View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by PORTO@iris (Publications Open Repository TOrino - Politecnico di Torino)


POLITECNICO DI TORINO
Repository ISTITUZIONALE

# Minteos mesh protocol and SystemC simulator

*Original*
Minteos mesh protocol and SystemC simulator / Man K.L.;Mercaldi M.;Lazarescu M.T.;Brini M.. - ELETTRONICO. - (2010), pp. 1-5. ((Intervento presentato al convegno Future Information Technology (FutureTech) tenutosi a Suzhou, China nel May 2010.

*Availability:*
This version is available at: 11583/2507485 since:

*Publisher:*
IEEE / Institute of Electrical and Electronics Engineers Incorporated:445 Hoes Lane:Piscataway, NJ 08854:

*Published*
DOI:10.1109/FUTURETECH.2010.5482669

*Terms of use:*
openAccess

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

*Publisher copyright*

(Article begins on next page)

04 August 2020

# Minteos Mesh Protocol and SystemC Simulator

K.L. Man
Department of Computer Science and Software Engineering
Xi'an Jiaotong-Liverpool University (XJTLU)
Suzhou, Jiangsu 215123, China
Email: ka.man@xjtlu.edu.cn

M. Mercaldi, M. T. Lazarescu and M. Brini
Minteos, Italy
Email: {mercaldi,lazarescu,brini}@minteos.com

***Abstract*—This paper presents our industrial experience on the implementation of Minteos Mesh Protocol which is a memory, power and delay efficient mesh protocol; and Minteos SystemC Simulator for mesh networks.**

**Experiments are carried out to validate the adequate use of Minteos Mesh Protocol. Also, simulation/test results are given to show the effectiveness and applicability of Minteos SystemC simulator for mesh networks.**

## I. Introduction

*Mesh networking* is a type of networking where each node in the network may act as an independent router, regardless of whether it is connected to another network or not. It allows for continuous connections and reconfiguration around broken or blocked paths by hopping from node to node until the destination is reached (taken from [1]).

Also, mesh networking provides communication between *GateWays* (GWs) nodes and repeaters. The protocol should be memory, power and delay efficient. In order to reach a good compromise between memory, power and delay, one should find a good trade-off between the type and size of the routing tables, the route discovery actions and the time required for propagating a message to destination. The goals of designing a fast and efficient mesh network are as follows:

1) efficient memory usage;
2) minimizing flooding messages for network discovery;
3) reducing the number of messages (GWs needed) to communicate with a known destination.

The above facts lead to the investigation on combining on-demand and proactive route discovery. Over the years, various efficient protocols have been developed: Distance-Vector Routing Protocol [2], Link-State Protocol Routing Protocol [3], Ad Hoc On-demand Distance Vector Routing [4], Optimized Link State Routing [5], Hybrid Wireless Mesh Protocol [6] and Better Approach To Mobile Adhoc Networking [7].

Although the above-mentioned protocols are well-suited for capturing mesh characteristics, non of them can be used for our peculiar needs because we aim to develop a mesh network in which:

- the mesh network protocol size is extremely small;
- thus it occupies very little memory and exhibits very low power consumption;
- for our needs such a mesh network protocol is intended to rule a relatively static network and hence our proposed solution is not comparable to a wide network where the mobility is a must (e.g.: mobile phone);
- within the proposed mesh network, (new) components, e.g. gateways, can be easily added or removed;
- components should be able to discover their neighbors and the rest of the network topology;
- however all these modifications in network topology should not be so frequent.

In addition to the good characteristics and features of existing mesh protocols, for our mesh protocol, it is desirable to have the following characteristics:

- there are few frequent destinations (e.g. the GWs with GPRS and extended storage capabilities are about $3-5$);
- nodes should be able to automatically discover special destinations and the best routes to them;
- there are hundreds of common nodes that need to reach the GPRS-enabled nodes quite frequently;
- seldom topology can be changed (due to failure, replacement, visibility change, etc);
- mesh protocol has to be efficient in terms of delay, power consumption, RF usage, etc;
- the number of hops and TTL should be limited (i.e., any destination should be at most, let us say, 10 hops or less away from its originator);
- an ACK mechanism is used to trace the proper reception of the data by the destination using the same route.

