



Advanced Seminar Embedded Systems 2008/2009

# **Simulation of wireless ad-hoc sensor networks with QualNet**

DOCUMENTATION

by Tobias Doerffel

Chemnitz, April 9, 2009

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	The scenario . . . . .	3
<b>2</b>	<b>QualNet overview</b>	<b>4</b>
2.1	Components of QualNet . . . . .	4
2.2	Installing QualNet on a recent Linux system . . . . .	4
2.3	Installing QualNet on Windows . . . . .	5
<b>3</b>	<b>Designing the scenario</b>	<b>6</b>
3.1	Data sink . . . . .	8
3.2	Mobile sensor nodes . . . . .	8
3.3	Wireless subnet . . . . .	8
3.4	Traffic generators . . . . .	9
<b>4</b>	<b>Running the scenario</b>	<b>11</b>
4.1	Graphical animation . . . . .	11
4.2	QualNet 3D Visualizer . . . . .	11
4.3	Command line interface . . . . .	13
<b>5</b>	<b>Analyzing the results</b>	<b>14</b>
<b>6</b>	<b>Conclusion</b>	<b>16</b>

# 1 Introduction

Nowadays network applications – especially in the area of wireless networks – are becoming more and more complex which makes the design and the testing almost impossible without appropriate software. There're a lot of different programs available to aid the user in simulating previously designed scenarios. This work covers the implementation of a given scenario using the software *QualNet* by *Scalable Networks* in version 4.5.1.

QualNet is a commercial software that runs on all common platforms (Linux, Windows, Solaris, OS X) and is specialized in simulating all kind of wireless applications. It has a quite clear user interface compared to other solutions while also offering an easy to use command line interface.

## 1.1 The scenario

We had been given the following wireless scenario to be implemented:

- sensor network with 10 mobile nodes and a statically placed data sink (root node)
- IEEE 802.15.4 wireless standard for PHY and MAC layer
- AODV (Ad hoc On-Demand Distance Vector) routing <sup>1</sup>
- Random Waypoint Mobility Model for the mobile nodes <sup>2</sup>
- 2 KB/s unidirectional continuous network load from each sensor node to data sink
- time synchronization with the root node every 10 seconds using the FTSP (Flooding Time Synchronization Protocol) <sup>3</sup>
- optional:
  - TCP/IP stack integration with UDP load (e.g. videostream) from root node to all nodes (10 KB/s)
  - acknowledged sensor data

It however was not possible to meet all these requirements due to missing features in the program. Details will follow.

---

<sup>1</sup><http://tools.ietf.org/html/rfc3561>

<sup>2</sup><http://mathstat.helsinki.fi/mathphys/EVERGROW/virtamo.pdf>

<sup>3</sup><http://www.eecs.harvard.edu/mdw/course/cs263/papers/ftsp-sensys04.pdf>

## 2 QualNet overview

### 2.1 Components of QualNet

QualNet consists of various components:

- QualNet Scenario Designer
- QualNet Animator
- QualNet 3D Visualizer
- QualNet Analyzer
- QualNet Packet Tracer

All components but the last will be described in detail in the according chapter.

### 2.2 Installing QualNet on a recent Linux system

Getting to run QualNet on recent Linux systems is a bit tricky as it has been built for rather outdated Linux distributions (Fedora Core 4 which has been released in June 2005). All steps described in the following refer to recent Debian based distributions but probably are applicable for other distributions as well.

First of all you need to register an account in order to download the software archive at <sup>4</sup>. Afterwards the archive can be extracted:

```
sudo su
cd /opt
tar xzf <download directory>/qualnet-4.5.1-evaluation.tar.gz
```

Next the QualNet core application has to be compiled. Since the provided source code does not follow the official C++ standard (ISO/IEC 14882:1998) it can't be compiled without some small modifications. An according patch named *fix-qualnet-core-sources.diff* can be found in the archive containing all files related to this work. To apply it and compile everything run the following commands:

```
sudo su
cd /opt/qualnet/4.5/
patch -p2 < ...../fix-qualnet-core-sources.diff
cd main
cp Makefile-linux-glibc-2.3-gcc-4.0 Makefile
make
```

---

<sup>4</sup><http://www.scalable-networks.com/products/qualnet/download.php>

In case your system is 64 bit you need to replace "Makefile-linux-glibc-2.3-gcc-4.0" with "Makefile-linux-x86\_64-glibc-2.3-gcc-4.0" in the above commands. Now some symlinks need to be created due to the already mentioned linking against old libraries.

```
sudo su
cd /usr/lib
ln -s libexpat.so libexpat.so.0
ln -s libssl.so libssl.so.5
ln -s libcrypto.so libcrypto.so.5
```

The last thing to do at this stage is to download the license file (qualnet-4.5.1-eval-YYYY.MM.DD.lic) from the same URL as the software and place it in `/opt/qualnet/4.5/license_dir`.

