An architecture for the implementation of Mesh Networks in OMNeT++

A. Ariza-Quintana E. Casilari A. Triviño Cabrera
Dpto. Tecnología Electrónica, University of Málaga
Campus de Teatinos, 29071 Málaga (Spain),
Tfno.: 34-952132755; FAX 34-952131447
aarizaq@uma.es ecasilari@uma.es atc@uma.es

ABSTRACT
This paper describes the implementation in OMNeT++ of a versatile protocol architecture for the simulation of 802.11 Wireless Mesh Networks (WMNs). The developed modules enable the routing at the 802.11 MAC layer as well as a packet forwarding technique based on label paths. The performance of the new architecture is compared with that of a typical IP OLSR ad hoc network proving that link layer routing (IP) can be completely substituted by the developed modules.

Categories and Subject Descriptors
I.6.5 [**Simulation and Modeling**]: Model development

C.2.1 [**Computer Communication Networks**]: Network architecture and design

General Terms
Performance, Design

Keywords
OMNeT++, mesh networks, MAC, routing.

1. INTRODUCTION
During last years Wireless Mesh Networks (WMN) have become an appealing research topic in the field of networking. Industry has also paid attention to WMN and successful 'mesh companies' have appeared to offer different mesh networking products to customers.

As in the case of ad hoc networks, mesh architectures allow the association of peer wireless nodes to conform a network in an adaptive, infrastructureless and self-organizing way. For this purpose, all the nodes in the network must work cooperatively and perform routing functionalities (if necessary) to define multi-hop routes that permit the interconnection of terminals that cannot communicate directly. Moreover, through multi-hop connections, coverage area of the network can be expanded with much lower transmission power (and consequently with a lower power consumption in the mesh nodes).

Although there is not a clear border between the concepts of ad hoc and mesh networks, we can emphasize certain differences. In contrast with the typical (and 'academic') conception of Mobile Ad Hoc Networks (or MANETs), mesh networks are mainly intended for static radio nodes (normally Internet Access Points) with very reduced or no mobility. Similarly, the classical notion of an ad hoc network. assumes that routing is executed at network layer (in essence IP) while mesh networks mainly base routing on the information at MAC (link) layer [1]. This last feature reduces the economical and computational cost of mesh network deployment as it simplifies the hardware and software of the mesh routers (which are not obliged to implement IP layer).

IEEE 802.11s [2] draft has been proposed as an amendment to 802.11 standard to enable the formation of WMNs with 802.11 capable-nodes. In this sense, IEEE 802.11s extends 802.11 to support both broadcast/multicast and unicast communications through multi-hop self-configuring topologies. IEEE 802.15, IEEE 802.16 and IEEE 802.20 working groups have also made efforts to develop new protocols for WMNs.

This work describes and proposes a simple protocol architecture for MWNs. The architecture, which has implemented and simulated in OMNeT++ [3], permits to perform routing and label based packet forwarding just employing the 802.11 MAC layer. The implementation does not follow any specific standard or draft. Its main goal is to offer an open platform to emulate mesh networks and to evaluate the performance of future functionalities and proposals for this type of communication systems.

When compared with of 802.11s, our label-based switching architecture presents the following advantages:

-The label based switching process is simpler and faster than routing based on addresses (MAC or IP).

-Our implementation of label based switching enables source routing while 802.11s only permits to employ distributed (hop-by-hop) routing. Source routing eases the implementation of QoS policies.

-As our architecture defines a special header between layers 2 and 3, it is basically independent of 802.11 and can be easily utilised with other MAC layers (such as 802.15 or 802.16 layers). Label switching enables to route the same packet through interfaces with different link layers.

-The main inconvenience of the label bases switching is the need of creating the label paths. If the mobility of the terminals is

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
OMNeT++, 2009, Rome, Italy.
Copyright 2009 ICST, ISBN 978-963-9799-45-5.

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5549
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5549

reduced, this is not a problem as paths will be stable. Consequently, the use of label based routes will be more efficient than address based routing in static networks.

Paper is structured as follows. Section 2 comments the general structure of the architecture and details the functionalities implemented through different fields in a special packet header. Section 3 and 4 describe the techniques that have been implemented for the forwarding and routing of packets, respectively. Section 5 shows some simulation results proving the correctness of the implementation. Section 6 discusses the benefits of the proposal while section 7 summarizes the main conclusions.

