A MANET WITH CACHING CAPABILITIES VISUALIZATION TOOL

F.J. González-Cañete, L.B. Ríos-Sepúlveda, E. Casilari, and A. Triviño-Cabrera
Dpto. Tecnología Electrónica, University of Málaga
ETSI Telecomunicación, Campus de Teatinos, Universidad de Málaga, 29071 Málaga (Spain)

ABSTRACT

This paper presents a desktop application tool for visualizing simulations of Mobile Ad Hoc Networks with caching capabilities. This tool uses the log files generated by any network simulator that implements the required log format and the mobility file with the NS2 format in order to represent both, the network traffic and the mobility respectively. This tool works as an interface to represent the network activity and the node mobility by means of animations. On the other hand, this tool also calculates and shows, on real time and at any simulation time, the statistics needed to evaluate the MANET performance as well as the cache states. Finally, reports in PDF format including the statistics of each node and the global statistics can be generated. The application is implemented using the Java language and hence it is multiplatform.

KEYWORDS

MANET, caching, simulations, visualization tool.

1. INTRODUCTION

A Mobile Ad Hoc Network (MANET) is a set of autonomous mobile terminals (MT) that can communicate among them using wireless links. As the MTs are forwarding packets to the MT in their coverage area, some kind of routing protocol is needed to make the routing decisions. The management of this kind of networks is performed in a decentralized way and all the nodes in the network have to cooperate forwarding and routing the packets to the other MTs in the network. The main characteristics of the MANETs are:

- Dynamic topology - Due to the mobility pattern, the MTs can enter or leave the coverage area of other MTs. This situation forces to recalculate the routes to other MTs in order to enable packet forwarding.
- Limited bandwidth and variable capabilities – The wireless medium is characterized by a reduced bandwidth and a greater error probability than the wired medium, since the radio medium is always shared and prone to interferences and packet collisions.
- Limited batteries and processing capabilities – Due to the mobile nature of this kind of networks, the mobile devices have to be portable (wearable in some cases) and hence the processing and battery capabilities are also restricted.

Due to the kind of mobile devices available in the MANET (such as laptops and smart cellular devices) more and more users demand the access to external networks, data servers or the Internet, thus these kind of networks have to be prepared to use these services. Figure 1 depicts an architecture where a MANET is connected to the Internet using two alternative Access Routers provided by two Access Networks.

Because of the nodes mobility the Access Routers can be out of the coverage area of any mobile node in the MANET. This causes the disconnection to the external networks. In order to avoid this situation some caching mechanisms have been proposed. According to these mechanisms nodes may store some of the data requested to the servers and cooperate to share information about where a valid copy of the data is stored in the MANET.

Developing, validating and evaluating cooperative caching architectures is hard work because of the distributed nature of the ad hoc networks. Moreover, the existing MANET simulators do not allow to visualise all the internal structures of the nodes or the network traffic as the simulation evolves. Due to this limitation we propose the implementation of a MANET visualization tool that allows to observe the nodes

movements as well as the information related to the caching schemes implemented. On the other hand the network visualization tool transforms the difficulty to interpret trace files through a friendlier interface.

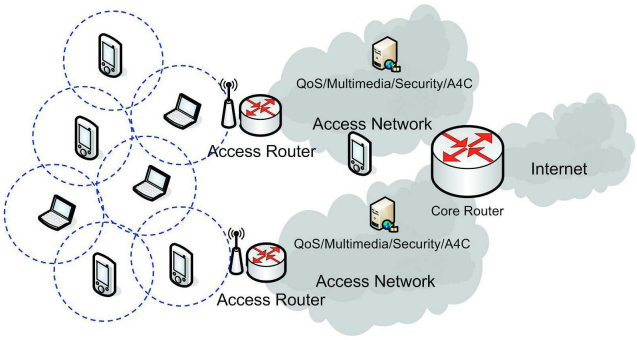


Figure 1. MANET with Internet connection

The rest of this paper is organized as follows. Section 2 describes some alternative visualization tools for MANETs pointing their advantages and flaws. Section 3 comments the caching architecture supported by the implemented application. Section 4 illustrates the application capabilities and specifications. Finally, Section 5 outlines the main conclusion of this work and proposes future works.

