A Remote Code Editing Framework for AMRITA Remote Triggered WSN Laboratory

Conference Paper · August 2015

DOI: 10.1109/ICITCS.2015.7292930

7 authors, including:

Sangeeth Kumar
Amrita Vishwa Vidyapeetham
15 PUBLICATIONS **52** CITATIONS
SEE PROFILE

Preeja Pradeep
Amrita Vishwa Vidyapeetham
14 PUBLICATIONS **18** CITATIONS
SEE PROFILE

Rekha Prabha
Amrita Vishwa Vidyapeetham
21 PUBLICATIONS **43** CITATIONS
SEE PROFILE

Maneesha Vinodini Ramesh
Amrita Vishwa Vidyapeetham, Amritapuri, Ko…
114 PUBLICATIONS **417** CITATIONS
SEE PROFILE

A Remote Code Editing Framework for AMRITA Remote Triggered WSN Laboratory

Sangeeth K, Preeja Pradeep, Divya P, Rekha P, Arya Devi R D, Sheeja L, Maneesha Vinodini Ramesh
Amrita Center for Wireless Networks and Applications
Amrita Vishwa Vidyapeetham
Kollam, Kerala, India
{sangeethk, preejapradeep, divyap, rekhap, aryadevird, sheejal, maneesha}@am.amrita.edu

***Abstract*— Our AMRITA remote triggered lab (RT Lab) for wireless sensor networks (WSN) offer the students and researchers, an easy, efficient, interactive and user friendly environment to trigger their inquisitiveness by providing them with the sensors, equipments, hardwares and study materials for conducting the lab experiments. RT Lab offers a web-based e-learning platform for the registered users to perform experimentation and coding remotely based on the provided study materials which are shared to them virtually. The users can learn nesC programming language and conduct the coding by means of the code editing interface. The sensor nodes in the WSN testbed, deployed in indoor and outdoor environment, undergoes remote reconfiguration and the sensor data's are collected by the WSN gateway. The users can observe the experimentation result such as the plotted sensor data and physical representation of the sensor network along with the remote video through the visualization tool. The paper details the design and implementation of remote code editing platform for RT Lab.**

Keywords—remote triggered lab; remote code editing; remote reconfiguration; over-the-air programming; wireless sensor network

I. INTRODUCTION

The high cost of wireless sensor networks, its non flexibility in programming, energy management etc., has hindered the growth and development of the research area on WSN. Remotely triggered laboratories are an easy and efficient way to provide access to costly and best-in-class equipment without having to acquire them. They are becoming increasingly popular especially in remote areas with fewer resources. Our Wireless Sensor Network Remote Triggered Laboratory (RT Lab) [1] gives opportunity to educators and learners to acquire the expensive equipments and materials needed for the WSN lab experiments via an intuitive web-based e-learning platform. This lab comprises of a set of eleven WSN experiments having micaz motes running TinyOS which are permanently deployed in the WSN testbed. The lab is also equipped with sensors, data acquisition boards, multimeters, cameras, computers and servers. Our RT Lab, where the physical hardware is involved and timeslots are limited, has the provision to remotely reprogram the wireless sensor nodes, execute multiple experiments at the same time, flexible result display for better understanding and debugging of the program.

Reprogramming, a major factor in WSN, is the capability to change software functionality of sensors within the network at run time. The modification may come in the form of updates, bug fixes or modified parameters. Reprogramming is important both during development, for fast prototyping and debugging, and after deployment, for adapting functionality [2].

The paper is organized as follows: Section II discusses the related work done in the remote code updating and reconfiguration in WSN. Section III describes the remote triggered lab code editing framework. Section IV details the implementation and results of the system. Section VII describes the conclusion and the future work.