2. ARCHITECTURE IMPLEMENTATION

2.1 Block diagram

The protocol architecture for the simulation of 802.11 mesh network essentially requires the definition of specific modules that implement the routing functionalities at the MAC layer. Additionally we also considered of interest to include in our implementation a specific packet forwarding mechanism. In this sense, a policy based on label paths such as MultiProtocol Label Switching (MPLS) [4] can offer a very flexible tool for traffic engineering. With a MPLS-like strategy, packet forwarding is not based on the destination address but on a pre-defined path. The path is identified in every packet by incorporating a special header with a simple label. This label (and not the MAC or IP addresses) are analysed and switched in the nodes along the route.

Figure 1 shows the diagram of the modules used for the implementation. The figure also represents the links with the rest of the components of the INET implementation of 802.11 standard.

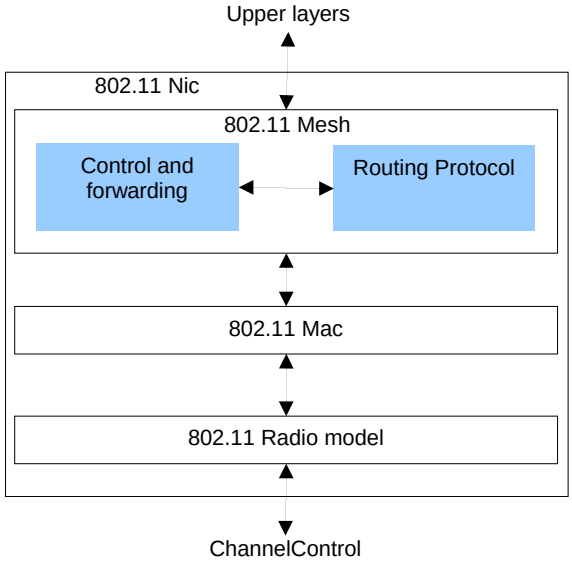

Figure 1. Block diagram of the proposed architecture

As it can be observed from the figure, the developed block (802.11 Mesh) consists of two intercommunicated sub-blocks: the module of Control and Forwarding is in charge of managing and creating the paths. This sub-block is also responsible for sending the packets from any node (to the following node in the route) in a transparent and seamless way to the upper layers. The goal of the second part of the architecture (the Routing Protocol) is to find a path to a final destination node when necessary. Next sections briefly summarise the structure and functionality of these two sub-blocks.

```
packet LWMPLSPacket
{
    fields:
       int label;
       int labelReturn;
       int type;
       bool nextHeader;
       unsigned int counter;
       int byteLength;
       MACAddress source;
       MACAddress dest;
       MACAddress vectorAddress[];
};
```

Figure 2. Message header

2.2 Packet header

802.11s draft proposes to perform routing decisions employing the information contained in the typical packet header of 802.11 protocol. However, to enable an alternative label based forwarding, a specific packet label header has been added in our architecture for WMNs. The format of this new header is presented in the Figure 1 according to the packet definition language of OMNeT++ simulator.

The header incorporates the following fields aimed at providing a MPLS-like packet forwarding:

-The `label` field includes the label that identifies the path that is being used by the message. The label for the return path is defined in the `labelReturn` Path.

-The `type` field defines the functionality of the message that is being transported. 11 different types of messages have been defined to support forwarding through label paths:

1. WMPLS_BEGIN. By this message, a node announces to the following hop in the path that a label based path is being created. The header includes the label that the next hop must employ to forward packets to the path source. When the WMPLS_BEGIN is received, the node responds with a WMPLS_ACK message informing about the label to employ in the contrary sense.
2. WMPLS_BEGIN_W_ROUTE. It is a special case of WMPLS_BEGIN as it also enables source routing (the origin node includes in the message the addresses of all

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5549
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5549

the nodes in the path).

3. WMPLS_NORMAL. This header indicates that the routing decision must be based on the transported label (a label based route must have been created previously).
4. WMPLS_REFRESH. This message is sent to keep active paths that are not in use currently. In normal conditions, a path that is not supporting traffic is considered to be obsolete after a certain time out.
5. WMPLS_END. This message permits to break a path explicitly (the path is also removed if it is not updated).
6. WMPLS_BREAK. This message informs that the connectivity in a hop is lost so that the corresponding path is not available
7. WMPLS_NOTFOUND. This message informs that a received label is unknown (the packet that contains it cannot be forwarded as no entry is found in the forwarding table for the label).
8. WMPLS_ACK. It is used during the path creation when a node informs to the previous node about the label to utilise when sending packets through the path. This message must be sent as an answer when a WMPLS_BEGIN message is received.
9. WMPLS_SEND. It is employed to send packets when MPLS-like paths are not created. As for 802.11s, every node in the route decides the following hop depending on the destination MAC address.
10. WMPLS_BROADCAST, aimed to broadcast packets.
11. WMPLS_ADDITIONAL (not employed) intended to extend the protocol and include new functionalities in a future.