2. RELATED WORK

The NAM tool (Network AniMator) (NAM, 2010) is the default NS2 (Network Visualization) (NS2, 2010) visualization tool and it is able to process a great amount of data and to calculate some statistics about the simulations. NAM was designed to provide a graphical user interface in order to configure simulations of wired network topologies, allowing to show the links created and the packets flows. Unfortunately, it has not been adapted to the mobile nature of the MANETs. There was a project called ad-hockey (ad-hockey, 2010) created by the Monarch project (Monarch, 2010) in the Carnegie Mellon University started in the late 90s in order to add this capability, but it was discontinued.

iNSpect (interactive NS-2 protocol and environment confirmation tool) (Kurkowski, 2005) is a visualization tool developed using C++ that allows the analysis of wireless networks simulated by NS2. Its main characteristic is that it is able to manage different input log files, accepting the log and mobility files generated by NS2, but it also needs a special kind of trace file denoted by vizTrace. In addition, iNSpect can also validate the NS2 because it can test if NS2 correctly manages the mobility pattern. Finally, this tool also calculates statistics parameters about the simulated networks. However iNSpect does not offer information about the current simulation such as the number of processed packets, size, source and destination IP addresses of each hop and does not permit the examination of the internal structures of the mobile nodes.

Huginn (Scheuermann, 2005) is another visualization tool for wireless networks using NS2 implemented using C++. Its main capability is the usage of 3D graphics for representing the networks and a sophisticated schema to display the information. The user can specify a flow diagram that defines the data filtering. Thus, the user can specify the kind of data in which the application has to concentrate on. However Huginn has the same flaws as the previous tools, as it does not represent the internal structures implemented in the mobile nodes.

EXAMS (EXtensible Animator for Mobile Simulations) (Livathinos, 2009) is a full simulator written in Java that adds functionalities which are not available in the above mentioned tools, such as the possibility to represent the nodes internal state, to view the coverage area of the nodes and even calculate statistic data about the network performance. Unfortunately it does not implement any kind of caching mechanisms, which is the main goal of our studies. In addition EXAMS does not perform a node position interpolation so it does not perform a validation of the nodes movement.

Finally, MobiSim (Mousavi, 2007) is a free application developed with the aim to support the investigations in MANETs and the study of the mobility in these kind of networks. The user can create mobility scenarios and simulate them graphically. It supports a wide range of mobility patterns that can also

be merged. However MobiSim is only able to manage mobility models and not the traffic along the network. It is only a mobility pattern generator and a visualization tool for the movements.

Additionally, we must consider that all the above mentioned visualization tools, except EXAMS, are only Linux compatible and hence, they are not possible to be used in other operating systems such as Microsoft Windows or MacOS.

3. CACHING ARCHITECTURE

In this section we present the caching architecture for ad hoc networks supported by the visualization tool. In this scheme the wireless nodes in the network request documents that are located in data servers. The procedure works as follows: a node requests a document to a data server; the request is routed through the ad hoc network using the routing protocol defined for this network. When the data server receives this request it responds by sending the document to the requester node. This scheme is similar to other request-response algorithms proposed in the literature (Yin, 2006).

Each mobile node in the wireless network will implement a local cache where the requested documents will be stored when they are received. In that way if the mobile node needs the same document in the near future it will be served by the local cache instead of the remote server. This situation is called a local cache hit. There are some local cache parameters that have to be taken into consideration:

- The storage space reserved to store the documents will determine the amount of documents that will fit in the local cache. As the cache size increases more documents will be stored and the probability of a local cache hit will also increase.
- Each document is associated to a TTL (Time To Live). This parameter defines when the document will expire and hence it has to be deleted from the local cache because the information is obsolete.
- The replacement policy defines the algorithm that decides which documents stored in the local cache will be evicted in order to make room for the new ones. Many replacement policies have been proposed in the literature and they all try to select for eviction the documents with the lowest probability of being requested again in the near future.

