



UNIVERSITY OF TRENTO - Italy



9th European Conference on Wireless Sensor Networks

EWSN 2012

February 15-17, 2012

Polo Scientifico Fabio Ferrari - Povo, Trento (Italy)

Poster and Demo Proceedings



Poster and Demo Chairs

Amy L. Murphy - Bruno Kessler Foundation, Italy
Thiemo Voigt - Swedish Institute of Computer Science, Sweden

Organizing Secretariat

Events, Magazines and Internal Communication Office
University of Trento

Contents

Applications	5
Demo Abstract: IMAC, Enabling Flexible Configuration and Result Analysis for Diverse Wireless Sensor Network Experiments. <i>Richard Figura, Sascha Jungen, Ramin Soleymani, Chia-Yen Shih, Pedro José Marrón.</i>	6
Demo Abstract: Wireless Sensor Networks for the Real World: A Collage of Experiences in Trento. <i>Matteo Ceriotti, Ramona Marfievici, Davide Molteni, Amy L. Murphy, Gian Pietro Picco.</i>	8
Demo Abstract: Route Selection of Mobile Sensors for Air Quality Monitoring. <i>Olga Saukh, David Hasenfratz, Abouzar Noori, Tamara Ulrich, Lothar Thiele.</i>	10
Poster Abstract: Development Of A Wireless Sensor Network To Understand And Monitor Environmental Variability In Precision Viticulture. <i>Alessandro Matese, Filippo Di Gennaro, Jacopo Primicerio, Lorenzo Genesio, Edoardo Fiorillo, Tiziana De Filippis, Leandro Rocchi, Francesco P. Vaccari.</i>	12
Poster Abstract: Smart AC Power Metering through WSNs and the Cloud. <i>Domenico Balsamo, Giacomo Paci, Davide Brunelli.</i>	14
Poster Abstract: Adaptive Wireless Sensor Network for Urban Crisis Management. <i>Milan Simek, Lubomir Mraz, Vladimir Cervenka, Vilem Pechanec.</i>	16
Poster Abstract: A Wireless Sensor Network Architecture for Solid Waste Management. <i>Sauro Longhi, Davide Marzioni, Emanuele Alidori, Gianluca Di Buó, Mariantonio Prist, Massimo Grisostomi, Matteo Pirro.</i>	18
Poster Abstract: Surveillance Application using Cooperative Robots and Sensor Networks: Challenges and Solutions. <i>Anis Koubâa, Yasir Kayani, Sahar Trigui, Imen Chaâri, Maissa Ben Jamâa, Olfa Gaddour, Rihab Chaâri, Hachemi Bennaceur, Miled Tezeghdanti, Khaled Al-Shalfan, Mohamed Abid.</i>	20
Programming	22
Demo Abstract: Enabling Transparent WSN Resource Access via RESTful Web Services. <i>Walter Colitti, Niccoló De Caro, Jelmer Tiete, Ha Phung, Kris Steenhaut, Abdellah Touhafi.</i>	23

Demo Abstract: From Business Process Specifications to Sensor Network Deployments. <i>Fabio Casati, Florian Daniel, Guenadi Dantchev, Joakim Eriksson, Niclas Finne, Stamatis Karnouskos, Patricio Moreno Montero, Luca Mottola, Felix J. Oppermann, Gian Pietro Picco, Antonio Quartulli, Kay Römer, Patrik Spieß, Stefano Tranquillini, Thiemo Voigt.</i>	25
Poster Abstract: WSN-Erlang: a Functional, High Level Approach to WSN Development. <i>Alessandro Sivieri, Gianpaolo Cugola.</i>	27
Poster Abstract: Compiler-Assisted Thread Abstractions for Resource-Constrained Systems. <i>Alexander Bernauer, Kay Römer.</i>	29
Poster Abstract: SEAL: An Easy-to-use Sensor Node Application Development System. <i>Atis Elsts, Janis Judvaitis, Leo Selavo.</i>	31
Protocols	33
Demo Abstract: GinLITE — A MAC Protocol for Real-Time Sensor Networks. <i>James Brown, Utz Roedig.</i>	34
Poster Abstract: Agreement for Wireless Sensor Networks under External Interference. <i>Carlo Alberto Boano, Kay Römer, Thiemo Voigt, Marco Antonio Zúñiga.</i>	36
Poster Abstract: Distributed Protocol Stacks for Wireless Sensor Networks. <i>Peter Rothenpieler.</i>	38
Poster Abstract: A Framework for a Modal Wireless Sensor Network. <i>Paulo Martins, Ronaldo Menezes, Ieda Hidalgo, Udo Fritzke Jr.</i>	40
Poster Abstract: Sensor Data Collection Using Constructive Interference Flooding. <i>Chao Gao, Makoto Suzuki, Takuto Kuroiwa, Hiroyuki Morikawa.</i>	42
Poster Abstract: Coordination For TDMA Operation In WSNs: Comparison Between Centralized And Distributed Mechanisms. <i>Antonio Vittorioso, Dujdow Buranapanichkit, Giancarlo Fortino, Yiannis Andreopoulos.</i>	44
Poster Abstract: Information Quality Aware Transport for Wireless Sensor Networks. <i>Vinay Sachidananda, Abdelmajid Khelil, Neeraj Suri.</i>	46
Poster Abstract: Packet Analyser for IEEE 802.15.4 Networks. <i>Lubomir Mraz, Vladimir Cervenka, Milan Simek.</i>	48
Poster Abstract: Effective Capacity Model in the Discrete Time Domain. <i>Yu Chen, Izzat Darwazeh.</i>	50
Localization	52
Demo Abstract: RSS-based Localization in Sensor Networks Does Not Need Pre-Deployment Profiling. <i>Maissa Ben Jamâa, Anis Koubâa.</i>	53

Poster Abstract: Low cost sensor design for non-cooperative geolocation via RSS. <i>Michael Butler, Richard Martin, Russell Lenahan.</i>	55
Poster Abstract: Localization based on Reflected Signals in Wireless Sensor Networks. <i>Kaushik Mondal, Partha Sarathi Mandal, Bhabani Sinha.</i>	57
Testbeds	59
Demo Abstract: MOTEI— A Mobile Robotic-Assisted Wireless Sensor Networks Testbed. <i>Alexander Förster, Anna Förster, Kamini Garg, Daniele Puccinelli, Silvia Giordano, Luca Gambardella.</i>	60
Demo Abstract: Testbed-Independent Experiment Specification and Execution Using the COTEFE Platform. <i>Claudio Donzelli, Vlado Handziski, Adam Wolisz.</i>	62
Demo Abstract: CoojaTrace, Extensive Profiling for WSNs. <i>Moritz Strübe, Florian Lukas, Rüdiger Kapitza.</i>	64
Poster Abstract: TUDμNet, a Metropolitan-Scale Federation of Wireless Sensor Network Testbeds. <i>Pablo Guerrero, Alejandro Buchmann, Abdelmajid Khelil, Kristof Van Laerhoven.</i>	66
Security	68
Demo Abstract: On preventing GTS-based Denial of Service in IEEE 802.15.4. <i>Roberta Daidone, Gianluca Dini, Marco Tiloca.</i>	69
Poster Abstract: DoS Detection with Markov Chains. <i>Denise Dudek.</i>	71
Poster Abstract: Reusing AES Coprocessor in Public Key Cryptography. <i>Vladimir Cervenka, Lubomir Mraz, Milan Simek.</i>	73
Poster Abstract: Topology and Deployment Impact on Key Distribution in Wireless Sensor Networks. <i>Bruno Trevizan de Oliveira, Cíntia Borges Margi, Wilson Vicente Ruggiero.</i>	75
Poster Abstract: Wormhole Detection with Location Information in Wireless Ad-Hoc Networks. <i>Jian-Hua Xiao, Takashi Minohara, Seikoh Nishita.</i>	77
Hardware and Energy	79
Demo Abstract: Evaluating Energy-Efficiency of Hardware-based Security Mechanisms. <i>Christian Haas, Anton Hergenröder, Joachim Wilke, Thomas Wiskot, Markus Niedermann.</i>	80
Poster Abstract: Energy Assessment in Praxis. <i>Christian Renner, Florian Meier, Volker Turau.</i>	82
Poster Abstract: Energy-Harvesting Wireless Sensor Networks. <i>Xenofon Fafoutis, Dusan Vuckovic, Alessio Di Mauro, Nicola Dragoni, Jan Madsen.</i>	84

Poster Abstract: Endless Smart Power for WSNs: Combining Multiple Harvesting and Fuel Cells. <i>Michele Magno, Danilo Porcarelli, Davide Brunelli, Luca Benini</i>	86
Poster Abstract: Wake-up architecture for Wireless sensor nodes based on ultra low power FPGA. <i>Victor Rosello, Jorge Portilla, Teresa Riesgo</i>	88
Industrial Demonstrations	90
Industrial Demo Abstract: custom design and prototypes of WSN devices. <i>Michele Corrá, Bruno Dalvit, Emiliano Fusari</i>	91
Industrial Demo Abstract: MyriaNed: A biology inspired self-organizing, gossiping Wireless Sensor Network. <i>Lex van Gijsel, Bob Peters, Zeno Korsmit</i>	93
Industrial Demo Abstract: Industrial demo of two Office Applications based on WSN-SI and MyriaNed. <i>Joost van Velzen, Herman Tuininga</i>	95
Industrial Demo Abstract: Energy Monitoring. <i>Manuel Fernández, Juan Pablo Viñuela</i>	97
Author Index	99

Applications

Demo Abstract: *IMAC*, Enabling Flexible Configuration and Result Analysis for Diverse Wireless Sensor Network Experiments

Richard Figura, Sascha Jungen, Ramin Soleymani, Chia-Yen Shih, Pedro José Marrón

Networked Embedded Systems Group

University of Duisburg-Essen, Germany

{richard.figura, ramin.soleymani-fard, chia-yen.shih, pjmarron}@uni-due.de, sascha.jungen@stud.uni-due.de

Abstract—Field experiments are very important in the process of Wireless Sensor Network (WSN) application development and performance evaluation. Typically, the WSN experiments are conducted with various configurations, and different experiments may be required by a WSN application with several WSN deployments. Effective management of the field experiments is the key to the success of WSN application development. In this demo, we present an experiment management tool, *IMAC*, which supports flexible pre-experiment application configuration, runtime experiment data monitoring/controlling and post-experiment result analysis for field experiments.

I. INTRODUCTION

With increasingly advanced *Microelectromechanical systems* (MEMS) and wireless communication technology, Wireless Sensor Networks (WSNs) offer a pragmatic option for acquiring physical parameter measurements. Thus, deployment of WSNs has been widely considered in a variety of application domains such as habitat monitoring, surveillance, industrial automation, etc. During the development of WSN applications, empirical experiments are often conducted to evaluate the performance of the application under various configurations. Effective management of configurable experiments is a decisive key to the success of the application development, especially when a complex application requires several WSN deployments each of which runs a different software program. Handling such combinations of empirical experiments, an effective management tool must allow flexible configuration for different runs of various experiments. Moreover, a management tool that offers on-site experiment monitoring and control during the runtime as well as post-experiment result analysis can greatly shorten the empirical experiment process and thus speed up the application development.

Our work was motivated during the preparation of experiments in an EU project, PLANET[1], for a wildlife monitoring application in *Doñana Biological Reserve* (DBR)[2] in Spain. The application scenario involves Cooperating Object[3] deployments that consist of a set of ground sensor nodes and unmanned aerial and ground vehicles. We needed to conduct different experiments for static sensor network monitoring and the communication between the mobile vehicles and static sensors. To manage different experiments, we developed a tool,



Fig. 1: Field Experiments in Doñana Biological Reserve

IMAC (abbr.: Installation/configuration, Monitoring, result Analysis and Control), which offers flexible configuration and allows managing numerous experiment runs with various configurable parameters for different experiments.

In general, application development and performance evaluation involve an iterative procedure of running experiments with configurable parameters. The procedure can be divided into three phases: *pre-experiment*, *execution* and *post-experiment* phases. In this demo, we present the versatility and flexibility of our IMAC in supporting the iterative field experiment procedure in aforementioned three phases. Note that the current version only supports TinyOS[4] applications. However, it can be easily extended to support other applications, e.g., Contiki[5] ones.

II. *IMAC*

IMAC aims to enable flexible field experiment management in all three experiment phases. The basic functionality of IMAC is to collect the experiment data and to manage the data based on the performed experiments. In addition, other main services provided by IMAC include: pre-experiment application installation and configuration, runtime experiment monitoring/controlling and the post-experiment result analysis (illustrated in Figure 2). These services are detailed as follows.

A. Pre-experiment Installation and Configuration Service

During the pre-experiment phase, the most tedious task is to install the application on and to assign the node ID to the nodes, especially when the deployment involves an enormous number of nodes. Moreover, in certain cases where the inexpensive hardware (e.g., radio chips) can impose a great impact on the experiment result, it is often preferable to assign the fixed ID to the same node. This makes the installation and configuration task even more error-prone and time consuming.

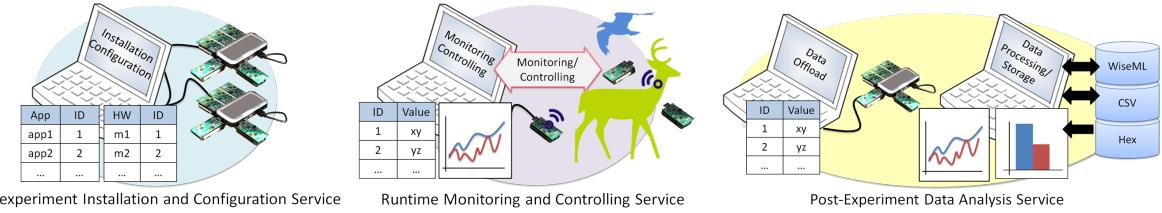


Fig. 2: Experiment Management Services Provided by IMAC

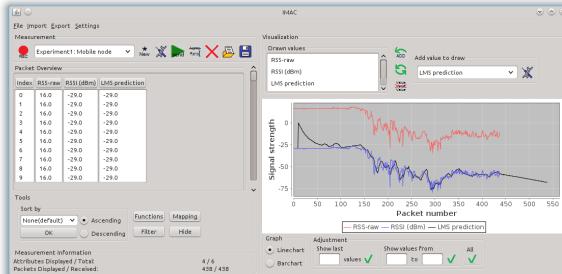


Fig. 3: IMAC, Displaying the Experiment Data in Real-Time

IMAC can achieve significant time reduction by only requiring two configuration files specified by the user. The first file associates the software mote ID to the TinyOS application, while the second one specifies the association between the software mote ID and hardware addresses. The installation process can thus be automated and the sensor nodes can even be installed in parallel on several machines with various intended applications using node IDs that remain fixed across all repeated experiment runs.

With the assistance of IMAC, we were able to conduct various planned experiments using over 90 nodes. During three days, we collected almost 5 millions packets using IMAC.

B. Runtime Monitoring and Controlling Services

IMAC offers two services for enabling runtime experiment monitoring and controlling. For these, the user needs to configure IMAC in the pre-experiment phase with the header files, which include the message structures used by the application.

The experiment monitoring service allows visualizing and processing collected TinyOS packets transmitted by the application (illustrated in Figure 3). During the experiment execution, experiment data is often gathered at a sink through the radio. IMAC displays the containing values at runtime, allowing real-time experiment monitoring. The user can also freely specify the data field in the messages for visualization. Furthermore, the converted data can then be filtered and processed on-line using customized functions, which are described in section II-C.

In addition to experiment monitoring, IMAC also provides runtime controlling service, which allows injecting control messages into the network in order to adjust experiment configuration. The control messages can be defined either in the pre-experiment phase or at runtime. In the case that the controlling process involves introducing a series of control messages, IMAC provides a simple scripting language allowing specification of a batch of messages.

C. Post-experiment Data Analysis Service

The process of post-experiment data analysis usually undergoes the steps of downloading the experiment data, processing the data, visualizing the data and storing the analysis result.

For downloading the experiment data, IMAC can automate the process of reading the data flash and erasing the flash (in case the node is needed for another experiment) by the user specifying the structure of the flash data and the volume-partitions used on the flash at the pre-experiment phase.

To process the raw data, IMAC allows converting and normalizing the data (e.g., the raw RSSI bits to data in dBm) by providing a set of transformation functions. To filter and process the converted data, IMAC offers a modular approach by allowing the user to define filtering and processing functions, which can be applied in sequence allowing complex data processing. The functionality of IMAC can be easily extended by custom functions, which can be added as plug-ins into the program.

To support visualization for data analysis, IMAC is integrated with JFreeChart Library[6] to generate line graphs and bar charts for displaying raw, converted or processed data. Last but not least, IMAC also provides data storage management allowing to store the various experiment results for later comparison or analysis. IMAC supports several import and export formats including CSV and WiseML[7]. In addition, IMAC can import hex-based traces and convert them into human readable CSV for storage.

III. CONCLUSION

With IMAC, we tackle the efficient management of experiments, which is crucial for fast application development. In the future, we would like to enhance several aspects of IMAC including support for applications not based on TinyOS, conditional control messages, etc.

Acknowledgements. This work has been partially supported by EU PLANET, *PLAtform for the deployment and operation of heterogeneous NETworked cooperating objects*, and CONET, *the Cooperating Objects Network of Excellence* (PLANET: FP7-2009-5-257649, CONET: FP7-2007-2-224053).

REFERENCES

- [1] Planet. [Online]. Available: www.planet-ict.eu
- [2] Doñana biological reserve. [Online]. Available: <http://www.ebd.csic.es>
- [3] P. J. Marrón, S. Karnouskos, D. Minder, and A. Ollero, Eds., *The Emerging Domain of Cooperating Objects*. Berlin, Heidelberg: Springer, 2011, ISBN 978-3-642-16945-8.
- [4] Tinyos. [Online]. Available: <http://www.tinyos.net/>
- [5] Contiki. [Online]. Available: <http://www.contiki-os.org>
- [6] JFreeChart. [Online]. Available: <http://www.jfree.org/jfreechart/>
- [7] WiseML. [Online]. Available: <http://wisebed.eu>

Demo Abstract: Wireless Sensor Networks for the Real World: A Collage of Experiences in Trento

Matteo Ceriotti[†], Ramona Marfievici[‡], Davide Molteni[‡], Amy L. Murphy[†], Gian Pietro Picco[‡]
Fondazione Bruno Kessler, Trento, Italy[†]
University of Trento, Italy[‡]

Abstract—Wireless sensor networks can be exploited in a variety of settings due to their flexibility. We present our experiences in four domains in which we collaborated with engineers, biologists, and medical professionals. We also outline several elements of our custom toolset, developed and applied in these projects.

I. INTRODUCTION

Real world deployments of Wireless Sensor Networks (WSNs) present a myriad of opportunities, but also an equally diverse set of challenges. Here we outline four recent deployment experiences in Trento, addressing the objectives and challenges. We then outline some of the supporting tools we have developed to overcome the challenges. Our demonstration will offer a glimpse into these projects.

II. DEPLOYMENTS

A. Torre Aquila: Structural health

Torre Aquila, located in Trento (Italy) is a 31 meter-tall medieval tower whose second floor contains “Il ciclo dei mesi” (“The Cycle of the Months”), a series of internationally-renowned frescoes that represent a unique example of non-religious medieval painting. Preservation of the frescoes is a source of concern for the local conservation board as the modern state of the city forces the consideration of a road tunnel to bypass the castle compound. Construction has long been delayed due to fear that the work might cause unwanted settling of the tower foundation. Estimation of the potential risk to the frescoes requires real-time monitoring and appropriate response models to reproduce the structural behavior of the tower.

In collaboration with a group of civil engineers, our task was to design a monitoring infrastructure to measure deformation, environmental parameters, and vibrations for a time span of months or years [1]. Our contributions ranged from the hardware to the graphical front-end. Customized hardware deals efficiently with high-volume vibration data, and specially-designed sensors acquire the building’s deformation.



Fig. 1. Torre Aquila.

Dedicated software services provide: *i*) data collection, to efficiently reconcile the diverse data rates and reliability needs of heterogeneous sensors; *ii*) data dissemination, to spread configuration changes and enable remote tasking; *iii*) time synchronization, with low memory demands. The system ran for about two years, collecting data useful for the civil engineers to assess the health of the tower structure.

B. TRITon: Road tunnel

State-of-the-art solutions for road tunnel lighting either use pre-set light levels based on date and time, or adjust the lights based on an open-loop regulator relying on an external sensor. Both solutions disregard the actual lighting conditions inside the tunnel, and may endanger drivers or consume more power than needed. The solution developed within the TRITon ([Trentino Research & Innovation for Tunnel Monitoring](#), triton.disi.unitn.it) project deployed a WSN along the tunnel walls to measure the light intensity and report it to a controller, which closes the loop by setting the lamps to match the lighting levels mandated by law [2]. Unlike conventional solutions, our system adapts to fine-grained light variations, both in space and time, and dynamically and optimally maintains the legislated light levels. This enables energy savings at the tunnel extremities, where sunlight enters, but it is also useful inside the tunnel to ensure the target light levels even when lamps burn out or are obscured by dirt.



Fig. 2. Tunnel deployment site.

The system was developed with the goal of reducing the management costs of road tunnels and improving their safety. Our WSN-based control system has been installed in an *operational* tunnel on a high-traffic freeway, where it has been running for more than a year without any required intervention. One of the key contributions of this work is the tight integration with the industrial control system.

C. Wildlife monitoring

In collaboration with local biologists, we are exploring the uses of WSNs for understanding the behavior of roe deer. The biologists have fundamental questions about the movement and social interactions of these solitary animals and answers

will help to analyze the spread of diseases among the animals as well as the impact of human development.

The core idea involves both small, fixed deployments in locations where the animals are known to move in combination with nodes carried by the animals in specially designed collars. During the course of the deployment we will track encounters among the deer and with the fixed infrastructure, as well as the locations of all encounters. To date, we have performed several test deployments with domestic animals such as sheep and cows in order to evaluate the system behavior.

D. ACube: Ambient Assisted Living

The goal of ACube (acube.fbk.eu) is to improve the quality of life for elderly and disabled persons through technology. Multiple technologies including video, audio, and WSNs are combined to form a distributed infrastructure to offer care givers information about the patients and their care. As a concrete example, the WSN is used to detect the proximity of patients to hazards such as exits in an Alzheimer's day care facility. By tagging the exit, and requesting each patient to carry a device, the WSN raises an alarm when the two nodes are in proximity. Other sensors, e.g., PIR sensors, connected to WSN nodes, are used in situations when patients are not expected to carry a nodes, e.g., in the bathroom, to detect falls.

III. TOOLS

To support the deployments above, we have developed a set of tools outlined below.

A. TeenyLIME

Programming WSNs is a difficult task with extensive effort spent to develop functionalities related to sharing information among local and distributed components. As the common approach is to implement system services directly on top of the operating system, the developer must implement proper interfaces among components and handle the sharing of information at the level of packets and variables. We took a different approach by designing a programming abstraction, TeenyLIME [3], that provides the programmer with a data sharing model spanning neighboring nodes. We tested its effectiveness in the aforementioned deployments: the routing service, employed in both Torre Aquila and TRITon, required half the lines of code taken by a comparable TinyOS implementation; in ACube, TeenyLIME enabled independent development of the system services, as well as memory savings, by grouping both access to shared information and common communication primitives behind the same interface.



Fig. 3. Temporary sheep deployment.

B. TRIDENT

Assessing the connectivity of WSNs in the environment in which they are deployed is crucial to developing reliable system services and understanding their behavior. TRIDENT is our tool to measure communication with an untethered infrastructure [4]. By automating the collection of raw packet statistics such as RSSI, LQI, and PRR with nodes that are freely placed and moved, TRIDENT allows users to observe low level characteristics of a network, giving insight into its behavior. TRIDENT has been used by the biologists themselves in a cloud forest in Ecuador [5]. We have also used it in local mountains and forests to provide foundational information for our other work with biologists.

C. RUTH

Neighbor discovery, or knowing the identity of nodes in communication range, is a fundamental problem in mobile WSNs. Discovery *latency* and system *lifetime* are two key aspects along which this problem can be formulated. RUTH addresses these, allowing an optimal maximum lifetime to be achieved for latency-constrained scenarios and, dually, trading discovery latency to meet a strict lifetime requirement. By offering a choice between *deterministic* and *probabilistic* guarantees on discovery latency, RUTH gives developers another knob for optimizing their applications. RUTH has been used in for proximity detection in both the wildlife and ACube settings.

ACKNOWLEDGMENT

The authors wish to thank everyone who has participated in the deployments and tools presented here, especially Štefan Gună, Matteo Chini, and Luca Mottola.

This work was made possible by our participation in a number of funded projects: the Province of Trentino projects TRITon and ACube; the EU Cooperating Objects Network of Excellence (CONET—FP7-2007-2-224053); and the Italian Ministry of Education (MIUR) project PRIN06-2006084179_003.

REFERENCES

- [1] M. Ceriotti, L. Mottola, G. P. Picco, A. L. Murphy, S. Guna, M. Corra, M. Pozzi, D. Zonta, and P. Zanon, "Monitoring heritage buildings with wireless sensor networks: The torre aquila deployment," in *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2009.
- [2] M. Ceriotti, M. Corra, L. D'Orazio, R. Doriguzzi, D. Faccin, S. Guna, G. P. Jesi, R. Lo Cigno, L. Mottola, A. L. Murphy, M. Pescall, G. P. Picco, D. Pregnolato, and C. Torghele, "Is there light at the ends of the tunnel? wireless sensor networks for adaptive lighting in road tunnels," in *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2011.
- [3] D3S Research Group, teenylime.sourceforge.net.
- [4] M. Chini, M. Ceriotti, R. Marfievici, A. Murphy, and G. Picco, "Demo: TRIDENT, Untethered observation of physical communication made to share," in *Proc. of the Int. Conf. on Embedded Networked Sensor Systems (SenSys)*, November 2011.
- [5] M. Ceriotti, M. Chini, A. L. Murphy, G. P. Picco, F. Cagnacci, and B. Tolhurst, "Motes in the jungle: lessons learned from a short-term wsn deployment in the ecuador cloud forest," in *Proc. of the Int. Wkshp. on Real-World Wireless Sensor Networks (REALWSN)*, 2010.

Demo Abstract: Route Selection of Mobile Sensors for Air Quality Monitoring

Olga Saukh, David Hasenfratz, Abouzar Noori, Tamara Ulrich, and Lothar Thiele

Computer Engineering and Networks Laboratory

ETH Zurich, Switzerland

{saukh, hasenfratz, ulrich, thiele}@tik.ee.ethz.ch, anoori@student.ethz.ch

Abstract—Monitoring air pollution in cities requires installation of a huge number of static sensors which is both cost and labor intensive. For this reason, we consider the installation of mobile sensors on top of public transport vehicles. This leads to the problem of selecting a subnetwork of a city’s public transport network to achieve a good coverage in the area. Since failures and signal drift over time are typical for low-cost sensors, the selected vehicles have to occasionally be in each others vicinity to allow sensor recalibration. If the vehicles can pass by reference stations, the recalibration quality can be further improved. In this demonstration, we illustrate different solutions to the problem of subnetwork selection based on the tram network of Zurich and visualize the results in Google Maps and Google Earth.

I. AIR QUALITY MONITORING IN A CITY

Air pollution has risen to be the most pressing matter of environmental policy in modern urban areas. High concentration of traffic and industrial facilities heavily impact ecological sustainability and quality of living in the area. Monitoring air pollution has been addressed at many levels, but mostly through legislative decisions and standards. Due to high cost, weight, and size of traditional air pollution measurement instruments, there are still no *detailed* maps of air pollutant distributions in cities. To address this deficiency, we work on combining low-cost gas sensors, that become available on the market in the last several years, with wireless sensor networks for air pollution monitoring applications. In particular, we install sensing stations on top of several public transport vehicles to achieve better coverage than provided by statically deployed measurement stations. Public transport networks form an attractive backbone for performing periodic measurements due to a large number of spatially spread predefined routes, a fixed timetable, and a usually good operability.

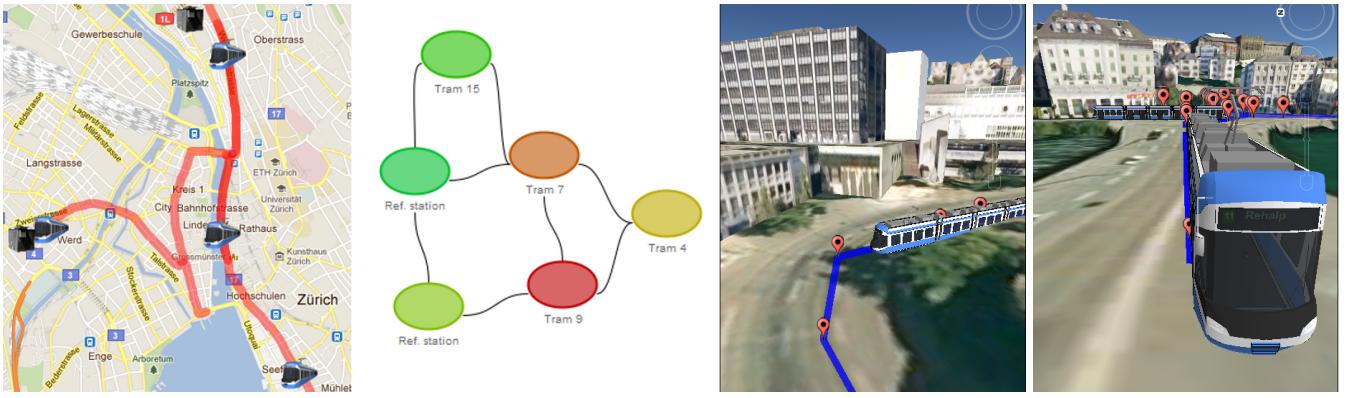
The problem we face is, given a route plan and a timetable, how to choose a subnetwork of the public transport network to provide a good coverage of the city with a limited number of sensors. Additionally, since our measurement stations are equipped with low-cost gas sensors, the final subset of selected vehicles has to allow comparing sensor readings of different sensors, *i.e.*, we require that different sensors occasionally take measurements at the same time and location. We refer to this problem as *sensor checkpointing*. This allows detecting sensor failures and provides the necessary support for sensor recalibration, possibly over multiple hops [3]. Sensor checkpointing naturally implies the simultaneous presence of the corresponding mobile vehicles at the same place.

We analyze the coverage problem with and without sensor checkpointing based on the pollution monitoring scenario in the city of Zurich, Switzerland as part of the OpenSense project [1]. The long-term goal of OpenSense is to raise community interest in air pollution data and to encourage involvement of the public using enhanced cell phones or pocket sensors. We hope to foster community interest and involvement in data gathering by establishing an initial coverage of the city with a network of ten measurement stations installed on top of trams in Zurich. The goal of this demonstration is to visualize the solution to the problem under various parameter settings including the number of deployed mobile stations, the spatial and temporal checkpointing constraints, and the locations of the reference stations. The solution is demonstrated in Google Maps and a simulation drive with a selected tram in Google Earth.

II. SENSOR CHECKPOINTING

We consider an area of interest to be the spatial spread of a city and a time interval. A mobile vehicle with a sensing station installed on top of it is moving along a spatiotemporal curve. It can take measurements everywhere during its movement with no restrictions. A measurement consists of a location and a timestamp, and has no duration. A measurement is valid in a certain area and for a certain time. The validity of measurements taken by a mobile vehicle gradually decreases towards zero with the distance from its spatiotemporal movement curve. A checkpoint occurs when the validities of two measurements performed by different vehicles are above a certain threshold, *i.e.*, if the spatial and temporal distance between them is below a certain threshold, which is a parameter of the route selection algorithm. In this demonstration we assume that a uniform coverage of the city is the goal of the route selection algorithm. One of the uniformity metrics presented in [2] can be used as coverage measure. Given checkpoints between all pairs of mobile vehicles, it is possible to construct a *checkpoint graph*, where the set of nodes corresponds to the set of mobile vehicles equipped with sensing stations. There is an edge between any two vehicles if they make at least one checkpoint, see Fig. 1(b).

To enable sensor checkpointing we need to ensure that the checkpoint graph is connected. We refer to this type of checkpointing as X-checkpointing (*cross checkpointing*). X-checkpointing facilitates pairwise tests among low-cost sen-



(a) Selected trams and their positions

(b) Checkpoint graph

(c) Making a turn at Voltastr., Zurich

(d) Checkpoint at Central, Zurich

Fig. 1. Given two reference stations, four trams of the Zurich tram network are selected for the installation of measurement stations. Visualization with Google Maps and Google Earth. Cobra tram model from the Sketchup 3D warehouse, <http://sketchup.google.com/>.

sors which are essential for being able to identify sensor faults and sensor recalibration needs.

The quality of checkpointing can be further improved if the sensors are regularly synchronize with a set of reference stations. Reference stations can be static or mobile and are usually capable of performing high-quality sensing. We assume that all reference stations reflect the ground truth and can be used to calibrate low-cost sensors from time to time over one or several hops, *e.g.*, using the calibration approaches described in [3]. This problem is referred to as R-checkpointing (*reference checkpointing*). Many big cities have a very sparse network of highly precise fixed stations, which can be used as references. For example in Zurich, Switzerland, there is one station of the national air quality monitoring network NABEL¹ and four smaller stations are part of the cantonal measurement network OstLuft². The availability of this infrastructure allows to considerably improve the system reliability by selecting a timetable subnetwork which fulfills R-checkpointing constraints.

III. ALGORITHM AND DEMONSTRATION SETUP

The problem of route selection involves high computational complexity even for moderate-sized cities. For this reason, we use an evolutionary algorithm to solve the problem. A chromosome is a subnetwork of size k . The fitness function is the coverage of the city provided by the subnetwork. The calculation of the coverage involves discretization of the region with Monte Carlo sampling and computation of the level of coverage at sampled points. A chromosome satisfies the X-checkpointing constraint if the checkpoint graph is connected. To test a chromosome for R-checkpointing, we extend the checkpoint graph with the set of reference stations as additional nodes connected among themselves and check the extended graph for connectivity. Recall that R-checkpointing requires that each mobile vehicle is connected to a reference station, possibly over multiple hops. We use random crossover and mutation operators to generate new offsprings.

Zurich is one of the target cities in the OpenSense project to deploy a network of mobile sensors on top of public transport vehicles. The first out of ten nodes is already installed on top of a tram³. We run the algorithm on up-to-date data of the Zurich tram network. The track plan is obtained from OpenStreetMap⁴ and the timetables from the ZVV information service⁵. The Zurich tram network comprises 13 tram lines with the involvement of maximal 260 individual trams. All demonstration runs of the algorithm are based on the tram timetable of a work day. The speed of the tram between two stations is linearly interpolated. Among others, the following input parameters can be adjusted as part of the demonstration:

- The number of *measurement stations* k .
- Evolutionary algorithm parameters: *population size, recombination and mutation rates*.
- *Locations of the reference stations* within the city.
- *Checkpointing constraints*. Relaxation of temporal and spatial checkpointing constraints generally results in a greater coverage achieved by the algorithm.

With this demonstration we aim to tackle the difficulty of visual representation of a problem solution that involves several movement paths in a three dimensional space. This problem is found when solving planning and scheduling problems on top of a timetable network. As part of this demonstration, we simulate a drive through the city on Google Earth with a selected tram and visualize checkpoints during the drive.

REFERENCES

- [1] K. Aberer, S. Sathe, D. Chakraborty, A. Martinoli, G. Barrenetxea, B. Faltings, and L. Thiele. Opensense: open community driven sensing of environment. In *ACM IWGS*, 2010.
- [2] M. Gunzburger and J. Burkardt. Uniformity measures for point sample in hypercubes. 2004.
- [3] David Hasenfratz, Olga Saukh, and Lothar Thiele. On-the-fly calibration of low-cost gas sensors. In *EWSN*, 2012.

Acknowledgements: This work was funded by NanoTera.ch with Swiss Confederation financing. The authors thank Marco Zimmerling and Robert Sauter for valuable input.

³OpenSense data online access, data.opensense.ethz.ch

⁴OpenStreetMap, www.openstreetmap.org

⁵Zürcher Verkehrsverbund, www.zvv.ch

¹Swiss National Air Pollution Monitoring Network, www.bafu.admin.ch

²Cantonal Network OstLuft, www.ostluft.ch

Poster Abstract: Development Of A Wireless Sensor Network To Understand And Monitor Environmental Variability In Precision Viticulture

A. Matese ⁽¹⁾, F. Di Gennaro⁽¹⁾, J. Primicerio⁽¹⁾, L. Genesio ⁽¹⁾, E. Fiorillo ⁽¹⁾, T. De Filippis ⁽¹⁾, L. Rocchi ⁽¹⁾ and F. P. Vaccari ⁽¹⁾

⁽¹⁾ National Research Council - Institute of Biometeorology (CNR-IBIMET), Via G. Caproni 8, 50145 Firenze (Italy)

Corresponding author: Alessandro Matese e-mail: a.matese@ibimet.cnr.it

Abstract—The application of wireless monitoring systems finds large use in Precision Viticulture (PV), in order to understand vineyard variability and to suggest appropriate management practices for improving wine quality. PV aims to optimize vineyard performance, in particular maximizing grape yield and quality and minimizing environmental impacts and risk. New and emerging technologies, such as a Wireless Sensor Network (WSN), provide an useful and efficient tool for remote and real-time monitoring of micro-meteorological parameters. In PV the WSN can be fundamental where the measurement of environment parameters is difficult to access, and when a multi-point monitoring system is necessary. In the present work a WSN system was developed, following an open-source approach, and tested in heterogeneous experimental vineyards of Tuscany in order to develop a low cost tool to monitor and characterize vineyard variability.

I. INTRODUCTION

THIS Few studies deal with WSN applied for Precision Agriculture (PA): some papers present an overview on the development of wireless sensor technologies and standards for wireless communications as applied to wireless sensors [2, 4, 8], others have focused on the importance of energy efficiency of WSN [5]. Papers have mainly reported the results of field experiments for real-time monitoring and control of important farming operations [1]; in particular, WSN designed for air temperature monitoring for frost/freeze protection [6], for scheduling irrigation [7] and for site-specific irrigation [3]. In the present work a WSN system was developed and tested in the complex environment of Tuscany, where the climatic, geomorphologic and pedologic characteristics are highly variable and affect the quality parameters.

II. MATERIALS AND METHODS

The system includes a base agrometeorological station (Base Unit) and a series of peripheral wireless nodes (motes) located in the vineyard (Fig. 1). The Base Unit is a typical single-point monitoring station and collects traditional agrometeorological data, placed outside the vineyard in a representative location; it utilizes a wireless technology for data communication and transmission with motes and with the remote central server.



Fig. 1. Peripheral wireless nodes (motes).

The motes are multiple nodes placed within the vineyard and equipped with micro-meteorological sensors for site specific environmental monitoring (Fig. 2), which are able to store and transmit data to Base Unit. A software was developed for setup and configuration functionality. A graphical user interface, operating on the remote central server, was implemented for data-collecting, data-processing and real-time display. The firmware has been developed under TinyOS and software for data acquisition and network management has been developed using nesC programming language and compiled creating a Java application.



Fig. 2. Air temperature sensor and solar radiation sensor.

III. RESULTS

Extended testing of the devices used was performed on hardware functionality, data acquisition, power consumption, radio and GSM/GPRS communication. Using data collected by each Base Unit, it was possible to obtain a detailed picture of the variability of weather data between the experimental vineyards. The data has been used to calculate the bioclimatic index used in the description of a terroir, which Winkler index (WI), Huglin index (HI), sum of daily temperature excursion (SET), Gladstones index, Ribéreau-Gayon-Peynaud index, thermal accumulation (Fig. 3). The motes multipuntual monitoring system, allows a micro-meteorological characterization to understand the vineyard internal variability, and to indentify differences between canopy management.

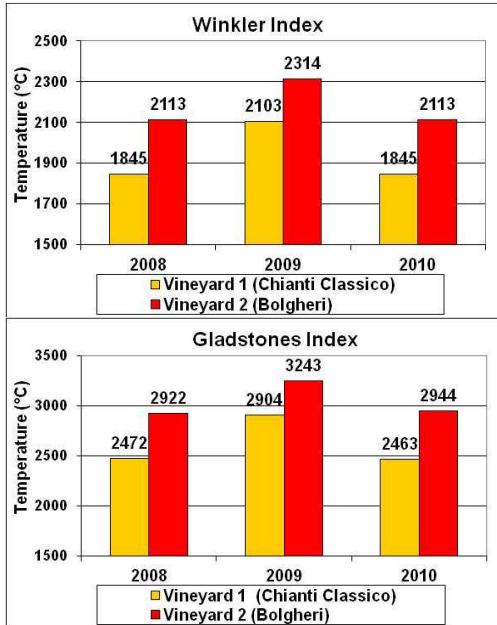


Fig. 3. Winkler Index and Gladstone Index calculated for two vineyards and for three years using meteorological data acquired by the system.

IV. CONCLUSION

The results presented show that WSN system is very useful in the vineyard for the study of soil and micro-meteorological

conditions variability and is characterized by flexibility of planning and installation. The system has shown interesting results in terms of climatic characterization of the vineyard.

REFERENCES

- [1] Beckwith, R., D. Teibel, e P. Bowen. 2004. Report from the field: Results from an agricultural wireless sensor network. p. 471–478. In *29th Annual IEEE International Conference on Local Computer Networks*, 2004.
- [2] Camilli, A., C.E. Cugnasca, A.M. Saraiva, A.R. Hirakawa, e P.L. Corrêa. 2007. From wireless sensors to field mapping: Anatomy of an application for precision agriculture. *Computers and Electronics in Agriculture*. 58(1): 25–36.
- [3] Kim, Y., e R.G. Evans. 2009. Software design for wireless sensor-based site-specific irrigation. *Computers and Electronics in Agriculture*. 66(2): 159–165.
- [4] Matese, A., S.F. Di Gennaro, A. Zaldei, L. Genesio, e F.P. Vaccari. 2009. A wireless sensor network for precision viticulture: The NAV system. *Computers and Electronics in Agriculture*. 69(1): 51–58.
- [5] Morais, R., M.A. Fernandes, S.G. Matos, C. Serôdio, P. Ferreira, e M. Reis. 2008. A ZigBee multi-powered wireless acquisition device for remote sensing applications in precision viticulture. *Computers and Electronics in Agriculture*. 62(2): 94–106.
- [6] Pierce, F.J., e T.V. Elliott. 2008. Regional and on-farm wireless sensor networks for agricultural systems in Eastern Washington. *Computers and Electronics in Agriculture*. 61(1): 32–43.
- [7] Vellidis, G., M. Tucker, C. Perry, C. Kvien, e C. Bednarz. 2008. A real-time wireless smart sensor array for scheduling irrigation. *Computers and Electronics in Agriculture*. 61(1): 44–50.
- [8] Wang, N., N. Zhang, e M. Wang. 2006. Wireless sensors in agriculture and food industry—Recent development and future perspective. *Computers and Electronics in Agriculture*. 50(1): 1–14.

Poster Abstract: Smart AC Power Metering through WSNs and the Cloud

Domenico Balsamo*, Giacomo Paci*, Davide Brunelli^o, Member, IEEE

* Department of Electronics, Computer Sciences and Systems (DEIS) - University of Bologna

^o Department of Computer and Information Science (DISI) - University of Trento

Abstract— In this paper we present the design of a Wireless Sensor Network (WSN) system for monitoring power consumption of AC appliances in different kind of buildings. The system consists of: individual measurement devices and network architecture which is open to cloud computing. Each node integrates a Jennic JN5148 module to provide power measurements and analysis.

I. INTRODUCTION

Energy saving in residential and commercial buildings is one of the key themes in sustainability strategies for the near future. The building sector is responsible for the 40% of the overall energy consumptions and it is divided in conditioning (heating, cooling and ventilation) and electrical (appliances) consumptions. Despite numerous energy efficiency policies, the electricity consumption continues to grow.

The aim of the project is to build a WSN system for real-time monitoring power consumption of AC appliances in different kind of buildings, so that end-user can understand their power consumption and reduce them. The system provides the following parameters:

- Real-time current evolution;
- RMS current and power consumption values;
- FFT analysis of current samples;

Information on power consumption is collected by sensors in a WSN based on standard IEEE802.15.4. We adopt this standard since it allows ZigBee protocol which defines specific profiles tailored on smart homes [2] and energy metering (i.e. ZigBee “Smart Energy” and ZigBee “Home Automation”).

This system requires an ultra-low power operation because each node uses two batteries, even if it can be also powered with energy harvesting units.

II. SYSTEM DESIGN

The photo of the sensor node with current sensor is shown in Fig.1. The node is built around Jennic JN5148 module, which is an ultra-low-power, high performance wireless microcontroller targeted at WSN applications. The device features an enhanced 32-bit RISC processor, 2.4GHz IEEE802.15.4 compliant transceiver, 128kB ROM, 128kB RAM, and a rich mix of analogue and digital peripherals.

We used commercial non-invasive AC current transformers (SCT-013-000 Model). These sensors have the benefit of non-intrusive measurement because they do not require any breaking of the mains wire and decouples the digital circuit from the high voltage input.

The main electrical specifications of AC current transformer are: maximum current 100A, output type current (33mA at 100A) and number of turns 1500. Thus

the AC current transformer produces a small secondary current that is 1500 times smaller than the current in the main wire.

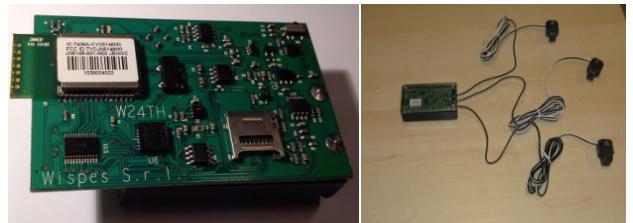


Fig. 1. Sensor node platform and the complete node with 3 current sensors

The current-to-voltage converter as shown in Fig 2. The current on the transformer is converted to voltage through a resistor R_{SENSOR} which is parallel to the current sensor and produces a voltage proportional to current. The minimum sensitivity (minimum value of current measured) and the maximum current value also depend on R_{SENSOR} , then this value must be carefully chosen.

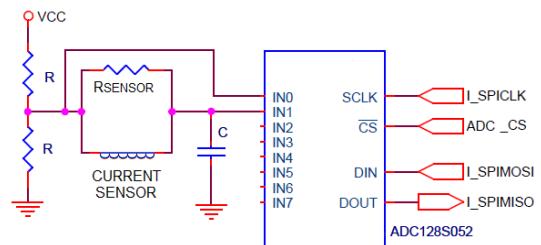


Fig. 2. Typical I-V circuit used to convert the output of the current sensor

For example, with a $R_{SENSOR}=560\Omega$ the current measured ranges from 4,3mA to 4.3A. A dedicated bias circuit and an ADC converter (ADC128S052) are used to perform the necessary analog to digital conversions. Current values are sampled with a frequency of 12.5KHz.

We have implemented a Raised Cosine Filter (RCF) to reduce the high frequency noise. A RCF is a low-pass filter which is commonly used for pulse shaping in data transmission systems. The frequency response $|H(f)|$ of a perfect raised cosine filter is symmetrical about 0 Hz, and is divided into three parts: it is constant in the pass-band, it sinks in a graceful cosine curve to zero through the transition region and it is zero outside the pass-band. The response of a real filter is an approximation to this behavior. The filter is designed as a Finite Impulse Response filter (FIR) ($f=12.5\text{KHz}$, corner frequency 500Hz, -6dB). We assume a constant RMS voltage in converting

current to power. This is acceptable for applications which monitor only Apparent Power. The power measured by the system ranges from 1W (230V*4,3mA) to 1KW (230V*4.3A).

We implemented also a Fast Fourier Transform (FFT) on the incoming current signal to separate the component frequencies, up to the 7th harmonic (7f). The Fourier Transform has the capability of taking functions from the time domain to the frequency domain. The implemented FFT algorithm on the node is simplified since the complex FFT can be replaced from the following real functions:

$$X_k = \operatorname{Re}(F_k) = \sum_{n=1}^N x_n \cos\left(\frac{2\pi kn}{N}\right); \quad k = 1, \dots, 7 \quad (1)$$

$$Y_k = \operatorname{Im}(F_k) = \sum_{n=1}^N x_n \sin\left(\frac{2\pi kn}{N}\right); \quad k = 1, \dots, 7 \quad (2)$$

Where k is the number of harmonics and N the number of samples in a single period. We obtain the spectrum amplitude from X_k and Y_k functions respectively:

$$A_k = \sqrt{X_k^2 + Y_k^2} \quad (3)$$

The network consists of several sensor nodes organized with IEEE802.15.4. As shown in the Fig. 3 the network consists of a coordinator that will interface with a service cloud [1] and sensor devices. The coordinator generates a beacon which wake-up end nodes and starts the ADC sampling, processing and data transmission.

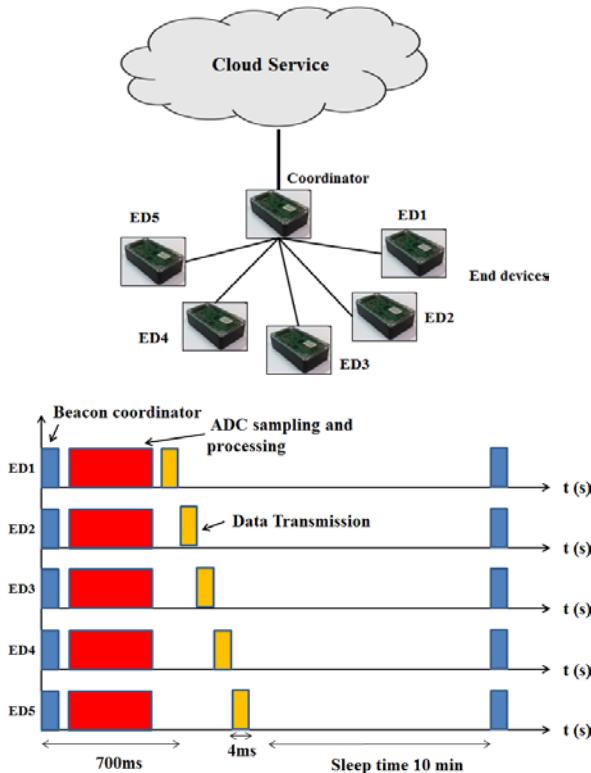


Fig. 3. WSN architecture and time requirements

III. EXPERIMENTAL RESULT

Our goal is to design a system that can perform a reliable current monitoring and operate for several years. Design of the node enables ultra low power consumption in sleep state. To reduce power consumption is possible to monitor the current periodically (e.g. every 10 minutes) and to put the nodes in deep-sleep mode during the remaining time.