If everything finished without errors, QualNet is ready to run.

## 2.3 Installing QualNet on Windows

The installation of QualNet on Windows should be quite straight-forward via the installer and thus is not described in detail here. The installation brings a pre-compiled qualnet.exe so there's no need to compile components of QualNet on your own.

## 3 Designing the scenario

Given QualNet has been started successfully, a new empty scenario has to be created using a wizard which can be started via *File*→*New*. After it has finished a workspace with an empty scenario can be seen as shown in figure 1.

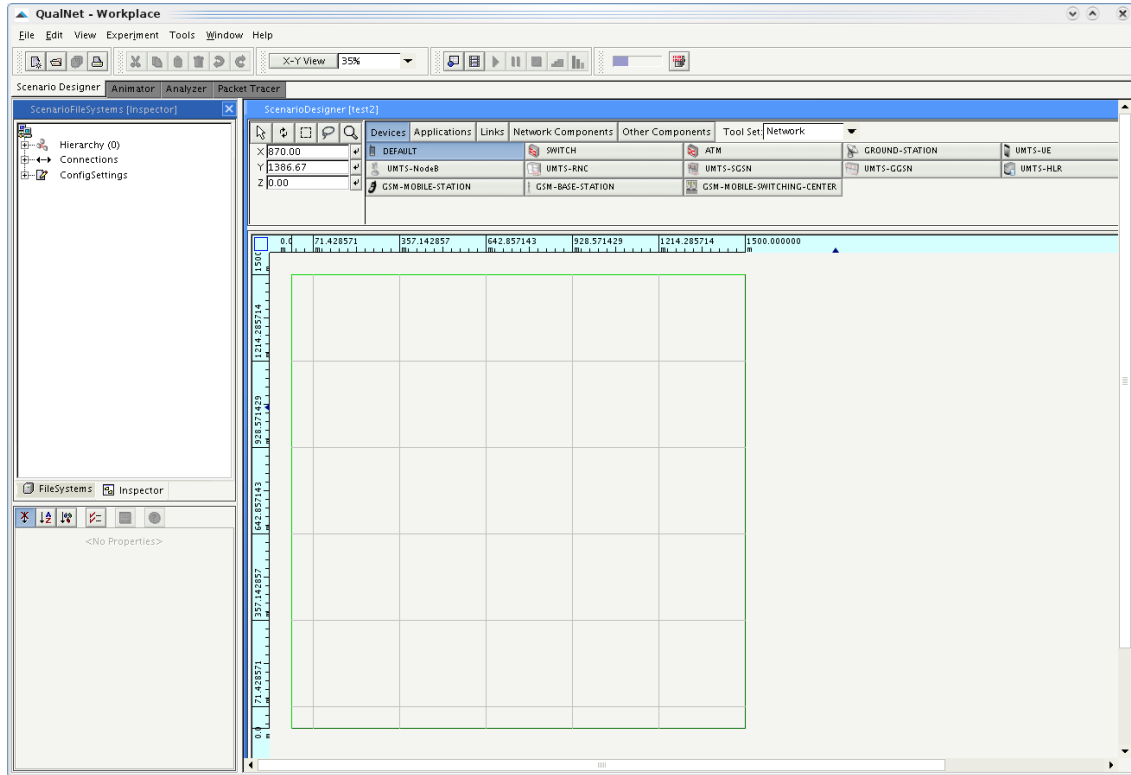


Figure 1: QualNet Designer: an empty scenario to start with

All properties of the scenario can be viewed and changed using the Inspector on the left side. There's an amazing amount of properties, subproperties and parameters that can be altered although we only need to touch a very few of them.

For not having to change common properties for each node there's a possibility to set global properties below *ConfigSettings*. In particular only some of the items are of interest in the following. The changes as listed below were made to configure the nodes according to the requirements:

- *Wireless Settings*→*Radio/Physical Layer*→*Radio Type*: 802.15.4 Radio
- *Wireless Settings*→*MAC Protocol*→*MAC Protocol*: 802.15.4
- *Network Protocols*→*Routing Protocol*→*Routing Policy*→*Routing Protocol for IPv4*: AODV

- *Network Protocols*→*Network Protocol*→*IP Fragmentation Unit*: 70
- *Node Positioning*→*Mobility*→*Mobility Model*: Random Waypoint
- *General*→*Terrain*→*Coordinate-System*→*Dimensions*: 100 100

Now we can start adding nodes by selecting *Devices*→*DEFAULT* in the toolbar and clicking in the workspace. This is merely useful in scenarios with only a few nodes.

