SWiFiNet: a real field WSN testbed

A.W. Rohankar*
Department of IT,
Sinhgad College of Engineering,
Pune University,
Pune 411 041, India
E-mail: atulrohankar@gmail.com
*Corresponding author

Mrinal Kanti Naskar
Department of ETCE,
Jadavpur University,
Kolkata 700 032, India
E-mail: mrinalnaskar@yahoo.co.in

Amitava Mukherjee
IBM India Pvt. Ltd.,
Salt Lake, Kolkata 700 091, India
E-mail: amitava.mukherjee@in.ibm.com

Abstract: Proliferation of technologies and services related with wireless sensor network has made a great impact on all aspects of life. Research and development in this domain naturally calls for real-time WSN testbeds. Present day WSN testbeds are expensive and complex in operation and simulators are unable to justify real-time field interaction. Design of SWiFiNet completely eliminates expensive back channel infrastructure for reprogramming, communication data logging and setting of test experimental setup control parameters. This has provided true real-time portability and testability of the network in field with around 90% reduction in testbed infrastructural setup cost. SWiFiNet is hardware and system software independent testbed. SWiFiNet permits time synchronised data recording, capturing, processing and repeatability of an experiment. Extensive experiment parameter configuration, wide scalability of the network, in-node status monitoring are significant features of SWiFiNet.

Keywords: WSN testbed; network architecture; real field testing; communication data logging; back channel.

Biographical notes: A.W. Rohankar is currently pursuing his PhD at Jadavpur University. He completed his ME in Electronics and Telecommunications from Government College of Engineering Pune in 1999, and BE in Instrumentation and Control from Government College of Engineering, Pune in 1984. His research interest includes embedded system design and development, embedded device network and wireless sensor network systems.

Mrinal Kanti Naskar received his BTech and MTech from IIT Kharagpur, and PhD from Jadavpur University, India. He is a Professor in the Department of Electronics and Tele-Communication Engineering, Jadavpur University, Kolkata, India. His research interests include ad-hoc networks, optical networks, wireless sensor networks, wireless and mobile networks and embedded systems. He is author/co-author of several published articles in WDM optical networking field that include ‘Genetic evolutionary approach for static traffic grooming to SONET over WDM optical networks’ [Computer Communication, Elsevier, 2007], and ‘Genetic evolutionary algorithm for optimal allocation of wavelength converters in WDM optical networks’ [Photonic Network Communications, 2008].

Amitava Mukherjee has done his PhD from Jadavpur University and is working as a Senior Manager of IBM GBS India Pvt Ltd. His consulting experiences include global delivery, IT strategy consulting and implementation in SAP and custom software, software quality management and infrastructure management, resource allocation and hiring, and corporate/internal training. He has over 120 published papers in journals and conference proceedings of international repute (like IEEE, ACM, etc.) and five books in wireless communication and societal engineering. He has collaboration with different research labs/institutes across the continents.

1 Introduction

Progress in wireless sensor network in last couple of years is astonishing. There seems to be a lot of diversity in WSN applications tried and so is reflected in testbeds for WSN experimentation (Bello et al., 2007). However, many of the testbeds developed are either application specific (Albesa et al., 2007; Zhang et al., 2004; Ting Liu et al., 2004) or routing protocol specific (Etxaniz and Aranguren, 2008; Werner-Allen et al., 2005; Sheu et al., 2008; Dimitriou et al., 2007) or single parameter characterisation specific (Baccour et al., 2010; Luo et al., 2011). But to study sensor networks behaviour and performance by means of real-time deployment or setting up a testbed is complex and requires much effort and financial resources. Various network simulators therefore have been developed or extended to simulate sensor networks (Khemapech et al., 2008, Yick et al., 2008). Existing widely-used network simulators such as ns-2 (Mccanne and Floyd, 1997) have been extended to simulate sensor networks. Application specific simulators for sensor networks have also been developed (Levis et al., 2003; Polley et al., 2004). These simulators with all their key features like, radio model, CPU, memory features, routing capabilities, etc. which make them promising but can be used to obtain preliminary results only and real-time field testing still remains imperative for realistic and usable results (Bello et al., 2007; Hackman, 2010; Akyildiz and Chowdhury, 2008).
Swift WiFi network-testbed (SWiFiNet) design draws its inspiration from difficulties and bottlenecks experienced by researchers while using testbeds and simulators. SWiFiNet is designed with objective of combining simulator environment features like configurability, comprehensive result collection, simple user interface, ease of operations and testbed features like hands on experience of real-time wireless communication, data generation and field testing. User has been distanced from nitty-gritty of hardware programming and embedded system handling giving much ease and confidence in carrying out the experimentation and getting feel of real-time behaviour of wireless communication. SWiFiNet design is aimed towards simple, user friendly and single station monitored testbed.

