



The University of Electro-Communications

Tutorial for SUMO and OMNeT++ interfacing by libsumo

Author: Chanarisu

Celimuge laboratory, Chofu, Tokyo, Japan

November, 2022

Contents

1	Introduction	2
2	Installation	3
2.1	Installing SUMO	3
2.2	Installing OMNeT++	4
2.3	Install INET framework	5
2.3.1	Modify the file <i>Makefile.inc</i>	9
2.3.2	Include libsumo libraries	10
3	Examples	10
3.1	Install the example	10
3.2	Running the example	11
3.3	Random route	11
3.4	Add a vehicle	12

1 Introduction

Simulation of **Urban MObility** (SUMO) can simulate a realistic vehicular traffic. So far, four kinds of programming language can process the traffic data from SUMO, python, matlab, java and C++. Here, we mainly introduce SUMO coupling with OMNeT++ by C++ programming language. Regarding C++ language, three interfacing approaches can couple SUMO with OMNeT++, including TraCIScenarioManager, Veins and libsumo.

- TraCIScenarioManager appeared first and is an interface based on **Traffic Control Interface** (TraCI) TraCI command, which can control on-line in the bi-directional between OMNeT++ and SUMO. For TraCI command, more information see the website <https://sumo.dlr.de/docs/TraCI.html>.
- **Vehicles in network simulation** (Veins) was developed by Christoph Sommer and his team. Veins is as a middle-ware based on TraCI command for coupling SUMO with OMNeT++. Veins provides a suit of modules for simulating vehicular network with the mobility. Before running the veins module, `sumo_launchd.py` (so far, is updated `veins_launchd.py` in the *veins_of_dir/bin*) needs to be started, to create a daemon for listening the request. For veins, more information see the website <https://veins.car2x.org>.
- libsumo, is an interfacing library developed by SUMO. Due to the communication overhead running the TraCIScenarioManager, libsumo can provide a more efficient coupling without the need for socket communication.

The libsumo library provides a flexible interface between SUMO and OMNeT++. Some functionality that veins simulator could not implement can be easily achieved. Here, we implement the functions below.

1. Bus, truck, coach, taxi and e-vehicle can be mapped different nodes into OMNeT++.
2. The configured route for each vehicle is valid route and does not occur no connection or other TRACIException. In contrast, the random trips generated by the `randomTrips.py` (in the `sumo_home_dir/tools`) has not been validated.
3. All vehicle entered the map in the same time.
4. The random route for each vehicle is generated when the simulator starts.
5. Does not start `veins_launchd.py` to listen port.

This tutorial and related code have been updated to the github and can be downloaded from https://github.com/nrs018/omnet_libsumo.

2 Installation

The libsumo library is tested on Ubuntu and on windows and MacOS, has not been tested yet.

For the version of the software are listed below.

- Ubuntu: 20.04.2 LTS.
- OMNeT++: 5.5.1, download from <https://omnetpp.org/download/>.
- INET framework: 4.2.2, download from <https://inet.omnetpp.org/Download.html>.
- SUMO: 1.11.0, download from <https://sumo.dlr.de/docs/Downloads.php>.

We recommend that installing SUMO, OMNeT++ and INET framework are manual.



This tutorial does not yet tested on the Windows and MacOS.

2.1 Installing SUMO

Download sumo-src-1.11.0.tar.gz package from the site <https://sumo.dlr.de/releases/1.11.0/> and unzip the package to your dir.

And then run the command following.

```
> sudo apt-get install cmake python g++ libxerces-c-dev libfox-1.6-dev
```

```
> sudo apt-get install libgdal-dev libproj-dev libgl2ps-dev swig
```

```
> cd <yout_sumo_dir> # your_sumo_dir needs to be replaced real address in  
your computer.
```

```
> export SUMO_HOME="\$PWD" # SUMO_HOME can be set to be the system en-  
vironment variable.
```

```
> mkdir build/cmake-build && cd build/cmake-build
```

```
> cmake ../..
```

```
> make -j\$(nproc)
```

After installing SUMO, check whether the following files exist in the directory *your_sumo_dir*.

```
| -bin  
| - ...  
| -libsumocpp.so ← This file will be linked to the OMNeT++.  
| -libtracicpp.so ← This file will be linked to the OMNeT++.  
| - ...  
| -build
```

```
| -data  
| -docs  
| -include  
| -src ← This directory will be referenced in the OMNeT code.  
| -tools  
| -unittest  
| -AUTHORS  
| -ChangeLog  
| -CMakeLists.txt  
| -CMakeSettings.json  
| -cmake-variants.yaml  
| -CONTRIBUTING.md  
| -Jenkinsfile  
| -LICENSE  
| -NOTICE.md  
| -README.md  
| -sumo.doxyconf
```

2.2 Installing OMNeT++

Download OMNeT++ 5.5.1 from the site https://omnetpp.org/download/old#omnetpp-551_linux and unzip the package to your dir.

