you can run in a terminal) as well as to configure information to create a graphical menu.

The purpose of its editing table is to solve three different problems: first one is to create a complex process sequence to run in order, with no maximum parameters defined. Second one is to create a graphical options in the main menu, and the third one is to do it friendly and a simple way.

The aim of it is to configure different processes needed, including any parameters for each one, and run all those process one after another, in a single click. The correct configuration will create a button in the main menu, and will create context menu for each parameter setted in the table.

### 5.2. Sequence table: Properties

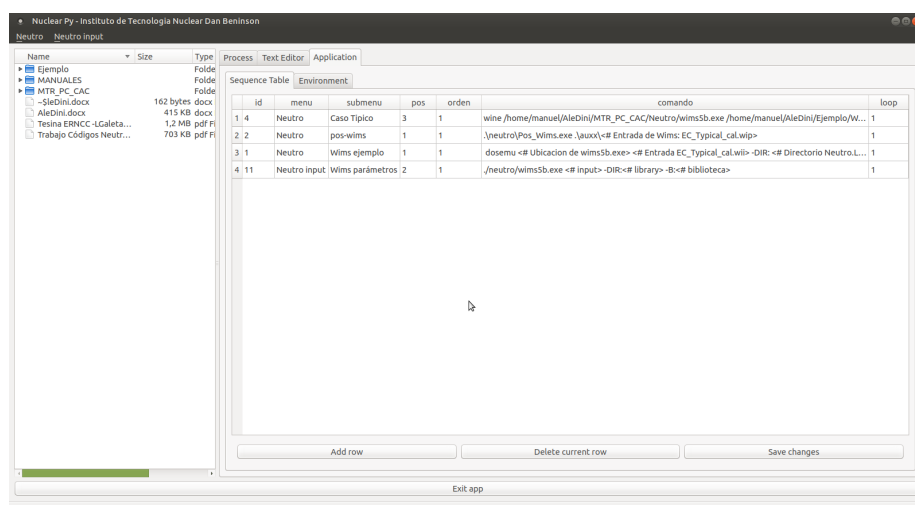
This table has been designed with 7 columns:

- **id**: simple auto numeric id
- **Menu**: is the first level name menu, is a string value
- **Submenu**: is the second level name menu (sub menu), is a string value
- **MenuPosition**: indicates the sub menu position, is a integer value
- **Order**: indicates the order of this process in the list, is a integer value
- **Command**: the system command will run, is a string value
- **Loop**: to set how many times to run a process before to the next in the list, is an integer value

In the ST application, one process must be defined in one row. So, we define a list of process in ST. We number it, and execute in that ascending order.

### 5.3. Command String Syntax

As simple as it looks, is possible to fill every gap in the command column. You can run a simple system command filling the cell. You can also run a complex command using parameters, exactly the same as you do in a regular system terminal.



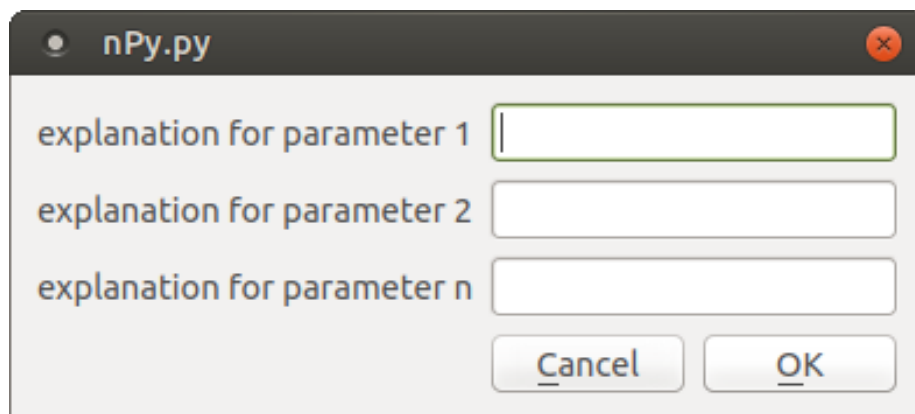
Is possible to create a context menu to set any different parameters on the fly for any command. To set a correct row in the ST is needed to know a simple syntax rule to define the string in the command column to create properly an a useful context menu. A typical running command presents a very simple structure in a system terminal. Usually we write commands in it as follows:

```
user@I7-system ~ $ program [parameter 1] [parameter 2]
... [parameter n]
```

In the **Command** field –in our ST– the syntax is very similar with a small difference: write <> instead [], and describe the parameter between them. Finally the command must be written as follows:

```
program <explanation for parameter 1> <explanation for parameter 2>
...
<explanation for parameter n>
```

The string between the <> will be shown in the graphic context menu as a short message, in a built dynamic form:



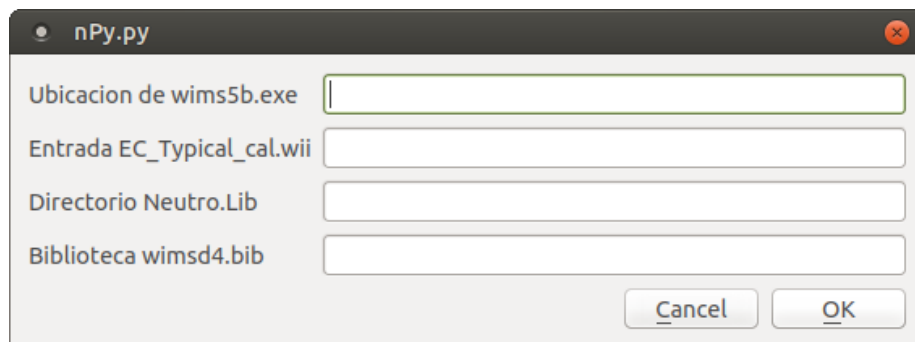


After those settings, nPy will be ready to launch lists of programs, commands, scripts, and so on. Anything you can launch from a terminal.

As an example you can write on the ST:

```
dosemu <Ubicacion de wims5b.exe> <Entrada EC_Typical_cal.wii>  
-DIR: <Directorio Neutro.Lib> -B:<Biblioteca wimsd4.bib>
```

and nPy will show the next context menu:

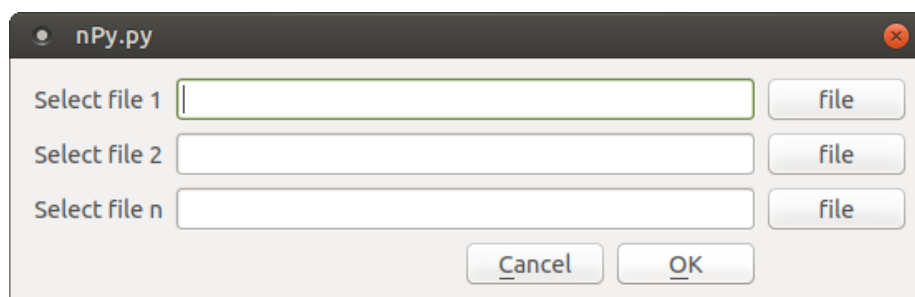


The screenshot shows a window titled "nPy.py" with four input fields and two buttons. The fields are labeled: "Ubicacion de wims5b.exe", "Entrada EC\_Typical\_cal.wii", "Directorio Neutro.Lib", and "Biblioteca wimsd4.bib". The "Cancel" button has an underline on the 'C' and the "OK" button has an underline on the 'O'.

Often is needed to pick a file (usually the path of a file) to select as a parameter. This is possible to set in nPy to get a graphic menu for a selection file. Simply write # as the first character after <, as follows:

```
program <# Select file 1> <# Select file 2> ... <# Select file n>
```

On this way nPy will create a dynamic menu to select a file into the system for each < #... > found on the string.



The screenshot shows a window titled "nPy.py" with three rows of file selection. Each row has a label "Select file 1", "Select file 2", and "Select file n" followed by an input field and a "file" button. At the bottom are "Cancel" and "OK" buttons with underlines on the 'C' and 'O' respectively.

An example:

```
dosemu <# Ubicacion de wims5b.exe> <# Entrada EC_Typical_cal.wii>  
-DIR: <# Directorio Neutro.Lib> -B:<# Biblioteca wimsd4.bib>
```

Is possible to combine both techniques as is needed:

```
program <# Select file 1> <explanation for parameter 1>
```

An example:

```
dosemu <Ubicacion de wims5b.exe> <# Entrada EC_Typical_cal.wii>
-Dir: <# Directorio Neutro.Lib> -B:<Biblioteca wimsd4.bib>
```

## 5.4. Create a Sequence of Processes

Knowing how to write a single process, will create a sequence putting those in order. An example will be any good to start an explanation.

Sequence Table		Environment							
	id	Menu	submenu	MenuPosition	Order	Command			loop
1	5	Heat Transfer	CANDU Fuel	3	1	relap <# Input geometry file> <Material Code>			1
2	1	Neutro	Reactor Calc	1	1	wims5b.exe <# Entrada EC_Typical_cal.wii> -Dir: <# Directorio Neutro.Lib> -B:<Biblioteca ...			1
3	2	Neutro	Reactor Calc	1	2	hxs <# Entrada HXS.wii> -Dir: <# Directorio CrossSecc.Lib> -B:<Biblioteca crossSec.bib>			1
4	3	Neutro	Reactor Calc	1	3	cltvap <Output name: > -Dir:<# Library directory> -B:<# Library>			1
5	4	Shield	Steel Wall	2	1	shield <# Input geometry>			1

In the following configuration has been defined three processes: two single processes (**CANDU Fuel** and **Steel Wall**) and one sequence of processes (**Reactor Calc**).

