Journal of Network and Computer Applications 37 (2014) 12–24

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

A multi-channel architecture for IPv6-enabled wireless sensor and actuator networks featuring PnP support

Paulo A. Neves [a,b], Joel J.P.C. Rodrigues [a,*], Min Chen [c], Athanasios V. Vasilakos [d]

[a] Instituto de Telecomunicações, University of Beira Interior, Portugal
[b] Polytechnic Institute of Castelo Branco, Castelo Branco, Portugal
[c] Seoul National University, Seoul, Korea
[d] National Technical University of Athens (NTUA), Greece

ARTICLE INFO

Article history:
Received 7 July 2010
Received in revised form
26 December 2010
Accepted 30 March 2011
Available online 10 May 2011

Keywords:
Wireless sensor and actuator networks
Data gathering on WSANs
IPv6
6LoWPAN
Plug-and-Play
ContikiOS

ABSTRACT

Wireless sensor and actuator networks provide a distributed system composed of wirelessly connected smart sensor and actuator nodes, suitable for cost-efficient control applications. An important research challenge is deployment, where features like node auto-configuration, unattended operation, and Internet connectivity are becoming mandatory. Moreover, on off-the-shelf solutions the user typically must be network technology-savvy to take advantage of sensing and actuation services. This paper presents a novel multi-channel architecture for sensor data gathering and actuation, featuring Plug-and-Play like functionality for node attachment and operation, IPv6 at the node level, and dedicated communication semantic protocols—the ZenSens system architecture. The architecture features the sensor/actuator nodes, a personal computer application (SenseLab), a mobile application (SenseLab mobile), and World Wide Web access (WebSensor), presenting the user with a complete sensing and actuation solution. As a result the user can operate the network without technological background, and near-zero configuration. All developed software and firmware are presented, and validated through a series of experiments on real hardware, namely using a test-bed with TelosB motes running ContikiOS.

1. Introduction

Wireless sensor and actuator networks (WSANs) present a distributed environment of wirelessly connected nodes that can present sensing and/or actuator control capabilities, thus making them suitable for cost-efficient deployments (Xia et al., 2007). Research community has centered much of its efforts on wireless sensor networks (WSNs) (Baronti et al., 2007), leaving WSANs as a relatively new area of research. Typically WSNs are considered for many control applications at the sensing level, leaving the control and actuation planes for other systems to perform.

A WSAN is also composed of small smart nodes that communicate wirelessly with processing core and memory, a battery, one or more base stations (sinks), and sensor nodes. However, WSANs introduce actuator nodes, small smart nodes similar to sensor nodes, but with actuation capabilities. This small difference brings a new plethora of applications and challenges. Small smart nodes can have sensors, actuators, both or none. Sensors capture information, actuators make decisions and take actions based on the input from sensors, and sinks monitor overall network, communicating with both sensor and actuator node types to reach network's goals (Rezgui and Eltoweissy, 2007).

A WSAN can also benefit from Internet connectivity, exposing its sensing services and actuation information worldwide. Two main approaches can be considered when Internet connectivity is mandatory, namely proxy-based and node stack-based (Rodrigues and Neves, 2010). WSN/WSAN with Internet connectivity makes ubiquitous computing realistic (Stankovic, 2008), turning sensor and actuator networks into ubiquitous networks. The node stack-based approach has been gaining popularity (Hui and Culler, 2008; Yang et al., 2008; Silva et al., 2008; Han and Ma, 2007), namely with support from two of the major operating systems for embedded objects—TinyOS (Levis et al., 2004) and ContikiOS (Dunkels et al., 2004).

This paper proposes a system architecture for WSANs, called ZenSens, with user-centric perspective, namely featuring different software access channels, and zero-user configuration. As a result the user can operate and take advantage of the network's services without technological background. The architecture supports Plug-and-Play (PnP) from the ground up, where nodes present themselves to the sink, with automatic network attachment, in a hassle-free and transparent way. With the ZenSens system a

* Corresponding author at: Instituto de Telecomunicações, University of Beira Interior, 6201-001 Covilhã, Portugal.
E-mail addresses: pneves@co.it.pt (P.A. Neves), joeljr@ieee.org (J.J.P.C. Rodrigues), minchen@ieee.org (M. Chen), vasilako@ath.forthnet.gr (A.V. Vasilakos).

1084-8045/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jnca.2011.03.033

user can deploy and automatically monitor a WSAN through different channels, putting the user in control without digging into WSAN technology, mobile or personal computer technologies. ZenSens enables the user to focus on the physical system to be controlled, obtaining results in an effective way.

PnP in WSNs/WSANs (Sung et al., 2009) is a fundamental aspect of network operation, enabling automatic configuration of nodes, abstraction from the attachment process, while contributing for dynamic topology designs. Another added benefit is also the capacity to integrate new hardware platforms without system changes or existing node reprogramming.

The software layers have knowledge of network condition and assets through a USB-connected sink that manages and coordinates connected remote nodes. For user access, the system presents a multi-channel approach: computer-based (SenseLab), mobile-based (SenseLab mobile), and Internet-browser based (WebSensor). The personal computer application is also in charge of distributing information obtained from the WSAN to the other two channels through XML files. The mobile application brings portability and convenience to the solution, while World Wide Web access takes the system into a global scale.

