# simITS: An Integrated and Realistic Simulation Platform for Vehicular Networks

Fatma Hrizi
EURECOM, Mobile Communications Department, Sophia-Antipolis, France
fatma.hrizi@eurecom.fr

Fethi Filali
QU Wireless Innovations Center
Doha, Qatar
filali@quwic.com

## ABSTRACT

Vehicular communication systems are becoming one of the most emerging research areas. Many efforts have been made to provide new applications and communication protocols adapted to this new kind of networks. The performance evaluation of these proposals is a crucial step for their validation. Therefore, the need of a new open-source, realistic and integrated simulation environment for vehicular networks is a fundamental and extremely pervasive motivation behind this work. In this paper, we propose simITS, an open-source simulation platform, aiming at a realistic evaluation of protocols for large scale vehicular networks. SimITS aims to simulate properly both road traffic conditions and wireless communications characteristics and at the same time to ensure the real time exchange of simulation data between them. Additionally, a new communication protocol stack designed for vehicular communications has been implemented. Finally, we validate the platform by a basic simulation study.

## Categories and Subject Descriptors

I.6.m [**Computing Methodologies**]: Simulation and Modeling—*Miscellaneous*; C.2.1 [**Computer Systems Organization**]: Computer-Communication Networks—*Network Architecture and Design, Wireless Communication.*

## General Terms

Performance, Design.

## Keywords

Realistic simulation, Network simulation, Road traffic simulation, Wireless vehicular networks.

## 1. INTRODUCTION

In the last few years, a rapid emergence of vehicular networks has been perceived. The premise that vehicular communications can enhance road safety and efficiency has increased the interests of both governments and private entities to support several national and international projects around the globe. Considering the challenges of vehicular networks e.g. the high mobility of nodes, the road and traffic conditions that can influence vehicles propagation, designing efficient communication protocols is crucial since ad hoc networks protocols are not suitable for this kind of wireless networks. Studying vehicular networks often requires an assessment of various network architectures, proposed protocols and applications. These evaluations must take road traffic details and conditions, wireless communication properties and drivers' behavior into consideration. In particular, analytic analysis, simulation, and real word testbeds are possible methods for assessing system performance. Analytic analysis offers a way to reach some basic understandings of system behavior. Although, it often does not require very detailed data - rather statistical information is often adequate - making it especially attractive where detailed information is not readily available. Finally, since large scale realistic experiments (composed by hundreds of equipped vehicles) are not feasible, the performance evaluation of research on vehicular ad hoc networks depends mainly on simulations. In this paper, we provide a new, open-source and realistic simulation tool for vehicular networks. The remaining of this paper is structured as follows. In Section 2, we discuss related works. More details about the current implementation of simITS are given in Section 3. Section 4 presents configurations and simulation results. Finally, Section 5 concludes the paper and provides directions for further research.

## 2. RELATED WORKS

Vehicular simulations require both a network and a mobility component. It is necessary to simulate vehicle movement, radio signal propagation, and protocol behavior. Commonly, the mobility models are created by traffic simulators and fed to a network simulator which is used to model communication and routing data. For example, GrooveSim [10] is a modular integrated traffic and network simulator which uses the approach mentioned above. It includes a variety of mobility models but it lacks of validated communication modules. One key disadvantage of this approach is the inability to modify the traffic data in response to application layer events. There is no way for an application to change a vehicle behavior during runtime. This reduces the level of realism that can be achieved for key applications like active safety which in reality influence the node's movement significantly. There have been some works that addressed this shortcoming and developed integrated simulators achieving the interaction between traffic and network simulation environment. For example NCTUns [11] integrates some traffic features and allows the control of vehicles' mobility. Unfortunately, it does not implement any known traffic mobility model. Moreover, traffic and network simulations are highly integrated, which makes the possibility of extension difficult. TraNS [5] links two open-source simulators SUMO [4] and ns2 [2] using an interface called TraCI [12]. Another proposal called Veins presented in [7]. As TraNS, it couples SUMO and OMNeT++. iTETRIS [1] is also implementing a generic platform to allow for a realistic and accurate evaluation of cooperative vehicular communication systems and traffic management policies under realistic large-scale scenarios.

