

Name: Andres Ricardo Garcia Escalante

18/01/18

Student Id Number: 4335957

Coursework 3 Report

Design and implement a portable version of the SolverCommandLineGenerator developed in QT framework (C++) in to Java using Netbeans.

1. Appraisal of migrating to Java

There are many problems for migrating into a new Language:

- The interfaces designs are different: QT Framework has different designs of their components (Graphical aspect), so this will not allow us to mimic the interface 100%.
- Due to Java is more well-known as an app developer language. There will not be no problems to migrate the components that were implemented in C++ into to Java.
- Java have different IDEs that provide a lot of help in programming this kind of tasks such as Android Studio and NetBeans.
- Developing code in Java gives the program more portability.

2. Java Design

2.1 General Overview

The design is pretty similar to the C++:

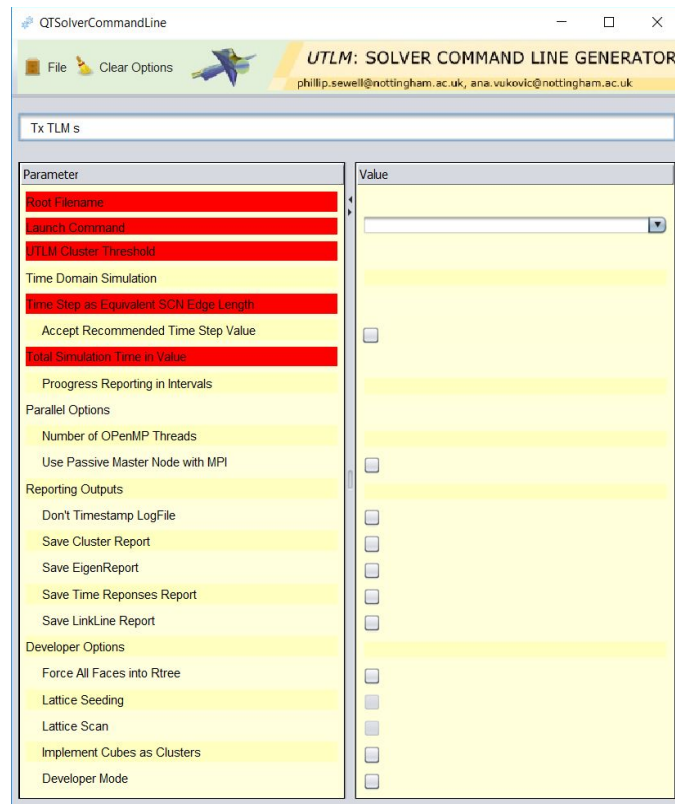


Image1. QTSolverCommandLine Designed in Java.

This Frame is divided into 3 parts:

- **Menu:** It has 2 options available and each of them have different applications.
 - **File:**

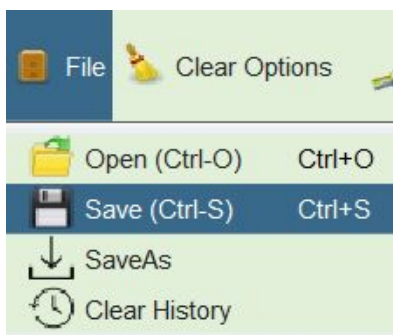


Image2. Options available in the File Menu.

- **ClearOptions:**

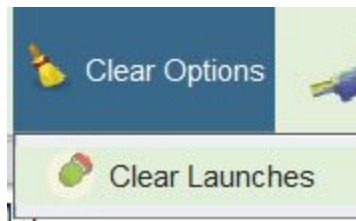


Image3. Clear Options available in the File Menu.

- **Command Line:** Provides the status of the selected options of the Parameters/Values Section.



Image4. Command Line displaying all the options chose.

- **Parameters/Values:** A collection of TextBoxes, Comboxes, and Check Boxes to store the information that the user provides.