Paper is divided into six sections; out of which Section 1 starts with introduction. Section 2 discusses present scenario of real-time testbeds and evaluation of their infrastructure requirements, design focus. This in turn calls for addressing the points to be focused in designing testbed architecture as discussed in Section 2.2. Section 3 is about SWiFiNet testbed architecture including firmware and software and testbed GUI explanation. Section 4 evaluates SWiFiNet design features and their fulfilment of the challenges outlined in Section 2.2. Section 5 presents a laboratory experiment with discussion and performance evaluation. Section 6 concludes and scope for further research and enhancements.

2 WSN testbeds

There are many WSN testbeds developed for WSN research purpose. Categorically they can be classified as indoor and outdoor testbeds. The infrastructural difference is wired back channel and wireless back channel deployment (Bello et al., 2007; Rensfelt et al., 2009). The wired or wireless back channel support is used for reprogramming, reconfiguration and communication data logging.

General architecture of the present day testbeds invariably follow web-based distributed architecture involving complex infrastructure for deployment. All these testbeds have common architecture viz. a server running main control application of testbed activities, data base storage, and programming resources. Integrated testbed application is executed on main server in client server model. This server is connected through back channel to nodes via expensive network devices such as micro stations, switches, hubs. These network devices are in turn interfaced to specially designed hardware and software configured sensor nodes via USB, RS232, Ethernet buses. Some sensor nodes make use of RTOS like RTLINUX or TinyOS. A testbed, at Washington University, WSSUSTIL is claimed to be largest WSN testbed which has main server with client terminals for user access of the testbed (Hackman, 2010). Server is connected to micro-servers through LAN deployed in different laboratories, offices, departments, floors. These micro-servers are wired to nodes fixed at various locations through USB, RS232 or Ethernet switches. A testbed, TWIST uses server control unit connected on Ethernet backbone coupled with super nodes. These super-nodes are in turn connected to WSN nodes through USB bus. Power control feature of USB bus gives binary power control of node power when powered through USB bus along with reprogramming and configuration control (Handziski et al., 2006). WSTB uses main server connected to the gateways having USB, RS232 buses to connect WSN nodes based on ATMEL processor.
(octopus I) and MSP430 processors (octopus II). These nodes use TinyOS real-time operating system (Ho and Le-Ngoc, 2010). Mote lab (Werner-Allen et al., 2005) another web-based testbed uses permanently deployed nodes. Network back channel is created through MIB600 PDA having Ethernet interface to connect the main server through local area network switches. These PDA are interfaced with mica2, micaZ sensor nodes through USB. These nodes are TinyOS-based nodes. Users are provided with motelab nodes to develop TinyOS-based code and then use the testbed to reprogramme deployed nodes. KonTest testbed (Iwanicki et al., 2008) developed in Vrije University Amsterdam, uses two tier structure of desktop machines and USB hubs connected to TelosB class nodes from Crossbow corporation. Wired back channel infrastructure is used to programme and gather information from the nodes. Another testbed developed makes use of TI’s second generation ZigBee/IEEE 802.15.4 compliant system-on-chip development boards to build three tiers testbed architecture, namely gateway, router and end-device (Ho and Le-Ngoc, 2010). Gateway is connected to PC through Ethernet or USB bus and uses wireless 802.11b bus as back channel (Bello et al., 2007; Rensfelt et al., 2010; Dimitriou et al., 2007). In an education testbed developed. 802.11b gateways are used for connecting sensors while main server, application server and data base servers uses Ethernet connectivity (Bello et al., 2007), Sensei also uses 802.11b enabled devices to control and communication data logging of the sensor nodes (Rensfelt et al., 2009).