The WSAN level is based and demonstrated on a test-bed using TelosB motes (Polastre et al., 2005), running the Contiki operating system. All motes are IPv6-enabled through uIPv6 (Durvy et al., 2008), using the current implementation of 6LoWPAN available on Contiki (sicslowpan). This "future-proof" approach enables seamless Internet connectivity of network nodes.

The remainder of the paper is organized as follows. Section 2 presents some background and related work, while Section 3 elaborates on the ZenSens system architecture. Section 4 presents USB and UDP communication, node's firmware, and PnP support, while Sections 5 and 6 detail SenseLab and SenseLab mobile applications. Section 7 presents WebSensor web solution, while Section 8 elaborates on system tests and validation. Finally, Section 9 presents the conclusions and planned future work.

2. Background and related work

This section elaborates on the system's technological base—IPv6, 6LoWPAN, and ContikiOS—and presents motivation for this work. IPv6 brings several benefits to embedded smart objects that require seamless Internet connectivity (Silva et al., 2008).

The 6LoWPAN specification enables the transmission of IPv6 packets over standard IEEE 802.15.4 networks with support for header compression (Montenegro et al., 2007). This specification defines frame format for transmission of IPv6 packets, establishment of local-link addresses, and stateless auto-configuration. IEEE 802.15.4 defines physical and link-layer communication for small embedded devices, suitable for WSN/WSAN. Moreover, since IEEE 802.15.4 is present in the majority of sensor hardware (motes), the application of 6LoWPAN is almost mandatory for IPv6 node stack approaches. Furthermore, a very recent book presents 6LoWPAN and its benefits in several application scenarios, helping the adoption of IPv6-enabled sensor networks (Shelby and Bormann, 2009). Another work presents a study on performance of 6LoWPAN implementation over TinyOS, with TelosB and MICAz motes, using ICMP payloads (Cody-Kenny et al., 2009).

Contiki is an operating system for embedded smart objects, realizing the vision of "The Internet of Things", by enabling IP communication on very constrained smart sensor node. It uses ANSI C, as opposed to TinyOS nesC, and it is currently adopted by many research teams worldwide. Among the assets of Contiki, it is the uIPv6 communication stack that enables IPv6 communication for smart embedded devices, with MAC and link-layer agnostic features. Namely, uIPv6 stack can potentially run over IEEE 802.15.4, Ethernet, and IEEE 802.11.

Contiki supports an event-driven model with a form of multiprocessing designated as protothreads. Protothreads present a multiprocessing-like environment, but with a shared stack system, thus resulting in a memory-efficient alternative to threads. However, since all protothreads share the same stack, care must be taken with internal variables. Two types of events can be used—system and in-program defined. Event timers are also available, enabling support for periodic events. Protothread processing may be event-driven, thus turning it into an event handler.

The choice of ContikiOS over TinyOS was based in several parameters. ContikiOS presents almost no learning curve for C programmers, when compared to TinyOS, which uses nesC as the development language. When considering IPv6 implementation, ContikiOS is pioneer, offering a set of tools and the uIPv6 stack. Third TelosB, the current test-bed hardware platform, is very well supported in Contiki. Features like the availability of documentation, a pre-configured virtual machine and open source code are common with TinyOS, although we found the virtual machine in ContikiOS, instant-Contiki over Ubuntu, to be more adequate than TinyOS over XUbuntos. Although ContikiOS documentation is far from perfect, the discussion lists are simply phenomenal with real help from the code contributors, including ContikiOS' creator.

Authors believe IPv6 paves the way for the Future Internet, by enabling not only hosts, but also small smart devices to be part of Internet, realizing the "Internet of Things" vision. In this regard 6LoWPAN efforts may accelerate the adoption of IP-enabled WSN designs and deployments. IP-enabled WSN research has been focused on simulation-based approaches, use of test-beds for performance assessment, some real deployments, and commercial solutions. However, application layer, namely user interaction, has been overlooked. To the best of the author's knowledge, no other solution presents multi-channel visualization tools, PnP-like operation, and support for IPv6-enabled WSANs.

Developing over real sensor devices can be difficult and tedious (Ramanathan et al., 2005; Köpke et al., 2008), since unlike personal computer programming, programs must be sent to the target hardware, which is a time consuming "non-productive" task. This time consuming process may lead to simulation-based approaches, thus shortening the development cycle. On ContikiOS two tools that can be used for simulating node behavior, namely COOJA (Österlind et al., 2006) and MSPsim (Eriksson et al., 2009, 2007). The latter one can be used for stand-alone debugging on MSP430 processor-based boards such as TelosB, while COOJA can simulate complete networks. Nevertheless, real hardware deployments always provide a deeper insight, enabling test-bed environments, clearly presenting a better match to real scenario conditions.

The current approach is a step-up on previous work on WSNs (Neves et al., 2010a,b), taking the architecture and assets further to actuator networks. The system is based on a WSN, and as a result does not feature actuator assets. Also the UDP communication inside the network was fully revised on this new version, separating node attachment from the data gathering process. The inclusion of actuators also implied changes on the node's firmware, SenseLab, and WebSense, with the introduction of new features.

The current work is evaluated and demonstrated through the implementation on real devices (motes, computers, and mobile devices), debugging and throughout testing, with development assist from simulation software at early stages. Our motivation is based on bringing WSN/WSANs into mainstream use, namely finding "killer applications" for a technology that is still used on very specific applications, with Internet connectivity to achieve the vision of "Internet of Things". Applications stem from smart