Practical Methodology for Adding New MANET Routing Protocols to OPNET Modeler

Rani Al-Maharmah, Guido Bruck, and Peter Jung
Department of Communication Technologies
University of Duisburg-Essen
Duisburg, Germany
e-mail: info@kommunikationstechnik.org

***Abstract*—Optimized Network Engineering Tool (OPNET) Modeler is a comprehensive development environment that enables users to model communication networks and distributed systems by performing discrete event simulations. The OPNET Modeler accelerates the design and model of Mobile Ad Hoc Network (MANET) routing protocols by providing tools for all phases, including model design, simulation, data collection, and data analysis. Using the already defined MANET routing protocols is straightforward, while modifying or extending them is a tricky process that can be time and effort consuming. This paper provides an overview of OPNET Modeler architecture and describes a practical methodology to add new MANET routing protocols to OPNET Modeler by using the Multi-Aware Cluster Head Maintenance (MACHM) as an implementation example.**

Keywords-communication systems modeling; simulation-based approach; OPNET Modeler; MANET routing protocols; MACHM

I. INTRODUCTION

The performance of communication systems can be evaluated using different approaches and techniques. Those techniques can be classified into: formula-based calculations, waveform-level simulations, and hardware prototyping and measurements [1]. It is obvious that the performance evaluation based on measurements obtained from hardware prototypes of designs is accurate and useful, especially in later stages of production. In general, this approach is very costly and time-consuming. Besides, it is not very flexible, especially in the earlier stage of the design when the number of design alternatives may be large. Instead, powerful computer-aided analysis and design tools can be used for the modeling and simulation of complex communication systems. Those computer-aided analysis and design tools can be classified further into: formula-based and simulation-based approaches.

Using the formula-based techniques, which are based on simplified models, offers a considerable insight into the relationship between the different design parameters and the resulted system performance. It is useful to use such techniques in the early stages of the design, because it enlarges the design space. On the other hand, it is extremely difficult to just use such techniques to evaluate the performance of complex communication systems with a high degree of accuracy. Simulation-based approaches enable any level of detail to be modeled with the chance of wider design space than the one that is possible with formula-based approaches or hardware prototypes measurements. In simulation-based approaches, it is possible to combine both mathematical and empirical models easily. A simulation-based approach can be used to produce designs that are: timely, cost-effective, and error-free. On the other hand, the major disadvantage of using the simulation approach is the resulted computational burden.

OPNET Modeler is a flexible and powerful commercial tool [2], which is used to analyze and design communication networks, devices, protocols, and applications. OPNET Modeler incorporates a broad suite of protocols and technologies. OPNET Modeler includes a development environment that used to enable modeling of different network types. This includes any network with mobile devices such as cellular, mobile ad hoc, wireless LAN, personal area networks, and satellite. OPNET Modeler uses a combination of state transition machine diagrams and C (or C++) codes to implement the different technologies. Those codes interact with the different state transition machine diagrams, which are defined in the different process models. The codes themselves are scattered in different physical and logical places. In OPNET Modeler the code can be found in header files, external files, process models, header blocks, function blocks, diagnostic blocks, termination blocks, and so on. The challenging task is to understand the structure of the simulator and being able to track and modify or add the different required construction parts. In case of adding new MANET routing protocols, it becomes even more challenging and frustrating, because of the amount and type of modifications needed to enable the network nodes to use this new modified protocol. With lack of such knowledge, it will be hard to take advantage of the powerful tools provided by the OPNET Modeler, especially for the model design, simulation, data collection, and data analysis parts. A practical methodology showing the way and steps to do such modifications is needed. It will ease the process of merging the new MANET routing protocols into the OPNET Modeler and allow researchers to take a full advantage of the software capabilities.

This paper provides researchers with a systematic and easy to accomplish way for adding new models to OPNET Modeler, such as new MANET routing protocols. In this paper, a detailed description of the steps required for

 ISBN: 978-1-61208-308-7

modeling and merging a new MANET routing protocol named Multi-Aware Cluster Head Maintenance (MACHM) into OPNET network simulation software is presented. This work is intended to help and guide other OPNET Modeler researchers, who are interested in studying and investigating new MANET routing protocols.

The rest of this paper is organized as follows. Section II briefs the new MACHM, while Section III provides an overview of the OPNET Modeler architecture and the MACHM simulation project cycle. Section IV describes in detail the methodology of adding the MACHM to the OPNET Modeler. Simulation study of MACHM is presented in Section V, while Section VI concludes the paper.

II. MULTI-AWARE CLUSTER HEAD MAINTENANCE (MACHM)

MANETs are self-organizing mobile wireless networks with a decentralized control of operations, which does not rely on a preexisting infrastructure like access point to communicate [3]. Network nodes have the ability of free movement around. Normally, each node has to act as a router in order to keep the network operating. MANETs can be used in different areas and applications, examples include military scenarios, rescue operations, conferences, any application that needs mobility, and areas where it is hard to build a wired network [4]. Typically, node's resources are limited and valuable in MANETs. Most importantly, the battery power because it is limited due to the relatively small size of mobile nodes. Managing this limited resource is a key challenge in MANET's environments.