Installing required packages.

```
> sudo apt-get install build-essential clang lld gdb bison flex perl  
> sudo apt-get install python3 python3-pip qtbase5-dev qtchooser qt5-qmake  
> sudo apt-get install qtbase5-dev-tools libqt5opengl5-dev libxml2-dev zlib1g-dev  
> sudo apt-get install doxygen graphviz libwebkit2gtk-4.0-37  
> python3 -m pip install --user --upgrade numpy pandas matplotlib scipy  
> python3 -m pip install --user --upgrade seaborn posix_ipc
```

And then, start to install OMNeT++.

```
> cd your_omnet_dir  
> . setenv  
> ./configure  
> make && make install
```

Finally, open OMNeT simulator.

```
> omnetpp
```

For more information related to the installation of SUMO and OMNeT, check the site <https://sumo.dlr.de/docs/Installing/index.html> and <https://doc.omnetpp.org/omnetpp/InstallGuide.pdf>.

2.3 Install INET framework

The network related all protocols, the algorithms and the structures are implemented in the INET framework which developed by OMNeT++ researcher. By default, the INET framework is installed automatically after OMNeT++ installs. However, in the most cases, some scenarios need other version of INET framework. Here, we recommend to install INET framework manually.

Download INET 4.2.2 from the site <https://inet.omnetpp.org/2021-01-11-INET-4.2.2-released.html> and unzip the tar package to *your_inet_dir*.

Next, the detailed installation process to INET framework are presented.

1. Open OMNeT++.

Click File in the menu -> New -> Project....

And then select OMNeT++ Project from OMNeT++, as shown in Figure 1.

At last, click Next button.

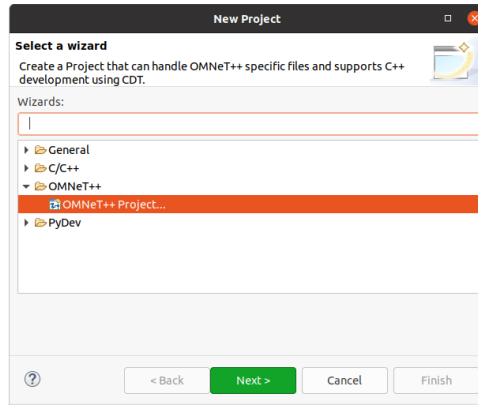


Figure 1: *Step 1.*

2. Input the project name and select empty project with 'src' and 'simulation' folders. And then, click Finish button, shown as in Figure 2 and 3.

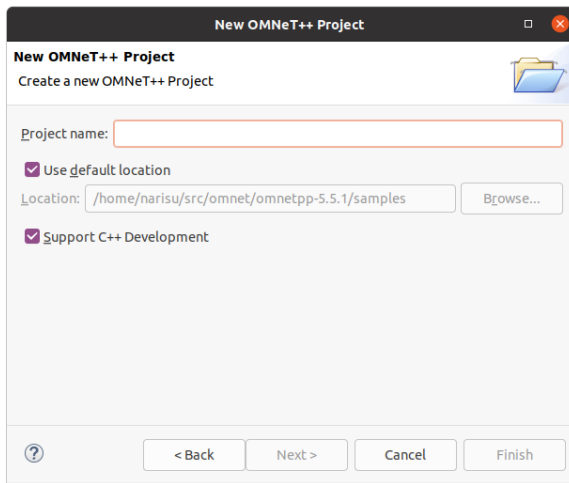


Figure 2: Step 2.

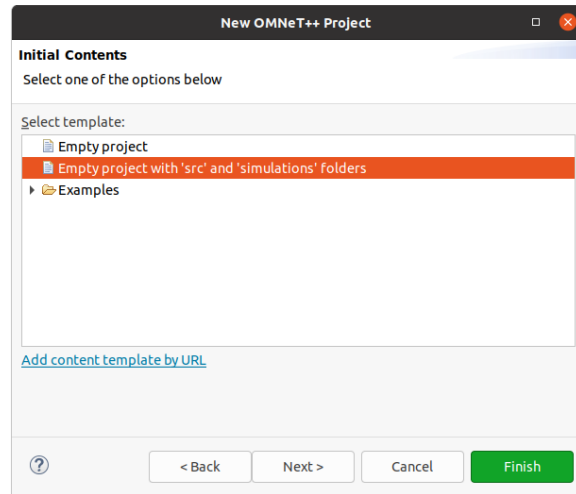


Figure 3: Step 3.