In our caching architecture we proposed to use the LRU (Least Recently Used) replacement policy although the visualization tool supports any replacement policy.

On the other hand, the mobile nodes in the wireless network also work as a proxy for the other mobile nodes. Each node in the path of a request from the node requester to the data server will check if it has a valid copy of the requested document in its local cache. If so, the intermediate mobile node will directly reply to the requester node with the document. This situation is called an interception cache hit and it reduces the response time and the server load as the requests do not reach them.

Since each mobile node in the wireless network has to forward the requests and responses acting as a proxy, they can also store information about the position where documents are stored in the other mobile nodes. In this way, each time a node receives a request to forward it stores that the document requested may be stored in the local cache of the requester node and how far the node is (in number of hops). Moreover, when a response is forwarded each node will also annotate which node replied and the corresponding distance in hops. The TTL of the document is also taken into account in order to set a validation time for this information. Each time a node receives a request to forward it searches for a mobile node that has a copy of the requested document in its local cache and that it is closer to the origin node than the data server. If so, the request is redirected to the node that is supposed to have the document, which will reply to the requester node with the document. This situation is called a redirection cache hit.

Unfortunately, the redirection of requests has some drawbacks that have to be considered:

- The nodes mobility and disconnections could cause the information stored about the documents location in the mobile network and the distances among the nodes to be invalid.
- The replacement policies can also decide to evict certain documents of a local cache and hence, the information about these documents in this node will also be invalid.

The previous situations could cause that a node decides to redirect a request to another node which is not reachable and hence, the request will never reach its destination. This can be partially solved implementing a timeout in the soliciting node in order to request the document again if this timeout is reached. When a mobile node receives a redirected request and it does not have a valid copy of the document (due to the

replacement policy), the mobile node will send a redirection error message to the node that decided the redirection in order to update the information about the location of this document.

The preceding caching architecture was previously presented and evaluated by the authors in (González-Cañete et al, 2009a) and (González-Cañete et al, 2009b). This caching policy is shown to reduce the network traffic and server load. In addition, the energy consumption of each node is also decreased because the amount of messages to be forwarded is also diminished.

4. THE VISUALIZATION TOOL

The visualization tool proposed has been developed using the Java language and the Java3D API. As the Java language is multiplatform, the application can be executed in any platform that supports it. Nowadays these platforms are Microsoft Windows, Linux, Solaris and Mac OS X although the tool has only been tested in the Microsoft Windows and Linux platforms. The application main window can be observed in Figure 2.

