

# Motes Sensor Networks in Dynamic Scenarios: an Experimental Study for Pervasive Applications in Urban Environments

Giuseppe Anastasi\*, Marco Conti<sup>†</sup>, Enrico Gregori<sup>†</sup>, Carlo Spagoni\*, Giuseppe Valente<sup>†</sup>

\*Dept. of Information Engineering, University of Pisa, email: {firstname.lastname}@iet.unipi.it

<sup>†</sup>Institute for Informatics and Telematics, National Research Council (CNR), email: {firstname.lastname}@iit.cnr.it

**Abstract**—Recent works have shown that using mobile elements (or data MULEs) to collect and carry data from sensor networks to a collection point may have significant advantages over traditional ad hoc sensor networks. Previous papers on the MULE architecture are mainly based on simulation and numerical analysis. In this paper we investigate the performance of the data MULE model and its ability to support sensor-based applications in a real urban environment by means of an experimental analysis. Specifically, we use a testbed based on Berkeley motes, and analyze the impact of several parameters (i.e., MULE’s speed, distance between the static node and the MULE, duty cycle) on the contact time and the total amount of data that a static node is able to transfer to the mobile MULE when they happen to get in contact. The results obtained are very promising and show that the MULE architecture is really suitable for a large set of sensor-based applications in urban environments, both in low and high mobility scenarios.

**Index Terms**—Sensor Networks, Data MULEs, Motes, TinyOS, Mobility

## I. INTRODUCTION

Mark Weiser envisioned a world in which computing is so pervasive that everyday devices can sense their relationship to us and to each other [1]. This vision is based on the assumption that sensing a set of physical phenomena, rather than just data input, will become a common aspect of small, embedded computers and that these devices will communicate with each other to organize and coordinate their actions. Wireless sensor networks thus constitute a basic component for the development of pervasive applications.

Sensor nodes can be networked together to enable a wide variety of applications that involve environmental monitoring. Data collection in such applications usually happens infrequently. Sensors may be spread over a large geographical area resulting in a sparse network, which brings to two possible approaches to ensure connectivity: the installation of multiple base stations to relay the data from sensor nodes in their coverage area, or the deployment of enough low-power relay nodes to effectively form a dense connected network [2]. The base station approach trades off the communication power needed by the sensors with the installation costs of the base stations. On the other hand, deploying nodes to form a dense, fully-connected ad-hoc network may not be cost-effective either.

An architecture involving data MULEs (Mobile Ubiquitous LAN Extensions), as described in [3], would have the advan-

tages of both approaches, and would be easy to achieve in the case of urban environmental monitoring as this role could be served by vehicles outfitted with transceivers. By definition, MULEs are assumed to be capable of short-range wireless communication and can exchange data from sensors along the way. They should be capable of picking up data from the (static) sensors encountered along their path, buffer data and then retransfer it to a data collection point. This approach has the advantage of large power savings that occur at the sensors because the communication takes place over a short-range. A disadvantage would be the increased latency because sensors have to wait for a MULE to approach them for the data transfer to occur. However, in data collection applications in urban environments a relatively large latency is usually acceptable.

The purpose of this work is to investigate the performance of the data MULE model and its ability to support sensor-based applications (e.g., environmental monitoring applications) in a real urban environment. For this aim, we performed an extensive set of experiments on experimental testbed in a realistic urban environment. Our testbed consisted of one mobile sensor node, acting as the MULE, that collects data from a static sensor node. In the real world we could think of the static node as a sensor placed at a bus stop or anywhere along a street or a park, whereas the mobile node could be a person walking along the street, a bus, a cab, etc. In such a scenario it is important to evaluate the *contact time*, i.e., the time interval during which the static node and the mobile MULE are able to communicate with a given communication quality. In addition, it is important to estimate the amount of data the MULE is able to receive correctly from the static sensor node every time they occur to get in contact. Obviously, the above quantities depend on several physical parameters like distance between nodes and MULE’s speed. Furthermore, as the static node may be energy constrained (the MULE’s energy is assumed to be renewable), power management is another important issue that needs to be analysed very carefully [4], [5].

We performed an extensive measurement campaign on our experimental testbed. Specifically, we used Mica2 Berkeley Motes as sensor nodes, and considered different scenarios and operating conditions. We investigated the performance of the system both in terms of contact time and amount of data transferred during each contact. The results obtained are very encouraging and show that the MULE model is really suitable

for most sensor-based applications in an urban environment.

The rest of the paper is organized as follows. Section II describes related works. Section III introduces the MULE architecture. Section IV describes the experimental testbed and the methodology used in our analysis, while the experimental results are presented in Section V. Finally, some conclusions are drawn in Section VI.

