Point-to-Point OMNeT++ Based Simulation of Reliable Transmission Using Realistic Segmentation and Reassembly With Error Control

Erwin Anggadjaja and Ian McLoughlin
Earth Observatory of Singapore/School of Computer Engineering
Nanyang Technological University
Singapore 639798
er0001ja@ntu.edu.sg

***Abstract*—As many experimental simulations were conducted in the field of wireless sensor networks, OMNeT++ has gained popularity due its scalability and rich graphical user interface. The use of OMNeTT++ based simulation for reliable point-to-point wireless transmission and realistic Segmentation and Reassembly (SAR) with error control for a distributed wireless sensor network are explored. Simulations demonstrated that OMNeT++ is a reliable method for bridging theory and practice and building confidence in this tool for implementing future networks. A particular example of SAR with error control was implemented, and showed promising and realistic outcomes regarding the relationship between BER, PER, window size, and system efficiency.**

***Keywords*-BER, Error control, OMNeT++, PER, SAR.**

I. INTRODUCTION

Many researches based their results on experimental simulation. This stemmed from the need of a reliable model to foresee the performance of a real world environment, especially in the field of wireless sensor networks [1]–[3] and particularly for complex high-level data transmission protocols. This approach is usually good at tackling scalability issues (large number of nodes) and especially for systems with costly hardware components.

Several simulation frameworks have been widely used in networking and telecommunication research including ns-2, GloMoSim, OPNET, SensorSim, J-Sim, OMNeT++ etc [4], with most of the available network simulation tools were based on the paradigm of discrete event-based simulation [5]. OMNeT++ is chosen in this paper for the following reasons: OMNeT++ handles scalability well [6], OMNeT++ and along with ns-2 are particularly mature [2], and OMNeT++ provides a rich graphical user interface (GUI) with abstract modeling language.

Further research indicated that OMNeT++ has better performance than ns-2 in terms of execution time and memory overhead [7]. It also boasts an animated graphical user interface, and more convenient topology. Recent works by [8] and [9] also show that OMNeT++ has gained confidence as a simulator prior to implementing real model.

Motivated by the above, case studies of point-to-point wireless transmission which study the reliability of OMNeT++ are presented and showed how it can be used to simulate more advanced models of segmentation and reassembly (SAR) in wireless networks. In addition, an error control mechanism, i.e. number of retries and reACKs within the simulation model is also implemented.

To our knowledge, this is the first paper detailing OMNeT++ use with ACK based error control. The remainder of the paper is organized as follows. In Section II and III, OMNeT++ and the concepts of SAR are reviewed respectively. Section IV evaluates the reliability of OMNeT++ through simulation, continued with more advanced SAR simulation concepts with retries and reACKs in Section V. The result will be displayed and discussed before concluding in Section VI.

II. OMNET++

OMNeT++ (Objective Modular Network Testbed in C++) is an object-oriented modular discrete event simulator [10], mainly focused on the simulation of communication networks. Components (modules) are programmed in C++, and then assembled into larger components and models using a high-level language called NED. The active modules are termed simple modules (written in C++, using the simulation class library). Simple modules can be grouped into compound modules (Fig. 1), and communicate with each other through message via gates. Both gates, input and output, are linked with a connection. For each module, properties are assigned, such as such as propagation delay, data rate and bit error rate and parameters (string, boolean, character, numeric, constants). Since it is based on C++, modules (and sub-modules) are able to pass parameters to each other and through inheritance.

Although OMNeT++ is not a network simulator itself, it is currently gaining widespread popularity as a network simulation platform in the scientific community as well as in industrial settings, mainly due to ease of use and comprehensive capabilities.

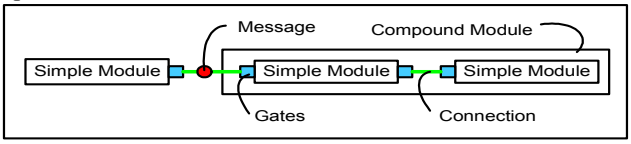

Figure 1. OMNeT++ Model Structure.

978-0-7695-4269-0/10 $26.00 © 2010 IEEE
DOI 10.1109/ACT.2010.25

Other available frameworks supported include INET (contains models for several Internet protocols like TCP, UDP, IP, etc), INETMANET (INET with mobile/ad-hoc protocols), Mobility Framework (contains a library of wireless sensor networks simulation models for OMNeT++), and the promising MiXiM project [11].