Our main goal is to use a similar approach in order to design an easy to use and to configure simulation environment that can provide a proper evaluation of newly developed protocols for large scale vehicular networks. Thus, we provide simITS, a new open-source realistic simulation platform, which couples two independent simulation packages SUMO and ns-3 [3]. First, we use ns-3 as network simulator since it ensures large scale simulation and supports emulation and distributed simulation. Second, in order to retrieve traffic data and influence SUMO behavior in real time, we propose a new mobility model for ns-3 and an interface ensuring the interaction between both simulators. Finally, we implement a new communication protocol stack in ns-3 consistent with the current draft of ETSI/ITS [9] to use it instead of existing TCP/IP stack in case of vehicular scenarios.

## 3. SIMITS ARCHITECTURE

The simITS general view is illustrated in Figure 1. The main idea consists in periodically producing realistic traffic traces, based on SUMO, and using them as an input for ns-3. Thus, both simulators run at the same time. Moreover, a realistic flavor is given by allowing ns-3 to control the behavior of SUMO in real time. Mainly, two building blocks have been added to ns-3: the new mobility model *SUMO-MobilityModel* and the novel ITS communication protocol stack.

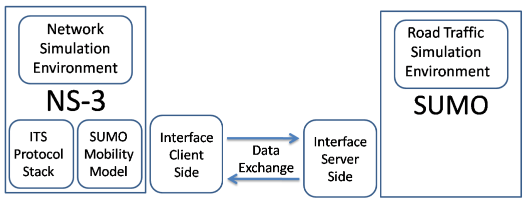

**Figure 1: Interlinking simulators**

In this section, we give a brief description of the simulators that we have used in simITS. Furthermore, we analyze the designed mobility model built on the concept of interaction with the vehicular simulation. Finally, we present implementation details in ns-3 of the new ITS protocol stack which is designed to be used in case of vehicular communications along with the existing IPv4 and IPv6 stacks.

### 3.1 Traffic and network simulation

#### *3.1.1 Vehicular traffic simulation*

One of the most important steps in designing an open-source integrated simulation platform is to find an open-source and adequate vehicular traffic simulation tool. We need to use microscopic simulators since they provide detailed estimates of evolving network conditions by modeling individual drivers' behavioral decisions, different lane changing models and real-life intersection management. Roads, crossings and common traffic rules should be implemented as well. Moreover, another issue that must be considered is the pollutant emissions estimation which is an increasingly important matter when studying vehicular traffic. Also, the capability to influence the behavior or even the movement of cars during simulation is a crucial feature that has to be taken into account. This is ensured by the interactivity of the traffic simulator with the network simulation environment.

One possible solution for traffic simulation is VanetMobiSim [6] which an is open-source featuring new realistic automotive motion models at both macroscopic and microscopic levels. However, it does not support neither accident scenarios nor interaction with network simulation. Another solution can be SUMO which is an open-source microscopic simulator. It is designed on the basis of realistic driver behavior and uses models to estimate pollutant emissions. Furthermore, it supports accident scenarios and offers TraCI which allows full control about most aspect of the road traffic simulation. TraCI aims to interlink road traffic and network simulators. In order to control simulation in real time, it communicates with SUMO via the control-related commands. In addition, the mobility-related commands are used to manipulate vehicles' mobility.

We decided to use SUMO within simITS because of its open and extensible architecture, its compatibility with existing map formats and the interface TraCI that allows the interaction with the network simulation environment.

#### *3.1.2 Network simulation*

The next step for the definition of our platform is the choice of a network simulator. An option that can be considered is OMNeT++ which has a clean design. However, its major drawback is the lack of available protocols and common models in its library, compared to other simulators. Another potential option is the Network Simulator 2, ns-2. Nevertheless, ns-3 seems to us the suitable network simulator candidate since it provides various protocol entities designed to be closer to real computers (e.g. true IP stack) and supports network virtualization and testbed integration. Moreover, it includes sophisticated simulation features and an object oriented design for rapid coding and extension. Ns-3 has a stable design for improved scalability which is the major concerns about ns-2. It is capable to simulate large scale scenarios and high traffic densities. Being open also to commercial use, its software architecture is well structured and enables attaching software modules for data exchange with other programs and is thus ideal for setting up a coupled simulation environment and for designing a new protocol stack.