Routing protocols can be classified to proactive (table driven) and reactive (on-demand) routing protocols [6]. Table-driven routing protocols try to maintain consistent, up-to-date routing information from each node to every other node in the network. One obvious problem is the resulted network overhead. This occurs because of the amount of information collected, especially when the number of network nodes is large. Additional disadvantage is the wasted resources used to collect unnecessary routing information, which neither used nor useful later because of the MANET's dynamic nature. The advantage of using table-driven technique is that the routes are known immediately when they are needed. On the other hand, on-demand routing protocols create routes only when they are needed. This results in a reduced routing overhead cost but at the expense of a route establishment delay.

In order to achieve scalability while maintaining a good routing, hierarchical or hybrid solutions are adopted, like the cluster-based concept. The main idea is to group the nearby nodes into logical groups known as clusters, then assigning nodes different functions inside and outside the group (cluster) [5]. Each group contains a special node, which acts as a leader of the group based on some criteria. Different cluster-based techniques use different basis to decide this special node, such as highest ID, lowest ID, node's connectivity, node's power level, or simply a random node. The special chosen node is used to label the cluster and to communicate to other nodes on behalf of the cluster [7]. Researchers refer to this special node with different terms. Some terms used to express the leader are: cluster head (CH) [8], coordinator [9], core [10], member of dominating set [11], and backbone network [12].

Another advantage of using the cluster-based concept (beside the scalability) is the ability to mix the two different techniques of routing. Proactive routing technique can be used inside the clusters, where the number of network nodes is relatively small. While reactive routing technique can be used outside, between the created clusters using the cluster head nodes as access points.

Electing the cluster heads and maintaining them throughout the progress of the ad hoc networks are vital and critical. The importance comes from the role they are playing during the network lifetime. A multi-aware approach, namely Multi-Aware Cluster Head Maintenance (MACHM), have been designed for electing the cluster heads and maintaining them. MACHM aims to reduce the total amount of the power consumed by the nodes in the network. Especially, the cluster head nodes that are more sensitive to power drain, because of their extra roles in the network. MACHM implements a cluster-based approach to achieve scalability and to take advantage of the hybrid routing technique.

MACHM involves in cluster head election, clusters formation stage, and cluster head re-election procedures. It invokes the cluster head election and clusters formation at the time of system activation. In MACHM, not all the nodes are allowed to participate in the cluster head election. Instead, the decision of participating or not will depend on the initial node's battery power value. The reason for allowing smaller set of nodes to participate is to ensure that the candidate's battery power will be reasonable and will not reach the re-election threshold quickly.

To be more efficient, many factors will be considered in all stages. MACHM takes into account the ideal number of nodes that a cluster can handle (load balancing consideration), the distance between the node and its neighbors (geographical consideration), the speed of nodes (mobility consideration), and most importantly the node's battery power (energy consideration). The cluster head election is based on a weighted formula that includes the previous factors as states in Formula (1).

$$W_n = \Delta_n\, w1 + L_n\, w2 + M_n\, w3 + P_n\, w4 \qquad (1)$$

where *Wn* is the weight value for node *n*, Δ_n is the degree difference for node *n*, *Ln* is the summation of the distances for node n with all its neighbors, *Mn* is the speed average for node *n*, *Pn* is the current battery energy consumption value for node *n*, and *w1*, *w2*, *w3* and *w4* are constant values used to decide the relative importance of each factor, where $w1+w2+w3+w4=1$. This weighted formula enables the usage of some or all factors in the cluster head election calculation, based on the network scenario or setting. By assigning zero to any constant from *w1* or *w2* or *w3* or *w4* it will ignore the related factor in the election calculation. The previous calculated weight values will be exchanged between nodes to determine the cluster heads.

 ISBN: 978-1-61208-308-7

The same method will be used later in the cluster head re-election procedure. A mechanism has been proposed to maintain the previously elected cluster heads based on their energy levels. If the elected cluster head battery power level reaches the defined threshold, it will then initialize a new cluster head election. The scope for the replacement cluster head will be from the set of direct neighbors. This is to reduce the effect of changing the cluster head and to use the already gathered information. Figure 1 shows the flowchart used for the weight calculation.