When the node is ON, it samples the data of current for a limited number of periods (e.g. two) and elaborates them to get the real-time analysis, RMS value and harmonics of the current. The processing time is about 700ms. The consumption of the node in active state is about 12mA at 3.3V due to the microcontroller while in sleep mode it is about 8µA (ultra low power current consumption). In this mode of operation each node can operate for several years using the same battery.

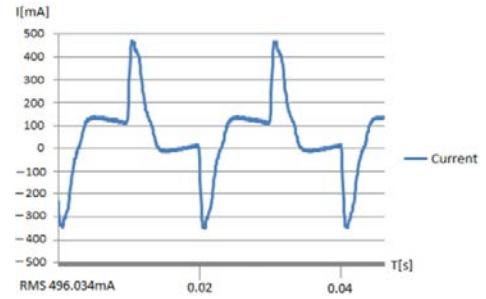


Fig. 4. Current evolution of an energy saving lamps measured by a sensor node

We tested the system with a typical application - current monitoring of energy saving lamp. Fig. 4 shows the current evolution of the lamp: x-axis displays the time (in seconds) and y-axis the current value (in mA).The RMS value calculated is equal to 496.034mA.

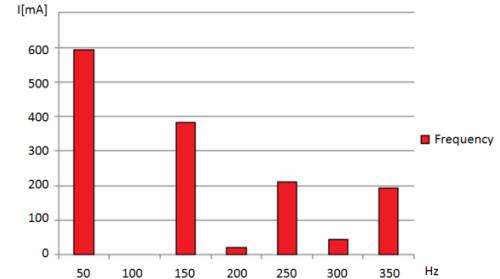


Fig. 5. Frequency analysis of AC consumption of an energy saving lamps

Fig.5 shows the frequency analysis of the load processed through FFT algorithm: x-axis displays the frequency (in Hz) and y-axis the current value of each component (in mA).

IV. CONCLUSION

We realized a distributed system which can interface with any cloud service. This system enables ultra low power consumption (down to 8µA) and it is adaptable to current standards for home energy efficiency.

V. ACKNOWLEDGMENT

The research leading to these results has been supported by the 3ENCULT Project (Grant agreement n. EeB.ENV. 260162) funded by the EU 7th Framework Programme.

REFERENCES

- [1] Yongquan Yang, Zhiqiang Wei, Dongning Jia, Yanping Cong, Ruobing Shan "A Cloud Architecture Based on Smart Home" Department of Computer Science, Ocean University of China
- [2] Ming Xu, Longhua Ma, Feng Xia, Tengkai Yuan, Jixin Qian, Meng Shao "Design and Implementation of a Wireless Sensor Network for SmartHomes" Department of Control Science and Engineering, Zhejiang University, China

Poster Abstract: Adaptive Wireless Sensor Network for Urban Crisis Management

Milan Simek, Lubomir Mraz, Vladimir Cervenka
Department of Telecommunications
Brno University of Technology, Czech Republic
Email: simek@feec.vutbr.cz

Vilem Pechanec
Department of Geoinformatics
Palacky University of Olomouc, Czech Republic
Email: vilem.pechanec@upol.cz

Abstract—The goal of this paper is to introduce the running national project referred to as AWSN (Adaptive Wireless Sensor Network for Crisis Management with Data Visualization - No.FR-TI2/571) and its continuous results. The state of research covers the investigation of nowadays snow sensors. Also the communication architecture standing on the Zigbee PRO stack is outlined together with the description of the Zigbee Sensor Unit already developed as the part of the project.

I. INTRODUCTION

Every year the Czech republic is stricken by regular snowstorms that create snow masses on roads and urban roofs. A ruined roofs under the heavy snow cover caused several disasters in recent years. The most affected are the wide flat roofs with the declination less than 30°. These roofs can be seen on the shopping centers, sport halls, cinemas etc. A photography from February 2010 (see Fig.1) shows the hall roof ruined under the mass of snow (no victims fortunately) [1].



Fig. 1: Ruined hall roof.

The main objective of the AWSN project launched in 2010 in cooperation with the 'Satturn Holesov spol. s.r.o.' company [2] is to develop the adaptive wireless sensor system controlling the snow load of the roofs of the crisis infrastructures and objects where people used to accumulate. The system will be deployed for the snow weight monitoring of the large roofs of the supermarkets, stadiums etc. The word 'Adaptive' in the title means that the wireless system should be easily applicable to another crisis scenarios such as water flood monitoring, air pollution monitoring around the factories etc. Issues that are covered within this paper include: i) Survey of sensors for monitoring of snow weight, ii) proposal of Zigbee communication architecture and iii) design of Zigbee Sensor

Unit. Further sections of this paper summarize the results of the introduced issues.

II. SURVEY ON SNOW SENSORS

To measure the amount of snow, two approaches are generally used: i) direct measurement of weight with a Snow Scale or Snow Pillow and ii) indirect measurement by using the Ultrasonic Snow Depth Sensor.

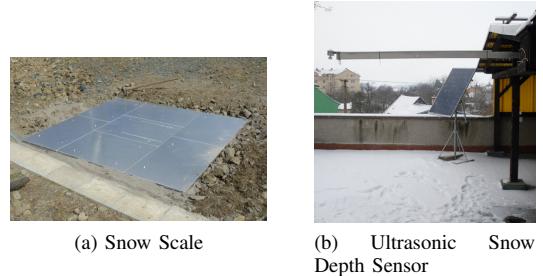


Fig. 2: Some methods of snow measurement

The Snow Scale (Fig.2a) determines the water-content of the snow covering. The working principle of the sensor is based on the detection of the hydrostatic pressure caused by the layer of snow. The large dimensions (standard size 3 x 3m) of the pillow prevent any bridging that might occur from having an effect on the measurement readings. However, the dimensions of the pillow together with its abnormal weight being more than 100 kg and with the high price that is more than 6000 EUR [3] makes this solution unsuitable for the project objectives. Ultrasonic Snow Depth Sensor (Fig.2b) is a solution for remotely measuring snow depth or water levels. The sensor works by measuring the time required for an ultrasonic pulse to travel to and from a target surface. As the part of the project flow, we have installed the ultrasonic sensor (Fig.2b) and evaluate its ease of use and mainly the accuracy of the measurement. Lesson learnt from application of the ultrasonic sensor showed that snow depth measured is highly dependent on the quality and type of snow and thus it is necessary to estimate this extra parameter for the achievement of the optimal results. From this reason we are working on the development of the own inexpensive sensor that is able to measure both parameters such as depth and type of snow

all at once and in very accurate manner while keeping current and price very low. Next advantage of novel sensor is physical parameters that are up to 30 cm of height.

III. COMMUNICATION ARCHITECTURE AND ZIGBEE SENSOR UNIT

The communication architecture of AWSN stands on the Atmel BitCloud - Zigbee PRO stack [4]. The full mesh architecture, which is illustrated in Fig.3 consists of the three essential communication units: i) Zigbee Sensor Unit (ZSU) is Zigbee router that collects measurement from the sensors equipped and sends them toward the gateways, ii) Zigbee Sensor Alarm (ZSA) provide the similar operation as the ZSU and in addition it is equipped with the visualization and sound alarm devices such as horn and light beacon, iii) Gateway (GTW) consists of ZSU and the external Converter Unit for the remote connection with the Crisis Management.

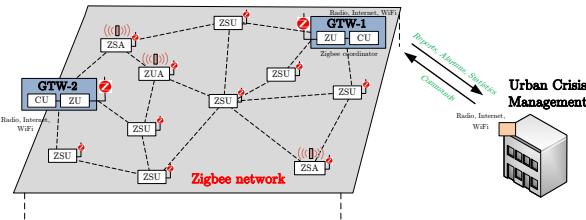


Fig. 3: AWSN communication architecture proposal

The messages such as Report, Alarm, Statistics and Commands are represented by the Zigbee APS endpoints. Therefore to assign the type and priority of the messages only thing that must be accomplished is to set the correct endpoint for the communication. ZSU save measured data and sends them in the regular intervals to the gateway in the form of the Report message. To increase the reliability of data delivery, we assume to deploy more gateways within the network, all with the same priority and function. ZSU unicasts Reports to all gateways and the Crisis Management server takes care of the message redundancy. Once ZSU measures value that overcomes the configured low or high thresholds or it is significantly different from the last measurement, ZSU immediately generates ALARM message having the highest priority of the proceeding in all the devices. During the alarm scenario the gateways firmware controls the alarms setting and its triggering. The optimal values of measuring and report intervals, thresholds and measuring tolerances are configured over-the-air by means of the Control messages sent from the crisis management server.

We have developed the ZSU device which is controlled by the Atmel Zibit 900 MHz module [5]. As it is illustrated in Fig. 4, ZSU offers three digital and four analog inputs, two digital outputs, one I₂C and two UART peripherals. The Zibit module provide 128 kB of Flash memory for firmware, 8 kB of RAM for data and 4 kB of EEPROM for non-volatile parameters storing. The memory for stored measured data can be expanded by the external FRAM. According to

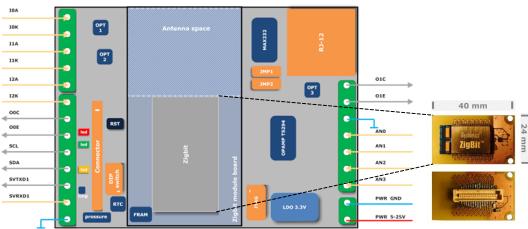


Fig. 4: Layout of Zigbee Sensor Unit

the parameters of the ambient radio environment, the Zibit module can be easily replugged to communicate via the more suitable IEEE 802.15.4 frequency band such as 2.4 GHz or 916/868 MHz. We have already published the results of the measurement of the wireless technology coexistence, see [6]. The Zigbee testbed used for the evaluation is deployed at the Department of Telecommunications and consists of the 28s ZSU and two gateways connected with the data server through the Internet.

IV. CONCLUSION AND FUTURE WORK

In this paper we have presented the research project AWSN that aims to develop the adaptive wireless sensor network for snow weight monitoring of the large plane roofs. Our investigation of the contemporary snow sensors showed that commercial sensors do not offer the suitable solution mainly because of their high price or snow type dependent accuracy. The work on the development of novel snow sensor is briefly mentioned within the paper. Furthermore, we have described the proposed communication architecture, which is based on the Atmel Zigbee PRO stack. The sensing and communication unit developed within this project is referred to as Zigbee Sensor Unit (ZSU) and its design is also discussed at the end of the paper. For future work we plan to finish the development of the gateway units and the novel snow sensor and to perform the long term validation in the experimental and real environments.

ACKNOWLEDGMENT

This paper was prepared within the framework of No. FRTI2/571 grant project of the Ministry of Industry and Trade of the Czech Republic.

REFERENCES

- [1] I. Mitacek. (2010) Zrcena hala znicila mnoho aut @ONLINE. [Online]. Available: <http://www.katastrofy.com>
- [2] Satturn Holesov spol s.r.o. (1992-2010) official website. [Online]. Available: <http://www.satturn.cz>
- [3] Sommer Mess-Systemtechnik. (2007) official website. [Online]. Available: <http://www.sommer.at>
- [4] Atmel Corporation. (2011) BitCloud - ZigBee PRO. [Online]. Available: http://www.atmel.com/dyn/products/tools_card.asp?tool_id=4495
- [5] —. (2011) MCU Wireless - Zibit Modules. [Online]. Available: http://www.atmel.com/products/zigbee/zibit_modules.asp
- [6] Simek M., Fuchs M., Mraz L., Moravek P., Botta M., "Measurement of lowpan network coexistence with home microwave appliances in laboratory and home environments," in *The 6th International Conference on Broadband and Wireless Computing, Communication and Applications*, August 2011.

POSTER ABSTRACT: A WIRELESS SENSOR NETWORK ARCHITECTURE FOR SOLID WASTE MANAGEMENT

Sauro Longhi¹, Davide Marziani², Emanuele Alidori¹, Gianluca Di Buò¹,
Mariorosario Prist², Massimo Grisostomi¹ and Matteo Pirro²

¹ Università Politecnica delle Marche, Dipartimento di Ingegneria dell'Informazione (Italy),

² IDEA - Informatics, Domotics, Environment, Automation Soc. Coop. a r.l.(Italy)

Abstract – In many application fields such as home, industry, environment and health, different Wireless Sensor Network (WSN) applications have been developed to solve management problems with smart implementations. In this contest, the solid waste management is a field where this approach can be applied. In this paper a new architecture is proposed with the aim to improve the on-site handling and transfer optimization in the waste management process. The system architecture is based on TelosB sensor nodes and makes use of Data Transfer Nodes (DTN) in order to provide to a remote server the data retrieved from the garbage bins filling measurements. A remote monitoring solution has been implemented, providing user possibility to interact with the system by using a web browser. Several activities have been developed to provide a Decision Support System (DSS) to simplify the finding of solutions for resources organization problems linked to solid waste management.

I. INTRODUCTION

High population density in large urban areas makes hard the solution of solid waste management problems [1]. To reduce the environment impact of the waste dumping many municipal corporate are involved in the development of efficient waste management systems. Solid waste management is a complex process that involves many steps, it includes generation, on-site handling and storage, collection, transfer, processing and disposal of solid wastes [5]. This paper is focused on the on-site handling and storage processes and on the transfer process, with the main topic at developing a smart solid waste management system capable to ensure the public health with costs reduction and quality improvement. In order to improve the efficiency of solid wastes on-site collection and transfer an innovative solution for the monitoring and management system has been proposed. An innovative Wireless Sensor Network (WSN) has been developed to improve the garbage bins monitoring process. In such architecture each sensor node performs data acquisition and data transfer. The acquired data are sent to a supervisor system, which supports the user in the finding of solutions to decision problems, such as the optimization of resources organization (trucks, people and specific machines), with the main task to perform costs reduction. A WSN consists of many autonomous sensor nodes, spatially distributed, capable to monitor physical characteristics (e.g. temperature, humidity, light, vibration, pressure, etc.) and designed to exchange their data through the network. A wireless sensor node has not only sensing components, but also on-board processing unit, chip radio and storage units. With these enhancements, a sensor

node is often not only responsible for data collection, but also for in-network analysis based on correlation and fusion of its own data with those retrieved from other nodes. Moreover, sensor nodes can differ in their communication capabilities with different data rates and latencies. In this architecture some sensors are equipped with more powerful computation and communication capabilities in order to perform extensive processing processing and data fusion features. Such devices perform data transfers among different resource constrained networks. In the supervisor system a key role is reserved to the Decision Support System (DSS) for supporting the system administrator [2, 3]. In this system data-mining and path optimization tools are also considered to improve the efficiency of the solid waste management process.

II. SYSTEM ARCHITECTURE

The main components of the developed system are decomposed into three layers, as shown in Figure 1.

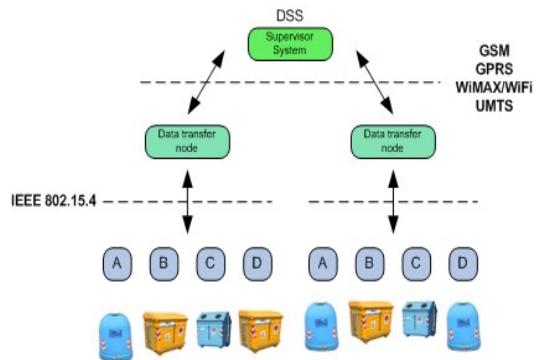


Figura 1: SEA project system architecture

Each garbage bin is supported by sensor nodes which provide the filling monitoring and the transfer of the retrieved data to a supervisor system, through the DTNs. The entire system allows the interaction among different type of wireless networks through different standard sets, such the IEEE 802.15.4, WiFi, GSM and GPRS. Taking into account an urban context, the main task is the fusion of the different Low Power Area Networks. The LowPans have been developed by using Crossbow Telos Rev.B sensor nodes and were built on TinyOS Operative System. The LowPans communicate between them through Gateway terminals that consist of

embedded solutions developed on New/Linux OS and attach the LowPans to the Internet. This perform one of the data exchange mechanism between the supervisor system and the bins as illustrated in Figure 2.

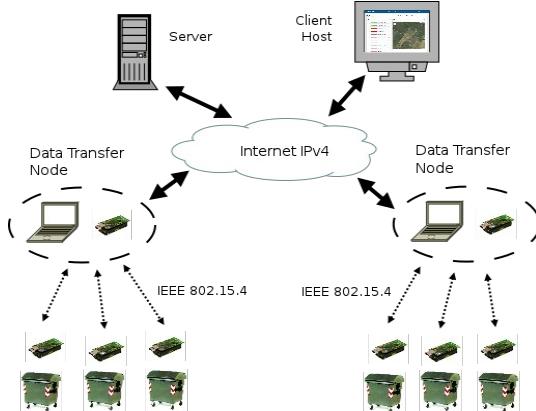


Figura 2: WSN Architecture

The architecture is composed of three parts:

1) Long range communication modules

Long range communication boards have been developed starting from the Quectel M10 GSM/GPRS modules in order to provide flexible and reliable low-cost DTNs. These modules include an embedded ARM processor and are programmable by using embedded OpenCPU [4].

2) Server layer

Server layer implements the role of mediator between users and WSNs. The interaction with the long-range communication modules has to be carefully designed. At this regard, in the SEA project architecture, two solutions have been implemented to allow data transfer between modem and server: one based on the TCP/IP socket approach and one on SMS. The first one makes use of a daemon which performs a preprocessing of the received data ensuring the consistency of them. While the second one has been implemented to provide the data acquisition when the GPRS connection is missing. In the classical solution SMS is sent to a gateway node which forwards an HTTP GET request with the SMS data.

3) User interface

The whole system provides two ways to allow a user to interact with it: a custom software client and a WEB application. The first one consists of a client installed on the user PC and has access to the central DB. This solution is a bit old-fashioned and lacks of flexibility. On the other side, the second one uses the modern cloud computing and provides access through a WEB application. Referring to the SEA project, one of the most important requirements is the remote filling monitoring of the garbage bins. Starting from this, the server provides the best path for the gathering (Traveling Salesman Problem) algorithm (Figure 3).

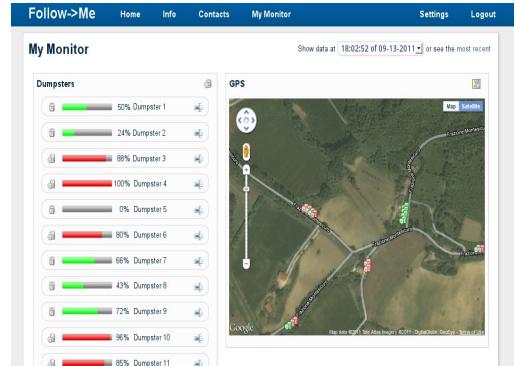


Figura 3: User Interface

III. CONCLUSION

In this paper a possible solution for solid waste management has been proposed. To monitor the bins filling, Wireless Sensor Networks have been employed to get data from specific sensors. The development of sensor nodes has led to a stable data collection and measurement repeatability and allowed nodes to interface to the Internet through the gateway terminals. Firmware for modem Quectel-M10, developed by using OpenCPU, is stable for the testing time. Software application provides data for route planning and presents them through a user-friendly interface. Several efforts are doing for the whole system optimization. These include:

- the increasing of sensor node's battery life-time;
- the improvement of electromagnetic shielding interfaces;
- the improvement of remote querying efficiency and robustness;
- the improvement of the GUI adaptation to the local administrations needs.

ACKNOWLEDGMENT

The authors would like to thank "Regione Marche" which supports the project by a grant: POR MARCHE FESR 2007-2013 - INTERVENTO 1.3.1.07.01 "SOSTEGNO ALLA NASCITA E ALLO SVILUPPO DI NUOVE IMPRESE INNOVATIVE". Thanks are also reserved to Quectel customer support which has provided contributions on the development of the long-range communication modules.

REFERENCES

- [1] Kreith, F., Tchobanoglou, G.: *Handbook of solid waste management*. McGraw-Hill (2002)
- [2] Saxena, K.B.C.: Decision support engineering: a DSS development methodology. In: 24th Annual Hawaii International Conference on System Sciences (1991)
- [3] Zhou, F., Yang, B., Li, L., Chen, Z.: Overview of the New Types of Intelligent Decision Support System. In: 3rd ICICIC (2008)
- [4] OpenCPU Quectel Cellular Engine - OpenCPU Development Guide - QUECTEL-M10
- [5] Finnveden, G.: Methodological aspects of life cycle assessment of integrated solid waste management systems. *Resources, Conservation and Recycling*, 26, 173-187 (1999)

Poster Abstract: Surveillance Application using Cooperative Robots and Sensor Networks: Challenges and Solutions

Anis Koubâa[¶], Yasir Kayani[¶], Sahar Trigui*, Imen Chaâri*, Maissa Ben Jamâa[‡], Olfa Gaddour*, Rihab Chaâri*, Hachemi Bennaceur[¶], Miled Tezeghdanti[¶], Khaled Al-Shalfan[¶], Mohamed Abid*,

[¶] Al-Imam Mohamed bin Saud University, Saudi Arabia.

* CES Research Lab, National School of Engineering, Tunisia.

[‡] REDCAD Research Lab, National School of Engineering, Tunisia.

[§] CISTER Research Unit, Polytechnic Institute of Porto (ISEP/IPP), Portugal.

Emails: aska@isep.ipp.pt, {kayaniyasir, imen.chaari}@rtrackp.com, mbenj@redcad.org, miled@softmote.com {sahar.trigui, olfa.gaddour, rihab.chaari, mohamed.abid}@ceslab.org, {hachemi, kshalfan}@ccis.imamu.edu.sa,

Abstract—In this Poster paper, we present an overview of the results we achieved during two years in the context of the R-Track Project. In this project, we designed an indoor surveillance application where a team of mobile robots supported by a sensor network infrastructure cooperate together to track intruders. The surveillance application encompasses several challenges including localization, multi-robot task allocation, communication model, and path planning, for which we provided efficient solutions. This Poster paper illustrates the different research thrusts of the project and gives insight on challenges and concerns with respect to cooperation between mobile robots and sensor networks.

I. INTRODUCTION

Mobile robots and Wireless Sensor Networks (WSNs) have enabled great potentials for ubiquitous and pervasive applications. Surveillance is one typical example of such applications, for which the literature proposed several solutions using mobile robots and/or WSNs. However, robotics and WSNs have mostly been considered as separate research fields and little work has investigated their marriage. This issue was investigated under the R-Track research project [1], and we illustrated our vision by the design of an indoor surveillance application, SURV-TRACK, which puts into play a team of multiple cooperative robots supported by a WSN infrastructure. In this Poster paper, We present the system model for SURV-TRACK to demonstrate how robots and WSNs can complement each other to efficiently accomplish the surveillance mission in distributed manner. Further, we present the main challenges tackled in the R-Track project, namely: (i.) low complexity RSS-based localization, (ii.) multi-robot task allocation (MRTA) for target capturing, and ((iii.)) robot path planning. The novelty of our solutions lies in incorporating a WSN in the robotic problems models. In what follow, we describe these challenges and present an overview of the proposed solutions.

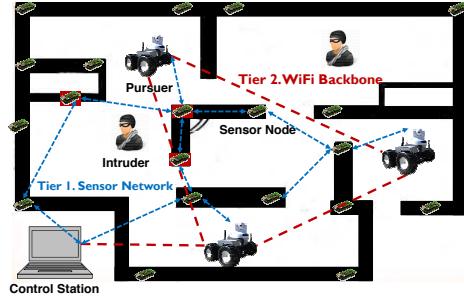


Fig. 1: SURV-TRACK Two-Tier Network Architecture

II. THE SURV-TRACK APPLICATION

The objective of SURV-TRACK is to provide monitoring services of a certain indoor geographical region (office, building floor, industrial plants) using a cooperative team of mobile robots and a fixed network infrastructure of sensor nodes (see Figure 1). SURV-TRACK consists of three main components: (i.) *mobile robots* (i.e. pursuers), which are responsible for following and catching the intruders based on the control commands sent by the control station or from data provided by the WSN, depending on the collaboration strategy. (ii.) *static WSN*: it is responsible for detecting intruders, locating mobile robots, and supporting robots in accomplishing their missions. (iii.) *central control station*: it represents a data collection center used to remotely monitor and control the area of interest. It collects data from the mobile robots and the WSN to get an updated system status (robot locations, sensor events, etc.), and remotely controls the robots and the WSN based on available information.

We considered several important requirements in SURV-TRACK design, mainly: (i.) *real-time*: as mission time must be bounded and short to ensure high system responsiveness

in case of alarms, (i.) *reliability*: providing reliable service for critical messages, such as alarms when detecting intruders, is a must; otherwise, the surveillance service will be flawed, (iii.) *self-organization*: mobile robots and sensor nodes must be able to cope with the dynamicity of the system as the robots and intruders move inside the monitored area. To this end, we proposed two-tier architecture for SURV-TRACK, as depicted in Figure 1. Tier-1 represents a basic WSN, which ensures the communication between sensor nodes themselves, and between sensor nodes, mobile robots and the control station. We investigated the use of the recently drafted RPL protocol as WSN routing protocol [2]. We experimentally observed using ContikiRPL [3] that the packet delay increases linearly with number of intermediate hops and may reach 3 seconds in a 4-hop path. To promote real-time, scalability, and reliability, Tier 2 represents a more powerful wireless network (e.g WiFi) acting as a backbone for the underlying WSN. This tier provides another alternative for the mobile robots to communicate between each other or with the control station.

III. RESEARCH CHALLENGES

A. Localization

Localization is a key issue in SURV-TRACK as it is necessary to identify the location of intruders and mobile robots during their mission. We have opted for RSS-based localization; indeed, it represents a practical solution for WSNs as it induces a low computational complexity as compared to other techniques (e.g. TDOA), and relies on built-in transceivers. The main challenge was that RSS-based localization typically induces a labor-intensive pre-deployment phase as the RSS behavior is heavily dependent on the environment. In the R-Track project, we filled this gap and proposed a new plug-and-play RSS-based localization technique that does not require any cumbersome pre-deployment environment profiling, fully distributed and scalable, as it does not require any central station for location estimation. Further, ease of deployment was considered as a primary key design requirement. Our idea relies on a simple concept: instead of building a global RSS to distance mapping based on cumbersome profiling of the environment, each anchor node determines a local mapping by exploiting the knowledge of its distances to other neighbor anchors. Each node collects a vector of RSS values after some messages exchange with neighbor anchors, and determine a linear mapping between the RSS and the log of the distance, using statistical regression. Trilateration will be used to estimate the location of unknown nodes after anchors have their mapping completed. Our preliminary experimental results in an $2mx3m$ indoor environment show that the localization error is 80% less than 1 meter. Although such an error is acceptable in indoor environment, we are working towards designing new statistical mapping functions to achieve more accurate results.

B. Multi-Robot Task Allocation

The multi-robot coordination in SURV-TRACK was considered as a an instance of the task allocation problem, which is

formulated as follows: *Given n pursuers and m intruders and a WSN distributed in the environment, how to efficiently allocate tasks to robots to capture the intruder(s) with a minimum cost, using the WSN*. We proposed three coordination strategies: (i.) *centralized strategy*: where decisions (i.e. robot mission assignments) are performed at control station based on global system status, (ii.) *distributed strategy*: where decisions are made by robots in ad-hoc fashion based their local view of the system, (iii.) *market-based strategy*: decisions are based on an auction-bid process where the control station acts as an auctioneer asking for the best price to accomplish the mission. Mobile robots submit their bids to the control station and the robot with the lowest bid for tracking a particular intruder is selected.

Our simulation results using Player/Stage simulator showed that centralized and market-based approaches provide smaller traveled distance and mission time as compared to the distributed strategy. This comes at the expense of the assumption of global knowledge of the system status either periodically in case of centralized approach, or on demand in case of market-based approach. One potential problem of centralized approach is their tractability for a large-scale system. It is also demanding in terms of communication overhead and would not be reactive to fast changes.

C. Path Planning

We also presented smartPATH, a new hybrid path planning method for a the specific environment model of cooperative robots and WSNs. Our system model incorporates a WSN infrastructure to support the robot navigation, where sensor nodes are used as signposts that help locating the mobile robot, and guide it towards the target location. the smartPATH algorithm is a hybrid Ant Colony Optimization (ACO) and Genetic Algorithm (GA) mechanism that solves the global robot path planning problem. The algorithm consists of a combination of an improved ACO algorithm for efficient and fast path selection, and a modified crossover operator for avoiding falling into a local minimum. We found out the smartPATH outperforms native ACO and GA algorithm for solving the path planning problem both and improves the solution quality up to 11% and reduces the execution time up to 12% in comparison with native ACO.

IV. ACKNOWLEDGMENT

This work is funded by R-Track project under the grant 8-INF-2008 of the National Plan for Sciences and Technology (NPST).

REFERENCES

- [1] The R-Track Project, viewed December 05 2011., [Online]. Available: <http://www.rtrackp.com>
- [2] O. Gaddour and A. Koubaa, “RPL in a Nutshell: a Survey,” *R-Track technical report TR-04-2011, submitted*.
- [3] N. Tsiftes, J. Eriksson, N. Finne, O. Fredrik, J. Hglund, and A. Dunkels, “A Framework for Low-Power IPv6 Routing Simulation, Experimentation, and Evaluation,” *SIGCOMM10, India*, pp. 479–480, Nov 2010.

Programming

Demo Abstract: Enabling Transparent WSN Resource Access via RESTful Web Services

Walter Colitti, Niccolò De Caro, Jelmer Tiete, Ha Phung, Kris Steenhaut and Abdellah Touhafi
Dept. ETRO-IRIS, Vrije Universiteit Brussel, Dept. IWT, Erasmushogeschool Brussel
{wcolitti, ndecaro, jtiete, kphung, ksteenha, atouhafi}@etro.vub.ac.be

Abstract—The Constrained Application Protocol (CoAP) is a Representational State Transfer (REST) based web transfer protocol which provides several Hypertext Transfer Protocol (HTTP) functionalities, re-designed for constrained embedded devices. CoAP enables transparent WSN resource access by means of HTTP-CoAP proxies. The transparent proxy facilitates the seamless integration of WSNs with web applications and reduces the complexity of Internet of Things (IoT) architectures. This work demonstrates the prototype design and development of a web service based platform for WSN monitoring. The system is based on an HTTP-CoAP proxy, running on a Linux embedded board, which provides transparent WSN resource access.

I. INTRODUCTION

IPv6 over Low power Wireless Personal Area Networks (6LoWPAN), a protocol standardized by the Internet Engineering Task Force (IETF), introduces IPv6 in Wireless Sensor Networks (WSNs) [4].

IP based connectivity in WSNs enables the use of embedded web service technologies, such as the ones based on Representational State Transfer (REST) architectures and on Hypertext Transfer Protocol (HTTP). Standard web service technologies improve interoperability, scalability and software reusability. In addition, the Internet is largely dominated by web applications based on REST architectures, and therefore the use of embedded REST technology would facilitate the integration of WSNs with the Internet/Web and ease the Internet of Things (IoT) deployment and management [5].

The use of standard REST/HTTP technologies in the IoT is not straightforward. Web services reside in battery operated devices and IoT applications require a multicast and asynchronous communication compared to the unicast and synchronous approach used in standard web applications. To this end, the IETF has defined a REST based web transfer protocol called Constrained Application Protocol (CoAP). CoAP includes several HTTP functionalities re-designed for small embedded devices such as sensor motes [2].

When the WSN is equipped with a REST based web transfer protocol with functionalities similar to HTTP, web applications can access WSN resources via transparent proxies. The proxy is a dual HTTP-CoAP stack which translates HTTP requests/responses into CoAP ones and vice versa [1].

The transparent resource access simplifies IoT architectures. In fact, in case of WSN applications which are not developed

using standard REST based architectures, the WSN resources can only be accessed by means of complex application gateways which have complete knowledge of the internal mechanisms of the WSN application. In case of a CoAP based WSN, WSN resources can be accessed by means of standard HTTP requests, which are automatically intercepted by the proxy and translated into CoAP requests. Since HTTP and CoAP are both based on the same REST functionalities, the protocol translation taking place in the proxy is significantly less complicated than the operations executed in an application gateway.

The aim of this work is to demonstrate how standard web technologies facilitate the integration of WSN with web applications and reduce the complexity of IoT architectures. The work shows the prototype design and development of an end-to-end web service based platform for WSN monitoring. The platform allows users to access resources on a REST/CoAP based WSN via a REST/HTTP based web application or directly from a standard web browser. The system also provides WSN resource visualization on mobile devices. A key building block of the system is a preliminary implementation of an HTTP-CoAP proxy running on a Linux embedded board.

II. SYSTEM DESCRIPTION

The system's main building blocks are depicted in Fig. 1.

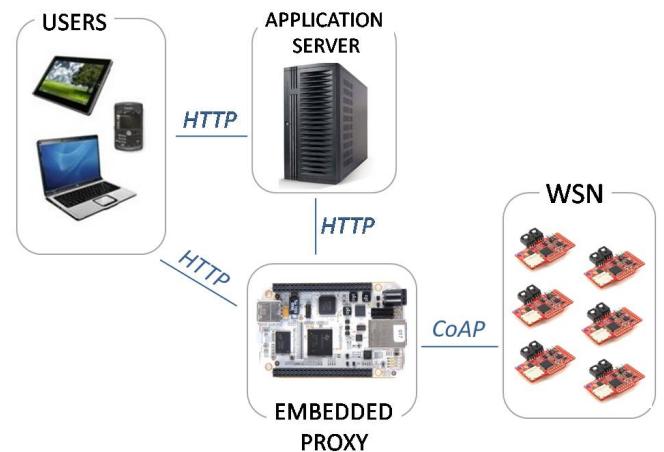


Fig. 1 Main building blocks of the REST based WSN monitoring system.

A. The Wireless Sensor Network

The WSN consists of a set of Zolertia Z1¹ wireless sensor motes running Contiki OS. Every mote has an embedded temperature sensor and has been expanded with a humidity and a light sensor. The motes are equipped with the Contiki implementation of the 6LowPAN stack and run the Erbium REST engine, which is a low power CoAP server for Contiki [3]. The resources are represented using Java Script Object Notation (JSON), a lightweight text based open standard for data interchange.

B. The HTTP-CoAP embedded proxy

1) Software implementation

The HTTP-CoAP (HC) proxy transparently maps an HTTP request coming from an HTTP client into a CoAP request to be forwarded to a CoAP server. It has been implemented following the IETF specifications given in [1].

The preliminary implementation of the HC proxy consists of three main building blocks: an HTTP server, a CoAP client and a Java module which connects the two aforementioned modules. The intermediate module reads the HTTP request arriving at the HTTP server and extracts the name of the CoAP server to be accessed and the resource path. The intermediate module resolves the name of the CoAP server mote via the Domain Name System (DNS) and passes the IPv6 address of the CoAP server mote and resource path to the CoAP client. The CoAP client creates the CoAP request and queries the CoAP server mote. After receiving the resource back, the Java module translates the CoAP response into HTTP response and gives it back to the HTTP server. The HTTP server in turn sends back the HTTP response with the resource to the HTTP client.

In the described example, the proxy is not aware of the IPv6 addresses of the sensor motes and needs to query a DNS. However, in case the proxy knows the names and IPv6 addresses of the server motes the DNS query would not be needed. Although our choice of querying the DNS requires a further DNS look up operation, it decouples the proxy from the WSN and consequently enhances the flexibility and scalability of the overall architecture.

In our work, the HTTP server chosen is the *com.sun.net.httpserver* package included in the Java JDK 1.6. The CoAP client is Californium, a CoAP framework implemented in Java².

2) Hardware implementation

The HC proxy runs on a BeagleBone development board³. It is an Ethernet connected development platform built around the TI AM3358 ARM Cortex-A8-based microprocessor; this small board of 87mm on 53mm is capable of running full-featured Linux. The IEEE 802.15.4 interface on the proxy server is implemented by connecting one of the 6 UART interfaces of the processor to a Zolertia Z1 wireless sensor mote which acts as border router.

¹ <http://www.zolertia.com/ti>

² <http://people.inf.ethz.ch/mkovatsc/californium.php>

³ <http://beagleboard.org/bone>

C. Client access to the resources

The client can access the resources either via a web application or directly in a simple web page (see Fig. 1).

In the first case, the application server provides the client with a simple user interface to query the sensor motes and to visualize their temperature, humidity and light values. The HC proxy allows the application server to be decoupled from the WSN and therefore to have a rather simple architecture. In fact, the application server only contains a simple logic to activate an HTTP client which sends HTTP requests intercepted by the HC proxy and forwarded to the sensor motes. Consequently, the application server does not have awareness that the WSN resources are being accessed via CoAP. Since the web application has been developed with Google Web Toolkit (GWT), the server needs to be a servlet container. In our implementation we have used Apache Tomcat 6. The application server also contains an adaptation for accessing the application from mobile devices (smartphone or tablet).

In the second case, the user can visualize the WSN resources directly in the mobile or non-mobile web browser and without contacting the application server. In this scenario, the user simply inserts the HTTP URI related to the sensor resource being requested in the browser's address bar. The client receives the JSON data which are directly visualized into the browser in text format.

III. CONCLUSION

This work described the prototype design and development of an end-to-end web service based platform for WSN monitoring. The preliminary implementation of an HTTP-CoAP proxy was illustrated. The proxy enables transparent WSN resource access. The platform demonstrates how standard web technologies and in particular CoAP facilitate the integration of WSN with web applications and reduce the complexity of IoT architectures.

ACKNOWLEDGMENT

This work has been done in the scope of the ITEA project ISN and the PRFB project ISEM. The authors acknowledge INNOVIRIS Brussels and the Belgian Technical Cooperation (BTC) for its financial support.

REFERENCES

- [1] A. Castellani, S. Loreto, A. Rahman, T. Fossati, and E. Dijk, "Best practices for HTTP-CoAP mapping implementation," Internet-Draft, draft-castellani-core-http-mapping-02.txt, 2011 (Work in progress).
- [2] Z. Shelby, K. Hartke, C. Bormann, and B. Frank, "Constrained Application Protocol (CoAP)," Internet-Draft, draft-ietf-core-coap-08, 2011 (Work in progress).
- [3] M. Kovatsch, S. Duquennoy, A. Dunkels, "A low-power CoAP for Contiki," Proc. IEEE Conference on Mobile ad-hoc Sensor Systems, 2011.
- [4] Z. Shelby, and C. Bormann, "6LoWPAN: The Wireless Embedded Internet," Wiley, 2009.
- [5] D. Guinard, V. Trifa, and E. Wilde, "A Resource Oriented Architecture for the Web of Things," Proc. Internet of Things 2010 International Conference (IoT 10), 2010.

Demo Abstract: From Business Process Specifications to Sensor Network Deployments

F. Casati[‡], F. Daniel[‡], G. Dantchev[†], J. Eriksson^{*}, N. Finne^{*}, S. Karnouskos[†], P. Moreno Montero^{**}, L. Mottola^{*}, F.J. Oppermann⁺, G.P. Picco[‡], A. Quartulli[‡], K. Römer⁺, P. Spiess[†], S. Tranquillini[‡], T. Voigt^{*}

^{**}Acciona Infraestructuras S.A. (Spain), [†]SAP AG (Germany), ^{*}Swedish Institute of Computer Science,

⁺University of Lübeck (Germany), [‡]University of Trento (Italy),

Abstract—The industrial adoption of wireless sensor networks (WSNs) is hampered by two main factors. First, there is a lack of integration of WSNs with business process back-ends. Second, programming WSNs is still challenging as it is mainly performed at the operating system level. To this end, we provide *makeSense* – a unified programming framework and a compilation chain that, from high-level business process specifications, generates code ready for deployment on WSN nodes.

I. INTRODUCTION AND APPLICATION SCENARIOS

Wireless Sensor Networks (WSNs) are small, untethered computing devices equipped with sensors and actuators. WSNs can be easily deployed and are able to self-organize to achieve application goals. Research has made significant progress in solving WSN-specific challenges such as energy-efficient communication. Industry, however, is reluctant to adopt WSNs. We believe this is due to two unsolved issues, integration and unification.

Integration refers to the need for strong cooperation of business back-ends with WSNs. Current approaches typically consider the WSN as a stand-alone system. As such, the integration between the WSN and the back-end infrastructure of business processes is left to application developers. Unfortunately, such an integration requires considerable effort and significant expertise spanning from traditional information systems down to low-level system details of WSN devices. Moreover, these two sets of technologies satisfy very different goals, making the integration even harder. This paper presents a holistic approach where application developers “think” at the high abstraction level of business processes, but the constructs they use are effectively implemented in the challenging reality of WSNs.

Unification refers to the need for a single, comprehensive programming framework. It is notoriously difficult to realize WSN applications. They are often developed atop the operating system, forcing the programmer away from the application logic and into low-level details. The many programming abstractions existing [1] are hard to use since they typically focus on one specific problem. To drastically simplify WSN programming, particularly for business scenarios, we need a broader approach enabling developers to use several abstractions at once. In this demo, we showcase a unified comprehensive programming framework where existing WSN programming abstractions can blend smoothly.

A paradigmatic example of our target scenarios is ventilation in buildings. Fans are commonly operated at a fixed

rate, independent of room occupation, resulting in unnecessary ventilation of unoccupied rooms and over-ventilation of sparsely occupied ones, ultimately wasting energy. A smarter strategy may consider room occupation, resulting in sustainable building management. Consider an office environment, where employees book meeting rooms on the Web through a back-end process notifying the expected participants. Room ventilation is minimal when no meeting is scheduled. Sensors and actuators driven by the business process increase ventilation before the meeting and until human presence is detected or CO₂ levels are above threshold.

Realizing this system requires a tight integration between the business process and the network of sensors and actuators dispersed in the environment, as the application logic needs to extend to the latter. Moreover, implementing the processing for adaptive ventilation complicates application development, as it departs from traditional data collection most common in WSN to encompass possibly distributed control loops.

II. APPROACH

Our design revolves around three fundamental goals:

- *makeSense* must *seamlessly integrate* with existing business process technology, providing an adoption path that complements, instead of disrupting, existing methodologies and technologies with WSN ones.
- *makeSense* must be *modular* and *extensible*. As we aim for our system to be useful across several real-world applications, extensibility is key to ensure that the programming abstractions and their implementation can be easily adapted to the specificity of the target domain as well as to unforeseen needs.
- *makeSense* must *self-optimize* w.r.t. high-level performance goals. This ability to self-adapt is necessary to support long-lasting, operational business processes immersed in the physical environment and subject to the vagaries of wireless communication.

These goals are directly reflected in the *makeSense* architecture which is based on the separation of concerns provided by a distinction in layers of functionality: *i*) an *application* layer concerned with business processes and their modeling; *ii*) a *macroprogramming* layer concerned with the distributed execution of activities within the WSN; *iii*) a *run-time* layer concerned with the low-level aspects supporting the above and enabling self-optimization.

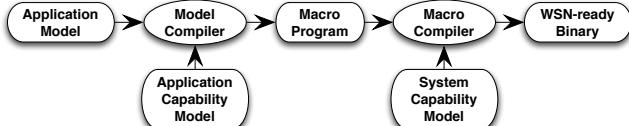


Fig. 1. Compiling business process models into WSN-executable code.

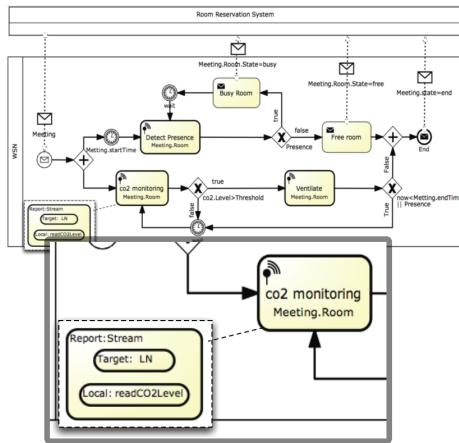


Fig. 2. BPMN diagram for a fragment of the ventilation scenario.

A model-driven approach connects the three layers (Figure 1). Using an extended version of the Business Process Model and Notation (BPMN), the application model represents a holistic, network-agnostic view of the entire business process, i.e., including the WSN and the process back-end. It includes performance requirements (e.g., a certain level of reliability, or a minimum lifetime).

The semantic link among layers is achieved by two compilation steps. The model compiler takes as input the application model and an application capability model. The latter is a coarse-grained description of the WSN, providing information such as the type of sensors/actuators available and their operations. The model compiler translates these descriptions into a program written in a macro-programming language, serving as an intermediate language closer to the reality of WSN systems, yet high-level enough to be potentially used directly by a developer. The macro language is similar to Java, but offers static memory management and threading optimized for motes. It integrates several existing programming and networking abstractions such as Logical Neighborhoods [2] to specify a set of nodes or collection tree protocols into a common object-oriented framework that is based on the notion of Actions whose execution can be configured by embedded declarative languages as illustrated in the next section. The macro compiler takes as input the macro-program generated by the model compiler and a system capability model. The latter provides finer-grained information on the deployment environment (e.g., how many sensors of a given type are deployed at a location). The macro-compiler generates executable code that relies only on the basic functionality provided by the run-time support available on the target nodes. By leveraging the system capability model, the macro compiler can generate different code for different nodes, based on their application role.

```
...
code nhoodTemplateS = {:  
    neighborhood template CO2Sensors()  
        f.getFunction() = "sensor" and t.getType() = "co2" :};  
code sensorNeighborhoodDef = nhoodTemplateS + {:  
    create neighborhood co2Sensors from CO2Sensors () :};  
  
Target co2Sensors = lnew LN();  
co2Sensors.instantiate(sensorNeighborhoodDef);  
  
Report co2Stream = lnew Stream();  
co2stream.setTarget(co2Sensors);  
co2Stream.setParameter("period", 5 * 60);  
co2Stream.execute();  
...
```

Fig. 3. Macro-programming language fragment for Figure 2.

III. CASE STUDY

Figure 2 depicts a fragment of the business process model for the ventilation scenario discussed above. The whole process is modeled with two participants, the WSN-aware participant on top and the intra-WSN participant (modeled in more detail) that is converted into an application by generating macrocode. The zoomed part of the process shows a WSN activity that sets up and executes a periodic reading of CO₂ sensors in a certain room. A *target* identifies a set of nodes satisfying application constraints, and gives the ability to apply a distributed action to the nodes in this set.

By graphically combining abstractions (here, a *target* to specify the room and a *local action* to read the sensor are placed inside a *report* action to collect the sensor readings), setting all necessary parameters, and using meta-information of the current WSN setup, the model becomes rich enough to be transformed into the macrocode in Figure 3.

This code describes the instructions to define a *target* including all CO₂ sensors and to collect periodic data from them using an instance of *report* action implementing a *Stream* concrete abstraction. Specifically, the abstraction-specific code inside the *code* variable is the Logical Neighborhood [2] custom language. This is used to create an instance of *target*, referring to local actions to retrieve the function and type of node to possibly include in the target. The *target* is given as parameter to a *setTarget* method invoked on an instance of *report*. The remaining method invocations are used to set parameters for the functioning of the *Stream* instance, e.g., its reporting period.

The BPMN model may also contain application performance objectives. Based on this and monitoring data, the self-optimization functionality tunes the protocols' parameters, e.g., by going into a very low power mode when no meeting is scheduled and no presence of people has been detected.

Acknowledgments. This work is supported by the European Commission through the projects makeSense (www.project-makesense.eu) and CONET (www.cooperating-objects.eu).

REFERENCES

- [1] L. Mottola and G. Picco, "Programming Wireless Sensor Networks: Fundamental Concepts and State of the Art," *ACM Computing Surveys*, vol. 43, no. 3, 2011.
- [2] ———, "Logical Neighborhoods: A Programming Abstraction for Wireless Sensor Networks," in *Proc. of the Int. Conf. on Distributed Computing in Sensor Systems (DCOSS)*, 2006.

Poster Abstract - WSN-Erlang: a Functional, High Level Approach to WSN Development

Alessandro Sivieri

Dipartimento di Elettronica e Informazione
Politecnico di Milano, Italy
sivieri@elet.polimi.it

Gianpaolo Cugola

Dipartimento di Elettronica e Informazione
Politecnico di Milano, Italy
cugola@elet.polimi.it

Abstract—The complexity in designing, coding, and testing WSN applications is considered among the most relevant factors that limit the diffusion of WSN technology. In this work we claim that Erlang, a concurrency and distribution-oriented, functional language, could greatly reduce this complexity. Moving from this premise we introduce a new platform, named WSN-Erlang, which leverages the Erlang peculiarities adapting them to the requirements of WSNs. WSN-Erlang also includes a testing and simulation framework, fully supporting the entire life-cycle of WSN applications.

I. INTRODUCTION

Most of the researchers operating in the WSN area agree that what limits the diffusion of WSNs and pervasive systems in general is the complexity in designing, programming, testing, and deploying real scale applications based on such technologies. In particular, the research community is still debating about the best programming paradigm to use [1].

Indeed, the peculiarities of the hardware platform introduce several new challenges: WSN applications are executed on a distributed environment composed of heterogeneous devices, with different capabilities and limited resources, interconnected using short and unreliable wireless links.

The research community has proposed two main paradigms to tackle these issues: *micro-programming* gives the programmer the responsibility of decomposing the problem and developing code for each of the network nodes, comprising the low level hardware and protocol details; this approach does not try to hide the complexity of WSNs, it gives its burden on the developers' shoulders. *Macro-programming* allows developers to access the network as a single entity, which can be programmed using abstract primitives, while the underlying framework takes care of the low level and communication details; this approach reduces the complexity of development, but it loses generality with respect to the previous one, so that many of the frameworks developed under this paradigm can be applied to a limited number of similar scenarios. Besides these differences, many of the approaches analyzed in [1] have never been tested in real world scenarios and their apparent applicability remains limited. At the same time, the main development process adopted for WSN applications is the “code and fix” one [2], which relies on the developers’ skills to overcome the limitations of the platform and the difficulties in programming it, with limited attention to the issues of reusability and maintainability.

To address this situation, we need a new language and programming framework, and we claim that Erlang [3] could represent a good starting point: it is a high-level, functional programming language, which supports facilities like:

- lightweight concurrency using the actor model [4];
- distributed programming, with high-level communication primitives;
- transparent resolution of process names over the network;
- handling of heterogeneity through the use of a virtual machine;
- pattern matching on bit streams;
- support for fault-tolerant applications.