## II. RELATED WORK

Wireless sensors are currently a very hot topic within the research community, as witnessed by the increasing number of papers on that topic. Considerable work has been done on static sensor networks, but only few papers refer to dynamic sensor networks. In addition, most of the papers appeared in the literature are based on simulation and/or numerical analysis, while very few of them rely on experimental measurements. The importance of an experimental approach in evaluating wireless networks is discussed in [6], [7], [8]. Specifically, an experimental approach runs into many difficulties, such as the dependence of wireless links' quality on several environmental parameters that results in asymmetric links, non-isotropic communication zones and large gray zones [9]. These problems do not arise in simulation experiments.

In [10] the authors perform an extensive measurement study on a real testbed based on Berkeley motes. However, they focus on static sensor networks. The analysis there is aimed at evaluating the real performance of Mica2 and Mica2dot motes in terms of maximum transmission range, and maximum achievable throughput. Several physical parameters are taken into account, such as environmental conditions (weather conditions, humidity), node distance from the ground, transmission data rate. The present paper takes a similar approach but focuses on a mobile sensor network and, specifically, on the MULE architecture.

The MULE architecture was first proposed in [3] to address the problem of energy-efficient data collection in sparse sensor networks. In a subsequent paper [5] the same authors presented an analytic model and a simulation approach to evaluate the key performance indices of the proposed MULE architecture. Specifically, they investigated the impact on the data success rate, latency, and energy cost due to a large set of operating parameters: sensor related parameters (data generation rate, sensor buffer size, sensor duty cycle), MULE related parameters (MULE inter-arrival distribution and MULE buffer size), Access Point related parameters (distribution and number of Access Points) and radio related parameters (radio transmission range and bit rate). In the present work we focus on the MULE architecture as in [5], but we take an experimental approach. Instead of simulation we rely on measurements on real Mica2 Berkeley Motes in a realistic urban environment.

As the use of a single MULE may be insufficient for data collection, in [11] the authors investigate the use of multiple MULEs. In particular, they consider load balancing to balance the number of static sensor nodes served by a MULE, and show by simulation the benefits of load balancing.

In [12] it is shown that predictable mobility can be used to significantly reduce the energy consumption in sensor

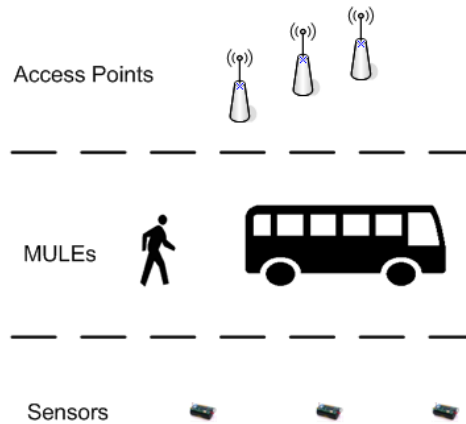


Fig. 1. Data MULE three-tier architecture

networks. The MULE is mounted on a public transportation bus moving with a periodic schedule. The static sensor nodes learn the times at which they have connectivity with the MULE and wake up accordingly to transfer their data. As the sensor nodes only transmit when the MULE is close to them, the energy consumption is significantly reduced. Controlled mobility is also used in [13], [14], [15], [16].

## III. THE MULE ARCHITECTURE

In applications involving sensor nodes, power budgets' limitation is usually the most critical constraint. As discussed in [3], in a delay-tolerant scenario a three-tier MULE architecture provides the benefits of both the infrastructure and the infrastructure-less approach, in terms of sensor energy consumption, data success rate and infrastructure cost. It comprises of a three-tier layered abstraction (Fig. 1) that can be adapted to different situations and needs:

- a top tier for WAN connection;
- a middle tier of MULEs;
- a bottom tier of static wireless sensor nodes.

The top tier is composed of Access Points (i.e., central data repositories) and is meant to be set up at convenient locations, where reliable network connectivity and power are guaranteed. Access Points should communicate with a central data warehouse to synchronize the data they collect, and if needed they should return acknowledgments to the MULEs.

The intermediate layer of mobile MULE nodes is characterized by large storage capacities (relative to sensors), renewable power, and the ability to communicate with both sensors and networked Access Points. As a result of their motion, MULEs collect and store data from the sensors, and possibly deliver acks back to the sensor nodes and/or communicate with each other to improve system performance (for example to reduce latency between the two top tiers). The intermediate layer provides the system with scalability, flexibility and robustness for a relatively low cost (and if we consider sensor nodes to be the MULEs, the cost is decreasing over the years). No sensor depends on any single MULE, and hence the failure of any particular MULE does not disconnect the sensor from the

sparse network. It only degrades the performance. Moreover, the deployment of new sensors or MULEs does not require any network reconfiguration.