II. RELATED WORK

Almost all sensor network applications are deployed in hostile regions which demands real-time sensor network access for efficient maintenance and effective management of the network. In the current literature, this has been achieved either by reprogramming schemes or by incremental network programming schemes. The paper [3] discussed various challenges and approaches in WSN reprogramming. Various reprogramming protocols are developed for WSN including MOAP [4], Deluge [5], Trickle [6], MNP [7] etc. The Incremental Network Programming approach [8], provides a faster network reprogramming by reducing the amount of data transmission, minimizing the external flash memory usage and avoiding expensive operation for the sensor nodes. This approach is restricted to a large network because of its incapability of multi-hop delivery.

Various testbeds like Indriya, Motelab, Twist etc uses remote reconfiguration mechanism for their sensor network testbed. In Indriya [9], the input to the web layer is the uploaded application exe plus MIG generated Java class files which defines the structure of messages that application dumps to the USB port. In Motelab [10], the user uploads executables and class file. Using MySQL database, the user details and program details are stored. It has a DBLogger which starts at the beginning of every job. In Twist [11], the web layer input for remote reconfiguration is the executable. The user uploads the executable and gets the data logged results.

This work was supported by the Government of India, Ministry of Human Resources Development (MHRD), titled as "Wireless Remote Sensing, Experimentation, Monitoring and Administration Lab".

978-1-4673-6537-6/15/$31.00 ©2015 IEEE

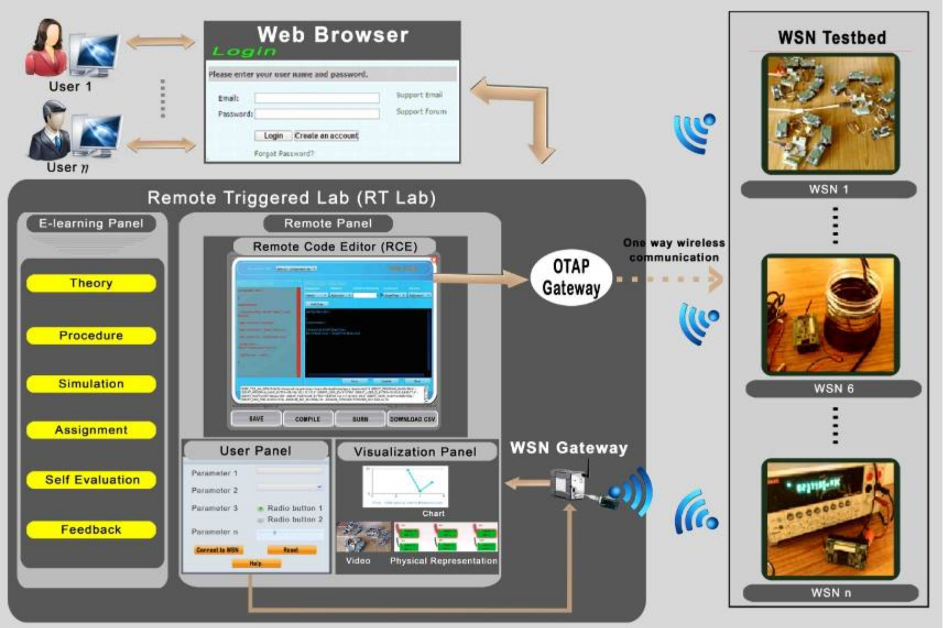


Fig.1. Remote Triggered Wireless Sensor Network Laboratory

We have conducted a study on various testbeds having remote reprogramming facility, which helps us to bring a better remote code editing platform design along with reconfiguration mechanism for our RT Lab.

III. Remote Code Editing For Amrita RT Lab

Our RT Lab [12] offers a remote code editing mechanism for registered users, which allows them to have hands on experience with programming the sensor nodes. The central aim is to provide an efficient and effective high level interaction between users and wireless sensor nodes with the help of a code editing panel which includes options for editing, saving, compilation of the source code and burning of the firmware image in to the nodes. After finishing the code editing, the user can burn the new firmware image in to the sensor nodes using our provided reprogramming facility. The user can experience the rebooting of the motes by watching the real time video of the setup which is streamed using Media Server [12] and also verify the result from the chart which shows the plotted data from the sensor node.

