

OOSIML SIMULATION MODELS

On Windows and Linux

Using a Terminal (or Command) Window and Codeblocks

Dr. José M. Garrido
Department of Computer Science

Spring 2017

College of Computing and Software Engineering
Kennesaw State University

© 2016 J. M. Garrido

1 Introduction

Software can be developed using directly text operating systems shell commands with an appropriate editor, or an Integrated Development Environment (IDE). There are several IDEs that help developing programs written in C, C++, Fortran, and other programming languages. Eclipse is one of the most complete and powerful tools, it is mainly useful for Java programs and in general, it can be very slow. For C++, C, and Fortran programs, Codeblocks and CodeLite are faster, lighter, and more convenient tools to use.

This document briefly explains implementing OOSimL simulation models and C++ programs using:

- The text Linux and Windows commands necessary that can be used on a Terminal or Command window.
- CodeBlocks on Linux and Windows

Note that the OOSimLC compiler/translator, and the simulation library were compiled with GNU C++ compiler tools.

2 Downloading and Installing the OOSimLC Software

The preliminary step required is the installation of the the OOSimL translator, the simulation library, and the corresponding header files in the computer system running Linux or Windows.

1. On Linux, download the archive file `oosimlc.tar.gz` from the following web page and extract all files from archive.

`ksuweb.kennesaw.edu/~jgarrido/oosimlc`

2. Select the executable file of the OOSimL compiler/translator, `oosimlc.out` for 32-bit Linux or `oosimlc64.out` for 64-bit Linux depending on the version of Linux your computer (32-bit or 64-bit).
3. Rename `oosimlc64.out` to `oosimlc.out` in a 64-bit Linux.
4. The compiler file (`oosimlc.out`), the simulation library (`liboosimlc.a`), the `comp1` script file and the header files (including `oosiml.h`) must be stored on a folder such as `~/oosimlkdir`.

5. The subdirectory **models**, contains some pre-defined simulation models and have the **osl** extension.

On Windows, the name of the archive is **oosimlc.zip**. The name of the OOSimL translator is **oosimlc.exe**. The simulation library and the header files have the same names as mentioned for Linux.

3 Using a Terminal or Command Window

3.1 Using Linux

1. On Linux, open a Terminal window and change to the directory where the OOSimLC software is stored. Type the command:

```
cd oosimlmdir
```

2. Verify that the compiler/translator, the simulation library, and the header files are in the directory. Type **ls**, and this will list all files and subdirectories in the current directory.
3. The subdirectory **models** stores all the OOSimLC simulation models. These have an **.osl** extension. Change to the subdirectory by typing **cd models**, then type **ls** to verify that all simulation models from the downloaded archive are present.
4. To compile the simulation model in file **Carwash.osl**, type the following command:

```
../oosimlc.out Carwash.osl
```

This is equivalent to the command:

```
~/oosimlmdir/oosimlc.out Carwash.osl
```

5. Verify that the new file generated, **Carwash.cpp**, is in the current directory. Type **ls -lt**, and this will list all files and subdirectories in the current directory ordered by date and time.
6. Compile and link the generated file (**Carwash.cpp**) with the GNU C++ compiler tool set. To compile and link file **Carwash.cpp**, type the following command:

```
g++ Carwash.cpp -I ~/oosimlmdir -L ~/oosimpcdir -loosimlc
```

```
-lpthread -lm
```

This will generate the executable file (by default) `a.out`, which can be renamed with the `mv` command.

7. To run the executable file generated, type the command:

```
./a.out
```

When the simulation model runs, it produces two output text files, the statics file and the simulation trace file. For the *Carwash* model, the files produced are: `carwstatf.txt` and `carwtrace.txt`.

3.2 Using Windows

1. On Windows, open a command window by typing `cmd` on the Search option. Change to the directory where the OOSimLC software is stored. Type the command:

```
cd oosimlkdir
```

2. Verify that the compiler/translator, the simulation library, and the header files are in the directory. Type `dir`, and this will list all files and subdirectories in the current directory.
3. The subdirectory `models` stores all the OOSimLC simulation models. These have an `.osl` extension. Change to the subdirectory by typing `cd models`, then type `dir` to verify that all simulation models from the downloaded archive are present.
4. To compile the simulation model in file `Carwash.osl`, type the following command:

```
C:\oosimlkdir\oosimlc.exe Carwash.osl
```

5. Verify that the new file generated, `Carwash.cpp`, is in the current directory. Type `dir /p/o-d`, and this will list all files and subdirectories in the current directory ordered by date and time.
6. Compile and link the generated file (`Carwash.cpp`) with the GNU C++ compiler tool set. To compile and link file `Carwash.cpp`, type the following command:

```
g++ Carwash.cpp -I C:\oosimlkdir -L C:\oosimpcdir -loosimlc
```

```
-lpthread -lm
```

This will generate the executable file (by default) `a.exe`, which can be renamed with the `ren` command.