Many of these characteristics are typically found in WSN applications, or can be quite useful for this area. Moreover, the combination of the actor-model of concurrency with high-level communication facilities, and the use of a virtual machine, ease the development of simulation tools for testing applications before deployment.

II. THE FRAMEWORK

WSN-Erlang is a new programming framework and run-time system adapting Erlang to WSN requirements. Indeed, the standard Erlang platform, even if originally developed for embedded systems (telecommunication devices), has grown overtime and nowadays contains a huge number of accessory libraries and facilities, most of which make no sense in WSNs. Because of this, we had to strip the run-time from all those parts that are not useful for our target applications. This reduced the memory, storage, and processing requirements of the platform, which as of today runs on an embedded device having an ARM926EJ-S CPU with 64 MB of RAM and 64 MB of flash, while we are currently porting it to a smaller device with a RT3050 CPU with 32 MB of RAM and 8 MB of flash. This is enough to show the advantages of our platform in terms of expressiveness and ease of use. Next step will be to enter more deeply into the virtual machine, to further reduce its requirements.

On the other hand, reducing the memory, storage, and processing requirements of the virtual machine is only a prerequisite to use Erlang in WSNs. The most important part of our work was to adapt the language and library to the peculiarities of the platform. In particular, Erlang assumes

the availability of a full fledged TCP/IP stack, offering reliable message passing among nodes, independently from their physical location. This is a strong assumption for most WSN scenarios, which we relaxed in WSN-Erlang to assume the only availability of a (unreliable) link-layer protocol. In a wireless network this means that a message may reach its destination only if the sender and the receiver are in range. Accordingly, WSN-Erlang changes the semantics of message passing to reflect this assumption. A similar change applies to the spawning of new processes among different nodes, which is unreliable and only happens if the two nodes are in range. At the same time, we leverage the availability of a shared medium to offer a broadcast version of the communication and process spawning facilities.

All these changes not only adapt the distribution model of Erlang to the peculiarities of WSNs (whose nodes rarely implement a full fledged TCP/IP stack), but it also represent the key pre-requisite to remove all the IP-based mechanisms from the standard Erlang interpreter, our next step to further reduce the WSN-Erlang requirements.

As pointed out previously, testing and simulation are an area not well supported by existing WSN frameworks; by leveraging the Erlang peculiarity of blurring the distinction between centralized and distributed applications, which are both organized as a set of concurrent processes that communicate with each other, we were able to integrate a WSN simulator in WSN-Erlang, capable of handling a completely simulated network of Erlang processes, running the very same code that runs on real nodes. In particular, the communication between nodes is simulated using the same channel and propagation models used by *TinyOS* [5]; two modes can be employed:

- complete simulation of the network, where each physical node is virtualized and executed inside a single computer; the developer can also monitor each node and inject messages in the network;
- mixed simulation of the network, where part of the nodes is virtualized and part is executed inside the target devices.

This approach allows to start debugging a WSN application in a fully simulated deployment, to move afterwards toward a mixed deployment with only one node running on the target device and the others being simulated, to conclude with a mixed deployment in which few physical nodes interact with the simulated ones. All with the guarantee that the code that is being tested coincides, line by line, with the code that will build the final application.

III. EVALUATION

We tested our prototype by implementing several algorithms that realize typical WSN activities, like collecting data from sensors or broadcasting information to the whole network. We compared them with the same algorithms developed using common WSN platforms, i.e., *TinyOS* and *Contiki*.

Results of our experiments show improvements at the source code level, in terms of readability with respect to the other implementations (which use different dialects of C). Among

TABLE I
ALGORITHMS LOC

Algorithm	TinyOS	Contiki	WSN-Erlang
Opportunistic flooder	495	187	100
Trickle	219	194	61
Collection algorithm	2169	1470	303

the features we benefit were bit sequence parsing and pattern matching on bit groups, which increased readability and code compactness of the main networking operations. Overall, the WSN-Erlang versions of the various algorithms gain one order of magnitude in terms of number of lines of code, with respect to the other versions (see Table I).

Reusability is also increased through the modularization facilities that WSN-Erlang inherits from Erlang. In particular, the usage of multiple processes inside each node can be leveraged for separating the different aspects, which can be more easily reused inside other applications, also improving the readability of the code.

IV. CONCLUSIONS AND FUTURE WORKS

WSN-Erlang addresses two limitations of currently available platforms to develop WSN applications: the lack of adequate, high-level programming abstractions to easily write reusable, maintainable code, and the difficulty in developing and testing code that may run on heterogeneous networks, with good support for debugging. Preliminary testing shows that this approach gives good results and improves the overall process of writing WSN applications.

Our current prototype cannot support the most resource constrained WSN scenarios, but at the same time WSN-Erlang runs smoothly on the kind of devices that can be found in several WSN scenarios, like health-care, where the need of saving resources is balanced by a need of reliability and fault-tolerance. Moreover, we consider the current WSN-Erlang prototype only as a first step to demonstrate the advantages of the Erlang programming model when applied to WSNs. In the future we plan to continue our work on the WSN-Erlang run-time to further reduce its requirements, putting our hands more deeply into the virtual-machine, if required.

ACKNOWLEDGMENTS

This work was partially supported by the European Commission, Programme IDEAS-ERC, Project 227977-SMScom.

REFERENCES

- [1] L. Mottola and G. P. Picco, "Programming wireless sensor networks: Fundamental concepts and state of the art," *ACM Comput. Surv.*, vol. 43, pp. 19:1–19:51, 2011.
- [2] G. P. Picco, "Software engineering and wireless sensor networks: happy marriage or consensual divorce?" in *Proceedings of the FSE/SDP workshop on Future of software engineering research*, 2010.
- [3] J. Armstrong, *Programming Erlang: Software for a Concurrent World*. Pragmatic Bookshelf, 2007.
- [4] C. Hewitt, P. Bishop, and R. Steiger, "A universal modular actor formalism for artificial intelligence," in *Proceedings of the 3rd international joint conference on Artificial intelligence*, 1973.
- [5] P. Levis, N. Lee, M. Welsh, and D. Culler, "Tossim: accurate and scalable simulation of entire tinyos applications," in *Proceedings of the 1st international conference on Embedded networked sensor systems*, 2003.

Poster Abstract: Compiler-Assisted Thread Abstractions for Resource-Constrained Systems

Alexander Bernauer

Institute for Pervasive Computing
ETH Zurich
bernauer@inf.ethz.ch

Kay Römer

Institute of Computer Engineering
Universität zu Lübeck
roemer@iti.uni-luebeck.de

Abstract—Major operating systems for wireless sensor networks (WSN) enforce an event-based programming paradigm for efficiency reasons. However, practice has shown that the resulting code complexity leads to problems during development, deployment, and operations. Although thread-based programming is known to solve these problems, the scarce resources of common WSN devices make it non-trivial to actually support it.

As opposed to existing runtime-based thread libraries, our goal is to explore the potential of compiler-assisted thread abstractions by introducing a comprehensive and platform-agnostic system of compiler and debugger which supports cooperative threads with minor restrictions.

The compiler allows to write thread-based programs that are automatically translated to equivalent event-based programs, while the debugger provides source-level debugging of the initial program, thus sustaining the thread abstraction.

Our preliminary results and ongoing evaluations suggest that the resource-wise overhead of the abstraction is moderate and can be below the overhead of runtime-based solutions. We also demonstrate that the transformation is platform-agnostic by supporting both Contiki and TinyOS.

I. INTRODUCTION

Major WSN operating systems (OS) such as TinyOS [4] and Contiki [2] account for the scarce resources of WSN devices by offering asynchronous application programming interfaces (API) and by imposing the event-based programming paradigm to its applications.

Although efficient, practice has shown that the implications of this paradigm often pose significant problems to developers. This is in particular true for the WSN domain, as deployment environments tend to differ strongly from lab environments and debugging of deployed networks is usually very time- and energy-consuming. Therefore, mistakes resulting from the inherent complexity of event-based programming tend to be very expensive to cope with.

In order to eliminate this source of mistakes altogether, researchers have investigated the question how to support thread-based programming on WSN devices despite the scarce resources. The prevalent system in the Contiki world is Protothreads [3] where a set of C preprocessor macros enable the syntactical illusion of threads and synchronous operations. In contrast, the most wide-spread approaches for TinyOS are runtime-based thread libraries such as TOSThreads [5], but with TinyVT [6] there is also a translation-based solution. There, a dedicated compiler generates a component's imple-

mentation from sequential nesC code which is enriched with special await statements where the control flow blocks until the occurrence of an event.

As opposed to runtime-based solutions, compilers can exploit application-specific properties and apply optimizations. This is why we hypothesize that compiler-assisted thread abstractions can be more resource-efficient than thread libraries. Additionally, we argue that the provided thread abstraction should also be sustained during debugging. None of the existing thread abstractions achieve this, though, and existing compiler-assisted thread abstractions have significant limitations concerning the supported thread semantics.

Thus, our goal is to take the next step of compiler-assisted thread abstractions. We aim to verify our claims by presenting a platform-agnostic source-to-source translation scheme. This scheme translates ISO/IEC 9899 (C99) applications using cooperative threads with synchronous OS APIs into C99 applications which use events with asynchronous OS APIs while preserving the operational semantics.

In preliminary work we have evaluated an initial translation scheme using a worst-case application [1]. Given the fact that we have considerably improved the translation scheme since then and there are still many opportunities for optimizations that we are going to exploit, we are confident that the efficiency of generated applications can be very close to the efficiency of hand-written applications. In the following we sketch this new translation scheme.

II. TRANSFORMATION

The input to the compiler is an OS API specification consisting of declarations of synchronous functions, and thread-based application code that uses this API (T-code). In this context, every synchronous API function is a so-called *critical function* and every function that calls a critical function, i.e. contains a *critical call*, is also critical. Non-critical functions are not altered by the transformation.

Concerning the critical functions, there are some limitations to the supported thread-semantics due to the fact that compilation is restricted to decidable problems. Thus, it is forbidden a) for critical functions to be recursive, b) to call critical functions via function pointers, and c) to perform pointer arithmetics that escapes the memory location of an object. Additionally, the number of threads is a compile-time constant.

The input OS API is translated to an asynchronous API and the T-code is translated to an equivalent event-based application that uses this API (E-code). To actually execute E-code, it is necessary to provide a *platform abstraction layer* (PAL) that implements the E-code API by using the existing API of the employed OS to trigger the desired operations and register the corresponding callbacks.

Both T-code and E-code are non-deterministic programs. We thus define an E-code to be equivalent to a T-code if and only if every possible execution of the E-code has the same *observable behavior* as at least one possible execution of the T-code. Hereby, the observable behavior of a program is the sequence of all API calls including all input parameters. Note that not including the exact timing of API calls imposes no additional restrictions, as cooperative threads are not viable for timing-critical applications anyway [5].

The transformation is sound and complete regarding the equivalence of T-code and E-code, because translating the control flow preserves the sequence of language statements while translating the data flow preserves their individual effect. We achieve this by the following means:

Concerning the data flow, for every critical function, a C structure that we call *T-frame* is generated. It stores the function's local variables, its parameters, its return value when appropriate, the caller's continuation and a union of all T-frames of its callees. For every *thread starting function*, i.e. a critical function without callers, its T-frame is instantiated once which constitutes the *T-stack* of this thread. By design, a T-stack simulates the runtime stack of a T-code thread if it would be actually executed. Thus, the translation can replace all read and write accesses to local variables with read and write accesses to T-stack variables. Furthermore, T-stacks can help managing the control flow as follows.

For every thread starting function, a so-called *thread execution function* is generated that comprises the inlined bodies of all critical - but not blocking - functions that are directly or indirectly invoked by the thread starting function. Every call to a critical function is then rewritten to the following sequence: First, the function parameters are written to the T-stack. Then, the continuation, which is the address of the label¹ preceding the first statement after the call, is written to the T-stack. Last, the control flow jumps to the label that precedes the callee's function body. Similarly, returning from a critical function results to writing the function result to the T-stack and jumping to the continuation as stored on the T-stack. Then, the caller can retrieve the result of the function call from the T-stack and continue its computation.

Invoking blocking functions also involves writing the function parameters and the continuation to the T-stack. However, the next step is to trigger the desired operation by calling the PAL's implementation of the blocking function while passing the pointer to its T-frame. As soon as the operation is finished, the PAL's obligation is to invoke the thread execution function

¹Computed gotos is a GNU extension which can easily be recreated by jump tables if standard compliance is important.

and pass it the continuation information as previously saved on the T-stack. Then the thread execution function can jump to the continuation, fetch the operation's results from the T-stack and continue its computation.

As already mentioned, compilers can exploit application-specific optimizations. For example, if a critical function is only called once in the whole program, the caller's continuation can be hard-coded into the E-code instead of being memorized. Similar optimizations exist for read-only function parameters and automatic variables that are not read after a critical call and thus can stay on the E-code stack.

In either case, the compiler can create a log of all applied transformations which can be used by a T-code debugger to map locations and variables between T-code and E-code. By using a conventional C debugger that monitors the E-code, the T-code debugger can thus provide the well-known concepts of source-level debugging for T-code applications.

III. OUTLOOK AND CONCLUSIONS

In order to evaluate the efficiency of our approach, we are planning to implement various WSN applications on top of both Contiki and TinyOS. One variant of each application will use the native event-based OS API and one will use our compiler prototype. Given such applications, we will measure their resource consumption using both static tools and simulators. The interesting metrics are a) the size of the binary, i.e., the ROM consumption, b) the RAM consumption, c) the number of CPU cycles required for one iteration of each recurring application task, and d) the total energy consumption.

Overall, we have shown how compiler-assisted thread abstraction can support almost complete thread semantics in a platform-agnostic manner. Furthermore, we have explained why we expect them to be both more efficient than runtime-based solutions and almost as efficient as hand-written event-based applications.

REFERENCES

- [1] Alexander Bernauer, Kay Römer, Silvia Santini, and Junyan Ma. Threads2Events: An Automatic Code Generation Approach. In *Proceedings of the 6th Workshop on Hot Topics in Embedded Networked Sensors*, 2010.
- [2] Adam Dunkels, Björn Gronvall, and Thiemo Voigt. Contiki - A Lightweight and Flexible Operating System for Tiny Networked Sensors. In *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*, pages 455–462, 2004.
- [3] Adam Dunkels, Oliver Schmidt, Thiemo Voigt, and Muneeb Ali. Protothreads: Simplifying Event-Driven Programming of Memory-Constrained Embedded Systems. In *Proceedings of the 4th ACM Conference on Embedded Networked Sensor Systems*, pages 29–42, 2006.
- [4] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System architecture directions for networked sensors. *SIGPLAN Not.*, 35(11):93–104, 2000.
- [5] Kevin Klues, Chieh-Jan Mike Liang, Jeongyeup Paek, Razvan Musaloiu-E, Philip Levis, Andreas Terzis, and Ramesh Govindan. TOSThreads: thread-safe and non-invasive preemption in TinyOS. In *SenSys '09: Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, pages 127–140, 2009.
- [6] János Sallai, Miklós Maróti, and Ákos Lédeczi. A concurrency abstraction for reliable sensor network applications. In *Proceedings of the 12th Monterey conference on Reliable systems on unreliable networked platforms*, pages 143–160, 2007.

Poster Abstract: SEAL: An Easy-to-use Sensor Node Application Development System

Atis Elsts

Faculty of Computer Science
University of Latvia, and
Institute of Electronics and Computer Science
Email: atis.elsts@lu.lv

Janis Judvaitis

Faculty of Computer Science
University of Latvia

Leo Selavo

Faculty of Computer Science
University of Latvia, and
Institute of Electronics and Computer Science
Email: leo.selavo@lu.lv

Abstract—Majority of potential wireless sensor network (WSN) users have no formal education in computer science. To make WSN application development more feasible and attractive to this target audience, we have designed and implemented SEAL, a domain-specific language for WSN application description. Our idea is based on the observation that even though the *low-level* details of WSN are swarming with complexities, a broad class of WSN applications are *conceptually simple* – they can be described either as sense-and-send or event detection. SEAL decouples the low-level networking and hardware details present in sensor networks from application logic, thus reducing the complexity visible to the user. We suggest that our approach leads to reduced source code sizes and faster development times, without sacrificing user freedom and energy-efficiency.

I. INTRODUCTION

The WSN technology was envisioned as a tool for a broad range of purposes and target audiences. The fulfillment of this original WSN promise is hindered by the complexities inherent in WSN programming and maintenance, which can make these tasks forbidding for people without expertise in computer science (CS). To ameliorate these difficulties, we propose our solution: **Sensor Application Development Language (SEAL)**, a language and programming environment targeted towards domain experts and novice programmers. SEAL allows the user to avoid thinking about networking and hardware details and focus on high-level application logic instead. SEAL was evaluated on agriculture scientists and on CS students with no WSN programming experience. In future, SEAL is going to be used in a WSN project for precision agriculture [1].

II. THE LANGUAGE IN BRIEF

SEAL is a domain-specific language for WSN application description. Our initial idea was to make SEAL purely declarative. Compared to the imperative approach, this has the benefit of conceptual simplicity, because one has to specify only *what* a program should do, not *how* exactly it should do that. On the second thought, we decided to include in SEAL syntax for specifying state-machine semantics as well, because of the difficulties presented by describing some common tasks (such as event detection with hysteresis) in a declarative language.

SEAL syntax features three kinds of descriptive statements (for sensors, actuators and system outputs), conditional statements, and syntax for describing state variables.

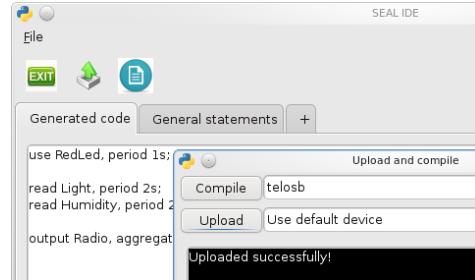


Fig. 1: SEAL graphical user interface

SEAL compiler works by translating SEAL code to C code, which is then compiled natively for a specific architecture.

SEAL is integrated in MansOS [2] and was experimentally evaluated using this WSN OS. Nevertheless, we believe it could be easily adapted for TinyOS, Contiki, or any other WSN OS that provides support for components used by SEAL (such as LED, software timers, serial port, radio, external flash memory, ADC channels etc.). Since SEAL is used to specify only user-level logic of the application, it relies on the OS for details such as scheduling algorithms or multi-hop networking.

SEAL has natural ties with MansOS run-time reprogramming system, which allows partial reprogramming of the motes. If a SEAL script is modified, only user part of the binary code must be reprogrammed, leaving majority of the code intact.

When designing SEAL, we paid attention to principles identified by usability experts [3] and recommendations for designers of novice programming systems [4]. For example, the language features “match between system and the real world” (every sensor present on a mote has a corresponding name in SEAL), “recognition rather than recall” (most of SEAL code is plain English), “aesthetic and minimalist design” (compared to relative richness of C or NesC).

To make developing WSN applications feasible even for users with no programming experience, we developed a GUI for SEAL (Figure 1), written in Python.

III. EVALUATION

As the SEAL source code examples given below show, when compared to native programming in TinyOS or MansOS (Table I), they tend to have much smaller code size, thus making software prototyping faster.

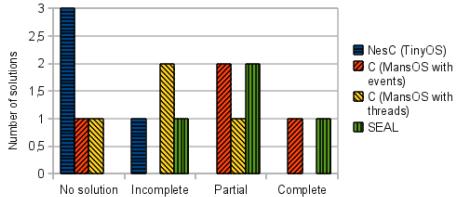


Fig. 2: Student solutions

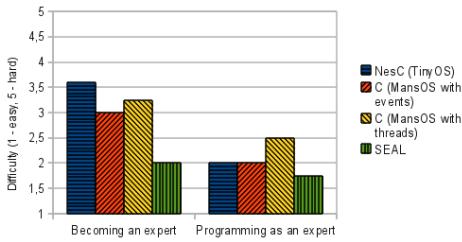


Fig. 3: Difficulty evaluation

The C code generated by SEAL compiler uses event-based control flow, and consequently works well together with duty-cycling. The sample applications presented here all have small duty cycles: 0.037% for blink, 0.627% for sense-and-send, 0.701% for event detection (evaluated on Tmote Sky at 4MHz).

Listing 1 Blink

```
use RedLed, period 1000ms;
```

Listing 2 Sense-and-send

```
read Light, period 2s;
read Humidity, period 6s;
output Radio;
```

Listing 3 Event detection with hysteresis

```
state temperatureCritical false;
when Sensors.Temperature > 50C:
    set temperatureCritical true;
when Sensors.Temperature < 40C:
    set temperatureCritical false;
when temperatureCritical: // blink red LED
    use RedLed, period 100ms;
```

SEAL was evaluated in tests on university students and domain scientists.

Firstly, four CS students from University of Latvia with no WSN experience were asked to write an advanced sense-and-send program in SEAL. The results (Figure 2) were compared with an earlier test, where different groups of students were asked to program the same application in MansOS and TinyOS. Three of the students produced a solution that was either completely correct or had only small syntactical defects. Despite their previous C programming experience, the students produced worse results when using plain C, and much worse results when asked to program in NesC. SEAL was also evaluated as both easier to learn and to use in expert level.

TABLE I: Lines of Code Comparision

Program	Lines of code		
	NesC (TinyOS)	C (MansOS)	SEAL
Blink	26	7	1
Sense-and-send	93	38	3
Event detection	42	22	7

Secondly, four agriculture scientists from Latvia State Institute of Fruit-Growing were asked to program the same solution using SEAL GUI. After initial help with installation and setup, all were able to complete this assignment on their own or using only discussions with colleagues. Such a result would be unthinkable with C, where they would have to battle with issues such as C pointer semantics and memory alignment.

IV. RELATED WORK

WSN query systems such as TinyDB [5] have the closest relation to our work. However, we believe their SQL-like syntax is not as intuitive for novice users, and, unlike SEAL, they are not agnostic to network protocols. Consequently, these systems are less portable and much harder to implement. Declarative WSN programming systems have been proposed before [6]. However, they were not aimed towards novice users and their Prolog-like syntax is intimidating to all but seasoned CS professionals. As for WSN macroprogramming systems such as Regiment [7] – we consider them as a complementary, not a conflicting solution.

V. CONCLUSION

SEAL has the potential to make WSN application development more accessible to non CS-majors, to make application prototyping faster, while keeping the benefits offered by low duty-cycling and partial reprogramming. It has been evaluated favorably on the target audience.

ACKNOWLEDGMENTS

This work was supported by European Social Fund grant Nr. 2009/0219/1DP/1.1.2.0/APIA/VIAA/020. We would like to thank all test subjects and Girts Strazdins from Institute of Electronics and Computer Science for helping to conduct the experimental evaluation.

REFERENCES

- [1] A. Elsts, R. Balass, J. Judvaitis, R. Zviedris, G. Strazdins, A. Mednis, and L. Selavo, “SADmote: A Robust and Cost-Effective Device for Environmental Monitoring,” accepted for publication in Proceedings of Architecture of Computing Systems (ARCS 2012).
- [2] G. Strazdins, A. Elsts, and L. Selavo, “MansOS: Easy to Use, Portable and Resource Efficient Operating System for Networked Embedded Devices,” in Proc. SenSys’10, 2010.
- [3] J. Nielsen, “Ten usability heuristics,” Useit. com, 2005.
- [4] J. Pane and B. Myers, “Usability issues in the design of novice programming systems,” 1996.
- [5] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, “Tinydb: an acquisitional query processing system for sensor networks,” ACM Transactions on Database Systems (TODS), vol. 30, no. 1, pp. 122–173, 2005.
- [6] D. Chu, L. Popa, A. Tavakoli, J. Hellerstein, P. Levis, S. Shenker, and I. Stoica, “The design and implementation of a declarative sensor network system,” in Proceedings of the 5th international conference on Embedded networked sensor systems. ACM, 2007, pp. 175–188.
- [7] R. Newton, G. Morrisett, and M. Welsh, “The regiment macroprogramming system,” in Proceedings of the 6th international conference on Information processing in sensor networks. ACM, 2007, pp. 489–498.

Protocols

Demo Abstract: GinLITE - A MAC Protocol for Real-Time Sensor Networks

James Brown and Utz Roedig
Lancaster University
Lancaster, UK

Abstract—In this demo we present the GinLITE medium access control protocol. GinLITE is a generic open source medium access control protocol designed to offer time-critical and reliable data delivery. GinLITE is easily extensible and customisable and is intended to be used as a building block for research sensor networks that have strict performance requirements. The protocol was developed as a component for the GINSENG system, a system designed for industrial process automation and control applications. A simple application is presented for the purpose of demonstrating the capabilities of GinLITE.

I. INTRODUCTION

The majority of manufacturing industries utilize systems to monitor and control their production processes. For instance, in a typical oil refinery, such systems are usually made up of a distributed network of various in-field devices monitoring and controlling pumps, valves, motors etc. These process control and automation applications can be considered as real-time. Automated control loops are mapped onto a wireless network and it is essential for their correct functioning that messages are delivered timely and reliable. Whilst traditionally fixed wired networks have been used for such scenarios, the use of wireless sensor networks is compelling due to increased flexibility and reduced economical cost. However, WSN's have conventionally been relatively unreliable, with little to no consideration for real-time behaviour, making them unsuitable for the outlined scenario.

Recently a number of systems have materialized that have been designed to offer real-time and reliable data delivery to support such time-critical applications. These systems generally use at their core parts of the IEEE 802.15.4 standard and a time division multiple access (TDMA) protocol. The TDMA MAC schedule is provisioned to support the requirements of the application with adequate spare capacity to increase reliability. Such systems have been commercial in nature and typically span across different entities outside of the WSN for instance management systems. The need for other external entities to the WSN and their closed commercial nature, blocks the use by the research community of these systems making research in this area challenging. Examples of such systems are WirelessHART and ISA100.11a.

The EU-funded GINSENG project was set up to investigate the problems associated with supporting industrial process and control applications through wireless sensor networks. The

designed GINSENG system [1] is a research platform built to support time-critical applications. The system provides novel software components such as real-time OS extension, real-time communication protocol, topology and traffic management and deployment and debugging facilities.

The GINSENG system assumes a carefully planned deployment of static sensor and actuator nodes as the basis to achieve performance control. The heart of the GINSENG system is the TDMA MAC protocol GinMAC [2]. It uses a virtual tree topology alongside a pre-computed exclusive TDMA schedule, optimized to the requirements of the application. Reliability is ensured by the provisioning of redundant slots within the schedule for retransmission whose number is determined during pre-deployment measurements. GinMAC provides attachment points for other GINSENG system components such as topology control, traffic management or performance debugging. The GINSENG system has seen extensive testing in a number of lab based experiments in addition to a long term deployment on a testbed in the Sines Oil Refinery in Portugal. The different components and capabilities of the overall GINSENG system are described in [3], [2], [4], [5].

In addition to industrial process control, many other domains also require reliable time-critical data delivery. With the lack of available open source off-the-shelf systems to support these requirements, commencing research in these areas is difficult. Whilst the GINSENG system could be used to support such research, it would often be considered excessive and to complex providing many unnecessary features which have been tailored to particular industry settings. Instead a reduced system is needed designed as a research enabler for real-time wireless sensor networks. This reduced system is GinLITE, a derivate of the original GinMAC, designed to provide only the essential minimum features and to support modification to adapt the protocol to a variety of real-time research tasks.

In this paper we describe core features of GinLITE and the demo showcases its capabilities.

II. GINLITE OVERVIEW

GinLITE is implemented for the Contiki sensor network operating system. GinLITE supports a number of advantages and improvements over GinMAC such as reduced resource use, a simpler implementation, better integration into ContikiOS

and increased stability. These features make the system more suitable as a reliable research platform.

The system requires less than 5KB of flash memory which is comparable to other Contiki MAC protocols that do not have to deal with the complexity of supporting real-time communication. With regards to RAM the system requires 1.2KB of RAM which includes support for a frame queue of 8 frames. This queue partially replaces the default queue of 16 packets held in the Contiki RIME system. With a modest Rime based application, only 23KB of flash and 3KB of RAM are used leaving 25KB of flash and 7KB of RAM available on a commonly used TelosB mote for system extension.

The implementation of GinLITE has a simple structure with the entire MAC protocol contained in a single file in a similar structure to other Contiki MAC protocols.

Additional components such as Topology Control, Overload Control or Performance Debugging can be attached. However, the basic system only includes minimal implementations of these add-ons. For example, the system utilizes a purely static topology with pre-computed and static TDMA schedule. However, if a research project requires, a complex topology management component can be added. The system provides mechanisms to transmit additional information piggybacked on data transmissions which allows researchers to construct complex performance monitoring components without disturbing real-time communication. The basic system supports a simple FIFO queue but this can be replaced by sophisticated queue management if the targeted research requires this.

GinLITE has been designed to be tightly integrated with ContikiOS, implementing the standard Contiki MAC interfaces. This allows the system to be used in the same way as other built-in MAC protocols. This allows the system to be used with ease, with the provided ContikiOS example RIME based applications and with some small modifications with 6lowpan applications.

The GinLITE system is currently compatible with ContikiOS 2.4 and can be downloaded from [1].

A. Using GinLITE

GinLITE is a TDMA mesh-under MAC protocol and as such it requires topology and TDMA schedule information for operation. If no complex topology management is implemented, static topology/schedule information can be supplied via configuration files. Examples for this configuration procedure are supplied with the available code.

For a simple static topology, the number of nodes at each layer of a tree beginning with the sink needs to be specified. Furthermore, each node's position in the tree must be configured.

Each node adheres to a configured schedule. This schedule consists of active slots used for transmission and reception of data and inactive slots that are used for application processing or to conserve energy. Transmission slots are used exclusively to avoid collisions. Slots are allocated to nodes for transmitting their own data and for forwarding data from/to nodes located

further down in the tree topology. Redundant slots can be specified which will be used for retransmissions if needed. By specifying the right number of redundant slots, reliability targets can be met.

The application can use the Contiki Rime stack with GinLITE in the same way as other protocols. Packets can be created and passed to the send function and received via a callback function. To ensure real-time processing, the MAC protocol utilises a Contiki rtimer executed function operating outside of the run-to-completion environment. As the callback receive function will be invoked from this rtimer function, to maintain real-time performance it should be minimal and simply store the packet for later processing by its thread.

B. The GinLITE Demonstration

To demonstrate the GinLITE capabilities a tree topology consisting of 11 nodes is used. The schedule length is set to be one second which ensures that all data arrives at the sink within one second. The application uses the Rime stack and each node reports temperature reading and button state to the sink node connected to a pc. The pc will display received data and, for demonstration purpose, additional performance metrics such as message delay and reliability. A button press will generate an actuator message that will be sent from the sink to a specific node to enable a led.

III. SUMMARY AND FUTURE WORK

This paper has presented GinLITE an open source MAC protocol built to provide reliable time-critical data delivery. It is our plan to continue the development of the GinLITE system providing bug fixes and optimizations as well as porting GinLITE to future versions of Contiki including the recently released Contiki 2.5 in addition to other hardware platforms such as the Zolertia Z1 mote.

ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 224282.

REFERENCES

- [1] “Ginseng,” 2011. [Online]. Available: <http://www.ict-ginseng.eu/>
- [2] P. Suriyachai, J. Brown, and U. Roedig, “Time-critical data delivery in wireless sensor networks,” in *6th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS '10)*. IEEE, 2010.
- [3] C. Sreenan, J. S. Silva, L. Wolf, R. Eiras, T. Voigt, U. Roedig, V. Vasiliou, and G. Hackenbroich, “Performance control in wireless sensor networks: the ginseng project - [Global communications news letter],” *Communications Magazine*, vol. 47, no. 8, Aug. 2009.
- [4] J. Brown, B. McCarthy, U. Roedig, T. Voigt, and C. J. Sreenan, “Burst-probe: Debugging time-critical data delivery in wireless sensor networks,” in *Proceedings of the Eighth European Conference on Wireless Sensor Networks (EWSN 2011)*, Feb 2011.
- [5] W.-B. Pottner, L. Wolf, J. Cecilio, P. Furtado, R. Silva, J. Silva, A. Santos, P. Gil, A. Cardoso, Z. Zinonos, B. McCarthy, J. Brown, U. Roedig, T. O'Donovan, C. Sreenan, Z. He, T. Voigt, and A. Jugel, “Wsn evaluation in industrial environments first results and lessons learned,” in *Proceedings of the 3rd International Workshop on Performance Control in Wireless Sensor Networks (PWSN/DCOSS 2011)*, 2011.

Poster Abstract: Agreement for Wireless Sensor Networks under External Interference

Carlo Alberto Boano and Kay Römer
University of Lübeck, Germany

Thiemo Voigt
SICS, Sweden

Marco Antonio Zúñiga
University of Duisburg-Essen, Germany

Abstract—Wireless sensor nodes often need to agree on fundamental pieces of information: at the MAC layer, sensor nodes may need to agree on a new time slot or frequency channel; at the application layer they may need to agree on handing over a leader role from one node to another. With the increasing congestion of the unregulated ISM frequencies, the quality of communications deteriorates, leading to packet loss and higher latencies that may break agreement in two different ways: none of the nodes agree on the new information (time slot, frequency channel) and stick with the previous state, or – even worse – some nodes agree on the new information and some do not. In this work, we propose a protocol that exploits jamming instead of message transmissions to confirm the reception of a packet. We show that, in the presence of common interference patterns, this approach outperforms packet-based handshake protocols in terms of both agreement probability and energy consumption.

I. INTRODUCTION AND MOTIVATION

In distributed systems, no delivery guarantee can be given on the messages that are sent [1], therefore traditional agreement protocols make use of several messages to agree on a piece of information among 2 nodes. A well-known agreement protocol is the *n-way handshake*, in which the first message conveys the descriptive information (e.g., "switch to channel x"), and the following $n \geq 1$ messages are sent in an alternated manner to acknowledge the reception of the previous packet(s). For example, TCP employs a 3-way handshake to establish connections over a network, and an agreement is reached only if all packets have been correctly received by both nodes.

This approach is however not optimal for wireless sensor networks challenged by external interference for two main reasons. Firstly, the probability of receiving successfully a long sequence of packets is very low. Secondly, the overhead introduced by the packet header and footer is much larger than the information carried by an acknowledgment message itself, making it unnecessarily more vulnerable to interference.

In unregulated ISM bands such as the 2.4 GHz frequencies, wireless sensor nodes coexist with higher-power transmitters such as WLAN and Bluetooth. As a result, low-power transmissions may result in corrupted and undecodable packets [2]. In IEEE 802.15.4 devices, a packet is composed of a synchronization, physical and MAC header in addition to the payload carrying the actual data. Even when the information enclosed into the packet is minimal (such as in the case of an ACK), the probability of having a corrupted packet is given by its complete size, inclusive of headers and footers.

The main idea of our work is to use jamming as the binary signal to acknowledge the packet reception, in order to remove

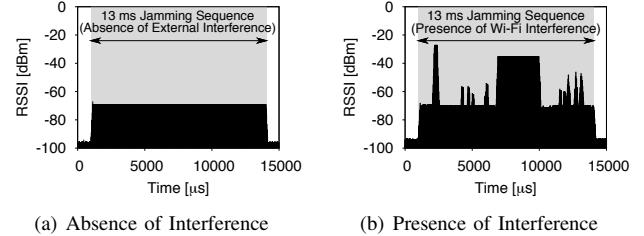


Fig. 1. RSSI values recorded during the transmission of a jamming sequence.

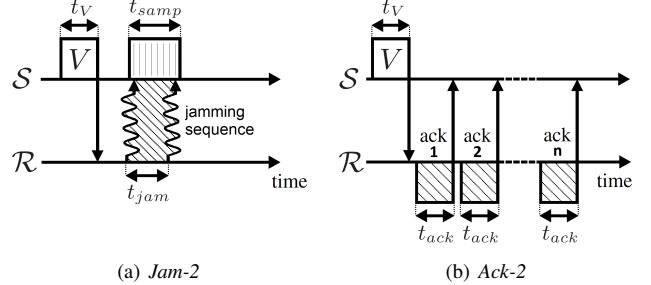


Fig. 2. Illustration of two-way handshake agreement protocols.

the overhead introduced by the packet headers and footers: after the transmission of the data packet(s), the subsequent acknowledgments can be sent in the form of jamming signals. The key insight behind this approach is that jamming can often be detected even under external interference, while ACK packets would instead be lost. We design *Jam-2*, a two-way handshake protocol in which the message receipt is sent in the form of a continuous jamming signal, and show its performance improvements under interference with respect to the traditional packet-based two-way handshake protocol.

II. CONTRIBUTION

Fig. 1(a) shows a jamming sequence lasting for a time window t_{jam} as perceived by a receiving node employing the CC2420 radio transceiver: in absence of interference, the RSSI values are stable and clearly above the sensitivity threshold of the radio. In the presence of external interference (Fig. 1(b)), the RSSI register will return the maximum interfering signal observed among the jamming signal (flat baseline) and the external source (bursty spikes) due to the co-channel rejection properties of the radio [3]. Typical interference sources – in contrast to a jamming signal – do not produce continuous interference for long periods of time, rather they alternate between idle and busy periods. That is, if t_{jam} lasts longer than the longest busy period of the interfering signal, we can detect if a jamming sequence was sent or not by checking if

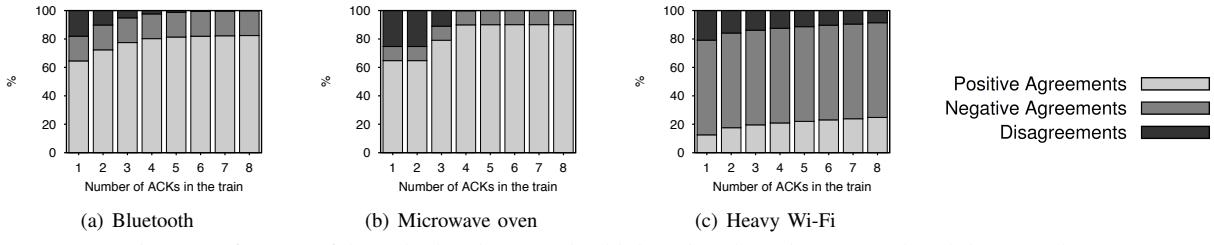


Fig. 3. Performance of the packet-based two-way handshake *Ack-2* when using $n \geq 1$ acknowledgment packets.

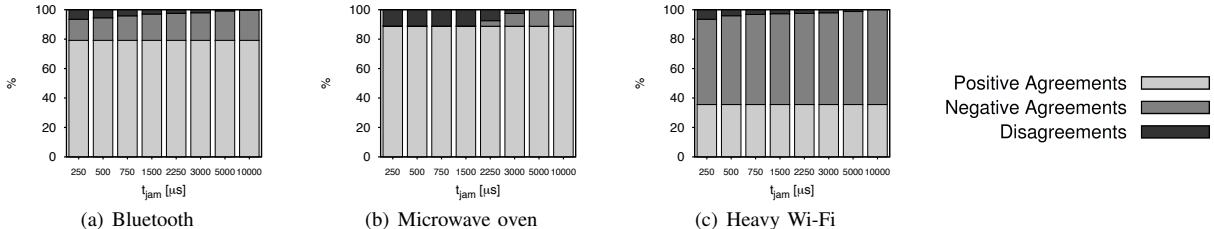


Fig. 4. Performance of *Jam-2*: compared to *Ack-2* (Fig. 3), *Jam-2* maximizes the amount of positive agreements and minimizes the amount of disagreements.

any of the RSSI samples equals the radio sensitivity threshold. We use this observation to design *Jam-2*. Our implementation on Maxfor MTM-CM5000MSP motes uses Contiki and two main building blocks: (i) the generation of a jamming sequence and (ii) the high-frequency RSSI sampling that detects the presence of a jamming sequence. We implement the former using the CC2420 test modes as in [4] and the latter following the approach in [3] to obtain an RSSI sample every $20\mu s$.

Fig. 2(a) shows a sketch of the protocol: given two nodes \mathcal{S} and \mathcal{R} , where \mathcal{S} initiates the exchange and sends the information V towards \mathcal{R} , we call *Jam-2* the two-way handshake in which \mathcal{R} accepts V and sends the acknowledgment to \mathcal{S} in the form of a jamming sequence of duration t_{jam} . While \mathcal{R} transmits a jamming signal, \mathcal{S} carries out a high-frequency RSSI sampling for a period $t_{samp} \leq t_{jam}$. Denoting r_{noise} as the maximum RSSI value measured in the absence of interference, and $\{x_1, \dots, x_n\}$ as the sequence of RSSI values sampled during t_{samp} , we define the binary sequence $\{X_1, \dots, X_n\}$ as follows: if $x_i \leq r_{noise}$, then $X_i = 1$, else $X_i = 0$. If $\sum_{i=1}^n X_i = 0$, \mathcal{S} assumes that a jamming sequence was transmitted by \mathcal{R} and deems the exchange as successful.

We compare the performance of *Jam-2* with the performance of *Ack-2*: a 2-way handshake protocol employing ACK packets to confirm the reception of the information (Fig. 2(b)). In order to increase the performance of *Ack-2*, we consider packet redundancy, that is, \mathcal{R} sends a sequence of $n \geq 1$ ACK messages to confirm the reception of V , and \mathcal{S} deems the exchange successful if it receives at least one ACK packet.

The exchange between \mathcal{S} and \mathcal{R} can have three possible outcomes. If both nodes deem the exchange as successful and accept V we have a *positive agreement*. If both nodes deem the exchange as unsuccessful and discard V we have a *negative agreement*. We have *disagreement* when one of the nodes deems the exchange as successful, while the second node deems the exchange as unsuccessful. While a disagreement is a potentially pernicious outcome, a negative agreement is often less severe. For example, if the shared value contains the next wireless channel to be used for communication, two nodes

are better staying in the same lossy wireless channel, rather than having only one of them move to a different channel.

We compare the performance of the 2 protocols under realistic interference patterns generated using *JamLab* (Bluetooth/Wi-Fi file transfer, active microwave oven) [3]. Fig. 3 and 4 show the results: the probability of disagreements with *Jam-2* is much lower than that of *Ack-2* even for short t_{jam} . Furthermore, due to the reliable detection of a jamming signal, in *Jam-2* the amount of positive agreements remains constant and reaches the maximum already with short t_{jam} .

We have verified experimentally that a 1-byte payload ACK message has a transmission delay of $782\ \mu s$. This implies that under a fair comparison ($t_{jam} = 750\ \mu s$ for *Jam-2* and one ACK packet for *Ack-2*), *Jam-2* significantly outperforms *Ack-2* (more positive agreements and less disagreements). In practice, the difference would be even more favorable to *Jam-2*, because the processing and sending time of 1-byte ACK takes $2083\ \mu s$.

III. OUTLOOK

We have proposed a jamming-based agreement protocol for wireless sensor networks challenged by external interference that overcomes the fundamental limitations of regular ACK packets being corrupted under interference. This approach can be used to build robust agreement protocols for wireless sensor networks. We are currently implementing a broadcast agreement protocol with encouraging preliminary results.

ACKNOWLEDGMENTS

This work has been supported by the European Union with contract FP7-2007-2-224053 (CONET) and by the DFG-funded Cluster of Excellence 306/1 "Inflamm. at Interfaces".

REFERENCES

- [1] E.A. Akkoyunlu et al. Some constraints and tradeoffs in the design of network communications. In *Operating Systems Principles*, 1975.
- [2] C. Liang, N. Bodhi Priyantha, J. Liu, and A. Terzis. Surviving Wi-Fi Interference in Low Power ZigBee Networks. In *ACM SenSys 2010*.
- [3] C.A. Boano et al. JamLab: Augmenting Sensorset Testbeds with Realistic and Controlled Interference Generation. In *IPSN 2011*.
- [4] C.A. Boano et al. Controllable Radio Interference for Experimental and Testing Purposes in WSN. In *4th IEEE SenseApp*, October 2009.

Poster Abstract: Distributed Protocol Stacks for Wireless Sensor Networks

Peter Rothenpieler

Institute of Telematics, University of Lübeck, Germany
rothenpieler@itm.uni-luebeck.de

Abstract—We propose the use of RPC techniques to compensate the code size requirements of existing IPv6 stacks for WSNs, while preserving full IP connectivity. In our model, neighbouring nodes share layers of their communication stack with each other in a cooperative fashion. This enables the use of IPv6 on every node without the need to include the stacks full programming logic on each node, while preserving transparent IP interconnectivity by forming a Distributed Protocol Stack (DPS). We give an overview on the concept behind DPS and evaluate our first prototype implementation against the native IPv6 implementation on the iSense WSN platform regarding code size and round-trip time and give an outlook on further advantages of our model.

I. INTRODUCTION

With the dawn of 6LoWPAN [1] and the transmission of IPv6 via IEEE 802.15.4 radios, WSNs are becoming one important cornerstone of the Internet of Things (IoT). One of the most prominent visions in this field is the concept of *Smart Dust* [2]: Hundreds or even thousands of low cost sensor nodes which form a wireless mesh network and sense environmental data with almost arbitrary spatial and temporal resolution, enabling the real-time representation of the physical world in the virtual world. While miniaturization is being used to produce even smaller nodes for even less money, the WSN software becomes more complicated and requires more memory: Contemporary WSNs allow the direct integration of each sensor node into the Internet using full featured and self configuring IPv6 stacks. In the work of [3] and [4], a comparison of the code size of different IPv6 stacks for WSN platforms is provided, showing that the smallest IP stacks require 10-20 KB (NanoStack, 6lowpancli, blip, uIPv6), whereas some implementations (including e.g. ND) require up to 50 KB of flash (m-Stack). Even though modern WSN platforms can provide enough memory, each additional KB of flash or RAM increases the cost of the WSN and reduces the memory available to the actual application. The MSP430, which is supported by each of the above mentioned IP stacks, is available with different flash configurations starting at 0.5 KB for \$0.34 and up to 256 KB of flash for \$5-7. Of all available MSP430 models, 60.3% (Data from [5]) don't have enough flash to even support the operation of the IP stack alone, considering an average requirement of 24 KB flash (c.f. [3] and [4]).

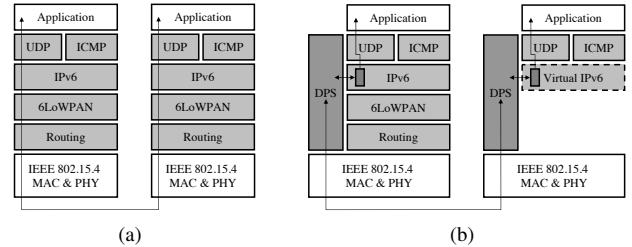


Fig. 1. TCP/IP stack (left) and proposed Distributed Protocol Stack (right)

II. CONCEPT

In computer networking, a protocol stack consists of a number of individual protocol layers, which are stacked on top of each other in a standardized order, like e.g. the 6LoWPAN protocol stack for WSNs in Fig. 1(a) which follows the TCP/IP model. Information, which is sent from one application to another on a different host, traverses the full stack downwards on the source node and then upwards on the destination node, passing every layer on both hosts in opposite directions on its way. In the classical case, each node in the network implements the full protocol stack, whereas in a DPS, neighbouring nodes share implementations of layers as depicted in Fig. 1(b): In this case, the left *Server* node implements all layers of the protocol stack, while the right *Client* node implements only the application and transport layers (UDP and ICMP), followed by a virtual IPv6 layer. The DPS thus acts as a bridge between the client and the server, which allows the client to use the remaining layers of the server, performing a similar task as an ORB (Object Request Broker, cf. CORBA). The virtual IP layer in the above example can also be described as the *Client stub* of the IP layer, which communicates the *Server skeleton* over the network as in RPC (Remote Procedure Call) or RMI (Remote Method Invocation) distributed computing environments. We will describe the underlying DPS protocol in the following section.

III. PROTOCOL

The DPS protocol consists of three phases: It starts with the Discovery phase, in which the clients send out DISCOVERY messages, which are answered by ADVERTISE messages from the servers. This is followed by the BINDING phase, which consists of a three-way handshake, initiated by the client node. During the handshake, client and server exchange

TABLE I
CODE SIZE OF THE NATIVE IPv6 AND DPS IMPLEMENTATIONS

	Native IPv6	DPS Server	DPS Client
IPv6	31.6 KB	31.6 KB	6.7 KB
ND	7.6 KB	7.6 KB	–
6LoWPAN	7.3 KB	7.3 KB	–
ICMP	1.8 KB	1.8 KB	1.8 KB
UDP	1.9 KB	1.9 KB	1.9 KB
DPS	–	14.3 KB	14.3 KB
Σ	50.2 KB	64.5 KB	24.7 KB

their packet counters and the connection nonce, which in conjunction with a pairwise key, are used to protect the integrity and authenticity of all DPS messages using a CBC-MAC and to protect the DPS against replay attacks. After establishing the connection, server and client may execute RPC calls between their protocol stubs and skeletons, following a request-response pattern. The RPC messages support fragmentation and reliability using per-fragment acknowledgements.

IV. EVALUATION

As a first evaluation scenario, we have implemented the DPS protocol for using a virtual IPv6 layer on the iSense WSN platform using the iSense6LoWPAN stack. The client stub offers an interface for the RPC functions *send*, *receive* and *setIPAddr*, whereas the server skeleton implements the corresponding functions. After the client has successfully established the DPS connection, the server determines the clients IPv6 address by using the clients MAC address. The server afterwards calls the *setIPAddr* function of the client, telling the client its IPv6 address and the network prefix. The code size evaluation in Table I shows that the DPS component takes up 14.3 KB, increasing the IP stack memory requirements for the DPS server from 50.2 to 64.5 KB, whereas the DPS clients requirements are halved to 24.7 KB. To evaluate the impact of the DPS on the performance of the IP stack, we compare the single-hop round-trip time (RTT) of the native and DPS implementation using ICMP echo request/reply messages. As depicted in Fig. 2, the RTT linearly increases with ICMP payload sizes and additionally increases in periodic intervals, caused by the fragmentation mechanism. Several factors increase the RTT of the DPS experiments: The missing 6LoWPAN compression layer increases the overhead by 36 byte per packet, whereas the DPS fragmentation has an 11 byte larger overhead per fragment. Additionally, the native 6LoWPAN stack does not use acknowledgements, whereas DPS sends one ACK of 14 byte per fragment, requiring also one additional clear channel assessment.