The main tab in the RT Lab is the Remote Panel, as shown in Fig.1, which offers User Panel and Remote Code Editor (RCE) for users to conduct experimentation and a Visualization Panel for visualizing the results. The User Panel allows the user to run the experimentation by changing the parameters provided and redirects the user to perform coding in the provided RCE. The Initialization in the User Panel performs the rebooting of the sensor nodes with the original firmware of the experiment. The Connect to WSN helps the user to initiate the connection to WSN Gateway and thereby initiate communication with WSN Testbed. The RCE offers the user a provision for writing/editing the experiment program code according to their requirement and reprograms the sensor nodes in WSN Testbed. The system uses over-the-air programming (OTAP) mechanism, to manage the large WSN. The updates to the WSN Testbed are done over-the-air. An OTAP gateway is used for conducting the reprogramming of the sensor nodes in the WSN Testbed. During the OTAP process, the OTAP Gateway sends a command to the node to reboot into the OTAP image. However this remote code editing setup allows the users to experience all the changes happened to WSN in real time.

Our system offers three ways of code editing mechanisms [12]:

- Parameter Configuration: In this method, the user is permitted to change the parameters of an already existing wireless sensor network program. Here, the user is not allowed to upload a new program code or modify an existing program code. However the user will be able to vary the parameter that will provide the opportunity to learn about different configurations and its impact on the wireless sensor network.
- Restricted Code Editing: The code editor window controls the code editing from user by restricting the modification of an already existing program. Here the user can act according to the instruction provided by the system.
- Full Code Editing: The code editor window permits the user to add or modify the sensor network program code. Here a user can rewrite the program code of an experiment without any restrictions.

In Remote Code Editing, a code editor window will provide the original code as well as the window for code editing to the users, where the user can make modification to the code. Only registered users can access the code editor window, RCE, of each WSN experiment from the Remote Panel. The code editing mechanism provides services such as edit, save, compile and burn the program. This architecture allows each registered user to access the program code of each of the experiments to make modification according to the allowed editing type such as parameter configuration, restricted code editing, or full code editing. After modification, the user can save the code into the Archive Server provided by the Remote Backend Server. An Archive Server is used to keep the program codes of each experiment as per the user. For compiling the program, the user has to initiate a compile request. Once the user receives the compile request, the server loads the code, invokes the batch file and executes the compile command. Then the compiled result is displayed in the web page.

A. OTAP Setup

The back bone of Remote Code Editing mechanism is the OTAP. OTAP plays a major role in our architecture which makes the system more scalable and flexible. The XMesh OTAP Service, XOtap [13] is a reliable model where the firmware images are controlled by the gateway. XOtap supports four different slots where slot 0 is reserved for the OTAP image and slots 1, 2 and 3 can be used for user-specified firmware. Here the firmware image is divided into pages and each page consists of N fragments. The main advantages of this page structure are to reduce RAM requirements for maintaining state, of which packets are needed, reduces traffic congestion during the process of image download and to help in fitting into standard TinyOS packets. XOtap Bootloader is an independent piece of code which guarantees the execution of the mote after each reset. It helps in copying image from specified slot into program memory in external flash and execution of newly loaded program.

The pre-conditions for conducting the over-the-air programming are:

- All the motes should be OTAP enabled
- Pre-install the OTAP Image in slot 0
- Bootloader should be installed

The main objective is to provide a scalable and flexible wireless sensor network reprogramming environment to users.

We used TinyOS 1.0 [14] for programming the sensor nodes which supports reconfiguration of sensor nodes on the run and MoteConfig [13] utility for testing and enabling the OTAP facility in sensor nodes. The reprogramming technique used in TinyOS is the system level reprogramming, in which the updated code is first stored in external flash memory and is copied into internal flash memory upon system restart.

Each user is allocated with a particular time slot for each experiment. Upon the completion of the whole reprogramming of a particular experiment, the server initiates a reboot command which reboots the deployed motes with the original image, so that the next user could able to use the original code.

The following steps are taken before deploying the wireless sensor nodes of each experiment.