The bottom tier normally consists of randomly distributed wireless sensor nodes. In our case however, the placement is strategic depending on the purpose of the application. Anyway, the work performed by sensor nodes has to be minimal, as their constraint on resource usage is the highest of all tiers.

The three-tier architecture supports increased reliability as the redundant Access Points and multiple MULEs create a fault-tolerant system where failures can only lead to reduced data success rate and increased latency [17], [11]. The benefits of the MULE architecture can be summarized as follows:

- less infrastructure than a fixed base-station approach is required
- given a sufficient density of MULEs, the system is more robust than a traditional fixed network;
- the endpoint applications do not need to be aware of the mobile portions of the network;
- the system's flexibility allows the same transport medium to be used simultaneously by different applications.

On the other hand, this approach may lead to the following problems:

- the high latency introduced by the MULE architecture may limit the set of applications that can rely on this architecture: deterministic delay bound guarantees can be achieved only if MULEs maintain fixed routes;
- the MULE model presupposes a sufficient amount of physical movement in the environment.

For our study we focused on the lower two tiers, which are the most critical for data transfer and power management.

#### IV. EXPERIMENTAL ENVIRONMENT AND METHODOLOGY

Our experimental testbed is based on Mica2 Motes with the TinyOS operating system, and laptops running alternatively Linux Debian and Windows Xp and equipped with the TinyOS development kit and MIB510 sinks.

TinyOS is a component-based operating system for sensor networks developed at UC Berkeley [18]. It is an advanced software framework supported by a large user and developer community due to its open source nature. It contains many pre-built sensor applications and algorithms, like ad hoc multi-hop routing for example, and supports different sensor node platforms. TinyOS applications are developed in NesC [19], a programming language for networked systems designed to embody TinyOS' structuring concepts and execution model.

We ran our code first under the TOSSIM [20] environment (a simulation environment to test NesC programs), and after a long testing we compiled and installed the programs on the Mica2 motes. For our experimental tests, we developed a tiny packet generator sending packets at constant intervals. Initially, we used the timer model provided by TinyOS, which has a resolution of 32 ms. Then, to increase the precision in transmission timing, we used the Su Ping's timer implementation [21] which has an accuracy of 1/4.096 ms. To flash the motes we linked them through the parallel port to a MIB510 sink. To retrieve the data to analyze we linked the sink to our computer

through the serial port. The MIB510 sink is a programming board that acts as gateway for the Mica2 motes to personal computers. Through the sinks we can flash our NesC compiled programs to the motes and retrieve or upload data from or to the motes. For our tests we used a sink to program the motes and to retrieve the transferred data and process it with a Java application.

We ran our experiments in a big parking area at the CNR campus at Pisa, a realistic urban scenario because during the tests cars happened to pass by and between motes. The actors were a mobile sensor node (the MULE), trying to collect the data packets sent by a static sensor node (see Fig. 2).

In our experiments we measured the system's performance in terms of *contact time* and total number  $N_{rx}$  of packets successfully transferred from the static node to the mobile MULE during the contact. These indices allow us to evaluate the suitability of the MULE architecture to support different kinds of pervasive applications in an urban environment.

In many papers the radio range  $R$  of the static node is assumed to be fixed. Hence, the packet loss experienced by the mobile MULE is assumed to be 100% when the distance between the mobile and the static node is greater than  $R$ , and immediately drops to 0% as soon as the MULE enters the radio range of the static node. Under this hypothesis, the contact time is defined in [5] as the time interval during which the mobile node is within the radio range of the static node. The contact time can be derived analytically in terms of the radio range  $R$ , the distance between nodes, and the MULE's speed [5]. Accordingly, the number of packets  $N_{rx}$  correctly received by the MULE can be easily derived based on the contact time and the bit rate of the wireless link (which is assumed constant as well).

The above communication model is simple and analytically tractable. However, it is not completely realistic for the following reasons. First, the transmission range is not perfectly circular since the wireless medium is non-isotropic. In addition, the packet loss experienced by the mobile node does not drop immediately to zero as soon as the MULE enters the static node's radio range. It progressively tends to zero as the mobile node approaches the static one. Moreover, it does not exhibit a regular behavior. It may happen that the packet loss increases temporarily even if the distance between nodes decreases (see Fig. 3).