### 3.2 Mobility model: Interaction with vehicular simulation

In order to give a realistic behavior to simITS, we propose to ensure the interaction between SUMO and ns-3 by the mean of an enhanced interface that relies at its core on TraCI. TraCI is dedicated basically to ns-2. Therefore, we have changed its implementation in order to make it more generic and independent of any network simulator and modify it to fit our requirements with the purpose of the interfacing between SUMO and ns-3. On the other hand, we have designed a novel mobility model namely *SUMOMobilityModel* in ns-3 which makes use of this adapted interface. It is basically responsible for controlling, influencing and obtaining mobility traces from SUMO. The classes involved are visualized in Figure 2. *SUMOMobilityModel* class inherits from the generic ns-3 class *MobilityModel* that keeps track of the current position of a node. Furthermore, it defines appropriate methods to update ns-3 simulation parameters using received data from SUMO. *SUMOHelper* is used by *SUMOMobilityModel* to update position data of each node. To comply with the ns-3 conceptual model, we have designed a helper *SUMOMobilityHelper* for our new mobility model in order to facilitate its use in simulation scenarios.

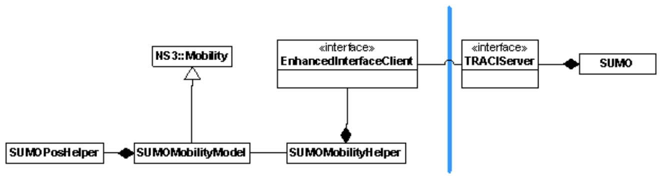

**Figure 2: *SUMOMobilityModel* class diagram**

As mentioned above, ns-3 retrieves traffic data from SUMO and, giving this data, updates nodes' position information. Moreover, it can control the simulation in SUMO by the mean of mobility commands provided by the amended interface. In the following sections, we give more details about the procedure of position update. In addition, we point out how to use mobility commands in order to influence SUMO behavior e.g. to trigger an accident in SUMO.

### *3.2.1 Position update*

The simulation time is split into small time steps in order to update vehicles' position in ns-3 at each one of these time points. Therefore, the operation of the network simulator ns-3 is controlled by *Command Simulation Step* which asks SUMO to perform the traffic simulation until a given time. With the intention of providing a more realistic simulation, we have set this interval to 1s. Then, the simulation is performed in the following way: before the beginning of a simulation run, we initialize the simulation start time, the simulation end time which represents the time until which the traffic and the network simulations will run. A connection between the server and the client sides of the road traffic interface is eventually opened in order to ensure the interaction in real time with SUMO. At each time step, *SUMOMobilityModel* updates the position of all participating nodes in the simulation by collecting different parameters from SUMO via *Command Simulation Step*. Algorithm 1 shows the pseudo code representing the program to update ns-3 nodes' position.

### *3.2.2 Triggering accident*

SUMO uses a collision-free traffic flow model. So, in an ordinary scenario, accidents do not occur. However, since ns-3 can control SUMO using mobility commands, such as stop, change lane, change speed and so on, it is possible to simulate accidents by making a vehicle stop at a predefined position. To do so, we set the maximum speed of a specific vehicle at a given simulation step to zero using the *Set maximum Speed* command. Accordingly, the vehicle stops during simulation run time. A potential accident may happen among vehicles traveling in the same direction.

**Algorithm 1** Pseudo code of the update positions

```
{Initialize simulation start and end time}
Stime = startTime
ETime = endTime
{Initialize simulation timeStep}
timeStep = 1
{Open connection with the Traffic Simulation Interface server}
startSimStepHandler()
while (Stime < ETime) do
   CommandSimumationStep()
   UpdatePosition()
   Stime = Stime + timeStep
end while
{closing connection with the server and finishing simulation}
CommandCloseConnection()
```

## 3.3 ITS communication protocol stack

To support safety as well as non-safety applications for wireless vehicular networks, it is crucial to design communication systems overcoming the challenges and fitting the requirements of such type of network. In this context, we propose to integrate in ns-3 a novel communication protocol stack for vehicular communications, as depicted in Figure 1, compliant with the draft of ETSI/ITS namely, ITS protocol stack. It is designed to replace the existing TCP/IP stack in case of vehicular safety applications. Figure 3 illustrates the class diagram of the ITS stack. Basically, a safety application, a transport protocol, a network beaconing protocol and a network protocol have been implemented. In the following, we provide a detailed description of the different layers of the stack.