A vast number of nodes can be created easily utilizing *Experiment*→*Place Nodes*. In the dialog the number of nodes to be created can be set - in our case 11 nodes. Keeping the placement strategy *Uniform* works quite well. After clicking *Finish*, 11 new nodes should appear on the workspace. The first node (which will be our data sink) should be dragged to the center of the workspace (make sure to select the move-tool before) so the average distance to all the other mobile nodes will be minimal.

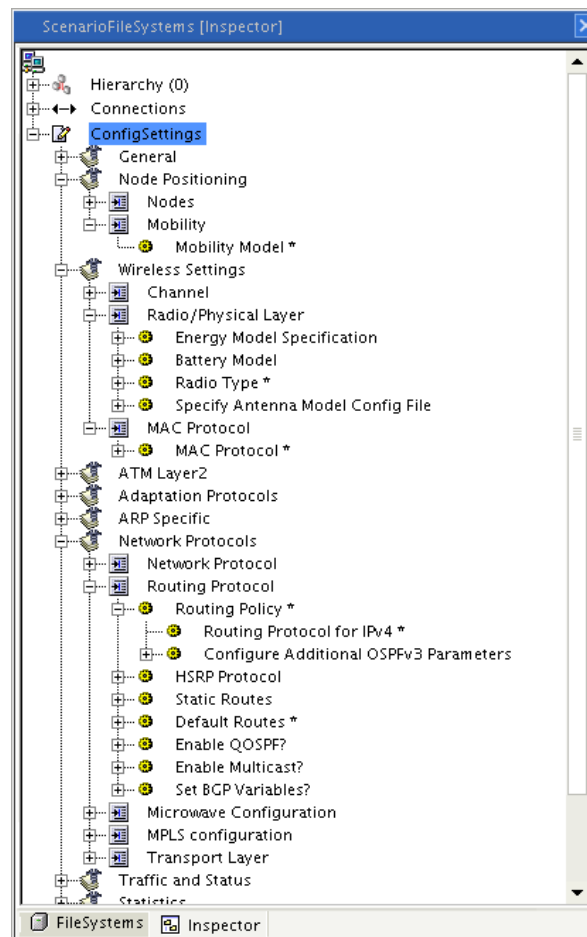


Figure 2: Inspector with all modified global properties

## 3.1 Data sink

The data sink differs from the other (sensor) nodes only regarding mobility and its role in the 802.15.4 network. Therefore the following properties below *Hierarchy(0)* → *Nodes* → *host1* → *Node Configuration* explicitly have to be overridden:

- *Mobility* → *Mobility Model*: None
- *MAC Protocol* → *MAC Protocol*: 802.15.4
- *MAC Protocol* → *MAC Protocol* → *Device Type*: Full Function Device

The latter two settings are required to make the root node act as PAN Coordinator which is required in every 802.15.4 wireless network. The other nodes act as so called Reduced Function Devices (RFD).

## 3.2 Mobile sensor nodes

No further changes have to be made for mobile sensor nodes although it's possible to tune various interesting parameters in later experiments, for example transmission power or energy model specifications.

## 3.3 Wireless subnet

Until now all the nodes will only communicate on PHY and MAC layer. Therefore a wireless subnet has to be added to the scenario. The most convenient way to achieve this is selecting all hosts either using the rectangular- or the lasso selection tool. Then enable *Network Components* → *Wireless Network* in the toolbar and click somewhere into the selected region. This will add a new wireless subnet and automatically attach all hosts to it.

Now the PHY and MAC layer need to be configured by modifying some items below *Hierarchy(0)* → *Nodes* → *Wireless Subnet N8-192.0.0.0*:

- *Wireless Subnet Properties* → *Radio Type* → *Radio Type*: 802.15.4 Radio
- *Wireless Subnet Properties* → *MAC Protocol* → *MAC Protocol*: 802.15.4

### 3.4 Traffic generators

The last important thing that is still missing in our scenario are traffic generators so that we can actually do some sensible simulations. For this select *Applications*→*CBR* from the toolbar in the designer view. Via drag'n'drop it's possible to link all sensor nodes with the root node. Make sure to drag **from** sensor node **to** root node so that there's an unidirectional traffic to root node. After finishing the scenario looks like in figure 3.