III. SAR With Error Control

Segmentation and Reassembly (SAR) is one of the concepts used to fragment and re-accumulate packets as it allows them to be safely transported across error-prone transmission links and networks. The data to be transmitted is simply chopped up into smaller segments (according to the desired size), augmented with a header (as shown in Fig. 2) and perhaps some error check or error code information, and at the other end these segments are then fitted back together to reform the original packet. Then an error control mechanism [12] (just as TCP does) was added for simulating more realistic models.

In our work, the implementation was limited in terms of number of retries and reACKs, both were used in conjunction with a timeout counter. A sender will send retries if it does not receive an appropriate ACK for the corresponding packets while a receiver will send reACKs if it gets the same packet from the sender (indicates a loss in ACK).

IV. Reliable Point-to-Point OMNeT++ Simulation

A. Modelling and Scenario

In this section, a simple concept of lost and received packets was simulated in order to gain confidence in using OMNeT++. The simulation was established by two nodes defined to be NodeA and NodeB through wireless transmission. NodeB will stop the simulation once it received n-correct packets regardless the numbers of the actual number of packets sent/resent by NodeA. The environment was simulated under the condition of variable Bit Error Rate (BER), recording the link statistics as the simulation progressed.

Assume:

$$Np = Lp + Rp. \quad (1)$$

The relation of Packet Error Rate with BER is:

$$PER = 1-(1-BER)^{N}. \quad (2)$$

Np is total number of packets sent by NodeA, *Rp* is the number of received packets at NodeB, *Lp* is the lost packets in the air, and *N* is the uniform length of each packets in bits.

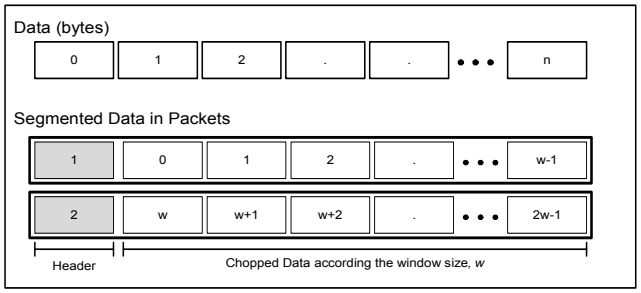

Figure 2. Segmented Data.

Since:

$$Lp = PER * Np \quad (3)$$

then it can be concluded that:

$$Np = \frac{R_p}{(1-PER)} \quad (4)$$

and

$$Lp = \frac{R_p}{(1-PER)} - R_p. \quad (5)$$

The simulation was run with the conditions stated in Table I. The results of comparing the theories in (4) & (5) were shown in Fig. 3 and Fig. 4.

B. Discussion

After comparing the lost packets to BER as referred in (2), it is obvious that smaller packet length of N = 1024, will resulted lower PER, compared with a larger packet length of N = 2048. Small PER will resulting in the loss of fewer packets, as suggested in (3). As shown in Fig. 3, a significant increase in the number of lost packets for bigger packet length (at constant BER value = 10^{-3}) can be observed. In terms of BER, progressing from bad to good, the results showed that most of the packets were lost (213 packets up to 811 packets) for small BER. Conversely, for good BER the number of lost packet is very small (i.e. approximately zero). Since for BER = 10^{-3}, as expected, bigger packets showed a strong tendency to be lost.

From the angle of sent packets (depicted in Fig. 4), the outcome was similarly encouraging. For very bad BER, more packets were sent from NodeA since most of them are lost in the transmission. When it comes to a good BER transmission, the number of sent packets was equal to 125 (*Rp*) which indicated that no packets were lost in the transmission. For both scenarios, a very encouraging conclusion can be observed that the discrepancies between theory and experiments are very minor thus builds confidence in the reliability of OMNeT++.

V. Point-to-Point SAR With Error Control

A. Modelling and Scenario

The modelling was started with a simple concept of segmentation and reassembly (SAR). NodeA needed to break the data into the appropriate window size (segmentation), added a sequence number before it began to transmit them to NodeB, then it defined the total number of segments needed to be sent. NodeB received all the segments until no more segments are left. If the segments were correctly received, reassembly will occur and full output data will be generated. Besides SAR, now both nodes have an error control mechanism.

