MobiLab: A Testbed for Evaluating Mobility Management Protocols in WSN

Jianjun Wen, Zeeshan Ansar, and Waltenegus Dargie

Technical University of Dresden, 01062 Dresden, Germany
{jianjun.wen,zeeshan.ansar,waltenegus.dargie}@tu-dresden.de

Abstract. Wireless sensor networks that support the mobility of nodes are finding applications in different areas such as healthcare, elderly care, and rehabilitation from total knee and hip replacement. However, these application areas also require reliable and high throughput networks. Considering the high fluctuation of link quality during mobility, protocols supporting mobile wireless sensor nodes should be rigorously tested to ensure that they produce predictable outcomes. In this paper we present a wireless sensor network testbed for carrying out repeated and reproducible experiments, independent of the application or protocol types which should be tested. The testbed consists of, among others, a server side control station and a client side traffic flow controller which coordinate inter- and intra-experiment activities. We fully implemented the testbed for the TinyOS and TelosB platforms. We employed Diddyborg robots for emulating different types of movement in indoor and outdoor environments. The paper includes also an extensive evaluation of the testbed.

Key words: Testbed, mobility management, experiment management, wireless sensor networks

1 Introduction

Wireless sensor networks which support the mobility of nodes are useful for different applications. For example, in the healthcare domain, they have been proposed to monitor patients with Parkinson Disease[9], gastroparesis [6], and asthma [8]. As a result, there is an endeavour to integrate medical devices and make them interact with existing wireless sensor platforms. For instance, the wireless mobility capsule integrating pH, pressure, and temperature sensors for the diagnosis of gastroparesis has officially been approved by the US drug and food administration since 2006; it has produced promising results and may replace existing invasive and painful procedures (such as endoscopy) [4]. Similarly, there are commercially available wireless electrocardiograms which can be integrated with existing sensor platforms.

There are, however, some challenges associated with mobility, one of the most significant challenges being the difficulty of maintaining link quality during mobility. Independent experiments show that link quality quickly deteriorates

when nodes are mobile while communicating, resulting in high packet loss, drift, and jitter. This aspect particularly affects applications which require relatively high throughput. Devices such as wireless electrocardiograms typically generate data at tens of kilobits per second rate. While this in itself may not be high, if other sensors such as 3D accelerometers and gyroscopes have to be sampled at comparatively the same rate, then the aggregate data rate from a single node can be high. Whereas the effect of mobility on link quality fluctuation has been extensively studied in the context of cellular communications (owing, luckily, to the ability of collecting ample statistics from a large number of users in different settings and locations), investigation of link quality fluctuation in mobile wireless sensor networks is a work in progress.

In this paper we propose a testbed for evaluating the effect of mobility in wireless sensor networks. The testbed separates the concern of application development from the evaluation of the application in different mobile scenarios. By doing so, complex and reproducible experiments can be carried out to ensure that the behaviors of applications are both reproducible and predictable. We fully implemented the testbed for the TinyOS and TelosB platforms. Our mobile nodes are carried by Diddyborg robots[1], each of which is controlled by 6 powerful gear motors, so that the robots can be tasked to emulate different types of movements in indoor as well as outdoor environments.

The remaining part of this paper is organized as follows: In Section 2, we review related work and position our own work. In Section 3, we present the system architecture of our testbed. In Section 4, the performance of MobiLab is evaluated from different perspectives. Finally, in Section 5, we give concluding remarks and outline future work.

2 Related work

Testbeds are intended to efficiently test wireless sensor networks before they are actually deployed in real-world environments. Compared to the area or volume an actual deployment occupies, testbeds are considerably compact, so that they can be installed in labs or in areas which are easily accessible. This means, some communication parameters are intentionally scaled and events can be deliberately injected into the network to suit the test setting and to emulate actual events. There are online testbeds which are available to the WSN research community, most of them establishing two types of networks. One of the networks is the actual wireless link the characteristic of which is investigated and the other network serves as a backbone, reliable network for collecting performance indicator metrics. This network can be wireless (for example, a WLAN) or wired (using USB hubs or serial interfaces). As far as the software architecture is concerned, the existing testbeds also share similar aspects such as: (a) provision of web-based infrastructure and experiment management services; and (b) functionalities for dynamic reprogramming, specification, configuration, and execution

[1] https://www.piborg.org/diddyborg

of experiments. Some of the testbeds employ robots [5, 7, 2] while others employ toy trains [10] as mobile platforms, to which wireless sensor nodes are attached. Besides providing mobility, the mobile platforms also serve as power suppliers and node managers, through which new program images can be installed and experiment procedures are controlled and managed.