This paper presents our industrial experience on the implementation of Minteos Mesh Protocol; and Minteos SystemC Simulator for mesh networks.

The reminder of the paper is organized as follows. Section II shows the specification of a mesh protocol that we proposed. A detailed description of Minteos Mesh Protocol is given in Section III. Section IV presents Minteos SystemC Simulator for mesh networks.

In Section V, experiments are carried out to validate the adequate use of the Minteos Mesh Protocol. Also, simulation/test results are given in the same Section to show the effectiveness and applicability of Minteos SystemC Simulator for mesh networks. Finally, concluding remarks are made in Section VI.

## II. Customize Specification of a Mesh Protocol

We have developed and proposed a memory, power and delay efficient mesh protocol, also upon boot up, nodes must

have either:

- generally speaking, every gateway introduced in a mesh network (switched on) starts a network discovery that fills routing and service table. To achieve our targets (i.e. to implement a memory, power and delay efficient mesh protocol), two routing options have been analyzed carefully. The two routing options are:
  1) the packets should be fully routed at the source (i.e., carrying the full route in the header when created)
  2) the packets which start only with the final destination set and they are routed to the first hop and then each intermediate hop redirects packets to the best next hop for such a final destination. This option has been chosen for our implementation because such an option allows us to store the best hop towards that destination and not the full route, this complies with low memory consumption specification, and also is a good choice for the route selection in a relatively sable network.
- no hard-coded routing table, every gateway starts a discovery process of GPRS-enabled nodes and the best routes; and uses such routes either always the best or in a *Round Robin Scheduling* [8] to average the power consumption across the mesh network.
- the recipient ACK which should have been sent back via the reverse route of the received packet.

## III. Minteos Mesh Protocol

The specification of Minteos Mesh Protocol consists of:

- elements:
  - routing table
    - when a gateway needs to communicate with another gateway in order to obtain some service, a packet is sent according to the routes described in this table.
  - service table
    - when a gateway requires a service (e.g. GPRS communication), it checks in the route table if a known gateway providing such a service already.
    - if there is no known gateway offering the requested service, then a service discovery packet is sent.
  - packet exchanged.
- actions:
  - re-broadcasting rules for packet sent during the network discovery phase;
  - update rules for the routing table when receiving special packets;
  - delivery rules for the packets directed to service-enabled nodes.

### A. Elements

For the routing table, we selected the format that only the information of the *best next hop* traverses to known service-enabled nodes. We made this choice because of the followings (see also the routing options in Section II for details):

- this format permits each node to maintain just the next hop towards all known destinations;
- it provides a fall-back mechanism in case of communication trouble and uses a Round Robin Scheduler to adjust the power consumption;
- such a format appears to be much more resilient to network failures when requiring an ACK for each message forwarded and for each segment.

In addition, each element in the routing table contains the destination identity, the neighbor identity and the route rank. The heart beats of the neighbors are monitored in such a way that any heart beat failure is noticed preventing faulty information forwarding amongst nodes.

### B. Actions

These rules are applied to the packets sent for the network discovery such that each node transmits broadcast messages to inform the neighboring nodes about its existence. Also, the neighbors transmit messages (by means of re-broadcasting) to inform their neighbors as well about the existence of the original initiator of the message being transmitted and so on and so forth.

Therefore, the network is flooded with the originator messages. Due to these, the size of the packet is minimized as much as possible because the transmitting messages contain only the address of the originator, the address of the node transmitting the packet, the full list of the nodes traversed so far and a TTL.

A packet is not re-broadcasted if a packet timeout occurs and/or the packet has been traversed already (reported by full list of the nodes traversed).