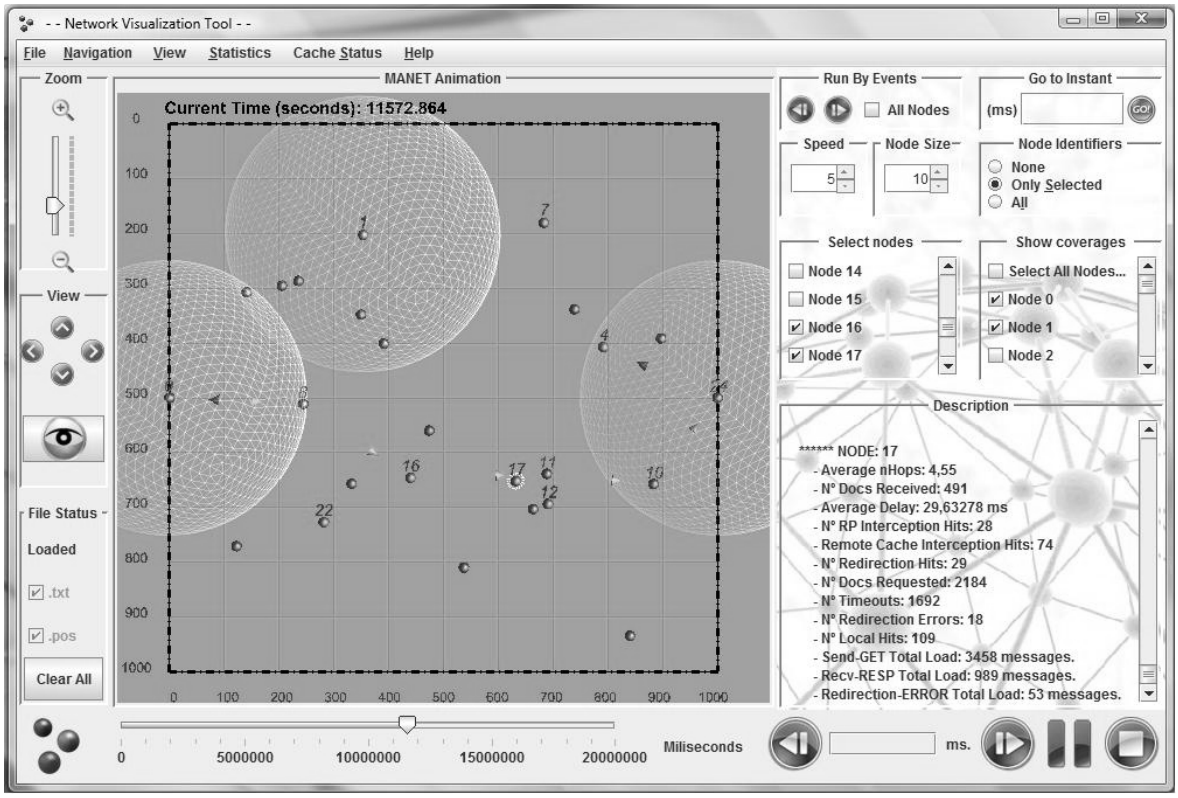


Figure 2. Visualization tool main window

The visualization tool accepts two input files:

- The mobility file – It includes information about the nodes positions, movements and speeds as well as the simulation area dimensions. The mobility file must follow the NS2 mobility file format and it can be created with any compatible generator such as the BonnMotion scenario generation tool (Aschenbruck, 2010)
- The traffic file – It specifies the information about the traffic along the network and the local and redirection caches states for each node. The traffic file format specifications are available at http://pc23te.dte.uma.es/FilesFormat.pdf.

When the mobility file is loaded the scenario is depicted in the MANET Animation panel of the application. The simulation area and its dimensions are shown using a grid of 100 meters. The mobile nodes are represented as spheres that can be selected in order to view their identification number and coverage area. The tool can be used to validate the generated scenarios because it is able to visualize and animate the nodes positions along the network according to the mobility file without specifying any traffic file. On the other

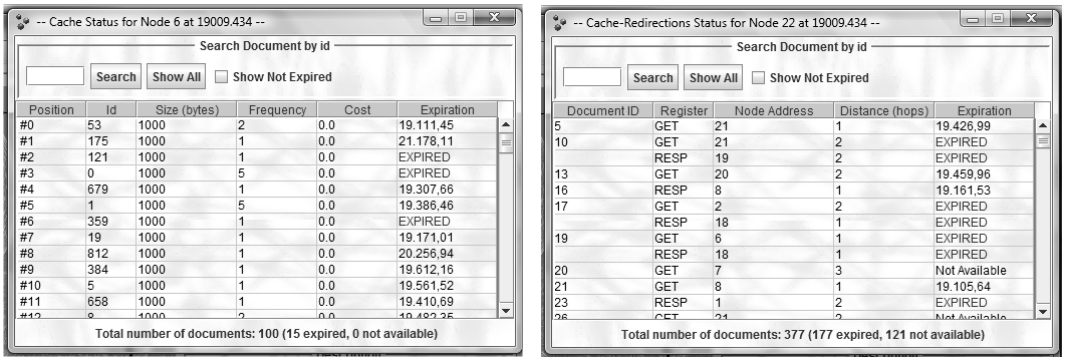
hand, when the traffic file is loaded in conjunction with the mobility file, the application is ready to visualize the traffic and node movement in the network.