Based on the above evidences we need to consider a different, more realistic, definition of contact time. In our analysis the contact time is defined as the total time during

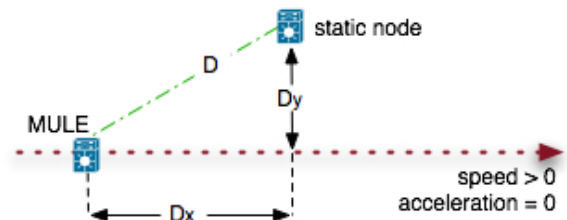


Fig. 2. Testbed environment

which the packet loss experienced by the MULE is below a given threshold (e.g., 15%). The contact time thus gives an indication of the amount of time during which the MULE is able to receive data from the static node with a given communication quality (expressed in terms of packet loss). In our experiments we also measured the total number of packets correctly received by the MULE during the contact,  $N_{rx}$ .  $N_{rx}$  provides an indication of the total amount of data that can be exchanged between the nodes. It may be worthwhile pointing out that, due to the above definitions,  $N_{rx}$ , can no longer be derived immediately from the contact time. In general,  $N_{rx}$  may be *greater* than the number of packets received (correctly) by the MULE during the contact time.

We evaluated the effect on the aforementioned performance indices of the following parameters:

- MULE's speed
- perpendicular distance between the static node and the MULE's line of motion ( $D_y$  in Fig. 2)
- packet generation rate at the static node
- duty cycle

We considered a reference configuration whose parameter values are shown in Table I. Throughout the paper, unless stated differently, the parameter values in Table I are used. In the reference configuration the static node is always on (100% duty cycle), and sends a 17-byte packet every 100 ms. The MULE moves at a speed of 1 m/s along a trajectory which is 25 m far from the static node. With reference to Fig. 2 we have the following values.

$$\begin{cases} D_x \in [-90; 90] \\ D_y = 25 \end{cases}$$

Starting from the reference configuration we varied the value of each single parameter in order to analyze the effect of that parameter on the performance indices. We replicated all the experiments several times to increase the accuracy of the results. The results presented below are averaged on all the replicas.

## V. EXPERIMENTAL RESULTS

As mentioned above, in an urban environment the role of MULE may be played by different actors (e.g., persons, buses, cabs) and, hence, MULEs may have different speeds. Therefore, we decided to divide our analysis into two parts. First, we considered a *low mobility* scenario assuming that MULEs are pedestrians and, hence, their speed is limited in the range [1, 2] m/s. In the second part, we considered a *high mobility* scenario where MULEs are buses (or cabs) and their speed varies in the range [20, 40] km/h.

TABLE I  
REFERENCE CONFIGURATION

MULE's speed	1 m/s
MULE's trajectory length	180 m
$D_y$	25 m
packet generation period (static node)	100 ms
duty cycle (static node)	100%
packet size	17 bytes

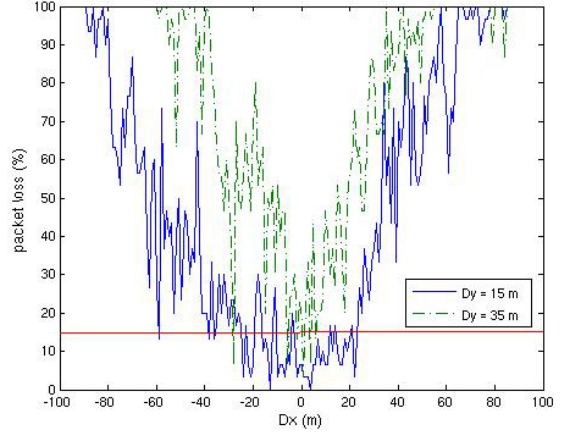


Fig. 3. Impact of distance in low mobility scenarios

### A. Low Mobility Scenario

The low mobility scenario is characterized by a limited range of speeds (from 1 to 2 m/s). In a such a scenario, in addition to the MULE's speed, the distance between the nodes, and the duty cycle of the static node may impact both the contact time and the total number of packets received by the MULE. Fig. 3 shows the packet loss as a function of the distance between the nodes. In our experiments we considered three different values for the perpendicular distance,  $D_y$ , between the static node and the MULE's line of motion: 15, 25 and 35 m. For the sake of clarity in Fig. 3 the curve at 25 m is omitted (the results for 25 m are in between those for 15 and 35 m, respectively). From the packet loss curve we can easily derive an estimate for the contact time. Assuming a threshold value for the packet loss of 15% we have approximately the following values for the contact time: 61 s (15 m), 36 s (25 m), and 12 s (35 m).

From the above results it emerges that the perpendicular distance  $D_y$  has a strong impact on the contact time. A similar effect can be observed if we look at the total number of packets successfully transferred to the MULE when it is in the proximity of the static node (see Table II).

