

# [Tutorial at ACM/IFIP/USENIX Middleware 2008]

## Programming Wireless Sensor Networks: From Theory to Practice

### Information for Attendees

Luca Mottola<sup>1</sup> and Gian Pietro Picco<sup>2</sup>

<sup>1</sup> Bruno Kessler Foundation, Trento, Italy, [mottola@fbk.eu](mailto:mottola@fbk.eu)

<sup>2</sup> University of Trento, Italy, [gianpietro.picco@unitn.it](mailto:gianpietro.picco@unitn.it)

This document targets attendees of the tutorial “Programming Wireless Sensor Networks: From Theory to Practice” at ACM/IFIP/USENIX Middleware 2008. It provides instructions on how to prepare a laptop machine for the hands-on session of the tutorial. Additionally, a brief list of references is also included for those willing to give a look at the subject in advance or to have additional printed material during the tutorial.

## 1 Preparation for the Hands-On Session

The steps required to setup a machine for the hands-on session of the tutorial are divided in two parts: i) setting up the development environment, and ii) setting up the tutorial tools and code examples. In case you faced problems not described here or in the external pointers provided, you are welcome to send an email to Luca Mottola at [mottola@fbk.eu](mailto:mottola@fbk.eu).

### 1.1 Development Environment

The hands-on part of the tutorial is partly based on nesC/TinyOS, the most widespread programming platform for WSNs to date. The reference version is TinyOS 2.0.2. There exist various ways to install the corresponding development environment. General instructions can be found at:

[http://docs.tinyos.net/index.php/Getting\\_started](http://docs.tinyos.net/index.php/Getting_started)

For the purpose of the tutorial, however, we *strongly* recommend to use XubunTOS v7.04, a customized Ubuntu Linux distribution that is already equipped with the entire nesC/TinyOS tool-chain and is already configured in every aspect. More information on XubunTOS can be found at:

<http://toilers.mines.edu/Public/XubunTOS>.

Essentially, there are three ways to run XubunTOS. They all yield the same result, but some of them may not be applicable depending on the laptop at hand:

1. *Bootable LiveCD*: provided your laptop is equipped with a CD/DVD driver and is able to boot from that, you can download the ISO file from the URL above, burn it on a CD, and use that CD to boot your machine. In this case, the configuration of all hardware devices, possibly including display and network card, is automatically carried out when the operating system boots. If that fails, however, you need to deal with that directly. Thus, we recommend this option if all the hardware is correctly identified and you are *not* planning on using XubunTOS again after the tutorial.
2. *Dual-boot Operating System*: the LiveCD functionality described above provides an option by which XubunTOS can be permanently installed on the laptop’s hard-drive. This choice might solve some issues in hardware detection, but it may be more difficult to remove XubunTOS afterwards. Therefore, we recommend this option if you do plan to keep using XubunTOS afterwards.
3. *Virtual Machine*: the LiveCD can be run inside a virtual machine such as VMWare. Instructions on how to run XubunTOS inside VMWare Player are found at:

[http://sing.stanford.edu/klueska/running\\_xubuntos\\_vm.html](http://sing.stanford.edu/klueska/running_xubuntos_vm.html)

Similar procedures apply also to MAC OSX users running VMWare Fusion. In this case, you can still decide whether to run XubunTOS in a (virtualized) LiveCD or install it permanently on a (virtualized) hard-drive. This option is ideal if your machine is powerful enough to run a Linux distribution inside a virtual machine, as the virtualized hardware is always correctly recognized and the host operating system remain simultaneously accessible (and unchanged).

## 1.2 Tools and Code Examples

The tools and code examples are included in a .zip that can be downloaded from the website of Logical Neighborhoods, a programming abstraction that will be described during the hands-on session. Logical Neighborhoods' website is:

<http://logicalneighbor.sourceforge.net>

Click on the tutorial link on the left to download the file. Once downloaded, you need to unzip it in some convenient location—e.g., your home directory—*inside* your Xubuntu installation. The exact path is irrelevant, as long as you have read/write privileges in that location.