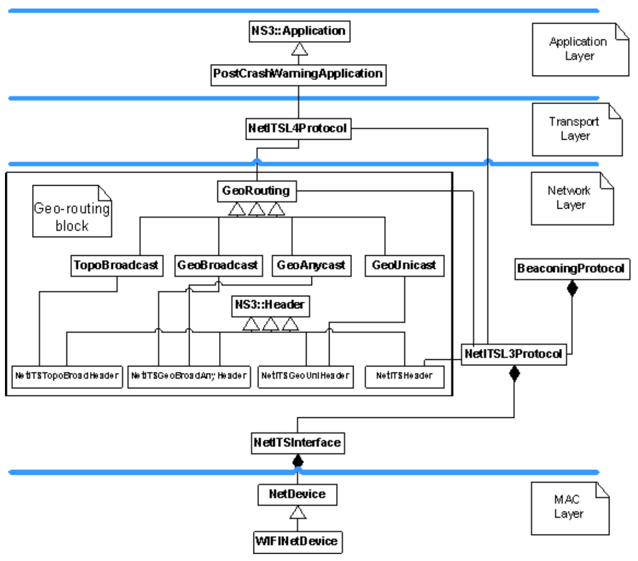

**Figure 3: Class diagram of the ITS communication protocol stack**

### 3.3.1 *Application layer*

The current design of the stack aims basically to support safety applications. This can be ensured by two means: by the periodic transmission of status messages of each node and by the dissemination of safety messages once a potential danger has been detected. Accordingly, we consider a scenario where network beacons are sent periodically. Moreover, in case of hazard detection, a dissemination of emergency messages is triggered. Therefore, we have designed a *PostCrashWarning* application [8] which corresponds to a road safety application which consists in alerting other approaching vehicles of the risk when an accident is happening using a multi-hop strategy. The geographic propagation distance, which is expressed in terms of distance from the originator, has to be specified. The safety information should be propagated as quickly as possible along the road into larger geographical areas than the direct communication range. *PostCrashWarning* class inherits from the generic class ns-3 Application.

### 3.3.2 *Geo-routing and Network layer*

From Figure 3, we can notice that the current implementation of ITS stack comprises geo-routing protocols. Depending on the destination, several geo-routing schemes are used such as geo-unicast, geo-broadcast, geo-anycast and topology broadcast. The current implementation of geo-routing in our ITS stack is limited to geo-broadcast and topo-broadcast, other geo-routing schemes are planned to be implemented in future work. Geo-broadcast and topo-broadcast regards both delivering data from a source to all nodes within a specific area. This area is specified in terms of hops in topo-broadcast and in case of geo-broadcast, is determined according to the geographic distance from the sender. All the geo-routing classes inherit from *GeoRouting* and a specific geo-routing header is associated with each class e.g. both *GeoBroadcast* and *GeoAnycast* use *NetITSGeoBroadAnyHeader* since they have the same header format. In case of packets transmission, the transport protocol *NetITSL4protocol* requests a route from the geo-routing protocol. The route and the packet with the added geo-routing header are then passed to *NetITSL3protocol* which, in his turns, transmits the packet using the 802.11 MAC layer after adding the ITS network header. When a packet is received, the geo-routing protocol decides either to deliver it to upper layer, to forward it or to discard it. We have implemented also a network beaconing protocol represented by *BeaconingProtocol* class which is responsible for exchanging position information periodically in a single-hop mode. Additionally, each node maintains a location table including location related data for itself and a list of its neighboring nodes e.g. geographic position, speed, time stamp etc. At a given frequency, every node sends beacons. When a node receives a packet, it updates its location table using the network header of the received packet.

## 4. VALIDATION

In this section we show how to make use of simITS. We first describe the main parts of ns-3 simulation script, then we present some basic simulation results aiming at validating the implemented mobility model and the ITS protocol stack.

### 4.1 Example of simulation script

There are two basic simulation scenarios that must be set up, the traffic and the network simulation scenarios. In the following, we give more details about the use and the configuration of *SUMOMobilityModel* and ITS stack in ns-3. Listing 1 presents the main parts of ns-3 simulation script. The first part of the script consists in creating nodes and configuring MAC and physical layers for each device. The second part is dedicated to the configuration of nodes' mobility and the aggregation of *SUMOMobilityModel* to each node. Then, *NetITSStackHelper* and *InterfaceHelper* are used to install ITS protocol stack. Once we have built the topology and aggregated protocol stack to nodes, we need now to set up applications. *PostCrashWarningAppHelper* aggregates the Post Crash Warning application to the nodes. Subsequently, the simulation is started for a given time.