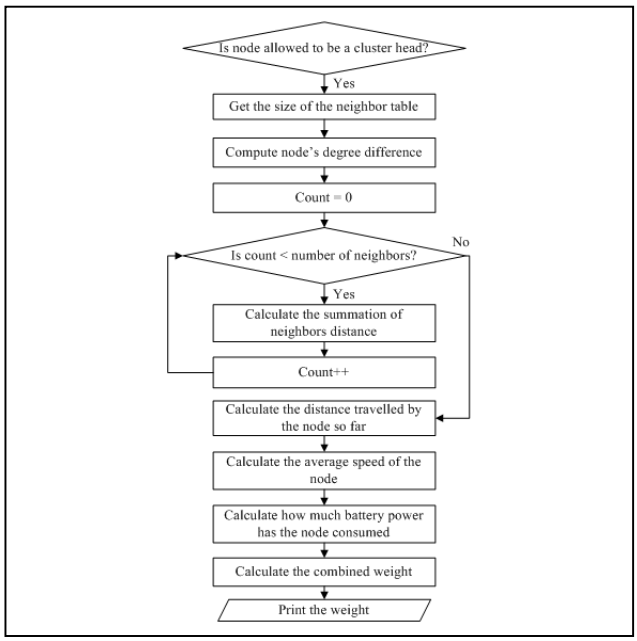

Figure 1. MACHM weight calculation flowchart

In Figure1, if the node's initial battery power level is over the defined threshold, then it is allowed to compete for the cluster head role. The node can immediately know the number of neighbor nodes by accessing the neighbor table and fetching its size. The node's degree difference is simply the abstract value of the number of neighbors subtracted from the *allowed_number_of_nodes* parameter, which is used for the load balance purpose. In MACHM, the nodes capsulate and send their coordinates in the *hello* messages. This enables the receiving nodes to calculate the distance and store this information along with the neighbor node. Each node then can calculate the distances summation with its neighbors, which gives an idea about the geographical position of each node.

In MACHM, each node tracks the distances it travels during the lifetime of the network. In a certain simulation time, it can calculate the speed it is travelling with by dividing the summation of distances by the current simulation time. This is the speed and mobility indicator used in MACHM. For the battery power consumed by each node, a battery energy consumption model is used. This model tracks and updates the battery power using the current draw values defined for the *SLEEP*, *IDLE*, *SEND*, and *RECEIVE* states. Finally, the node can calculate its weight value using Formula 1 and broadcast this value to its neighbors.

In order to implement the previously described method, network nodes need to gather different pieces of information relevant to their neighbors and keep them in different data structures. This collection of information can be done by exchanging control messages, which are broadcasted periodically or per event. Every control message intends to provide a certain piece of knowledge or invokes a certain action. MACHM uses route request, route reply, route reply acknowledgment, route error (link break detect, data packet no route, and route error received), hello, node weight, adjacent cluster head, and invoke request control messages. The complementary elements in MACHM are the timers and tables data structures. The timers used in MACHM are route entry invalid, route entry expired, route request expiry, connectivity loss, and cluster-head table timers. While the tables are route, IP common route, packet queue, route request, connectivity, adjacent cluster heads, and invoke request tables.

III. OPNET MODELER ARCHITECTURE

OPNET Modeler is a flexible and powerful tool, which provides a comprehensive development environment for the communication networks and distributed systems. It can be used to model and evaluate the performance of communication systems. OPNET Modeler contains a number of different construction tools. Each tool is concentrating on a specific phase or aspect of the modeling task. In OPNET Modeler, the MACHM simulation cycle is constructed from three major phases, namely: model specification, simulation and data collection, and analysis.

First step is to develop a representation of the intended system to be studied, this known as a model specification. OPNET Modeler environment provides different editors for the primitive building blocks. The available editors are project, node, process, external system, link model, packet format, ICI, and PDF editors. In OPNET Modeler, the model-specification editors are organized in a hierarchical fashion, which are network, node, process, and external system modeling environments.

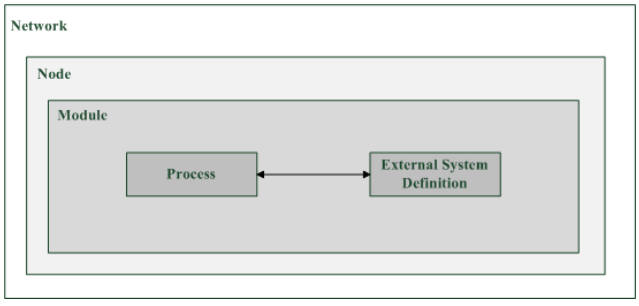

Figure 2. The relationship of hierarchical levels in models in OPNET

Figure 2 shows the relationship of the hierarchical levels in OPNET Modeler models. Network domain focuses on the network topology, which is described in terms of sub-networks, nodes, links, and geographical context. Node

domain focuses on the internal architecture of the node, which is described in terms of functional elements and data flow between them. Process domain focuses on the behavior of processes (protocols, algorithms, applications), which are represented in the form of finite state machines and extended high-level language (C or C++ code). External system domain focuses on the interfaces to models provided by other simulators running concurrently with a discrete event simulation.