Emulab [5] is perhaps the first publicly reachable mobility-enabled testbed for WSNs experimentation. The testbed is deployed in an L-shaped area and consists of (1) 25 Mica2 static nodes installed on the walls and ceiling of a building to form a grid-like topology, (2) 6 mobile nodes attached to robotic platforms, which can perform user-specific and accurate way-point walking models (according to the authors, the position of the robots can be determined within 1 cm error, the worst-case), (3) 6 cameras which are installed on the ceiling to track the robots, and (4) additional 3 web-cams to provide live-monitoring. One of the limitations of the testbed is the difficulty of influencing the movements of the robots during experiment execution, because their movement pattern is predetermined and is not accessible at runtime.

Kansei [3] is a testbed employing the same types of robots like Emulab to support mobility, but it does not provide any positioning system. The testbed uses five robots integrating TMote Sky nodes and Extreme Scale Mote (XSM). These robots are deployed on top of a Plexiglas plane in which 210 XSMs and TMote Sky nodes are arranged in a 15×14 grid bench-work. In addition to the common functionalities the previous testbed provides, Kansei provides a mechanism to inject events into individual nodes and gateways. Sensei-UU [7] employs a Lego NXT robot as the mobile platform, on which a TelosB node and a smartphone are attached. Its unique feature is employing WL-500GP wireless access point as a control station to provide programming, experiment monitoring, and data logging functionality via a wireless channel. While it is relatively easy to reproduce and repeat experiments with this testbed, it has some drawbacks: (1) the robot requires the installation of tapes on the floor, which limits the types of movement that can be emitated by the mobile platform (i.e., undertaking different random movements is difficult); and 2) it is difficult to support multiple mobile nodes at the same time.

SensLAB [2] and TrainSense [10] are two recently proposed testbeds for mobile platforms. Both utilize toy trains as mobile platforms. Since the trains run on tracks, which physically limit their motion, the testbeds are difficult to extend. It is also difficult to introduce random walks into experiments. One of the merits of these testbeds is their ability to provide better accuracy of localization and control of mobility compared with the other mobile platforms.

3 Architecture of MobiLab

The main purpose and, therefore, contribution of our testbed is the flexible but reproducible execution of complex experiments with wireless sensor networks in which some of the nodes are mobile. The testbed enables users to upload their own program image onto individual nodes and to specify experiment procedures

and the movement pattern of mobile robots independent of the types of applications the wireless sensor networks are supporting. To achieve these goals, our testbed separates resource management into different concerns.

3.1 HardwareArchitecture

The hardware architecture consists of four modules: a control station, a wireless sensor network, a node manager, and a backbone wireless channel. The control station serves as the main interface between the user and the testbed. A group of dedicated software services run in the control station to manage the testbed resources and to control experiments. In the next subsection we provide a detail description of the software architecture of the control station.

The wireless sensor network consists of three types of nodes: static relay nodes, mobile nodes, and sniffer nodes. The sniffer nodes are special stationary nodes used for monitoring the state of the wireless channel to obtain complementary information about experiment execution during debugging. By changing the firmware, it can also serve as interference source. A node manager interfaces a node with the control station. Each node manager (for our implementation we use the raspberry Pi 2) is connected to a sensor node via a USB port. We use a Wi-Fi *ad hoc* network as our backbone network because of its scalability and flexibility.