1) Local Programming

a) Generating the firmware image: This step is to build the program code of each experiment and make the firmware image.

b) Setting up the motes by assigning channels, mote id and group id

c) Adding OTAP capability

We used MoteConfig utility to enable OTAP capability of the motes. The local program component in the MoteConfig can be used to assign the mote id, group id- a unique id of the mote, RF power – the radio power of the mote and RF channel for each mote according to the experiment. OTAP functionality of the mote can be enabled by selecting the OTAP enable option in the local programming window. After selecting the configuration details, firmware and OTAP enable, the mote can be programmed by choosing Program which enable the OTAP feature in motes and downloads the firmware and configuration into the motes. An OTAP gateway is used for conducting the reprogramming which is assigned with a node ID of 0 and programmed with the XMeshBase [13] application. There is no need of enabling the OTAP option for the OTAP gateway. For each experiment, the group id of the OTAP gateway and the motes should be the same.

Once all the motes in the WSN Testbed are OTAP-enabled according to the experiments, the next procedure is to start the remote programming using MoteConfig utility.

2) Remote Programming

The Over-The-Air-Programming (OTAP) feature allows users to reprogram a mote over a wireless channel. OTAP allows one or more Motes in the network to receive new firmware images from XServe (via the XOtap service) [13]. With the help of the remote program component in the MoteConfig, the remote programming of the WSN Testbed has been conducted. The following are the steps used in testing the remote programming capability of the motes.

a) Search the motes: The first step is to search the remote motes with the help of OTAP gateway using the Search command.

b) Prepare the motes: After the motes starts listening to the OTAP gateway, then the motes can be rebooted to the OTAP image by the help of Prepare command. I.e. the motes must be prepared by enabling the bootloader and loading the OTAP image into slot 0, before programmed it over-the-air.

c) Program the motes remotely: The remote programming of a WSN Testbed can be done by loading the firmware images into the selected slot of the motes according to the experiment using the Program command.

d) Reboot the motes: The final step is to select the newly loaded image from the slot and reboot the motes using the Reboot command.

After testing the remote programming capability of each node, we deployed the sensor nodes in the indoor and outdoor environment.

B. Remote Code Editor

Each experiment in the RT Lab provides the user a provision for writing their own program code or editing the original code according to their requirements. RCE is a platform that allows users to perform the remote code editing.

A user can write their own code after understanding each experiment. RCE provides the user with the original code of the experiment in one window on the editor.

When an educator/learner conducts the remote programming using the RCE, the following steps happens.

- Save: After writing the program code, user has to save the code before proceeding to compilation process.
- Compile: The compile helps the user to know whether the wiring of the components is done correctly or not. A firmware image is created after the successful compilation process.
- Burn: After initiating the Burn command, flashing the mote with the firmware image happens via OTAP and the mote gets rebooted with the new program. When the user initiates the Burn command, first the script for downloading the firmware into the mote and later the mote reboot happens.

Here XOtap is used to perform the over-the-air programming where the firmware image is broken up into fragments and transmitted to the mote. The OTAP Gateway sends a message to transfer the newly uploaded firmware into the program flash and reboot the mote.

- Visualization: The results, including video, chart and physical representation can be viewed in the Visualization Panel after the remote programming completes.

The RCE provides the users a provision to download the csv file of the experimentation they done. The Archive Server logs the data and upon getting the user request it fetches the data file and returns it to the user.