## 2 Reading Material

As introduction to WSNs, we take [1] as reference. With respect to the specific topics taught during the tutorial, the core material is represented by [2–16]

## References

1. Zhao, F., Guibas, L.: *Wireless Sensor Networks: An Information Processing Approach*. Morgan Kaufmann (2004)
2. Gay, D., Levis, P., von Behren, R., Welsh, M., Brewer, E., Culler, D.: The nesC language: A holistic approach to networked embedded systems. In: Proc. of the ACM SIGPLAN Conf. on Programming Language Design and Implementation (PLDI). (2003)
3. Mottola, L., Picco, G.P.: Logical Neighborhoods: A programming abstraction for wireless sensor networks. In: Proc. of the the 2<sup>nd</sup> Int. Conf. on Distributed Computing on Sensor Systems (DCOSS). (2006)
4. Mottola, L., Picco, G.P.: Programming wireless sensor networks with Logical Neighborhoods. In: Proc. of the the 1<sup>st</sup> Int. Conf. on Integrated Internet Ad hoc and Sensor Networks (InterSense). (2006)
5. Costa, P., Mottola, L., Murphy, A.L., Picco, G.P.: Programming wireless sensor networks using the TeenyLIME middleware. In: Proc. of the 8<sup>th</sup> ACM/USENIX Int. Middleware Conf. (2007)
6. Madden, S., M.J. Franklin, J.M. Hellerstein, Hong, W.: TinyDB: an acquisitional query processing system for sensor networks. *ACM Trans. Database Syst.* **30**(1) (2005)
7. Luo, L., Abdelzaher, T.F., He, T., Stankovic, J.A.: Envirosuite: An environmentally immersive programming framework for sensor networks. *Trans. on Embedded Computing Sys.* **5**(3) (2006)
8. Kothari, N., Gummadi, R., Millstein, T., Govindan, R.: Reliable and efficient programming abstractions for wireless sensor networks. In: Proc. of the ACM SIGPLAN Conf. on Programming Language Design and Implementation. (2007)
9. Newton, R., Morrisett, G., Welsh, M.: The regiment macroprogramming system. In: Proc. of the 6<sup>th</sup> Int. Conf. on Information Processing in Sensor Networks (IPSN). (2007)
10. Frank, C., Römer, K.: Algorithms for generic role assignment in wireless sensor networks. In: Proc. of the 3<sup>rd</sup> Int. Conf. on Embedded Networked Sensor Systems (SENSYS). (2005)
11. Gummadi, R., Gnawali, O., Govindan, R.: Macro-programming wireless sensor networks using Kairos. In: Proc. of the 1<sup>st</sup> Int. Conf. on Distributed Computing in Sensor Systems (DCOSS). (2005)
12. Fok, C.L., Roman, G.C., Lu, C.: Rapid development and flexible deployment of adaptive wireless sensor network applications. In: Proc. of the 25<sup>th</sup> IEEE Int. Conf. on Distributed Computing Systems (ICDCS). (2005)
13. Li, S., Lin, Y., Son, S.H., Stankovic, J.A.: Event detection services using data service middleware in distributed sensor networks. *Telecommunication Systems* **26**(2) (2004)
14. Bakshi, A., Prasanna, V.K., Reich, J., Lerner, D.: The Abstract Task Graph: A methodology for architecture-independent programming of networked sensor systems. In: Workshop on End-to-end Sense-and-Respond Systems (EESR). (2005)
15. Whitehouse, K., Sharp, C., Brewer, E., Culler, D.: Hood: a neighborhood abstraction for sensor networks. In: Proc. of 2<sup>nd</sup> Int. Conf. on Mobile Systems, Applications, and Services (MobiSys). (2004)
16. Welsh, M., Mainland, G.: Programming sensor networks using abstract regions. In: Proc. of 1<sup>st</sup> Symp. on Networked Systems Design and Implementation (NSDI). (2004)