3. Select new project and click File in the menu -> import, as shown in Figure 4. And then, select File System and click Next button, as shown in Figure 5.

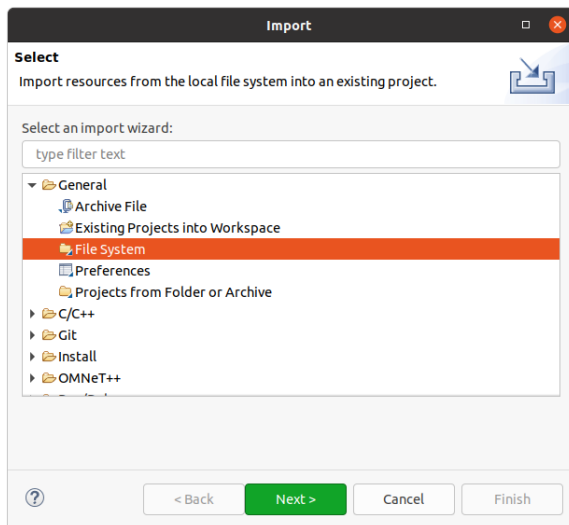


Figure 4: Step 4.

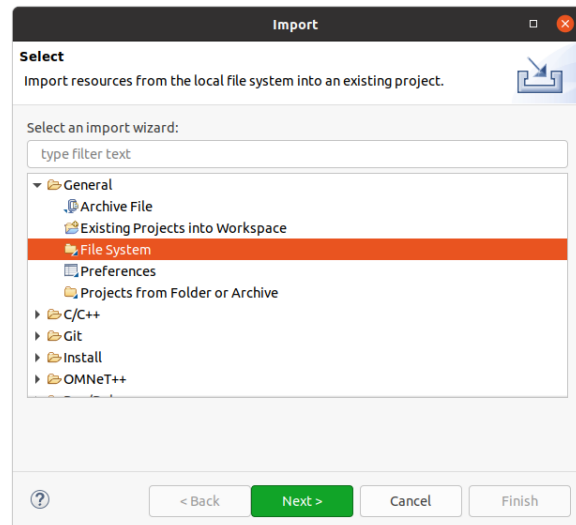


Figure 5: Step 5.

4. Click Browse button and select unzipped INET package from your computer, as shown in Figure 6.

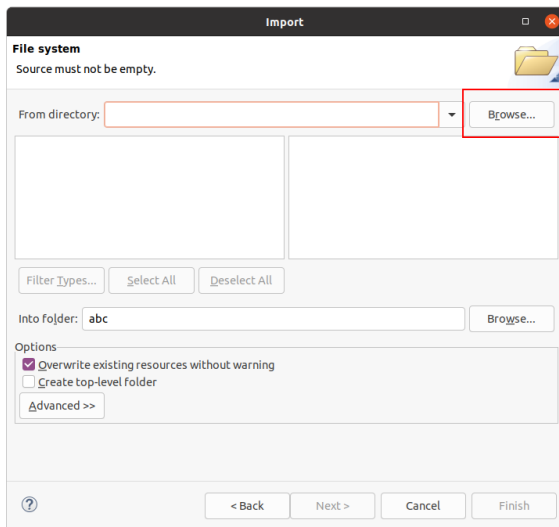


Figure 6: Step 6.

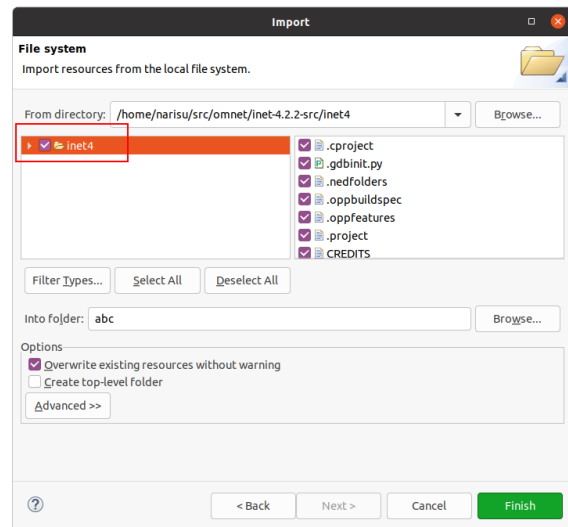


Figure 7: Step 7.