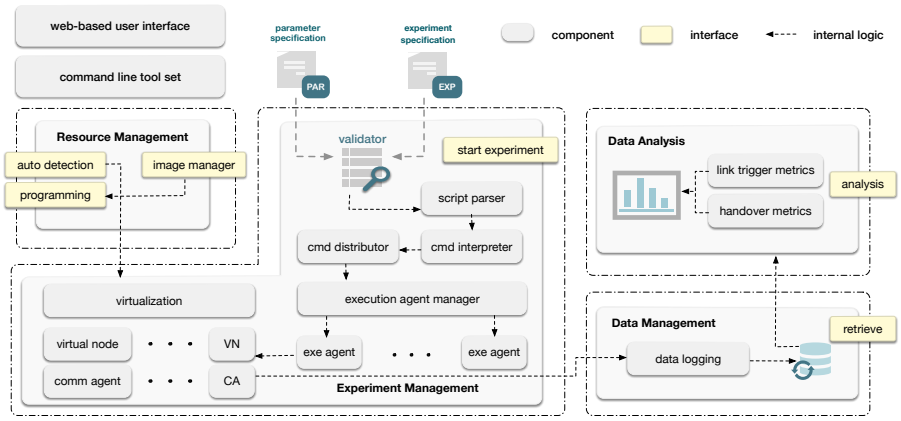

Fig. 1: The software architecture of the control station.

3.2 Software Architecture

The control station is the most important module in MobiLab. It ensures that the testbed as a whole functions as a unified system. It is through the control station every program image or command is propagated to the wireless sensor

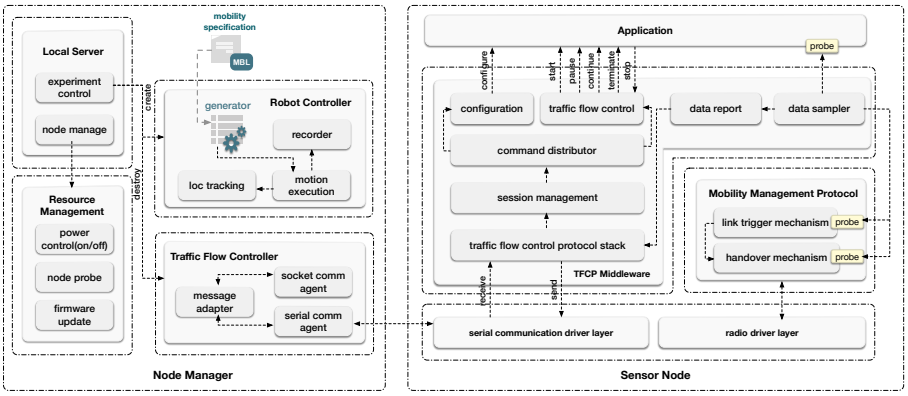

Fig. 2: An overview of the software architecture of the testbed from a single node perspective: (Left) The software architecture of the node manager. (Right) The software architecture of a sensor node.

network. Fig. 1 displays its software architecture, which consists of a user interface, a resource management service, an experiment management service, a data management service, and a data analysis service.

User Interface MobiLab provides both a web-based and a command-line-interface through which users can access the testbed and conduct experiments remotely. Users can browse active nodes and their status, upload program images into the wireless sensor network, and specify and manage experiment procedures using experiment execution primitives we defined (to be discussed below).

Rsource Management MobiLab does not require a fixed infrastructure (a specific network size or topology) to run experiments. The resource management service is responsible for authorizing nodes to join the network and users to access individual nodes; for managing binary images, and for ensuring proper program installation. Moreover, the resource management service uploads and deletes program images to and from nodes and controls versions. In it, a synchronization daemon runs in the background to ensure that program images in the control station and the node managers are consistent.

Experiment Management Service The experiment management service enables users to define and manage inter- and intra-experiment activities. As regards management, users can initiate, interrupt, suspend, modify, and end experiments at runtime by using experiment execution primitives (see Table 1). The primitives enable users to configure interaction (transmission power, channel, partner nodes) and to specify communication durations, among others. When an experiment procedure is submitted to it, the experiment management service validates the procedure to ensure that it is executable, parse the procedure to extract experiment parameters, translates the parameters into binary, creates a

control flow (execution sequence), and passes the control flow to the execution manager. The execution manager is responsible for coordinating the execution of an experiment procedure until it terminates. A virtual node manager within the control station's architecture creates a virtual representation for each physical node. The aim is to hide differences in hardware architecture between nodes from users and to provide common interfaces for accessing and interacting with them.