TABLE II  
NUMBER OF PACKETS SUCCESSFULLY TRANSFERRED TO THE MULE FOR DIFFERENT  $D_y$  VALUES

distance $D_y$	avg	min	max
15 m	920	878	994
25 m	526	499	569
35 m	417	372	456

To evaluate the effect of the MULE's speed on the contact time and  $N_{rx}$  we performed some experiments at a speed of 2 m/s (all the other parameters being as in Table I). As expected, the contact time is approximately halved when passing from 1 m/s to 2 m/s. The number of packets transferred successfully to the MULE is shown in Table III). We can see that in the low mobility scenario, variations in the MULE's speed does not produce a very significant impact on the performance indices.

This is because in such a scenario speed variations are limited.

TABLE III

NUMBER OF PACKETS SUCCESSFULLY TRANSFERRED TO THE MULE FOR DIFFERENT MULE'S SPEEDS

speed	avg	min	max
1 m/s	526	499	569
2 m/s	306	255	345

So far we have assumed that the static node is always on. This would not be realistic in many situations as the sensor node has a limited energy budget that often cannot be renewed. To investigate the effect of power management we performed a set of experiments with the static node operating with a duty cycle lower than 100%. Tiny OS allows 7 different duty cycle modes that are listed in Table IV. We ran experiments with Motes' duty cycle set on modes 0 (100%), 4 (5.61%), 5 (2.22%) and 6 (1%).

TABLE IV

DUTY CYCLE MODES IN TINYOS

Mode	Duty Cycle
0	100 %
1	35.5 %
2	11.5 %
3	7.53 %
4	5.61 %
5	2.22 %
6	1.00 %

Figures 4-6 show the average number of packets successfully transferred to the MULE in the four different duty cycle modes described above, for different  $D_y$  values. As expected, there is a significant reduction in the  $N_{rx}$  average value. However, even with a very low duty cycle (e.g., 1%) and a distance  $D_y = 35$  m, the average number of packets correctly transferred to the MULE is 35, which should be enough for most of applications.

In conclusion, in the low mobility scenario the distance  $D_y$  has a significant impact on both the contact time and  $N_{rx}$  value. The MULE's speed has a negligible effect due to the limited range of speeds in such a scenario. Finally, it is possible to set a very low duty cycle (of 1-2%) at the static sensor node, thus saving a lot of energy, still achieving a good communication quality, in terms of total number of packets successfully transferred to the MULE.

### B. High Mobility Scenario

A MULE scenario in an urban environment can theoretically involve mobile nodes attached to anything. For example we can think of buses acting as MULEs, as they are intrinsically characterized by coordination and attitude to "full coverage" of a zone. Based on the above remarks, we extended our analysis to a high mobility scenario where the MULE moves at speeds typical of vehicles in residential areas (20 to 40 km/h).

By the time we were planning those experiments we observed that, at these speeds, when  $D_y > 15$  m the communication becomes very difficult and unreliable. Therefore, we

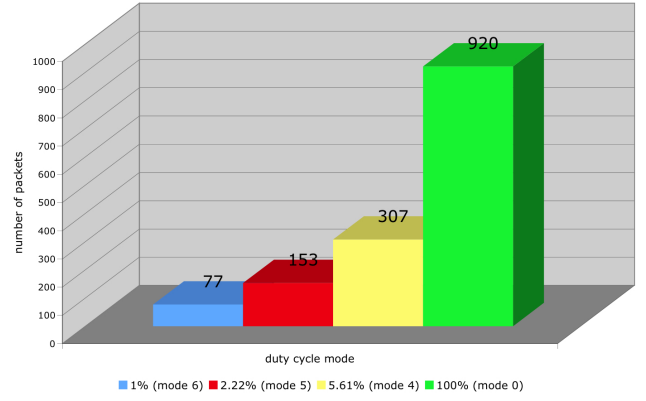


Fig. 4. Average number of packets successfully transferred to the MULE in the low mobility scenario for different duty cycles when  $D_y=15$ m

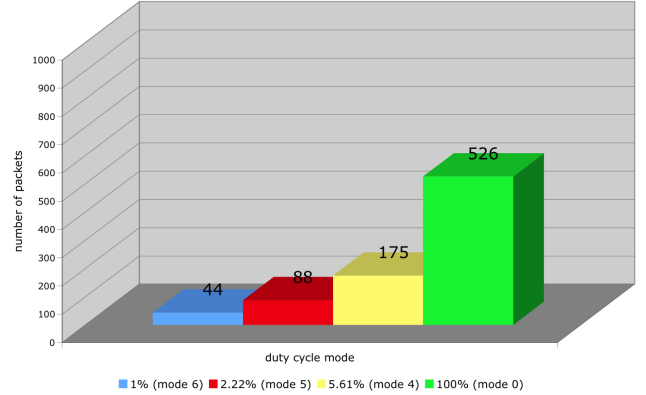


Fig. 5. Average number of packets successfully transferred to the MULE in the low mobility scenario for different duty cycles when  $D_y=25$ m