*1) Updating rules for the routing table:* Elements in the routing table should be added and updated as described previously. When receiving a packet for network discovery a new element in the routing table is added if route towards a service-enabled device passing from the transmitter neighbor still does not exist yet.

A route can be modified changing its rank value and/or the time of last ping sent according to type of packet received. The deletion of route is performed when the table is full and a new element has to be added. Not only discovery network packet but also packets sent for message delivery purpose can update the routing table.

*2) Delivery rules for packets containing messages:* When a node sends a message, delivery rules are applied for packets containing messages. Note that the packets are not always re-broadcasted. We should obtain from the routing table the best next hop toward the final destination.

Every intermediate node must keep the packets and wait the ACK from the neighbor the packet was sent. Also every node receiving this packet must send an ACK to the transmitter (which is always a neighbor).

## IV. Minteos SystemC Simulator for Mesh Networks

After having investigated the applicability of several existing simulators for Adhoc networks (e.g. Omnet++ [9], EYES

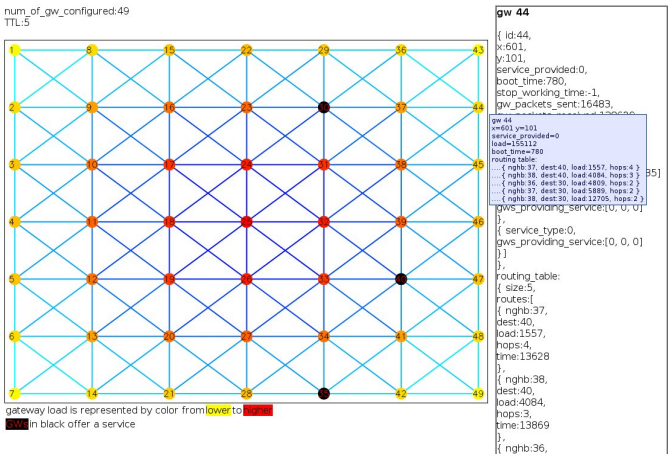


Fig. 1. Initial Network Discovery

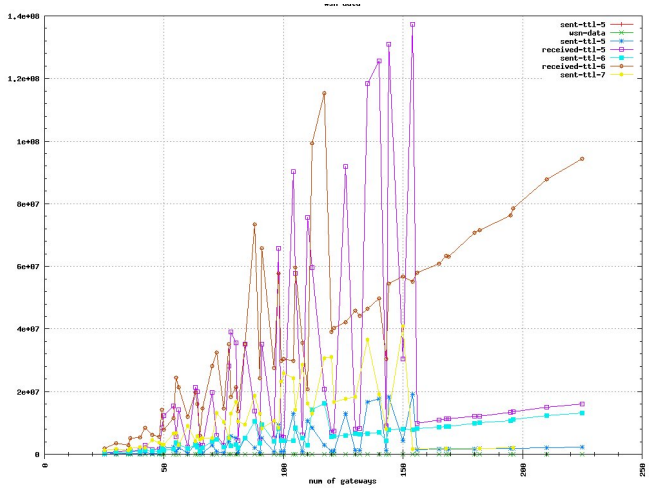


Fig. 2. Sample of Test Benchmark

WSN Simulation Framework [10] and NS-2 [11]), in our opinion, none of them would be suitable for the simulation of mesh networks. Thus, we decided to develop our own simulator much more oriented to our mesh network. Next section will also present an example which motivates the development of a new simulator (i.e. our simulator) to complement the above-mentioned simulators for mesh networks.

### A. Motivational Example

For a mesh network simulation, there is a field which consists of several hundreds of gateways that are placed in a square grid as shown in Figure 1. Some gateways have a special service like GPRS service so that it can be used to communicate with a remote central operating unit.

Each gateway collects environmental information like temperature, humility, pressure, etc, via many sensors surrounding it. In order to send data to the remote central operating unit, that normally is a data center far away from the field, every gateway has to look for a GPRS enabled gateway that serves as a proxy.

As described previously, our Minteos Mesh protocol offers the possibility to the gateway to learn and keep the knowledge of the network topology.