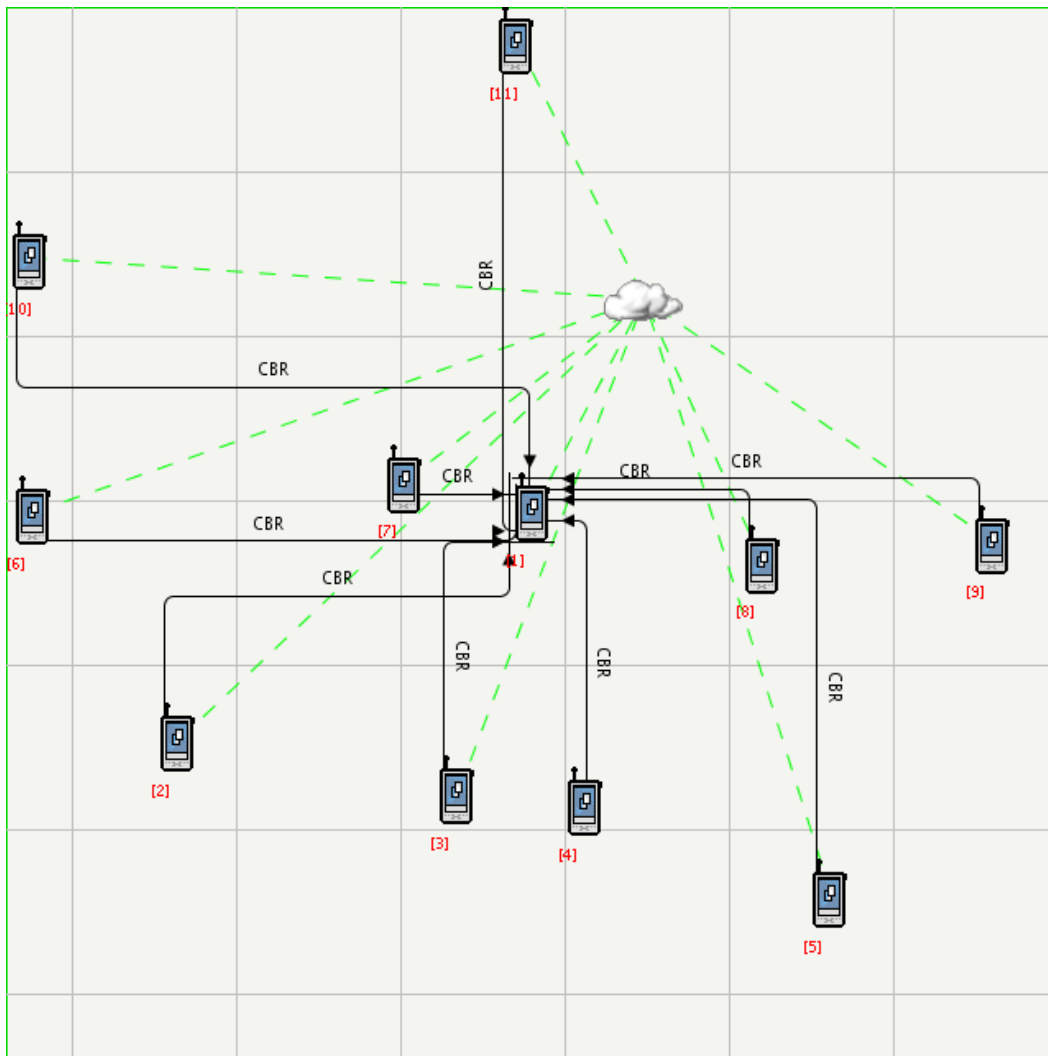


Figure 3: Final scenario with traffic generators

### 3.4 Traffic generators

---

Finally the traffic generators have to be configured according to the scenario specifications. For each traffic generator in the inspector change the following properties below *Connections*→*CBR XX-1*:

- *Items to Send*: 30
- *Item Size (bytes)*: 2048
- *End Time*: 0
- *Traffic type*→*Data Size*→*Fixed Size*: 2048

As a last tweak the simulation time (*ConfigSettings*→*General*→*Simulation Time*) should be altered to 300 seconds as per default the simulation runs faster than realtime and thus would be finished just after a few seconds.

Now the scenario should be ready to run.