Table 1: traffic flow control primitives.

primitive	**description**
configure	setup the application dependent parameters
start	initiate the test round
stop	notify finish of test round
pause	suspend execution
continue	resume execution
terminate	stop execution permanently

Data Management and Analysis Data management or logging is one of the useful features of testbed frameworks. When an experiment is launched, MobiLab creates an instance of a data logging module which is then associated with the communication agents of the corresponding virtual nodes. During experiment execution, the physical nodes log the desired data locally and forward them to their virtual node managers at the control station, which then stores the data in a database. Alternatively performance indicators can be directly streamed to virtual node managers as they are generated.

3.3 Node Manager

The software aspect of a node manager has three components, which are the local server, a resource manager, and a traffic flow controller. A robot node manager includes an extra module for managing mobility.

Local Server Its main responsibility is managing the physical node and controlling the proper execution of experiments. The server is logically connected with the resource management service at the control station, thus it is able to provide the functionalities for probing the sensor node, updating firmware; it is also responsible for coordinating experiment control flows and commands pertaining to the motion of a robot.

Traffic Flow Controller The procedure of an experiment is first encoded using the traffic primitives we specified in Table 1. By the time it reaches the traffic flow controller at the node manager, it is translated into a sequence of commands and parameters. The traffic flow controller is responsible for creating

a channel between the node manager and the physical node and for transmitting the commands and parameters in their sequence and appropriate delay to the physical node. It also channels the logged data from the physical node to the node manager. The node managers are time synchronized with the control station at the beginning of each run of an experiment. For a detailed description of the Traffic Control Flow protocol, we refer the reader to [11].

Robot Controller Different mobility models can be implemented and integrated into the node manager a priori and an instance of a model can be loaded when the robot controller is first instantiated. The parameters of this model can be modified at runtime by using the experiment primitives in Table 1. Currently, we are experimenting with straight line walking and the random waypoint model [1].

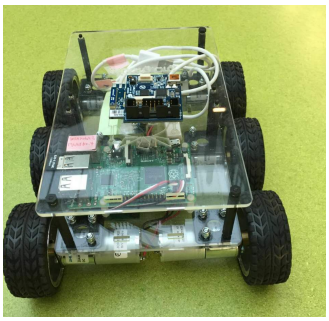

Fig. 3: MobiLab deployment: A MobiLab robot carrying a wireless sensor node.

3.4 Sensor Node

Fig. 2 (right side) illustrates the software architecture of a wireless sensor node. Most relevant to this paper is the traffic flow control protocol middleware (TFCP). The TFCP middleware is an application independent layer for managing inter and intra-experiment activities. It is loosely coupled with the OS layer, interacting with communication drivers by `send` and `receive` interfaces and exposing six interfaces to the higher layers (MAC, network and application layers), so that users can setup experiment and application specific parameters and control the execution steps of experiments. The data sampler and report module locally collects and aggregates performance indicator metrics from relevant layers and communicates them with the control station via the TFCP middleware. The TFCP middleware running on each sensor node is built on top of TinyOS and has a footprint of 1058 bytes of ROM and 84 bytes of RAM.

The mobility management protocol does not belong to the MobilLab testbed. We integrated it to investigate the performance of different mobility management protocols under the same setting.