Is simple as define the “Order” as follows in the same place in the menu:

Sequence Table		Environment							
id	Menu	submenu	MenuPosition	Order	Command	loop			
1 5	Heat Transfer	CANDU Fuel	3	1	relap <# Input geometry file> <Material Code>	1			
2 1	Neutro	Reactor Calc	1	1	wimsSb.exe <# Entrada EC_Typical_cal.wii> -DIR: <# Directorio Neutro.Lib> -B:<Biblioteca ...	1			
3 2	Neutro	Reactor Calc	1	2	hxs <# Entrada HXS.wii> -DIR: <# Directorio CrossSecc.Lib> -B:<Biblioteca crossSec.bib>	1			
4 3	Neutro	Reactor Calc	1	3	citvap <Output name: > -DIR:<# Library directory> -B:<# Library>	1			
5 4	Shield	Steel Wall	2	1	shield <# Input geometry>	1			

nPy will create a context menu with all options setted at ST.

The screenshot shows the 'nPy.py' application window. It contains a configuration form with the following fields and buttons:

- Entrada EC\_Typical\_cal.wii**: Text input field with a 'file' button to its right.
- Directorio Neutro.Lib**: Text input field with a 'file' button to its right.
- Biblioteca wimsd4.bib**: Text input field.
- Entrada HXS.wii**: Text input field with a 'file' button to its right.
- Directorio CrossSecc.Lib**: Text input field with a 'file' button to its right.
- Biblioteca crossSec.bib**: Text input field.
- Output name:**: Text input field.
- Library directory**: Text input field with a 'file' button to its right.
- Library**: Text input field with a 'file' button to its right.
- Buttons**: 'Cancel' and 'OK' buttons at the bottom.

## 5.5. Setting and ordering the main menu

Is possible to configure the main menu arbitrarily filling correctly the columns menu, submenu, submenu position and sequence order.

If we consider a menu and its submenus as a set and subsets system, we can order the information as levels. nPy has been create to configure until a second level (as a third level we have the context menu).

The menu column is a string type value, and will show the text at the front of the main menu. This is the first level.

A second level is given by the values submenu and the submenu position: submenu is a string type value and its button will be shown inside the first level. Its corresponding submenu position indicates the position in that menu and it is marked with a integer value.

### 5.5.1. A Simple Example

The next example will create two buttons in the main menu: Draw program and Example Menu. Inside the Example Menu option, will be listed three buttons: Sequence test, Python parameters test and Python GPS.

Inside the Sequence test submenu nPy will run three processes ordered by submenu position.

```
id;menu;submenu;posicion en menu;orden en secuencia;comando;loop
1;Draw program;gimp;1;1;gimp;1 2;Example Menu;Sequence test;1;
2;python /home/alete/git/PyCGI/scripts/simple_print.py;1
3;Example Menu;Sequence test;2;3;python ./scripts/print_loop.py;1
4;Example Menu;Sequence test;3;4;python ./scripts/donde.py;1
5;Example Menu;Python parameters test;2;1;python ./scripts/
parameters.py <un param><otro param>;1
6;Example Menu;Python GPS;4;1;python ../../Desktop/donde.py;1
```

Finally, there is no limit defined to add files to the ST.

## 5.6. ST and the Main Menu

When a process is setted in the ST, one button is added in the main menu. If the command setted in a row is a simple system command or program, nPy will launch immediately and will show the standard output. If the command has been setted with parameters, nPy will create a context menu before launching.

At the process tab, nPy shows two fields: Current Process and Standard Output. The first one shows the current process in the list setted in ST. The second one is the standard output for each process in the list.

Is possible to stop any process and to continue with the next, using the kill process button.

## 5.7. A single execution process

A list of processes will be executed in order, one after another. It means something important: nPy can't launch more than one process setted in the ST at a time. This point has been a design decision to guarantee nPy -as a launcher program- does not modify or interfere anything on a running current process.

A single execution program will create a simple button with no context menu. It means the sequence will start immediately showing the output.