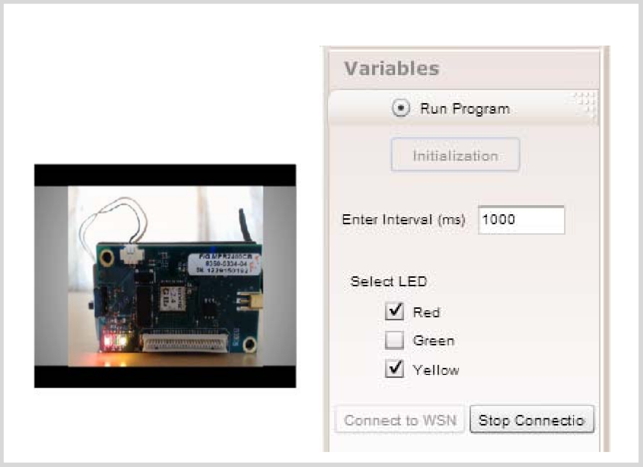
IV. Implementation and Result

Our WSN remote triggered lab consists of 11 WSN experiments [15] where the testbed includes 9 indoor and 2 outdoor sensor networks with more than 150 Memsic micaz motes, dielectric moisture sensors, rain gauge, on-board sensors, data acquisition boards, digital multimeters etc. The WSN Testbed is permanently deployed which provides data logging, reprogramming of nodes, and test bed scheduling.

Using Flash Media Server (FMS), the videos of the WSN test bed are getting streamed. A minimum of 11 parallel webcams are used to stream our 11 deployed experiments simultaneously. Flash Media Live Encoder is used to stream the webcam's video-feed to FMS and FMS streams the video-feed onto internet [1].

We have conducted the testing of remote reconfiguration and found that the estimated time taken to download the new firmware image varies with respect to the size of the firmware image. I.e. For a 1KB image, the time taken to download the image into the motes is 0.0095 seconds.

To better understand the difference in the result of remote code editing and remote experimentation of a specific experiment, initially we have to start with the remote experimentation. The Fig.2 shows the result of the remote experimentation where the red and yellow LED of the mote starts blinking for an interval 1000ms. The stop connection shown in the Fig.2 is used for stopping the connection to WSN Testbed. The Fig.3 and 5 shows the remote code editing platform and the console window respectively. The console window shows the messages regarding the compilation and burning process to the user. Next, we performed the remote code editing of the same experiment by changing the LED as green, as shown in Fig.4, and the result in visualization panel shows the blinking of green LED, as shown in Fig.6.

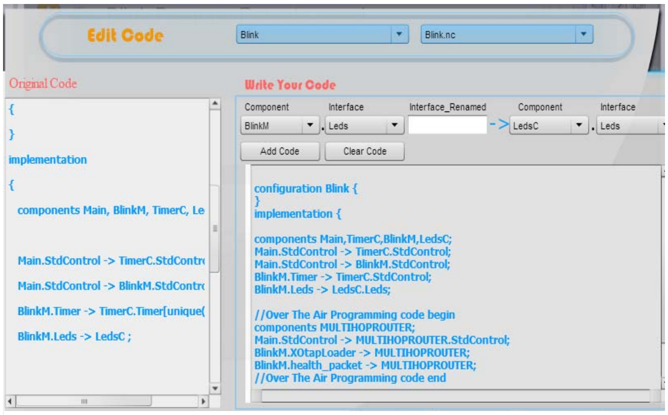


● Run Program

Enter Interval (ms) 1000

Select LED
☑ Red
☐ Green
☑ Yellow

Fig.2. User Panel of RT Lab experiment after remote experimentation

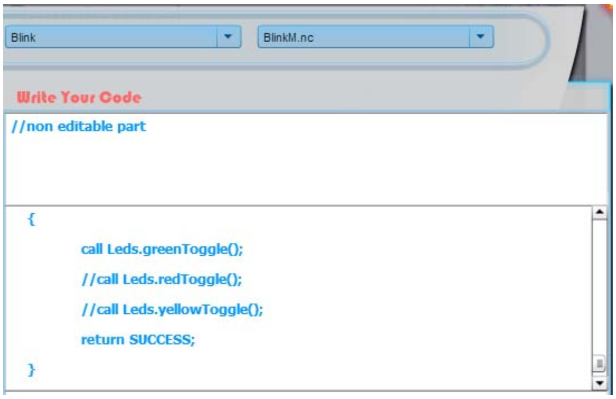


Component: BlinkM ; Interface: Leds ; Interface_Renamed: ; -> Component: LedsC ; Interface: Leds

```
{
}
implementation
{
  components Main, BlinkM, TimerC, Le


  Main.StdControl -> TimerC.StdContr

  Main.StdControl -> BlinkM.StdContr

  BlinkM.Timer -> TimerC.Timer[unique(

  BlinkM.Leds -> LedsC ;
```