Blue tooth-based testbed (Etxaniz and Aranguren, 2008) uses Bluetooth devices for network protocol development severely limited by use of base Bluetooth protocol. Real net is application-based testbed used for environment parameter data collection which emulates actual WSN deployment, with limited configuration capability (Albesa et al., 2007).

2.1 Complexity of architecture

All these designs follow a network-based testbed design route, resulting in bulky and complex designs. The testbed designs explored above has a focus of web-based infrastructure rich design providing comprehensive features to end users. Many of them due to wired or with wireless 802.11b access point backchannel; give static topology for testbed nodes. Certainly they provide advantages of easier reprogramming and greater control due to high end devices used for building the infrastructure. All testbed end nodes use highly sophisticated hardware and software firmware for end node. As elaborated above MIB600 PDA, TinyOS, Linux-based nodes, deviate user from the ground reality that WSN node has to have a small form factor in terms of size, cost and availability of COTS component. It also naturally follows that the experimentation based on such rich infrastructure may not be useful in real life scenarios.

Simplicity of laboratory operations and hands on operation on actual devices is missing. Overall operations of testbeds have become complex and tedious resulting in making the learning bit difficult for the user. The user has to learn operations of developed testbed system and embedded node programming even for performing simple experiments.

The testbed therefore needs to be simple GUI-based solution with portable and configurable nodes for various testing parameters. Provision of in-built network layer reduces frequent programming of the nodes for various experimental setups. In majority of cases this can be substituted by providing maximum configurable parameters which can be commanded through main station GUI would ease user from complex learning.
curve of using the testbed system and embedded system programming. Programming required for testing network protocols can be made available at master logging stations or providing ‘on the fly’ programming.

3 SWiFiNet testbed architecture

The SWiFiNet testbed architecture has three main components. Desktop machine connected to main master (referred as Master) emulating master sink node in real WSN application. Router or range extender [alternately referred as Network Processing Device (NPD)] for communicating nodes beyond the range of master which can be cascaded to achieve larger distances, and WSN nodes [referred as End Device (ED)].

3.1 Node firmware

SWiFi node uses self designed hardware and firmware using off the shelf components available in the market. Following section describes hardware and software modules used to build the testbed nodes. Design is not specific to the hardware neither the software is specific to a processor used. The software is completely portable on any module with 802.15.4 functionality.

SWiFiNet node hardware schematic is shown in Figure 1. A 32 bit CPU integrated with trans-receiver JN5139 is selected as a single chip solution to design Master, NPD, and ED node firmware. This microcontroller, suitable for applications, is IEEE 802.15.4 compliant. The microcontroller has integrated peripherals of SPI bus, I2C bus, two UART ports, 12 bit ADC, 11 bit DAC, five timer/counters and digital I/O Port lines. I2C-based sensor chips are interfaced for sensing temperature, humidity and tilt measurement. MMC memory card is provided for logging local data communication and other events. This card is interfaced through SPI bus operating at 1 Mbps. This rate is sufficiently more than radio communication rate that is 250 Kbps to record all activities without interfering with ongoing communication. Real-time clock (RTC) is included to synchronise timing stamp of logged communication packets along with date.

