

# SUMO ET OMNET++

Ce document se focalise sur l'intégration des données générées avec SUMO dans OMNeT++.

Vous trouverez dans la première partie l'explication générale des différents fichiers de configurations nécessaires à l'intégration des données de SUMO dans OMNeT++.

La seconde partie est la création d'un nouveau projet OMNeT++ avec l'intégration des fichiers SUMO créés ultérieurement. Les fichiers SUMO utilisés sont ceux générés dans le document [04-découverte de Veins et SUMO.pdf](#).

D'autres documents seront publiés pour expliquer l'intégration du protocole GeoNetworking dans OMNeT++.

# I. Explication générale

Cette partie est inspirée de la vidéo [suivante](#). Cependant les fichiers nécessaires au fonctionnement du tutoriel donné en lien ne semblent pas/plus fonctionner sur Windows.

Cette partie présentera les généralités sur les fichiers nécessaires à la création d'une simulation liant SUMO, VEINS, INET et OMNeT++.

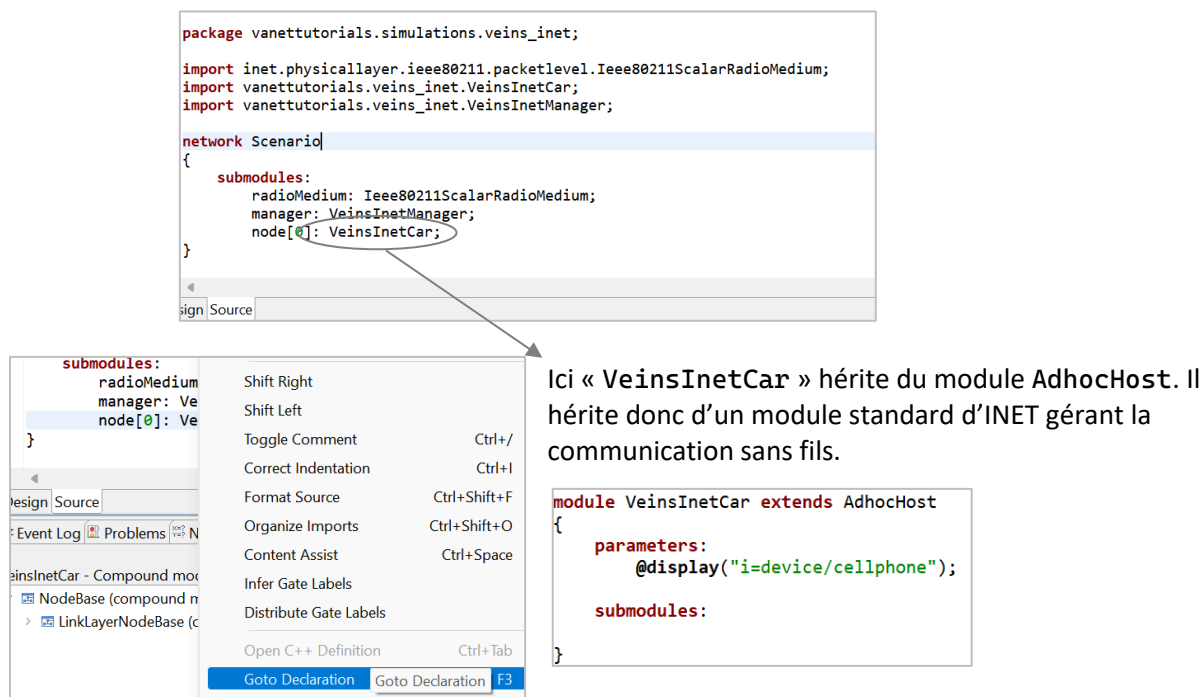
Je vous invite à consulter les fichiers [02-comprendre IDE OMNeT.pdf](#) et [03-Comprendre OMNeT.pdf](#) pour plus de détails sur les différents types de fichiers présents sur OMNeT++. En effet, ils ne seront pas détaillés dans cette partie.

## 1. Le fichier NED

*Ce type de fichier permet de voir la structure du réseau / la description du réseau.*

Un fichier « `Scenario.ned` » sera nécessaire pour lancer une simulation. Il est déclaré dans le fichier INI détaillé dans la sous-partie suivante.