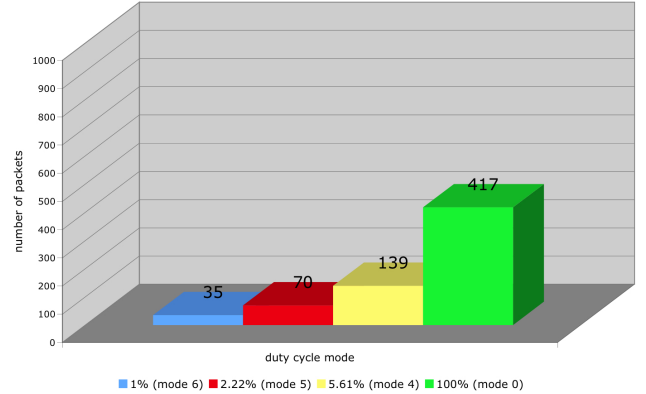


Fig. 6. Average number of packets successfully transferred to the MULE in the low mobility scenario for different duty cycles when  $D_y=35$ m

decided to run this second set of experiments at a distance  $D_y = 15$  m.

Fig. 7 shows the packet loss as a function of the distance  $D_x$  measured along the MULE's line of motion, for different MULE's speeds. As expected, the contact time becomes shorter and shorter as the MULE's speed increases (Fig. 7). With a MULE moving at 20 km/h, the two nodes are able to communicate with a packet loss lower than 15% for no more than 10 meters ( $-4 < D_x < 6$ ), resulting in a contact time reduced to around 2 s. In the case of a MULE moving at 40 km/h the contact time is approximately 0 s. However,

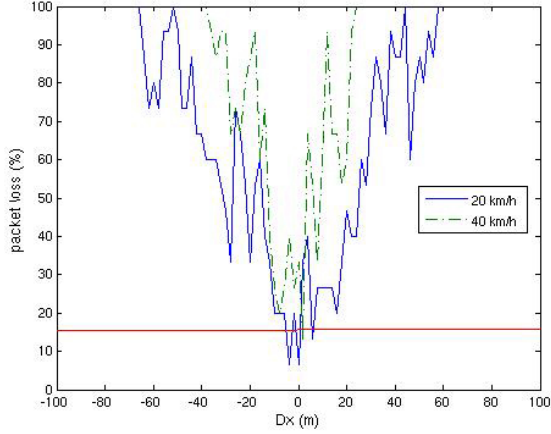


Fig. 7. Impact of the MULE's speed on packet loss and contact time

even with such low values of the contact time, the number of packets successfully transferred to the MULE is not negligible, as shown in Table V.

TABLE V  
NUMBER OF PACKETS SUCCESSFULLY TRANSFERRED TO THE MULE IN HIGH MOBILITY CONDITIONS ( $D_y = 15m$ )

speed	avg	min	max
20 km/h	119	88	141
30 km/h	108	89	143
40 km/h	61	48	79

We investigated the effect of the duty cycle in the high-mobility scenario as well. Figures 8-10 show the (average) number of packets the static node is able to transfer, running in several duty cycle modes, when the MULE's speed is 20, 30 and 40 km/h, respectively. We can observe that, even when the duty cycle is 1% and the MULE's speed is 20 km/h (40 km/h), the static node is still able to transfer correctly 10 (5) packets, at a distance  $D_y$  of 15 m. This should be enough in all cases where the static sensor has a small amount of data to transfer. Moreover, several optimizations could be introduced to our simple communication scheme to increase the number of packets successfully transferred to the MULE in a high mobility scenario. In our experiments, for simplicity, the duty cycle remains unchanged (e.g., at 1%) even when the MULE is in the proximity of the static node, i.e., during the communication phase. However, in a real system, the static node should normally operate at a low duty cycle (e.g., 1%), and switch to a higher duty cycle as soon as it discovers the MULE in its proximity. This would increase the amount of data that the static node is able to transfer correctly to the MULE without a significant increase in the energy consumption.

## VI. SUMMARY AND CONCLUSIONS

In this paper we have discussed the results of an extensive experimental analysis on the performance of the MULE architecture in a real dynamic urban environment. Our testbed

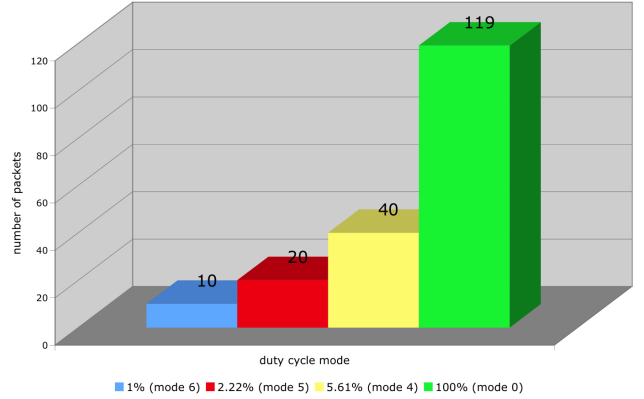


Fig. 8. Average number of packets successfully transferred to the MULE in the high mobility scenario (20 km/h,  $D_y=15m$ ) for different duty cycles