Figure 1  Hardware schematic for testbed devices (see online version for colours)

Backbone of the testbed is a proprietary task distributed master centric network layer which works transparently providing reliable and robust connectivity within the network. The connectivity is provided between master sink node and end node only. Peer to peer connectivity is not considered. A range extender or routing device is designed for
connecting nodes not in the range of master sink node. These routing devices provide seamless connectivity of the nodes. All devices work in two threads, switching alternately between them. First thread is network operation where networking operations are carried out. In this thread, all incoming and outgoing packets are recorded in the MMC memory. To timestamp the activities a 1 ms timer is activated and all messages are recorded with respect to this timer value. Second thread executes application layer commands like sampling selected sensor, preparing packet for destined device, reading incoming packet and executing respective task associated with incoming message.

3.2 Network topology

Sample network topology is shown in Figure 2. Master is attached to desk top PC through serial RS232 interface. The testbed application GUI communicates with master using pre-designed packet format for trans-receiving commands and data exchange. The network is dynamically organised and is centrally controlled by the master. Master can directly connect maximum 31 devices including routing devices. Routers can join maximum 31 devices including neighbour routers and end nodes. Overall presently maximum network capacity is of 255 devices. All nodes are commanded by master under control of testbed application GUI.

**Figure 2** Sample network topology for SWiFiNet network testbed (see online version for colours)

3.3 SWiFiNet GUI

SWiFiNet GUI is an interface between master sink node and PC. GUI communicates with master through serial port through pre-designed packet format containing command and associated data. The master executes testbed commands on behalf of testbed application. SWiFiNet GUI allows user to configure and control testbed environment.
**Figure 3**  Base parameter configuration and reading network (see online version for colours)

**Table 1**  Parameter configuration and operation commands (shown in GUI Figure 3)

<table>
<thead>
<tr>
<th>Sr. no.</th>
<th>Description/command</th>
<th>Significance/operation</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Power</td>
<td>Basic setting of power in steps of 6 dB</td>
<td>Settings effected after reset of the device</td>
</tr>
<tr>
<td>2</td>
<td>LQI</td>
<td>Minimum LQI value for joining within network</td>
<td>Defines robustness of link quality of joining device</td>
</tr>
<tr>
<td>3</td>
<td>Tries</td>
<td>Maximum number of tries for getting ACK response</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Reset network</td>
<td>Master broadcast reset command, All NPD re-broadcast the command</td>
<td>Resets whole network</td>
</tr>
<tr>
<td>5</td>
<td>Get NPD list</td>
<td>Master gives list of NPDs attached</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Get ED list</td>
<td>Master gives list of EDs attached</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Ping NPD</td>
<td>A 32 byte packet is Pinged to selected NPD result is in round trip time in ms</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Ping ED</td>
<td>A 32 byte packet is Pinged to selected ED result is in round trip time in ms</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Save file</td>
<td>Saves the text in accompanying text box in notepad file</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Clear text box</td>
<td>Clear the text in accompanying text box</td>
<td></td>
</tr>
</tbody>
</table>
Figure 3 shows GUI snapshot for base parameter configuration and reading network topology from master. Device base parameters includes power settings in range of 0–5 corresponding to –30 dBm–0 dBm power setting of module with step increment of 6 dBm. LQI value setting is used for network device JOIN metrics. If joining device has LQI value more than set value then it will join else JOIN request will be discarded. Tries defines max_retry parameter to be set for each message when ack is expected. Table 1 summarises entire base configuration and network topology commands for testbed GUI shown in Figure 3. The command set allows user to reset network, get list of attached end nodes, routers and attached end nodes to routers. It also allows to Ping specific end node and router to verify link capability.

Setting and performing experiments with SWiFiNet testbed is a simple procedure. GUI snap shot for experimental parameter configuration and other commands to read experiment results is shown in Figure 4.