Voici un exemple de fichier ned :



The screenshot shows the OMNeT++ IDE. The top window displays a NED file with the following code:

```
package vanettutorials.simulations.veins_inet;

import inet.physicallayer.ieee80211.packetlevel.Ieee80211ScalarRadioMedium;
import vanettutorials.veins_inet.VeinsInetCar;
import vanettutorials.veins_inet.VeinsInetManager;

network Scenario
{
    submodules:
        radioMedium: Ieee80211ScalarRadioMedium;
        manager: VeinsInetManager;
        node[0]: VeinsInetCar;
}

design Source
```

A context menu is open over the `node[0]: VeinsInetCar;` line. The menu includes options like 'Toggle Comment', 'Correct Indentation', 'Format Source', 'Organize Imports', 'Content Assist', 'Infer Gate Labels', 'Distribute Gate Labels', 'Open C++ Definition', and 'Goto Declaration' (highlighted in blue). A text box on the right explains: 'Ici « VeinsInetCar » hérite du module AdhocHost. Il hérite donc d'un module standard d'INET gérant la communication sans fils.'

Below the text box, a code snippet shows the definition of `VeinsInetCar`:

```
module VeinsInetCar extends AdhocHost
{
    parameters:
        @display("i=device/cellphone");

    submodules:
}

module VeinsInetCar - Compound model
{
    NodeBase (compound model)
    > LinkLayerNodeBase (compound model)
```

Cela permet d'utiliser l'entièreté des fonctionnalités du module INET dans une simulation Veins. Il permet par exemple d'accéder aux modules IPv4, IPv6, aux réseaux sans fils, mobiles etc.

## 2. Le fichier INI

*Le fichier INI contient les paramètres et les configurations qui permettent d'exécuter un modèle de simulation.*

Le fichier `omnetpp.ini` permet de donner toutes les configurations de notre simulation. Ici il est important que le type de mobilité soit la suivante :

```
# VeinsInetMobility
*.node[*].mobility.typename = "VeinsInetMobility"
```

Ce type, `VeinsInetMobility`, doit être défini dans les sources :

- VANETTutorials >> simulations >> src >> veins\_inet >> VeinsInetMobility.cc
- VANETTutorials >> simulations >> src >> veins\_inet >> VeinsInetMobility.h
- VANETTutorials >> simulations >> src >> veins\_inet >> VeinsInetMobility.ned

Ces fichiers représentent le lien liant Veins et INET.

Le fichier `omnetpp.ini` comporte bien sûr d'autres éléments nécessaires au bon fonctionnement de la simulation :

- La partie `UDPBasicApp` :

```
# UDPBasicApp
*.node[*].numApps = 1
*.node[*].app[0].typename = "vanettutorials.veins_inet.VeinsInetSampleApplication"
*.node[*].app[0].interface = "wlan0"
```

Ici `VeinsInetSampleApplication.cc` est une classe qui doit être retrouvée dans les sources. [Ce fichier](#) est bon exemple pour comprendre comment un scénario est codé. *Pour plus de détails sur le code de ce fichier consulter la vidéo dont le lien est disponible dans la partie I (6,00min – 7,02min).*

- La partie `VeinsInetManager` :

Située à la fin du document permet de donner les paramètres de connexion nécessaire pour SUMO :

```
# VeinsInetManager
*.manager.updateInterval = 0.1s
*.manager.host = "localhost"
*.manager.port = 9999
*.manager.autoShutdown = true
*.manager.launchConfig = xmldoc("square.launchd.xml")
*.manager.moduleType = "vanettutorials.veins_inet.VeinsInetCar"
```

### 3. Démarrer SUMO (rappel)

Une fois les fichiers créés et chargés il faut penser à lancer à établir les connexions TCP entre OMNeT++ et SUMO.

Pour établir la connexion :

- Ouvrir le terminal MSYS2 :
  - o `cd C:\Users\nom_user\source\omnetpp-5.6.2`, appuyer sur entrer
  - o `mingwenv.cmd`, appuyer sur entrer
- Rentrer la commande suivante :  
`C:/Users/nom_user/source/veins-veins-5.2/bin/veins_launchd -vv -c`  
`C:/Users/nom_user/source/sumo-1.8.0/bin/sumo.exe`

Ne pas oublier de charger le fichier `.sumocfg` dans un autre terminal avant de rentrer la commande précédente.

## II. Créer sa simulation

Cette partie s'inspire de cette [vidéo](#).

En guise d'exemple, les fichiers SUMO, créés dans le fichier [04-découverte de Veins et SUMO.pdf](#) partie II.1.b, seront utilisés.