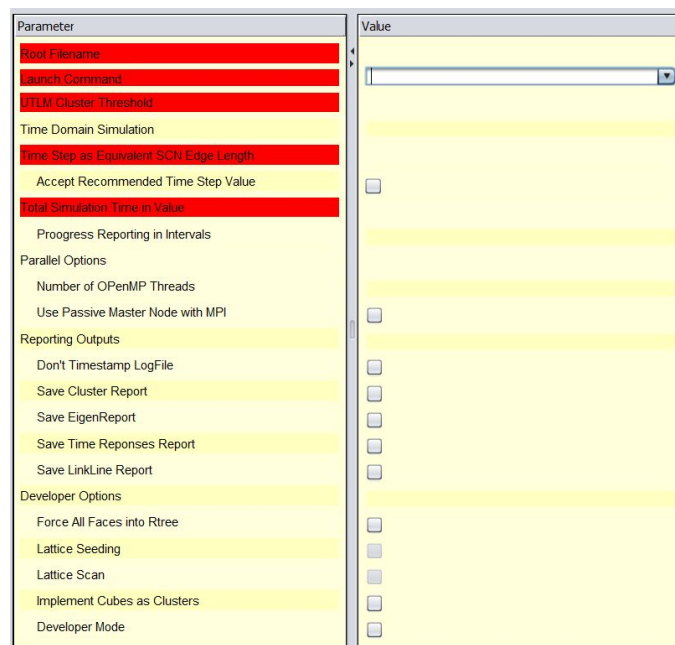


Image5. Design of the Parameters/Value.

2.2 Events when running the GUI

When the program is running the cursor can take 3 different the cursors forms:

- **Hand Cursor:** It shows the user that there is an object that can be modified (Textbox,combobox,checkbox,menu).
- **Writing Cursor:** It tells the user that it cannot modify the content of it, but it can copy it (Command Line).
- **Default Cursor:** It tells the user that the there is not any event if clicked.

In the **Menu's** section:

- **OpenAs:** Opens an additional Frame, A Filechooser will pop up. Here you can access to all the directories of the PC and seek for the .scl Files. It also works with the short cut (Ctrl + o).

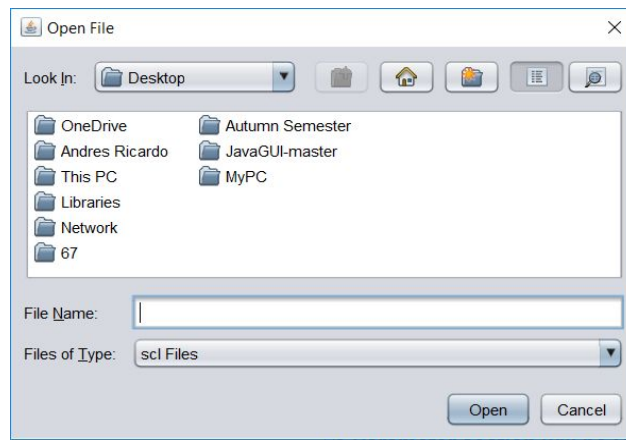


Image6. Open Filechooser.

- **Save As:** Opens an additional Frame, A Filechooser will pop up. Here you can access to all the directories of the PC and store the information from the Parameters/Values section into a File with any given name.

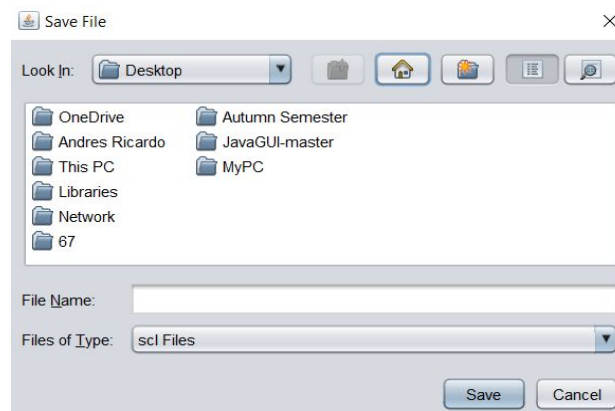


Image7. Save Filechooser.

- **Save:** If the current values assigned were not saved before then this will do the same approach as Save As. If it was already Saved before then by using this option will update all the information to the current file. It also works with the short cut (ctrl+s).
- **Recent Files:** A list of names of the recent Opened and Saved Files.
- **Clear History:** It will Remove all the list of the recent files accessed.

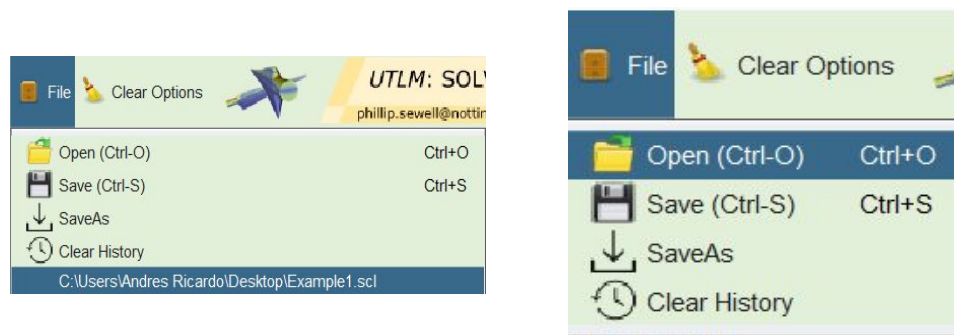


Image 8 and 9. Shows the before and the after of using the Clear History.