V. CONCLUSION

Our first experiments have shown the feasibility of the DPS paradigm. In our next steps, we will test different optimizations and their impact on the evaluated parameters. We will introduce a better fragmentation mechanism with less overhead, a cumulative ACK mechanisms to speed up the RPC message transfer and investigate the impact of packet compression techniques. We will also implement additional

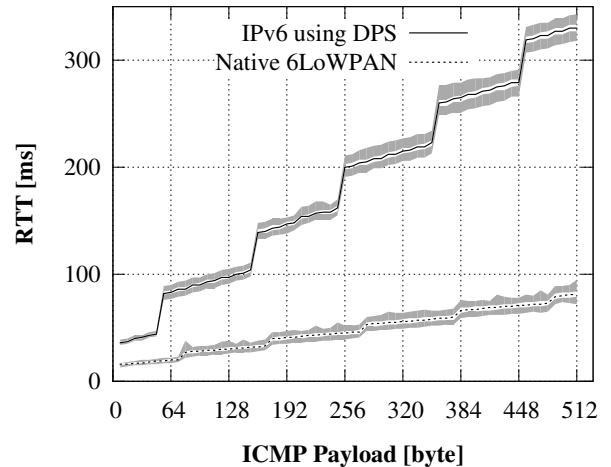


Fig. 2. Singlehop RTT: Native 6LoWPAN (bottom) vs. DPS (top). Curves depict mean RTT, grey corridors mark the minimum and maximum RTT. (Based upon 1000 ICMP echo requests per payload size)

stubs and skeletons to use the routing or transport layers via RPC in addition to the described virtual IPv6 layer. We will further evaluate which additional benefits can be generated by the use of DPS in the field of security: The use of DPS may increase the security of the WSN by distributing the stack layers over multiple nodes, which limits the effect of IP layer security vulnerabilities to the nodes implementing the vulnerable layers, whereas in a classical IP stack, the complete node would be compromised. Furthermore, surrounding servers which implement the same layer may jointly identify and report malicious behaviour of compromised nodes and advertise themselves as an uncompromised alternative to the affected client. clients may also use multiple DPS servers to achieve load balancing and fault tolerance. We will investigate, if the reliability and security features of the DPS may be re-configured to fit the specific needs of the WSN, e.g. if they may be deactivated to further decrease the code size and RTT.

REFERENCES

- [1] J. Hui and P. Thubert, “Compression Format for IPv6 Datagrams in Low Power and Lossy Networks (6LoWPAN),” Network Working Group, Internet-Draft draft-ietf-6lowpan-hc-15, Feb. 2011. [Online]. Available: <http://tools.ietf.org/html/draft-ietf-6lowpan-hc-15>
- [2] B. Warneke, M. Last, B. Liebowitz, and K. S. J. Pister, “Smart dust: Communicating with a cubic-millimeter computer,” *Computer*, vol. 34, pp. 44–51, January 2001. [Online]. Available: <http://dx.doi.org/10.1109/2.963443>
- [3] Ricardo Silva, Jorge Sá Silva and Fernando Boavida, “Evaluating 6LoWPAN implementations in WSNs,” Department of Informatics Engineering – University of Coimbra, Tech. Rep., 2009.
- [4] U. Sarwar, G. S. Rao, Z. Suryady, and R. Khoshdelniat, “A Comparative Study on Available IPv6 Platforms for Wireless Sensor Network,” in *International Conference on Wireless Communications and Mobile Computing (ICWCMC 2010)*, 2010.
- [5] Texas Instruments, “MSP430 Ultra-Low Power 16-Bit Microcontrollers,” http://www.ti.com/lscs/ti/microcontroller/16-bit_msp430/overview.page [Online; accessed 14-October-2011].

Poster Abstract: A Framework for a Modal Wireless Sensor Network

Paulo Martins^{*} [†], Ronaldo Menezes[†], Ieda Hidalgo[‡], Udo Fritzke Jr.[§]

* Chaminade University

Email: pmartins@chaminade.edu

† Florida Institute of Technology (FIT)

Email: rmenezes@cs.fit.edu

‡ Universidade Estadual de Campinas (Unicamp)

Email: iedahidalgo@ft.unicamp.br

§ PUC Minas

Email: udo@pucpcaldas.br

Abstract—We propose and demonstrate an agent-based mode-change framework, consisting of a model and a protocol for implementing modes of operation in wireless sensor networks (WSN). A mode-change framework allows a WSN to change from one mode to another according to changes in the environment, thus turning applications more adaptive. The framework uses mobile agents to implement modes of operation as well as to manage the transitions from one mode to another. We demonstrate the feasibility of the approach on a real WSN prototype, using network latencies as the key performance evaluation criteria.

I. INTRODUCTION

WSNs have recently attracted significant attention due to their potential to bring solutions to many areas of our economy and life. Like other systems, these networks can benefit from the partitioning of an application into modes of operation. Several application domains ask for systems that can adapt their behavior at run-time by changing their operating mode. Modes of operation are suitable to WSN applications that demand flexibility, i.e. a change of behavior in response to external events. Adaptive, reactive mode changes, where the system autonomously changes mode, may become a critical requirement for the next generation of flexible WSNs instead of an optional feature. A mode change can be requested for several reasons. For example, the system might need to switch to an emergency mode in order to adapt its behavior to changed conditions in the environment. One advantage of using modes is to allow the system to extend its functionality. Modes of operation can be discarded, freeing resources to new modes, which implement new functionality. When the environment changes, the system can be reconfigured by a mode change to optimally reallocate its computational resources to the new requirements. Reconfigurability and adaptability can also be achieved without modes, but the resulting system is not scalable.

In this paper, we introduce a framework that allows the integration of modes of operation in WSNs. The framework combines agents and modes of operation as fundamental programming models for self-adaptive applications in WSNs. A mode is implemented by a set of agents. Agents are small

programs that are deployed (injected) to the network from a base station, and then can clone themselves, or move from node to node carrying out specific tasks, such as searching for information. A mode-change model and protocol is required, allowing the dynamic reconfiguration of the layout of agents at any time. This reconfiguration occurs through the proper elimination of unwanted agents, the maintenance of the ones that are needed by the application, and the injection of new agents which are required but not currently present in the network.

II. THE MODE-CHANGE FRAMEWORK

The mode-change model consists of six types of agents: *Old-Mode Completed Agents* are under execution when a mode-change command arrive and are allowed to complete during the mode change; *Old-Mode Aborted Agents* are terminated at the time a mode-change command arrives at a node; *Wholly-New Agents* implement the behavior required by the system in the new mode. These agents are not present in the mode-changing node, and therefore need to be injected in the network from a base station and move to the target node where they need to execute; *Unchanged Agents* execute before, during and after the transition is complete. This class of agents preserves all their timing attributes and execution code throughout the mode change; *Changed Agents* implement the system's changed behavior. The period (if periodic agents), execution time, priority (if a priority-based system), or code are examples of parameters that can be changed in an agent [1]; the *Mode-Manager Agent* is responsible for the dissemination of a mode-change command (MCC) across the network. The mode-change protocol consists of the following steps: 1) Generation of a mode-change request (MCR), which is carried out from a requesting node to the base station using either an agent or a simple message; 2) Arbitration of the request (MCA), where the request may be accepted, rejected or deferred. A *mode arbiter* entity manages these requests and if one is accepted, the next mode of operation is selected; 3) Generation of a mode-change command (MCC), which travels from the base station to the target nodes, i.e. the part

of the network where the mode change is to be executed; 4) Execution of the mode change (MCE), where agents are aborted, completed, changed or invoked conforming to the mode-change specifications.

One of the most critical performance attributes in a modal WSN is the latency of a mode change. It needs to be minimized because during the transition the system delivers only partial functionality. It depends on many network attributes such as topology, number of agents and network exploration algorithm. The latency can be measured by the cumulative delays incurred by the mode-change request, its arbitration time, the mode-change command, and the longest mode-change execution delay among all changing nodes.

The mode-change model and protocol were implemented in a prototype network as a software layer on top of Agilla [2] and the TinyOS operating system. The system uses Telos B motes from Crossbow. The network topology consists of six clusters and each cluster is composed of a cluster head (CH) and five cluster members (CM). The cluster heads perform aggregation and also serve as a bus (or path) for the mode-change traffic (i.e. MCR and MCC) across the network.

III. EVALUATION

We evaluate the model and protocol using our prototype network. At the base station, messages received from motes were time stamped, facilitating the collection of information about the latency of mode changes. Measurements were repeated for each event 20 times and these times were then averaged.

The first goal of the evaluation was to assess the feasibility of the approach on a real network and to verify the correctness of the functional specification, i.e. to test the ability of the network to implement modes of operation and perform mode changes. The second goal of the evaluation was to collect timing information and to study the mode-change latency in order to minimize it (future paper). In addition to studying latencies, we looked at the scalability of the approach, i.e. how the framework behaves in an incrementally larger network. The experiment consisted in an acyclic mode change from a weather monitoring mode (WMM) to a perimeter defense mode (PDM). It was modeled by a completed old-mode agent running the WMM that, upon the arrival of a MCC from the base station, is replaced by a wholly new agent, which implements the PDM.

In this experiment, the mode-change request is generated at the base station. The total mode-change latency has three components: 1) the time it takes for a node to receive the MCC; 2) the delay to complete the old-mode agent, and 3) the time it takes to receive the wholly-new agent from the base station and run it in the mode-changing node.

In the graph shown in Fig. 1, each bar represents the mode-change latency for the specified node. The nodes with lower integer IDs are physically closer to the base station. Each bar has three components representing the three different measurements taken for each node: the lower portion of the bar (dark grey) represents the latency from the time the mode change began (when the first cluster head received the MCC

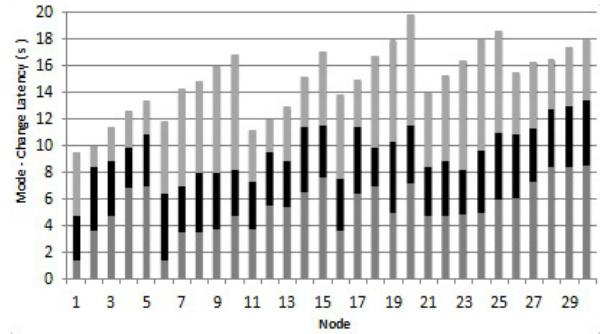


Fig. 1. Latency of a mode change on each node

message) to the time the node received the MCC message. The middle portion of the bar (black) represents the time taken to complete the old-mode agent. Finally, the top portion of the bar (light grey) represents the time taken to receive and run the wholly new agent.

The MCC agent hops from the base station to each and every cluster head. Once in a cluster head, it sends a MCC message to each cluster member. This process of distribution of the MCC command gives the peculiar shape to the graphs (ascending bars) within each cluster. The first node in the cluster spends the least amount of time waiting for the MCC message, leading to a small component of the total latency, and the last node in the cluster spends the largest amount, leading to a larger latency. Since each node takes approximately the same amount of time to complete the old-mode agent, the graphs shows a consistent latency throughout each node.

Extending the size of the network does have reduced impact on the overall timing behavior of the network. To some extent this is due to the topology used: The cluster heads work as a main bus between each cluster and the base station, whereby the mode-change traffic can propagate without the interference from the application traffic that is confined within the clusters. In addition to that, because the MCC propagates across the network through the cluster heads, the mode changes initiate approximately by the same time within each cluster, causing the overall operation to be parallel.

IV. CONCLUSION

In conclusion, this paper is a first step towards demonstrating that the mode-change framework introduced lays a solid ground to the design and implementation of flexible-modal wireless sensor networks. It also provides a background for further research of mode changes in more complex applications such as those in real-time and embedded systems.

REFERENCES

- [1] P. Martins (Pedro) and A. Burns, "Schedulability Analysis of Mode Changes in Flexible Real-Time Systems", In Proceedings of the 10th Euromicro Conference on Real Time Systems (ECRTS'98), pp. 172 - 179 , Jun.1998.
- [2] C.L. Fok, G.C. Roman, and C. Lu, "Rapid Development and Flexible Deployment of Adaptive Wireless Sensor Network Applications", In Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05), pp. 653 - 662, Jun. 2005.

Poster Abstract: Sensor Data Collection Using Constructive Interference Flooding

Chao Gao, Makoto Suzuki, Takuto Kuroiwa, Hiroyuki Morikawa

Research Center for Advanced Science and Technology, The University of Tokyo
Email: {kocho, makoto, kuroiwa, mori} @mlab.t.u-tokyo.ac.jp

Abstract—In this paper we propose a flooding-based data collection protocol by taking advantage of constructive interference (CI). Different from conventional approaches for data collection in wireless sensor networks, which are based on unicast forwarding, our protocol is based on CI flooding without any routing topology. We analyze the performance of applying CI to ZigBee under the influence of frequency offset with a GNU Radio simulation. Then we design and implement an initial periodic data collection protocol based on CI flooding, and show that both low power consumption and high reliability can be achieved.

I. INTRODUCTION

In wireless sensor networks (WSNs), data collection is one of the most fundamental protocols, which allows for data from multiple sources to be delivered to common sinks. Different from conventional unicast forwarding-based collection protocols, e.g. CTP[1] and Dozer[2], this paper proposes a CI flooding-based collection protocol with both low power consumption and high reliability.

There have been many research efforts addressing data collection with low power consumption. They are almost all unicast forwarding-based protocols. However, our proposed collection protocol takes advantage of CI flooding[3]. The collection protocol does not require routing protocols which consume battery in wireless sensor networks with lossy links. Besides, flooding makes scheduling methods simplified while a complex scheduling method is required in multi-hop TDMA configuration because routing topology has to be taken into consideration. What is more, back-off time for avoiding collisions and long preambles for awakening listeners are also not required so that even less battery energy is consumed.

The rest of the paper is organized as follows: Section 2 introduces CI flooding and performs a GNU Radio simulation to show the performance of generating CI under the influence of frequency offset. Initial protocol design and duty cycle analysis are described in Section 3. In Section 4, experimental evaluations are discussed. Finally, we make a conclusion in Section 5.

II. CI FLOODING

Flooding is widely used in WSNs for command dissemination, time synchronization, routing establishment and so on due to its simplicity. Similarly, flooding can be beneficial to data collection because an additional routing protocol is not required. On the other hand, conventional flooding method mainly has two problems by which it is not an efficient

approach for data collection. One problem is its low reliability. Collisions happen frequently and packet broadcasting makes it impossible to obtain *ack* in the link layer. The other problem is that it unnecessarily consumes energy since the number of nodes relaying packets increases.

To overcome these problems and efficiently apply flooding to data collection, we take advantage of CI, an efficient approach for flooding presented in [3]. It allows multiple transmitters to concurrently broadcast packets without collision when they transmit the same packet. What is more, nodes can have their time synchronized by the CI flooding itself and backoff time and long preamble are not required. It still has an extremely small duty cycle despite of the increased number of nodes relay packets.

CI flooding can be effective only if the conditions for generating CI are satisfied. [3] has performed extensive simulations to show that packet SFD detection rate is over 75% when the temporal displacement of two concurrent IEEE 802.15.4 transmitters is smaller than $0.5\mu s$. However, packet delivery rate and the influences of other conditions, such as frequency offset, are unknown. Therefore, we further perform simulations with GNU Radio using ZigBee modules[4].

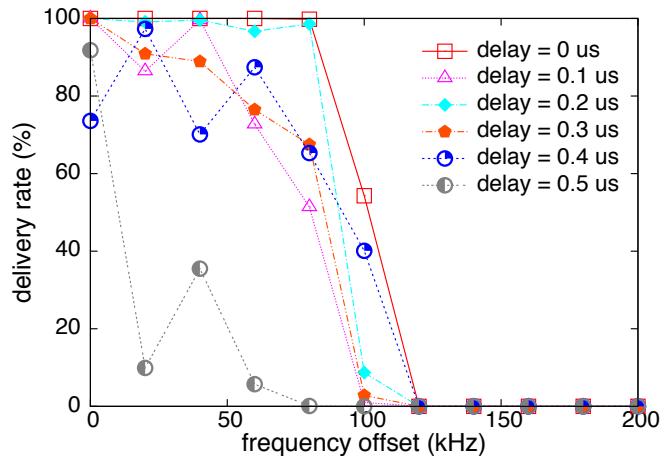


Fig. 1. frequency offset vs delivery rate (no noise, 2 nodes)

As shown in Fig. 1, packet delivery rate changes with the frequency offset of multiple concurrent transmitters. Compared with the maximum frequency offset permitted of 96kHz by IEEE 802.15.4, when the frequency offset is smaller than

70kHz and the temporal displacement is smaller than $0.4\mu s$, packet delivery rate is over 70%.

III. INITIAL DESIGN

We design and implement an initial periodic data collection protocol using CI flooding, and analyze the duty cycle to theoretically show the feasibility of low power consumption.

A. Protocol Design

In order to apply CI flooding for periodic data collection, a scheduling method is required to determine when and which node initiates a flooding, because during one flooding only one packet can be delivered through the network.

Assuming that the node number is K , each node has a unique node ID varying from 1 to K . As shown in Fig. 2, the scheduling method is that each node initiates a packet in the order of its node ID. For instance, the node with ID = 1 sends its packets when the cyclic counter $c = 1$.

An example of the process of CI flooding is illustrated in Fig. 2, i.e. Node 1 as a flooding initiator sends a packet to Node 4 as a sink node. In detail, before Node 1 transmits the packet, Node 2, 3 and 4 wake up with a time margin of T_{guard} . Then Node 2, 3 receive the packet concurrently in time slot $t = 0$, and in $t = 1$ they concurrently transmit the packet which interferes constructively so that the packet can be correctly received by Node 4. For improving reliability, each node rebroadcasts the packet for N_{tx} times ($N_{tx} = 2$ in this example) and then turns radio off.

B. DUTY CYCLE ANALYSIS

The duty cycle d_i of node i , can be described as $d_i = \frac{T_{radio,i}}{T_{ipi}}$, where $\overline{T_{radio,i}}$ is the average of $T_{radio,i}$, which is the radio-on time of node i during one flooding. $T_{radio,i}$ can be described as

$$T_{radio,i,j} = \begin{cases} (2N_{tx} - 1)T_{slot}, & (i = j) \\ T_{guard} + (2N_{tx} + h_{i,j} - 1)T_{slot}, & (i \neq j) \\ T_{default}, & (\text{packet not received}) \end{cases}$$

where T_{slot} is the length of the time slot, $h_{i,j}$ is the number of hops between node i and the initiator j , $T_{default}$ is the

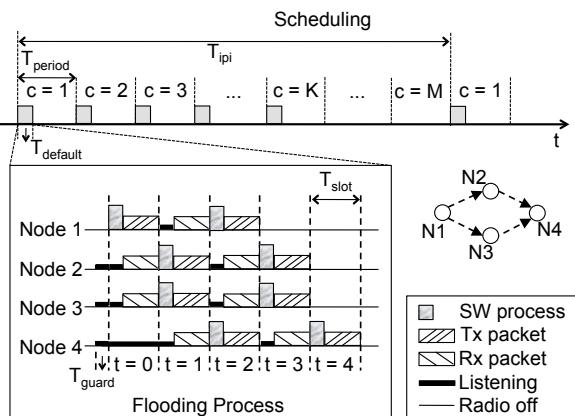


Fig. 2. Process of data collection

default maximum channel listening time. $T_{radio,i}$ of a receiver increases as the hop number $h_{i,j}$ increases when it receives packets successfully. Otherwise, its $T_{radio,i}$ is the default listening time $T_{default}$.

If a node transmits or receives packets without losses, its duty cycle is

$$d_i = \frac{\{(2N_{tx} - 1)K + (K - 1)\overline{h}_i\} T_{slot} + (K - 1)T_{guard}}{T_{ipi}}$$

where \overline{h}_i is the average number of hops between node i and the other nodes, T_{ipi} is the inter-packet interval (IPI) of a node.

According to the CTP implementation settings on Motelab with BoX-1s link layer[1], by which CTP has a median duty cycle of 2.8%, our proposed protocol achieves a theoretical duty cycle of 0.6% when $N_{tx} = 3$, $T_{guard} = 0.977ms$ and $T_{slot} = 1.3ms$ (packet size = 40Byte).

IV. EXPERIMENTAL EVALUATION

Our periodic data collection protocol is implemented on a local testbed and MoteLab. Parameter settings are as follows: listening interval $T_{period} = 1s$, $T_{default} = 15.625ms$, $T_{guard} = 0.977ms$ and Tx power = -10dBm. Packet delivery rate and duty cycle are evaluated as shown in Table 1. It can be concluded that high delivery rate can be achieved when N_{tx} is not large, though delivery rate decreases slightly as the number of hops increases. What is more, although duty cycle increases as packet size or rebroadcast times increases, our proposed protocol achieves a very small duty cycle.

Testbed	Nodes (K)	Max Hop	IPI (T_ipi)	Packet Size	Rebroadcast Times (N_tx)	Avg. Delivery	Duty Cycle
Local	6	3	10s	40B	2	99.8%	0.4%
Local	6	3	10s	72B	2	99.5%	0.7%
Local	15	7	20s	40B	2	97.5%	0.6%
Local	15	7	20s	40B	3	98.4%	0.8%
Local	15	7	20s	72B	2	98.4%	0.9%
Local	15	7	20s	72B	3	98.6%	1.2%
MoteLab	47	6	50s	24B	2	96.8%	0.8%
MoteLab	47	6	50s	40B	3	98.0%	1.3%

Table 1 Experimental Results

V. CONCLUSION

This paper proposed an efficient data collection protocol using CI flooding. A GNU Radio simulation is performed to evaluate the performance of applying CI to ZigBee under the influence of frequency offset. We also design and implement an initial periodical data collection protocol on the local testbed and MoteLab, showing that both low power consumption and high reliability can be achieved.

REFERENCES

- [1] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss and P. Levis, Collection Tree Protocol, Proceedings of SenSys, 2009.
- [2] N. Burri, P. von Rickenbach, and R. Wattenhofer, Dozer: Ultra-Low Power Data Gathering in Sensor Networks, Proceedings of IPSN, 2007.
- [3] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh, Efficient Network Flooding and Time Synchronization with Glossy, Proceedings of IPSN, 2011.
- [4] UCLA ZigBee PHY, <https://www.cgran.org/wiki/UCLAZigBee>

Poster Abstract: Coordination For TDMA Operation In WSNs: Comparison Between Centralized And Distributed Mechanisms

Antonio Vittorioso[†], Dujdow Buranapanichkit[†], Giancarlo Fortino[‡] and Yiannis Andreopoulos[†]

[†] University College London [‡] University of Calabria

Abstract—We compare centralized and distributed approaches for the coordination of transmission slots of wireless sensors for collision-free time division multiple access (TDMA) operation. We focus on the guaranteed time slot (GTS) mechanisms of IEEE 802.15.4 standard as an example of centralized coordination and on the DESYNC algorithm (and its time-frequency extension) as an example instantiation of distributed coordination. Our work aims to examine distributed coordination mechanisms as alternatives to the centralized ones. Distributed coordination is by nature more robust and scalable as it does not have a single point of failure and can be scaled to multi-channel operation. However, our early results indicate that it requires higher startup delay in order to converge to the steady state. Therefore, it appears that distributed coordination can be used in contexts where the number of nodes and the network topology does not change very rapidly.

I. INTRODUCTION

TIME synchronization is a critical component in any distributed system. In wireless sensor networks, a confluence of factors makes flexible and robust time synchronization particularly important, while simultaneously making it more difficult to achieve than in traditional networks. Beyond the well-known coordinator-based approaches for synchronization in TDMA operation that are standardized within the IEEE802.15.4 [1], new solutions, based on the principle of reactive listening, have been proposed: wireless sensors autonomously schedule their transmission times by reacting to beacon or “fire” message broadcasted by each node without requiring a central coordinator node [2]. Furthermore, for centralized and distributed TDMA, multi-channel MAC protocols have appeared [3]-[5], which aim for load balancing via time-frequency division multiple access. Following this principle, we proposed a distributed algorithm, the DTFDMA [4] (distributed time-frequency division multiple access), a low complex scheme that allows for self-organization of an arbitrary number of sensors in a number of channels.

The contribution of this work is in the practical comparison of GTS and DTFDMA in order to evaluate the effective bandwidth achievable derived via measurements with real TinyOS-based sensors using the CC2420 transceiver.

II. AN OVERVIEW TO GTS AND DTFDMA

In this section we briefly describe the two mechanisms analyzed.

A. Centralized GTS

In this case, the network has exactly one PAN coordinator, which is the primary controller responsible for PAN identifier and device synchronization. The 802.15.4 PANs can either be *nonbeacon-enabled* or *beacon-enabled*. In nonbeacon mode PAN frames are transmitted according to an unslotted CSMA-CA algorithm and, if the channel is not idle, the devices defers the transmission for a random period.

In beacon-enabled mode, the coordinator periodically transmits beacons which mark the beginning of a superframe, as depicted in Fig. 1.

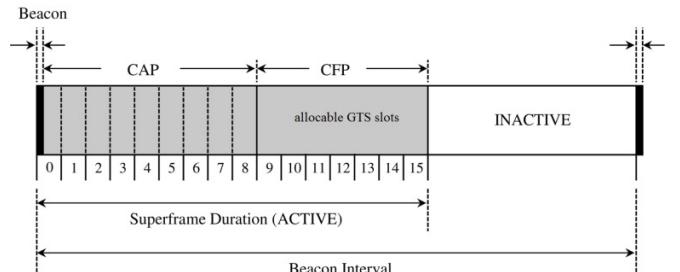


Figure 1. Superframe in IEEE802.15.4 with beacon enabled mode.

The contention access period (CAP) is followed by an optional contention-free period (CFP), which is portioned in seven guaranteed time slots (GTS). GTSs are allocated dynamically and the corresponding time interval can be used exclusively to transmit packets. In the inactive period, all nodes can sleep (preserve energy) and achieve low duty cycle.

The superframe duration (SD) and beacon interval (BI) are calculated by:

$$BI = aBaseSuperframeDuration \times 2^{BO} \quad (1)$$

$$SD = aBaseSuperframeDuration \times 2^{SO} \quad (2)$$

where: $0 \leq SO \leq BO \leq 14$ and $aBaseSuperframeDuration$ defines the minimum length of the superframe (the default setting is 15.36ms).

B. Distributed TFDMA

Considering a network of fully-connected wireless sensors, it is possible to use the DESYNC protocol [2] in order to make each node adapt its next fire (or beacon) message time to be in the middle between the fire time of its predecessor and its successor (as shown in Fig. 2).

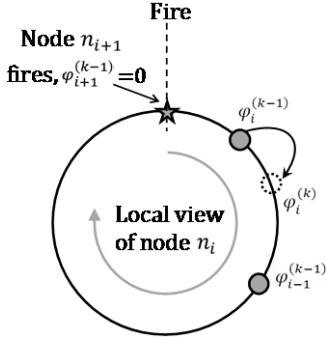


Figure 2: Fire message adaption of node n_i

During the k th iteration (period), each node n_i adapts its phase (fire time) φ_i by:

$$\varphi_i^{(k)} = (1 - \alpha)\varphi_i^{(k-1)} + \alpha \frac{\varphi_{i-1}^{(k-1)} + \varphi_{i+1}^{(k-1)}}{2} \quad (3)$$

where $\alpha \in (0,1)$ is a parameter that scales how far n_i moves from its current fire phase ($\varphi_i^{(k-1)}$) toward the desired midpoint. A near-optimal TDMA behavior is obtained after k_{ss} periods where all fire messages are periodic with:

$$|\varphi_i^{(k_{ss})} - \varphi_i^{(k_{ss}-1)}| < q_{ss} \quad (4)$$

at the time the update of (3) is performed, with q_{ss} a preset threshold.

Distributed TFDMA attempts to split the number of nodes in order to balance them into all channels [4] and to increase throughput. At the beginning, each node chooses a channel randomly and utilizes the DESYNC algorithm. After the steady state of the TDMA, each node jumps in a new channel (either by increasing or decreasing the channel number) and checks the number of nodes in the new channel. If there are fewer nodes present than in the old channel, it decides to stay; otherwise the node will switch back to the old channel. This can be shown to converge to balanced number of nodes in each channel after a few iterations [4].

III. EXPERIMENTAL RESULTS

To evaluate the achievable bandwidth with the two mechanisms proposed we used TinyOS wireless sensors (TelosB and iMote2). All messages use the TinyOS standard while the data message contains 96-bytes payloads. For distributed TFDMA we use 8 nodes and the maximum data rate with single transmitter and receiver was experimentally found to be 137.4 kbps. The results for this case are given in Table 1.

The normalized throughput defines the ratio between the total throughput and the maximum measured throughput. The

highest and lowest throughput per individual node are represented by the “max” and “min” rows of Table 1.

Total Channels	2	1
Tot. throughput (kbps)	271.9	133.4
Normalized, %	194.1	92.4
Max per node (kbps)	34.1	16.7
Min per node (kbps)	33.7	16.5
Message loss (%)	0.01	0.01

Table 1. Results under the distributed TFDMA.

For GTS, we examine the effect of the beacon order on data rate. GTS allows for (up to) 7 stations to join, which creates a total of 8 nodes including the coordinator. The results are shown in Table 2. Evidently, for small numbers of nodes (5) and a single channel, the total throughput and the throughput per node can be higher than the case of DTFDMA. DTFDMA becomes superior for higher number of nodes and more channels. However, while the setup delay for GTS is equal to one superframe duration, DTFDMA may require a few seconds to converge to steady state [4].

Beacon Order	5	6	7
Tot. throughput (Kbps)	177.0	89.6	43.5
Average per node (Kbps)	25.3	12.8	6.2
Message loss (%)	0.00	0.00	0.00

Table 2. Results under the centralized GTS.

IV. CONCLUSION

Distributed TFDMA’s throughput with multiple channels can be significantly higher in comparison to the GTS scheme, furthermore no centralized coordinator is required. Further work will aim to improve throughput and connectivity between the nodes by incorporating DTFDMA with channel hopping, i.e. a new algorithm that would combine DESYNC’s distributed beaconing system with a dynamic scheduling pattern in the available channels.

ACKNOWLEDGMENT

This work has been partially supported by the EC under contract FP7-2007-IST-2-224053 (CONET project) and by a PhD scholarship from the Government of Thailand (Dujdow Buranapanichkit).

REFERENCES

- [1] A. Koubaa, M. Alves and E. Tovar, “GTS allocation analysis in IEEE 802.15.4 for real-time wireless sensor networks,” *Proc. IEEE 20th Int. Symp. On IPDPS*, Apr. 2006.
- [2] J. Degesys *et al*, “DESYNC: Self-organizing desynchronization and TDMA on wireless sensor networks,” *Proc. IEEE IPSN*, pp. 11-20, 2007
- [3] H. K. Le *et al*, “A practical multi-channel media access control protocol for wireless sensor networks,” *Proc. IEEE IPSN*, pp. 70-81, 2008.
- [4] D. Buranapanichkit, A. Vittorioso, G. Fortino and Y. Andreopoulos, “Performance comparison of Centralized and Distributed Coordination for TDMA operation in WSNs,” *London Communication Symposium*, Sept. 2011.
- [5] X. Lin and S. B. Rasool, “Distributed and provably efficient algorithms for joint channel-assignment, scheduling, and routing in multichannel ad hoc wireless networks,” *IEEE/ACM Trans. Netw.*, vol.17, no.6, Dec. 2009.

Poster Abstract: Information Quality Aware Transport Protocol for Wireless Sensor Networks

Vinay Sachidananda, Abdelmajid Khelil and Neeraj Suri
TU Darmstadt, Hochschulstr. 10, 64289 Darmstadt, Germany
{vinay,khelil,suri}@informatik.tu-darmstadt.de

Abstract—A key task in wireless sensor networks is to deliver information from the sensor nodes to the sink. Many applications require the delivery to be reliable and timely. However, increasing reliability/timeliness comes at the cost of higher energy consumption as in both cases additional messages have to be sent: Retransmissions to increase reliability and information delivery via second, faster path to ensure timeliness. Existing transport protocols either over- or under-provide reliability and/or timeliness and lack optimized efficiency. In this poster, we sketch a new approach to tune reliability and timeliness in composition for a maximized efficiency.

I. INTRODUCTION

In Wireless Sensor Networks (WSNs) delivering the gathered information with the user required quality is a key concern. To satisfy the user required quality, we should carefully design the functional blocks. In particular, the transport scheme should ensure reliable and timely delivery of the information to the sink.

To reduce deployment costs, WSNs are required to serve multi-users for multi-purposes. For instance, the purpose of a WSN deployment may suddenly need to be changed. For instance, upon a catastrophic event, the WSN should support rescue operation and stop unnecessary monitoring activities. In smart cities and/or rural areas, public WSNs should deliver different information entities for varied authorities or users. Multi-users may use multi-sinks with possibly different actuation plans to react on the information delivered by the WSN. In a future wireless automation scenario such as the smart grid, different sinks may rely on wireless sensor information to control wireless actuators such as valves and switches. Common to all these observations is that different information entities are generated and should be transported to their corresponding users/sinks. Typically users require different requirements on transport timeliness and reliability. Timeliness requirements, may range from strict/real time to soft deadlines that can vary from seconds to minutes to hours. Varied WSN users usually require best effort reliability with different levels of efficiency. Best effort reliability requirements can be expressed in message delivery success rate or ratio of event detection false positives or false negatives. Achieving the best possible timeliness and reliability is related to a large overhead regarding resources, is particularly because sensor nodes rely on batteries. A higher reliability usually is achieved through a higher number of retransmissions resulting

in a higher energy and bandwidth overhead. Timeliness may require path splitting instead of simple retransmissions on the same path, thus, causing higher traffic related with higher energy and bandwidth overhead. Hence, besides attaining the required quality levels, it is indispensable to maximize energy/bandwidth efficiency. To achieve a maximized efficiency, reliability and timeliness should be tuned in combination to avoid contradictory decisions.

Available approaches usually target the best possible reliability or timeliness while optimizing efficiency [7]. As it is not always required to provide best effort reliability or timeliness, it is challenging to just provide the user required performance. Some of the existing works just concentrate on timeliness requirements by providing end-to-end timeliness and neglecting reliability and also the key factor of tuning both reliability and timeliness in composition [5], [4], [2]. In [1], authors presents a technique that just meets the required reliability. However, in [1] the attribute timeliness and tuning both reliability and timeliness in composition are neglected. Moreover, some works which consider both reliability and timeliness [3], [6], overlook to provide the user defined timeliness and reliability and also neglecting to tune both reliability and timeliness in combination. In our work, we plan to address this tradeoff by providing the user required varying/evolvable reliability and timeliness levels while maximizing efficiency regarding energy/bandwidth.

Achieving tunability of both transport reliability and timeliness while maximizing efficiency requires a sophisticated tradeoff technique, which is the main objective of this work. In this poster, we discuss the challenges to develop the required algorithms to provide such technique for generalized WSNs. Then, we present our methodology and a brief overview of our planned research.

II. TRADING TIMELINESS AND RELIABILITY FOR EFFICIENCY

Before sketching our approach to provide for tunable reliability and timeliness, we briefly discuss the main challenges that we face.

A. The Key Research Challenges

We consider a WSN composed of N static homogeneous sensor nodes and one sink. The WSN executes different data collection applications for different users with varied requirements on reliability and timeliness. Typically, each node is

Research supported in part by DFG GRK 1362 (GKmM).

equipped with short range wireless communication, and shows limited processing, storage and energy capabilities. Usually, the network conditions are dynamic and impose various perturbations with varying severity levels. The key challenge in our work is satisfying the end user's evolving requirements despite the numerous communication level perturbations such as message loss due to congestion, collisions and contentions. As we are aiming in ensuring tunable timeliness and reliability in composition, we encounter the problem of defining the value of message time out and detecting message loss. Furthermore, to satisfy the user required reliability within the tolerated timeliness, finding the suitable number of retransmission and to achieve fully distributed solution needs a sophisticated mechanism. Achieving the maximized efficiency contradicts the design of the composite tunability.

B. A Road Map Towards Composite Tunability

In order to allow for a fully distributed solution, we propose to make per hop decisions. For instance, it has been proven that per hop reliability in WSN outperforms the end-to-end acknowledgment and retransmissions [1]. Accordingly, [1] proposed to conduct hop-by-hop retransmissions towards the sink. Similarly, we design a timeliness strategy on hop basis. Our approach aims in providing the desired application reliability with dynamic network conditions by adopting the adaptive retransmission techniques for tunable reliability. In addition, we couple the decision on how many retransmissions per hop to the allowed tolerated link latency. In the best case, all required retransmissions can be performed within the tolerated link latency on all hops along the path.

Now, we provide an overview on how our planned solution progresses towards tuning both reliability and timeliness for information transport. In Fig. 1, we illustrate three typical scenarios for information transport, i.e., the information entities sent by S_1 , S_2 and S_3 . These scenarios are the drivers to develop our algorithms.

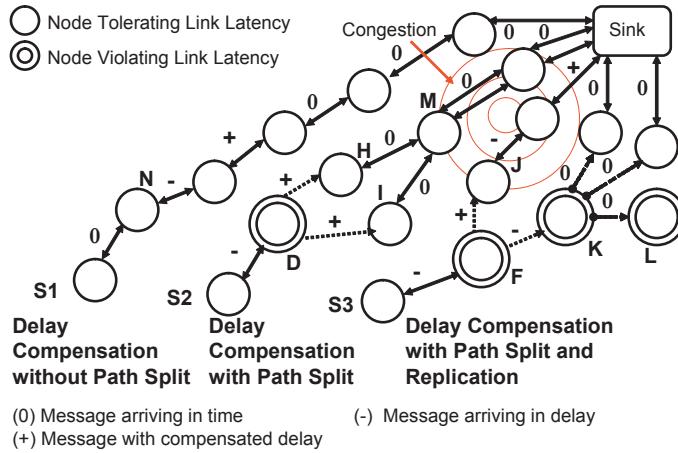


Fig. 1. Three Illustrative Scenarios for the Proposed Information Transport.

Delay Compensation without Path Split: Consider S_1 that generates an information and sends it to the sink. We assume

that Node N requires a number of retransmissions so that the tolerated link latency is violated. If the caused delay does not exceed a portion (say α) of the tolerated link latency of next hop, we propose a scheme for delay compensation without path split. This strategy ensures strict timeliness notion while giving the best effort reliability.

Delay Compensation with Path Split: We refer to path split by the fact of sending the same message for two neighboring nodes that are closer to the sink. Consider S_2 has made delay compensation, however, Node D can not conduct delay compensation anymore as the tolerated link latency exceeds by a delay larger than α . Accordingly, we develop a new mechanism that improves the reliability within the required timeliness.

Delay Compensation with Path Replication: We refer to path replication by the fact of sending the same message for three or more neighboring nodes that are closer to the sink. Consider the scenario of S_3 . Node F requires delay compensation and path split to two neighboring sensor nodes J and K . However, delay compensation (as the caused exceeds α) and path splitting are not sufficient at Node K . Hence, Node K has to conduct path replication to three neighbors (the number three is based on the remaining number of retransmissions).

III. CONCLUSIONS AND ONGOING WORK

Through this road map, we have briefed on how we plan to achieve the tuning of timeliness and reliability as per the application requirements. We plan to introduce the tunable timeliness algorithm which calculates the tolerated link latency, compensates delay and path split for the intermediate hops. The on going optimal solution combines the re-transmission approach meeting the tolerated link latencies and the path replication approach when the tolerated link latency are violated to satisfy the reliability and timeliness in composition.

In our ongoing work we aim to provide:

- The tunable timeliness algorithm that provides with best effort reliability.
- The reliability and timeliness algorithm that provides tuning in composition.

REFERENCES

- [1] F. K. Shaikh et al., Generic Information Transport for Wireless Sensor Networks. In Proc. of SUTC, 2010, pp. 27 - 34.
- [2] K. Akkaya and M. Younis, An energy-aware QoS routing protocol for wireless sensor networks. In Proc. of DCS Workshops, 2003, pp. 710 - 715.
- [3] E. Felemban et al., Probabilistic QoS guarantee in reliability and timeliness domains in wireless sensor networks. In Proc. INFOCOM, 2005, pp. 2646 - 2657.
- [4] B. Jiang et al., CFlood: A Constrained Flooding Protocol for Real-Time Data Delivery in Wireless Sensor Networks. In Proc. of SSS, 2009, pp. 413-427.
- [5] T. He et al., SPEED: A Stateless Protocol for Real-Time Communication in Sensor Networks. In Proc. of DCS, 2003, pp. 46 - 55.
- [6] Huang, X., Fang, Y., Multiconstrained qos multipath routing in wireless sensor networks. In Proc. of Wirel. Netw, 2008, pp. 465 - 478.
- [7] Suriyachai, P. et al. A Survey of MAC Protocols for Mission-Critical Applications in Wireless Sensor Networks. In Proc. of Communications Surveys and Tutorials, IEEE, 2011, pp. (99). 1 -25.

Poster Abstract: Packet Analyser for IEEE 802.15.4 Networks

Lubomir Mraz, Vladimir Cervenka, Milan Simek

Department of Telecommunications, Brno University of Technology, Czech Republic
xmrazl00@stud.feeec.vutbr.cz, cervenka.v@phd.feeec.vutbr.cz, simek@feeec.vutbr.cz

Abstract— Deployment, debugging and analysis of wireless sensor networks require a special device, a so-called packet analyser. In this article we present design and implementation of the analyser. We also performed test in order to determine its suitability for analysis of IEEE 802.15.4 networks.

I. INTRODUCTION

Wireless Sensor Networks (WSNs) are becoming a wide spread technology in many application areas. The vast majority of these networks is based on IEEE standard 802.15.4 [1]. The ability to capture and analyze communication over the air is extremely important for the purposes of deployment process of those networks. A packet analyser is a device dedicated to this task. Several commercial analysers are currently available, however, their price is rather high. Our goal is to design and implement a simple, low-cost packet analyser, which can be operated over entire IEEE 802.15.4 ISM band and run on several operating systems i.e. Windows and Linux. We proposed two approaches for an analyser with a local and remote connection. The local connection is implemented via a serial interface. The remote connection is enabled by the analyser with Ethernet interface. The article is organized as follows. The basic principles of the WSN packet analyser are described in the section II. Section III shows packet analyser with serial interface. Its performance evaluation is also given in this section. Section IV is dedicated to the proposal and design of analyser with Ethernet interface. The last section summarizes the conclusions of this paper.

II. WSN PACKET ANALYSER INTRODUCTION

In general, a packet analyser for WSNs can be divided into two parts: a Receiving Unit and a Packet Inspection Unit, see Fig. 1. The receiving unit includes a RF transceiver and a general purpose microcontroller (MCU). The analysed network is not aware about the presence of the packet analyser. It does not send a single bit. Conversely, the RF transceiver is set to the receiving mode, sometimes called promiscuous, where all data transmitted over the air are captured, sent via digital interface to MCU and then processed. Communication between the RF transceiver and MCU is mostly handled by a Serial Peripheral Interface Bus (SPI). In general, the Receiving Unit works as follows: At the beginning, MCU sends control commands to the RF Transceiver. It sets transceiver to the promiscuous mode,

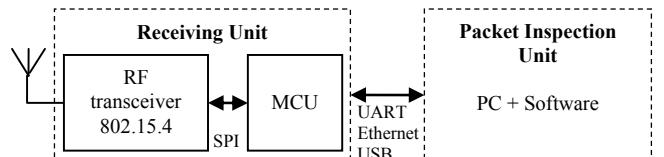


Fig. 1. WSN Packet analyser

configures the channel, optionally turns off the CRC validation etc. Then the RF transceiver starts to sniff every 802.15.4 packet transported over the air. Subsequently, analyser sends received data to the Packet Inspection Unit. The main features of this unit are following: Receive, store, inspect incoming packets and visualize it in a human readable form. This unit is typically implemented as software in a personal computer due to the high computational and memory requirements. We have chosen Wireshark [7] as packet inspection software. It currently includes dissectors practically for any wired network protocol and also for the most popular WSN technologies namely: IEEE 802.15.4, Zigbee and 6LoWPAN. Furthermore other dissectors can be easily added due to the open-source nature of this product.

III. PACKET ANALYSER WITH SERIAL INTERFACE

Based on the information above packet analyser with serial interface has been proposed (Fig. 2). The Receiving Unit is based on Zigbit modules. These modules are mutually pin-to-pin compatible which might be very useful for modular architectures design. It comprises AT86RFxx RF transceiver

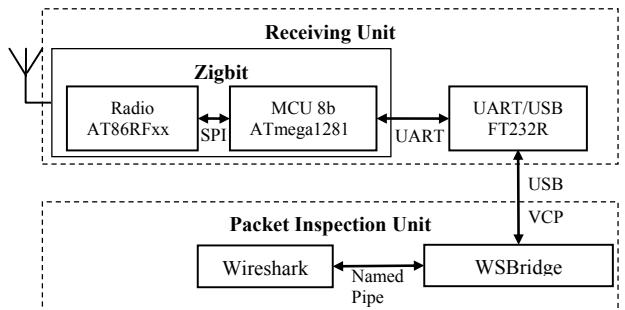


Fig.2. Packet analyser with serial interface

[2], MCU, low speed oscillator and necessary passive components. Currently AT86RFxx chips have one of the best parameters on the market and cover all of frequency bands supported by IEEE 802.15.4. The Zigbit MCU Atmega1281

[3] comes from the ATMEL AVR 8-bit family. It is equipped with UART interface which maximum throughput is 500 kbit/s. For interconnection of the Receiving Unit board and a computer, single-chip convertor FT232R (UART/USB) is applied.

The firmware in MCU is based on an open-source library called *uracoli* [4] written in C. This provides full control over the AT86RFxx transceivers and basic control of API for AVR microcontrollers. We used a slight modification of the sniffer sample firmware which is already included in the *uracoli* package.

Let us investigate the data path in the Receiving Unit. An IEEE 802.15.4 compliant packet is captured over the air by the RF transceiver and further transported to the MCU over SPI. The MCU stores the packet into a packet buffer, adds a timestamp and optionally performs checksum validation. Next, the packet is placed into a UART buffer as soon as the previous packet transmission is finished. From the UART buffer the packet is sent to the computer byte by byte through the FT232R. This stream is captured by a simple utility called WSBridge [5] on the computer side. The main aim of this tool is to provide data forwarding from serial port to a named pipe. Finally Wireshark receives the data from the named pipe and visualise them.

We performed measurements in order to get quantifiable point of view on the analyser performance. For this purpose we selected the 2.4 GHz frequency band, because of the highest possible data rate. This means that maximum packet size (127B, Link layer) defined in the IEEE 802.15.4 at 250 kbit/s takes almost 4ms. The RF transceiver has a RAM buffer equal to 127B. Therefore, only one packet can be stored at a time and each following incoming packet overwrites the previous one. Thus, the packet has to be transferred to the MCU with higher speed than 250 kbit/s. This can be achieved by the SPI interface which is clocked at 4MHz in our prototype. To evaluate a time delay of each stage of the Receiving unit, we sent several IEEE 802.15.4 packets with different size to the air. Measurement results are shown in Table 1. Data can flow continuously through the analyser when the packet wait period is longer than sum of other periods in a data path.

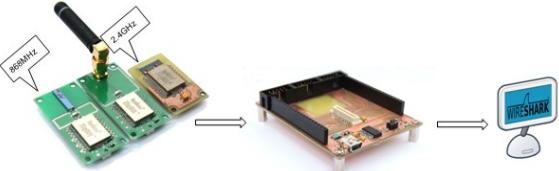


Fig.3. Packet Analyser prototype

Table 1 RECEIVING UNIT DELAY MEASUREMENT

Packet Size [B] (Link layer)	5	30	70	127
T _{packetwait} [μs]	204	1000	2300	4120
T _{RFbuffer→MCUbuffer} [μs]	75	268	576	1010
T _{MCUbuffer→USARTbuffer} [μs]	51	76	118	324

T _{USARTsend} [μs]	248	744	1540	2700
T _{Total} [μs]	374	1088	2234	4034

From the table above we can see a potential problem with the throughput considering small packet sizes. Data show that the UART is bottleneck in the data path. This is mainly caused by two reasons. The first one is the 8B timestamp. For example 5B packet is received by a 250 kbit/s. However, UART has to transfer 14B (Length + Timestamp + Data) which lead that more than 500 kbit/s is required. The problem can be solved by timestamp size reduction from microseconds to milliseconds resolution. This resolution is still enough for common analysis. The second problem comes from an overhead of the UART handling since in this firmware version, the UART is interrupt driven.

From the results it can be concluded that the analyser is able to capture roughly 242 packets with size of 127B (Link layer) per second, which reaches maximum throughput defined by the IEEE 802.15.4 standard.

IV. PACKET ANALYSER WITH ETHERNET INTERFACE

Due to several limitations of the analyser with a serial interface we proposed more powerful solution with an Ethernet interface. In this case, the *Receiving Unit* consists of couple AT86RFxx RF transceivers and ARM-CortexM3 Stellaris® LM3S8962 microcontroller. The Ethernet interface is natively supported in MCU. Analyser configuration can be done remotely over web based interface. Tiny TCP/IP stack called *μIP* might be used to provide this ability [6].

V. CONCLUSION

We proposed two platforms for packet analyser implementation. Our goal is to design and implement multiplatform and low-cost packet analyser which can be operated among various frequency bands. We performed evaluation of the analyser with serial interface. Based on the results it can be concluded that the analyser is suitable for 802.15.4 networks. However, there are some minor issues that need to be solved.

REFERENCES

- [1] IEEE Computer Society, et al. Part 15.4: MAC and PHY Specifications for WPANS, 2nd rev. edition, Available: <http://standards.ieee.org/getieee802/download/802.15.4-2006.pdf>
- [2] Atmel, AT86RF230, Available: http://www.atmel.com/dyn/resources/prod_documents/doc5131.pdf
- [3] ATMELE datasheet ATmega1281, Available: http://www.atmel.com/dyn/resources/prod_documents/doc2549.pdf
- [4] uracoli, The μController Radio Communication Library, Available: <http://www.nongnu.org/uracoli/>
- [5] WSBridge, Freaklabs Open Source wireless, Available: <http://freaklabs.org/index.php/WSBridge.html>
- [6] Open-source uIP stack homepage, Available: <http://www.sics.se/~adam/uip>
- [7] Wireshark homepage, Available: <http://www.wireshark.org/>

Poster Abstract: Effective Capacity Model in the Discrete Time Domain

Yu Chen and Izzat Darwazeh

Department of Electronic and Electrical Engineering, University College London (UCL), London, the U.K.
Email: {y.chen, i.darwazeh}@ee.ucl.ac.uk

Abstract—Queueing delay is a connection-level concept that has been studied over decades and is also a primary Quality of Service (QoS) metric in wireless sensor networks (WSNs). Recently, Effective Capacity (EC) model was proposed and is capable of providing accurate modelling of single-hop queueing delay distribution in wireless environments. However, this model was developed in time domain, which accounts for a discrepancy in sample-based real applications. Therefore, in this poster, we revise and adapt the model into discrete time domain and derive new mathematical formula of Delay Bound Violation Probability (DBVP) in such setting. The results show that the revised EC model gives better performance in DBVP characterisation than the conventional EC model does when the transmission rate is as low as that of wireless sensor nodes.