### 1. Création d'un nouveau projet dans OMNeT++

- Ouvrir le simulateur OMNeT++
  - o Ouvrir le terminal MSYS2
  - o Rentrer la commande « `omnetpp` » et appuyer sur entrer.
- Créer un nouveau projet :
  - o Dans le menu du haut, cliquer sur `File >> New >> OMNeT++ project`
  - o Nommer ce projet `test_reims`.
  - o Choisir le répertoire où le projet sera stocké. Mon projet sera stocké, pour le tutoriel, dans le dossier `C:/Users/nom_user/source/projets`.
  - o Appuyer sur « `Next` »
  - o Sélectionner « `Empty project` »
  - o Terminer en cliquant sur « `Finish` »

### 2. Rajouter les fichiers SUMO

- Copier l'intégralité des fichiers générés dans le tutoriel précédent, qui se trouvent dans le dossier `C:/Users/nom_user/Sumo`, dans le répertoire projets créé à l'étape 1. Les fichiers sont les suivants :
  - o `reims.net.xml`
  - o `reims.trips.xml`
  - o `routes.rou.alt.xml`
  - o `routes.rou.xml`
  - o `reims.rou.xml`
  - o `reims.sumocfg`

Les fichiers apparaissent alors dans OMNeT++.

- Créer le fichier `test_reims.launchd.xml` dans le dossier `projets`.
  - o Ouvrir le avec l'éditeur de votre choix afin de le compléter avec les lignes suivantes :

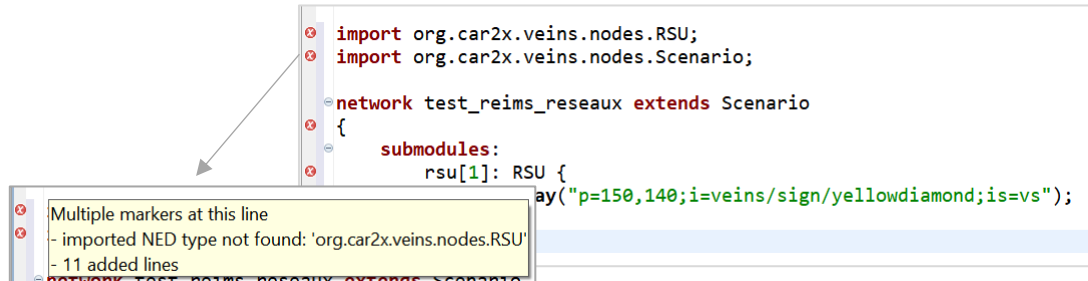
```
<?xml version="1.0" encoding="utf-8"?>
<!--debug config-->
<launch>
  <copy file="reims.net.xml"/>
  <copy file="reims.rou.xml"/>
  <copy file="reims.sumocfg" type="config"/>
</launch>
```

- Dans l'explorateur de fichier placer vous dans le dossier suivant :  
`C:/Users/nom_user/source/veins-veins-5.2/examples/veins`
- Copier les fichiers suivants dans le dossier `projets` :
  - o `antenna.xml`
  - o `config.xml`

### 3. Création du fichier NED

- Dans l'IDE OMNeT++ cliquer sur `File >> New >> Network Description File (NED)`
- Nommer le fichier `test_reims.ned` et vérifier que le dossier sélectionné soit bien `test_reims`.
- Cliquer sur « `Next` ».
- Sélectionner « `Empty NED file` ».

- Cliquer sur « Finish ».
- Afin de faciliter la création du fichier copier le code présent dans le document C:/Users/nom\_user/source/veins-veins-5.2/examples/veins/RSUExampleScenario.ned dans le fichier créé à l'étape précédente : test\_reims.ned.
- Changer le nom de la fonction par un nouveau nom :



On remarque que les fonctions Veins ne sont pas encore reconnues. En effet, il faut les rajouter dans notre projet en passant le module `veins` en référence.

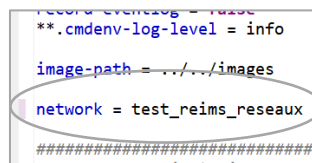
- Pour importer les classes Veins dans notre projet :
  - o Dans la fenêtre de gauche, clique droit sur le nom de notre projet `test_reims`.
  - o Cliquer sur « Properties » qui se trouve tout en bas de la fenêtre s'étant affichée.
  - o Cliquer sur « Project References »
  - o Cocher la case à côté de « veins »
  - o Cliquer sur le bouton « Apply and Close » pour terminer.

Les croix rouges à côté des lignes devraient avoir disparues. Penser également à sauvegarder votre fichier.

#### 4. Création du fichier INI

Comme pour le fichier précédent il est plus facile de se baser sur un fichier INI déjà existant.