-The Boolean `nextHeader` field is activated if more than one header exists in the message. As in the case of MPLS, our architecture allows to accumulate several headers in the same message.

-The `counter` field is utilised in the messages sent by broadcast to all the network nodes. The field (together with the origin address) permits to identify the message in order to guarantee that it is retransmitted by the nodes just the first time that the message is received (the packet is retransmitted only if the sequence number in the field is higher than the number of the last broadcast packet received from the same node). Every node has its own counter so this field is incremented in a node as soon as it originates a broadcast message. Similarly, all the nodes have to store in a table the sequence number of the last received broadcast packet received from each node.

-The `byteLength` field defines the total length (in bytes) of the header.

-The `vectorAddress` fields include the addresses of all the nodes in a route when a label path is being created with the mechanism of source routing. This field is analysed (or not) depending on the value of the `byteLength` field.

Finally the header also includes the MAC address of the source and destination nodes in the link (`MACAddress source` and `MACAddress dest` fields). Although these values are present in the 802.11 header of the packets, they have been included to simplify the processing of the C++ code.

3. CONTROL AND FORWARDING SYSTEM

The goal of this sub-block is to create and keep the virtual paths generated by labels. The module also manages the data forwarding and the communication with the higher layer and the 802.11 MAC layer.

3.1 Creation of label based paths

The actual MPLS protocol defines a specific additional mechanism [5][6] for the path creation, the resource reservation and the label assignation in the nodes along the path. Conversely, in our proposed architecture, the signalling information to create, maintain and destroy the label paths can be easily included in the message headers. Consequently, the path can be created as the first packet between the origin and destination nodes progresses through the network. For this purpose, this first packet must include the header WMPLS_BEGIN which indicates the next hope that a virtual label path is being defined. The message simultaneously defines the label to utilise for the return path by the receiving node (as an underlying 802.11 physical layer is assumed, links are considered to be bidirectional). Thus as the WMPLS_BEGIN evolves, the label path is configured in both senses without requiring to repeat the operation for the return path. This clearly minimises the time and the bandwidth demanded by the setup phase of the connection (which in most practical cases will require a bidirectional communication of packets). The node receiving the WMPLS_BEGIN message responds with an acknowledgment (WMPLS_ACK) assigning the label that the emitting node will have to use to send packets through the new path. While the acknowledgment is not received, the following packets will also be sent with a WMPLS_BEGIN header. Conversely after the reception of the acknowledgment subsequent outgoing packets will be transmitted with a MPLS_NORMAL header and the corresponding forwarding label. The implementation also contemplates the possibility that a node receives packets with a MPLS_NORMAL header while the path in the following hop is not still created. In that case, the retransmitted packet will change the MPLS_NORMAL header by a WMPLS_BEGIN header. The opposite operation is also possible.

The example of a path with 3 hops depicted in Figure 2 illustrates this exchange of messages during the label path setup.

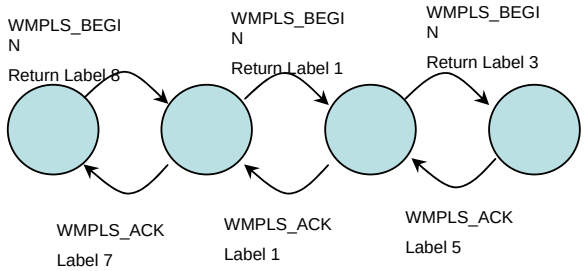

Figure 3. Information Exchange between nodes for the creation of a label path

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5549
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5549

Labels are selected from a pool of available labels. This selection is performed basing on the following criteria: the first found label that has not been in use during a certain interval is selected. If all the available labels were selected during that interval, the algorithm chooses the label that was least recently used. Making so, we minimise the probability of selecting a label that has not been released by the following hop (the node receiving the label) without exchanging any special release message. Here, we must remember that path labels are automatically released (and removed from the forwarding table) after a time-out without been utilised. Thus, if the age of a label (the time without being utilised) exceeds that time-out, we guarantee that it has been released by the corresponding node in the link.