Figure 1 shows an initial network discovery. The color of the nodes indicates the load of the gateways (yellow is the lowest and red is the highest); and the black ones are GPRS enabled nodes. Moreover, Figure 2 presents a sample of a test benchmark.

### B. Key Elements for Minteos SystemC Simulator

The key elements for Minteos SystemC Simulator (implemented with sc_module) are as follows:

*1) gateway:*

- this is a model of gateway which collects data from the neighborhood sensors and sends notification to the remote central operating unit.
- at boot time, a discovery process of network packet is launched to first notify the presence of the gateway and then explore neighbors and the rest of the network.
- every gateway can transmit 2 frequencies:
  - the first one is *rf1* which is dedicated for *sensor-to-gateway communication*;
  - the second one is *rf2* which is dedicated for *gateway-to-gateway* communication and data and network discovery.

*2) rf_medium:*

- this medium imitates the communication behavior between gateways towards mediums, geographical coordinates and environmental interferences which are also taken into account for the simulation.

### C. Implementation of Minteos SystemC Simulator

Architecture, key features and design decision of the Minteos SystemC Simulator are summarized as follows:

- the structures and the functions implementing the behavior of the Minteos Mesh Protocol are developed in the source of the firmware for the gateway, this is then compiled as a shared library and linked to the Minteos SystemC Simulator which is written/implemented in SystemC [12].
- all these allow us to distinguish between the part of the software for carrying out the Minteos Mesh Protocol and the part implementing the simulator (which entails representing certain key characteristics or behaviors of the geographical distribution of the gateway on the field, quality of communications, medium, etc).
- geographical coordinates of the gateways are assigned at the run-time from a configuration file provided by the user. This also means that a set of gateway can be pre-allocated.
- actually, rf_medium introduces an attenuation of signals for simulation. For the implementation of Minteos SystemC Simulator, an isotropic medium is used and its attenuation is only based on the distance between gateways.

## V. Simulation/Test and Validation of Minteos SystemC Simulator

This section presents several simulation results which validate the operation of the Minteos Mesh Protocol as well as Minteos SystemC Simulator; and allow us especially to deduce their performance.

Let us consider a configuration file which consists of coordinates (represented by common coordinate pair `(x,y)`) of each gateway along with the type of service provided. We aim to test the following network topology in which each point of the network is described as a 3-tuple as follows: `gw_id(x,y,service if provided)`.

The test should cover the discovery network functions. This means that at boot time every gateway launches a discovery network packet. Then, the gateway waits for ACKs, and updates its service table as well as routing table.

In simple words, tables are only updated when a discovery network packet hits a gateway and is acknowledged eventually or the gateway is not the final destination of the packet.