The application capabilities will be enumerated as the panel's functionalities are commented:

- Zoom and View panels – These panels allow to change the point of view of the camera that represents the scenario. The camera can be moved in the four basic directions as well as to zoom in and out in the scenario.
- Playback panel – This panel contains the play back functionalities. With the slider control the simulation can be positioned instantly at any simulation time. On the other hand, the visualization can be played, paused and stopped. Finally the visualization can run at fixed and controlled time steps defined by the user.
- Run By Events panel – This panel permits to run the visualization and stop automatically once an event has occurred at any of the selected nodes. The considered events are: the node sends, forwards or receives a message, the node reaches its destination coordinates in the mobility model and/or the local or redirection caches are modified.
- Go To Instant panel – An absolute simulation time can be specified in order to set the visualization instant to this time.
- Speed panel – By default the simulation speed is at real time. This control allows to speed up or down the visualization.
- Node Size panel – The nodes sizes can be increased or decreased using this control.
- Node Identifiers panel – The node identification number can be viewed for all, none or only the selected nodes in the MANET.
- Select Nodes panel – The nodes can be selected or deselected checking the node identifiers shown in a list.
- Show Coverage panel – Similarly, the node coverage area visualization can be enabled or disabled selecting the nodes in the corresponding node list.
- Description panel – This panel lists and depicts the events and main statistics of the selected nodes.
- MANET Animation panel – This is the main panel in the application and represents the scenario to be simulated. It represents the mobile nodes and their movements along the simulation area. Moreover, the mobile nodes can be selected directly clicking over the sphere with the left mouse button. In addition, if the right mouse button if pressed over a node a menu will appear offering to show the statistics, coordinates, identifier, coverage area and the local and redirection cache of the node.

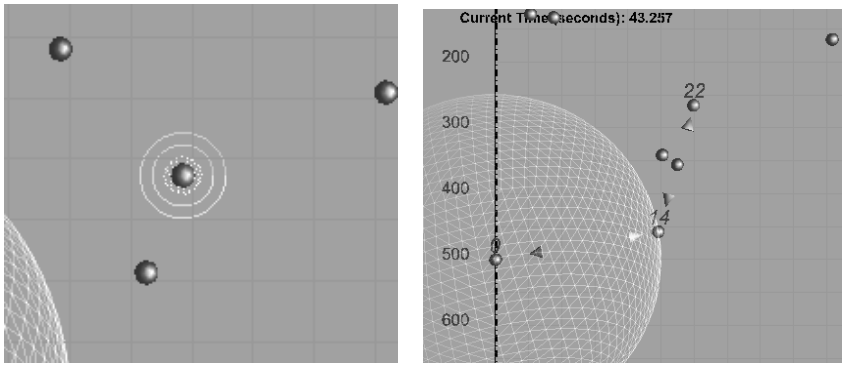
At any time in the simulation the local and redirection caches of the nodes can be shown. Figure 3.a illustrates the local cache content window. In this popup window the document identification, size, number of accesses, cost (for cost based replacement policies) and expiration time are listed ordered by the positions in the local cache. The document located in the highest position in the local caches will be the next document to be evicted. This local cache window also allows to filter the documents by the document identification number or showing only the documents which have not expired. Figure 3.b depicts the redirection cache window where the document identification number (*id*), the type of information register (could be GET or RESP depending on whether the information was obtained by a request or a response respectively), the node identification where the document is located, the distance in hops to the node that contains the document and the information expiration time are shown. Similarly to the local cache window the information listed in the redirection cache window can also be filtered showing only the information which is not obsolete as well as selecting a specific document.