I. INTRODUCTION

In telecommunications and computer engineering, queueing delay is one of the most important metrics in Quality of Service (QoS) requirements and the research of this topic has been carried out over several decades. Wireless Sensor Networks (WSNs) is an emerging technology but also requires QoS support, however, the question of the behaviour of packet queueing delay in such system remains open.

Effective Capacity (EC) theory was proposed in 2003 by Wu and Negi [1], and can be used to characterise the queueing delay distribution in wireless single-hop scenario. This model uses a fluid traffic model assumption, meaning the sample time is infinitesimally small. Although this assumption may arguably be valid for high-speed transmission, it might not hold well when the transmission rate is low, for example, in the IEEE 802.15.4 standard, the maximum transmission rate of a sensor node is only 250kbps.

The results reported in this poster is a continuous work of Wu's EC model. Specifically, we extend the conventional EC model into discrete time domain and give new mathematical formula of Delay Bound Violation Probability (DBVP), both of which are functions of sample time. Furthermore, our proposed new model is validated via extensive simulation.

The rest of the poster is structured as follows. Section 2 explains the system model and the analysis of delay performance in discrete time domain by extending continuous EC theory. The evaluation and simulation results of our model are presented in Section 3 and Section 4 concludes the poster.

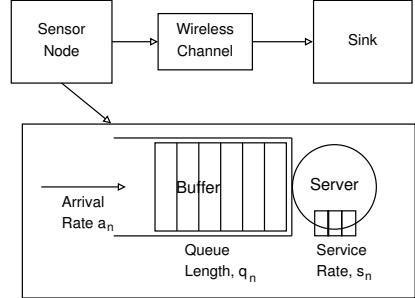


Fig. 1: A Single-hop System Model in a WSN

II. QUEUEING DELAY ANALYSIS

A. System Model

Fig. 1 shows the system model that we study in this poster. The first node is labelled as “sensor node” and is constantly sensing data and sending packets back to the “sink” node. The queueing system inside the sensor node is also shown in this figure and it is sampled by a fixed sample time, T_s , so that a_n stands for the number of packets arrived in the period $[T_s n, T_s(n+1))$ and s_n stands for the number of packets that can be served in the period $[T_s n, T_s(n+1))$. q_n means the queue length at time step, $T_s n$.

Since the purpose of this poster is to understand the *queueing delay* distribution of the packets that are generated at sensor node and destined to the sink node, it is straightforward to have the following lemma that describes the very basic behaviour of packet queueing delay.

Lemma 2.1: If the queueing system inside the sensor node is sampled by a fixed sample step, T_s , the sample space, Ω , of packet queueing delay, W_d , will be

$$\Omega = \{T_s n : n \in \mathbb{N}_0\}$$

where $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$.

Throughout the poster, any symbol with a subscript, d , is used to indicate itself as a function or variable in the discrete time domain and its counterpart, the symbol without this subscript, carries the same meaning but is in the time domain context.

B. Queueing Delay Analysis in the Time Domain

The conventional Effective Capacity (EC) model in the time domain tells us that the probability of W exceeding

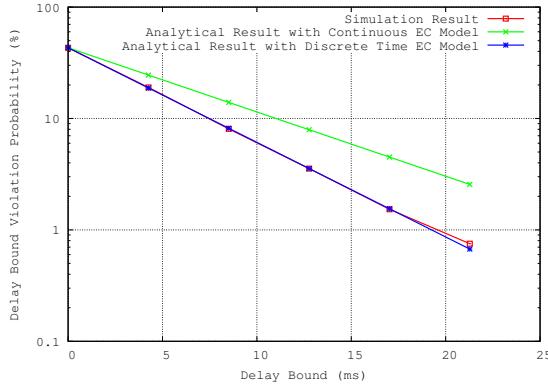


Fig. 2: Simulation and Analytical Results

delay bound W_b , termed Delay Bound Violation Probability (DBVP), will satisfy

$$\Pr(W > W_b) \approx \gamma \exp(-\theta W_b) \quad (1)$$

in most distributions [1]. γ in (1) denotes $\Pr(W > 0)$ that stands for the probability of queueing delay is non-zero as the system has approached to stability.

C. Queueing Delay Analysis in the Discrete Time Domain

Continuous EC model in Section II.B assumes the sample space of the random variable, W , is continuous. However, such assumption does not consider the fact that real systems can only be sampled, which eventually makes the sample space of random variable, W , discrete. With the help of Lemma 1, we are able to adapt DBVP in Section II.B to a discrete version of DBVP, shown in the proposition below.

Proposition 2.2: When a queueing system is sampled by a fixed sample time, T_s , DBVP is given by

$$\Pr(W_d > T_s n) \approx \gamma_d \exp[-\theta_d(T_s n)],$$

where T_s is the sample time of the queueing system.

Proof: According to Lemma 2.1, in a sampled system, there is no chance to have queueing delay in $(T_s n, T_s(n+1))$. Therefore, when we use (1), the following equation always holds,

$$\begin{aligned} \Pr(W_d > W_b) &= \Pr(W_d > T_s \lfloor \frac{W_b}{T_s} \rfloor) \\ &= \gamma_d \exp\left(-\theta_d T_s \lfloor \frac{W_b}{T_s} \rfloor\right). \end{aligned}$$

By substituting $\lfloor \frac{W_b}{T_s} \rfloor$ with n , we have Proposition 2.2. ■

III. EVALUATION AND RESULTS

The wireless channel between the sensor node and sink is principally responsible for causing uncertainty of service rate, s_n . Hence, we introduce the mechanism of ARQ to guarantee the 100% reliability of communications, specifically,

TABLE I: Simulation Parameters

Parameter	Value
Transmission rate	250 kbps
Packet Size	133 Bytes
Average traffic load, λ	125 kbps
Sample time, T_s	4.256 ms
Parameter of Bernoulli distribution, p	0.75
Simulation Duration	60 seconds

the failed packets will be retransmitted (by receiver sending negative feedback to the transmitter) until they are received successfully. We assume that in a certain time period, the wireless channel is stationary and successful transmissions of packets are independent, identically distributed (i.i.d.) random variables and those random variables are Bernoulli distributed with parameter, p . In this poster we set p to be 0.75, which is a typical value in Wireless Sensor communications [2]. The rest values for simulation are listed in Table I.

Fig. 2 shows simulation and analytical results. The X-coordinate is Delay Bounds (the unit is milli seconds) and the Y-coordinate is DBVP. The simulation result is shown as a red line, the analytical result obtained by using Wu's EC model is plotted in a black line and the analytical result obtained by using our proposed EC model is plotted in a blue line. Results indicates the successful characterisation of queueing delay in sample-based systems by using our proposed EC model and further suggests adopting this model in low-transmission scenarios rather than the conventional EC model.

IV. CONCLUSION

In this poster, we developed a discrete link-layer queueing model and derived mathematical formula of characterising the queueing delay performance in wireless sensor networks (WSNs). Specifically, we extended the existing Effective Capacity model from the time domain into the discrete time domain. Delay Bound Violation Probability (DBVP) were then derived and validated. Results showed our newly proposed model is more suitable than the conventional EC model in modelling queueing delay distribution in WSN scenarios.

ACKNOWLEDGMENT

This research has been supported by the Cooperating Objects Network of Excellence (CONET), funded by the European Commission under FP7 with contract number FP7-2007-2-224053. The authors would like to thank Dr. Luca Mottola for sharing his knowledge of how to acquire real traces and several valuable discussions in terms of their field experiments.

REFERENCES

- [1] Wu, D., Negi, R.: Effective capacity: a wireless link model for support of quality of service. *Wireless Communications, IEEE Transactions on*, 2, 630 - 643 (2003)
- [2] Publicly available real connectivity traces from tunnels and vineyards, <http://d3s.disi.unitn.it/tunnelvineyard>

Localization

Demo Abstract: RSS-based Localization in Sensor Networks Does Not Need Pre-Deployment Profiling

Maissa Ben Jamâa [§], Anis Koubâa ^{¶†}

[§] ReDCAD Research Unit, National School of Engineers of Sfax, University of Sfax, Tunisia.

[¶] COINS Research Group, Al-Imam Mohamed bin Saud University, Saudi Arabia.

[†] CISTER Research Unit, Polytechnic Institute of Porto (ISEP/IPP), Portugal.

mbenj@redcad.org, aska@isep.ipp.pt

Abstract—In this Demo paper, we demonstrate that Received Signal Strength (RSS)-based localization can be performed with no cumbersome operations and still results in a good accuracy. Indeed, traditional RSS-based localization methods typically require an offline labor-intensive, time-consuming and manual pre-deployment calibration. Such cumbersome pre-deployment phase represents a major handicap constraining the wide adoption of RSS-based localization methods and its practical deployment. Further, the resulting static caliber is prone to error as it is not robust to the dynamics of the environment. To fill this gap, we propose EasyLoc, a simple yet effective and practical method for RSS-based localization, which provides good localization accuracy. We also present, *iLoc* a tool that we specifically designed for experimental data collection and data analysis for localization purposes in sensor networks. *iLoc* has been used to evaluate the performance of our EasyLoc approach.

I. INTRODUCTION

Localization based on Received Signal Strength (RSS) is an attractive method for locating objects in Wireless Sensor Networks (WSNs) due to its cost effectiveness and computation simplicity. Classical RSS-based localization techniques are roughly classified in two categories: (1) *map-based* and (2) *model-based*. Map-based techniques consist in building a map of radio fingerprints, each refers to a unique cell in the deployment area. On the other side, model-based techniques aim at establishing a mathematical model capturing the variation of RSS as a function of distance. For both mechanisms, it is central to pass through an offline and tedious environment profiling phase to collect empirical data to map the distance (or cells) to RSS. This operation becomes even more complex as environment changes will compromise the mapping. To overcome these shortcomings, recent RSS-based localization works, such as [1], [2], [3], [4], proposed solutions to automate the calibration process and execute it in runtime. However, these works concern either Wi-Fi or GSM networks, which fundamentally differ from low-power WSNs. Further, most of these works rely on centralized approaches as they require high computational level, thereby their scalability and practical use are constrained. To fill this gap, we proposed EasyLoc, a plug-and-play distributed RSS-based localization method that does not need any offline pre-deployment phase. The idea consists in exploiting the available distance information between anchors to derive an online distance mapping. We

experimentally evaluated EasyLoc using our *iLoc* tool.

In what follow, we describe EasyLoc in Section 2, and present *iLoc* experimental tool in Section 3.

II. EASYLOC DESIGN

The objective of EasyLoc is to provide an RSS-based localization with no cumbersome pre-deployment phase, while still being accurate.

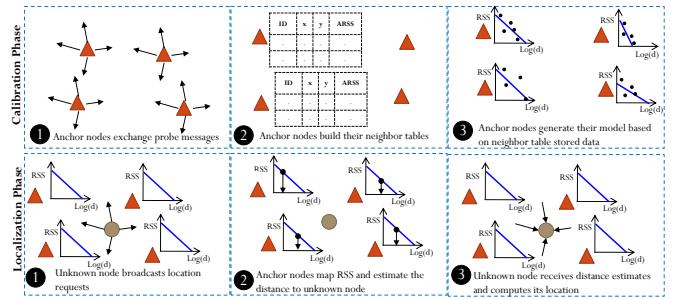


Fig. 1. EasyLoc: Calibration and Localization Phases

EasyLoc illustrated in Fig. 1, is a distributed RSS-based localization approach that comprises two main phases: (i.) *online calibration phase*: where, instead of building a global RSS to distance mapping based on cumbersome profiling phase, each anchor node determines a local mapping by exploiting the knowledge of its distances to other neighbor anchors. Each anchor collects in its neighbor table a raw of RSS values after some probe messages exchange with neighbor anchors, and then determines a linear mapping between the RSS and the log of distance, using statistical regression. The model is periodically updated making EasyLoc adaptive to environment changes in real-time. (ii.) *a localization phase*: aims at determining the absolute location of an unknown node using trilateration techniques, namely min-max and/or centroid. The unknown node triggers the localization procedure by sending a burst of location requests. Anchor nodes on their sides, measure the signal strengths from received location requests (RSS), average them and use their mapping models to extract the relative distance. Then, a location response message containing the distance estimate in addition to the location coordinates of the anchor is sent to the unknown node. Based

on the received data, the unknown node estimates its location coordinates using trilateration.

III. *iLoc* OVERVIEW

iLoc was designed to facilitate the experimentation and performance evaluation of RSS-based localization methods. It represents a major extension to our RadiaLE framework [5] dedicated for experimenting and evaluation link quality estimators. *iLoc* toolset architecture, depicted in Fig. 2, shows both hardware and software components. Hardware components involve a set of self-powered anchor nodes and one unknown node connected to a control station (PC) via a USB cable. Software components comprise two independent applications: (i.) Experiment Control (*iLocController*), which is a Java application responsible for the experimental data collection and experiments configuration, and (ii.) data analysis (*iLocAnalyzer*), which is a Matlab application that allows empirical data analysis to assess the statistical properties of RSS-based localization, such as cumulative distribution functions of distance errors, location errors, Mean Absolute Error (MAE), Root Mean Square Error (RMSE).

iLocController application allows the user to specify the unknown node parameters, such as the radio channel, the transmission power, the number of location requests (*logreq*) bursts, the size of these bursts and the inter-packet interval between each *logreq*. The aforementioned settings are transferred to the unknown node once the localization procedure starts. During the execution of the localization procedure, the unknown node receives packet statistics from neighboring anchors. Packet statistics such as anchor_id, anchor_x, anchor_y, estimated_distance and time stamp are then sent via USB to the *iLocController* application in the PC, which in turns stores these log data into a MySQL database. Also, *iLocController* offers the possibility to view the raw data retrieved from anchor nodes in real-time and to preview network estimated map. On the other side, *iLocAnalyzer* provides a user interface that connects to the database and processes data, with the aim of providing an assistance to *iLoc* users to evaluate the accuracy of their localization technique in terms estimated distances and locations. Currently, *iLoc* implements Weighted Centroid (*WC*) and Min-Max (*MM*) trilateration techniques. Other techniques can be easily integrated.

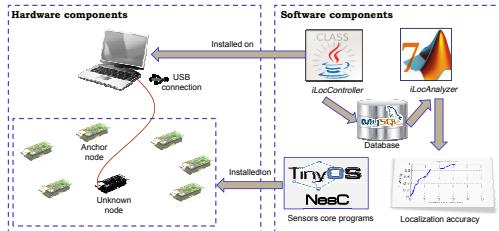


Fig. 2. *iLoc* Architecture

IV. EXPERIMENTAL RESULTS

We used *iLoc* to evaluate the performance of our proposed EasyLoc RSS-based Localization method. For that, we con-

sidered a network composed of N anchors deployed in $3m \times 2m$ area, where N is set to 4, 6, 8 and 9. We considered six unknown nodes at different locations. The transmission power was set to -7 dBm and the localization accuracy was evaluated by analyzing the distribution of the distance estimates and location estimates errors, illustrated by the empirical CDF. The experiment results depicted in Fig. 3 show that distance error is 80% less than 1 meter. The Figure also illustrates the different cumulative probabilities of location errors for both Weighted Centroid and Min-Max algorithms in this particular scenario. It can be seen than the location error with Weighted Centroid is concentrated in 1 meter, whereas as it is distributed in the range [0, 1.6] in the case of min-max.

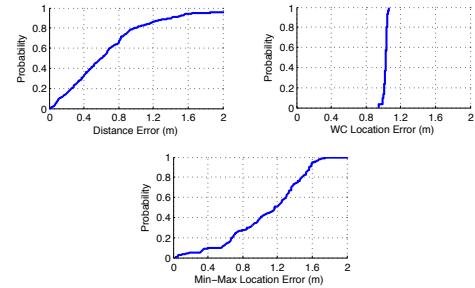


Fig. 3. Localization Accuracy

V. CONCLUSION

In this Demo paper, we illustrated through EasyLoc how to RSS-based localization can be easily adopted with no pre-deployment profiling operations. The main idea was to exploit the knowledge of distance to neighbor anchors to derive an anchor-specific RSS to distance mapping. The demonstration of the behavior and performance of EasyLoc is performed using *iLoc*, a tool we designed for the experimental data collection and data analysis for localization purposes in sensor networks. We believe that EasyLoc will open new ways for the ease of deployment of RSS-based localization method.

VI. ACKNOWLEDGMENT

This work is funded by R-Track project under the grant 8-INF-2008 of the National Plan for Sciences and Technology (NPST).

REFERENCES

- [1] H. Lim, L.-C. Kung, J. C. Hou, and H. Luo, "Zero-configuration, robust indoor localization: Theory and experimentation," in *INFOCOM*, 2006.
- [2] A. Varshavsky, D. Pankratov, J. Krumm, and E. Lara, "Calibree: Calibration-free localization using relative distance estimations," in *Proceedings of the 6th International Conference on Pervasive Computing*, ser. *Pervasive '08*. Berlin, Heidelberg: Springer Verlag, 2008, pp. 146–161.
- [3] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan, "Indoor localization without the pain," in *Proceedings of the sixteenth annual international conference on Mobile computing and networking*, ser. *MobiCom '10*. New York, NY, USA: ACM, 2010, pp. 173–184.
- [4] A. M. Bernardos, J. R. Casar, and P. Tarro, "Real time calibration for rss indoor positioning systems," *East*, no. September, pp. 15–17, 2010.
- [5] N. Baccour, A. Koubaa, M. B. Jamâa, D. do Rosário, H. Youssef, M. Alves, and L. B. Becker, "Radiale: A framework for designing and assessing link quality estimators in wireless sensor networks," *Ad Hoc Networks*, vol. 9, no. 7, pp. 1165–1185, 2011.

Poster Abstract: Low Cost Sensor Design for Non-Cooperative Geolocation via RSS

Michael S. Butler, Richard K. Martin, and Russell Lenahan
The Air Force Institute of Technology Dept. of Elec. & Comp. Eng.

Abstract— Obtaining accurate non-cooperative geolocation is vital for persistent surveillance of a hostile emitter. Current research for developing a small, cheap and energy efficient sensor network for non-cooperative geolocation measurements via received signal strength (RSS) is thin. Most existing work focuses on simulating a non-cooperative network (NN) and in doing so, simulated models often ignore localization errors caused from the hardware processing raw RSS data and often model environment-dependent errors as random. By comparing real-time measured non-cooperative geolocation data to a simulated system a more accurate model can be developed. In this poster we discuss the development of a sensor network that can locate a NN via RSS.

I. INTRODUCTION

Geolocation is the process of using a wireless sensor network (WSN) to locate and track the position of a radio emitter. Four common measurement methods can be used for localization of wireless devices. They are RSS, AOA, TOA, and TDOA. For comparison, AOA requires more complex hardware on each sensor (such as an antenna array). TOA requires cooperation between the emitter and sensors for precise timing. TDOA uses relative time measurements at each receiving sensor in place of absolute time measurements [1], [2]. Though each measurement type has its own merits, this paper focuses on RSS.

There are two types of methods by which RSS measurements can be obtained: cooperative and non-cooperative. In a cooperative network, the device to be located may share parameter values with the WSN. In such cases, the reported RSS is just the signal power, as the signal can be demodulated and segregated from additive noise [1], [3].

In NNs, many properties of the emitter are unknown. The RSS may be determined by energy detection such as integrating the observed Power Spectral Density (PSD) [1], [3]. Fig. 1 shows the differences between a CN and NN.

II. NON-COOPERATIVE SYSTEM ARCHITECTURE

RSS measurements are achieved in a NN by utilizing a Poor Man's Spectrum Analyzer (PMSA) and a Sun

This work is funded in part by the Office of Naval Research. The views expressed in this paper are those of the authors, and do not reflect the official policy or position of the United States Air Force, Navy, Department of Defense, or the U.S. Government. This document has been approved for public release; distribution unlimited.

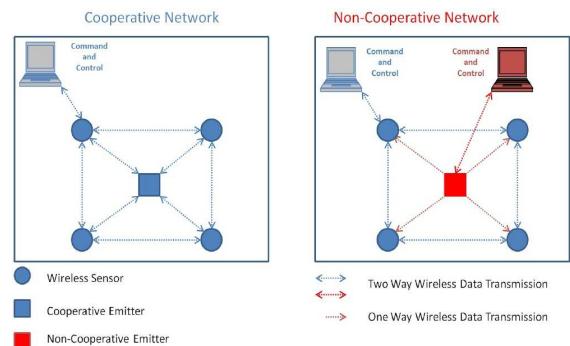


Fig. 1. Cooperative vs. Non-Cooperative Network

Programmable Objective Technology (SPOT) device. By using a spectrum analyzer to interface with a wireless sensor, visual detection and analysis of electromagnetic signals over a defined band of frequencies can be made. A SPOT can be used to simulate transducers and WSN motes. A majority of the research effort had been focused on successfully designing the PMSA. A sensor mote is defined when the PMSA is integrated with a SPOT.

Another component of the RSS system architecture is a non-cooperative emitter transmitting in the ISM band. The controlling operations and algorithm implementations of the sensor motes are performed in the Command and Control (C2) center. The C2 center consists of a SPOT base station and laptop computer. The base station unit communicates wirelessly with the SPOT, which then streams the data via a USB connection to the host computer.

III. PMSA THEORY OF OPERATION

As shown in Fig. 2, the potentiometer is a digitally controlled variable resistor (VR) device. Changing the VR settings is accomplished by pulsing the clock pin (CLK) while the chip select (CS) is active low. The direction of the increment is controlled by the up/down (U/D) input pin. By pulsing the potentiometer, the VR will provide an output voltage that tunes the oscillator to achieve an intermediate

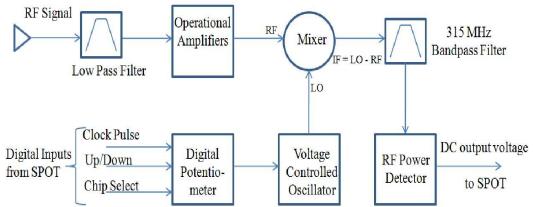


Fig. 2. PMSA Block Diagram

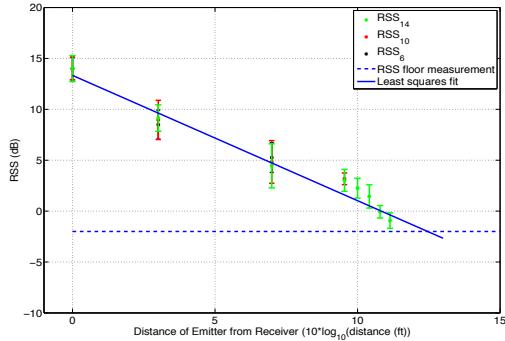


Fig. 3. RSS vs. Distance

frequency (IF) signal that is between 314.7 MHz and 315.3 MHz. The intermediate frequency (IF) signal at the output of the mixer is filtered by a band-pass filter with a center frequency of 315 MHz with a 3 dB bandwidth of 600 KHz.

When the frequency of the IF signal is between 314.7 MHz and 315.3 MHz the filtered IF signal is passed to the logarithmic RF power detector. The DC voltage at the output of the power detector approximates the logarithm of the filtered IF signal's amplitude. The SPOT is programmed to provide the inputs to the CLK, CS pin, U/D input pin and measure the DC output voltage.

IV. SENSOR NETWORK MODELING

A signal generator was used as the non-cooperative emitter to model the sensor network. A RSS measurement algorithm was developed for the SPOT to convert the maximum DC output voltage to a RSS value. The algorithm was developed by measuring the output voltage at varying distances, transmitted powers (P_0) and frequencies.

Fig. 3 is a plot of the RSS at each distance a measurement was taken for $P_0 = 6, 10$ and, 14 dB. The RSS data points for $P_0 = 6$ and 10 were shifted up to obtain a better line fit. The dotted line at -2 dB represents a measurement floor indicating the lowest P_0 the sensor can measure at the reference distance d_0 . The sensor mote's ranging limit is where the measurement floor and data fit line intersect.

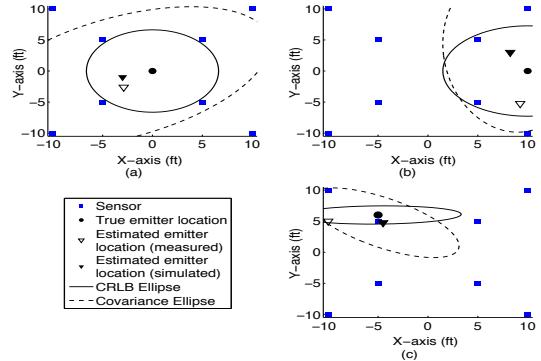


Fig. 4. Measured vs. Simulated geolocation

V. INITIAL OPERATION AND TEST RESULTS

Fig. 4 shows the averaged measured and simulated results of 10 trials locating one emitter at a time placed in three locations. For the simulated results the power received at each sensor is modeled as a normal distribution

$$\mathbf{P} = [P_1, \dots, P_s]^T \sim N(\mathbf{m}, \sigma^2 \mathbf{I}) \quad (1)$$

where the fading standard deviation, $\sigma = 6$ dB. The linear model is given by

$$P_s = m_s + w_s \quad (2)$$

where m_s is the mean power value received at each sensor and w_s is Additive White Gaussian Noise.

$$m_s = P_0 - \eta \cdot 10 \cdot \log_{10} \left(\frac{d_s(x_0, y_0)}{d_0} \right) \quad (3)$$

where the path loss exponent η is the slope of the best fit line in Fig. 3. Test results show the measured estimates are within a 90% confidence interval of the Cramer-Rao Lower Bound (CRLB) and covariance error ellipses.

VI. CONCLUSION

This poster has presented a small, low cost and energy efficient sensor network to measure non-cooperative geolocation via RSS. The successful design of the sensor network was supported by presenting measured geolocation estimates. Future work will consist of improving sensor network modeling and collecting RSS measurements on practical non-cooperative devices.

REFERENCES

- [1] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, III, R. L. Moses, and N. S. Correal, "Locating the nodes: cooperative localization in wireless sensor networks", *IEEE Signal Processing Mag.*, vol. 22, no. 4, pp. 54-69, July 2005.
- [2] G. Mao, B. Fidan, and B. D. O. Anderson, "Wireless sensor network localization techniques", *Comput. Netw.*, vol. 51, no. 10, pp. 2529-2553, Jan. 2007.
- [3] H. Wymeersch, J. Lien, and M. Z. Win, "Cooperative localization in wireless networks", *Proc. IEEE*, vol. 97, no. 2, pp. 427-450, Feb. 2009.

Poster Abstract: Localization based on Reflected Signals in Wireless Sensor Networks

Kaushik Mondal and Partha Sarathi Mandal, IEEE Member
Indian Institute of Technology Guwahati, India
Email: {mondal.k, psm}@iitg.ernet.in

Bhabani P. Sinha, IEEE Fellow
Indian Statistical Institute, Kolkata, India
Email: bhabani@isical.ac.in

Abstract—Localization in an urban area is a challenging problem due to the blocking of Line-Of-Sight (LOS) signal by various obstacles and also the multipath effect arising out of reflections and scattering of signals. Assuming that there are a few anchor nodes which know their positions accurately and which transmit ultrasonic signals, we propose here a technique to find the position of other sensor nodes based on receiving these ultrasonic signals reflected by some reflectors. Our proposed technique can calculate the position of a node correctly by receiving two reflected signals (non-line of sight) from an anchor. We, however, assume that a signal is reflected at most once before reaching a node and the two reflecting surfaces are non-parallel to each other.

I. INTRODUCTION

The goal of localization in wireless sensor networks (WSNs) is to establish the position of each node as accurately as possible. For localization without taking help of GPS, we often need a few anchor nodes whose positions are known very accurately. Beacon signals are being transmitted from these anchor nodes to be received by other sensor nodes. In an urban area, the received signal may be a Line-of-Sight (LOS) one, or one that is reflected and/or scattered by various obstacles before reaching the destination node. Various localization techniques have already been proposed in the literature which are often based on measuring the time of arrival (ToA) [1] or angle of arrival (AoA) [2] of the received signals, to calculate the distances and angles, respectively. However, suitable technique for finding the location of sensor node which mitigates the effect of multiple reflections and/or scattering in the most general environment, is still called for.

A. Our Contribution

In this paper we propose a deterministic protocol that can calculate the position of a sensor node accurately by using one anchor node which transmits an ultrasonic signal, and assuming the presence of two reflectors (in the absence of a direct path or LOS communication) for this signal to be received by the node in question whose position is to be estimated. We assume the following in our model: (i) A node receives two beacons (ultrasonic signals) from any particular anchor where the reflectors are not parallel to each other. (ii) Each beacon is reflected only once before reaching the destination node. (iii) Each node is equipped with the

The first author is thankful to the Council of Scientific and Industrial Research (CSIR), Govt. of India, for financial support during this work.

appropriate mechanism to measure both the time of arrival (ToA) and angle of arrival (AoA) of the received signal.

It follows from the last assumption above that in presence of a direct path or LOS communication, one beacon is sufficient for estimating the position of the node in question.

In most of the earlier works [1], [2] usually three anchor nodes are used to locate a sensor node. But in our approach, only one anchor node is sufficient to calculate the position of a node. This gives us an advantage particularly in dealing with the sparse networks.

II. SYSTEM MODEL

We assume that the sensor nodes have been deployed on a two dimensional plane and each has been assigned a unique *id*. There are some reflectors and anchor nodes in the same plane. The position of a node is calculated based on a chosen coordinate system. The position of the anchor nodes are known and an anchor can be identified uniquely by its position. An anchor node sends its own position as *id* with the beacon signal. We assume that all anchor nodes as well as the sensor nodes are equipped with an omnidirectional antenna for sending a beacon and also a directional antenna for the measurement of angle of arrival of a signal from other sensor nodes. An anchor is said to be a *neighbor* of another anchor if it is located within twice the transmission range of the second anchor. Anchor nodes are synchronized with some global clock (possibly through GPS) such that at a time only one anchor sends a beacon to avoid collision with the beacons from neighboring anchors. This ensures that a receiving sensor node receives only one beacon at a time from a particular anchor node. A sensor node receives the beacon from the anchor by using a directional antenna so that it can also measure the angle of arrival of the beacon signal. Because of this directional antenna, the sensor node does not experience a collision even when it receives more than one beacon from an anchor coming through different paths at different angles. After receiving a beacon, a sensor node transmits back a signal to the anchor, in the same direction in which it has received the beacon from the anchor. This signal carries the *id* of the sensor node and the angle of arrival (AoA) of the beacon. We now state the following results:

Lemma 1: Consider a fixed point *S* on a straight line *l* with gradient m_l . Let *L* be the set of all parallel straight lines with gradient m_L such that $m_l \neq m_L$. Let *P_i* be the point of

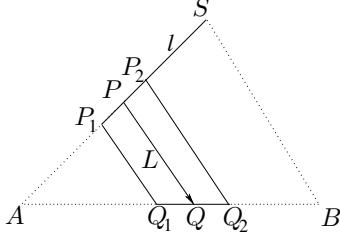


Fig. 1. Figure showing one *fixed_distance_line*, Q_1Q_2

intersections of l with a line $\ell_i \in L$, for $i = 1, 2, \dots$ (refer to Fig. 1 for illustration). Let Q_i s be the points on ℓ_i such that $SP_i + P_iQ_i = d$, for $i = 1, 2, \dots$, where d is a fixed distance. Then all the Q_i s must lie on a straight line.

The straight line mentioned above we define as the $\text{fixed_distance_lines}_{S, m_l, m_L, d}$. There are four possible $\text{fixed_distance_lines}_{S, m_l, m_L, d}$ as shown in Fig. 2. Now, we can easily verify the following lemmas.

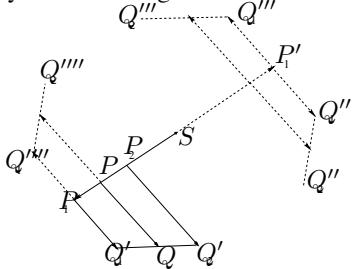


Fig. 2. Figure showing four possible *fixed_distance_lines*, $Q'_1Q'_2$, $Q''_1Q''_2$, $Q'''_1Q'''_2$, $Q''''_1Q''''_2$.

Lemma 2: Among the $\text{fixed_distance_lines}_{S, m_l, m_L, d}$, the intersecting lines are perpendicular to each other, and the non-intersecting lines are parallel to each other.

Lemma 3: A bisector of one of the angles between the straight lines l and any line $\ell \in L$ is parallel to the $\text{fixed_distance_line}_{S, m_l, m_L, d}$.

Here we state another lemma:

Lemma 4: The position of a sensor node cannot be uniquely identified by the above method, if and only if the node receives two beacons from the same anchor node which are reflected from two parallel reflectors.

We can state the following theorem using the above lemmas.

Theorem 1: A node finds its position correctly if it receives either i) the direct (LOS) signal from an anchor node, or ii) two reflected signals from an anchor node with the corresponding reflectors not being parallel to each other.

III. SIMULATION

According to our proposed algorithm the position of a node is calculated perfectly, based on received signals, which are up to one bound. But in practical situation a node may receive multi-bound (more than one bound) signals. Now, if the position of a node is calculated considering multi-bound signals as one bound signals using proposed algorithm then error may be accumulated. Considering the above fact, we have measured the error using simulation mentioned below.

In simulation multi-bound signals are filtered out to some percentage. We have calculated the position of a node several times by considering rest of the multi-bound signals as one

TABLE I
TABLE SHOWING ERROR FOR ACCEPTANCE OF 2BS AS 1BS

No of runs	1	2	3	4	5
% of acceptance					
2bs as 1bs	23.27	24.69	23.03	18.35	23.19
Avg % error	17.10	23.60	21.50	22.50	18.10

bound. So, errors are accumulated in the calculation and we have measured the percentage error. According to the Fig. 1 range of Q is AB and so the average theoretical error is taken as $\text{length}(AB)/2$. The table I given above shows percentage of acceptance of the two bound signals as one bound signal and the corresponding average percentage of error.

A. The Algorithm

Based on the above discussions, Algorithm: FINDPOSITION given below finds the position of a sensor node.

Algorithm 1 FINDPOSITION

- 1: Anchor S sends a beacon with $\langle \text{anchor_id} \rangle$ using omnidirectional antenna.
- 2: **for** each node Q who hears the beacon **do**
- 3: Q measures the angles of arrival (θ_i) of all received beacon(s), $i = 1, 2, \dots$, and transmits back $\langle \text{node_id}, \theta_i \rangle$ to S via the same path(s).
- 4: **end for**
- 5: S measures the angles of arrival (δ_i) while receiving all replies and computes the corresponding distances (d_i) traveled by the beacon by measuring the ToA. From the θ_i and δ_i values, S determines whether Q received the signal(s) along a direct path and/or reflected paths. After that, S executes the steps 6, 7 and 8 below.
- 6: If Q received at least one beacon from S through the direct (LOS) path, then S computes the location of Q following case 1 of theorem 1.
- 7: If Q received at least one pair of beacons through two non-parallel reflectors, then S computes the location of Q following case 2 of theorem 1.
- 8: For all other cases, S reports the inability to compute the location of Q unambiguously.

IV. CONCLUSIONS

In this paper we have proposed a deterministic protocol to find the position of a sensor node based on receiving two reflected signals from only one anchor node. The proposed technique does not need to know the positions of these reflectors. In the presence of some parallel reflectors, the position of any node can also be determined if the node receives two reflected signals from the same anchor through any two non-parallel reflectors.

REFERENCES

- [1] Y. Zhang, W. Liu, Y. Fang, and D. Wu, "Secure localization and authentication in ultra-wideband sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 4, pp. 829–835, 2006.
- [2] G. Oberholzer, P. Sommer, and R. Wattenhofer, "SpiderBat: Augmenting wireless sensor networks with distance and angle information," in *IPSN*, 2011, pp. 211–222.

Testbeds

Demo Abstract: MOTE — A Mobile Robotic-Assisted Wireless Sensor Networks Testbed

Alexander Förster*, Anna Förster†, Kamini Garg†, Daniele Puccinelli†, Silvia Giordano† and Luca M. Gambardella*

*Istituto Dalle Molle di Studi sull’Intelligenza Artificiale (IDSIA), Switzerland

†Networking Laboratory, ISIN-DTI, University of Applied Sciences of Southern Switzerland

Abstract—We present MOTE: a robotic-assisted mobile wireless sensor network testbed. It is able to deploy and conduct mobile WSN experiments, where sensor nodes are piggybacked on mobile robots. The system consists of two main components: Multi-Robot Architecture for Coordinated Mobility (MuRobA) and Flexible WSN Runtime Management Software Architecture (FLEXOR). The first controls the mobility of the robots and the second controls the sensor nodes without the use of a backchannel. In this demonstration, we show the general architectures and usage of both components. Most importantly, we show the deployment and usage of MOTE as whole, which can be conducted in any indoor environment in only few hours.

I. INTRODUCTION AND MOTIVATION

Wireless sensor network testbeds have developed to be the de facto standard for testing WSN applications and algorithms. However, most of these testbeds rely on a complex infrastructure (backchannel) to provide power to the nodes and to disseminate code and data. This infrastructure is not only costly, but makes the testbed rigid. The nodes cannot be moved between or during experiments and re-deployment of the complete testbed is time and effort-consuming.

On the other side, mobile WSN testbeds offer a new environment to the WSN developer. However, it exhibits two major challenges: First, the backchannel disappears and second, the mobility of the nodes needs to be implemented. The problem of the backchannel has been addressed many times in terms of remote re-programming or debugging of nodes, e.g. the tool Marionette [1] or the Contiki toolchain [2]. However, such tools are rather rigid and support only one operating system (TinyOS or Contiki), while we are looking for a general-purpose lightweight user-controlled tool, which enables the testbed user to implement its own debugging commands and to easily exchange software modules at runtime.

The second challenge, the mobility of sensor nodes, has been usually addressed by piggybacking sensor nodes on large indoor robots. This has been implemented in the Wisebed platform as RoombaNet [3] or in the integrated CONET testbed (*conet.us.es*). However, all these systems use autonomous robots, which rely on complex localization and navigation algorithms, which often prove to be unreliable and slow. On the other side, there exist also centralized solutions, called also *global vision*, for mobile robots, like the ones used in the RoboCup small size league [4]. Here, the robots are observed by overhead cameras and localization and navigation becomes fast and reliable. This suits the purposes of a WSN testbed

much better and is also less costly and more scalable. In the next paragraphs, we present our solutions to both problems: FLEXOR and MuRobA and will demonstrate their deployment and usage in MOTE.

II. FLEXOR: FLEXIBLE RUNTIME MANAGEMENT SOFTWARE ARCHITECTURE FOR WSN

FLEXOR is a general-use software architecture for programming WSNs. It is platform-independent and has extensive graphical support for implementing, programming and managing WSNs. Its general architecture is detailed in [5] and depicted also in Figure 1. For MOTE, the most important properties of FLEXOR are its general-use remote function call mechanism (implemented by the Callback Manager) and the possibility to exchange software components at run-time. For the latter, we define *Images*, which consists of several *specifications*, which in turn describe software components architectures. For example, one specification might consist of an application and routing modules and another of the same application, but different routing module. At run-time, these specifications can be exchanged with a single 1-byte command instead of complex re-programming and reboot of the complete system.

FLEXOR provides the following further functionalities and abilities to MOTE:

- Remote function call of user-defined functions
- Parameter change of individual modules
- Runtime exchange of software modules
- Remote debugging, status inquiries and data logging

All together, these properties enable MOTE to run sophisticated, structured experiments with WSNs in both mobile and static environments without the need of a backchannel. In order to enable the mobility of the nodes, we piggyback them on mobile robots and implement the robotic architecture MuRobA, described briefly in the next section.

III. MUROBA: MULTI-ROBOT ARCHITECTURE FOR COORDINATED MOBILITY

The general architecture of MuRobA is presented in Figure 2. It consists of one to several cameras, overlooking the robots; the component *camview*, which tracks the colorful dots on top of the robots to localize them; the *FleetManager*, which consolidates the information of all cameras and decides the movement commands for all robots; and finally one or several

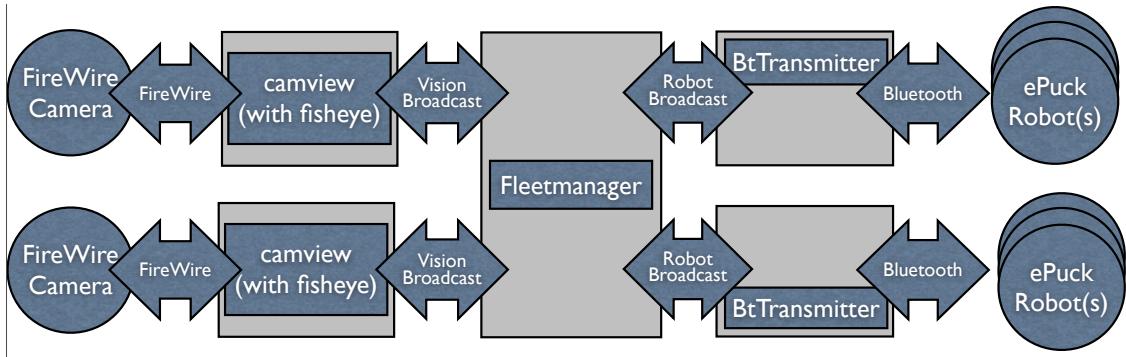


Fig. 2. MuRobA architectural overview, with its main components the camera input, the fleet manager and the bluetooth-supported robot control. Note that the system can have several cameras as input and several bluetooth-supported robot controllers as output.

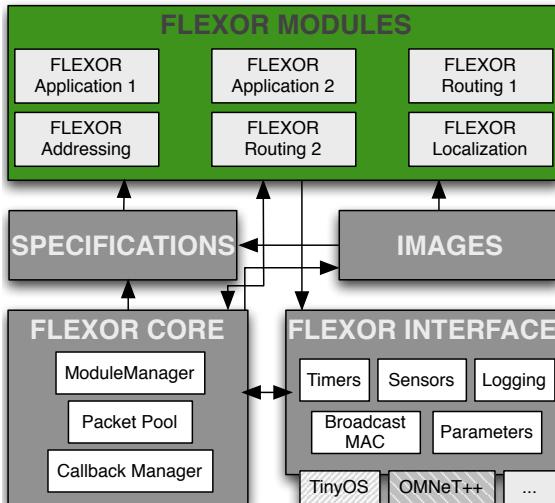


Fig. 1. FLEXOR system overview with its main components. Most important for MOTEL is its remote function calls (CallbackManager) and the possibility to exchange software components at run-time (the specifications).

BluetoothTransmitter, which sends commands to the robots via bluetooth. Note that the system is very flexible and scalable, as it allows for several camera inputs and several bluetooth controllers. Note also the modularization of the system makes it hardware independent. It allows for the usage of any cameras with any lenses and any robotic platforms.

IV. OVERVIEW OF MOTEL

Figure 3 depicts the MOTEL deployment. It consists of the playground with the camera overlooking it, the robots with sensor nodes piggybacked on them, and the two control stations for MuRobA and FLEXOR. Additional sensor nodes can be freely placed wherever it fits the experiment and will be also controlled through FLEXOR.

V. NEXT STEPS

In the next future, we will implement and enable a remote testbed control unit, so that experiments can be planned and conducted also remotely on MOTEL. This is a major challenge especially because of the batteries needed by both robots and sensor nodes. We plan to supply the sensor nodes with power through the robots and to implement hardware and software for the robots, which will enable them to re-charge autonomously.

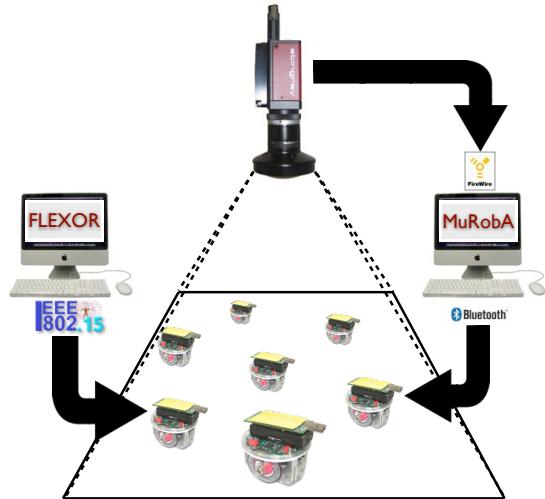


Fig. 3. General architecture of MOTEL with its main components for controlling the robots and the sensor nodes.

ACKNOWLEDGMENTS

The work presented in this paper has been funded partially by SUPSI under grant number 32RAX3MOTE, "MOTEL: Mobile wireless sensor network testbed" and partially by the EU Cooperating Objects Network of Excellence (CONET).

REFERENCES

- [1] K. Whitehouse, G. Tolle, J. Taneja, C. Sharp, S. Kim, J. Jeong, J. Hui, P. Dutta, and D. Culler, "Marionette: using rpc for interactive development and debugging of wireless embedded networks," in *The Proceedings of the 5th International Conference on Information Processing in Sensor Networks, 2006. IPSN 2006*, 2006, pp. 416–423.
- [2] A. Dunkels, N. Finne, J. Eriksson, and T. Voigt, "Run-time dynamic linking for reprogramming wireless sensor networks," in *Proceedings of the 4th ACM International Conference on Embedded Networked Sensing Systems (SenSys)*, Boulder, CO, USA, 2006.
- [3] M. A. Hail, J. Pinkowski, T. Teubler, M. Danckwardt, D. Pfisterer, and H. Hellbrück, "Roombanet - testbed for mobile networks," in *Proceedings of the Workshops der wissenschaftlichen Konferenz Kommunikation in verteilten Systemen 2011 (WowKIS 2011)*, T. Margaria, J. Padberg, and G. Taentzer, Eds., vol. 37. Electronic Communications of the EASST, 2011, accepted for publication.
- [4] A. Egorova, A. Gloye, C. Göktekin, A. Liers, M. Luft, R. Rojas, M. Simon, O. Tenchio, and F. Wiesel, "Robocup 2004 symposium: Papers and team description papers," in *RoboCup 2004 Symposium: Papers and Team Description Papers*, 2004.
- [5] A. Förster, K. Garg, D. Puccinelli, and S. Giordano, "Flexor: User friendly wireless sensor network development and deployment," 2011, under submission.

Demo Abstract: Testbed-Independent Experiment Specification and Execution using the COTEFE Platform

Claudio Donzelli

Telecommunication Networks Group
Technische Universität Berlin
donzelli@tkn.tu-berlin.de

Vlado Handziski

Telecommunication Networks Group
Technische Universität Berlin
handziski@tkn.tu-berlin.de

Adam Wolisz

Telecommunication Networks Group
Technische Universität Berlin
awo@ieee.org

Abstract—The demo illustrates the capabilities of the CONET Testbed Federation (COTEFE), focusing on the support for testbed-independent specification and execution of experiments. It leverages a prototypical implementation of the platform and its two core programming interfaces that follow the REST architectural style. The Testbed Abstraction API, that exposes the services of the individual testbeds thorough a uniform Testing-as-a-Service (TaaS) abstraction, has been implemented and deployed on top of the TWIST testbed. The Testbed Federation API, that leverages these abstractions to build higher-level testbed-independent services, has been implemented and deployed using the Google App Engine infrastructure.

I. INTRODUCTION

The design, implementation and evaluation of cooperating object (CO) protocols and applications is a challenging task that is further complicated by their distributed and heterogeneous nature and the tight coupling with the environment. In the advanced design stages the evaluation of the system performance, error resilience and other nonfunctional properties necessitate use of real hardware, realistic environments and realistic experimental setups.

Testbeds offer convenient middle ground on the realism axes between simulation and full deployment and enable rigorous and controlled experimentation with the System-Under-Test (SUT). Like full deployments, however, they lock the evaluation to one particular environment making it hard to differentiate between the intrinsic properties of the SUT and the influence of the specific features and context at a given testbed site.

One way of decoupling these influences is to cross-validate the functional and non-functional behavior of the SUT under various conditions as provided by different testbeds. Unfortunately, the realization of such measurement campaigns is currently accompanied by significant overheads in configuring the experiments and collecting the results on the individual testbeds, since easy experiment migration is hindered by a lack of common management, experiment specification and control infrastructure.

The CONET Testbed Federation (COTEFE) has been designed to addresses these challenges (Figure 1). It follows a novel Testing-as-a-Service (TaaS) approach that has been

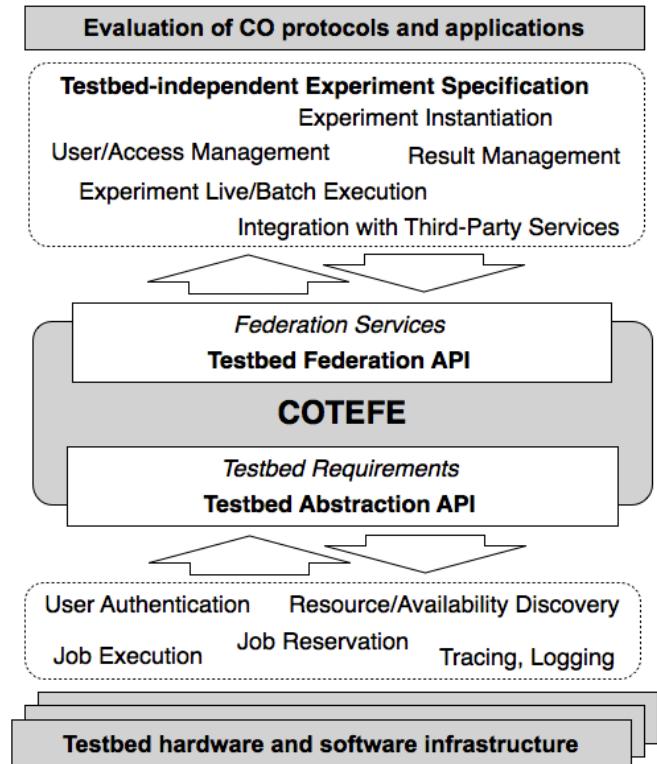


Fig. 1. Overview of the CONET Testbed Federation platform

adapted to the specific requirements of the cooperating objects domain. Our platform offers convenient access to the resources of multiple testbeds, organized in a federation of autonomous entities. It provides remote services supporting the complete testing life-cycle including resource discovery and reservation, testbed-independent experiment specification, experiment execution and data collection and management.