5. Click the check box of inet4 in the left bar, and then click *Finish* button, as shown in Figure 7.
6. Imported INET framework seems to be errors, as shown in Figure 8. In this time, open the file *package.ned* from src/inet folder and remove the code, as shown in Figure 9.

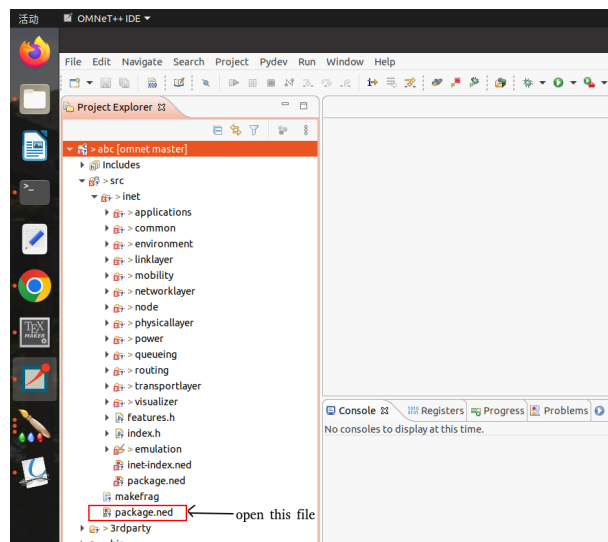


Figure 8: Step 8.

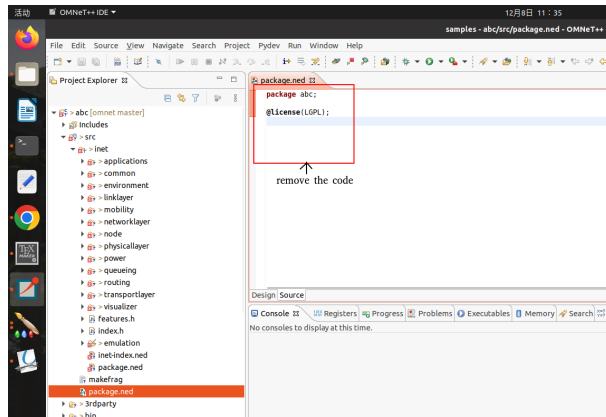


Figure 9: Step 9.

7. Select this project and build the project. By now, INET framework has been imported into OMNeT++.



For the convenient, other unrelated projects are closed.

8. The following is the process for linking *.so and including libsumo library. To compile and link libsumo library and OMNeT, some option of make files needs to be modified. The file *Makefile.inc* locates in the OMNeT root directory, as shown in Figure 10. And open *Makefile.inc*.

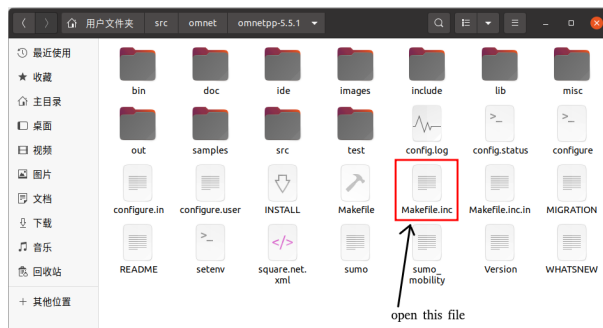


Figure 10: Step 10.

9. *Makefile.inc* needs to be edited, as shown in Figure 11. *Makefile.inc* is also uploaded to Github.

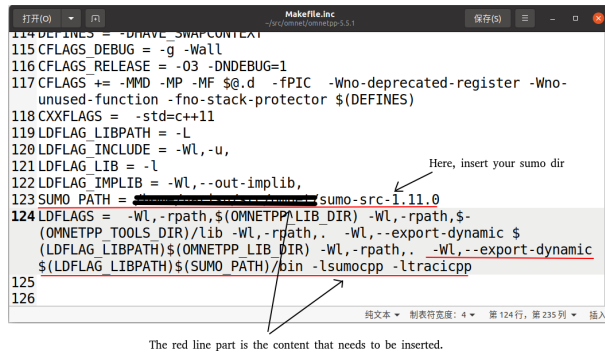


Figure 11: Step 11.