Performance measurements of communication systems are the main goal of modeling them. This enables designers to study the behavior of such systems. Next step in MACHM simulation cycle is to run the simulations and collect the resulted data. OPNET Modeler allows a realistic estimation of performance and behavior for the executed simulations. It has several mechanisms to collect the desired data from one or more simulations of a system. OPNET Modeler uses discrete event simulations to produce different types of outputs (output vectors, output scalars, and animations). Users can define their own output file types as well, if this is desired. Normally, vast amount of output data will be generated after each simulation. Particular statistics or animations are explicitly activated in OPNET Modeler by recording them in the appropriate output files. This is done by specifying a list of probes when running a simulation. Each probe indicates that a particular statistic or form of animation should be collected in this run. Advance forms of probes can be defined by users in the probe editor.

The third and last phase of the MACHM simulation project cycle is analyzing and examining the collected results during the simulation. OPNET Modeler provides both a graphical and numerical processing environments, where user can investigate the generated results in depth. Additional data for plotting can be generated by a number of numerical processing operations in the analysis panels, including Probability Mass Function (PMF), Cumulative Distribution Function (CDF), histograms (occurrence and duration-based), confidence interval calculation, and mathematical filters defined in filter editor. In case of additional modifications for the communication system modeling, another round of the simulation cycle can be applied with the new specifications.

IV. IMPLEMENTING MACHM

In [13], a practical methodology for modeling wireless routing protocols using OPNET Modeler has been proposed. The authors implemented a modified wireless routing protocol named Geographical Ad-Hoc On-Demand Distance Vector (GeoAODV) as an implementation example. The methodology explains the way to modify an already existed routing protocol in OPNET Modeler. In case of adding a completely new routing protocol as MACHM, another practical methodology is needed, this is shown in this section.

In OPNET Modeler, the MANET is connected to the IP network through a MANET gateway that is running a MANET routing protocol and an IP routing protocol (or a static routing) on one of its interfaces. Figure 3 shows the node model of a MANET station in OPNET. MANET related files are gathered in the *manet* folder, except the header files, which are gathered in the *include* folder.

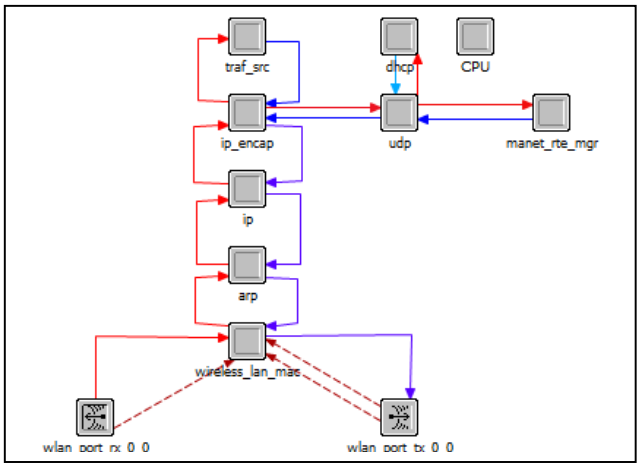

Figure 3. Node model of a MANET station in OPNET

The starting point to add a new MANET routing protocol to OPNET Modeler is the *manet_mgr.pr* process model. The MANET manager state machine functionality is to spawn the appropriate child routing process. This is based on the type of MANET routing protocol configured on the interfaces of the node. First, there is a need to register the newly MACHM routing protocol as a *manet_rte_protocol*. This is has to be accomplished in the header and function block of the *manet_mgr.pr* process model. Precisely, it is should be defined in the *manet_mgr_routing_protocol_determine* and *manet_mgr_routing_process_create* functions. A slight modification for *ip_higher_layer_proto_reg_sup.h* header file is needed as well to complete the registration process. A new child process named *machm_rte.pr* has been registered and added to the recognized list of the routing protocols in OPNET Modeler, as shown in Figure 4.