NodeA must resend the segments that were lost in the transmission, while NodeB had to resend the ACK (acknowledgement) when it received the same segment from NodeA (which means the previous ACK is lost). The maximum number of retries and re-ACK were set to 3 (which can be changed easily in the simulation framework).

TABLE I
Simulation Parameters

Attribute	Value
Rp	125
BER	10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
N	1024, 2048

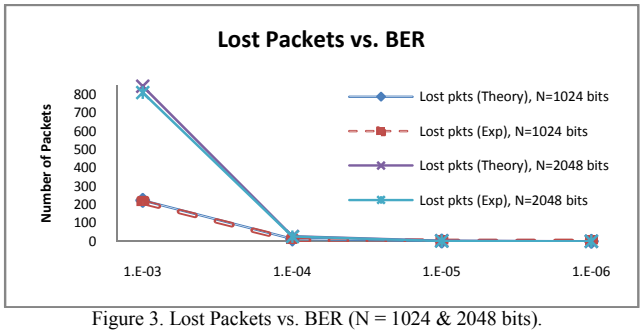

Figure 3. Lost Packets vs. BER (N = 1024 & 2048 bits).

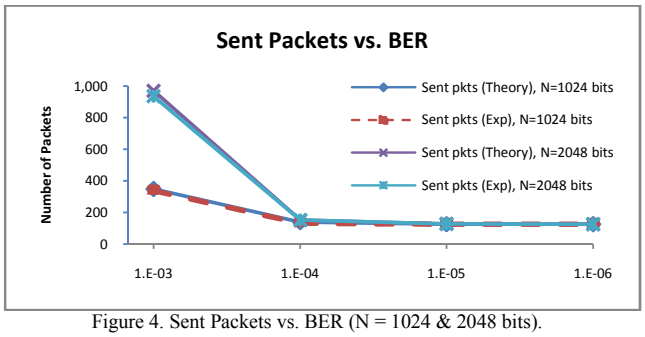

Figure 4. Sent Packets vs. BER (N = 1024 & 2048 bits).

The flowchart of the simulation is depicted in Fig. 5 for both NodeA and NodeB, while the results were plotted in Fig. 6 and Fig. 7, in terms of BER and PER for different data sets, number of retries & reACKs, and window size.

B. *Discussion*

In the relation with BER and PER the result showed that, as expected, low BER caused more data to be lost in transmission, thus increasing the PER percentage.

The percentage value itself was increasing gradually as the window size increasing. A very careful choice of window size is suggested to maximize system efficiency [13]. In terms of retries and reACKs, more segments were obviously re-sent in bad BER conditions. It is interesting that for different window size graphs, the plot showed similar values, which can lead to the conclusion here that increasing window size does not make the retries and reACKs values reduce.

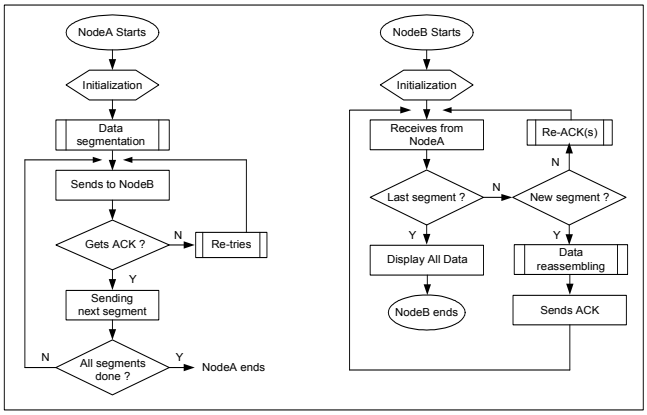

Figure 5. NodeA & NodeB Flowchart.

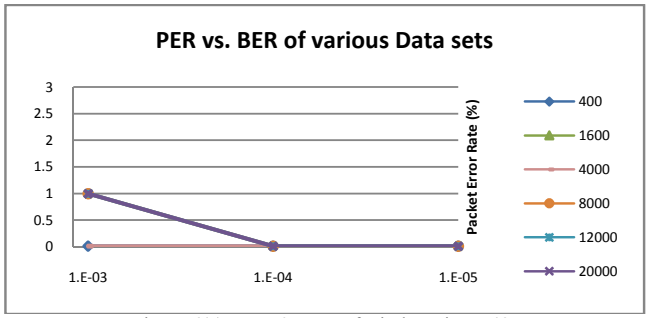

Figure 6(a). PER & BER of window size = 49.