The implemented architecture also allows source routing, that is to say, the definition of the whole label path from the source node. In this case the initial message creating the label path incorporates the addresses of all the nodes that compose the route. As when the route is set up hop by hop, once the path is configured, the message with data packets just include the corresponding label. Obviously source routing can be executed only if the routing protocol (OLSR in our case) permits the node to know the whole network topology so that a route to the destination can be directly defined from the source.

3.2 Operating modes

The architecture permits three operating modes for the exchange of information between nodes: label based forwarding mode, the hop-by-hop mode and the broadcast mode.

The label based (MPLS) forwarding mode creates label based paths between the source and destination nodes. Once the path has been created and the labels are assigned to the different links, routing decisions are purely based on these (active) labels. Thus, when a packet arrives to an intermediate node in a path, the control module searches in the forwarding table the label of this incoming packet. In the table this label is associated to three values: the return label, the MAC address of the following node in the route and the outgoing label (these two last values will be inserted in the packet to be retransmitted). Every active label has also a lifetime counter. If this counter exceeds a certain timeout the path is considered to be finished. In that case the label and its associated parameters are removed from the table. So, the label is considered to be available to create another path. On the other hand, whenever an active label is utilised, the label lifetime counter is set to zero. If an incoming packet transports an available label, an error message is transmitted to the emitting node (WMPLS_NOTFOUND). This message will erase the entry in the forwarding table corresponding to the missing label.

If the MAC layer detects that a packet cannot be delivered to the following node successfully, the link is considered to be broken. In this situation, the labels reserved for this link will be removed from the table and a special error message (WMPLS_BREAK) will be generated and transmitted to the rest of nodes in the contrary sense of the paths that use this link. The reception of this error message also implies the release of the labels reserved for the broken path. As part of a path can be active during a certain interval after a certain link falls, the labels that are removed from the routing table because of this reason are kept unavailable for a time interval. This prevents new paths to employ those labels that were assigned to paths that may be still in use with broken links.

In the 'hop by hop' operating mode, no label paths are created so the packet forwarding and routing is autonomously performed in every node along the path (in a similar way to typical IP or 802.11s routing). Under this mode, when a node receives a packet with a destination MAC address (set in the `MACAddress dest` fields) different from its own, it searches the corresponding entry in the routing tables (generated by the routing protocol) to determine the MAC address of the following hop (the node to which the incoming packet will be retransmitted). Under this operating mode, no label paths are created, so the routing/forwarding operation is similar to the procedure of the 802.11s draft.

The third operating mode is the packet broadcast (or packet flooding). This mode is conceived to send the packet to all the network nodes. To reach the whole network, all the nodes have to rebroadcast any incoming broadcast packet the first time that they receive it. To prevent the retransmission of duplicated packets, a sequence number is included in the `counter` field of the label headers of all the broadcast messages.

4. ROUTING PROTOCOLS

For the deployed architecture the routing functionality is executed at the link layer, as in the proposal of 802.11s standard. In particular, the routing protocol is a sub-process of the control system.

As routing algorithm, we decided to employ an existing mechanism for ad hoc networks. Several ad hoc routing protocols (including reactive strategies such as AODV, DYMO and proactive policies such as OLSR) have already been implemented for Inet in OMNeT++ [7]. These protocols were initially conceived to work with IP addresses. However, OLSR also enables to base the routing decision and the path search on the MAC addresses [8]. Consequently OLSR was chosen for our architecture of mesh network. In any case, other future or present routing strategies can be easily integrated in our implementation.

In order to habilitate OLSR to work at both layers (link and network) a special container class (`Uint128`) was specifically created. The class, aimed at managing the node addresses, can store 128-bit IPv6, 32-bit IPv4 and 40-bit MAC addresses. The class has overloaded operators that return the container classes of the corresponding INET address for the different types. Thus, the same code of OLSR (intended for routing at IP layer) can be utilised at the link layer.

Apart from the different addresses utilised for routing (MAC and IP addresses), the main divergence between the implementations of the routing procedures at link and network (IP) layer resides in the routing table. IP routing employs tables stored in the RoutingTable class. Conversely, our link layer implementation makes use of the internal tables of OLSR protocol. Thus, the control system directly accesses the content of these tables by executing the `getRouteMac` method which returns the complete route (a sequence of MAC addresses) to the destination node. The developed code is capable of discriminating if routing is performed at link or network layer. Consequently routing will be based on MAC or IP addresses.