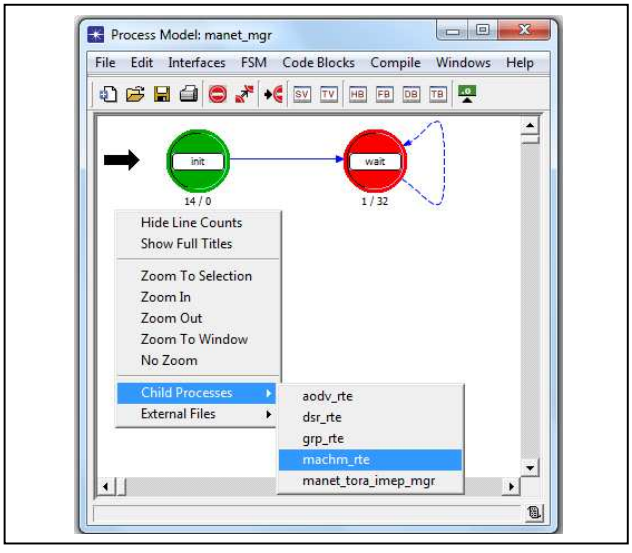

Figure 4. The modified *manet_mgr.pr* process model and its child processes

Next step is to define a new MACHM parameters group under the *AD-HOC Routing Parameters* group, which can be found in the *manet_mgr.pr* model attributes. The parameters are used to configure the behavior of the protocol. The parameters can be either a primitive or a compound type, it is also possible to control the range of parameter values and define a default ones. In order to make this new routing protocol available to the network devices, a change to the interface configuration of different types of nodes is needed. Like *manet_station_adv.nd*, *wlan_wkstn_adv.nd*, etc. nodes model.

The main work is done in the *machm.pr* process model, where the MACHM routing protocol functionality is defined by a finite state machine. The finite state machine initializes the required state variables and implements the different stages of the protocol. The implementation is achieved through the codes scattered in the entrance and exit parts, combined with the different blocks such as the header, function, termination, diagnostic blocks, and so on.

As stated before, the basic building blocks of MANET routing protocols are the control packets. MACHM packets format is defined using the packet format editor along with the external files. Normally, the definition contains functions to allocate and de-allocate the necessary memory needed in an explicit fashion. The memory allocation/de-allocation functions depend on the data types and storage sizes defined for the different control packets fields. The other important data structure in MANET routing protocols is tables. Tables are used to store useful information about the network, like the connectivity, data routes, data packets, and so on. The definition and handling of such data structures can be accomplished using the external files. A good practice is to define a single external file for each data structure (table) to ease the modification process in later stages.

For the communication systems performance evaluation purpose, different kinds of statistics are needed. OPNET Modeler offers two levels of statistics, local and global statistics. The local statistics can be collected on every node in the network, while the global statistics can be collected for the whole network. After defining the needed statistics in the *machm_rte.pr* process model, a registration of them is needed. The registration takes place in the function block using the *op_stat_reg* command, while the *op_stat_write* command is used to record the values during the simulation.

As mentioned before, in OPNET Modeler the structure of MANET routing protocol is scattered through different parts and files. First, MACHM defines the state and temporary variables, which are needed inside the routing protocol. They are used to hold some MACHM parameters or simply they provide a global scope for the different procedures and functions. The header and function blocks are responsible for the complete actions taking place in the MACHM routing protocol. Actions include: state variables initialization, battery energy level initialization, statistics registration, packets sending and arrival handling, updating the different tables, electing the cluster heads and so on. The defined finite state machine represents the connecting point for all those pieces. It makes the required transitions and calls of the different procedures and functions.

Instead of configuring the running routing protocol on node's interface one by one, another practical modification can be done to the *wireless_deploy_wiz_helper.xml* file. A couple of HTML code lines will include the newly defined MACHM routing protocol in the wireless deployment wizard routing protocols drop list, which then can be used directly in the project editor.

The practical methodology for adding a new MANET routing protocol to OPNET Modeler can be summarized as follows:

- Register the new MANET routing protocol through the *ip_higher_layer_proto_reg_sup.h* header file.
- Modify the *manet_mgr_routing_protocol_determine* and *manet_mgr_routing_process_create* functions in the *manet_mgr.pr* process model. This is to enable the network nodes to configure the new MANET routing protocol on their interfaces.
- Add the new MANET routing protocol as a child process to the *manet_mgr.pr* process model.
- Define the new MANET routing protocol parameters as a new group under the *AD-HOC Routing Parameters* group in the *manet_mgr.pr* model attributes. Additional levels of nesting for the parameters can be done too.
- Define the local and global statistics for the new MANET routing protocol through the header files.
- Register the newly defined statistics in the OPNET Modeler. The global statistics registration can be done in the header file, while the local ones in the function block of the process model.
- Configure the interface of the MANET and WLAN nodes by adding the new defined statistics.
- Define the finite state machine with the required transitions and codes.
- Define the state and temporary variables that will be used through the different blocks.
- Define the control packets and data structures needed for the new MANET routing protocol through the packet format editor and external files.
- Program the new MANET routing protocol through the header, function, diagnostic, and termination blocks.
- Modify the *wireless_deploy_wiz_helper.xml* file to enable the usage of the new MANET routing protocol by the wireless deployment wizard.

V. SIMULATION STUDY OF MACHM

After modeling MACHM in OPNET Modeler, a data collecting and analyzing is needed to complete the MACHM project cycle. In this section, a simulation and performance study of MACHM is presented. One important point is to decide which performance metrics better clarify the behavior of the new MANET routing protocol. In MACHM, the following performance metrics are considered: the number of weight and invoke request messages sent, the number of cluster heads, the average time a cluster head survives, and the energy consumption of the network nodes.