- Copier le fichier C:/Users/nom\_user/source/veins-veins-5.2/examples/veins/omnetpp.ini dans le dossier `test_reims`.
- Ouvrir le fichier une fois copier pour procéder à certaines modifications :
  - o Au début du document, première section, changer la valeur du paramètre « network » pour le nom du réseau que nous avons créé dans le fichier `test_reims.ned` :



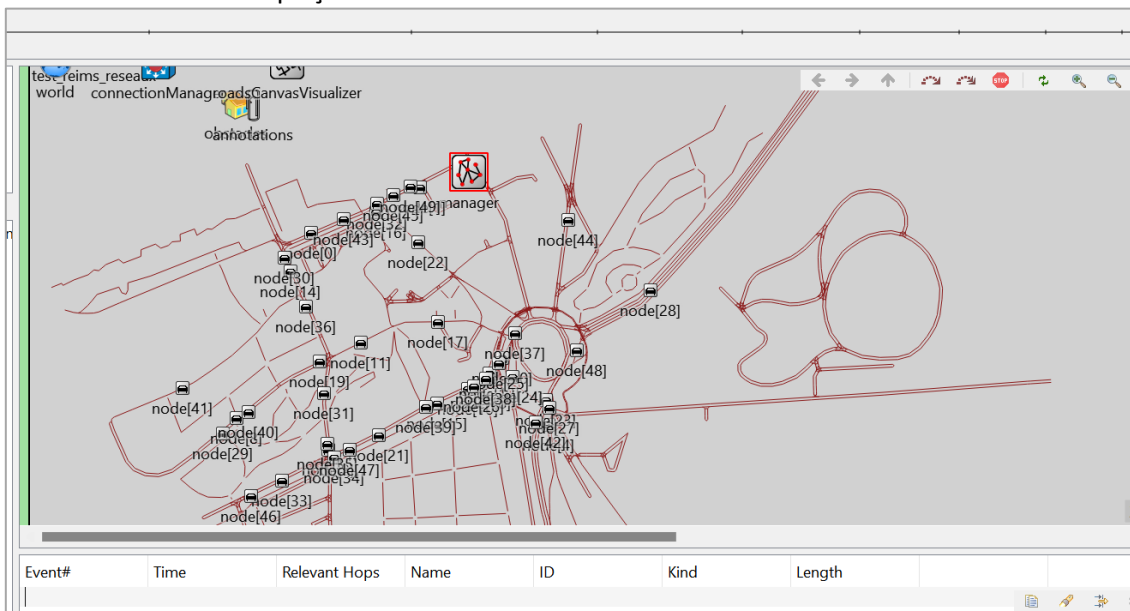
Aucune croix rouge ne devrait apparaître à gauche de la ligne en question. Si c'est tout de même le cas vérifier que le nom corresponde exactement au nom de la fonction présente dans le fichier `test_reims.ned`.

- o Dans la section « TraCIScenarioManager » changer la valeur du paramètre « \*.manager.launchConfig » par « `xmlDoc("test_reims.launchd.xml")` » que nous avons créé ultérieurement.
- o Dans la section « Mobility » changer la valeur du paramètre « \*.node[\*0].veinsmobility.accidentCount » à 0.

SI CA NE FONCTIONNE PAS REVOIR VIDEO [\(1\) Simulation of SUMO Files in OMNeT++ - YouTube](#) 4min10

## 5. Simulation

- Il faut dans un premier temps charger la simulation :
    - o Ouvrir un terminal « cmd » et taper la commande suivante :  
`C:\Users\nom_user\source\sumo-1.8.0\bin\sumo.exe -c C:\Users\nom_user\source\projets\reims.sumocfg`  
La simulation devrait se lancer sur votre terminal.  
Attendre l'apparition du prompt qui indique que la simulation s'est terminée.  
Le fichier comporte de nombreux warnings, ne pas en tenir compte pour le moment.  
L'amélioration de la simulation sera traitée dans un autre document.
  - Lancer SUMO comme expliqué dans [la partie I.3](#).
  - Sur l'IDE OMNeT++, dans la fenêtre de gauche, clic droit sur `test_reims` >> `omnetpp.ini`
  - Cliquer sur `RUN AS >> OMNeT++ Simulation`.
  - Après le chargement cliquer sur « OK » en laissant « Default » comme nom de configuration.
  - Lancer la simulation avec le bouton « Run ».
- Vous devriez voir apparaître la carte de Reims générée dans le document précédent ainsi que des voitures se déplaçant dessus :



Une fois la simulation arrêtée, des statistiques sont disponibles dans le terminal MSYS2.

```
Simulation ended at time: 63.00
Reason: TraCI requested termination.
Performance:
Duration: 22.66s
Real time factor: 2.78084
UPS: 78.658133
Vehicles:
Inserted: 55 (Loaded: 184)
Running: 49
Waiting: 4
Statistics (avg):
RouteLength: 181.82
Speed: 6.77
Duration: 26.83
WaitingTime: 1.17
TimeLoss: 8.16
DepartDelay: 0.00
```