With a `TTL=3` and no rank calculation, below are the test/simulation results:

```
---------------------------------
gw1 -  x:1 y:1   service_provided:0
 service table:  (size=1)
    - service_type:1 -  gws_providing_service:  5 10 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -

 routing table:  (size=5)
    - nghb:2 - dest:5 - rank:0 - time:0 -
    - nghb:2 - dest:10 - rank:0 - time:0 -
    - nghb:3 - dest:5 - rank:0 - time:0 -
    - nghb:6 - dest:10 - rank:0 - time:0 -
    - nghb:6 - dest:5 - rank:0 - time:0 -
---------------------------------
gw2 -  x:4 y:6   service_provided:0
 service table:  (size=1)
    - service_type:1 -  gws_providing_service:  5 10 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -

 routing table:  (size=5)
    - nghb:5 - dest:5 - rank:0 - time:0 -
    - nghb:10 - dest:10 - rank:0 - time:0 -
    - nghb:10 - dest:5 - rank:0 - time:0 -
    - nghb:1 - dest:5 - rank:0 - time:0 -
    - nghb:1 - dest:10 - rank:0 - time:0 -
---------------------------------
gw3 -  x:7 y:1   service_provided:0
 service table:  (size=1)
    - service_type:1 -  gws_providing_service:  5 10 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -

 routing table:  (size=5)
    - nghb:5 - dest:5 - rank:0 - time:0 -
    - nghb:1 - dest:5 - rank:0 - time:0 -
    - nghb:1 - dest:10 - rank:0 - time:0 -
    - nghb:2 - dest:10 - rank:0 - time:0 -
    - nghb:2 - dest:5 - rank:0 - time:0 -
---------------------------------
gw4 -  x:9 y:11   service_provided:0
 service table:  (size=1)
    - service_type:1 -  gws_providing_service:  5 10 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -

 routing table:  (size=4)
    - nghb:5 - dest:5 - rank:0 - time:0 -
    - nghb:8 - dest:5 - rank:0 - time:0 -
    - nghb:7 - dest:5 - rank:0 - time:0 -
    - nghb:5 - dest:10 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -

---------------------------------
gw5 -  x:9 y:6   service_provided:1
 service table:  (size=1)
    - service_type:1 -  gws_providing_service:  10 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -

 routing table:  (size=2)
    - nghb:3 - dest:10 - rank:0 - time:0 -
    - nghb:2 - dest:10 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
---------------------------------
gw6 -  x:1 y:6   service_provided:0
 service table:  (size=1)
    - service_type:1 -  gws_providing_service:  5 10 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -

 routing table:  (size=5)
    - nghb:10 - dest:10 - rank:0 - time:0 -
    - nghb:10 - dest:5 - rank:0 - time:0 -
    - nghb:1 - dest:5 - rank:0 - time:0 -
    - nghb:1 - dest:10 - rank:0 - time:0 -
    - nghb:2 - dest:5 - rank:0 - time:0 -
---------------------------------
gw7 -  x:15 y:11   service_provided:0
 service table:  (size=1)
    - service_type:1 -  gws_providing_service:  5 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -

 routing table:  (size=1)
    - nghb:4 - dest:5 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
---------------------------------
gw8 -  x:9 y:15   service_provided:0
 service table:  (size=1)
    - service_type:1 -  gws_providing_service:  5 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -

 routing table:  (size=1)
    - nghb:4 - dest:5 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
---------------------------------
gw9 -  x:13 y:16   service_provided:0
 service table:  (size=1)
    - service_type:1 -  gws_providing_service:  5 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -

 routing table:  (size=2)
    - nghb:8 - dest:5 - rank:0 - time:0 -
    - nghb:7 - dest:5 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
---------------------------------
gw10 -  x:1 y:11   service_provided:1
 service table:  (size=1)
    - service_type:1 -  gws_providing_service:  5 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -
    - service_type:0 -  gws_providing_service:  0 0 0 -

 routing table:  (size=2)
    - nghb:2 - dest:5 - rank:0 - time:0 -
    - nghb:6 - dest:5 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
    - nghb:0 - dest:0 - rank:0 - time:0 -
---------------------------------
----- packet counters:
---------------------------------
```

```
rf_medium:  num_events_received_on_rf2: 70
rf_medium:  num_packets_notified_to_gw(1): 212
rf_medium:  num_packets_notified_to_gw(2): 294
rf_medium:  num_packets_notified_to_gw(3): 219
rf_medium:  num_packets_notified_to_gw(4): 106
rf_medium:  num_packets_notified_to_gw(5): 195
rf_medium:  num_packets_notified_to_gw(6): 194
rf_medium:  num_packets_notified_to_gw(7): 57
rf_medium:  num_packets_notified_to_gw(8): 57
rf_medium:  num_packets_notified_to_gw(9): 40
rf_medium:  num_packets_notified_to_gw(10): 146
rf_medium:                              tot: 1520

gw(1) num_packets_from_medium_to_gw: 212
rebroadcasted: 63
gw(2) num_packets_from_medium_to_gw: 294
rebroadcasted: 88
gw(3) num_packets_from_medium_to_gw: 219
rebroadcasted: 65
gw(4) num_packets_from_medium_to_gw: 106
rebroadcasted: 39
gw(5) num_packets_from_medium_to_gw: 195
rebroadcasted: 65
gw(6) num_packets_from_medium_to_gw: 194
rebroadcasted: 56
gw(7) num_packets_from_medium_to_gw: 57
rebroadcasted: 19
gw(8) num_packets_from_medium_to_gw: 57
rebroadcasted: 19
gw(9) num_packets_from_medium_to_gw: 40
rebroadcasted: 16
gw(10) num_packets_from_medium_to_gw: 146
rebroadcasted: 40
gateways:  tot num_packets_from_medium_to_gateways: 1520
rebroadcasted: 470
```