In the following we briefly outline the main architectural features of the COTEFE platform and the implementation status of its prototype, before presenting the demonstration scenario in greater detail.

II. COTEF

COTEF introduces two core APIs which have been designed according to the REST principles [1]. The Testbed Federation API (TFA) answers to the emerging need of supporting design and execution of complex CO experiments and migration across different testbeds by exposing high-level abstraction services enabling experimental research with cooperative objects . The Testbed Abstraction API (TAA) defines the requirements that a CO testbed must fulfil in order to be part of COTEF. It exposes a selected set of capabilities of the heterogeneous member testbeds under a uniform interface which can coexist with the legacy testbed APIs and allow full autonomy of member testbeds in terms of user management and access policies.

This demo makes use of a prototypical implementation of the two core APIs included in COTEF which has been developed and deployed for testing. In particular, a prototype of the Testbed Federation API has been implemented and deployed on Google App Engine [2], which enables easy development, deployment and administration of web applications on the Google infrastructure. An instance of the Testbed Abstraction API, based on Django [3], is currently operating on TKN Wireless Indoor Sensor network Testbed (TWIST), the 200+ nodes WSN testbed deployed over four floors in our building at Telecommunication Networks Group (TKN), Technische Universität Berlin. The prototype currently fully supports experiment specification and execution. Visualization of results and debug data is currently supported by a customized Java stand-alone application, which is however based on a testbed-independent configuration script.

Being the two APIs based on RESTful HTTP, every HTTP client (curl, web browsers) or library is a potential client of COTEF. Two kinds of client have been developed so far: a series of Python scripts (using the modern *httpplib2* HTTP library) and a rich HTML5 web interface.

III. DEMO DESCRIPTION

The demo shows the use of the COTEF APIs for specification and execution of a testbed-independent experiment for cooperating object. The experiment aims to study the behavior of Collection Tree Protocol (CTP) [4] in different interference scenarios. In particular, the experiment focuses on the impact of the interference on the routing topology and the ability of CTP to recover. Interference is activated and disabled by the experimenter at run-time in order to analyze its effects on the network topology. The demo shows how the above mentioned experiment can be performed by interacting with COTEF in terms of HTTP RESTful APIs. The demo is comprised of three parts:

- *Experiment Specification*: the user describes the experiment using a testbed-independent specification language and uploads its representation to the Testbed Federation server by exclusively using the Testbed Federation API (no testbed is involved here). We show how an experiment can be specified throughout manipulation of REST resources expressed in JSON format. From the specification

```
{
  "uri": "https://www.twist.tu-berlin.de:8001/nodes/I5304185",
  "media_type": "application/json",
  "name": "WSN node",
  "id": "I5304185",
  "platform": "http://api.cotefe.net/platforms/eyesIFXv20",
  "image": null,
  "serial_tunnel": "ssh://twistextern@www.twist.tu-berlin.de:9030",
  "location_x": "1.72",
  "location_y": "10.98",
  "location_z": "9.08"
}
```

Fig. 2. JSON Representation of the Node resource

of required resources, to organization of such resources in subgroups, to the deployment of software images to the different subgroups, and finally the specification of required steps composing the experiment.

- *Job Reservation*: the user uses the created experiment specification (identified by its URL) in order to perform discovery and reservation of required resources on a testbed (in this example TWIST) whose capabilities can satisfy the given requirements. The testbed-independent experiment specification is then translated by COTEF into a testbed-dependent job specification.
- *Experiment Execution* In this part, the user can perform the experiment by running the corresponding job on the reserved testbed. We show how the experiment can be monitored at runtime and an insight on involved resources can be obtained at any time by submitting a simple HTTP GET requests. In Figure 2 the JSON representation of a WSN node is shown.

IV. CONCLUSION

In this demo we show how a given experiment can be specified in a testbed-independent way and then executed in any of the federated testbeds.

ACKNOWLEDGMENT

This work has been partially supported by CONET, the Cooperating Objects Network of Excellence, funded by the European Commission under the contract number FP7-2007-2-224053.

REFERENCES

- [1] Roy T. Fielding and Richard N. Taylor, “Principled design of the modern Web architecture,” *ACM Transactions on Internet Technology*, 2002.
- [2] “Google App Engine - Google Code,” <http://code.google.com/appengine/>.
- [3] “Django — The Web framework for perfectionists with deadlines,” <http://www.djangoproject.com>.
- [4] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, “Collection tree protocol,” in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys ’09, 2009.

Demo Abstract: CoojaTrace, Extensive Profiling for WSNs

Moritz Strübe, Florian Lukas

Friedrich-Alexander University Erlangen-Nuremberg
{struebe,lukas}@cs.fau.de

Rüdiger Kapitz

TU Braunschweig
rrkapitz@ibr.cs.tu-bs.de

Abstract—CoojaTrace extends the Cooja WSN-Simulator by offering extensive logging capabilities for debugging and analyses of multi-node WSN deployments. This is implemented using Scala-based Functional Reactive Programming (FRP)-techniques, enabling flexible and easily programmable in-depth access to the internal node, as well as simulator execution state.

CoojaTrace is part of the DryRun framework; a set of tools that instrument a WSN-Simulator for extensive analysis of WSN deployments. To achieve this, not only external accessible state like energy usage or serial output is needed, but also internal state like routing tables or operation state in general, which often requires the instrumentation of pointers, and can hardly be monitored by observing static memory addresses. Although some WSN-Simulators provide debug interfaces like a GDB-stub, an easy interface to log and analyze system state of a whole deployment is still lacking. This gap is filled by CoojaTrace by providing a simple and scriptable interface to access this data.

I. INTRODUCTION

CoojaTrace¹ is part of the RealSim/DryRun framework. These two frameworks work together to allow *Deployment-Targeted* development. RealSim automatically configures the simulator to match a real deployment as close as possible, by collecting information from a previously deployed network [4]. DryRun is a collection of tools that use the preconfigured simulator to improve the quality of the deployed network. Examples are finding optimal configuration settings or testing and comparing multiple software revisions.

While most WSN-simulators provide some kind of scripting interface that allows to interact with the serial interfaces of the motes, easy access to internal data and state is often neglected. For example program variables or the state of different hardware components can often only be accessed by directly extending the simulator. As a consequence these extensions are usually highly specialized and therefore not very flexible in terms of extensibility. With the threshold for extending the simulator being quite high, the required data is often exposed via the serial interface, which is not very satisfying solution, as it alters the behavior of the node.

We therefore present CoojaTrace, a plugin for Cooja [2], which allows to access an extensive amount of system state using a scripting language. It provides wrappers to most interfaces to enable monitoring them using Functional Reactive Programming (FRP). The results can be logged to different output formats, as well as displayed at runtime.

II. SYSTEM ARCHITECTURE

Most components within Cooja interact using the observer-pattern, which allows other objects to be notified upon changes to the component's state. This pattern can easily be mapped to the Functional Reactive Programming (FRP) programming paradigm, which introduces a type of variable that can propagate changes automatically. FRP variables can be derived from other FRP variables through arbitrary functional expressions. Once a variable changes, this change is automatically propagated to all derived variables, comparable to the way most spreadsheet programs update cells. Wrapping Cooja's observers into FRP variables allows further processing and logging steps to be described using a concise functional syntax, while the actual data propagation is handled by the FRP framework.

The implementation is based on Scala and the reactive-core framework². As Scala is compiled into Java byte code and its objects are compatible with Java objects, the Java based Cooja simulator can not distinguish the Scala code from any Java code. This allows an easy integration into the simulator. Further on Scala provides a runtime compiler and thus allows to write code without the need to restart the simulator.

Part of CoojaTrace is a comprehensive library, which maps the observer-pattern to FRP, and in addition provides abstractions to simplify the access to Cooja's data. Most notably is the easy access to the memory of the application running on a mote. While Cooja already provides symbol resolution (i.e. mapping a variable name to an address), as well as viewing and changing memory, there currently is no easy way of logging a variable. In addition to logging, CoojaTrace allows to dereference pointers and do pointer arithmetics. This is a very powerful feature as operating systems like Contiki make extensive use of pointers and structs. Even if the target of a pointer is not changed at runtime, obtaining the address of a certain data word currently is manual work, and it is unlikely that the address is unchanged after the next translation, thus requiring manual interaction after each compilation.

Especially when simulating long running experiments, or monitoring values that change often, like the stack pointer, data aggregation can significantly improve performance, as less data must be saved, as well as analyzed. For this CoojaTrace provides different operators that can, amongst others, calculate

¹<http://rdsp.cs.fau.de>

²<http://www.reactive-web.co.cc>

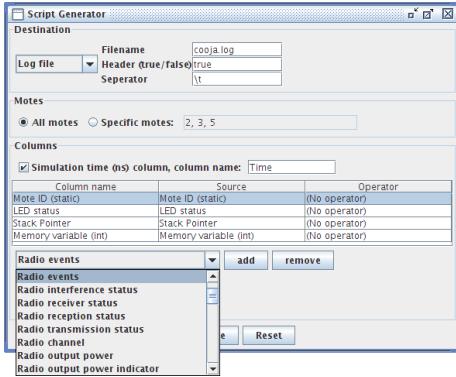


Fig. 1. CoojaTrace script generator

the maximum, minimum, average or standard deviation based on a sliding window.

While Cooja's integrated script editor is probably better suited to write test cases, this is of course possible using CoojaTrace, too. For one, as already mentioned, CoojaTrace can access all of Cooja's objects, and can therefore interact with it. Further on CoojaTrace provides an `assert()` statement, which can be used to either stop and analyze the simulation (e.g. using GDB) or terminate the simulation when running unattended. The latter is especially interesting to improve runtime when running a batch of simulations (e.g. to find an optimal configuration).

For logging CoojaTrace currently supports a log window, a simple text file format, as well as a SQLite³ database. Besides that, an output window, similar to the one provided by Timeline [3] is in development. In addition to the markers provided by Timeline, the visual log-target will also support plotting graphs, which will also allow to plot information like the stack pointer in the same window.

To lower the threshold of creating new queries CoojaTrace provides a wizard (Fig. 1), which supports the user in creating simple queries without the need to understand the syntax.

III. EXAMPLES

To show the power of CoojaTrace we present two examples. For both examples, which can be concatenated, we will use a very simple log format, using three columns `mote`, `what` and `val`, where the first two are used to distinguish what is logged. The time stamp is automatically added as an additional column.

```
1 val log = LogFile("ct.log", List("mote", "what", "val"))
```

A very simple example is the logging of the currently running process of every node.

```
1 for (mote <- sim.allMotes) {
2   log(logt, mote, "process", mote.currentProcess.name)
3 }
```

The `for` statement loops through each object in the `sim.allMotes` collection. For each `mote` the mote's FRP-variable `currentProcess.name` is assigned to the log-target. Thus each time the running process changes, this is

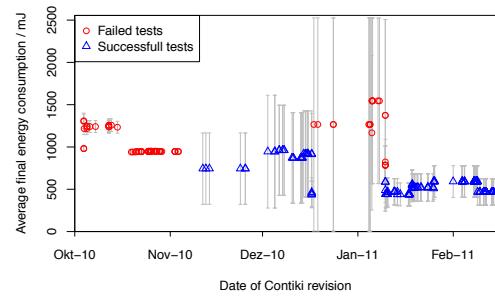


Fig. 2. Average energy usage and standard deviation of 20 nodes in an experiment instrumenting the collect protocol

propagated to the log-destination, and a new row is added. `mote` and "process" are needed to distinguish the log-entries by mote or variable.

A more complex example is the logging of the energy usage of each node, based on the model of Energest [1].

```
1 for(mote <- sim.allMotes) {
2   val rx   = timeSum(mote.radio.receiverOn)
3   val tx   = timeSum(mote.radio.transmitting)
4   val act  = timeSum(mote.cpuMode === "active")
5   val idle = timeSum(mote.cpuMode != "active")
6   val energy: Signal[Double] =
7     rx * 60 + tx * 53.1 + act * 5.4 + idle * 0.1635
8   log(logt, mote, "energy", energy)
9 }
```

The `timeSum`-Function sums up the time a certain condition is true. By multiplying this value with the energy required in this state the energy usage of the mote can be estimated. Due to the FRP based approach the energy is logged each time the state of the node or radio changes.

Using the listing above we measured the energy required to receive 10 messages from every node using the collect protocol. After about twice the expected time the experiment failed. The samples chosen in figure 2 show that there was a severe problem with the network stack in October 2010. While we did not investigate the cause, it also shows that the changes made around New Year improved the situation.

ACKNOWLEDGEMENTS

This work was supported by the Bavarian Ministry of State for Economics, Traffic and Technology under the (EU EFRE funds) grant no. 0704/883 25.

REFERENCES

- [1] A. Dunkels, F. Österlind, N. Tsiftes, and Z. He. Software-based sensor node energy estimation. In *Proc. of the 5th Int. Conf. on Embedded Networked Sensor Systems (SenSys 2007)*, pages 409–410. ACM, 2007.
- [2] F. Österlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt. Cross-level simulation in cooja. In *European Conf. on Wireless Sensor Networks (EWSN 2007), Poster/Demo session*. IEEE, 2007.
- [3] F. Österlind, J. Eriksson, and A. Dunkels. Cooja timeline: A power visualizer for sensor network simulation. In *Proc. of the 8th ACM Conf. on Embedded Networked Sensor Systems (SenSys 2010)*, pages 385–386. ACM, 2010.
- [4] M. Strübe, S. Böhm, R. Kapitza, and F. Dressler. RealSim: Real-time Mapping of Real World Sensor Deployments into Simulation Scenarios. In *Proc. of the 6th ACM Int. Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization (WiNTECH '11)*, pages 95–96. ACM, 2011.

³<http://www.sqlite.org/>

Poster Abstract: TUD μ Net, a Metropolitan-Scale Federation of Wireless Sensor Network Testbeds

Pablo E. Guerrero,
 Alejandro P. Buchmann
 Databases and Distributed Systems
 Technische Universität Darmstadt
 <lastname>@dvs.tu-darmstadt.de

Abdelmajid Khelil
 Dependable, Embedded Systems & Software
 Technische Universität Darmstadt
 khelil@cs.tu-darmstadt.de

Kristof Van Laerhoven
 Embedded Sensing Systems
 Technische Universität Darmstadt
 kristof@ess.tu-darmstadt.de

Abstract—To address the real-world challenges in sensor network evaluation, testbeds have been proposed to enable experimentation without taking the typical deployment hurdles of robustly mounting the hardware, installing batteries, and instrumenting sensor nodes for data collection. In the recent past, several research institutions across Europe proposed to federate their testbeds. However, providing scalability and transparency despite the high heterogeneity in hardware and software between sites proves to be a tough problem. In this paper we introduce TUD μ Net, a metropolitan-scale federation of sensor network testbeds that spans several buildings within a city. We describe its architecture, the current sites and the control infrastructure as solution for managing experiments at metropolitan scale.

I. INTRODUCTION

Current research in sensor actuator networks has mainly concentrated on lab work and simulation experiments which are reproducible yet simplified in nature, and realistic deployments that show feasibility yet make it difficult to explore parameters. Sensor network *simulators* like COOJA [7] or TOSSIM [6] are able to scale up to thousands of nodes, but do not always capture all phenomena from the target environments. Working directly on *deployments* as in [8], [5] exposes the system to the conditions of the real environment, but logistical hurdles such as mounting hardware, installing batteries, programming (i.e., flashing) sensor nodes and instrumenting them for experiment data collection make it harder to repeat experiments. *Testbeds* have been suggested as an in-between solution that allow for experimentation in realistic scenarios as well as theoretical exploration.

Related work in testbeds includes examples such as TWIST [3], an indoor testbed including 204 nodes (TelosB and eyesIFX) where users resort to a set of scripts to indirectly program sensor nodes and collect debug data. MoteLab [9] includes around 190 TelosB nodes spread through offices in a three-story building, where test jobs are defined and scheduled through a web interface while debug data is logged into a centralized database for later evaluation. The Kansei testbed [1] features higher sensor node heterogeneity at a comparable scale (15x14 grid with Stargates and XSM nodes).

This work has been partially supported by the German Research Foundation (DFG) Research Training Group Nr. 1362, *Cooperative, Adaptive, and Responsive Monitoring in Mixed Mode Environments*, GKmM, as well as the Hessian LOEWE Research Priority Program *Cooperative Sensor Communication*, Cocoon.

	realism	scale	controllability	examples
simulator	+	+++	+++	TOSSIM, COOJA
target deployment	+++	++	+	GDI, Agro
single testbed	++	++	+++	MoteLab, TWIST, Kansei
testbed federation	++ ++	+++ ++	++ +++	WISEBED TUD μ Net

TABLE I
 APPROACHES TO SENSOR NETWORK EXPERIMENTATION

In order to reach an even larger scale, the WISEBED EU project [2] aims at aligning several testbeds located in multiple european countries through a unified, loosely coupled management interface. The sheer variability in hardware and software encompassed by these testbeds highlights a challenge that remains open and is targeted in this work. (Table I summarizes the aforementioned experimentation approaches.) We present a metropolitan-scale federation of sensor network testbeds, TUD μ Net, that spans several buildings within a city. We describe our control infrastructure as a solution for managing a variety of experiments at a metropolitan scale, present the ongoing work and planned applications.

II. TUD μ NET OVERVIEW

Similarly to other testbeds, TUD μ Net's architecture is structured in three tiers (cf Fig. 1). The first tier is composed of the sensor nodes which run the software being tested. This can be generated from a normal build system like Contiki's or TinyOS's. Our testbed currently contains a mixture of TelosB, Tmote Sky, JCreate and Z1 nodes, all based on the MSP430 micro-controller, and a variety of sensors populated in each node. The second tier is composed of simple gateways which are permanently connected to a number of sensor nodes via USB cabling. Between 2 to 10 sensors nodes are managed by a gateway, and each testbed is composed of (currently 1 to 30) of these gateways. For this tier we've opted for the Buffalo WZR-300NH router, on which we run OpenWRT customized with sensor node management tools (e.g., serial forwarder, BSL). Finally, a central server orchestrates the entire federation activities. The traffic between the various testbeds (and in turn their gateways) and the server is routed through MANDA, a

Metropolitan Area Network (of DArmstadt) operated at Gbit/s speed. TUD μ Net currently encompasses three testbeds, chosen to fit certain well-defined scenarios. The first testbed is located at the CS Dept.'s office & lab building, which spans three floors. This is mostly for experimentation with networking and sensing/actuation aspects. The second testbed is located at the GKmM Lab at the Technology and Innovation Center (TIZ bdg.), where a disaster scenario arena is monitored with gas detectors uniformly spread through multiple planes. The third deployment is located at the Architecture Dept.'s solar house (the surPLUShome), an award-winning architectural design that produces surplus energy above what it uses. These sites are summarized in Table II.

	site		
	CS Dept.	GKmM Lab	surPLUShome
nodes	62 TelosBs, 20 Z1s	50 TelosBs	20 Z1s
sensors	light, humidity, temperature acceleration	light, humidity, temperature, CO CO ₂	light, high precision temp., humidity and CO
focus	networking aspects, sensing & actuation	gas plume detection	environmental monitoring

TABLE II
CURRENT TUD μ NET TESTBEDS

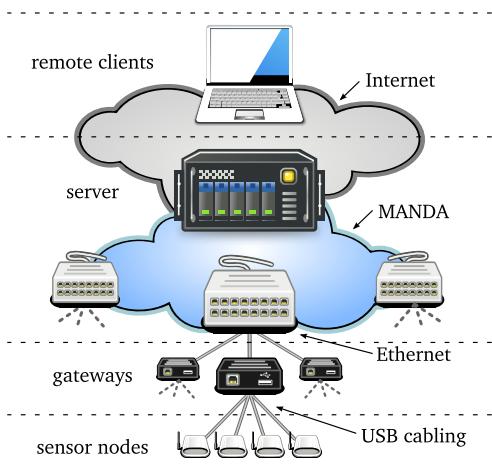


Fig. 1. TUD μ Net's architecture

A. Logical Organization

Beyond the physical structure, TUD μ Net organizes all its sensor nodes by means of node *zones*. We pick up the idea once started in MoteLab, and extend it with our concepts from Scopes [4], a network structuring mechanism for sensor networks. At their core, zones are simple subsets of a parent zone (at the top, the universal set contains all nodes). Fig. 2 depicts the current structure. The federation enables concurrent jobs (i.e., any two jobs that partially or totally overlap temporally), as long as they are scheduled for zones with disjoint sets of nodes. This requires a verification of node availability even among sibling zones, since these are not necessarily disjoint.

As with any flat organization, as the system scales up, the benefits of a hierarchy become more evident. The hierarchy can be easily altered through the administrative interface.

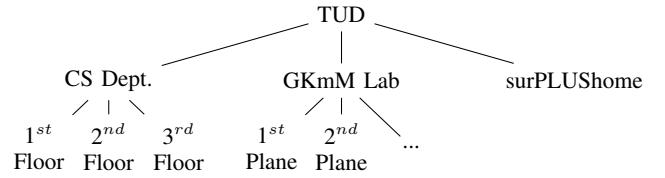


Fig. 2. Logical structure

B. Implementation

TUD μ Net's core is based on that of MoteLab, on which we've implemented our concept of zones. Given that we perform the management of the zone hierarchy centrally, verifying the availability of nodes for a submitted job is straightforward. Authorized users can log into the system, upload binary images, schedule jobs at different zones and retrieve their job's data. We have implemented a number of features like *public/private* jobs (e.g. for sharing common jobs like data collection, setting node ids, etc.); a simple visualization of the federation zones' status as an overlay on a map; scripts for direct node access, among others.

III. CONCLUSIONS AND FUTURE WORK

Building a testbed federation, even at metropolitan scale, poses exciting challenges which require neat engineering solutions. Our hardware infrastructure, together with the concept of zones, offers a simple and manageable solution to federating multiple testbeds. We are working on a number of aspects, including expanding the geographic coverage; deploying nodes on remote controlled and autonomous vehicles as well as human-worn; further exposing node and sensor heterogeneity to the user; zone-based access control, and sensor node fault emulation.

REFERENCES

- [1] Arora, A. et al. Kansei: A High-Fidelity Sensing Testbed. *IEEE Internet Computing*, 10:35–47, 2006.
- [2] Chatzigiannakis, I. et al. WISEBED: an Open Large-Scale Wireless Sensor Network Testbed. In *Proc. 1st Int. SENSAPEAL 2009*, volume 29 of *LNICST*, pages 68–87, ICST, September 2009. Springer.
- [3] Handziski, V. et al. TWIST: A Scalable and Reconfigurable Testbed for Wireless Indoor Experiments with Sensor Networks. In *Proc. 2nd Int. Workshop REALMAN*, pages 63–70, NY, USA, May 2006. ACM.
- [4] Jacobi, D. et al. Structuring Sensor Networks with Scopes. In *Proc. 3rd EuroSSC*, Zurich, Switzerland, October 2008. IEEE.
- [5] Langendoen, K.G. et al. Murphy Loves Potatoes: Experiences from a Pilot Sensor Network Deployment in Precision Agriculture. In *Proc. 14th Int. WPDRTS*, pages 1–8, April 2006.
- [6] Levis, P. et al. TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications. In *Proc. of 1st SenSys*, pages 126–137, NY, USA, November 2003. ACM.
- [7] Osterlind, F. et al. Cross-Level Sensor Network Simulation with COOJA. In *Proc. of 31st LCN*, pages 641–648. IEEE, November 2006.
- [8] Szewczyk, R. et al. Lessons from a Sensor Network Expedition. In Holger Karl, Andreas Willig, and Adam Wolisz, editors, *EWSN*, volume 2920 of *LNCS*, pages 307–322. Springer, 2004.
- [9] Werner-Allen, G. et al. MoteLab: a Wireless Sensor Network Testbed. In *Proc. 4th Int. IPSN*, Piscataway, NJ, USA, April 2005. IEEE.

Security

Demo Abstract: On preventing GTS-based Denial of Service in IEEE 802.15.4

Roberta Daidone
University of Pisa
Pisa, Italy

Email: roberta.daidone@iet.unipi.it

Gianluca Dini
University of Pisa
Pisa, Italy

Email: gianluca.dini@iet.unipi.it

Marco Tiloca
University of Pisa
Pisa, Italy

Email: marco.tiloca@iet.unipi.it

Abstract—The IEEE 802.15.4 standard features some optional services, including the *Guaranteed Time Slot (GTS)* mechanism. It provides network devices with collision-free access to the medium to assure Quality of Service. GTS suffers from a severe security vulnerability: an adversary can easily perform a *Denial of Service* attack by selectively jamming collision-free communications. We present *Secure GTS*, our solution to the GTS-based Denial of Service attack, and our implementation for the TinyOS platform on Tmote Sky motes. Our test application shows that Secure GTS manages to prevent Denial of Service attacks.

I. INTRODUCTION

The IEEE 802.15.4 communication standard [1] is designed for low-cost, low-power devices organized in a *Personal Area Network (PAN)*, and is widely adopted for applications based on Wireless Sensor Networks (WSNs). Typically, these applications rely on devices with scarce hardware resources, and require to boost communication performance, or even achieve some *Quality of Service (QoS)* guarantees.

IEEE 802.15.4 provides the *Guaranteed Time Slot (GTS)* mechanism. GTS allows network devices to ask the *PAN Coordinator* for a dedicated time slot. If the request is accepted, they can communicate during an exclusively pre-assigned slot, so that no collisions occur while accessing the medium.

However, GTS has been proved to suffer from a severe security vulnerability. As described in [2], an adversary can easily perform a selective jamming attack, and disrupt communication even during dedicated GTS slots. This can jeopardize even the entire network activity, thus resulting in an actual *Denial of Service* attack.

Secure GTS is our solution to prevent the GTS-based Denial of Service attack, and consists in the following steps:

- 1) Hide the GTS slots assignment process from the adversary, by encrypting related information.
- 2) Shuffle the allocation of dedicated slots over time. This forces the adversary to perform the attack randomly.

We implemented Secure GTS for the TinyOS platform [3] and the Tmote Sky motes [4]. Our implementation is compliant with the IEEE 802.15.4 standard.

In order to test Secure GTS, we consider a simple application scenario with real sensor devices. We show that we succeed in preventing the adversary from disrupting collision-free communications, and that Secure GTS reduces the attack success rate up to 1/7.

In the following, we provide a brief summary of IEEE 802.15.4 and its GTS mechanism. Then, we describe the GTS-based Denial of Service attack, and present our solution. Finally, we show how our implementation of Secure GTS effectively prevents Denial of Service attacks.

II. IEEE 802.15.4 AND GTS

An IEEE 802.15.4 *Personal Area Network (PAN)* includes a *PAN Coordinator*, which is responsible for managing network activity. Communication can rely on the *beacon-enabled* mode, in which the PAN Coordinator periodically broadcasts *beacon frames*. Thus, the medium access is bounded as a sequence of *superframes* delimited by two consecutive beacon frames, and network devices are synchronized with each other.

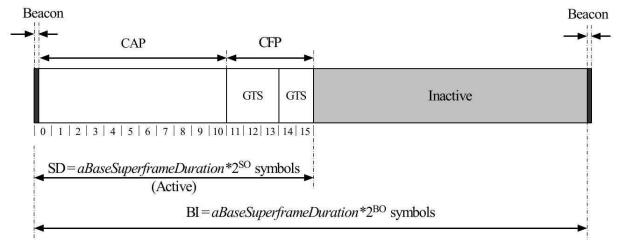


Fig. 1. IEEE 802.15.4 superframe structure.

A superframe is composed by an *active portion* and an *inactive portion* (see Figure 1). The active portion is composed by 16 equally sized *superframe slots*, and is divided into a *Contention Access Period (CAP)* and a *Contention Free Period (CFP)*. During the CAP, devices access the medium on a contention basis, according to a slotted CSMA-CA algorythm. Instead, during the CFP, devices can ask the PAN Coordinator for a dedicated portion of the superframe, namely a *GTS Slot*. Thus, they are able to access the medium without colliding with each other. This mechanism is known as *Guaranteed Time Slot (GTS)*, and is managed by the PAN Coordinator.

GTS Slots (*Slots* for short) can be composed by one or more superframe slots. Each device can send a *GTS Allocation Request* to the PAN Coordinator and ask for one Slot, specifying the amount of needed superframe slots and the traffic direction. Also, nodes can ask the PAN Coordinator to deallocate previously assigned Slots. Both allocation and

deallocation requests take place by means of *GTS Request Command* frames.

The PAN Coordinator assigns up to seven Slots in a *First Come First Served (FCFS)* fashion. Then, it includes in each beacon frame i) the list of devices whose request has been accepted; and ii) the time they are supposed to access the medium, i.e. when their Slot starts.

III. ATTACK

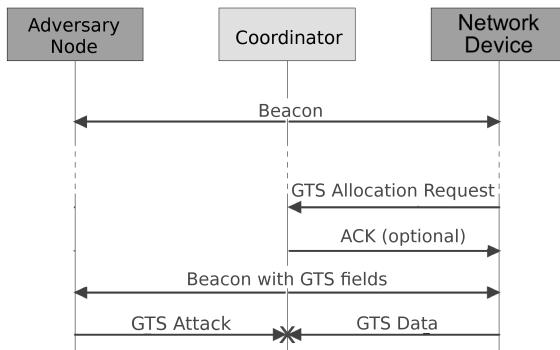


Fig. 2. GTS attack.

GTS allocation requests are managed by the PAN Coordinator, which notifies the accepted ones using beacon frames. Broadcasting a beacon results in a GTS vulnerability [2].

We define the *sniper attacker*. He selects a target (i.e. a user), and exploits the knowledge of which users have been granted a collision-free Slot, in order to interfere with communications of his victim. Thus, the sniper attacker realizes a *Denial of Service (DoS)* attack against his target.

Figure 2 shows a GTS-based Denial of Service attack. The sniper attacker eavesdrops the medium to extract the GTS-related information from the beacon frame. Thus, the adversary becomes aware of during which specific Slot a specific user will access the medium. This means it is very easy for the sniper attacker to cause collisions between legitimate GTS clients and the PAN Coordinator, corrupt data, and *selectively* interfere with transmissions.

IV. SOLUTION

GTS is vulnerable because the adversary can access the GTS-related information within beacon frames. A simple solution to avoid this would be *encrypting* and *authenticating* GTS-related information. However, the standard does not allow for encrypting the beacon payload portion containing GTS-related information. Thus, another solution is needed.

Secure GTS is our standard-compliant solution to the GTS-based DoS attack. Note that Secure GTS is not effective in case the adversary interferes with all transmissions by continuously jamming all available channels. However, we believe that, in a WSN, it is very likely that the adversary performs attacks by means of a sensor node. Therefore, it is very likely for him to behave as described, making Secure GTS effective.

Secure GTS consists in two steps:

1) *MAC frames smart authentication and encryption.* Secure GTS prevents the attack by moving the GTS-related information to a different portion of the beacon frame. Thus, GTS-related information can be authenticated and encrypted. MAC command frames should be encrypted to avoid the adversary recognizing GTS Request Commands even by analyzing network traffic. Authentication prevents the adversary from spreading fake GTS Request Commands all around the network.

2) *Random Slots allocation.* According to the standard, if no deallocations occur, assigned Slots do not change their position in the CFP. As a consequence, the attacker is still able to infer the GTS-related information by observing the sequence of transmissions during the CFP. Secure GTS changes the position of Slots randomly on a superframe basis to make this analysis pointless.

We implemented Secure GTS by extending an open-source TinyOS implementation of the IEEE 802.15.4 standard [5]. We tested Secure GTS on a realistic scenario which includes four Tmote Sky motes: one PAN Coordinator, two sender nodes and one sniper attacker. The effectiveness of our countermeasure has been evaluated considering the probability of success of the attacker. Secure GTS reduces the probability of success of the sniper attacker to 1/7. Since the GTS information carried within the beacon is encrypted, the sniper attacker can only pick a Slot at random.

V. CONCLUSION

We have presented Secure GTS, our solution to the GTS-based Denial of Service attack in IEEE 802.15.4. We implemented our solution for TinyOS on Tmote Sky motes, and tested it on a real application scenario. Our tests show that Secure GTS reduces the attack success rate up to 1/7.

ACKNOWLEDGMENT

This work has been supported by EU FP7 Network of Excellence CONET (Grant Agreement no. FP7-224053) and EU FP7 Integrated Project PLANET (Grant agreement no. FP7-257649). Thanks also to Mário Alves and Ricardo Severino from ISEP for the useful feedback during the early stages of this work and the standard GTS implementation for TinyOS.

REFERENCES

- [1] IEEE Std. 802.15.4-2006, *IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*, Institute of Electrical and Electronics Engineers, Inc., New York, September 2006.
- [2] R. Sokullu, O. Dagdeviren and I. Korkmaz, "On the IEEE 802.15.4 MAC Layer Attacks: GTS Attack," in *Proceedings of the Second International Conference on Sensor Technologies and Applications. SENSORCOMM '08*, August 2008, pp. 673-678.
- [3] "TinyOS Home Page." [Online]. Available: <http://www.tinyos.net/>
- [4] Moteiv Corporation, "Tmote Sky: Datasheet," 2006. [Online]. Available: <http://www.cs.jhu.edu/~cliang4/public/datasheets/tmote-sky-datasheet.pdf>
- [5] J.-H. Hauer, R. Daidone, R. Severino, J. Büsch, M. Tiloca and S. Tennina, "Poster Abstract: An Open-Source IEEE 802.15.4 MAC Implementation for TinyOS 2.1," in *Proceedings of 8th European Conference on Wireless Sensor Networks (EWSN)*, February 2011.

Poster Abstract: DoS Detection with Markov Chains

Denise Dudek

Institute of Telematics, Karlsruhe Institute of Technology, Karlsruhe, Germany, Email: denise.dudek@kit.edu

Abstract—The application of wireless sensor networks in safety-relevant scenarios often fails for reasons of security concerns. While some attacks may be prevented using cryptographic means, e.g., message authentication or encryption, preventing others, such as *Denial of Service* (DoS) with current technology is difficult. This poster proposes to detect DoS-related traffic anomalies using first order Markov chains. First results show that the achieved false positive error rates are low: 90% of the nodes reach 2.5% or less. Even with false positive rates of only 1%, hit rates of over 95% were achieved by most nodes.

I. INTRODUCTION

Although prevention of DoS attacks is hard, a network operator might well be interested in knowing if an attack occurs. While detection tools exist for classical networks, WSNs differ from those in two ways. First, they operate under much heavier resource constraints. Second, they are typically designed for a specific purpose. The application purpose frequently defines distinctive communication patterns that are usually absent in classical networks; however, those patterns can be exploited for attack detection.

The main design goals for attack detection in wireless sensor networks are as follows: In contrast to classical networks, there are no widely spread attack tools — and thus no known attacks. Therefore, attack detection in wireless sensor networks must necessarily be able to **handle new forms of attack**. Furthermore, sensor networks are designed to work autonomously with little human interaction. Thus, **no undue increase in configuration effort** should be introduced by the anomaly detector. Finally, considering that Denial of Service attacks usually cause heavy strain on a victim's resources, DoS detection schemes must exhibit **favourable runtime behaviour** and must not add disproportionately to this strain.

The Markov chain approach presented meets the above requirements. As an anomaly based attack detection scheme, it is able to deal with unknown attacks. The nodes calculate traffic profiles on their own, thus reducing human interaction. The model is application aware in that it captures an application's distinctive communication patterns. And finally, due to the simplicity of the model, the computational complexity is low.

II. ANOMALY DETECTION USING MARKOV CHAINS

A Markov chain C can be described as $C = \{I, P\}$, where I is a set of k states and P is a k -dimensional stochastical transition matrix, i.e., all its coefficients p_{ij} are non-negative and $\sum_{j=1}^k p_{ij} = 1$, for all i .

For anomaly detection, network traffic is regarded as a sequence of states. Due to the wireless environment's indeterministic properties, it is difficult to identify a distinct successor

state for any state of the Markov chain. Thus, ergodic Markov chains – chains that satisfy $p_{ij} \neq 0$ for all i, j are chosen to represent the traffic.

Within the ergodic model, the principle of anomaly detection is as follows: Time is divided into equal-length observation intervals. During the first phase – the *learning phase* – nodes collect information about the expected normal traffic. This is achieved by observing a predefined, potentially multi-dimensional set of *traffic features*. At the end of the learning phase, each node possesses a time series X of traffic observation points. From those points, a *traffic profile* is built that describes X .

Intuitively, each point in X could define a state in the Markov chain; however, if the feature space is not partitioned, the number of states usually becomes quite large. Contrary to works like [1] and [2], the approach presented in the poster allows to impose a strict limit on the number of states. For that purpose, the observation points are merged into a small number of clusters. This number is in $O(\sqrt{n})$, where n is the duration of the learning phase given in observation intervals. Each of the clusters is made up of similar observation points and corresponds to a state in the Markov chain. Clustering thus limits the state space.

Multidimensionality of the feature space is another often overlooked source of state space explosion. An example is [2]. Moreover, dimensionality complicates the cluster analysis. To reduce dimensionality while still allowing for a good cluster analysis, the feature space is projected onto the first principal component of the sample X . Algorithms to efficiently approximate the first principal component exist.

The transition matrix is constructed so that transitions that were observed rarely during the learning phase have a low probability. The matrix can be obtained using the chronological order of X and mapping each observation point to its according state.

At the end of an observation interval of the subsequent *runtime*, a new observation is classified as belonging to some state in the Markov chain.

The product of the last t transition probabilities is called the *anomaly score* of the last t transitions. The lower the score, the more rare transitions have occurred during that time. If the anomaly score is smaller than some threshold value, the sequence of the last t transitions is identified as *anomalous*. A sliding window scheme is applied so that an anomaly score can be computed each interval. Also, the anomaly threshold can be calculated at the end of the learning phase and does not need to be set manually.

Parameter	Value(s)
#nodes	9, 200
Topology	regular grid, 10m distance
Observation interval [s]	5
Learning phase [intervals]	120
Runtime [intervals]	240
Attack duration [intervals]	120
Attack strength [msg/s]	5
Attacking nodes [%]	5

TABLE I
SCENARIO PARAMETERS

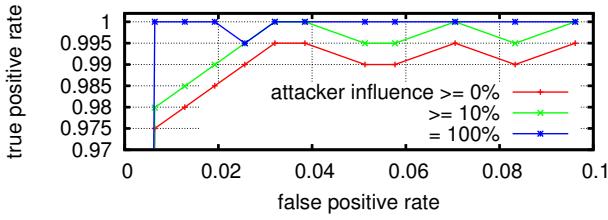


Fig. 1. ROC for the small scenario

III. RESULTS

The anomaly detection scheme described in the previous section was implemented and tested using OMNeT++ 4 and MiXiM for wireless network support. Each sensor node implements the IEEE 802.15.4 Narrowband MAC layer. On the network layer, multihop communication is achieved by flooding.

A 2-dimensional feature space was used for anomaly detection. As traffic features, the number of messages received per observation interval by the *application layer* and the *network layer* respectively were chosen. A protocol suite designed for a secure area surveillance scenario [3] was used for evaluation. The protocols generate traffic simultaneously, overlaying each other's traffic patterns. The protocol suite is referred to as *compound application*. Additionally, a Denial of Service attacker was implemented. An attack lasts 120 intervals and is executed by broadcasting messages into the network. Table I lists the scenario parameters.

The Receiver Operating Characteristic (ROC) is an evaluation metric typically used for anomaly detectors. It maps the *false positive rate* to the *true positive rate*. Figures 1 and 2 show the ROC for the 9 and 200 nodes scenario respectively.

The plots show the ROC for three cases: First, all nodes, regardless of attacker influence, were evaluated. Then, nodes with less than 10% attacker influence were excluded; finally, only nodes with 100% attacker influence were included in the evaluation. The difference between the 10% case and the 100% case is marginal; good results can be achieved even if a node only receives 10% of the attacker's messages. The results for both scenarios show no qualitative difference, although the respective network sizes are an order of magnitude apart. This suggests that the method scales well.

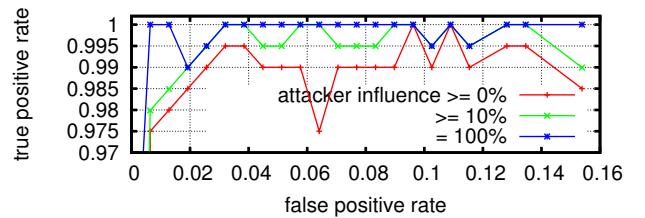


Fig. 2. ROC for the large scenario

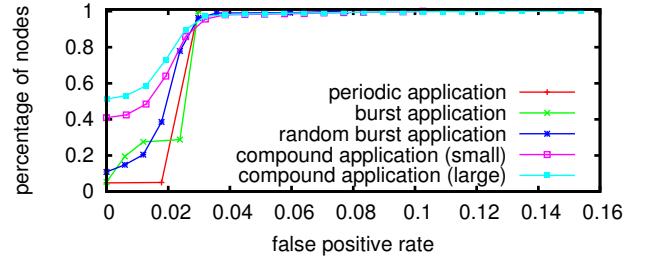


Fig. 3. Cumulative distributions for the false positive rate

To qualify the ROCs, Figure 3 shows the distribution of the false positive rates for both scenarios. Additionally to the compound application, three other applications were implemented for comparison: a periodic application that generates traffic periodically, a regular burst application that adds traffic bursts to the periodic application at regular intervals, and a random burst application that does the same randomly.

The figure shows that for all scenarios, about 90% of the nodes achieved a false positive rate around 2.5% or better. The highest false positive rates were only measured at less than 0.1 percent of the nodes. This suggests that reducing the false positive rates is possible by collaboration – e.g., local voting – between the nodes.

IV. CONCLUSION

It is possible to apply first order Markov chains to detect traffic anomalies in a Denial of Service scenario in wireless sensor networks. It was shown that even for very low false positive error rates, hit rates over 95% were achieved.

Ongoing work concerns adding collaboration between nodes to reduce the maxima in observed false positive rates. Future research also includes dynamically adapting the traffic profile to improve the accuracy of the model at runtime.

REFERENCES

- [1] I. C. Paschalidis and Y. Chen, "Statistical anomaly detection with sensor networks," *ACM Trans. Sen. Netw.*, vol. 7, pp. 17:1–17:23, September 2010.
- [2] Y. Gao, C. Chen, J. Bu, W. Dong, and D. He, "Icad: Indirect correlation based anomaly detection in dynamic wsns," in *Proceedings of the Wireless Communications and Networking Conference (WCNC), 2011 IEEE*, march 2011, pp. 647–652.
- [3] D. Dudek, C. Haas, A. Kuntz, M. Zitterbart, D. Krüger, P. Rothenpieler, D. Pfisterer, and S. Fischer, "A wireless sensor network for border surveillance," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems, ser. SenSys '09*. New York, NY, USA: ACM, 2009, pp. 303–304.

Poster Abstract: Reusing AES Coprocessor in Public Key Cryptography

Vladimir Cervenka, Lubomir Mraz, Milan Simek

Department of Telecommunications, Brno University of Technology, Czech Republic

cervenka.v@phd.feeec.vutbr.cz

Abstract— Providing proper security with reasonable energy consumption in Wireless Sensor Networks (WSNs) is still an important issue. Several security protocols have been developed to address this issue but most of them usually introduce great overhead in terms of energy or communication. Our effort is to propose energy efficient solution. We introduce and analyze energy efficient system providing symmetric key cryptography and public key cryptography so that both are possible to compute with help of AES hardware accelerator. We show that this hardware based solution is more than 100 times more energy efficient than purely software solution.

I. INTRODUCTION

Confidentiality is not an essential requirement for most applications; instead we are looking for a message authenticity. For example, someone can be interested in economic benefits in terms of Smart Grids. People can try to tamper power meters to reduce measured consumption. That is why the data authentication is needed.

As the key management is essential in WSN security, an appropriate key management scheme is needed. So far, the simplest solution of a key management scheme is a network-wide shared key. A slightly better solution is to use a network-wide shared key to establish a set of link pairwise keys and then erase the network-wide shared key. There are plenty of different solutions such as random pair-wise key establishment, variations of random key pre-distribution schemes, grid-based pre-distribution schemes, trusted key distribution center and so forth [1]. One of them is also the Public Key Cryptography (PKC), well known in classical computer networks and with the help of the Elliptic Curve Cryptography (ECC) it is suitable even for WSN.

II. HARDWARE PLATFORMS IN WSN

Probably the most used WSN platforms are TelosB, TmoteSKY, MICAz and IRIS. They have been designed to meet the LR-WPAN requirements, which means low bit rate, low power and low cost. They are equipped with only 8/16-bit microcontrollers operating at the maximum frequency of 8 MHz.

Even though, there are efforts to implement PKC, particularly ECC, to these platforms like TinyECC [2], NanoECC [3], WM ECC [4] or [5] the proposed implementations are still quite energy demanding due to long processing times.

For our implementation we chose the microcontroller EFM32G890F128 as the best options because of its extremely low power consumption, very powerful processing and hardware support for AES-128 and AES-256. This microcontroller is based on ARM Cortex-M3 core, includes a 32-bit RISC processor which can achieve as much as 1.25 DhystoneMIPS/ MHz, 128 kB Flash and 16kB RAM [6]. In spite of these advantages, a unit price is not higher than the one for 8/16-bit microcontrollers.

III. CRYPTOSYSTEM ARCHITECTURE

We decided to make use of great possibilities of AES services defined in IEEE 802.15.4 standard, particularly the AES-CCM-64 with only 8 Bytes of overhead. However, IEEE 802.15.4 does not define the processes of key distribution or node authentication. Taking advantage of ECC on constrained devices Ephemeral Elliptic Curve Diffie-Hellman (ECDHE) could be used to establish shared keys and ECDSA (particularly the ECC-based TLS handshake) for authentication.

Noting the process power of 32-bit ARM processor on one hand, and the hardware support for AES-256 on the other, the AES-CCM ECC Cipher Suites for TLS proposed in draft [7] seems to be the most efficient solution for WSN field. The cipher suites use ECDHE as its key establishment mechanism and can be used with DTLS. They are based on the authenticated encryption with associated data (AEAD) algorithm defined in RFC 5116. Moreover, the AEAD_AES_128_CCM algorithm, used in this cipher suite, actually uses AES-128 as a block cipher. That is the key element which allows utilization of an AES hardware accelerator. The chosen cipher suite TLS_ECDHE_ECDSA_WITH_AES_128_CCM requires additional support for secp256r1 curve and SHA-256 hash algorithm.

Although, there are more ECC-based algorithms optimized for 32-bit processors available like a CompactECC, [8], [9], they are supposed to be proceeded by processor as whole. Instead, the AES-CCM ECC Cipher Suites make use of AES algorithm, which is highly advantageous in such a constrained environment as WSN. Furthermore, it is possible to utilize the AES hardware accelerator and thus achieve extremely low power consumption without additional hardware because AES co-processors are available in the most radio transceivers compliant to IEEE 802.15.4 (AT86RF231, AT86RF231, TI CC2420, TI CC2520) and new microcontrollers.

IV. EVALUATION

The hardware AES operations on EFM32G890F128 are available in Energy Mode 0 (EM0) and Energy Mode 1 (EM1). The main processor is on, during the EM0 and can process other data, whilst in the EM1 the processor is in sleep mode, but the AES support is still available. Fig. 1 shows the dependency of current consumption on processing time for both possible modes.

A comparison of the energy consumption of hardware and software based AES solutions is available in Fig. 2. For this comparison the software implementation was based on [9]. This algorithm is specially optimized for architecture of ARM Cortex M-3. Even if we compare the best results for software encryption (749,52 nJ) and hardware encryption (7,29 nJ), at 32 MHz, it is clearly visible that hardware implementation reduces energy consumption more than by a factor of 100. For further comparison, the hardware implementation of AES encryption on MICAz consumes 1,83 μ J and 14,30 μ J on TelosB [10]. The software implementation then consumes 39,08 μ J and 28,16 μ J on MICAz and TelosB, respectively [10].

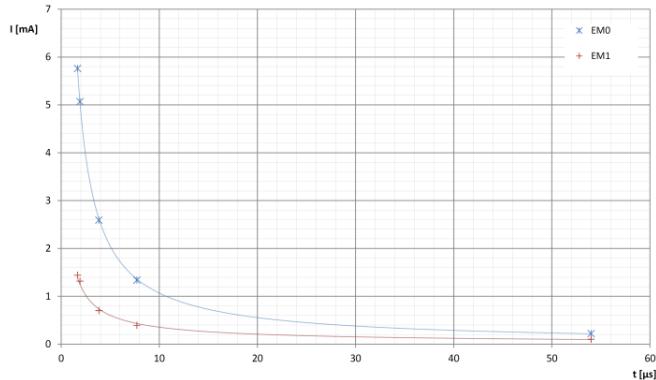


Fig.1. Current consumption of Hardware AES implementation on microcontroller EFM32G890F128. Encryption/ decryption one 128-bit data block with 128 bit key. EM0 = Energy mode 0 – Run Mode; EM1= Energy mode 1 –Sleep Mode. Measured for $V_{DD} = 3.0$ V

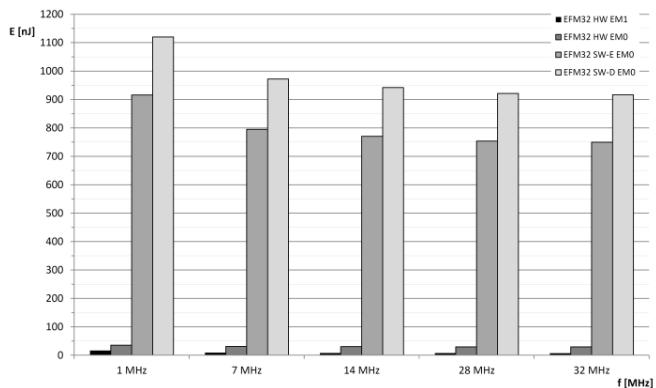


Fig. 2. Comparison of energy consumption: Hardware vs. Software AES implementation. Encryption/ decryption one 128-bit data block with 128 bit key. EM0 = Energy mode 0 – Run Mode; EM1= Energy mode 1 –Sleep Mode; HW = Hardware encryption/decryption; SW-E = Software encryption; SW-D = software decryption.