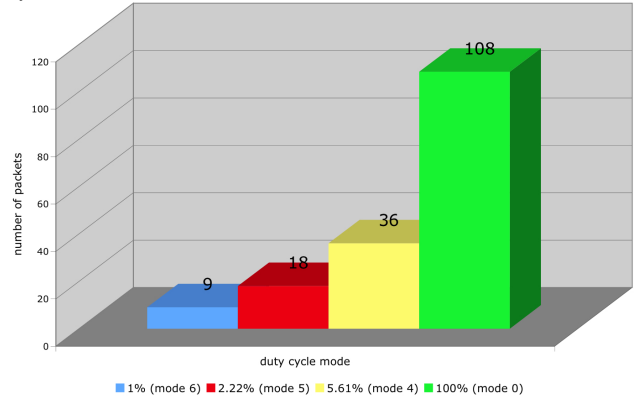


Fig. 9. Average number of packets successfully transferred to the MULE in the high mobility scenario (30 km/h,  $D_y=15m$ ) for different duty cycles

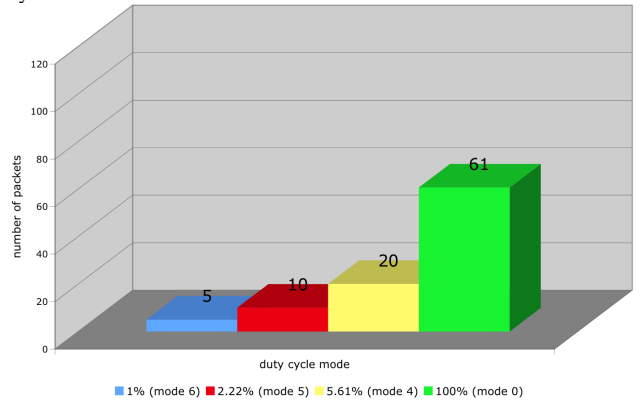


Fig. 10. Average number of packets successfully transferred to the MULE in the high mobility scenario (40 km/h,  $D_y=15m$ ) for different duty cycles

consisted of a mobile sensor node (acting as the MULE) collecting data packets from a static sensor node. In the real world the static node may be a sensor placed at a bus stop, along a street, in a park, or anywhere, sensing a physical phenomenon like temperature, air pollution, etc. On the other hand, the mobile node could be a person walking, a bus, etc. In such a scenario it is important to measure the contact time (i.e., the time interval during which the nodes are able to communicate with a sufficiently good quality), and the amount of data the static node is able to successfully transfer to the MULE during the contact. We have investigated the effect on the above indices of different parameters, like MULE's speed, distance between nodes and duty cycle. We have considered both a low mobility scenario where the MULE's speed is in the range [1, 2] km/h, and a high mobility scenario where the MULE is supposed to move at a speed of 20-40 km/h.

We have found that in the low-mobility scenario the speed of the MULE is not a big issue as the contact time is usually very large. Instead, the distance of the MULE's line of motion from the static node may severely impact the packet loss experienced in the communication, thus reducing the average number of packets the static node is able to successfully transfer to the MULE. This number is further reduced if the static nodes uses a duty cycle to save energy. However, even with a duty cycle of 1%, we found that, on average, the static node was able to transfer correctly 35 packets at a distance of 35 m. In the high-mobility scenario, the increased speed (20 ÷ 40 km/h) limits the maximum (perpendicular) distance to 15 m. However, we found that even when the duty cycle is 1% the static node is still able to transfer correctly 10 (5) packets to the MULE moving at a speed of 20 km/h (40 km/h). This should be enough in all cases where the static sensor node has a small amount of data to transfer. In addition, we discussed in the paper an optimization that could be introduced to our simple communication scheme to increase the number of packets successfully transferred to the MULE in a high mobility scenario without increasing significantly the energy consumption.

Based on the above results we can conclude that the MULE architecture is really suitable for a rich set of environmental monitoring applications in urban environments, both in low and high mobility scenarios. Our analysis was based on Mica2 Berkeley motes. We can reasonably expect that with future advancements in sensor networks technology, it will be possible to achieve even better results than those presented in this paper.