The SWiFiNet allows each device or set of devices to be configured separately. Packet throughput rate, packet load, Ack type and Packet type are set prior to conduction of the experiment. The preset configuration can also be read back through ‘Get Config’ command for specific device. Experiment can be started by issuing ‘Expt. start’ command. All the devices receiving this command reset the timer clock. Reconfiguration is not allowed while the nodes are in experimentation mode. The experimentation mode can be aborted by issuing ‘abort Expt’ command. Table 2 shows all parameters and commands used for conduction of experiment as show in Figure 4 GUI snapshot.

Figure 5 shows analysis of log files received from master, end devices and NPDs. Standard equations for calculating result parameters are used. A detailed result file is created in text format giving more detailed analysis of the data received.

Figure 4  Experiment setup, associated commands and data for testbed (see online version for colours)
Table 2  Parameter configuration and operation commands (shown in GUI Figure 4)

<table>
<thead>
<tr>
<th>Sr. no.</th>
<th>Command</th>
<th>Operation</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ED address</td>
<td>Specific end device address</td>
<td>Configures selected ED</td>
</tr>
<tr>
<td>2</td>
<td>Master address</td>
<td>Fixed</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Interval time</td>
<td>Packet interval time in ms</td>
<td>ED generate packet with interval time</td>
</tr>
<tr>
<td>4</td>
<td>Duration</td>
<td>Experiment duration in seconds</td>
<td>ED will continue to generate packets till this duration</td>
</tr>
<tr>
<td>5</td>
<td>Packet size</td>
<td>Packet payload in bytes</td>
<td>Maximum 100 bytes.</td>
</tr>
<tr>
<td>6</td>
<td>ACK value</td>
<td>N, H, A types</td>
<td>N: NOACK</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>H: HARDWARE ACK expected</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A: SOFTACK expected from master.</td>
</tr>
<tr>
<td>7</td>
<td>Packet type</td>
<td>S, R types</td>
<td>S: sequential</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>R: random in given packet interval</td>
</tr>
<tr>
<td>8</td>
<td>Start</td>
<td>L, I start</td>
<td>L: start generating packets after master issues start command. I: start generating packets after joining the network.</td>
</tr>
<tr>
<td>9</td>
<td>Start exp.</td>
<td>Master start command</td>
<td>All EDs start generating packets</td>
</tr>
<tr>
<td>10</td>
<td>Abort exp.</td>
<td>Master abort command</td>
<td>Aborting the experiment causing all EDs to stop generating packets</td>
</tr>
<tr>
<td>11</td>
<td>Get config. ED</td>
<td>Read configuration of the selected ED</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Get ED file</td>
<td>Read LOG file of the selected ED</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Get NPD file</td>
<td>Read LOG file of the selected NPD</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Get master file</td>
<td>Read LOG file from master</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Save file</td>
<td>Saves the text in accompanying text box in notepad file</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5  Analysing collected data (see online version for colours)
4 SWiFiNet features

Following sections discuss SWiFiNet design features. The requirements and challenges outlined in Sections 2.1 and 2.2 are tried to be addressed in SWiFiNet architecture to quite an extent for fulfilment of the necessities discussed.

4.1 Back channel replacement

The back channel is required for reprogramming, reconfiguring and communication data logging within the testbed. SWiFiNet uses a simple solution for logging/recording of communication by providing large inexpensive memory at each node. These nodes are supported by interfacing inexpensive micro SD memory card of memory size ranging from 512 MB to 8 GB. All incoming and outgoing packets are stored in this memory. 1 GB memory can store up to 480 hrs of data that is 20 days at a rate of 4 packets/second and packet size of 128 bytes with additional bytes for timestamp and other attributes. The testbed GUI provides facility for uploading this communication log file from each node using same wireless link established between node and master. A file transfer protocol is written specially for this task to ensure reliable data transfer.