V. CONCLUSION AND FUTURE WORK

In our poster we propose an implementation of a key management system for Wireless Sensor Networks based on the Public Key Cryptography. We investigated the efficiency of hardware accelerator for AES encryption and decryption in comparison with software solution. Furthermore, we present a method how to efficiently reuse this accelerator to achieve confidentiality, data integrity and even the key establishment with data origin authentication via Ephemeral Elliptic Curve Diffie-Hellman and Elliptic Curve Digital Signature Algorithms, while taking advantage of AES-CCM ECC Cipher Suites. We prove that the hardware based cryptographic solution is more than 100 times more energy efficient than the software based solution. Moreover, we use only one microcontroller so no additional special customized hardware is needed.

We plan to fully implement and evaluate our system with other implementations in future. Also more complete solution based on behavior monitoring and trust management would be desirable.

VI. REFERENCES

- [1] A. K. Pathan, Eds., Security of Self-Organizing Networks. Boca Raton: Taylor and Francis Group, 2011.
- [2] A. Liu and P. Ning, “TinyECC: a configurable library for elliptic curve cryptography in wireless sensor networks,” in Proceedings of the 7th International Conference on Information Processing in Sensor Networks, pp. 245–256, 2008.
- [3] P. Szczechowiak et al. “NanoECC: testing the limits of elliptic curve cryptography in sensor networks,” Proceedings of the 7th international Conference on Information Processing in Sensor Networks, pp. 305–320, 2008.
- [4] H. Wang and Q. Li, “Efficient implementation of public key cryptosystems on mote sensors,” in Proceedings of the International Conference on Information and Communication Security, pp. 519–528, 2006.
- [5] A. Kargl et al. “Fast Arithmetic on ATmega128 for Elliptic Curve Cryptography.” International Association for Cryptologic Research Eprint archive, October 2008.
- [6] Energy Micro, “EFM32G890 DATASHEET” datasheet, 2011 [Revised May. 2011].
- [7] D. McGrew et al. “AES-CCM ECC Cipher Suites for TLS” draft-mcgrew-tls-aes-ccm-ecc-02, Oct. 2011.
- [8] M. Aydos et al. “An High-Speed ECC-based Wireless Authentication Protocol on an ARM Microprocessor,” in Proceedings of the 16th Annual Computer Security Applications Conference, 2000.
- [9] Ø. Ekelund. “Low Energy AES Hardware for Microcontroller.” M.A. thesis, Norwegian University of Science and Technology, Norway, 2009.
- [10] M Jongdeog Lee et al. “The price of security in wireless sensor networks.” *Computer Networks*, Vol. 54, No. 17, pp. 2967–2978, December 2010.

Poster Abstract: Topology and Deployment Impact on Key Distribution in Wireless Sensor Networks

Bruno Trevizan de Oliveira, Cíntia Borges Margi and Wilson Vicente Ruggiero
Universidade de São Paulo (Department of Computer and Digital Systems Engineering) – Brazil
Email: {btrevizan, cbmargi, wilson}@larc.usp.br

Abstract—An important issue to the effectiveness of existing WSN security architectures is how secret keys are distributed. Topology and deployment characteristics impact the network energy consumption, memory usage and key connectivity. The main contribution here is an analysis of how these characteristics affect key distribution in WSNs.

I. INTRODUCTION

Many Wireless Sensor Networks (WSNs) applications require security services, such as confidentiality, data integrity and source authenticity. Given the resource constrained requirements of WSNs, several specific security architectures based on symmetric cryptography were developed, such as MiniSec [1]. However, an important issue to the effectiveness of these architectures is how secret keys are distributed.

Key establishment approaches for WSNs must satisfy several security and functional requirements, which are often conflicting, such as resilience, authentication, connectivity and reduced memory usage. Existing solutions for key distribution in WSNs are classified into three categories: pre-distribution, arbitrated, and self-enforcing [2]. Pre-distribution keying approach involves loading keys into sensor nodes prior to deployment, arbitrated protocols use more powerful nodes to execute complex tasks, and self-enforcing is the dynamic establishment of keys generally using asymmetric cryptography.

The contribution of this work is the analysis of key distribution in WSNs based on topology and deployment characteristics. We discuss how deployment characteristics can affect security and flexibility requirements, and how topology information can be useful to optimize the key distribution task.

II. ANALYSIS

The requirements considered are:

- *Resilience* – resistance against node capture and secret information recovery from its memory. Resilience is given by the network fraction affected through this attack on a single node.
- *Authentication* – assurance that the communicating nodes are able to verify each other's identity in a secure way.
- *Node Position Independence* – independence of nodes positioning information for initializing the network keys.
- *Scalability* – ability to support large networks and allow the introduction of new nodes with no security loss.

To evaluate the proposed approaches for each scenario three efficiency metrics are used: energy consumption, memory

usage and key connectivity, which is the probability of the sensor nodes to be able to share keys [2].

A. Key Distribution Approaches Analysis

Self-enforcing schemes using pairings have been identified as the most efficient solution to WSNs key distribution issue [3], since they avoid exchange and storage of large keys and certificates. Nevertheless, sensor memory constraints could make it impractical for many applications, since it uses a significant amount of memory. For instance, TinyPBC [3], uses 30% of ROM and 90% of RAM on the MICAz [4], and for the TelosB mote platform [4] it occupies 63% of ROM and 33% of RAM.

Arbitrated keying schemes rely on a trusted point for key establishment which is attractive when station or cluster heads are available. However, there are some drawbacks: the trusted point becomes a preferred target for attacks, and key exchange involves communication, increasing energy consumption.

Pre-distribution schemes are attractive since they do not generate a preferred target for attacks and avoid communication, thus decreasing energy consumption. The drawback of this approach is its application scenario dependence. Its main strategies are: (i) network-wide key; (ii) pairwise key sharing; and (iii) group-wide or random key pre-distribution [2].

B. Deployment Characteristics

1) *Tamper-proof*: this feature allows discarding the resilience requirement, simplifying the keying task. In this case, pre-distribution schemes using network-wide strategy provide sufficient authentication, since tamper-proof ensures that all nodes that know the shared key are legitimate. Network-wide keying provides node position independence and scalability; since all nodes share the same key, it has low memory usage, does not add energy consumption and provides full key connectivity. Network-wide keying approach is not secure for a WSN that is not tamper-proof, since compromising a single node reveals the entire network key. A solution with greater resilience would be pairwise keying, in which a node capture only compromises its communication, not affecting other nodes. However, pairwise keying requires higher memory usage, since it needs to store a different key for each node in the WSN.

2) *Placement*: Node placement can be deterministic, when the nodes are deployed in a controlled or random manner, when there is no control of the final node position and the

network layout, such as when nodes are thrown into the deployment area from an airplane. Deterministic placement facilitates pre-distribution schemes implementation, since it is possible to determine a node neighbors beforehand, and thus, which nodes can communicate with it. This feature voids the node position independence requirement. Thus, a pre-distribution scheme using pairwise keying, adding into each node only the necessary keys to communicate to their neighbors, would reduce the memory usage, maximize connectivity, and comply with authentication and resilience requirements. Random placement increases the requirement of node position independence, making it difficult to adopt the pre-distribution scheme. An option is to choose a random distribution with keys for node groups, reducing memory usage, but affecting authentication, resilience and key connectivity. Alternatively, one could use an arbitrated solution, simpler to implement and meeting the security requirements, but that increases energy consumption and is only effective if a trusted point is available to all nodes that need to establish keys.

3) *Dynamic Node Addition and Number of Nodes:* networks that receive devices dynamically after their deployment require greater scalability, which complicates the use of pre-distribution schemes with pairwise, even if their position is provided, since the neighboring node has to carry the keys from the beginning. The number of nodes is another deployment characteristic that affects the scalability requirement. If the network has a large number of devices, nodes have to carry a lot of keys, preventing more resilient strategies for pre-distribution due to high memory usage. For both cases it is interesting to use an arbitrated scheme, given its constraints and the consequences mentioned above, or alternatively, to adopt a pre-distribution with random strategy and deal with its limitations.

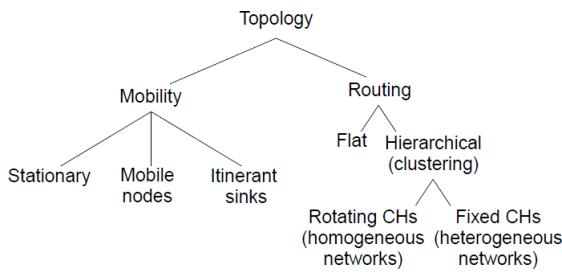


Fig. 1. Network topology taxonomy

C. Topology

The topology-oriented analysis is guided by the taxonomy presented in Figure 1.

1) *Mobility:* Stationary topology does not change the requirements for key distribution, or present any characteristic that changes the previous analysis. A mobile nodes topology, in general, makes the network deployment equivalent to random placement, regardless of their deployment characteristics. Mobility patterns could be used to apply a specific scheme for key management. However, a general solution is the same as that presented to the random placement (Section II-B2).

Itinerant or mobile sinks are the interface between sensor nodes and data server, roaming around to collect data from the sensor nodes. Since the sink usually has more memory resources and collects data from each node at a time, a pre-distribution scheme with pairwise strategy is appropriate. Thus, each sensor node keeps a single key to communicate with the sink and the sink carries a key for each sensing node.

2) *Routing:* Flat routing is the simplest and most common, and it does not change requirements for key distribution. Thus, the appropriate key distribution approach should consider the deployment characteristics.

A hierarchical network is usually organized in groups, in which common nodes account for sensing tasks and forwarding data to the group leader (or cluster head – CH), and CHs account for additional tasks, such as data aggregation and data forwarding to the base station. There are two variations: CH can be fixed (heterogeneous nodes topology usually) or rotated (homogeneous nodes that need energy balancing) as presented in Figure 1.

In fixed CHs, arbitrated schemes are more appropriate. The trusted point is a preferred target for attacks, but the CH is already a target, since all nodes in the group send data to it. For rotating CHs both arbitrated scheme and pairwise keying are complex to implement, given that the network organization changes regularly. Thus, the most appropriate scheme is a pre-distribution scheme with group-wide keys, taking advantage of the network configuration.

III. CONCLUSION

The use of self-enforcing schemes presents restrictions, since a significant amount of sensor node memory is used. There is no optimal (and practical) solution for any topology or deployment. We advocate that it is possible to choose an appropriate key distribution approach based on the analysis of the topology and characteristics of deployment.

Although symmetric cryptography architectures should take responsibility for the key life cycle, it would be important to analyze the re-keying task and its implications in the key distribution analysis presented.

ACKNOWLEDGMENT

This work was supported by the State of São Paulo Research Foundation (FAPESP) under grants 2010/02909-8 and 2010/16163-8.

REFERENCES

- [1] M. Luk, G. Mezzour, A. Perrig, and V. Gligor, "MiniSec: a secure sensor network communication architecture," in *IPSN '07: Proceedings of the 6th international conference on Information processing in sensor networks*. New York, NY, USA: ACM, 2007, pp. 479–488.
- [2] M. A. Simplicio, P. S. Barreto, C. B. Margi, and T. C. Carvalho, "A survey on key management mechanisms for distributed wireless sensor networks," *Computer Networks*, vol. 54, no. 15, pp. 2591 – 2612, 2010.
- [3] L. B. Oliveira, D. F. Aranha, C. P. Gouv  a, M. Scott, D. F. C  mara, J. L  pez, and R. Dahab, "TinyPBC: Pairings for authenticated identity-based non-interactive key distribution in sensor networks," *Computer Communications*, vol. 34, no. 3, pp. 485 – 493, 2011.
- [4] MEMSIC, "Wireless modules," 2011. [Online]. Available: <http://www.memsic.com/products/wireless-sensor-networks/wireless-modules.html>

Poster Abstract: Wormhole Detection with Location Information in Wireless Ad-Hoc Networks

Jianhua Xiao and Takashi Minohara and Seikoh Nishita

Department of Computer Science, Takushoku University – Japan

E-mail: y0m309@st.takushoku-u.ac.jp, {minohara, snishita}@cs.takushoku-u.ac.jp

Abstract—The wormhole attack is one of the most serious attacks to the wireless ad-hoc networks. It is difficult to detect wormholes, because they are created with regular routing procedure. In this paper we have noticed on the cost of a method of wormhole detection, which use location information to detect inconsistency in the distance between nodes. We have studied on the relation of the detectability of wormhole and the cost of location information for mobile ad-hoc networks.

I. INTRODUCTION

Recently wireless ad-hoc networks become an important technique where it is hard to deploy normal network infrastructure. However the open nature of the wireless communication make them vulnerable to various security attacks[1]. Wormhole attack[2], [3] is one of the most serious attacks among them. By employing preferable link called as 'wormhole', attacker arranges routing path and takes illegal actions such as data corruption and eavesdropping over communications. It is difficult to detect wormhole attacks, because wormholes are created with regular routing procedure. In this paper we have noticed on the cost of wormhole detection, which use location information to detect inconsistency in the distance between nodes. We have studied on the relation of the detectability of wormhole and the cost of location information for networks with mobile nodes. We assume a pair of illegal nodes ('W-node' hereafter) creates a wormhole by tunneling packets between them.

II. WORMHOLE DETECTION IN MOBILE NETWORKS

Our target system comprises mobile nodes('M-node' hereafter) which is capable of determining its location by using the Global Positioning System(GPS). Due to power consumption of GPS device, however, the determination of location is executed at certain time intervals. For detecting wormhole, the nodes append their last determined location for each packets, and the receiver node compares the location with its own location information. The two locations should be within wireless communication range (radius r_c) with taking account of nodes' movement. Otherwise, at least one wormhole exists between two mobile nodes.

Each M-node moves randomly after determining its location, and the accuracy of location declines. We assume the velocity of M-nodes and the time interval of determination of location are bounded, and current location of the M-node is equally distributed inside a circle with center at the the last determined location M_i and radius vt_{max} . As shown in Fig.

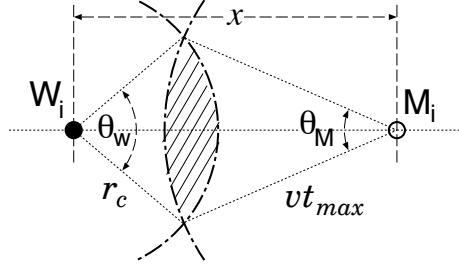


Fig. 1. Intersection of communication area and moving area

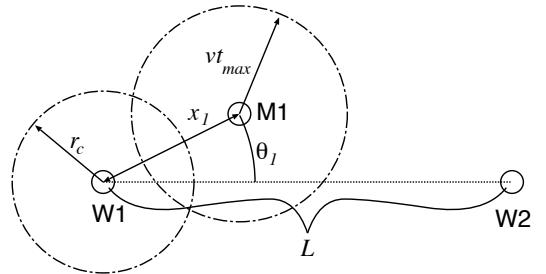


Fig. 2. Relation of W-node and M-node

1, if a W-node captures a packet sent by M-node, there is an intersection of two area: the wireless communication area of the W-node and the moving area of the M-node. When the distance between W-node and M-node is x , the area of intersection is given by

$$Area(x) = \begin{cases} \pi(vt_{max})^2 & (if (vt_{max} < r_c, x < r_c - vt_{max})) \\ \pi r_c^2 & (if (vt_{max} > r_c, x < vt_{max} - r_c)) \\ \frac{1}{2}(vt_{max})^2 \{\theta_M - \sin \theta_M\} + \frac{1}{2}r_c^2 \{\theta_W - \sin \theta_W\} & (otherwise) \end{cases} \quad (1)$$

Here,

$$\theta_M = 2 \cos^{-1} \left(\frac{(vt_{max})^2 + x^2 - r_c^2}{2vt_{max} \cdot x} \right) \quad (2)$$

$$\theta_W = 2 \cos^{-1} \left(\frac{r_c^2 + x^2 - (vt_{max})^2}{2r_c \cdot x} \right) \quad (3)$$

By using a polar coordinate relative to the W-node (Fig. 2), we consider the situation that an M-node determines its location at (x, θ) , and it sends a packet to its correspondent after moving into the communication area of the W-node. The

location of the M-node (x, θ) follows random variable, and its probability density function $f(x, \theta)$ is given by

$$f(x, \theta) = \frac{\text{Area}(x)}{\int_0^{r_c + vt_{\max}} 2\pi \text{Area}(x) dx} \quad (4)$$

If the communication between two M-nodes succeeds through a wormhole, the similar relation observed at the another end of the wormhole. A distance between determined locations of two M-nodes can be calculated with their coordinate $(x_1, \theta_1), (x_2, \theta_2)$ and a distance L between both end of W-nodes.

$$l((x_1, \theta_1), (x_2, \theta_2), L) = \sqrt{(L - x_1 \cos \theta_1 - x_2 \cos \theta_2)^2 + (x_1 \sin \theta_1 - x_2 \sin \theta_2)^2}$$

After the determination of their location, M-nodes may approach each other, and the distance diminishes by $2vt_{\max}$ at the maximum. Thus the wormhole is detected if there still remains a gap of more than the communication range r_c between expected spaces of M-nodes' movement (Fig.3). We use the following function to denote detectability of the wormhole.

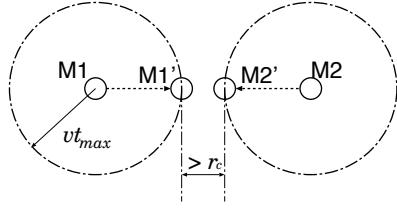


Fig. 3. Gap between expected spaces of M-nodes

$$C_d((x_1, \theta_1), (x_2, \theta_2), L) = \begin{cases} 1 & \text{if } l((x_1, \theta_1), (x_2, \theta_2), L) - 2vt_{\max} > r_c \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

Consequently, assume that M-nodes M_1 and M_2 determined their location (that is represented $(x_1, \theta_1), (x_2, \theta_2)$ from W-nodes W_1, W_2 respectively), and communicated through the wormhole after moving, the probability of wormhole detection is given as following equation.

$$P(L) = \int_0^R \int_0^{2\pi} \int_0^R \int_0^{2\pi} p((x_1, \theta_1), (x_2, \theta_2), L) dx_1 d\theta_1 dx_2 d\theta_2 \quad (6)$$

Here,

$$\begin{aligned} p((x_1, \theta_1), (x_2, \theta_2), L) &= C_d((x_1, \theta_1), (x_2, \theta_2), L) \cdot f(x_1, \theta_1) \cdot f(x_2, \theta_2) \\ R &= r_c + vt_{\max} \end{aligned}$$

Fig. 4 shows the probability of detecting a wormhole, and Fig. 5 shows the expected number of packets to the time of wormhole detection. From these results, the optimal time interval for determining location can be obtained against the required probability or latency of detection.

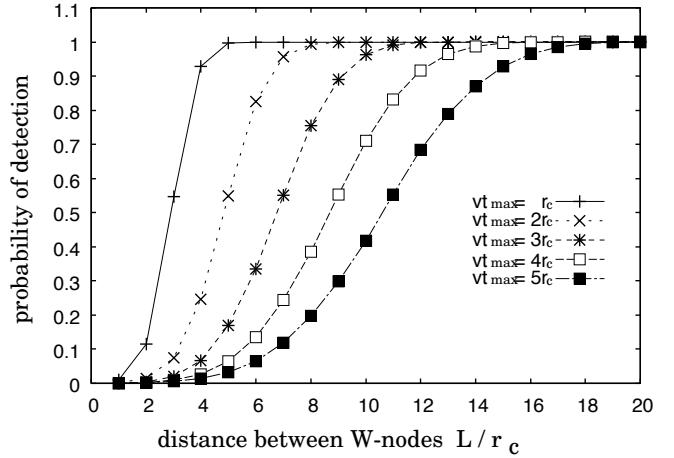


Fig. 4. Probability of detection

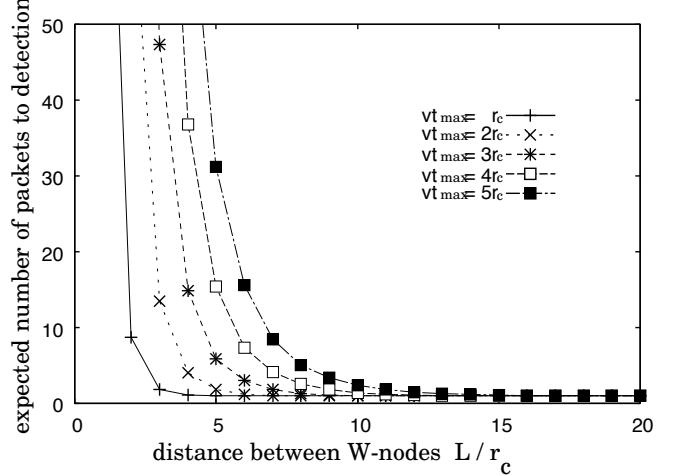


Fig. 5. Expected number of packets to detection

III. CONCLUSION

In this paper, we have studied on the relation of the detectability of wormhole and the cost of location information for mobile ad-hoc networks. We have obtained the equations to estimate optimal time interval for determining location on mobile devices.

ACKNOWLEDGEMENT

A part of this work was supported by JSPS Grant-in-Aid for Scientific Research (No.21560413).

REFERENCES

- [1] L. Buttyán and J. P. Hubaux, "Report on a working session on security in wireless adhoc networks," *ACM SIGMOBILE Mobile Computing and Communication Review*, vol. 7, no. 1, pp. 74–94, Jan. 2003.
- [2] I. Khalil, S. Bagchi, and N. Shroff, "Liteworm: a lightweight countermeasure for the wormhole attack in multihop wireless networks," in *International Conference on Dependable Systems and Networks(DSN2005)*, 2005, pp. 612–621.
- [3] Y. C. Hu, A. Perrig, and D. B. Johnson, "Packet leashes: A defense against wormhole attacks in wireless ad hoc networks," in *IEEE INFOCOM*, Mar. 2003, pp. 1976–1986.

Hardware and Energy

Demo Abstract: Evaluating Energy-Efficiency of Hardware-based Security Mechanisms

Christian Haas, Anton Hergenröder, Joachim Wilke, Thomas Wiskot, Markus Niedermann

Institute of Telematics, Karlsruhe Institute of Technology (KIT)

Karlsruhe, Germany

firstname.lastname@kit.edu

Abstract—We demonstrate an evaluation setup for the energy-efficient usage of hardware-based security mechanisms. We implemented a small use-case scenario for the surveillance of critical areas with both software and hardware-based security mechanisms. For the surveillance, we use passive infrared sensors (PIR) attached to IRIS sensor nodes. The detected trespassers are reported to a base station and shown in a graphical user interface. We use asymmetric cryptographic mechanisms to guarantee the integrity and authenticity of the reported trespassing events. In the demonstrator, we will show how much energy can be saved by using hardware-based security mechanisms compared to the software-based approach. For the energy measurements, we employ hardware from the SANDbed testbed at the Karlsruhe Institute of Technology (KIT).

I. MOTIVATION

Wireless sensor networks (WSNs) have been proposed for a vast number of scenarios and applications. One of the main scenarios is the surveillance of critical infrastructures or critical areas like borders or industrial complexes. In these scenarios, the need for secure communication and robust operation is widely accepted. Over the past years, many protocols and mechanisms to that end have been proposed, but they have rarely been implemented or evaluated in real-world applications or environments. One of the key restrictions of WSNs are the resource constraints of the used sensor nodes, e.g. the sensor nodes only possess a very small energy budget and have limited processing power. Up to now, most research has been done on efficiently implementing well known security protocols or mechanisms on the sensor nodes. More recent work has proposed the usage of hardware-based security mechanisms like a trusted platform module (TPM) or similar hardware [1]. One key motivation for using hardware-based security modules is limiting the energy consumption for the cryptographic operations. Unfortunately, reliable studies about the resulting energy consumption are missing. One reason is that the measurement of energy usage data in a sensor network is very challenging. For realistic evaluations, one has to use special purpose measurement hardware like *Sensor Node Management Devices* (SNMDs) [3]. These devices are capable of performing measurements on real nodes running a real, unmodified sensor network application. In this work, we will demonstrate the feasibility of hardware-based security modules in a real-world monitoring scenarios as well as a generic evaluation approach for the energy-efficiency of WSN (security) protocols.

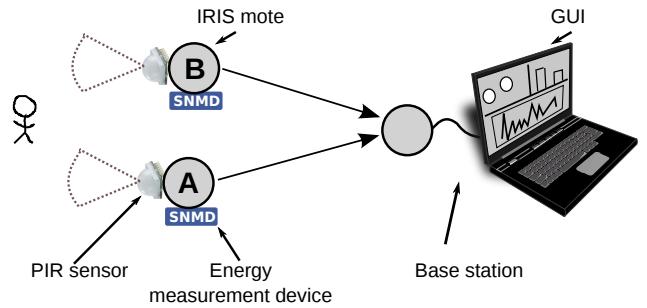


Fig. 1. Demonstrator scenario

Consequently, the aim of this work is twofold:

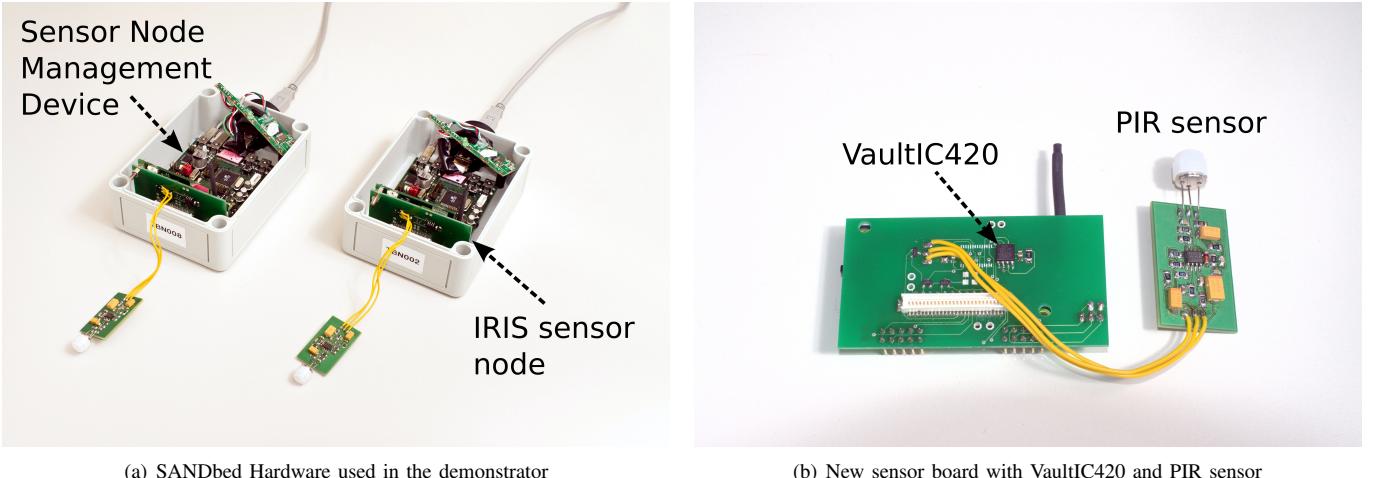
- 1) We present a generic evaluation setup for evaluating the energy efficiency of WSN protocols and algorithms. Therefore, we use SANDbed as evaluation platform and demonstrate how to use the platform in a concrete evaluation scenario.
- 2) Furthermore, we perform an evaluation of the energy-efficiency of hardware-based security mechanisms in a real use-case scenario. As an example, we implemented a sample application as well as software- and hardware-based security mechanisms.

II. DEMONSTRATOR

Our demonstration scenario is shown in Figure 1. For demonstration purposes, we implemented a small monitoring system for critical areas that is able to detect any movement in the critical area. The demonstrator is composed of two IRIS sensor nodes and one base station which is used to display detected movements on a graphical user interface.

Our application is written in TinyOS and contains the following protocols:

- *Trespasser Detection Protocol*: A PIR event is triggered as soon as a trespasser walks within the range of the PIR sensor. The PIR event will be transmitted to the base station as well as the timestamp and the node ID of the detecting node.
- *Node Failure Detection Protocol*: Every sensor node is periodically sending a heartbeat message to the base station. This enables the base station to detect node failures due to missing heartbeats.



(a) SANDbed Hardware used in the demonstrator

(b) New sensor board with VaultIC420 and PIR sensor

- **MAC Protocol:** We use TinyOS Low-Power-Listening as energy-efficient MAC-Layer protocol.

To guarantee the integrity and authenticity of the transmitted data, we use digital signatures based on elliptic curves.

A. Cryptographic Algorithms

In this demonstrator, we use ECDSA signatures to guarantee the integrity and authenticity of the reported PIR events. One of the nodes is using a software-based implementation, the other node is equipped with a hardware-based security module. On the software side, we integrated TinyECC [2] for the computation of the ECDSA signatures. As hardware-based security module, we apply a VaultIC420 security module from inside secure [4]. The hardware module is capable of performing different cryptographic services (e.g. Digital Signatures or Message Digest) and different cryptographic algorithms (e.g. DSA, ECDSA, SHA2, HMAC). The hardware module is integrated on a special purpose extension board and connected to the IRIS sensor node through the Two Wire Interface (I^2C Bus). The extension board we use in the demonstration is shown in Figure 2(b). Besides the VaultIC420, we integrated the PIR sensor on the same board for an easy integration into the sensor node platform.

B. Energy Evaluation

Both sensing IRIS nodes, *A* and *B*, are equipped with special energy measurement devices (SNMDs), which can be seen in 2(a). These SNMDs provide high resolute measurements of voltage and current draw of the attached sensor node and are part of our testbed SANDbed [3]. Energy measurement data is collected side-effect free using USB to avoid any interference with normal network operation. In the demonstration, we sample the current draw and the voltage of the attached sensor nodes with 10 kHz and display this energy usage data in a live measurement.

C. Evaluating the Energy-Efficiency of Hardware-based Security Mechanisms

We show the feasibility of hardware-based security modules in a real-world monitoring scenario. Therefore, we compare the energy usage data of the hardware-based security module with the software implementation in a live demo. For that purpose, we analyze the live measurements of the SNMDs to show how much energy has to be used to compute the digital signatures with either hardware or software. As we are using TinyOS Low-Power-Listening as MAC protocol, we also show the energy usage data while transmitting the PIR events and the heartbeats.

III. SUMMARY

Up to our knowledge, we are the first to evaluate the energy-efficiency of hardware based security mechanisms in a real-world scenario. We show how these hardware-based security mechanisms can save energy, especially when using asymmetric cryptographic operations. Additionally, we present a generic approach for evaluating the energy-efficiency of WSN protocols and algorithms. Our approach can easily be extended to other real-world experiments and evaluations.

REFERENCES

- [1] Hu, Wen and Corke, Peter and Shih, Wen Chan and Overs, Leslie, *secFleck: A Public Key Technology Platform for Wireless Sensor Networks*, Proceedings of the 6th European Conference on Wireless Sensor Networks, EWSN '09.
- [2] An Liu and Peng Ning, *TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks*, in Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN 2008), SPOTS Track, pages 245–256, April 2008.
- [3] A. Hergenröder and J. Wilke and D. Meier, *Distributed Energy Measurements in WSN Testbeds with a Sensor Node Management Device (SNMD)*, Workshop Proceedings of the 23th International Conference on Architecture of Computing Systems, pages 341–438, 2010.
- [4] ATVaultIC420 security module, <http://www.insidesecure.com/>.

Poster Abstract: Energy Assessment in Praxis

Christian Renner, Florian Meier, Volker Turau

Institute of Telematics

Hamburg University of Technology, Hamburg, Germany

Email: {christian.renner,turau}@tu-harburg.de

Abstract—Combining energy harvesting with energy-aware scheduling enables perpetually operating sensor networks. Practical realization yet requires precise holistic online energy assessment. The building blocks are available, but the analysis of their interaction has been neglected. To close the gap, we evaluate the joint performance of energy assessment components. Our experiments substantiate that holistic energy assessment is feasible and that small configuration errors are tolerable.

I. MOTIVATION

Energy harvesting promises unlimited and uninterrupted sensor node operation [1]. Non-intrusive monitoring yet demands devices of tiny size; thus, the amount of harvested energy is decreased while it must still satisfy the average power consumed by sensor node applications and algorithms. The actual extent of neither harvested nor consumed energy is known in advance. Consumption depends on hard to foresee network load caused by topology and routing changes. Energy harvest often depends on the exact positioning of the harvester, since local and seasonal effects dominate energy production.

Thus, a sensor node must adapt its consumption to the available energy resources [2]. Achieving uniform operation even in times of low harvest requires buffering energy in abundant periods. Supercaps are frequently used: they combine small size and low price and allow for reliable while facile energy reserve estimation [3]. To adapt its duty cycle or find a feasible schedule, a node must track its consumption [4], [5]. Solar cells are used as harvesters on many platforms. They deliver sufficient energy with a diurnal pattern, enabling energy intake forecasts [6] for depletion-safe and smooth operation with infrequently adapted duty cycles [2].

Tools for holistic online energy assessment are ready, but research has mainly focused on optimizing individual components. Yet, studying their interaction has been neglected despite the practical importance: Only if joining these techniques reflects the energy flow of an energy-harvesting sensor node, it is possible to forecast the future energetic state—i.e., the course of energy reserves. This in turn enables reliable and depletion-safe duty cycle adaptation and task scheduling.

To close this gap, we evaluate the accuracy of holistic online energy assessment for an energy-harvesting prototype in a five-node field test running over three weeks. We compare the recorded real energy flow with a simplified model derived from the harvester circuit. The evaluation verifies the model and proves the utility of existing energy-assessment techniques in praxis. We identify the main influencing factors of energy misjudging and those with low impact on system preciseness.

II. HARVESTER PROTOTYPE AND SYSTEM MODEL

We built a customized energy-harvesting power supply for the Iris node. A solar cell with a maximum current I_h of 35 mA serves as energy source. We use a direct charging circuit and integrated a sensor for measuring I_h . The harvester supports supercaps with a maximum voltage of $V_{max} = 2.7$ V. A capacity of 25 to 100 F offers small size, cheap prize and energy reserves of a few days for a 1% radio duty cycle [3]. The supercap's voltage V_c is accessible via an ADC port, and an overcharging protection disconnects the solar cell, if V_c reaches V_{max} . A TI TPS 61220 supplies the node with a constant voltage of $V_n = 2.7$ V. It has a cut-off voltage of $V_{cut} = 0.5$ V, and we measured an efficiency η of 75–95%.

The simplified circuit in Fig. 1 yields the energy flow model

$$I_c = I_h - I_r \quad \text{with} \quad I_r = (I_n \cdot V_n) / (\eta \cdot V_c). \quad (1)$$

Based on the findings in [3], we model the relationship of supercap voltage V_c and current I_c as an ideal capacitor, i.e., $I_c = C \cdot \dot{V}_c$. The resulting system equation is

$$C \cdot \dot{V}_c = I_h - (I_n \cdot V_n) / (\eta \cdot V_c). \quad (2)$$

Solving (2) for V_c enables reliable duty-cycle adaptation and task scheduling: Given a forecast of I_h , it is possible to find the maximum average current I_n , such that the supercap is not depleted within the prediction horizon. This method requires model accuracy and precise energy assessment.

III. EVALUATION

We implemented an energy-aware software layer for TinyOS. Besides reading V_c and I_h , energy consumption is measured online by tracking the time spent in each state of the hardware components (e.g., processor and radio) with a resolution of micro-seconds. The tracker uses a generic consumption profile comprising the average values of I_n per hardware state; costly per-node configuration is thus not required. Average efficiency $\eta = 85\%$ is used to calculate I_r as in (1); a constant η keeps computation overhead low.

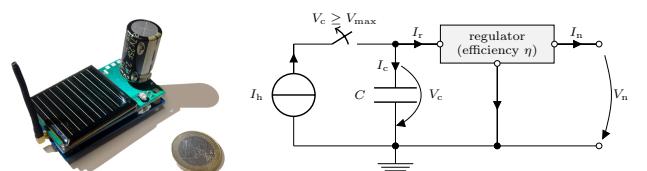


Fig. 1: Energy harvester hardware and simplified circuit

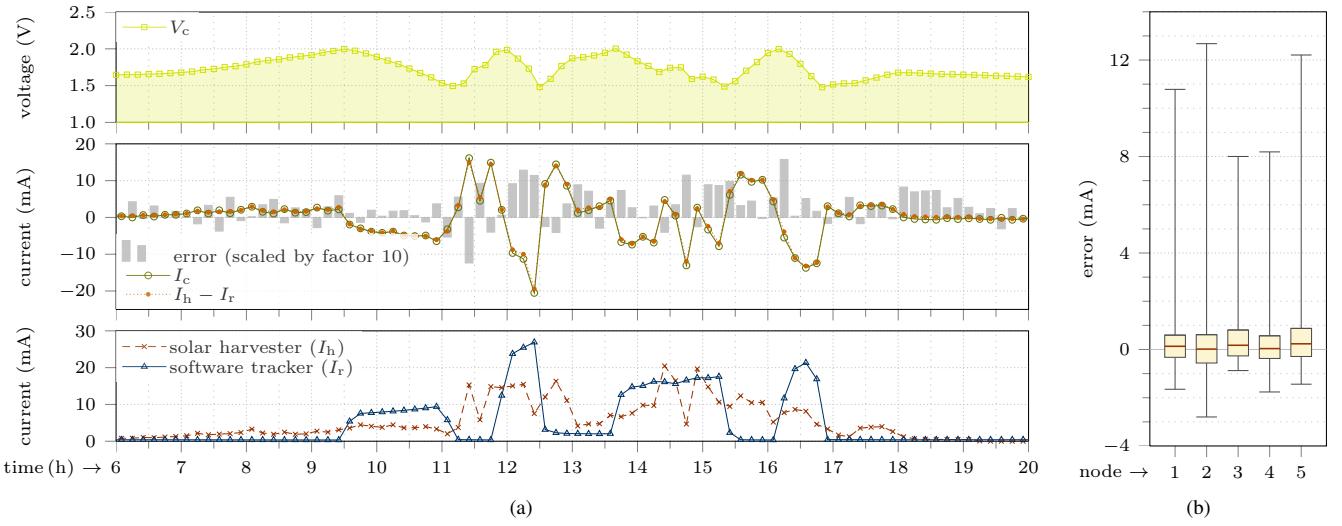


Fig. 2: (a) Compliance study of system model and online energy assessment, (b) error analysis (10%, 50%, and 90% quantiles)

We set up a three-week deployment of five nodes equipped with our harvester and a 50 F supercap outside on a window sill. Sensor samples and consumption are reported every 30 s to a sink. The radio duty-cycle is increased and decreased when V_c overshoots 2 V or underruns 1.5 V, respectively. Our analysis is based on 10 min snapshots and averages, resulting in more than 3 000 samples per node.

Figure 2a shows the energetic trace of node 1 on day 8 of the field test. The system model produces tolerable errors in all situations of the day, with three exceptions. Firstly, the error is high for large absolute values of I_c , if I_r is large at the same time. This is caused by configuration and calibration errors of η and C . Both factors influence I_r and I_h linearly. Secondly, the error is large, when V_c cuts across 2 V or falls below 1.5 V, the thresholds for consumption adaptation. These errors show the inherent, yet intentional error of the model: linearization and averaging. The current I_r and V_c are reciprocal: The higher V_c , the lower I_r , and vice versa. If a node switches, e.g., from low consumption to high, enduring consumption while $I_h \gg 0$, the linearized model using averages produces large errors. If the consumption changes moderately, e.g., at around 13:40, the error is considerably smaller. Thirdly, the errors before 7 h and after 18 h stem too large readings of I_h , which we verified by covering the solar cells for a few minutes.

The trace of node 1 substantiates the applicability of the model. These results are supported by Fig. 2b: 80% of all errors are smaller than 1 mA with a median between 0 and 0.3 mA, which is roughly the experienced error of the solar current sensor. The large errors in the plot only occur upon abrupt changes of consumption, when I_h is large at the same time. The practical importance of these errors is small in many cases. In contrast to our field test, radio duty cycling will rarely produce such consumption changes. We also found that supercap leakage and temperature dependency are low. While modeling their influence is generally possible, the expected model improvement is low.

IV. CONCLUSION

We presented a tiny, low-cost energy harvester with a solar cell and a supercapacitor. We derived an energy-flow model for this harvester that can be used for predicting a node's future energetic state and is simple enough to be executed on sensor nodes. We implemented a holistic energy-assessment layer for TinyOS and ran a five-node field test for over three weeks to collect real-world energy traces. Our evaluation verifies the presented energy model and supports practical collaboration of energy-assessment techniques. Precise measurements of the energy-intake and smooth consumption profiles are essential. Unevenly distributed consumption results in large errors. The system model can be used to predict a node's future energetic state, when it is paired with energy-intake forecasts. It is hence possible to derive duty cycles or task schedules ensuring effective while perpetual node operation.

V. ACKNOWLEDGMENTS

This research has been partially funded by the German Research Foundation under contract number TU 221/4-1. The authors would like to thank Janos Sallai for his valuable input.

REFERENCES

- [1] X. Jiang, J. Polastre, and D. Culler, "Perpetual Environmentally Powered Sensor Networks," in *IPSN*, 2005.
- [2] C. Moser, L. Thiele, D. Brunelli, and L. Benini, "Adaptive Power Management for Environmentally Powered Systems," *Trans. Computers*, 2010.
- [3] C. Renner, J. Jessen, and V. Turau, "Lifetime Prediction for Supercapacitor-powered Wireless Sensor Nodes," in *FGSN*, 2009.
- [4] P. Dutta, M. Feldmeier, J. Paradiso, and D. Culler, "Energy Metering for Free: Augmenting Switching Regulators for Real-Time Monitoring," in *IPSN*, 2008.
- [5] P. Hurni, T. Braun, B. Nyffenegger, and A. Hergenröder, "On The Accuracy of Software-based Energy Estimation Techniques," in *EWSN*, 2011.
- [6] A. Kansal, J. Hsu, S. Zahedi, and M. Srivastava, "Powermanagement in Energy Harvesting Sensor Networks," *Trans. on Embedded Computing Sys.*, 2007.

Poster Abstract: Energy-Harvesting Wireless Sensor Networks

Xenofon Fafoutis*, Dusan Vuckovic†, Alessio Di Mauro*, Nicola Dragoni*, and Jan Madsen*

*DTU Informatics, Technical University of Denmark, Denmark, {xefa,adma,ndra,jan}@imm.dtu.dk

†DELTA, Denmark, duv@delta.dk

Abstract—Energy Harvesting comprises a promising solution to one of the key problems faced by battery-powered Wireless Sensor Networks, namely the limited nature of the energy supply (finite battery capacity). By harvesting energy from the surrounding environment, the sensors can have a continuous lifetime without any needs for battery recharge or replacement. However, energy harvesting introduces a change to the fundamental principles based on which WSNs are designed and realized. In this poster we sketch some of the key research challenges as well as our ongoing work in designing and realizing Wireless Sensor Networks with energy harvesting capability.

I. INTRODUCTION

Wireless Sensor Networks (WSNs) are systems of multiple small and inexpensive embedded devices that can sense, measure and gather information from the environment they are deployed. Examples of application range from environmental monitoring and control to healthcare monitoring and traffic control, to name only a few. WSNs are typically required to run for long periods of time, often several years, only powered by batteries. This makes *energy-awareness* a particular important issue when designing WSNs. Indeed, one of the major limitation of battery-powered nodes is finite battery capacity [1], which means nodes operate for a finite duration, only as long as the battery lasts. Finite node lifetime implies finite lifetime of the WSN applications or additional cost and complexity to regularly change batteries. Indeed, batteries cannot easily be replaced, since typically WSNs consist of hundreds to thousands of sensors and may be deployed in unreachable places, such as mountains or underground. To make matters worse, depleted batteries constitute environmental problems.

Energy Harvesting Wireless Sensor Networks (EH-WSNs) can provide a solution to the energy problem by harvesting energy that already exists in the surrounding environment. Energy harvesting refers to harnessing energy from the environment and converting it to electrical energy. If the harvested energy source is large and continuously (or periodically) available, a sensor node can be powered perpetually. In this way, energy is essentially infinite; however, not always available.

Energy harvesting introduces a change to the fundamental principles based on which protocols for WSNs are designed. Instead of focusing on energy efficient protocols that aim to maximize sensors lifetime, the main design objective in EH-WSNs is to maximize the performance of the network given the rate of energy that is available to be harvested from the environment. In other words, the surplus of harvested energy can be used to improve the performance of the network.

Another important element that differentiates EH-WSNs from classical WSNs is that in EH-WSNs some sensor nodes can be more capable than others. This is due to the non-uniformly distribution of ambient energy. As an example, consider a solar-based EH-WSN where some nodes are covered by shadows while others are under direct sun light. In such environment, the more capable nodes can be used for performing the energy consuming tasks, on behalf of the incapable nodes that need to sleep and recharge.

Example: WSNs Powered by Vibration. *Vibration energy harvesting is the process by which otherwise wasted vibration (such as from a piece of industrial machinery) is harvested and converted to useful electrical energy to perpetually power wireless sensor nodes. One of the most prevailing applications for WSNs is to monitor the health and status of essential industrial machinery assets within light and heavy industrial manufacturing environments. As an example of health monitoring application, a WSN can be used for monitoring vibrations of a rotating machine and by analyzing the frequency domain of vibrations it is possible to determine when the machine is going to fail and schedule the maintenance in time, before malfunction of the machine. The fact that the WSN is being powered by the same vibration its measuring allows the system to continuously monitor the operation of the machine without the risk of the node running out of power.*

In this poster we sketch some of the key research challenges as well as our ongoing work in designing and realizing Wireless Sensor Networks with energy harvesting capability.

II. CHALLENGE: ENERGY HARVESTING

Potential sources of energy harvesting are all around us. Light, radio signals propagating through the air, wind, different kinds of vibration and movement, heat flow, just to mention only a few. With new materials and new ways of transforming energy, efficiency of harvesting has risen to the level that can be used for powering WSNs. Nevertheless, harvesting sources are not powerful enough yet to allow continuous operation of sensor nodes powered by energy harvesting. Therefore careful planning is needed in every stage of design.

First stage of design is finding a way of converting ambient energy to electrical energy. Our group will focus on harvesting light and vibration energy through use of small scale solar cells for light harvesting and macro fiber composites for capturing

vibration energy. The main challenge at this stage is efficient harvesting of available energy and storing the produced charge in a way so it could be used when needed. This task is not trivial due to the fact that, in order to harvest the most energy from the source, the load on the circuit producing energy needs to be matched through appropriate circuitry and the load is always application specific making it even harder to implement.

Second stage is to manage the energy stored in the most efficient way. This block is in charge of selecting when and how the energy is distributed throughout the system being powered by it. Furthermore this block needs to minimize the leakage from the storage and have the smallest possible quiescent current.

Third stage is the application that is run on the node. For battery powered systems the main challenge lies in prolonging of the battery life, but with energy harvesting the designer is aware that the energy storage will be replenished, therefore other approaches can be utilized, such as using the power when harvesting and keeping dormant when no energy is harvested. Changing the duty cycle of operation depending on energy levels is also one of the options.

The fact that energy harvesting is application specific and all parts in the design are much interconnected makes this kind of systems a big challenge to design in a robust way guaranteeing a very long, maintenance free operation.

III. CHALLENGE: NETWORKING

Communication has always been a major issue in WSNs due to the amount of energy it requires and the numerous performance benchmarks that depend on it, such as the delay and the delivery rate of measurements. Energy harvesting provides the wireless sensor nodes with an energy supply that varies over time and space. Communication protocols supporting the sensor network need to be adaptable; so that whenever there is an excess of energy, it is used to improve the performance of the system and whenever there is a shortage of energy, the sensors downgrade their performance, and thus their energy consumption, to an operation state that is sustainable by the available energy. In other words, the goal of the system is to operate at the maximum sustainable performance. Furthermore, communication protocols need to have certain additional qualities. First, they need to provide flexibility so that the network load is distributed to the nodes that have access to more environmental energy at any given time. Secondly, they need to support applications that have different performance requirements. For example, there are alarming applications that require low delays and monitoring applications that require a consistent data set. Last but not least, cross-layer optimization is important so that the energy is consumed in the most efficient way.

All these qualities constitute important considerations for designing communication protocols for EH-WSNs. In particular, MAC protocols need to efficiently support individual duty cycles to allow each node to effectively adapt its energy consumption to sustainable levels. Hence, the “traditional”

approach where nodes have coordinated and synchronized duty cycles is insufficient. Individual duty cycles can also support distributed autonomous load balancing. If each node is able to freely choose its own duty cycle based on its energy capabilities, the node with more access to energy will be awake more frequently and eventually perform more energy consuming tasks. Moreover, routing protocols need to select the path that best fits the energy conditions of the network as well as the requirements of the application. For instance, the paths that minimize the end-to-end delay might be the best candidates for a delay-sensitive application. Lastly, opportunistic routing schemes can be used to decrease the sleeping delay caused by the duty cycles. Instead of waiting for a node to wake up, a node may transmit to the node that wakes up first out of a set of nodes that meet certain criteria.

IV. CHALLENGE: SECURITY

Security is an extremely important issue for sensor networks, due to the various kind of data, in many cases sensitive, gathered by the network. The introduction of energy scavenging capabilities completely redefines how security in WSNs can be approached. For instance, energy independent attacks (e.g. node replication) become much more effective since the average lifespan of a node is greatly increased. On the other hand, power related attacks (e.g. energy depletion) can be neutralized by the fact that a node can reacquire energy over time. Furthermore, the potential of having more energy available could make possible to use better security tools such as sturdier encryption algorithms, longer keys and computationally heavier hash functions, just to name a few. Another class of approaches are the ones similar to what proposed in [2], where the amount of “strength” used in the active security scheme is function of the available harvested energy.

For these reasons, we aim to systematically analyze and classify attacks in EH-WSNs, so that distinctive characteristics can be pointed out and possible new attacks identified. As a first result, a taxonomy of attacks for EH-WSNs has been determined, with focus on how scavenging capabilities affect the network and which new and specific attacks can be depicted. This classification represents the basis for specifically tailored security solutions.

V. CONCLUSION

Energy harvesting is introducing some changes to the fundamental principles based on which Wireless Sensor Networks are designed and realized. In this poster we have sketched some of the key research challenges as well as our ongoing work in designing and realizing Wireless Sensor Networks with energy harvesting capability.