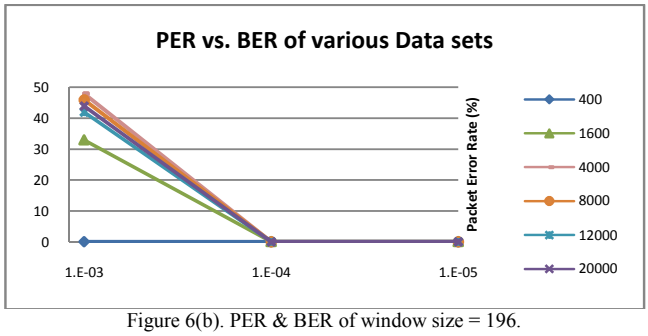

Figure 6(b). PER & BER of window size = 196.

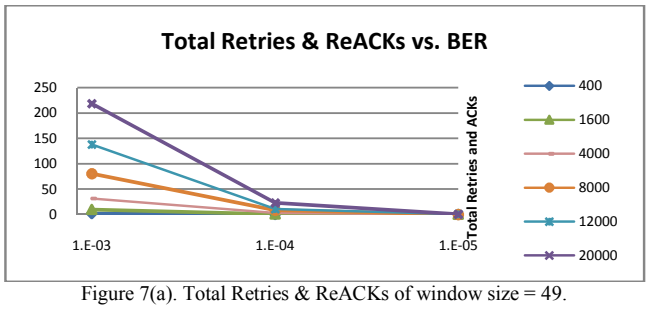

Figure 7(a). Total Retries & ReACKs of window size = 49.

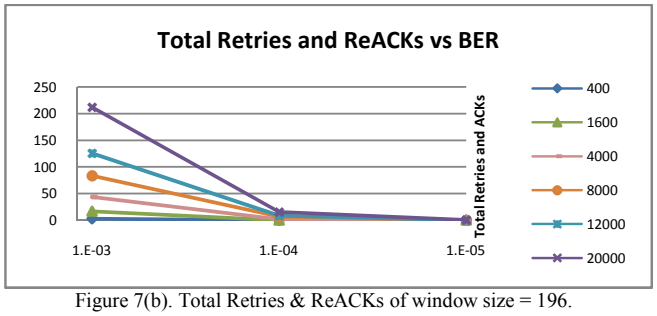

Figure 7(b). Total Retries & ReACKs of window size = 196.

Lasty, some parameters were altered to determine the relationship between number of retries/reACKs and (a) BER/PER of received segments, (b) total bits sent over the air. To identify the correlation, the value of retries/reACK was changed from 3 to 5. For these experiments, the BER value of 10-3 was chosen, and the results were shown in Table II. When the maximum number of retries were increased from 3 to 5, PER also reduced (meaning that all the segments have greater probability of arriving at NodeB) and total bits sent over the air increased (this can be seen also from the increasing number of retries and reACKs).

The system efficiency was also analyzed by calculating the total bits received over the total bits sent. The result showed that the efficiency decreased when the window size increasing. It is due to the straightforward correlation that when part of any segments is lost, so are all of the bits contained within it. For bigger window sizes, the probability of a segment being lost is higher. However we found something interesting in terms of the number of bits sent when the window size was extended from 49 to 196: the number of bits sent increased to almost 60%. This was a trade-off that needed to be considered in terms of system efficiency and window size within any system.

VI. CONCLUSION

OMNeT++ operates as a reliable simulator for studying the concepts of simple wireless transmission through developing a SAR simulation model incorporating error control mechanisms. Parameters such as BER, PER, effects of window size, maximum number of retries and reACKs were observed. It is shown as expected that the probability of lost packets is very low for good BER value (10-4 and above, with the tested window sizes) and bigger packets have greater tendency to become lost for the same BER value. For poor BER values, the number of segments needed to be sent is higher, since most of them will be lost in transmission. This also happens for the lengthier packets. In the last section, our confidence in using OMNeT++ simulation was built up showing that PER reduces through changing the maximum number of retries from 3 to 5, and that total bits sent over the air increases. The trade-off between window size and system efficiency was also noted.