5. SIMULATIONS AND RESULTS

In order to validate the implemented architecture a set of simulations were carried out. The main goal of the simulations

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5549
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5549

was to check the performance of the proposed layer-2 routing scheme when compared with the typical network layer (IP) routing of a MANET. Under both schemes OLSR was utilised to define the routes.

We have simulated networks with 45 and 50 nodes. The nodes were randomly distributed (according to an uniform random distribution) in a simulation area of 1500x1500 m. In the scenario of 45 nodes, 5 nodes implemented the complete protocol stack and assume the role of sources and/or destinations of the generated traffic. The other 40 nodes just implement up to the link (or network) layer and can only act as traffic routers. Source nodes are programmed to emit CBR (Constant Bit Rate) traffic at a rate of 5 packets/second through UDP connections. Packet size was set to 512 bytes. The duration of each simulation was 3000 s. Two cases for this scenario were considered: in the first one, the destination of all the flows is the same (predefined) node. In the second case, the destination for each packet is randomly chosen.

In the scenario of 50 nodes, 20 nodes implement the complete stack and may perform as sources and/or traffic destinations. In this scenario sources generate CBR traffic at a rate of 10 packets/second. The destination for each packet is randomly chosen. The rest of parameters are the same that the previous scenario. In all the experiments a packets is considered to be lost when it does not reach the destination node or when it arrives with a delay bigger than 1 second.

The simulations were repeated for the two compared policies (IP routing and layer-2 routing) using the same node distribution and the same traffic pattern (the origin and destination node of each packet was identical for both cases).

Tables 1, 2, 3 & 4 show that the proposed architecture with layer 2 routing achieves very similar results (in terms of packet delivery ratio and packet delay) to those obtained with classical IP routing.

Destination node:	Predefined	Randomly chosen
Layer 3 Routing	1	1
Layer 2 Routing	1	1

Table 1. Mean Packet delivery ratio. 5 sources

Destination node:	Predefined	Randomly chosen
Layer 3 Routing	0.49 ms	0.69 ms
Layer 2 Routing	0.52 ms	0.70 ms

Table 2. Mean Packet delay. 5 sources

The simulator does not model the processing delay at every layer. Consequently in an actual scenario (with real routers) the lookup process at the IP tables would introduce an additional component in the delay of IP routing, which is not reflected in the shown results. This reduction of the packet processing also impacts on the simulation time. Table 5 describes the mean number of simulated seconds per second for both policies. Table shows that layer 2 routing clearly reduces the duration of the simulation.

In any case, the comparison permits to assume that the developed modules have been implemented properly.

Destination node:	Randomly chosen
Layer 3 Routing	0.999
Layer 2 Routing	0.999

Table 3. Mean Packet delivery ratio. 20 sources

Destination node:	Predefined
Layer 3 Routing	1.35 ms
Layer 2 Routing	1.78 ms

Table 4. Mean Packet delay. 20 sources

Scenario	5 sources		20 sources
Destination node	Predefined	Randomly chosen	Predefined
Layer 3 Routing	0.83	0.87	0.26
Layer 2 Routing	1.48	1.53	0.30

Table 5. Simulation speed (simulated seconds per real simulation second)

6. DISCUSSION

When compared with typical IP routing, packet forwarding and routing at the link layer for ad hoc (and mesh) networks presents several advantages. Firstly, as it refers to the protocol stack, the network design is highly simplified: For the network layer all the internal mesh nodes are only one-hop away. Similarly, the node acting as the gateway to fixed Internet (if it exists) is not compelled to implement two different IP routing protocols (e.g.: OSPF [9] for communicating with any Internet node and a MANET protocol to interact with the other mesh nodes). Routing at MAC layers also enables an easier expansion of the mesh network. Thus, the network coverage area can be extended by deploying simple (and cheaper) routing nodes which are not obliged to implement the network layer (and consequently to have its own assigned IP address).

Figure 4 illustrates a test network with a mesh router. As it can be appreciated, the routing capacity, which is now located at the MAC layer, makes unnecessary the existence of upper layers in this node. The routing protocol is OLSR although other ad hoc routing policies could be easily incorporated if IP addressed can be substituted by MAC addresses.

Another interesting advantage of mesh networking is that routing at MAC layer can benefit (at least in a easier way than at the IP layer) from the link information that is available at link and physical layers (e.g.: received power, signal-to-noise ratio, channel occupation, etc). The utilisation of this information in the metrics employed to make the routing decision can clearly improve the implementation of QoS policies in wireless self-organising networks [10].