- **Clear Launches:** It removes all the options available of in the Combo Box.

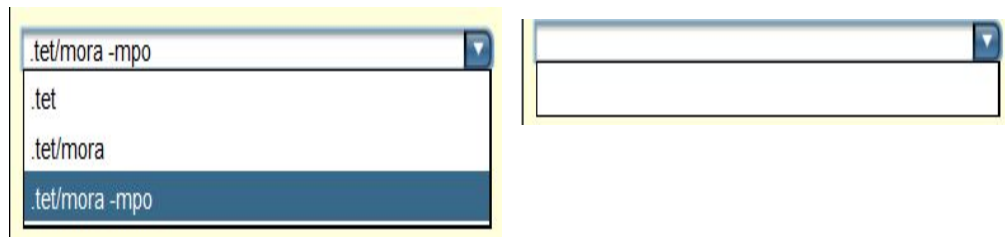


Image 10 and 11. Shows the before and the after of using the Clear Launches.

In the **Parameters/Values** section the different kinds of types of data are admitted:

- **TextBox:** It can admit words with numbers or numbers.
- **ComboBox:** It admits words with numbers.
- **CheckBox:** It admits only boolean type.

If wrong data is introduced the name of the Parameter section will be RED.

Finally the Command Line Section will update the data when any change is made (Save, Open, modify any content of the Parameters/Values Section) and it will display a text of all the selected options.



Image 12 and 13. Shows an empty and full data storage displayed in the Command Line respectively.

2.3 Files Involved

There are 2 types of Files in this GUI:

- **.scl file:** This file contains all the information of the Parameters/Values section data. It has a particular way to store the data (only for .scl files).
- **.ini file:** This file contains all the information from the recent files list and from all the options available from the ComboBox. When When Clear History or Clear Launches happens in will update the information (only when exiting the main Frame).

3. Test and New Features

In the **last section we tested the basic behaviors of each components**. Now we will test the program as a conjunction, for this we will use the old version of the SolverCommandLineGenerator to compare the creation of the files (.ini and .scl). Also we will discuss some new features added in the new version in Java.

3.1 Testing

In the following test we will provide the same data for the old and new versions of the SolverCommandLine and then compare both files. **Initial conditions both of the will have nothing stored in the .ini.**

QT Solver Command Line
—
□
×



File
 Clear Options

UTLM: SOLVER COMMAND LINE GENERATOR
phillip.sewell@nottingham.ac.uk, ana.vukovic@nottingham.ac.uk

b a T89x5r10DP4Mdf TLM s44cEDmld

Parameter	Value
Root Filename	a
Launch Command	b
UTLM Cluster Threshold	44
Time Domain Simulation	
Time Step as Equivalent SCN Edge Length	5
Accept Recommended Time Step Value	<input checked="" type="checkbox"/>
Total Simulation Time in Value	89
Proogress Reporting in Intervals	10
Parallel Options	
Number of OPenMP Threads	4
Use Passive Master Node with MPI	<input checked="" type="checkbox"/>
Reporting Outputs	
Don't Timestamp LogFile	<input checked="" type="checkbox"/>
Save Cluster Report	<input checked="" type="checkbox"/>
Save EigenReport	<input checked="" type="checkbox"/>
Save Time Reponses Report	<input checked="" type="checkbox"/>
Save LinkLine Report	<input checked="" type="checkbox"/>
Developer Options	
Force All Faces into Rtree	<input checked="" type="checkbox"/>
Lattice Seeding	<input type="checkbox"/>
Lattice Scan	<input type="checkbox"/>
Implement Cubes as Clusters	<input checked="" type="checkbox"/>
Developer Mode	<input checked="" type="checkbox"/>

Image 14. All empty spaces filled up with random data (New Version).