**Listing 1: Mobility model and ITS configuration**

```
//Create nodes and configure MAC and physical layers
NodeContainer wifiStaNodes;
wifiStaNodes.Create(nWifi);
YansWifiChannelHelper channel = YansWifiChannelHelper::Default();
YansWifiPhyHelper phy = YansWifiPhyHelper::Default();
phy.SetChannel(channel.Create());
 ...
//Configure mobility
SUMOMobilityHelper mobility;
mobility. Install (wifiStaNodes);

//Install ITS protocol stack
NetITSStackHelper stack;
stack. Install (wifiStaNodes);
InterfaceHelper address;
address.Assign(wifiStaDevices);

//Install application
PostCrashWarningAppHelper app = PostCrashWarningAppHelper();
ApplicationContainer apps = app.Install(wifiStaNodes);

//Run simulation
apps.Start(Seconds(1.0));
Simulator::Run();
 ...
```

### 4.2 Beaconing simulation results

In this section, we provide simulation results that validate the new vehicular stack by a simple beaconing protocol. In this scenario, we used the new mobility model *SUMOMobilityModel* and, consequently, the retrieved traffic data (position, speed...) to create realistic data packets.

**Listing 2: Tracing beacons**

```
+ 1000000000ns /NodeList/0/DeviceList/0/$ns3::WifiNetDevice/Phy/Tx
      ns3::WifiMacHeader (DATA ToDS= , FromDS= , MoreFrag= , Retry= ,
      MoreData= Duration/ID=0usDA=ff:ff:ff:ff:ff:ff, SA=00:00:00:00:00:01,
      BSSID=ff:ff:ff:ff:ff:ff, FragNumber= , SeqNumber=0)
      ns3::LlcSnapHeader (type 0x999) ns3::NetITSHeader (****ITS Network
      Header**** source ID: 0 source TS: 1 source Latitude: 6 source
      Longitude: 1835909400 source Speed: 0 packet length: 40)
      ns3::WifiMacTrailer ()
r 1000128054ns /NodeList/1/DeviceList/0/$ns3::WifiNetDevice/Phy/RxOk
      ns3::WifiMacHeader (DATA ToDS= , FromDS= , MoreFrag= , Retry= ,
      MoreData= Duration/ID=0usDA=ff:ff:ff:ff:ff:ff, SA=00:00:00:00:00:01,
      BSSID=ff:ff:ff:ff:ff:ff, FragNumber= , SeqNumber=0)
      ns3::LlcSnapHeader (type 0x999) ns3::NetITSHeader (****ITS Network
      Header**** source ID: 0 source TS: 1 source Latitude: 6 source
      Longitude: 1835909400 source Speed: 0 packet length: 40)
      ns3::WifiMacTrailer ()
 ...
```

#### 4.2.1 *Tracing beacons*

As we can see from Listing 2, node 0 sends first its beacon which is received by its neighboring vehicles. The first neighbor that receives the message is node 1. Then, node 1, in turns, triggers its beacon transmission. Periodically, this procedure is processed by all the nodes in the network. The basic information carried by the beacon is the identifier of the node, the geographic position and speed which constitute the ITS network header fields.

#### 4.2.2 *Using simITS to evaluate beaconing protocol performance*

We used a highway scenario composed of a simple one way road with length of 10km. Each vehicle is assigned a maximum speed of 42 m/s. In network scenario, we used the Logarithmic Distance propagation loss model. We configured each vehicle to send beacon in a shared wireless channel using IEEE802.11a with a data rate of 6 Mbps. The default implemented physical layer in ns-3 is used. We define the beacon generation rate (BGR) as the number of beacons sent per second. Each studied configuration is simulated for 60s. The performance metric chosen is the Beacon Delivery Rate (BDR) which is the percentage of vehicles that successfully received a packet amongst all vehicles positioned at a specific distance from the sender.