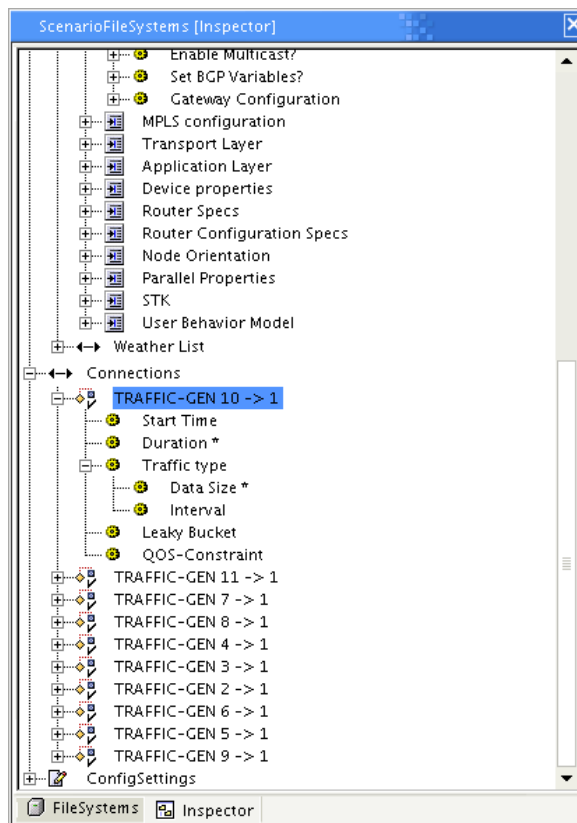


Figure 4: Inspector with properties of traffic generators

## 4 Running the scenario

There are various ways of running the scenario – depending on the results you want to have. Every procedure has its own advantages and disadvantages.

### 4.1 Graphical animation

To give an idea of how (in terms of quantitative) the scenario performs it can be run using the QualNet Animator. Figure 5 shows the Animator with the previously designed scenario in action. On the right side you can enable or disable various types of animations. In the *Layers* tab animations for each of the 7 OSI layers can be enabled or disabled individually.

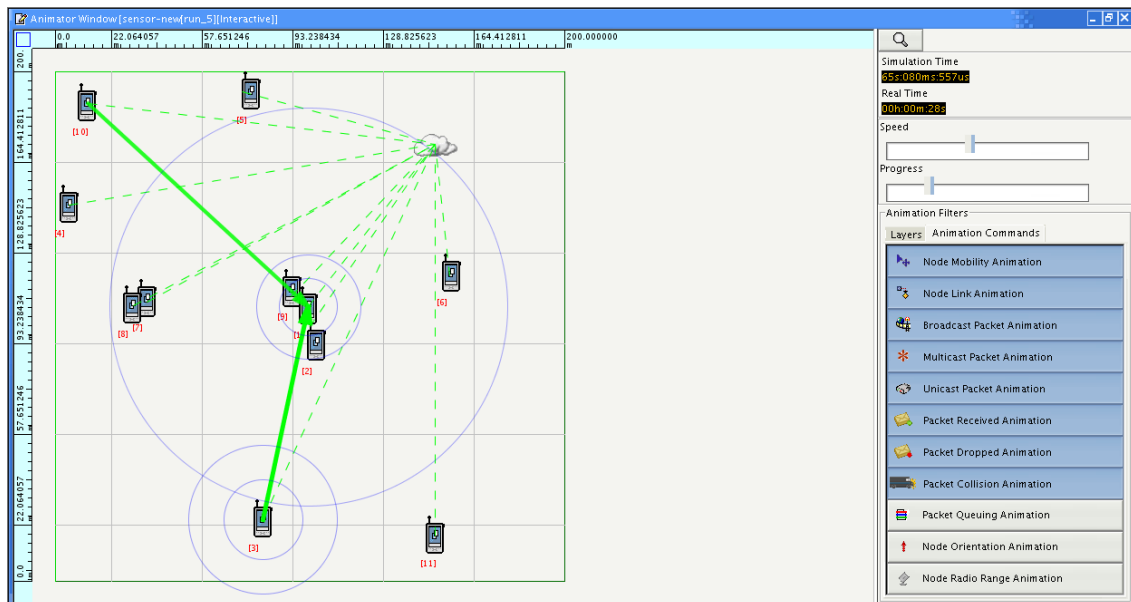


Figure 5: QualNet Animator in action

The worst drawback of the QualNet animator is its extrem high CPU utilization and its implementation in Java which makes it run very slowly on most machines.