4.2 Lightweight firmware

SWiFiNet uses off the shelf components for WSN node keeping small form factor as expected with WSN. A micro SD memory card is interfaced through SPI bus operating at 1 Mbps speed. The node logs all incoming and outgoing communication messages in this memory. Since writing speed is much more than the wireless data transfer speed, it gives interrupt free data logging. Except this card, rest of the hardware confirms with requirement of small factor wireless sensor node.

4.3 Reprogramming end node

Comprehensive device and experimental parameter configuration of end node is provided through simple testbed GUI thus reducing need for frequent programming of the end node. A command set is designed between end node, master sink node and desktop server running the testbed application GUI. Reprogramming feature is not provided in this design assuming that advanced user would like to download programme through integrated development environment (IDE) provided along with the development kit. Nonetheless ‘on the fly programming’ feature can be added using same wireless channel allowing reprogramming of the nodes, a necessary feature for protocol evaluation.

4.4 Simple learning curve, hands on operation

SWiFiNet provides windows-based GUI navigating user for setup and configuration of end nodes. GUI commands are simple and intuitive. The result analysis GUI allows button click performance evaluation with standard formulas. Detailed analysis can be made from result text file giving comprehensive details of the logged data.
4.5 Network connectivity

The embedded task distributed master centric network layer in master sink node and end node is self organising and self maintaining. The nodes are automatically detected by routers and master sink node. The router provides seamless connectivity to remote end nodes. The master maintains complete network routing and topology information. The network starts with every device transmitting ‘hello’ packets except Master device. The devices in the range of Master and have LQI value more than the LQI threshold value are explicitly notified by the master to join the network. The devices out of range of master but are in the range of router are notified to be joined provided that the router himself is joined to the network. Thus, user does not have to start with node programming for network connectivity, instead he can concentrate on other parameters of interest and carry out experimentation without programming node and master sink node. This frees user from complexities of node hardware programming eliminating embedded system programming learning curve for novice user.

4.6 Portability and real field emulation

There is no backchannel infrastructure thereby avoiding formation of static topology network. Since the nodes are not attached to any particular external network, these nodes can be placed at any real field location. The node portability enables testbed user to carry out experimentation on field. Results can be obtained without disturbing actual placement. This allows user to carry out same experiment repeatedly. This is very beneficial for application developers, giving insight of environmental conditions affecting the WSN performance.

4.7 Economical solution

SWiFiNet test bed is low cost solution for experimenting short range networking scenarios and applications. The test bed eliminates use of technology intensive infrastructure to setup wireless test bed yet providing all the facilities of test bed. Elimination of technology intensive back channel infrastructure has reduced cost by 90% of the test bed. Not only cost has been reduced but this elimination has provided complete portability and real-time field experimentation possible for the user.

5 Experimenting with SWiFiNet

A laboratory experiment is presented here which demonstrates evaluation of auto-ack feature of the 802.15.4 compliant trans-receivers.

5.1 Auto-ack feature

The automatic acknowledgment is hardware feature of the 802.15.4 compliant trans-receivers [RFC 802.15.4]. A frame transmitted with the acknowledgment request subfield of its frame control field set to one shall be acknowledged by the recipient. If the intended recipient correctly receives the frame, it shall generate and send
an acknowledgment frame containing the same data sequence number from the data frame that is being acknowledged. If an acknowledgment frame is received within macAckWaitDuration symbols the transmission is considered successful. If an acknowledgment is not received within this duration, transmitting device assumes transmission attempt has failed. The device shall repeat the process of transmitting the data or MAC command frame using same DSN and waiting for the acknowledgment, up to a maximum of macMaxFrameRetries (maximum 3) times. If an acknowledgment is still not received after macMaxFrameRetries retransmissions, the MAC sub-layer shall assume the transmission has failed and notify the next higher layer of the failure.