File Clear Options


UTLM: SOLVER COMMAND LINE GENERATOR
phillip.sewell@nottingham.ac.uk, ana.vukovic@nottingham.ac.uk

b a T89x5r10DP4Mdf TLM s44cEDmld

Parameter	Value
Root Filename	a
Launch Command	b
UTLM Cluster Threshold	44
Time Domain Simulation	
<div> <div>Time Step as Equivalent SCN Edge Length</div> <div>5</div> </div> <div> <div>Accept Recommended Time Step Value</div> <div><input checked="" type="checkbox"/></div> </div>	
<div> <div>Total simulation time in seconds</div> <div>89</div> </div> <div> <div>Progress Reporting in Intervals</div> <div>10</div> </div>	
<div> <div>Parallel Options</div> <div></div> </div> <div> <div>Number of OPenMP Threads</div> <div>4</div> </div> <div> <div>Use Passive Master Node with MPI</div> <div><input checked="" type="checkbox"/></div> </div>	
<div> <div>Reporting Outputs</div> <div></div> </div> <div> <div>Don't Timestamp Logfile</div> <div><input checked="" type="checkbox"/></div> </div> <div> <div>Save Cluster Report</div> <div><input checked="" type="checkbox"/></div> </div> <div> <div>Save EigenReport</div> <div><input checked="" type="checkbox"/></div> </div> <div> <div>Save Time Reponses Report</div> <div><input checked="" type="checkbox"/></div> </div> <div> <div>Save LinkLine Report</div> <div><input checked="" type="checkbox"/></div> </div>	
<div> <div>Developer Options</div> <div></div> </div> <div> <div>Force All Faces into Rtree</div> <div><input checked="" type="checkbox"/></div> </div> <div> <div>Lattice Seeding</div> <div><input type="checkbox"/></div> </div> <div> <div>Lattice Scan</div> <div><input type="checkbox"/></div> </div> <div> <div>Implement Cubes as CLusters</div> <div><input checked="" type="checkbox"/></div> </div> <div> <div>Developer Mode</div> <div><input checked="" type="checkbox"/></div> </div>	

Image 15. All empty spaces filled up with random data (Old Version).

Now let's save both programs data by using (ctrl + s). This should show the Save File chooser, and let's save it with the name Example1 and Example2 respectively.

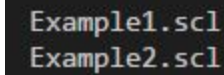
A dark rectangular box containing the text "Example1.scl" and "Example2.scl" in a light-colored, monospaced font, stacked vertically.

Image 16. Files created.

Using the terminal and the command diff, let's test if both of them are equal (they must be):

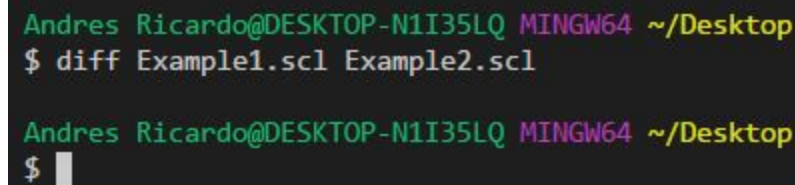
A terminal window with a black background and green text. The prompt is "Andres Ricardo@DESKTOP-N1I35LQ MINGW64 ~/Desktop". The command entered is "\$ diff Example1.scl Example2.scl". The output is blank, and the prompt "\$" is shown again on the next line.

Image 17. Using command diff and no difference in the files.

Now let's test the Open File Option (or Ctrl + o), we will open the opposited file that we are currently in. For New version is Example2.scl and for Old version is the Example1.scl. To check if the process worked, we will check the list of recent files.

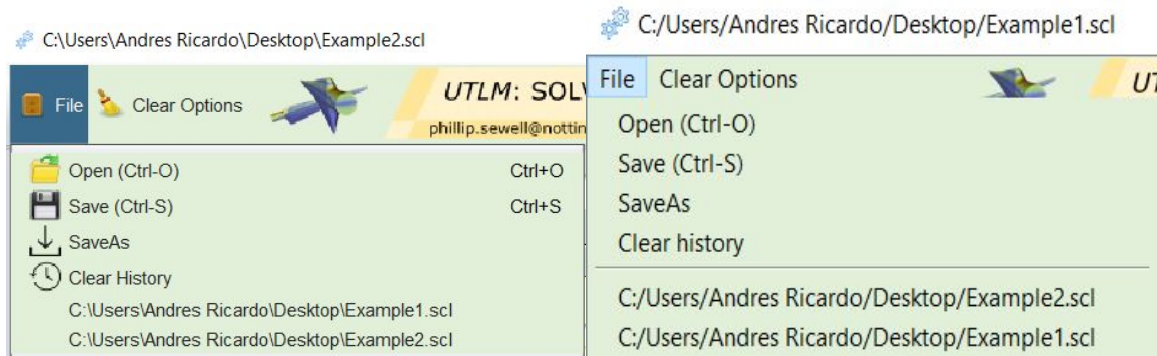


Image 18 and 19 . Recent files list for the new and old version respectively.