### 4.2 QualNet 3D Visualizer

To bypass the performance limitations of the Animator you can use the QualNet 3D Visualizer which makes use of OpenGL accelerated graphic cards. It also allows various threedimensional views at the running scenario. In figure 6 and figure 7 typical 3D Visualizer sessions are shown.

## 4.2 QualNet 3D Visualizer

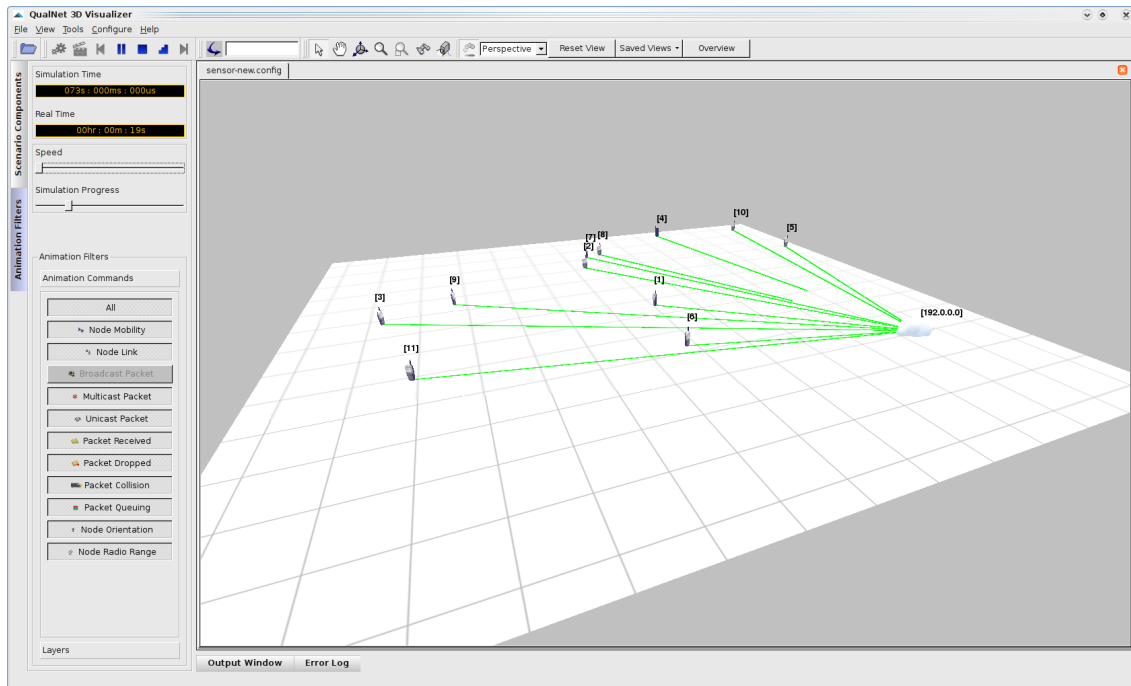


Figure 6: 3D Visualizer

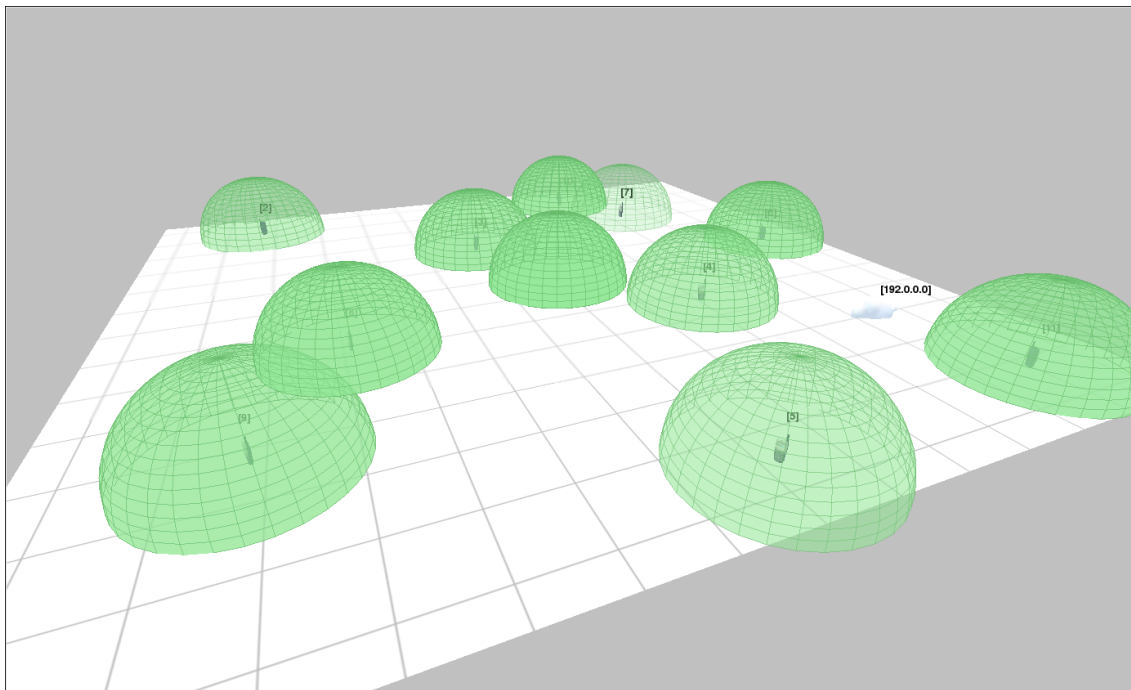


Figure 7: 3D Visualizer with broadcasts enabled

### 4.3 Command line interface

While offering a lot of easy-to-use graphical frontends, QualNet also brings a command line interface which is mainly useful if you're only interested in the actual statistical data of a simulation.

When creating a scenario using the Designer, besides the project file a subdirectory with the same name is being created and contains various other files related to simulation. For using the command line interface the file `<project name>.config` is of importance. This config file is a standard text file and settings can be changed easily by just editing this file. To run the scenario described in a config file, the QualNet CLI can be run as follows:

```
cd <path to scenario folder>/<project name>  
<path to qualnet>/bin/qualnet <project name>.config
```

If not specified differently, this will generate a file called *Qualnet.stat* which contains all statistical data related to the simulation. It can be viewed using the QualNet Analyzer as described in the next chapter.

## 5 Analyzing the results

The most interesting part probably is analyzing the statistical data generated during simulation. Depending on the configuration scenario statistics for all OSI layers can be viewed separately. Clicking the *Analyzer* tab at the top will bring up the QualNet Analyzer. Inside the subdirectory belonging to the current scenario there should be some files called "Qualnet....stat" - if not, the scenario hasn't been run before. Double clicking the according statistic file will open the Analyzer which allows you to browse through the statistics of the different OSI layers. Selecting items will show informative graphs such as shown in figure 8 and figure 9.