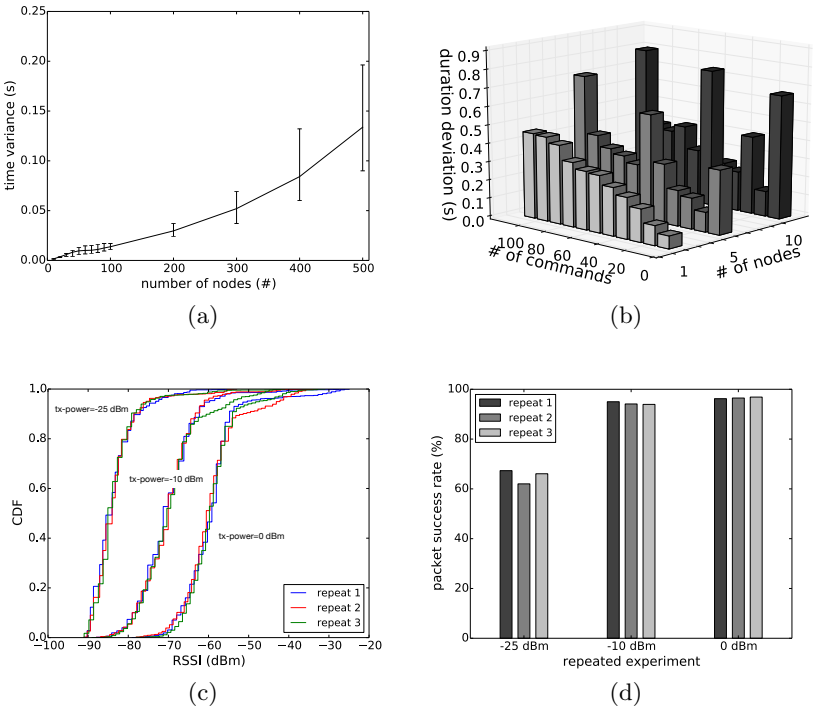

Fig. 4: Performance analysis of MobiLab as an experiment management tool: (a) The time variance of synchronized starting time for networks of different sizes; (b) The deviation in the duration of arbitrary control commands in a single experiment. (c) cdf of RSSI fluctuation of repeated experiments; (d) packet success rate of repeated experiments

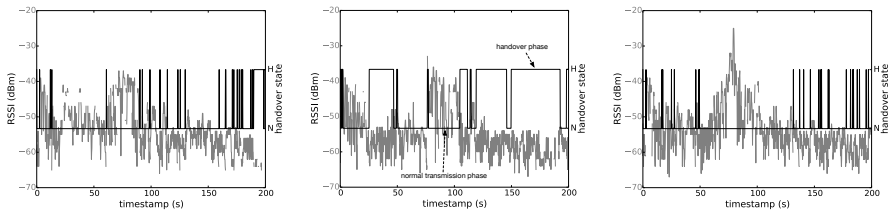

Fig. 5: RSSI fluctuation of incoming acknowledgement packets and handover oscillation. (a) Single threshold scheme. (b) Dual threshold scheme. (c) LMS.

4 Evaluation

One of the features of MobiLab is the simplicity with which it can be deployed in different places, since it does not relay on any fixed infrastructure. It takes

less than 10 minutes to setup the environment and start to perform the experiment. In order to carry out repeatable experiments, the detail of the experiment procedures are scripted, i.e., the beginning, end, and duration of every activity is specified. When the control station dispatches experiment procedures, they may not be executed by the individual nodes at precisely the same time. Consequently, nodes may not begin and end the execution of experiments at the same time. This phenomenon is an aspect of both the size of the network and the complexity of the experiments. To investigate this phenomenon, we launched a set of simple experiments with variable number of physical nodes (from 5 to 20) and emulated nodes (up to 500). We recorded the starting time of each node and calculated the maximum variance (time difference between the earliest starting node and the latest staring node). We observed that the maximum variance of experiment beginning time was 200 ms. Secondly, we inserted arbitrary number of control commands (*pause* and *continue* commands) in the experiments lasting up to 600 seconds, and varied the number of nodes from 1 to 10. We did not observe significant increments of experiment completion times when the number of commands increased (shown in Fig. 4 (b)). To show the repeatability of an experiment using MobiLab, we evaluated the RSSI fluctuation and packet reception rate under different configurations. We conducted a series of experiments and repeated each one three times. As Fig. 4 (c) and (d) shown, the CDF of RSSI values are almost the same for the experiment with the same configuration (transmission power, motion pattern, walking path etc.) and the packet reception rates are consistent for each repetition.