Now let's compare the .ini files. As for both files have the same name, I will copy the content of the file .ini of the old version and copy it into the new versions folder with the name Auxi. The test should show 2 differences because 1 opened the Example1 first meanwhile the other Example2.

```
Andres Ricardo@DESKTOP-N1I35LQ MINGW64 ~/Desktop/JavaGUI-master/JavaGUI-master/JavaGUI/dist (master)
$ ls
Auxi.ini  JavaGUI.jar  UTM_SOLVER_command_line.ini

Andres Ricardo@DESKTOP-N1I35LQ MINGW64 ~/Desktop/JavaGUI-master/JavaGUI-master/JavaGUI/dist (master)
$ diff Auxi.ini UTM_SOLVER_command_line.ini
3,4c3,4
< C:\Users\Andres Ricardo\Desktop\Example2.scl
< C:\Users\Andres Ricardo\Desktop\Example1.scl
\ No newline at end of file
---
> C:\Users\Andres Ricardo\Desktop\Example1.scl
> C:\Users\Andres Ricardo\Desktop\Example2.scl
\ No newline at end of file

Andres Ricardo@DESKTOP-N1I35LQ MINGW64 ~/Desktop/JavaGUI-master/JavaGUI-master/JavaGUI/dist (master)
$
```

Image 20. Command diff shows that there are only 2 differences which is logic according our example.

With this we can prove that the new version is working properly (and the little examples shown in section 2.2).

3.2 Additional Features

Some new Features were added:

- **Warning the user that the UTM.ini File is not found:**

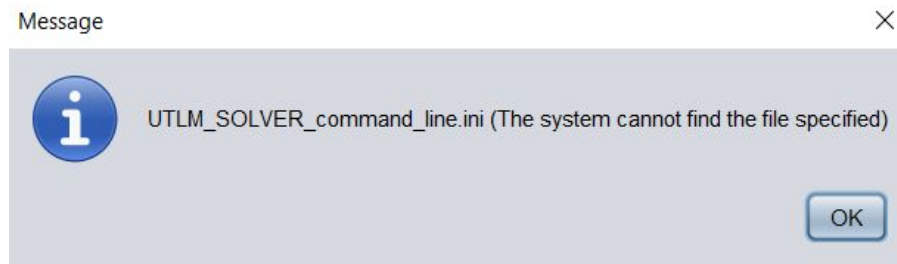


Image 21. Error Handler for not missing .ini File.

- **Notify that the files was saved successfully:**

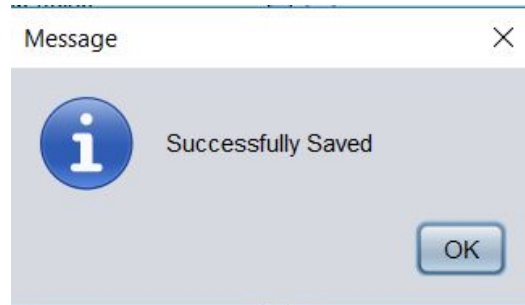


Image 22. Notifying the user that the File is saved.

- **Images to illustrate better the options of the menu:**

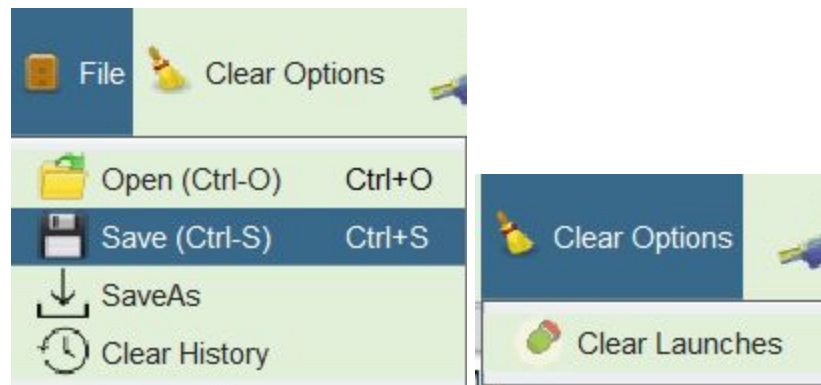


Image 23. Error Handler for not missing .ini File.

- **Warning the user that the process of Save and Open failed**

4. Conclusion

Developing GUIs in netbeans helps the programmer in the creation of any given Frame. This is possible with all the support of this IDE, also the Java has a flexibility in portability.