 ISBN: 978-1-61208-308-7

Network overhead can be measured in term of control messages sent in the network, such as the *MACHM_WEIGHT* and *MACHM_INVOKE_REQUEST* control messages. MACHM uses the weight messages to elect the cluster head, which is chosen from the set of the allowed nodes to compete. If the initial battery level is over the threshold, the node is allowed to participate in this competition. While the concept of invoke requesting is based on the battery power factor. An elected cluster head informs the nearby nodes that a re-election of the cluster head is needed, when its battery power level goes below the defined threshold. Another performance indicator is the number of cluster heads that exist in the network during its lifetime. It is important to have an idea about the average time an elected cluster head survives before it asks for a replacement. Saving the battery power of all nodes is the main concern of MACHM. Specially, the cluster heads that should not be loaded too much to ensure longer network life time. All nodes energy consumption metric is used to study this property.

Several combinations of OPNET Modeler and MACHM setups have been tested. Table I and Table II show the selected settings for this section. The changing parameter is the battery power level threshold, while the four factors have an equal importance of 25% in this scenario.

TABLE I. OPNET SIMULATIONS SETUP

Simulation Setup	Value
Number of Nodes	10
Area	100 X 100 m campus
Distribution of Nodes	Random
Mobility Model	Random Waypoint (1 m/s)
Operational Mode	802.11b
Data Rate	11 Mbps
Data Traffic	Complete VoIP mesh between nodes
Simulation Time	3600 seconds

TABLE II. MACHM PARAMETERS SETUP

MACHM Parameter	Value
Allowed Number of Nodes	3
Degree Difference Weight	0.25
Distance Summation Weight	0.25
Mobility Weight	0.25
Battery Consumption Weight	0.25
Battery Power Threshold	20%, 30%, 40%, 50%, 60% and 70%
Threshold Decrement	5%

As mentioned before, MACHM does not allow all network nodes to calculate and send their weight values. Only the nodes with initial battery power levels larger than the threshold are allowed to compete for the cluster head election. The tested threshold values are 20%, 30%, 40%, 50%, 60%, and 70% of the node's battery power level. Changing the threshold value will affect the number of weight messages send by the candidate nodes, as shown in Figure 5. In the simulations, the starting battery power level for each node is chosen randomly between 1% and 100% for more realistic scenarios. Increasing the threshold value will decrease the chance of having nodes with a starting battery power level over the threshold, and hence the weight messages send. When the threshold value is somehow high, for example 70%, only two nodes satisfy the condition, which means that only two weight messages are sent. As can be seen from Figure 5, reducing the threshold value allows more nodes to participate. The allowed nodes are five when the threshold is 60%, six for 50%, seven for 40%, eight for 30%, and nine for 20%.

According to the MACHM functionality, the winning cluster head continues its role in a normal way until its battery power level reaches the threshold. When this occurs, a cluster head reduces the threshold value by a predefined *battery_power_threshold_decrement* value. Also, it sends an invoke request message to all its neighbors. Receiving an invoke request message changes the behavior of MACHM. Now all the nodes are allowed to participate in the cluster head election process. In this simulation scenario, the heavy data traffic and the relatively small initial battery power levels for some nodes cause them to shut down earlier. In Figure 5, the second point of each line shows the number of network nodes that still alive in that simulation time.

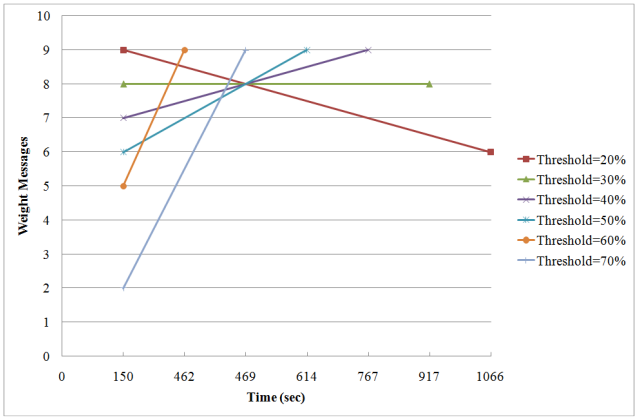

Figure 5. Weight messages sent during the simulation

On the other hand, the invoke request messages are used to notify the neighbor nodes that the current cluster head is looking for a better replacement (if available). This is why all the nodes will participate in the next cluster head election. Figure 6 shows that only one invoke request message is sent, which means that the new elected cluster head could survive till the end of the simulation. The chosen threshold affects how long the initial cluster head can survive in the network

 ISBN: 978-1-61208-308-7

before asking for a replacement. In general, it is noticeable that the lower the threshold value is the longer a cluster head survives and the later an invoke request message is sent. Figure 6 shows that the mechanism used by MACHM to elect the cluster heads is efficient, since it avoids multiple sending of invoke request message for the same threshold.