Fig. 5 shows how we evaluated the performance of three handover supporting MAC protocols using our testbed. The protocols evaluated the RSSI fluctuation of incoming ACK packets and determined the appropriate time to change a relay node with which a mobile robot (refer to Fig. 3) should communicate.

5 Conclusion

In this paper we introduced MobiLab, a testbed we developed to experiment with wireless sensor networks supporting mobile nodes. MobiLab separates the concerns of application and protocol developments from their testing phase. Our main motivation was performing repeated and reproducible experiments independent of the types of network topology, communication protocols, communication parameters involved in the experiments. From the software perspective, besides sharing the same design principles with existing testbeds, MobiLab provides several novel contributions such as supporting both inter- and intra-experiment management, TFCP middleware in a sensor node, and a robot motion management. Except for the sensor node architecture, which is implemented in C for the TelosB platform, all the remaining software components are implemented in python, which is relatively easy to port to other platforms.

Acknowledgment

This work is funded by the DFG under project agreement: DA 1211/5-2.

References

1. Camp, T., Boleng, J., Davies, V.: A survey of mobility models for ad hoc network research. Wireless communications and mobile computing 2(5), 483–502 (2002)
2. Des Rosiers, C.B., Chelius, G., Fleury, E., Fraboulet, A., Gallais, A., Mitton, N., Noël, T.: Senslab very large scale open wireless sensor network testbed. In: Proc. 7th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCOM) (2011)
3. Ertin, E., Arora, A., Ramnath, R., Naik, V., Bapat, S., Kulathumani, V., Sridharan, M., Zhang, H., Cao, H., Nesterenko, M.: Kansei: a testbed for sensing at scale. In: Proceedings of the 5th international conference on Information processing in sensor networks. pp. 399–406. ACM (2006)
4. Farmer, A.D., Scott, S.M., Hobson, A.R.: Gastrointestinal motility revisited: the wireless motility capsule. United European gastroenterology journal p. 2050640613510161 (2013)
5. Fish, R., Flickinger, M., Lepreau, J.: Mobile emulab: A robotic wireless and sensor network testbed. In: IEEE INFOCOM (2006)
6. Grimes, C.A., Ong, K.G., Varghese, O.K., Yang, X., Mor, G., Paulose, M., Dickey, E.C., Ruan, C., Pishko, M.V., Kendig, J.W., et al.: A sentinel sensor network for hydrogen sensing. Sensors 3(3), 69–82 (2003)
7. Rensfelt, O., Hermans, F., Larzon, L.Å., Gunningberg, P.: Sensei-uu: A relocatable sensor network testbed. In: Proceedings of the fifth ACM international workshop on Wireless network testbeds, experimental evaluation and characterization. pp. 63–70. ACM (2010)
8. Seto, E.Y., Giani, A., Shia, V., Wang, C., Yan, P., Yang, A.Y., Jerrett, M., Bajcsy, R.: A wireless body sensor network for the prevention and management of asthma. In: Industrial Embedded Systems, 2009. SIES'09. IEEE International Symposium on. pp. 120–123. IEEE (2009)
9. Sha, D.L., Xie, W.C., Fan, X.L., Li, Y.: Based on wireless sensor network (nwk) of non-contact tremor monitoring equipment improvement for parkinson's disease. In: Applied Mechanics and Materials. vol. 713, pp. 491–494. Trans Tech Publ (2015)
10. Smeets, H., Shih, C.Y., Zuniga, M., Hagemeier, T., Marrón, P.J.: Trainsense: a novel infrastructure to support mobility in wireless sensor networks. In: Wireless Sensor Networks, pp. 18–33. Springer (2013)
11. Wen, J., Ansar, Z., Dargie, W.: A system architecture for managing complex experiments in wireless sensor networks. In: The 25th International Conference on Computer Communication and Networks (ICCCN 2016), August 1-4, 2016, Waikoloa, Hawaii, USA (2016)