In conclusion, OMNeT++ is trustworthy, capable and relatively fast (the slowest simulations in this paper completed in no more than a few tens of minutes). Having this validation of OMNeT++ for a relatively simple wireless transmission system using SAR with ACK and limited retries—where performance against theoretical solutions can be compared—it is now possible to extend the system to more complex wireless systems in which theoretical or closed-form solutions are not readily obtainable.

REFERENCES

[1] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in *1st Int. Conf. Simulation tools and techniques for Communications, Networks and Systems,* Belgium, 2008, pp. 1-10.

[2] J. Lessmann, P. Janacik, L. Lachev, and D. Orfanus, "Comparative study of wireless network simulators," in *Proc. 7th Int. Conf. Networking,* 2008, pp. 517-523.

[3] J. Zhang, J. Chen, J. Fan, W. Xu, and Y.Sun, " OMNeT++ based simulation for topology control in wireless sensor network: a case study," in *Int. Wireless Communications and Mobile Computing Conf.,* 2008, pp. 1130 – 1134.

[4] D. Curren, (2005). A survey of simulation in sensor networks. University of Binghamton. Available e-mail: bj92489@binghamton.edu.

[5] G. S. Fishman. *Principles of Discrete Event Simulation*. John Wiley & Sons, Inc., New York, NY, 1978.

[6] E. Weingärtner, H. Vom Lehn, and K. Wehrle, "A performance comparison of recent network simulators," in *ICC 2009: IEEE Int. Conf. Communications*, 2009, pp. 1-5.

[7] X. Xian, W. Shi, and H. Huang, "Comparison of OMNET++ and other simulator for WSN simulation," in *3rd IEEE Conf. Industrial Electronics and Applications,* 2008.

[8] Z. Zhang, Z. Pang, J. Chen, Q. Chen, h. Tenhunen, L. Zheng, and X. Yan, "Two-layered wireless sensor networks for warehouses and supermarkets," in *3rd Int. Conf. Mobile Ubiquitous Computing, Systems, Services and Technologies*, Sliema, 2009, pp. 220-224.

[9] J. Maureira, P. Uribe, O. Dalle, T. Asahi, and J. Amaya, "Component based approach using OMNeT++ for train communication modeling," in *2009 9th Int. Conf. Intelligent Transport Systems Telecommunications (ITST)*, 2009, pp. 441-446.

[10] http://www.omnetpp.org.

[11] A. Köpke, M. Swigulski, K. Wessel, D. Willkomm, P. T. K. Haneveld, T. Parker, O. W. Visser, H. S. Lichte, S. Valentin, "Simulating wireless and mobile networks in OMNeT++ -- the MiXiM vision," in *2008 Proc. 1st Int. Workshop on OMNeT++*, Belgium, 2008, pp. 1-8.

[12] B. A. Forouzan, *TCP/IP ProtocolSuite 2nd Ed*. McGraw-Hill, 2003.

[13] M. Thakur, H. Sirisena, and I.V. McLoughlin, "Link layer mitigation in rural UHF-MIMO linking systems," in *TENCON 2005 IEEE Region 10 ,* 2005, pp. 1-6.

TABLE II
COMPARISON OF PACKET ERROR RATE AND TOTAL BITS SENT ON 3 AND 5 RETRIES

(a) With window size = 49

Data Length (bytes)	3 Retries					5 Retries				
	PER (%)	Bits sent (bits)	NodeA Retries	NodeB ReACKs	Efficiency	PER (%)	Bits sent (bits)	NodeA Retries	NodeB ReACKs	Efficiency
12000	1.22	94424	123	15	98.36%	0.00	95600	128	15	99.58%
50000	1.18	393696	499	68	98.42%	0.10	398008	516	68	99.50%
100000	1.37	785824	995	129	98.23%	0.10	796016	1037	129	99.50%

(b) With window size = 196

Data Length (bytes)	3 Retries					5 Retries				
	PER (%)	Bits sent (bits)	NodeA Retries	NodeB ReACKs	Efficiency	PER (%)	Bits sent (bits)	NodeA Retries	NodeB ReACKs	Efficiency
12000	41.94	55232	123	3	57.53%	27.42	69344	180	3	72.23%
50000	39.45	241632	490	9	60.41%	22.27	309048	671	12	77.26%
100000	38.75	487960	976	22	61.00%	23.48	608688	1324	26	76.09%