*Impact of beacon generation rate.*

Figure 4 illustrates BDR with respect to the distance from the transmitter and presents the results for four different BGR: 2, 4, 10 and 20 beacons/s. A constant transmission power is used to send beacons which corresponds to the default configuration of ns-3 (16 dbm). With respect to the different BGRs, as expected, a higher amount of generated beacons turns into a higher saturation on the medium. This simulation results demonstrate that the BGR 2 beacons/s performs better than the other BGRs in terms of BDR.

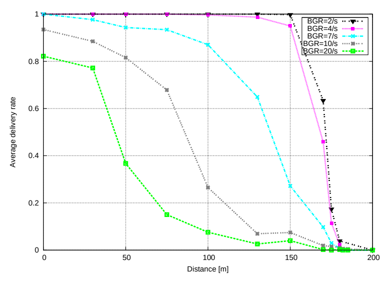

**Figure 4: Average BDR vs. distance varying BGR**

*Impact of transmission power.*

To clearly observe the impact of adjusting the transmission power, we set up a scenario with a fixed BGR 2 beacons/s and a data rate of 6 Mbps. The simulation has been carried out with six different transmission power: 5 dbm, 10 dbm, 15 dbm, 20 dbm, 30 dbm and 40 dbm. Figure 5 depicts the average BDR regarding the distance from the source node. We can point out that, while the channel is not saturated, increasing the transmission power does not decrease BDR at close distances, and provides improved reception rates at further ones. However, with a saturated channel (40 dbm), there exist a number of messages transmitted from nodes at close distances which can not be captured due to interferences. The BDR at close distance from the sender (less than 200m) is reduced in case of 40dbm compared to 30dbm.

## 5. CONCLUSIONS

This paper has given an insight about simITS and its different components. We have presented both mobility model and ITS communication stack. The main aim of simITS is to assess vehicular applications and protocols in a high level of

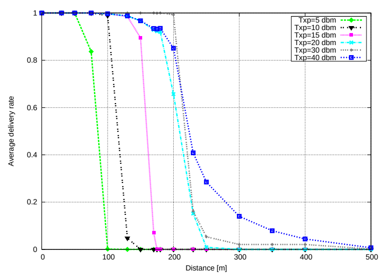

**Figure 5: Average BDR vs. distance varying transmission power**

realism. Therefore, simITS can be exploited by researchers working in the area of wireless vehicular communications to validate and evaluate the performances of communication protocols they propose for large scale wireless vehicular networks. Our platform may guide its user to better understand challenges in vehicular environment and to enhance proposed protocol and adopt it to the requirements of such networks. We plan to integrate additional features in the ITS stack in order to support additional types of applications such as traffic efficiency and value-added services.

## 6. REFERENCES

[1] iTETRIS official website. `http://ict-itetris.eu/`.
[2] Ns-2 official website. `http://www.isi.edu/nsnam/ns/`.
[3] Ns-3 official website. `http://www.nsnam.org/`.
[4] Sumo official website. `http://sumo.sourceforge.net/`.
[5] Trans official website. `http://trans.epfl.ch/mediawiki/index.php/Main_Page`.
[6] Vanetmobisim official website. `http://vanet.eurecom.fr`.
[7] Veins official website. `http://www7.informatik.uni-erlangen.de/veins/`.
[8] R. Bossom et al. European ITS communication architecture - Overall framework - Proof of concept implementation. Technical report, EC FP7 Deliverable, EC "Information Society Technologies" Programme, Mar 2009.
[9] Draft ETSI TS102 636-4-1 V0.0.5. Intelligent Transportation Systems (ITS); Vehicular Communications; Part 4: Geographical Addressing and Forwarding for Point-to-Point and Point-to-Multipoint Communications; Sub-part1: Media-Independent Functionality, Jan 2010.
[10] R. Mangharam et al. GrooveSim: a topography accurate simulator for geographic routing in vehicular networks. In *Proc. of the 2nd ACM International Workshop on Vehicular Ad hoc Networks (VANET'05)*, Sep 2005.
[11] S. Wang et al. NCTUns 4.0: An Integrated Simulation Platform for Vehicular Traffic. In *Proc. of the 1st IEEE International Symposium on Wireless Vehicular Communications (WiVec'07)*, Oct 2007.
[12] A. Wegener et al. TraCI: An Interface for Coupling Road Traffic and Network Simulators. In *Proc. of the 11th Communications and Networking Simulation Symposium (CNS'08)*, Apr 2008.