10. All dirs and files in the folder *your_sumo_dir/src* are copied into *omnet_dir/include*. Figure 12 is before the copying, and Figure 13 is after copying.

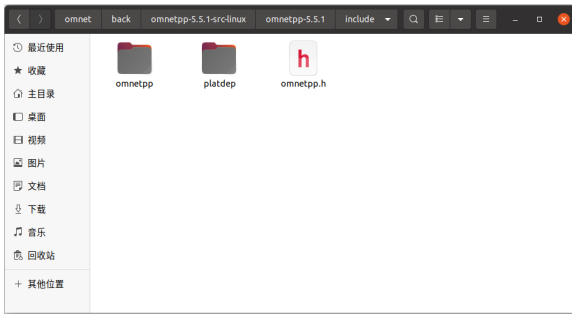


Figure 12: Step 12.

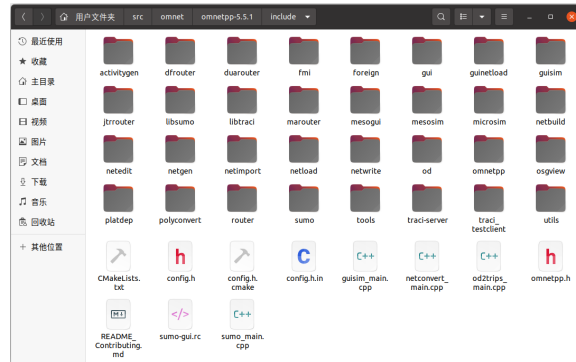


Figure 13: Step 13.

2.3.1 Modify the file *Makefile.inc*

This

Open Makefile.inc file and add the part of red.

```
...
SUMO_PATH = your_sumo_dir/sumo-src-1.11.0
LD_FLAGS = -Wl,-rpath,$(OMNETPP_LIB_DIR) -Wl,-rpath,
           $(OMNETPP_TOOLS_DIR)/lib -Wl,-rpath,.
           -Wl,--export-dynamic $(LDFLAG_LIBPATH)$(OMNETPP_LIB_DIR) -Wl,
           -rpath,. -Wl,--export-dynamic $(LDFLAG_LIBPATH)$(SUMO_PATH)/bin
           -lsumocpp -ltracicpp
...
```



your_sumo_dir needs to be replaced with the address in your computer.

The purpose of the modification is that two files `libsumocpp.so` and `libtraciccip.so` are linked to the `omnet` project when the OMNeT project compiles.

2.3.2 Include libsumo libraries

The libsumo library locates in the directory *your_sumo_dir/src*. All of elements contained in the directory *your_sumo_dir/src* are copied to the directory *omnet_root_dir/include*.



In general, *sumo_root_dir* is added to the `PATH` variable. As long as the source files of SUMO can be found out.

3 Examples

This ssection presents an example of SUMO coupling with OMNeT++ interfacing by libsumo library.

We upload the core code of the example to Github and the source code is downloaded from the site https://github.com/nrs018/omnet_libsumo.



The source code downloaded from the Github can not directly run in your environment. Because `Makefile.inc` file needs to be modified.

3.1 Install the example

Before starting the example, the settings accordding to Section 2 must be configured. And then, unzip the package downloaded from Github.

The directory includes the following.

```
|
| -src
|   |-mobility
|     |-single
|       |-SUMOMap.cc
|       |-SUMOMap.h
|       |-SUMOMap.ned
|       |-SUMOMobility.cc
|       |-SUMOMobility.h
```

```

        |SUMOMobility.ned
|-visualizer
    |-roads
        |-SUMORoadsCanvasVisualizer.cc
        |-SUMORoadsCanvasVisualizer.h
        |-SUMORoadsCanvasVisualizer.ned
|-examples
    |-SUMO
        |-erlangen.net.xml
        |-erlangen.sumocfg
        |-shinjuku.net.xml
        |-shinjuku.sumocfg
        |-odaiba.net.xml
        |-odaiba.sumocfg
        |-handeda.net.xml
        |-handeda.sumocfg
        |-omnetpp.ini
        |-SUMOwithOMNeT.ned

```

The directory **mobility** copies to the *inet/mobility* and the directory *visualizer* copies to the *inet/visualizer*. The *SUMO* directory in the *examples* directory copies to *inet/examples*.

3.2 Running the example

3.3 Random route

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

3.4 Add a vehicle

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu

libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.