According to the results illustrated in this section, we can confirm that the adequate use of our Minteos Mesh Protocol and Minteos SystemC Simulator is validated. Indeed, the discovery network functions are covered.

## VI. Conclusions and Future Work

In this paper, we have first presented the main aspects of Minteos Mesh Protocol and Minteos SystemC Simulator. Then, we have illustrated the use of the Minteos SystemC Simulator through experiments.

Also the key features and development of the Minteos Mesh Protocol and Minteos SystemC Simulator have been highlighted along with promising simulation/test results. Our future work will mainly focus on:

- adding function to retrieve a quality index (about the quality of service) for each gateway;
- adding/updating routing table rules based on route rank;
- formalizing a series of tests to evaluate the quality of the implementation of our Minteos mesh protocol.

## Acknowledgment

Many thanks go to the industrial partner Minteos - Italy (http://www.minteos.com) for the research work presented in this paper. Also, the authors wish to thank the engineers from Minteos for many stimulating and helpful discussions.

K.L. Man wishes to acknowledge with thanks the encouragement and support given by Professor Jeremy Smith, Professor Steven Guan and Xi'an Jiaotong-Liverpool University (XJTLU), Suzhou, Jiangsu, China.

## References

[1] Wikipedia, "Mesh_network," 2009, http://en.wikipedia.org/wiki/Wireless_mesh_network.

[2] J. Garcia-Luna-Aceves and S. Murthy, "A path-finding algorithm for loop-free routing," *IEEE/ACM Transactions on Networking*, vol. 5, no. 1, pp. 148–160, 1997.

[3] J. M. McQuillan, I. Richer, and E. C. Rosen, "The new routing algorithm for the arpanet," *IEEE Trans. on Comm.*, vol. 28, no. 5, pp. 711–719, 1980.

[4] C. Perkins, "Ad hoc on-demand distance vector AODV routing," Nokia Research Center, University of California, Santa Barbara, S. Das, University of Cincinnati, USA, Tech. Rep. RFC 3561, 2003.

[5] T. Clausen and P. Jacquet, "Optimized link state routing protocol OLSR," INRIA, France, Tech. Rep. RFC 3623, 2003.

[6] J. M. McQuillan, I. Richer, and E. C. Rosen, "The nominal capacity of wireless mesh networks," *IEEE Trans. on Wireless Communications*, vol. 10, no. 5, pp. 8–14, 2003.

[7] M. Abolhasan, B. Hagelstein, and J.-P. Wang, "Real-world performance of current proactive multi-hop mesh protocols," in *the 15th Asia-Pacific Conference on Communications APCC2009*, Shanghai, China, 2009.

[8] Wikipedia, "Round Robin Scheduler," 2009, http://en.wikipedia.org/wiki/Round-robin_scheduling.

[9] OMNet++, "An extensible, modular, component-based C++ simulation library and framework," 2009, http://www.omnetpp.org.

[10] S. Dulman, O. S. Kaya, and G. Koprinkov, "EYES WSN Simulation Framework," 2009, http://www.omnetpp.org.

[11] NS-2, "The Network Simulator - NS-2," http://www.isi.edu/nsnam/ns.

[12] SystemC, *SystemC Users Guide and SystemC Language Reference Manual (Version 2.2)*, http://www.systemc.org.