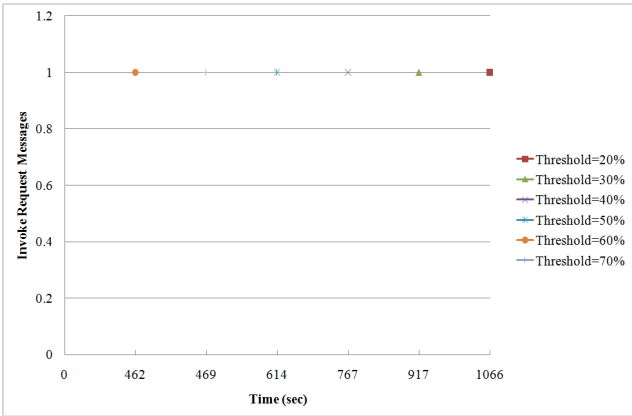

Figure 6. Invoke request messages sent during the simulation

In networks with large number of nodes, the number of cluster heads exist in the network gives a good idea about the balance and the distribution of the nodes. In this section, the chosen setup produces a connected network with no gaps. Figure 7 shows that only one cluster head exists for all the thresholds. The different points show the simulation time a new cluster head is introduced into the network. The first cluster head is chosen after some initializations, calculations, and weight messages exchange. The other cluster head is introduced into the network after receiving an invoke request message from the previously winning cluster head.

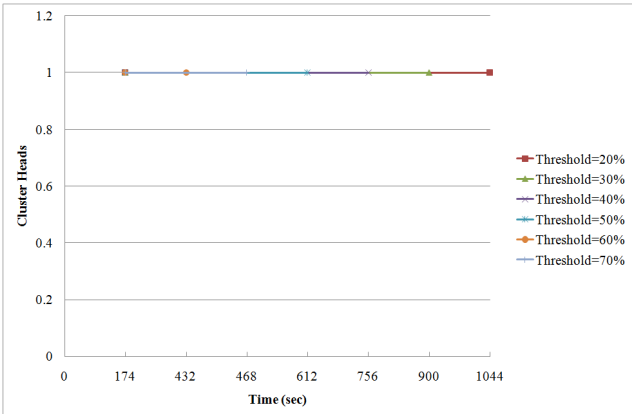

Figure 7. Total number of elected cluster heads during the simulation

MACHM aims to elect a cluster head that can survive for longer time to avoid the need of changing it frequently. This is can be achieved by implementing the multi-aware concept. MACHM makes a good combination of load balance, geographic distribution, speed, and power factors. Figure 8 shows the survive time for the first elected cluster head. It is noticeable that the survive time decreases when the threshold value increases. This is because the cluster head takes less time to reach the defined threshold, and then issues a re-election request to the neighbor nodes.

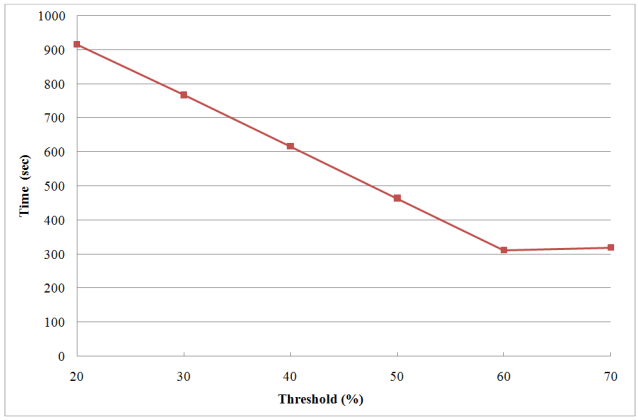

Figure 8. The survive time for the first elected cluster head

In OPNET, the information about node's battery power is not available as the other parameters. For this reason, MACHM creates its own battery energy consumption model. The created energy consumption model initializes the node's battery power levels with values between 1% and 100%. Also, it tracks and updates them through the life time of the network. The tracking and updating of those values is controlled by the current draw values. MACHM defines current draw values for the different node states, which are *SLEEP*, *IDLE*, *SEND*, and *RECEIVE* states.

Figure 9 shows the all nodes energy consumption for the VoIP simulation. The energy consumption varies according to the call buckets, showing the relatively low consumption values for *SLEEP* and *IDLE* states. While higher consumption values are recorded for the *SEND* and *RECEIVE* states.