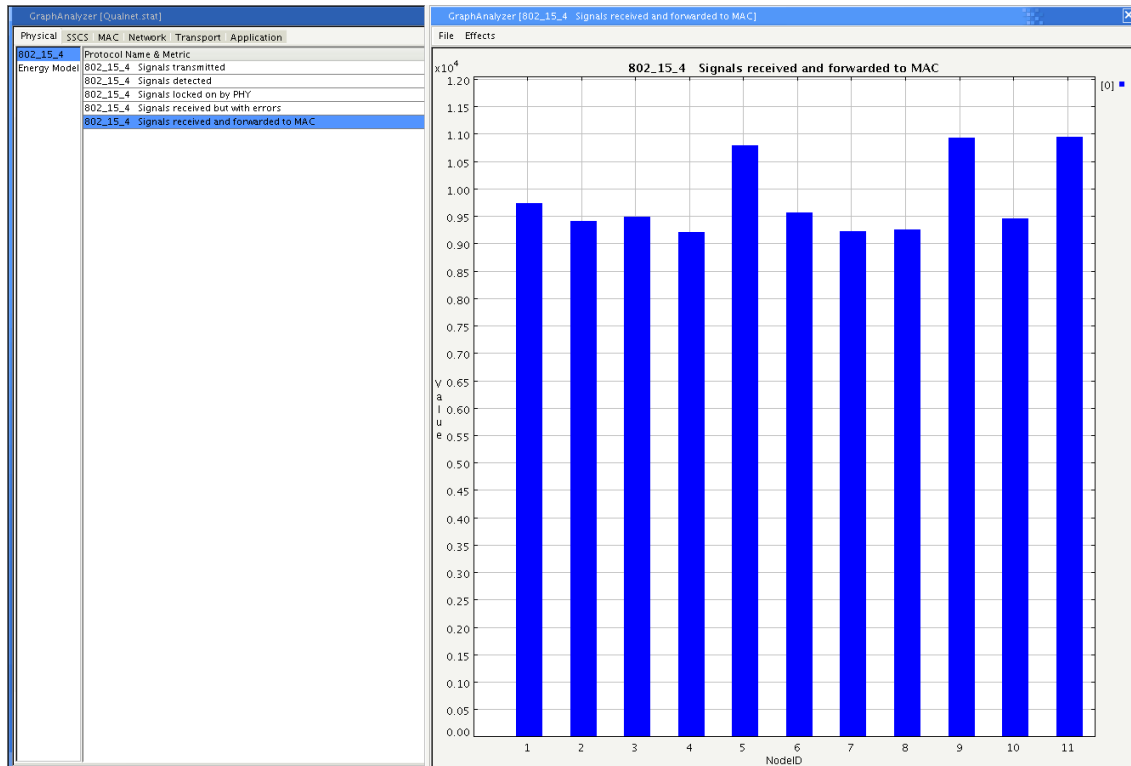


Figure 8: Analyzer with chart of signals received and forwarded to MAC

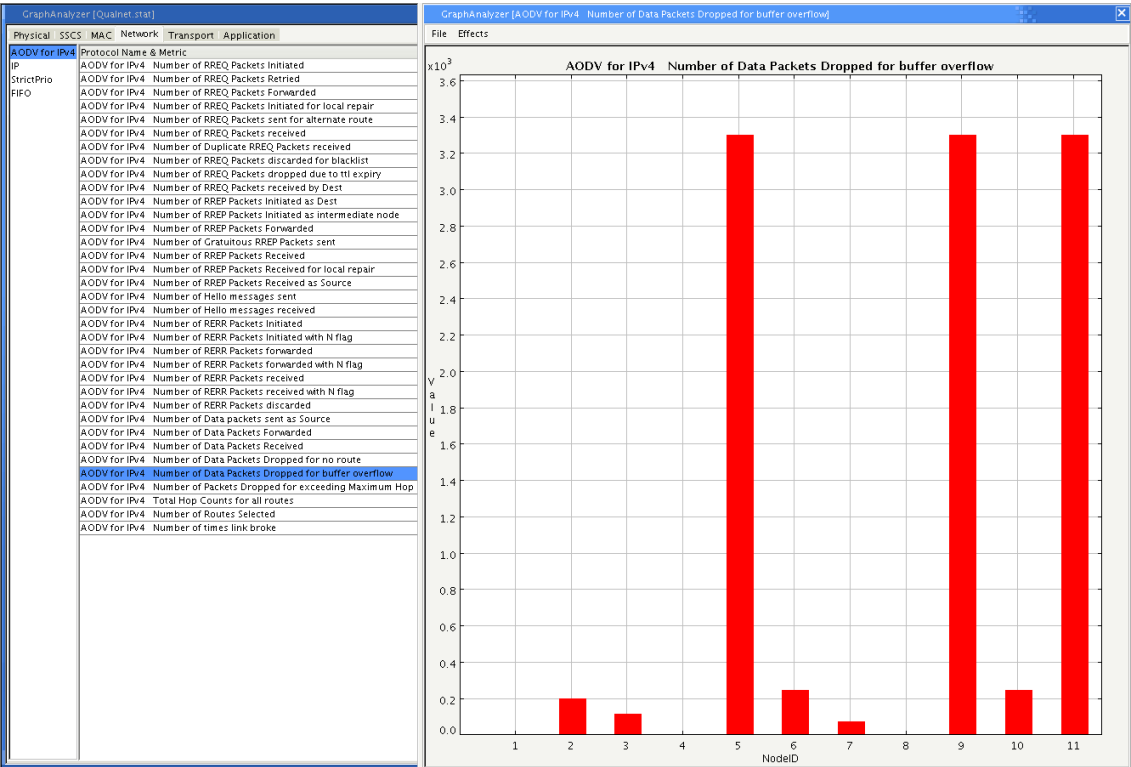


Figure 9: Analyzer with chart on AODV routing statistics

## 6 Conclusion

Without having compared QualNet to other similar solutions various advantages became apparent while using it. The most important ones are:

- easy-to-use and clear UI
- wide range of possible applications (even WiMAX MAC layer is supported)
- support for multiprocessor systems and distributed computing
- sophisticated animation capabilities
- extensive possibilities for analyzing scenario
- shipped with a lot helpful documentation and tons of example scenarios

There're some disadvantages as well:

- difficult installation on Linux
- slow Java-based UI
- very expensive (couldn't find any specific figure) – however there's a special QualNet University Program which - according to <sup>5</sup> - allows to get a license for about 3000 Euro per year.

---

<sup>5</sup><ftp://ftp.tik.ee.ethz.ch/pub/publications/TIK-Report-255.pdf>, page 11