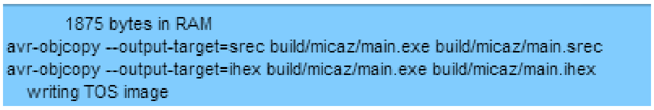
```
configuration Blink {
}
implementation {

components Main,TimerC,BlinkM,LedsC;
Main.StdControl -> TimerC.StdControl;
Main.StdControl -> BlinkM.StdControl;
BlinkM.Timer -> TimerC.StdControl;
BlinkM.Leds -> LedsC.Leds;

//Over The Air Programming code begin
components MULTIHOPROUTER;
Main.StdControl -> MULTIHOPROUTER.StdControl;
BlinkM.XOtapLoader -> MULTIHOPROUTER;
BlinkM.health_packet -> MULTIHOPROUTER;
//Over The Air Programming code end
```

Fig.3. Remote code editing platform



```
//non editable part

  {
        call Leds.greenToggle();
        //call Leds.redToggle();
        //call Leds.yellowToggle();
        return SUCCESS;
  }
```

Fig.4. Remote code editing of the experiment



```
        1875 bytes in RAM
avr-objcopy --output-target=srec build/micaz/main.exe build/micaz/main.srec
avr-objcopy --output-target=ihex build/micaz/main.exe build/micaz/main.ihex
    writing TOS image
```

Fig.5. Console window of remote code editing platform

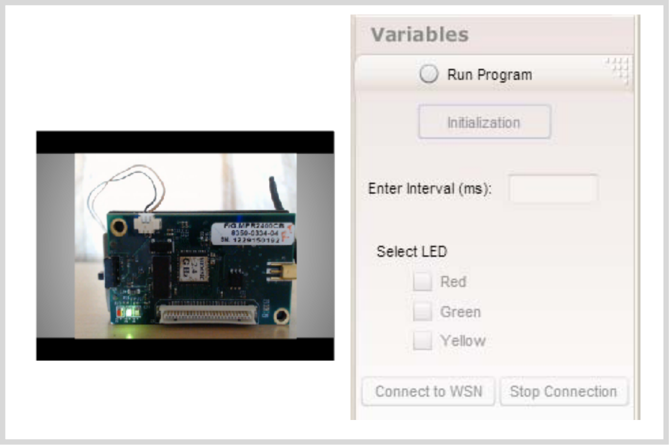


Fig.6. Result of remote code editing in visualization panel

We have conducted an evaluation of WSN experiments in a total of 22 PG students on Jan-Jun 2015 semester. From the evaluation, we found that 50% of the students rate the performance of the experiment as excellent and other 41% says it's very good. Also, 90% of the students reported that doing experiments through this lab gives scope for more innovative and creative research work.

V. CONCLUSION AND FUTURE WORK

Throughout the research work, we faced operational, managerial, and technical challenges. One of the major challenges for the implementation of RT Lab is the development and implementation of remote editing, and reprogramming of wireless sensor nodes. The current setup provides the opportunity to remote users to perform options like parameter configuration, restricted code editing and full code editing, which offers a better learning experience. In general, our complete system incorporates the concepts of virtual and remote triggered laboratory, thus providing an effective and efficient learning experience to remote users. To understand the impact of our RT Lab, we conducted a survey on students where the results show that 90% of the students found this lab very informative and helpful with very good control over the experiments.

In future, we will be looking into feasibility of users compiling their application and uploading the code in the sensor nodes. This includes validating the security aspects and also transfer time of the binary to the node. Also, we will focus on enhancing this testbed by adding more experiments, optimizing the network based on bandwidth utilization, and developing better medium access protocol to enhance the scalability of the complete system.