Aiming at obtaining a better comprehension of the events that occur in the wireless network the visualization tool also includes some visual indicators that facilitate the discrimination of certain situations. In that way, when a mobile node is trying to send a request a halo is displayed informing about this fact (Figure 4.a). Similarly, the traffic between mobile nodes is represented using animated arrows indicating the traffic direction (Figure 4.b).



(a) (b)

Figure 3. Local cache content window (a) and remote cache window (b)



(a) (b)

Figure 4. Visual helpers. Halo informing about a request (a) and traffic (b)

Apart from the above mentioned visualization capabilities the application also calculates performance and statistics parameters about the simulation at any simulation time. These parameters could be used as an example to compare the performance among network configurations or cache sizes. The main statistics that the application calculates for each node are:

- Number of sent requests and received documents.
- Number of timeouts – When a mobile node requests a document it waits for a certain amount of time to receive the response. If the document does not reach the originator node during this time, the request is considered lost and the document is requested again.
- Average delay and hops – For each document received the time the request took to be served is measured as well as the number of network hops needed to transfer the requests and the reply.
- Number of cache hits and errors – The local cache hits, interception hits, redirection hits and redirection errors are stored.
- Traffic – The application calculates the amount of requests, responses and errors sent or forwarded by each node.

Additionally, the statistics can be exported to a PDF file where the previous values are saved for each node at the current simulation time. Histograms about each previous parameter and all nodes are also included in the report as well as the average values for the entire network.

5. PERFORMANCE EVALUATION

In order to evaluate the application limits, 27 mobile scenarios have been tested varying the number of mobile nodes (25, 50 and 100), the node speed (1, 3 and 5 m/s) and the mean time between requests (5, 25, and 50 seconds). The simulation time was set to 20000 seconds that is twice the greater simulation time found in the caching in ad hoc networks literature. These tests were performed using an Intel Core 2 Duo 2.0

GHz computer using the Ubuntu 9.10 (32-bits) operating system and the maximum memory allocation value allowed for the Java Virtual Machine (JVM) in the operating system selected (2.5 GB). When the application failed because of memory limitations the simulation time was reduced until the visualization was possible.

Table 1 summarizes the scenarios tested as well as the trace file size, the number of events and the memory consumed by the JVM.

Table 1. Application performance evaluation scenarios

Nodes	Speed (m/s)	Mean time between requests (s)	Trace file size (MB)	Events (x 10^3)	Simulation time (s)	Memory load (MB)
25	1	5	82	1421	20000	970
		25	36	629	20000	550
		50	24	424	20000	510
	3	5	92	1590	20000	960
		25	36	637	20000	730
		50	22	395	20000	510
	5	5	92	1590	20000	1060
		25	37	644	20000	650
		50	23	405	20000	510
50	1	5	231	3862	20000	1640
		25	60	1016	20000	700
		50	33	549	20000	550
	3	5	244	4120	20000	1750
		25	66	1115	20000	730
		50	35	592	20000	560
	5	5	244	4159	20000	1760
		25	69	1167	20000	660
		50	38	638	20000	620
100	1	5	322	5369	13500	2540
		25	125	2091	20000	1050
		50	66	1088	20000	820
	3	5	344	5791	13500	2560
		25	140	2360	20000	1140
		50	73	1225	20000	930
	5	5	353	5982	13500	2560
		25	153	2594	20000	1450
		50	76	1273	20000	840

As can be observed, as the number of nodes or the nodes' speed increases or the mean time between requests decreases, the number of events in the network and the trace file size are also increased. For all the scenarios except the ones with 100 nodes and 5 requests per second the visualization was performed without problems. For these very loaded scenarios the simulation time had to be reduced until 13500 seconds that is a reasonable and useful simulation time.