#### ACKNOWLEDGMENTS

This work has been carried out under the financial support of the FET-IST HAGGLE project and the Italian Ministry for Education and Research (MIUR) in the framework of the FIRB-PERF and FIRB-VICOM projects.

#### REFERENCES

- [1] D. Estrin, D. Culler, K. Pister, and G. Sukhatme, "Connecting the Physical World with Pervasive Networks," *IEEE Pervasive Computing*, vol. 1, issue 1, 2002.
- [2] O. Dousse, P. Thiran, and M. Hasler, "Connectivity in Ad-hoc and Hybrid Networks," in *Proc. IEEE Infocom*, (New York, NY, USA), 2002.
- [3] R. C. Shah, S. Roy, S. Jain, and W. Brunette, "Data MULEs: Modeling and Analysis of a Three-tier Architecture for Sparse Sensor Networks," *Elsevier Ad Hoc Networks Journal*, vol. 1, issues 2-3, 2003.
- [4] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," in *Proc. Second ACM Conference on Embedded Networked Sensor Systems (SenSys)*, (Baltimore, MD, USA), 2004.
- [5] S. Jain, R. C. Shah, G. Borriello, W. Brunette, and S. Roy, "Exploiting Mobility for Energy Efficient Data Collection in Sensor Networks," in *Proc. Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, (Cambridge, UK), 2004.
- [6] R. Gray, D. Kotz, C. Newport, N. Dubrovsky, A. Fiske, J. Liu, C. Mason, and S. McGrath, "Outdoor Experimental Comparison of Four Ad Hoc Routing Algorithms," in *Proc. ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile System*, (Venice, Italy), 2004.
- [7] H. Lundgren, E. Nordström, and C. Tschudin, "The Gray Zone Problem in IEEE 802.11b based Ad hoc Networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 6, issue 3, 2002.
- [8] M. Takai, J. Martin, and R. Bagrodia, "Effects of Wireless Physical Layer Modeling in Mobile Ad Hoc Networks," in *Proc. of the 2001 ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, (Long Beach, CA, USA), 2001.
- [9] G. Zhou, T. He, S. Krishnamurthy, and J. Stankovic, "Impact of Radio Irregularity on Wireless Sensor Networks," tech. rep., University of Virginia, 2004.
- [10] G. Anastasi, M. Conti, A. Falchi, E. Gregori, and A. Passarella, "Performance Measurements of Motes Sensor Networks," in *Proc. Seventh ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, (Venice, Italy), 2004.
- [11] D. Jea, A. Somasundara, and M. Srivastava, "Multiple Controlled Mobile Elements (Data Mules) for Data Collection in Sensor Networks," in *Proc. IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, (Marina del Rey, CA, USA), 2005.
- [12] A. Chakrabarti, A. Sabharwal, and B. Aazhang, "Using Predictable Observer Mobility for Power Efficient Design of Sensor Networks," in *Proc. International Workshop on Information Processing in Sensor Networks (IPSN)*, (Palo Alto, CA, USA), 2003.
- [13] A. Kansal, A. Somasundara, D. Jea, M. Srivastava, and D. Estrin, "Intelligent Fluid Infrastructure for Embedded Networks," in *Proc. ACM International Conference on Mobile Systems, Applications, and Services (MobiSys)*, (Boston, MA, USA), 2004.
- [14] W. Zhao, M. Ammar, and E. Zegura, "A Message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks," in *Proc. ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, (Tokyo, Japan), 2004.
- [15] Y. Gu, D. Bozdağ, E. Ekici, F. Özgüner, and C. Lee, "Partitioning Based Mobile Element Scheduling in Wireless Sensor Networks," in *Proc. Second Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON)*, (Santa Clara, CA, USA), 2005.
- [16] A. Somasundara, A. Ramamoorthy, and M. Srivastava, "Mobile Element Scheduling for Efficient Data Collection in Wireless Sensor Networks with Dynamic Deadlines," in *Proc. 25th IEEE International Real-Time Systems Symposium (RTSS)*, (Lisbon, Portugal), 2004.
- [17] M. Ho and K. Fall, "Poster: Delay Tolerant Networking for Sensor Networks," in *Proc. First IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON)*, (Santa Clara, CA, USA), 2004.
- [18] Crossbow Technology Inc., *TinyOS Getting Started Guide*, 2003.
- [19] D. Gay, P. Levis, R. V. Behren, M. Welsh, E. Brewer, and D. Culler, "The NesC Language: A Holistic Approach to Networked Embedded Systems," in *Proc. ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation*, (San Diego, CA, USA), 2003.
- [20] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications," in *Proc. First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, (Los Angeles, CA, USA), 2003.
- [21] S. Ping, "Delay Measurement Time Synchronization for Wireless Sensor Networks," tech. rep., Intel Research, 2003.