ACKNOWLEDGMENT

We would like to express our immense gratitude to our beloved Chancellor Sri. Mata Amritanandamayi Devi (AMMA) for providing the motivation and inspiration for doing this research work. We also would like to thank Mr.Rayudu Y.V, Mr.Rakesh K, Mr.Tinu Vinod, Mr.Ratheesh Kumar P, Ms. Nithya G, Mr.Vysakh K, and many other colleagues and friends as well for their support.

REFERENCES

[1] Maneesha Vinodini Ramesh, Preeja Pradeep, Divya P, Arya Devi R D, Rekha P, Sangeeth K, Rayudu Y V, "AMRITA Remote Triggered Wireless Sensor Network Laboratory Framework", Proc. of the 11th ACM Conference on Embedded Networked Sensor Systems (Sensys), doi:10.1145/2517351.2517377, 2013.

[2] Stolikj, M., Cuijpers, P.J.L., Lukkien, J.J., " Efficient reprogramming of wireless sensor networks using incremental updates and data compression", Proc. of IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), Page(s): 584 – 589, doi:10.1109/PerComW.2013.6529563, 2013.

[3] Qiang Wang, Yaoyao Zhu, and Liang Cheng, "Reprogramming Wireless Sensor Networks: Challenges and Approaches", IEEE Network, vol. 20, pp.48 - 55, doi:10.1109/MNET.2006.1637932, June 2006.

[4] T. Stathopoulos, J. Heidemann, and D. Estrin, "A remote code update mechanism for wireless sensor networks, CENS technical report #30," http://lecs.cs.ucla.edu/ thanos/ moap-TR.pdf., 2003.

[5] J. Hui, and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale", Proc. of the 2nd International Conference on Embedded Networked Sensor Systems, , pp. 81-94, doi: 10.1145/1031495.1031506, Nov 2004.

[6] P. Levis, N. Patel, D. Culler, and S Shenker, "Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks", Proc. of the 1st conference on Symposium on Networked Systems Design and Implementation (NSDI), pp. 2-2, 2004.

[7] S. S. Kulkarni, and L. Wang, "MNP: Multihop Network Reprogramming Service for Sensor Networks," Proc. of 25th IEEE International Conference on Distributed Computing Systems, ICDCS, pp. 7–16, doi:10.1109/ICDCS.2005.50, 2005.

[8] J. Jeong and D. Culler, "Incremental Network Programming for Wireless Sensors", Int'l J. of Communications, Network and System Sciences, Vol. 2 No. 5, pp. 433-452. doi:10.4236/ijcns.2009.25048, 2009.

[9] Manjunath Dodda venkatappa, MunChoon Chan, and Ananda A.L, "Indriya: A Low-Cost, 3D Wireless Sensor Network Testbed", Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 90, pp. 302-316, 2012.

[10] Werner-Allen, G., Swieskowski, P. and Welsh, M., "MoteLab: a wireless sensor network testbed", Proc. of 4th International Symposium on Information Processing in Sensor Networks, pp. 483 - 488, doi:10.1109/IPSN.2005.1440979, 2005.

[11] Vlado Handziski, Andreas Kopke, Andreas Willig, Adam Wolisz, "TWIST: A Scalable and Reconfigurable Testbed for Wireless Indoor Experiments with Sensor Networks", Proc. of the 2nd Intl. Workshop on Multi-hop Ad Hoc Networks: from Theory to Reality, (RealMAN 2006), May 2006.

[12] Preeja Pradeep, Divya P, Arya Devi R D, Rekha P, Sangeeth K, Maneesha Vinodini Ramesh, "A Remote Triggered Wireless Sensor Network Testbed", Wireless Telecommunications Symposium (WTS), pp. 1 - 7, doi:10.1109/WTS. 2015.7117262, 2015.

[13] MoteConfig User's Manual, www.xbow.jp/moteconfigmanual.pdf

[14] [Online]. Available: http://tinyos.stanford.edu/tinyos-wiki/index.php/

[15] VALUE @ Amrita: Wireless Sensor Network Remote Triggered Lab, http://vlab.amrita.edu/index.php?sub=78.