## 5.8. The Standard Output

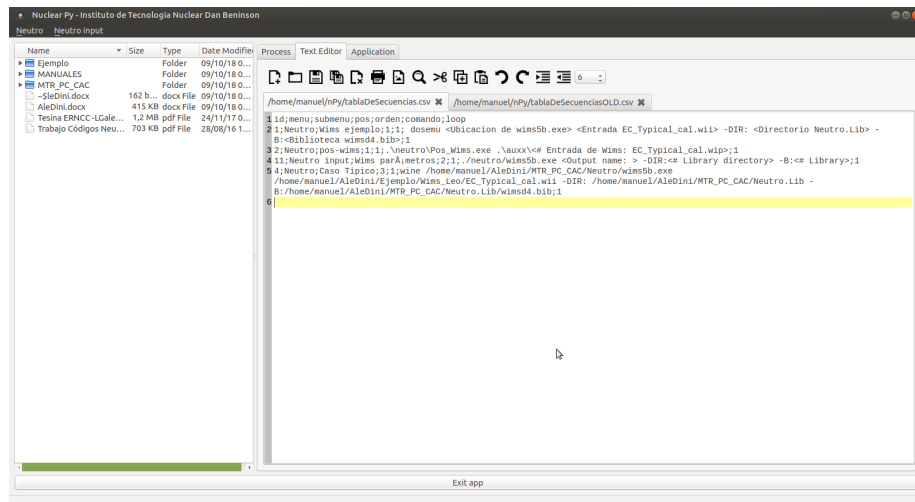
nPy will show the standard output for the current process, which one is running. When a process listed ends and continue with the next, nPy will notice to the user and will start to send the new current process to the standard output.

Is possible to catch all the output in the field because is a text field. So, is possible to copy and edit using the text editor. Many times is necessary to know the complete output. nPy provides a simple solution to capture all the outputs in sequence, one after another.

## 5.9. Customizing

## 5.10. Customizing the ST

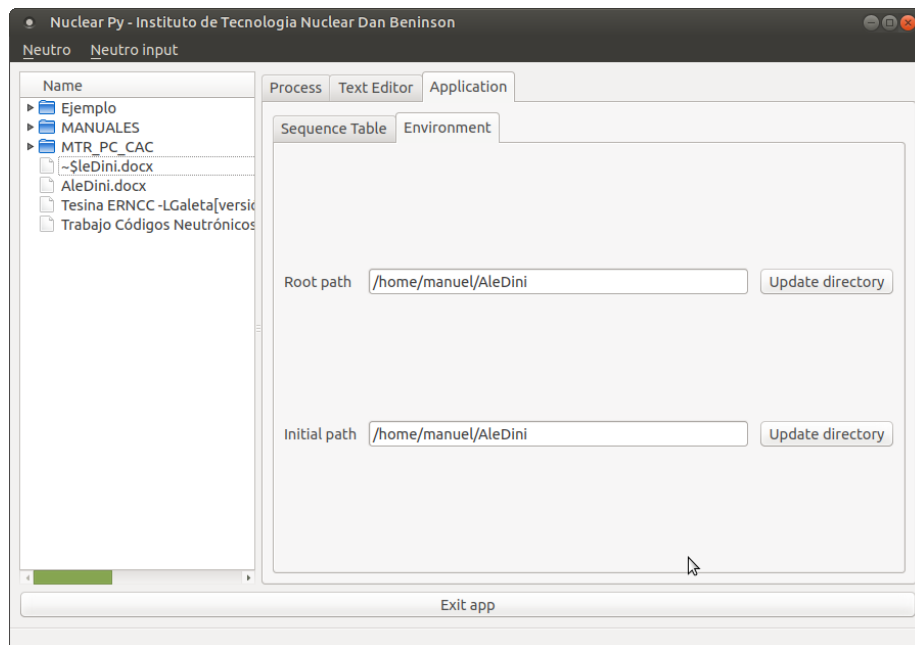
The ST application writes a simple .csv file saved in the nPy main directory. If its necessary to change the settings for a complete different lists of processes, it can be setted by beforehand and simply switchintg the .csv files, one by another.



As shows at the picture, is possible to edit a new ST by hand if the user knows exactly what he/she wants.

## 5.11. Customizing directories

Is possible to set the work directory and the treeview directory at the Environment tab.



The **root path** set the main place where the system starts to run by default, and the **intial path** set the initial view place for the treeview application.

## 6. Windows Build

You can always run nPy through python nPy.py either in windows or linux systems. We also provide windows .exe binares, that can be made following the next steps under a linux system:

### 6.1. Install wine, and Python under wine.

```
sudo apt install wine
```

```
wine msiexec /i python-2.7.15.msi /L*v log.txt
```

you may download latest Python 2.7 release from python web site.

### 6.2. Install PyInstaller on wine

```
cd ~/.wine/drive_c/Python27
```

### 6.3. Install PyQt4 on wine

Get .whl from here (<https://www.lfd.uci.edu/~gohlke/pythonlibs/#pyqt4>) move it to:

```
cd ~/.wine/drive_c/Python27
```

```
wine python.exe Scripts/pip.exe install PyQt4-4.11.4-cp27-cp27m-win32.whl
```

## 6.4. Build.exe

cd into nPy folder

```
wine ~/.wine/drive_c/Python27/Scripts/pyinstaller.exe --onefile -p  
"C:\Python27\Lib\site-packages\PyQt4" nPy.py
```

Check that it is working for Windows with wine

```
wine ./dist/nPy.exe
```