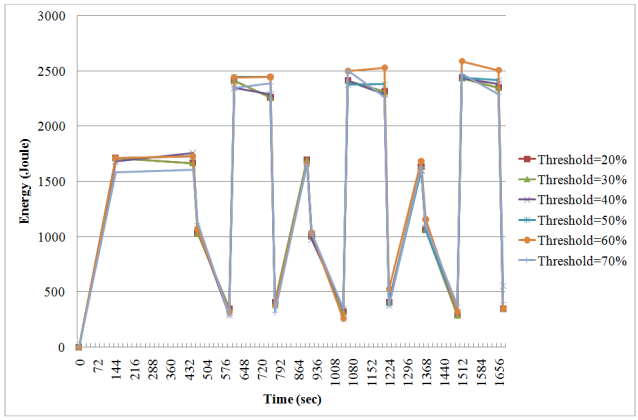

Figure 9. All nodes energy consumption during the simulation

This section showed the simulation results obtained for the new MACHM using the OPNET Modeler. The protocol itself is flexible and adjustable. It is important to select a

 ISBN: 978-1-61208-308-7

suitable combination of MACHM parameters setup to take advantage of its capabilities.

VI. CONCLUSION

OPNET Modeler is an important tool, used widely to model and study communication systems. MANETs attract many researchers because of the wide applications they can be used for. The aim of this paper is to guide the researchers who are interested in modeling and studying new MANET routing protocols using the OPNET Modeler. This paper proposed a practical methodology for adding such new MANET routing protocols to OPNET Modeler. It uses MACHM as an implementation example. A brief description of MACHM and the multi-aware concept is given. Also, the OPNET Modeler architecture is introduced for better understanding. The paper gives a detailed implementation of MACHM according to the proposed practical methodology, followed by general guidelines. The simulation study shows the performance evaluation of the implemented MACHM. Applying the ideas in this paper will accelerate the developing of new MANET routing protocols. This is because the researchers can take advantage of the powerful tools provided by the OPNET Modeler. This includes model design, simulation, data collection, and data analysis phases.

REFERENCES

[1] Michel C. Jeruchim, Philip Balaban, and K. Sam Shanmugan, “Simulation of Communication Systems: Methodology, Modeling, and Techniques,” New York: Kluwer Academic Publishers, October 2002, ISBN-10: 0306462672.

[2] http://www.opnet.com [retrieved: August, 2013].

[3] Yoav Sasson, David Cavin, and Andre Schiper, “A Location Service Mechanism for Position-Based Multicasting in Wireless Mobile Ad hoc Networks,” In Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS'05) - Track 9, vol. 9, 2005, pp. 321b-321b.

[4] L. A. Latiff, A. Ali, Chia-Ching Ooi, and N. Fisal, “Location-Based Geocasting and Forwarding (LGF) Routing Protocol in Mobile Ad Hoc Network,” In Proceedings of the Advanced Industrial Conference on Telecommunications/Service Assurance with Partial and Intermittent Resources Conference/E-Learning on Telecommunications Workshop (AICT-SAPIR-ELETE '05), IEEE Computer Society, Washington, DC, USA, 2005, pp. 536-541.

[5] X. Hong, K. Xu, and M. Gerla, “Scalable routing protocols for mobile ad hoc networks,” IEEE Network, vol. 16, no. 4, 2002, pp. 11–21.

[6] Petteri Kuosmanen, “Classification of ad hoc routing protocols,” Finnish Defence Forces, Naval Academy, Finland, 2002. Available from http://www.netlab.tkk.fi/opetus/s38030/k02/Papers/12-Petteri.pdf.

[7] P. Sinha, R. Sivakumar, and B. Vaduvur, “Enhancing Ad Hoc Routing with Dynamic Virtual Infrastructures,” In Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 3, 2001, pp. 1763-1772, Anchorage, USA.

[8] Z.J. Haas, and S. Tabrizi, “On some challenges and design choices in ad-hoc communications," Military Communications Conference, MILCOM 98. Proceedings., IEEE , vol. 1, October 1998, pp. 187-192.

[9] Benjie Chen, Kyle Jamieson, Hari Balakrishnan, and Robert Morris, “Span: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks,” Wireless Network Journal, vol. 8, issue 5, September 2002, pp. 481-494.

[10] R. Sivakumar, P. Sinha, and V. Bharghavan, “CEDAR: a core-extraction distributed ad hoc routing algorithm,” Selected Areas in Communications, IEEE Journal on , vol. 17, no. 8, August 1999, pp. 1454-1465.

[11] Jie Wu, Ming Gao, and I. Stojmenovic, “On calculating power-aware connected dominating sets for efficient routing in ad hoc wireless networks,” IEEE/KICS Journal of Communication Networks, vol. 4, no. 1, 2002, pp. 59-70.

[12] B. Liang, and Z.J. Haas, “Virtual backbone generation and maintenance in ad hoc network mobility management,” INFOCOM 19th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE , vol.3, March 2000, pp. 1293-1302.

[13] V. Hnatyshin, H. Asenov, and J. Robinson, “Practical methodology for modeling wireless routing protocols using OPNET Modeler,” In Proceedings of 21st International Conference on Modeling and Simulation (MS 2010), July 15 - 17, 2010, Banff, Alberta, Canada.

 ISBN: 978-1-61208-308-7