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5549
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5549

MAC layer routing also enables the possibility of having several MANET routing protocols working simultaneously. So, a proactive strategy (such as OLSR) could be employed to form the 'backbone' of fixed nodes of the mesh networks. In opposition, if fixed nodes implement more than one routing algorithm, a reactive protocol (such as DYMO), which is more appropriate for a dynamic network, could be employed in the mobile terminals.

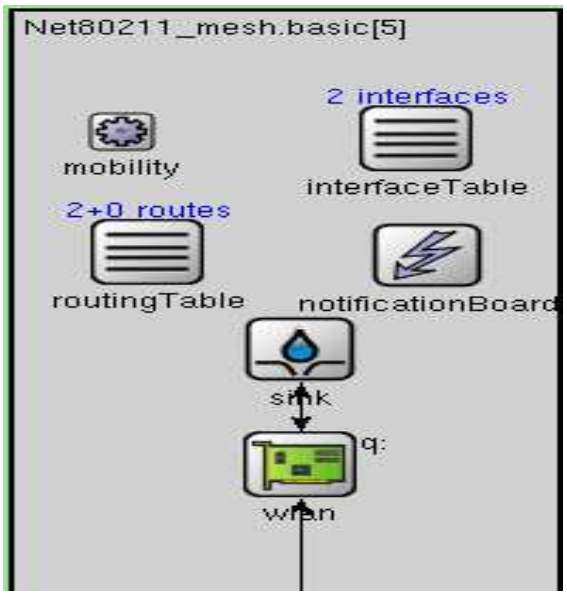

Figure 4. Test network with a simple layer-2 only router

7. CONCLUSIONS

This work has presented an open architecture to simulate self-configuring 802.11 mesh networks in OMNeT++. The architecture permits to simulate the routing at MAC layer proposed by 802.11s standard while it also incorporates a label based forwarding protocol that manages the creation, maintenance and removal of label paths. The flexibility of the implemented architecture eases its possible extension to emulate more complex cross-layered designs of new routing and/or forwarding protocols and it can be easily adapted to another wireless architecture like 802.15.4. . The developed code is publicly available at [11].

8. ACKNOWLEDGMENTS

This work was partially supported with public funds by the Spanish National Project No.TEC2006-12211-C02-01 (MCyT).

9. REFERENCES

[1] Akyildiz, F., Wang, X., Wang, W. 2005. Wireless mesh networks: a survey, Computer Networks 47, No. 4. (15 March 2005), pp. 445-487.

[2] IEEE P802.11s™/D0.01, Draft amendment to standard IEEE, 802.11™: ESS Mesh Networking. IEEE, March 2006, work in progress.

[3] OMNeT++, http://www.omnetpp.org

[4] Rosen, E., A. Viswanathan A., Callon R. L. 2001. Multiprotocol Label Switching Architecture, IETF RFC 3031, January 2001.

[5] Andersson, L., Minei, I. and Thomas, B. 2007. LDP Specification, IETF RFC 5036 (October 2007)

[6] Farrel, A., Papadimitriou, D., Vasseur, J.-P. and Ayyangar, A.. 2006. Encoding of Attributes for multiprotocol Label Switching (MPLS) Label Switched Path (LSP) Establishment Using Resource ReserVation Protocol-Traffic Engineering (RSVP-TE), IETF RFC 4420 (February 2006).

[7] Ariza, A., Casilari, E., and Triviño, A. 2008. Implementation of MANET routing protocols on OMNET++ , OMNeT++ Workshop, (March 2008).

[8] Clausen T., Jacquet P., 2003. Optimized Link State Routing Protocol (OLSR), IETF RFC 3626, (October 2003).

[9] Moy, J. 2008. OSPF Version 2. IETF RFC 2328, (April 1998).

[10] Campista, M.E.M., Esposito, P.M., Moraes, I.M., Costa, L.H.M., Duarte, O.C.M., Passos, D.G.; de Albuquerque, C.V.N., Saade, D.C.M., Rubinstein, M.G. 2008. Routing Metrics and Protocols for Wireless Mesh Networks, IEEE Network 22, 1 (Jan.-Feb. 2008), 6-12.

[11] Inet code with several Ah-hoc routing protocols, http://webpersonal.uma.es/~AARIZAQ/

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5549
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5549