5.2 Laboratory setup

The testbed with four nodes and one master sink node was used to setup the experiment in the laboratory environment. The programmed parameters were fixed for more packet size 128 bytes, and test duration of 1 minute. Experiment was carried out at various packet intervals of 250, 120, 60, 30 and 10 ms. Results were recorded for auto-ack on and auto-ack off configuration. The experiment was designed to find the usefulness of auto-ack feature of the 802.15.4 compliant radios. A router was added on to the scenario and three nodes were attached to this router and router attached to the master. In this case, the traffic generated by the nodes was passed through the router to the master.

5.3 Experimental results

The results are shown in Figures 6(a) and 6(b) for zero hop and one hop communication respectively. In both cases as expected a marked improvement in band width utilisation with auto-ack feature On is seen. In the second case, the communication was throttled using router as hop for the nodes to reach the master sink node. The bandwidth utilisation was poor as compared to zero hop results with and without auto-ack feature. This is justified by the fact that all the communication was passing through single routing node to the master.

The experiment successfully showed use of hardware auto-ack mechanism of the 802.15.4 compliant radios. The results should encourage the user to use this feature instead of soft acknowledgement. The soft acknowledgement increases programming overhead consuming programme memory and uses double bandwidth for the same data rate and more energy.

5.4 Testbed performance evaluation and discussion

The SWiFiNet uses a simple solution for logging/recording of communication by providing large inexpensive memory at each node. The node is supported by interfacing inexpensive micro SD memory card of the memory size ranging from 512 MB to 8 GB. All the incoming and outgoing packets are stored in this memory. 1 GB memory can store up to 480 hrs of data that is 20 days at a rate of 4 packets/second and packet size of 128 bytes with additional bytes for timestamp and other attributes.

The testbed GUI provides facility for uploading the communication log file from the node using the same wireless link established between the node and master. A file transfer protocol is written specially for this task to ensure reliable data transfer.
SWiFiNet: a real field WSN testbed

Figure 6  (a) Results with master directly connected to four nodes (b) Results with master connected to router connected to three nodes (see online version for colours)

The node portability enables the testbed user to carry out experimentation on field. Results can be obtained without disturbing actual placement. This allows the user to carry out the same experiment repeatedly.

SWiFiNet master, nodes and routers have inbuilt task distributed master controlled network layer embedded in the devices. Thus, user does not have to write node
programming for network connectivity, instead he can concentrate on the other parameters of the interest and carry out experimentation without programming node and master sink node. This frees the user from complexities of node hardware programming.

6 Conclusions

In the present day, WSN testbeds are expensive and complex in operation and simulators are unable to justify real-time field interaction. SWiFiNet is designed to be cost effective yet very useful in carrying out much basic as well as advanced experimentation in real-time environment. SWiFiNet is built using small form factor hardware and software for end node, routers and master sink node components. The embedded software is designed firstly for a user transparent network layer to establish connectivity within the network and secondly to accept testbed command set for configuration, experimental parameter controls and operation. Observer or the communication logging process is simplified in SWiFiNet node by providing large inexpensive memory at the node. This replaces the expensive wired/wireless backchannel network. This has many advantages viz. Communication data logging is done at source and destination, logging is at much higher speed without interrupting communication process and putting minimum overhead on the processor. In-built network layer protocol and comprehensive parameter configurability avoids frequent reprogramming efforts for the user. The user can concentrate on other parameters of interest and carry out experimentation without going through learning curve of embedded system programming of node and master sink node. SWiFiNet provides graphical user interface through which all activities of the network can be controlled and monitored. The architecture has been made extremely simple so that user can even deploy the testbed in field of interest. Single window user friendly GUI controlling and monitoring testbed activity makes the facility a novel one. This would help more to people working in application area for testing the communication pattern in real field scenario before actually deploying the system. The flexible device and experiment parameter configurations, wide scalability of network, node status monitoring and experimental data analysis are significant features of SWiFiNet.

6.1 Future enhancements

Considering the goal of designing WSN testbed we would introduce more programmable parameters so as to help researcher without writing code he can experiment with idea similar on the lines of simulator. We want to develop a script-based programming language which will help to program behaviour of the end nodes.

References