7. To run the executable file generated, type the command:

```
a.exe
```

When the simulation model runs, it produces two output text files, the statics file and the simulation trace file. For the *Carwash* model, the files produced are: `carwstatf.txt` and `carwtrace.txt`.

4 Using CodeBlocks IDE

For a more detailed introduction to CodeBlocks, refer to various websites for CodeBlocks tutorial and documentation.

The following sequence of steps involve the basic procedure for setting the appropriate options on Codeblocks for using the OOSimL translator then the remaining steps for editing C++ programs, compiling, and linking with the simulation library in file `liboosimlc.a`.

4.1 Setting the Codeblocks Options

1. Start CodeBlocks and click on Create a new project.
2. Select Console Application and click the Go button, which is located on the upper right corner of the window.
3. On the Console Application window, click the Next button. Select C++ language and click Next.
4. Type the project title (name), e.g. *Carwash*. In this example, the project will be created in folder: `~/oosimlcdir/models`. Click the Next button.
On Windows, the project will be created in folder: `C:\oosimlcdir\models`
5. Select GNU GCC Compiler and click the Finish button.
6. On the top bar, click on the Settings menu then select Compiler. A dialog box appears: Global compiler settings. On the tab group, activate the Other settings tab, and click the button Advanced options located on the lower right of the dialog box.
7. Click the Yes button when the box *Edit advanced compiler setting?* appears.

8. On the new window that appears, click the '+' button and type `osl` in the dialog box and click OK. This indicates that the OOSimL files have an *osl* extension.

9. In the field: the Command line macro, type the command that will be used by CodeBlocks to execute the translator with a given OOSimL source file. For example:

```
~/oosimlkdir/oosimlc.out $file
```

This assumes that the OOSimL translator (executable file `oosimlc.out`) is located in a folder named: `~/oosimlkdir`

On Windows, the Command line macro that will be used by CodeBlocks to execute the translator with a given OOSimL source file is:

```
C:\oosimlkdir\oosimlc.exe $file
```

10. In the Generated files field, type the generic file name generated that are C++ files:

```
$file_name.cpp
```

11. Click the Ok button located on the bottom of the window.

Codeblocks is now configured, for the current project, to recognize source files with an `osl` extension for editing and to execute the OOSimL translator.

4.2 Setting for Compiling and Running Simulation Models

4.2.1 Source Files

1. Activate the left pane of the screen (Management), click the Projects tab. Remove the source file `main.c` by right-clicking on it and selecting Remove file from project.
2. A new source file can be created by selecting File menu, then New, and Empty file. Click Yes to add this empty to the project. A new dialog window appears, type the name of the file with its `osl` extension. Now you can start editing this source file and when finished, save the file.
3. If one or more existing source files are to be included in the project, select the Project menu on the top bar, or right-click on the project name. Select Add files. Select the directory of the source to add to the project and select the OOSimL source file(s). Click Open.

4. The source files are now under Others and under the project name. Double-click on the desired source file to edit it further. The file now appears on the edit area of the screen.

4.2.2 Build Options

1. Right-click on the current project name, or activate the Project menu in the top bar, and select Build options.
2. On the tab Compiler settings, check Enable all compiler warnings.
3. On the tab Linker settings, click the Add button to add a library to the project. Repeat to add all necessary object files and libraries. For a typical simulation model, at the minimum, the file libraries are: `liboosimlc.a`, `pthread`, and `m` (standard math library).

Note that the simulation library in file `liboosimlc.a` is stored in a local directory, such as `~/oosimlkdir`. On Windows, the library would be stored in folder: `C:\oosimlkdir`.

4. On the tab Search directories and the tab 'Compiler', add the search directory for header files required by the source program while compiling. Assume the header files required by the programs are located in folder `~/oosimlkdir`.
5. On the tab Search directories and the tab 'Linker', add the search directory of the libraries if needed. On Linux (Ubuntu), the external libraries are always stored in standard system directories. The file of the simulation library `liboosimlc.a` can be stored in any location, for example `~/oosimlkdir`. For this, click the Add button, then click (...) and navigate to the appropriate directory.

4.2.3 Building and Running the Project

1. Build the project, which translates the OOSimL source file, then compiles and links the C++ files in the project. On the top bar, select the Build menu and select the Build option. The Build log appears in the lower pane of the Codeblocks screen.
2. To execute the program, select the Run option in the Build menu. A new screen appears with the results of the execution. After the program terminates execution, press the Enter key.

3. On the top bar, activate the File menu and select Close project. Codeblocks creates several new subdirectories and the executable file in directory `~/oosimlkdir/models/Carwash/bin/Debug`.