6. CONCLUSION

In this work a multiplatform visualization tool for mobile ad hoc networks with caching capabilities has been presented. The network behaviour accepted is a request/response protocol where the mobile nodes request documents to data servers that reply with the documents. Each mobile node in the network implements a local cache and can perform as a proxy for the other mobile nodes because each node is able to reply to the requests when it has a valid copy of the requested document in its local cache. In addition, the mobile nodes learn, using the traffic they forward, where and how far the documents are located in the network and hence they can decide to redirect a request to a nearer node than the original requested destination.

The implemented tool allows to visualize and animate the wireless nodes moving along the network. Moreover, if the traffic trace is defined the application also represents the requests and responses using animations. The local and redirection cache can be visualized for each node and even the information presented can be filtered in order to focus on the more relevant cache entries for the simulation.

The application offers several simulation playback controls allowing to view the simulation in real time, to jump to an absolute simulation time, to step back and forward a certain amount of time and to play event by event. The event by event playback functionality is very useful to monitor the node activity.

Additionally, the application calculates the most important statistical parameters that make it possible to study the performance of the wireless network and the caching mechanism. A detailed PDF report is also generated including all the statistics calculated for each node, the histograms for each statistical parameter as well as the global network statistics.

The performance of the application has been widely evaluated using several MANET scenarios and we concluded that the main limitation is the amount of memory allocated for the Java Virtual Machine.

Finally, we must emphasize that the main target of the presented tool is to study, develop and debug cooperative caching architectures in mobile ad hoc networks. This is the unique network visualization tool that allows having access to the internal caching structures of each node in the network apart from being a multiplatform application as it has been developed using the Java technologies.

As a future work we propose to enhance the functionalities offered by the application such as supporting broadcast messages that will enable to support broadcast caching architectures.

ACKNOWLEDGEMENT

We would like to thank Adela Isabel Fernández Anta for revising the syntax and grammar of this paper. This work was partially supported by the public project TEC2009-13763-C02-01.

REFERENCES

ad-hockey home page, 2010, http://www.monarch.cs.rice.edu/cmu-ns.html

Aschenbruck,N., et al., 2010. BonnMotion – A Mobility Scenario Generation and Analysis Tool. *Proceedings of the 3rd International Conference on Simulation Tools and Techniques (SIMUTOOLS 2010)*. Malaga, Spain.

González-Cañete, F.J., et al., 2009a. Proposal and Evaluation of an Application Level Caching Scheme for Ad Hoc Networks. *Proccedings of the 5th International Wireless Communications and Mobile Computing Conference (IWCMC 2009)*. Leipzig, Germany, pp. 952-957.

González-Cañete, F.J., et al., 2009b. Proposal and evaluation of a Caching Scheme for Ad Hoc Networks. *Proceedings of the 8th International Conference on Ad Hoc Networks and Wireless (Ad-Hoc Now 2009)*. Murcia, Spain, pp. 366-372

Kurkowski, S., et al.,2005. A Visualization and Animation Tool for NS-2 Wireless Simulations: iNSpect. *Proceedings of the 13º IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*. Atlanta, USA, pp. 503-506.

Monarch project home page, 2010, http://www.cs.cmu.edu/afs/cs.cmu.edu/Web/People/dbj/monarch.html

Mousavi, S.M., et al., 2007. MobiSim : A Framework for Simulation of Mobility Models in Mobile Ad-Hoc Networks. *Proceedings of the 3rd IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (IEEE WiMob 2007).* New York, USA,

NAM home page, 2010, http://www.isi.edu/nsnam/nam/

Livathinos, S.N., 2009. EXtensible animator for Mobile Simulations: EXAMNS, *Proceedings of the 17th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*. London, United Kingdom.

NS2 home page, 2010, http://www.isi.edu/nsnam/ns/

Scheuermann, B., et al., 2005. Huginn: A 3D Visualizer for Wireless ns2 Traces. *Proceedings of the 8º ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. Montreal, Canada, pp. 143-150.

Yin, L., and Cao, G., 2006. Supporting Cooperative Caching in Ad Hoc Networks. *In IEEE Transaction on Mobile Computing*, Vol. 5, No. 1, pp.77- 89.