REFERENCES

- [1] S. Sudevalayam and P. Kulkarni, “Energy harvesting sensor nodes: Survey and implications,” *Communications Surveys Tutorials, IEEE*, vol. 13, no. 3, pp. 443 –461, quarter 2011.
- [2] A. V. Taddeo, M. Mura, and A. Ferrante, “Qos and security in energy-harvesting wireless sensor networks,” in *SECRYPT*, S. K. Katsikas and P. Samarati, Eds. SciTePress, 2010, pp. 241–250.

Poster Abstract: Endless Smart Power for WSNs: Combining Multiple Harvesting and Fuel Cells

Michele Magno*, Danilo Porcarelli*, Davide Brunelli^o, Member IEEE, Luca Benini* Senior Member IEEE

* Dipartimento elettronica informatica sistemistica (DEIS) - University of Bologna

^o Dipartimento Ingegnerie e Scienze dell'Informazione (DISI) - University of Trento

Abstract— During recent years, there has been a growing interest on wireless sensor networks (WSNs) and on the opportunities opened by this technology. Since the energy consumption is a bottleneck in WSNs, extending the lifetime has a significant impact on the applicability of this technology. We present in this work the design, implementation and characterisation of a novel power unit including intelligence, ambient available energy harvesting (EH) from multiple sources, storage, electrochemical fuel cell integration, and recharging capability, which acts as the power layer for the node the node. The power unit is focused to allow the node to operate perpetually in outdoor scenario. We prototyped and tested the nodes, and used their characterization to demonstrate through measures the high efficiency.

I. INTRODUCTION

The development of wireless sensor networks (WSN) has highlighted a wider range of applications in recent years due to the flexible distribution of WSN sensor nodes. The clear advantages of using WSNs that include greater safety, and reduced maintenance cost have increased the demand for even more pervasive and sophisticated monitoring tools. However, sensor nodes are typically powered by batteries with limited capacity. Since wireless sensor nodes, are mostly used in outdoor settings, energy can be harvested from environmental sources such as sunlight, wind, vibration, water flow, etc.. Harvesting this energy efficiently is important to reduce the form factor and reduce the cost. For this reason maximum power point tracking (MPPT)[1] and power management is a key feature of the power unit. Finally, the possibility to take energy from very high density storage such as Fuel Cells (FC) [3]may be required. FC is an electrochemical device that uses fuel (i.e Hydrogen) to generate electrical power. This is a novel option for recharging the other storage units when they are empty and environmental energy is not available.

Therefore, flexible, energy efficient multi-harvesters and power management techniques are the key requirement to increase the lifetime of the node and hence the network to more than a year time, and ultimately, to achieve the energy sustainability. In addition, many other system design issues require the consideration of small form factor, output voltage level monitoring, capacity of the energy storage, cost efficiency and ‘plug and play’ functionality.

In this work we present the design and implementation of a novel multi-harvester power unit that addresses the above mentioned features. The power unit has been designed to be energy efficient and flexible by using different kind of energy sources and power management policies. The power unit is designed as a *smart* battery and provides continuous power to the nodes and sensors storing the converted energy into Li-Ion

batteries or supercapacitors.. Moreover when the power converted from the surrounding environment sources is too low, the power unit will use special micro hydrogen fuel-cells (FC), as last-option energy source to recharge the storage elements previously mentioned.

II. RELATED WORK

The literature related to power supply for wireless sensor networks concerns the use of EH and various storage devices such as batteries (commonly Li-ion, rechargeable) and supercapacitors [2]. Moreover in [3], the authors describe a fuel cell and battery hybrid (FC-Bh) system for use in portable microelectronic systems, characterising and analysing the performance of the system.

Considering the ambient power sources, the most commonly used are photovoltaic, wind turbine or mechanical energy harvesting from vibrations or strain [3][4][5][6]. The Ambimax system developed at the University of California [4] is a viable alternative, combining energy harvesting from wind and solar sources, again using batteries and supercapacitors for storage. This system has the added value of being able to perform maximum power point tracking (MPPT).The system in [5] describes a reconfigurable energy subsystem for WSN, inclusive of solar and vibrational energy scavenging with Li-ion rechargeable batteries and super-capacitor for storage. In [6], Kheng and Panda present a hybrid device with indoor light and thermal harvesting.

It is noticeable that very few projects have incorporated multiple energy resources in a single power unit, or platform and no of them include FC and energy harvesters.

III. SMART POWER UNIT

Fig.1 illustrates the architecture of the prototyped Smart Power Unit (SPU), capable of hosting various environmental sources and storages (Li-Ion batteries, supercapacitors and FC). The main components in terms of hardware of the SPU are

- Energy manager: It the on board MCU which performs the power managements algorithms, monitors the resources, takes decisions about power policies, exchanges information with the supplied devices.
- Harvesters modules: the transducers and the electronics converters to collect energy from the environmental sources (Wind and Solar in this prototype however other sources are available)
- Storages: Reservoirs of energy to store the collected energy (Supercapacitors and Li-Ion battery)
- Fuel cell: The hardware to convert the energy from fuel cell to recharge the energy reservoirs.

- Output: the electronics circuits to supply the device with the 3.3V voltage. Moreover a I²C and GPIO pins are available to exchange information with the node

The power unit is described as “smart” because it has been designed to provide advanced features, and the possibility to control and optimize operating parameters in the field. In

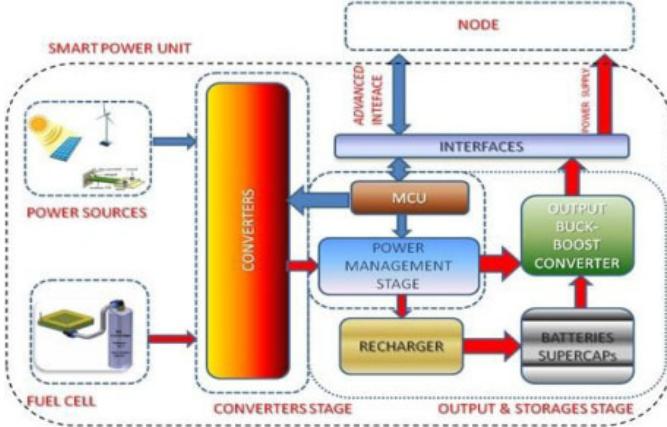


Fig. 1. SMP architecture

particular it is possible to monitor the current state of the harvesters, batteries and micro fuel-cells. Furthermore, it is possible to change the operating frequency used by internal DC/DC converters and chargers and perform MPPT algorithm to improve the efficiency of the harvesters. The power unit uses the TI MSP430F2274 microcontroller, a 16-bit ultra-low power microcontroller, with 32KB flash, 1KB RAM, 10-bit ADC, 2 op-amp and 2 Universal serial communicator interfaces. This device is chosen due to its ultra-low power consumption, it has the necessary ADC and peripherals for the microcontroller. It can select the adequate power resources to guarantee the best power efficiency and can interact with the supplied devices to improve the power management on both sides (power unit and supplied platform). The MCU executes programmable power management policies, and provides the required flexibility and energy awareness, considering the node may be deployed in various locations with varying environmental power availability.

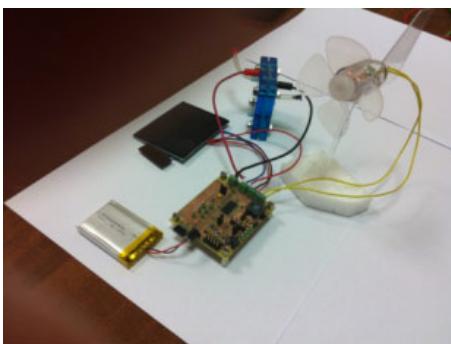


Fig. 2. SMP prototype with solar and wind harvesters and FC.

IV. EXPERIMENTAL RESULTS

In order to evaluate the performance of our approach the prototyped platform has been evaluated experimentally with

respect to operational power consumption and the energy efficiency. TABLE I. details the power provided from different sources. The energy lost through converter circuits is considered, and included in the efficiency of single harvesting. The efficiency is evaluated by measuring the power ($V \cdot I$) of source at the input of the power unit and the power transferred to the energy storage. Therefore, the efficiency accounts for both energy lost from converter stages and lost from SMP MCU.

TABLE I. ENERGY SOURCES

Sources	Performance	
	Max Power	Efficiency
Solar (Li-Ion Battery)	450mW	82%
Solar (SuperCap)	450mW	75%
Wind (SuperCap)	10mW	85%
FC (Li-Ion Battery)	1W (recharging limit @ 200mA)	80%

PERFORMANCE OF HARVESTERS AND FUEL CELL. THE EFFICIENCY WAS EVALUATED MEASURED THE POWER ($V \cdot I$) OF SOURCE ON THE INPUT OF POWER UNIT AND THE POWER TRANSFERRED TO THE ENERGY STORAGE. THUS THE EFFICIENCY COUNTS BOTH ENERGY LOST FROM CONVERTER STAGES AND FROM SMP MCU.

V. CONCLUSION

In this paper we presented the design and implementation of a smart and versatile power unit that is able to harvest energy from different sources and fuel cells. The smart power unit stores energy in both super capacitors and Li-Ion battery. The power unit considers the flexibility of multi harvesting and the MCU on board the power unit has ultra low power radio wake up capabilities. These features allow the board to perform MPPT to improve the efficiency of the energy harvesting process.

VI. ACKNOWLEDGMENT

The research leading to these results has been supported by the GENESI Project (Grant agreement n. 257916) funded by the EU 7th Framework Programme. In addition the work has been supported by the JTI-ENIAC grant agreement n 120214 (END project)

REFERENCES

- [1] T. Y. Kim, H. G. Ahn, S. K. Park, and Y. K. Lee, “A novel maximum power point tracking control for photovoltaic power system under rapidly changing solar radiation,” in IEEE International Symposium on Industrial Electronics. ISIE 2001., vol. 2, pp. 1011–1014, 2001
- [2] X. Jiang, J. Polastre and D. Culler, “Perpetual environmentally powered sensor networks”, in: Proc. of IEEE Workshop on Sensor Platform, Tools and Design Methods for Networked Embedded Systems (SPOTS), Los Angeles, CA, USA, April 2005W.-K. Chen, *Linear Networks and Systems* (Book style). Belmont, CA: Wadsworth, 1993, pp. 123–135.
- [3] K. Lee, N. Chang, J. Zhuo, C. Chakrabarti, S. Kadri, and S. Vrudhula, “A fuel-cell-battery hybrid for portable embedded systems,” *ACM Trans. Des. Autom. Electron. Syst.*, vol. 13, pp. 1-34, 2008.
- [4] C. Park and P. Chou, “AmBiMax: autonomous indoor energy harvesting platform for multi-supply wireless sensor nodes,” in Proc. SECON 2006, Reston, VA, 2006, pp. 168-177
- [5] A.S. Weddell, N.J. Grabham, N.R. Harris and N.M. White, “Modular Plug-and-Play Power Resources for Energy-Aware Wireless Sensor Nodes”, in Proc. SECON 2009. Piscataway, NJ, USA, 422-430.
- [6] Y.K. Tan and S.K. Panda, “Energy Harvesting from Hybrid Indoor Ambient Light and Thermal Energy Sources for Enhanced Performance of Wireless Sensor Nodes”, IEEE Transactions on Industrial Electronics, vol.58, in-press, 2011..

Poster Abstract: Wake-up architecture for Wireless sensor nodes based on ultra low power FPGA.

V. Rosello, J. Portilla, T. Riesgo

Centro de Electronica Industrial, Universidad Politecnica de Madrid

{victor.rosello, jorge.portilla, teresa.riesgo}@upm.es

Abstract— In this work a novel wake-up architecture for wireless sensor nodes based on ultra low power FPGA is presented. A simple wake up messaging mechanism for data gathering applications is proposed. The main goal of this work is to evaluate the utilization of low power configurable devices to take advantage of their speed, flexibility and low power consumption compared with traditional approaches, based on ASICs or microcontrollers, for frame decoding and data control. A test bed based on infrared communications has been built to validate the messaging mechanism and the processing architecture.

I. INTRODUCTION

Energy usage in wireless sensor nodes is the main barrier to get fully unattended or even perpetual wireless sensor networks deployments. In order to get more energy efficient wireless sensor nodes a lot of research activities are being carried out related with all the steps of the system design, from energy sources to software development. In this work we focus on the communication layer because usually a big amount of energy is wasted listening the radio channel while there are no active communications.

In order to eliminate the waste of energy in main communications a low power wake up device can be used to activate the node only when needed, remaining in low power mode most of the time. The main requirements of this wake up devices were defined in [1]: ultra low power consumption, false wake ups should be avoided, wake-up calls should not be lost and fast wake-up call detection, in order to not impact in the overall performance of the node and the network.

The basic architecture of nodes with wake-up capabilities is a regular node with a custom wake-up transceiver (or just a receiver) for wake-up communications. To process the information of the wake-up messages normally the main processing unit of the node (typically a low power microcontroller) is used.

In this work, a new approach is proposed, based in an ultra low power FPGA as the processing unit. The use of these devices against the implementation using processors permits to reduce the wake-up time and allows incrementing the baud rate of the wake-up channel which helps to reduce the latency of the network.

The implementation presented in this work uses infrared communications. The usage of this kind of communications permits a cheap and easy way to validate the proposed wake-up message mechanism designed. This kind of

communications based on commercial infrared components present important drawbacks as the low baud rate and high power consumption. These factors make them hard to be used in real applications. However, for proof of concept they are quite suitable.

II. NODE ARCHITECTURE

The architecture of the wireless sensor node used in this work is shown in Figure 1.

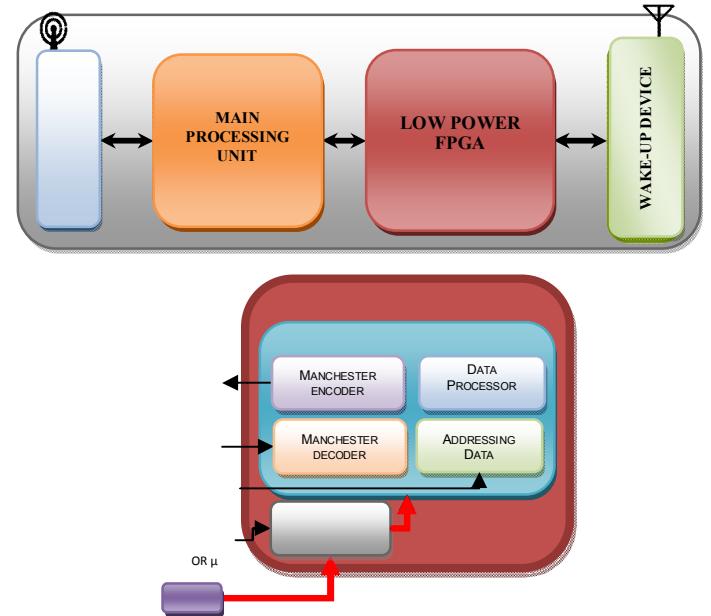


Figure 1. Node architecture.

Main characteristic are:

- Main radio channel: commercial chip based on ZigBee, working on the 2.4 GHz band.
- Microcontroller: low power microcontroller MSP430 FG438 from Texas Instruments, used for data processing.
- Ultra low power FPGA: Igloo AGL250V5 from ACTEL (250000 equivalent gates). Used to control the wake-up communications. The wake up processing architecture has been already presented in [2].
- Wake-up Device: based on an infrared LED for transmission and a commercial infrared receiver for the reception.

In order to avoid problems caused by use separate channels with different ranges the main radio channel range is limited to be under the wake-up communications range. The route between nodes and addressing assignment relays on the ZigBee communication channel. In this way asymmetrical routes are avoided and it permits to simplify the wake-up processing unit which helps to reduce the power consumption of the processing unit.

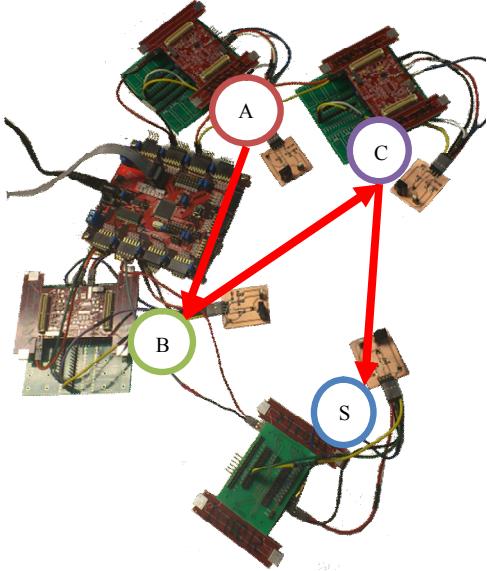


Figure 2. Testbed scenario

III. WAKE-UP MECHANISM

The mechanism proposed in this work has been designed for data gathering applications only, in order to reduce the complexity of the wake-up messaging. In this kind of applications a sensor node measures periodically their environment and sends the data to a concentrator node, named in this work sink. The communications between distant nodes are not used. With the characteristic of this scenario where only unidirectional communications are used (from a sensor node to the sink) the proposed mechanism only needs 2 types of messages to have a functional application. These messages are:

- R2SINK (Route to Sink): this message is sent to wake up the next node in the path to the sink.
- NWKRroute (Network reroute): this message is used to wake up all the nodes in the network when a route error is detected. When a node receives an NWKRroute message it broadcast the message till reach the all nodes in the network.

A. Wake-up channel characteristics

The transmission is modulated using Manchester code. This kind of codifications permits to avoid the use of preamble synchronization using very few resources of the FPGA.

The bit time is 300 μ s and the modulating frequency of the wake-up channel is 55 KHz.

IV. TESTBED

The test bed used is based on 4 wireless sensor nodes with the infrared transceiver attached to each one and an external microcontroller that acts as data logger for the state of the processing units of the nodes. The routings tables of the wake up processing units have been initialized in order to have the route shown in Figure 2.

A. Experimental results

Comparison about the implementation of the processing unit using the microcontroller and the FPGA has been made, and a summary of the results are presented in Table I.

TABLE I: WAKE-UP PROCESSING UNIT IMPLEMENTATION RESULTS

		t_{wakeup} (ns)	t_{start} (μ s)	t_{sleep} (μ s)	I_{on} (mA)	I_{sleep} (mA)
MSP	Max	--	17,22	38,79	3,2308	2,06
	Min	--	4,2	37,5	2,6194	1,47
IGLOO	Max	370	17,22	38,79	3,2308	2,06
	Min	386,5	4,2	37,5	2,6194	1,47

In Figure 3. are shown some experimental results for the message mechanism, in (a) a R2SINK message sent from A to B is shown and the original message and the response are shown. In (b) an NWKRroute broadcast is shown, the Node oA (out of figure) sent the first message that travel through the rest of nodes in the network.

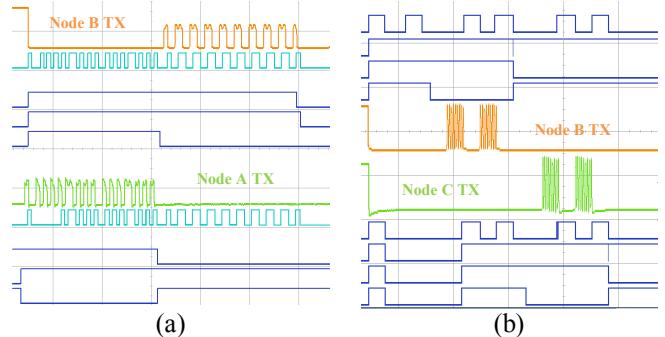


Figure 3. Experimental results (a) R2SINK (b)NWKRroute

V. CONCLUSION

As it can be seen in Table I, power consumptions are in the same range in FPGA and in microcontroller, taking into account that the FPGA works faster than necessary (11.0592 MHz) and a bigger device than necessary, the design fits on a AGL20 (20000 equivalent gates), important reduction can be achieved in the FPGA implementation.

A simple mechanism for nodes with wake-up capabilities has been tested in data gathering applications.

VI. REFERENCES

- [1] L. Gu, and J. A. Stankovic. "Radio-Triggered Wake-Up for Wireless Sensor Networks". Real Time Systems, vol. 29, pp. 157-182, 2005.
- [2] V. Rosello, J. Portilla, T. Riesgo. "Ultra Low Power FPGA-Based Architecture for Wake-up Radio in Wireless Sensor Networks" in Proceedings of the Industrial Electronics Conference (IECON'11), November 2011, Melbourne, Australia.

Industrial Demonstrations

Industrial Demo Abstract: custom design and prototypes of WSN devices

Michele Corrà, Bruno Dalvit, Emiliano Fusari

Treteco Srl, Trento, Italy

michele.corra@3tec.it, {bruno.dalvit, emiliano.fusari}@treteco.it

Abstract — The purpose of this document is to present some results of the research activity conducted on wireless devices and protocols since 2004 by Treteco Srl, a spin-off of the University of Trento. The first part of the document presents some of the most interesting wireless devices developed in order to support activities of university research groups and the ones that have been scaled up to product level. In the second part we discuss about some of the most appreciated deployments: a specific use-case is presented as example. HMPS, Hecht Museum Proximity System, is a full level system, developed for Hecht Museum (Haifa, Israel), which is able to estimate proximity between two (moving and/or static) WSN wearable devices. A complete server-side informative system parses such information in order to provide the visitor a dynamically updated support for his tour in the exhibits area.

Index Terms — WSN; IEEE 802.15.4; embedded devices; multipurpose certified platform; indoor proximity localization.

I. INTRODUCTION

Since early 2000 there has been a lot of research and interest in wireless sensor network devices, protocols and applications. This interest is demonstrated by growing availability of IC components from major worldwide manufacturers: the availability of continuously improved chip technology brings to more and more shrinking dimensions, to lower power consumption and to better radio performances. This is the way that gives the possibility to develop smaller and smarter devices which can be battery or harvester powered. All these features allow the possibility to implement pervasive systems composed by a large number of small smart devices, collecting data from the environment and able to elaborate, aggregate and share such information. A group of devices that can communicate each-other (with the rules defined by implemented protocol), interact and rely on other members of the group forms a network, a system that acts like a unique elaboration entity, but is spread in the environment.

This type of system can be used in a very large spectrum of activities starting from industrial process monitoring (e.g. sensors can be placed very close to sensitive or dangerous areas to be monitored, without the need of power and data cables), home and civil automation (e.g. monitoring of environmental parameters, actuation of lights, fans and HVAC systems, start/stop electrical devices), civil engineering monitoring (e.g. continuous monitoring of stability of buildings, roads, tunnels and bridges), human motion and presence detection (e.g. monitoring of patients in hospitals or visitors in museums).

So far many of this type of implementations are stuck in a lab and research level; few exceptions are brought up to product and industrial resellable systems. Treteco activity in

the last years was both focused in supporting research activity and investigating the opportunity of further enhancement of some of the most promising devices (and systems) for other worldwide research people and even for the end consumer market.

II. PRESENTATION OF TRETEC

Treteco [1] was founded in 2003 and mainly employs telecommunication engineers. Main provided services are devoted to the design of electronic devices on the basis of custom specifications, the development of first prototypes and series and/or re-engineering of products to be updated to new microcontroller technology (i.e. from analog to digital information elaboration). From the research point of view a strong technical relationship is active with local (FBK-Fondazione Bruno Kessler, UniTn, FEM-Fondazione Edmund Mach) and foreign research centers as Cesarea Rothschild Institute (Is) and NICTA (Au).

Since 2004 Treteco has been developing wireless devices, starting from 848 MHz radio technology up to the nowadays 2.4 GHz transceivers. Some of the designed devices take advantage of the widely known free operating systems with integrated transmission protocol stack (TinyOS/Contiki), other devices use manufacturer-provided ZigBee compliant stacks. Treteco provides also support for OS extensions to expanded hardware (i.e. drivers) or for firmware design of proprietary custom stacks in order to overcome OS and protocols limitations.

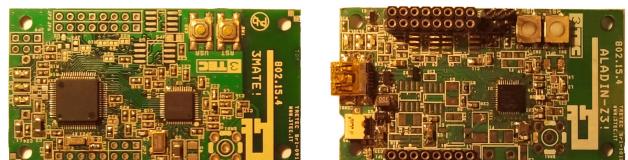


Figure 1: from 3MATE! (2006) to Aladin (2011)

The design and development process of a novel WSN device is divided into the following steps:

- ▲ component (sensors, transceivers, antennas and micro-controllers) selection and evaluation that ensures the required performances,
- ▲ schematic circuital design and simulation,
- ▲ board PCB layout design, minimizing routes and board dimensions, with particular care for RF section impedance match,
- ▲ manual SMD mount and assembly of prototypes, with feedback on schematic and PCB layout,
- ▲ design of test firmwares in order to test all board and sensors functionalities,
- ▲ design the full firmware code or provide to customer drivers and code templates for following research activity on application and algorithms.

III. PRESENTATION OF A SELECTION OF DEVELOPED DEVICES

First devices developed by Tretec belong to experimental boards, spare WSN nodes, composed only by a microcontroller, a transceiver and some I/O interfaces. These elements are indeed the core of an actual WSN node, but the most recent developed devices integrate other functionalities and features: on-board sensors (compass, accelerometer, gyroscope, barometer), additional ICs (FLASH/FRAM/uSD memories, voltage regulator, battery charge controller) or even off-the-shelf advanced modules as GPS and GPRS. Moreover the most recent devices provide flexibility in power supply input: different types and number of batteries (Li-ion, AA or AAA, alkaline or rechargeable), standard 24V AC/DC or even from harvesting sources.

The following table lists some of the devices that have been developed through years:

Year	Device	Field	On board features
2004	Tres	Research	848 MHz radio module, PIC. Sensors: temp, hum
2006	OIS7	Product (Optoi)	2.4 GHz CC2420, PIC18F2520, Vref, linear regulator, I/O expansion
2006	3MATE!	Product	2.4 GHz CC2420, MSP430F1611, external memory, I/O expansion. TelosB compatible
2008	3MATE! small_acc	Research	2.4 GHz CC2420, MSP430F1611, external memory, wearable . Sensor: 3D analog accelerometer .
2008	m3MATE! Z.POINT9	Research	2.4 GHz CC2520 , C8051F931, one coin battery
2008	TMPS_PIC	Research	848 MHz radio module, PIC. Sensor: pressure for TMPS applications
2009	3MATE! XLP	Research	2.4 GHz CC2520 , PIC18F46J50, Vref, Li-ion charger, voltage regulator, external memories, I/O expansion
2010	(mini) Aladin	Product (2011)	2.4 GHz CC2531 , external memory, Li-ion charger, voltage regulator, RS-485 driver, USB, I/O expansion
2010	Wildlife 3MATE!	Research	2.4 GHz CC2420, MSP430F2618 , FRAM memory. Sensors: temp, compass, accelerometer, I/O expansion. GPS and GSM daughter board
2011	3MATE! IGLOO	Research	2.4 GHz CC2520, MSP430F2618 , external memory, Li-ion charger, voltage regulator, FPGA , I/O expansion
2011	STM32-ARM 3MATE!	Research	2.4 GHz CC2520 , STM32, Li-ion batt & charger, voltage regulator. Sensors: 3D accelerometer, compass, 3D gyroscope, barometer, USB, uSD

IV. A SELECTION OF CUSTOM PROTOTYPES

The presented devices have been used in tests conducted by Tretec and the University of Trento. Some of them have been used in important test projects and deployments; following are the most relevant ones:

- ▲ Torre Aquila (UniTn, It) [2][3]: monitoring of heritage buildings (tower) with 3MATE! devices,
- ▲ bridges' monitoring (NICTA, Au) [4]: 3MATE! devices with custom sensor conditioning,
- ▲ deers activity monitoring (FEM, It): 3MATE! shrinked version with sensors, GPS and GPRS
- ▲ TRITon (Siemens & UniTn, It) [5], monitoring

tunnel light efficiency: 3MATE! Device w/sensors, HMPS (Hecht Museum, Is)[6]: indoor proximity system: OIS7 device,

▲ wearable inertial board (UniTN, It): 3MATE!-derived with STM32 ARM and 6D sensors (Fig 2).

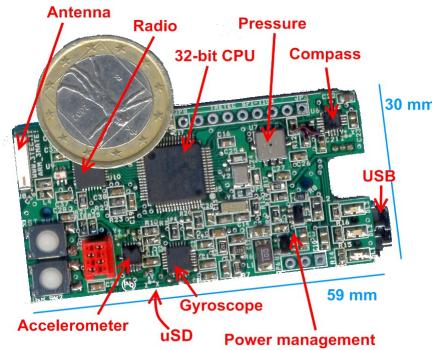


Figure 2: STM32-ARM 3MATE!

V. HMPS - HECHT MUSEUM PROXIMITY SYSTEM

HMPS is a system developed by Tretec in collaboration with two research centers: Fondazione Bruno Kessler, Italy, and Cesarea Rothschild Institute, Israel. The application runs in Hecht Museum in Haifa, Israel, and gives support both to researchers and to the visitors inside the museum. The scope is to track the visitor, his path, his interactions with exhibits as well as with other visitors and groups. All these data are used as a basis for sociological research and studies, but in the meantime a contextual server information feedback is provided to the visitor. Visitors' positions are estimated by a Tretec developed algorithm, running on OIS7 WSN device (CE certified). The WSN device, so called *blind*, is wearable, Li-ion battery powered and interacts with fixed devices called *beacons* or with other blinds for group detection. Once in a second the blinds device transmit information over their position, the voice activity and the group detection to an infrastructure of WSNtoTCP *gateways* that collect and forward data packets to a central server. As counterpart it returns, to PDAs or smart-phones, all the available information, documents, audio and/or video presentations concerning the exhibits. Particular care has been dedicated to develop the algorithm and the solutions that allow a long lifetime of the system which is up to 10 months between two battery replacements.

The implemented final system features easy and hidden installation, flexible OTA configuration, low maintenance and is user friendly. As mentioned, a feedback survey among visitors confirmed their fully positive impression. The system is active and working since November 2010.

REFERENCES

- [1] <http://www.3tec.it>; <http://www.wsnlab.it>
- [2] <http://d3s.disi.unitn.it/projects/torrequila>
- [3] M. Ceriotti, L. Mottola, G. P. Picco, A. L. Murphy, S. Guna, M. Corrà, M. Pozzi, D. Zonta, and P. Zanon. "Monitoring Heritage Building with Wireless Sensor Networks: the Torre Aquila Deployment". In Proc. of the 8th Int. Conf. on Information Processing in Sensor Networks (IPSN) - SPOTS track, 2009. Best Paper Award.
- [4] <http://www.nicta.com.au/>
- [5] <http://triton.disi.unitn.it/>
- [6] T. Kuflik, J. Lanir, E. Dim, A. Wecker, M. Corrà, M. Zancanaro & O. Stock, "Indoor Positioning: Challenges and Solutions for Indoor Cultural Heritage Sites" Proceedings of IUI-2011, International Conference on Intelligent User Interfaces, Palo Alto, 2011

MyriaNed: A biology inspired self-organizing, gossiping Wireless Sensor Network

Lex van Gijsel, Bob Peters, Zeno Korsmit, *DevLab*

Abstract— This demo shows a wireless sensor network with localization and environment monitoring capabilities, which is based on a gossiping protocol and shared state mechanism. This protocol enables very low power communication in a network that can be scaled up to many thousands of nodes.

Index Terms— gossiping, shared state, Wireless Sensor Network, scale free, self-organization.

I. INTRODUCTION

Wireless sensor networks (WSNs) are making the transition from the research field to real-world applications. These networks will become an integral part of our environment. Among many others application areas are home and building automation, elderly care, agriculture, transportation tracking & monitoring and indoor localization.

However, a number of issues have to be addressed before WSN can meet the technological requirements in the mentioned application areas. In most applications the network will consist of a very large number of sensor- and actuator nodes. So the nodes have to be low cost, very low power and small form factor. In order to have a manageable network it has to be robust, scalable, self-organizing and it has to adapt itself to the context in which it is used. This is necessary to ensure a long-life without any maintenance and low (initial) setup efforts.

II. INSPIRATION

Traditionally radio communication is organized according to a point-to-point mechanism. A command is sent top-down and a confirmation is sent bottom-up between two hierarchical levels. However in biology this is organized differently;

For instance adrenaline in the human body, the message (hormone and neurotransmitter) is sent to different types of cells. Every cell knows what to do with this message (increase heart rate, constrict blood vessels, dilate air passages) and does not send a confirmation.

Another inspiration is the basic radio broadcasting principle. A radio with an antenna is made to send and receive a message

to and from every direction. Implicitly it is not optimized to perform point-to-point communication. We believe that wireless communication should be structured in such a way that it uses the full potential of radio transmission. The third inspiration is that of human gossiping. The term is sometimes associated with spreading misinformation of trivial nature but the way information is disseminated is one of the oldest and most common in nature. Information is generated by a source and gossiped to its neighbours. They spread the message to their neighbours, thereby exponentially increasing the number of people familiar with the information.

Together these three inspirations led to the development of the MyriaNed platform. There is no notion of hierarchy in the network; rather each node is hierarchically equal. MyriaNed uses biological inspired information dissemination and is independent of the function of the node. Each node decides what to do with a message. Furthermore it sends the message to all its neighbours thereby using the basic radio communication characteristics.



Fig. 1. Logo MyriaNed

III. METHOD

The technology is based on a bottom up approach, where the behaviour of a single element (node) will result in emerging behaviour of the system (application). An arbitrary node will broadcast a new message into the network. In general this will be initiated by a new sensor reading. Because nodes will retransmit the received information by gossiping, the information is spread throughout the network.

Since the information is shared among the nodes a natural state in network emerges, which is called the **shared state**. By utilizing a smart algorithm the **shared state** can be used for data aggregation, sensor fusion, pattern recognition, and perform high level reasoning. This will lead to driving actuators or share the knowledge with the user.

The communication protocol is event driven and based on an energy efficient TDMA schedule, combined with gossiping and the shared state algorithm the network is scale-free. In order to orchestrate the periodic gossiping, the nodes are

synchronized and share a global notion of time, the so called global time established by the heartbeat of the network.

This complete concept has proven to be very robust and forms the basis of many applications that can be built on top of the MyriaNed Wireless Sensor Network Technology.

IV. DESCRIPTION OF THE DEMO

In this demonstration a proof of concept is shown on localization and environment monitoring with a MyriaNed network. Determining the location of a node is based on a so called collision based localization method.

Since context awareness is important for a distributed network, localization is a primary requirement for many applications. It also proofs the wide range of applications that can be implemented on the MyriaNed technology.

V. BENEFITS

By using this biological inspired approach the network has the following Unique Selling Points (USPs):

- Scalable
- Ultra low power
- Ad-hoc established network connectivity
- Elegant way of coping with mobility
- Low cost
- No hierarchy, implies no single point of failure
- Notion of global time
- Shared state

Industrial demo of two office applications, based on WSN-SI and MyriaNed

Joost van Velzen, Herman Tuininga, *SallandElectronics*

Abstract— This demo shows two office application based on the wireless sensor network MyriaNed developed by DevLab. DevLab is a research laboratory established by 13 high-tech SME companies. The WSN based office demo shows a smart sign and a smart chair.

Index Terms— MyriaNed, low power, Wireless Sensor Network, WSN-SI, large scale networks, connected, smart display, smart chair.

I. INTRODUCTION

Wireless sensor networks (WSNs) are making the transition from the research field to real-world applications. These networks will become an integral part of our environment. Among many others application areas are home and building automation, elderly care, agriculture, transportation tracking & monitoring and indoor localization.

However, a number of issues have to be addressed before WSN can meet the technological requirements in the mentioned application areas. In most applications the network will consist of a very large number of sensor- and actuator nodes. So the nodes have to be low cost, very low power and small form factor. In order to have a manageable network it has to be robust, scalable, self-organizing and it has to adapt itself to the context in which it is used. This is necessary to ensure a long-life without any maintenance and low (initial) setup efforts.

II. ABOUT WSN-SI

The Shared Infrastructure for Wireless Sensor Networks (WSN-SI), is a wireless sensor network invented by Joost van Velzen and Houwer de Geus of **SallandElectronics** BV. In part, WSN-SI is based on the MyriaNed Wireless Sensor Network developed by Devlab (see www.devlab.nl for references to academic research on which MyriaNed is based). **SallandElectronics** BV is a member of Devlab and is also contributing to MyriaNed development. The differences between the two networks are that MyriaNed is designed to be application specific and is an ad-hoc self-organising network.

WSN-SI is a generic shared infrastructure of router nodes that requires some configuration for the positioning feature.

WSN-SI distinguishes between generic router nodes and application specific nodes. The router nodes transport data through the network and may implement generic services such as the positioning service. Other nodes are typically application specific and only use the data transport and positioning services provided by the router nodes. The advantages are that the network is not dependent on application specific nodes and that the application specific nodes can be both mobile and extremely low power.

III. ABOUT MYRIANED

MyriaNed is a **wireless sensor network (WSN)** platform developed by **DevLab**. It uses an epidemic communication style based on standard **radio broadcasting**. This approach reflects the way humans interact, which is called **gossiping**. Messages are sent periodically and received by adjoining



Fig. 1. Logo MyriaNed

neighbors. Each message is repeated and duplicated towards all **nodes** that span the network, it spreads like a **virus** (hence the term epidemic communication).

This is a very efficient and robust protocol, mainly for two reasons:

- First, the nodes do not need to know who is in their neighborhood at the time of sending a message, there is no notion of an a-priori planned **Routing**, data is just shared instantaneously.
- Second, the network is implicitly reliable since messages may follow different communication routes in parallel. The loss of a message between two nodes does not mean that the data is lost.

Nodes can be added, removed or may be physically moving without the need to reconfigure the network. The **GOSSIP** protocol is a self-configuring network solution. The network may even be heterogeneous, where several types of nodes communicate different pieces of information with each other at the same time. This is possible due to the fact that no

interpretation of the message content is required in order to be able

IV. DESCRIPTION OF THE DEMO'S

To test and validate the WSN-SI and MyriaNed technology two applications are developed. Both demos are office applications which can be integrated in the near future.

Smart Display demo – The Smart Display shows a wireless, battery powered system, which can be used for office- and meeting rooms. The display shows information about the room. For an office room it could be the room number, the person who uses the room and if the person is in the room. For a meeting room it could be the name of the room, the reservation time of the room, what kind of meeting it is etc. Due to the used wireless sensor technology the deployment in a hallway is easy. Messages are relayed from the one display to the other. To put messages on the network only 1 RF stick and a PC is necessary within reach of the network.

office applications are:

- Easy integration
- Scalable
- Large scale integration
- Long battery lifetime
- Small form factor
- Flexible in use

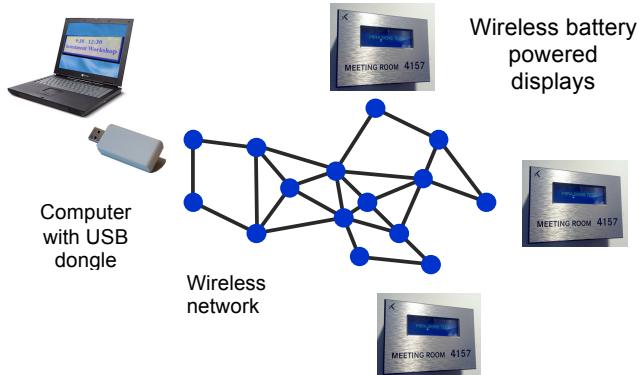


Figure 1 Overview demo

Smart Chair demo – The Smart Chair shows an ergonomic office chair that measures if the person is correct using the office chair. The chair has a wireless interface, which makes it possible to integrate the chair in a wireless sensor network. The chair is battery powered with a lifetime of 7 years. Besides the ergonomically benefits of the chair also integration within building automation can give information about how many people are working in a specific office space.



V. BENEFITS

The benefits for using wireless sensor technology for the demo

Industrial Demo Abstract: Energy Monitoring

Manuel Fernández, Juan Pablo Viñuela

Product development - ADVANTICSYS

{manuel.fernandez, juanpablo.vinuela}@advanticsys.com

Abstract— The purpose of this demonstrator is to show a system for monitoring energy consumption in the home and work environment. The solution is based on a wireless sensor network platform, combined with certified measuring components. Wireless operation of the individual devices is based on open standards for easy integration and interoperability with other subsystems.

I. INTRODUCTION

USING wireless sensor networks (WSNs) for auditing and managing the energy consumption in a building is an emerging research area that includes a number of novel applications such as activity pattern recognition, adaptive load shifting, and building energy profiling in domestic and industrial settings. ADVANTICSYS research is focused in the development of the necessary equipment to conduct temporary energy metering deployments, as well as metering over extended periods of time.

II. DESCRIPTION OF THE DEMONSTRATOR

A. Overview of the system

The demonstrator corresponds to a system capable of measuring energy consumption both in the building as a whole as in individual appliances inside it. From the start the architecture design demanded a distributed approach, in which the individual components presented a dual nature, as the deployed devices had to act as measuring points of the electric system and also as nodes of the wireless communication infrastructure.

All the gathered data had to be collected in a central location, processed and presented to the customer. A gateway component was therefore developed to provide easy access to the information, which involved the development of the necessary software in the form of a web application.

B. Hardware design

As mentioned, the system had to be capable of measuring the overall building consumption as well as that of the individual appliances. This resulted in the development of two devices:

- Individual appliance smart plugs
- Single-phase wall energy meter

Smart plugs were developed with the standard electrical socket design in mind. One of the requirements was to enable basic control of the connected device, and as a result an on-off relay switch was incorporated. A commercial energy metering chip was incorporated in the design, and the IEEE 802.15.4

wireless connectivity was added by including ADVANTICSYS well known wireless module platform.



Fig. 1. Smart Plug Concept. Smart plugs allow the measurement of electrical energy consumption in individual devices throughout the home or office. Combined with a basic on-off relay control chip, these devices can effectively reduce remnant energy waste.

The single phase wall energy meter was developed in a similar fashion. This provided the system with the ability to measure electrical lines as a whole from the electrical supply line of the building. Depending on the building, this allows easier monitoring of full sections of the electric installation, such as the lighting or the HVAC, resulting in minimization of the total number of devices deployed.

The smart plug device had to undergo a thorough certification process to ensure electric and electromagnetic compliance with existing regulations, as well as adequate accuracy of the measurements, while the single-phase wall energy meter will follow the same steps in the near future.

C. Measured Magnitudes

The developed hardware is capable of measuring a broad range of different magnitudes, including active, reactive and apparent energy, and RMS voltage and current.

For the purpose of the demonstration itself, active energy was the main scope of interest.

D. Network Topology

From the beginning the system focused on open standards for its firmware development. As such, a TinyOS approach was used, as this ensured the compatibility of the network of wireless meters with other ADVANTICSYS ambient monitoring sensors. Knowing the temperature and humidity data and gas concentration levels improve the customer analysis of the building conditions, without the expense of having a separate network coexisting.

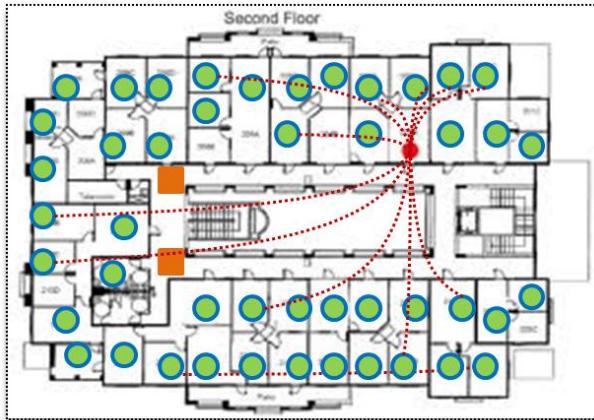


Fig. 2. Network Topology. Smart Plugs, being powered continuously act as master or repeater nodes of the network. This enables the rest of the deployed battery operated nodes to act as end devices, reducing the overall power consumption.

As network topology, a cluster tree design was adopted, as it was more than enough for the demonstration, where no scalability issues were considered.

E. Software Application

The system demonstration completes itself with a remote monitoring software application, consisting in local data base storage and remote access to the data. This ensures that the customer knows in real time the status of the deployed nodes, and can begin performing their analysis while it is being carried out, not having to wait until the devices are collected.

The software was included inside a gateway component, designed with an embedded IEEE 802.15.4 interface as sink node.

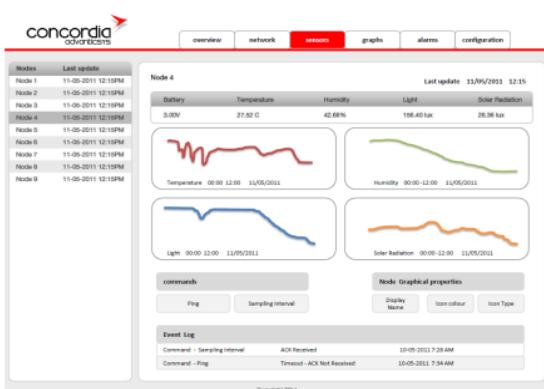


Fig. 3. Web-based software application for data gathering and analysis.

III. ACADEMIC BACKGROUND

This demonstration was the result of a collaboration between ADVANTICSYS and a company specialized in energy audits prior to building refurbishments. The partnership with an expert in the energy efficiency sector provided the necessary requirements, both for the hardware and the software, that resulted in the presented solution.

The most notable requirements were:

- Ease of deployment and installation
- Measurement accuracy error under 5%
- Open network infrastructure, compatible with existing ambient monitoring deployment
- Remote real time access to gathered data
- Secure storage of information at device level in prevention of gateway failure

The collaboration was reduced not only to the scope of a single project but has been also the base upon which new products are being developed.

IV. CONCLUSION AND FUTURE WORK

The results of this project has been the development of a portable easy to install energy audit kit, composed of a set of smart plug devices, preprogrammed in a standard operation mode, as well as the associated remote monitoring software application.

As natural evolution from the developed work, the following products will become available in the near future:

- Smart plugs for researcher and integrators, with no firmware installed, just API access to the measurement chip
- Implementation of IPv6 layer upon the deployed smart plugs, to solve potential scalability issues
- Single and Tri Phase wall meter for the industrial realm, focused on production line active and reactive energy measurement

Looking more into the far future, the possibilities of remote monitoring in energy consumption are almost endless. ADVANTICSYS is exploring new business lines, collaborating with different utilities in order to achieve a reliable smart metering device capable of removing the need of physical inspection of the devices. At appliance level, more refined control of the devices is also a subject to consider, especially related to lighting control.

Author Index

A

- Abid, Mohamed, 20
Al-Shalfan, Khaled, 20
Alidori, Emanuele, 18
Andreopoulos, Yiannis, 44

B

- Balsamo, Domenico, 14
Benini, Luca, 86
Bennaceur, Hachemi, 20
Bernauer, Alexander, 29
Boano, Carlo Alberto, 36
Brown, James, 34
Brunelli, Davide, 14, 86
Buchmann, Alejandro, 66
Buranapanichkit, Dujdow, 44
Butler, Michael, 55

C

- Casati, Fabio, 25
Ceriotti, Matteo, 8
Cervenka, Vladimir, 16, 48, 73
Chaâri, Imen, 20
Chaâri, Rihab, 20
Chen, Yu, 50
Colitti, Walter, 23
Corrá, Michele, 91
Cugola, Gianpaolo, 27

D

- Daidone, Roberta, 69
Dalvit, Bruno, 91
Daniel, Florian, 25
Dantchev, Guenadi, 25
Darwazeh, Izzat, 50
De Caro, Niccoló, 23
De Filippis, Tiziana, 12
de Oliveira, Bruno Trevizan, 75
Di Buó, Gianluca, 18
Di Gennaro, Filippo, 12
Di Mauro, Alessio, 84
Dini, Gianluca, 69
Donzelli, Claudio, 62
Dragoni Nicola, 84
Dudek, Denise, 71

E

- Elsts, Atis, 31
Eriksson, Joakim, 25

F

- Fafoutis, Xenofon, 84
Fernández, Manuel, 97
Figura, Richard, 6
Finne, Niclas, 25
Fiorillo, Edoardo, 12
Fürster, Alexander, 60
Fürster, Anna, 60
Fortino, Giancarlo, 44
Fritzke Jr, Udo, 40
Fusari, Emiliano, 91

G

- Gaddour, Olfa, 20
Gambardella, Luca, 60
Gao, Chao, 42
Garg, Kamini, 60
Genesio, Lorenzo, 12
Giordano, Silvia, 60
Grisostomi, Massimo, 18
Guerrero, Pablo, 66

H

- Haas, Christian, 80
Handziski, Vlado, 62
Hasenfratz, David, 10
Hergenröder, Anton, 80
Hidalgo, Ieda, 40

J

- Jamâa, Maissa Ben, 20, 53
Judvaitis, Janis, 31
Jungen, Sascha, 6

K

- Kapitza, Rüdiger, 64
Karnouskos, Stamatis, 25
Kayani, Yasir, 20
Khelil, Abdelmajid, 46, 66
Korsmit, Zeno, 93
Koubâa, Anis, 20, 53

Kuroiwa, Takuto, 42

L

Lenahan, Russell, 55

Longhi, Sauro, 18

Lukas, Florian, 64

M

Madsen, Jan, 84

Magno, Michele, 86

Mandal, Partha Sarathi, 57

Marfievici, Ramona, 8

Margi, Cíntia Borges, 75

Marrón, Pedro José, 6

Martin, Richard, 55

Martins, Paulo, 40

Marzioni, Davide, 18

Matese, Alessandro, 12

Meier, Florian, 82

Menezes, Ronaldo, 40

Minohara, Takashi, 77

Molteni, Davide, 8

Mondal, Kaushik, 57

Moreno Montero, Patricio, 25

Morikawa, Hiroyuki, 42

Mottola, Luca, 25

Mraz, Lubomir, 16, 48, 73

Murphy, Amy L., 8

N

Niedermann, Markus, 80

Nishita, Seikoh, 77

Noori, Abouzar, 10

O

Oppermann, Felix Jonathan, 25

P

Paci, Giacomo, 14

Pechanec, Vilem, 16

Peters, Bob, 93

Phung, Ha, 23

Picco, Gian Pietro, 8, 25

Pirro, Matteo, 18

Porcarelli, Danilo, 86

Portilla, Jorge, 88

Primicerio, Jacopo, 12

Prist, Mariorosario, 18

Puccinelli, Daniele, 60

Q

Quartulli, Antonio, 25

R

Renner, Christian, 82

Riesgo, Teresa, 88

Rocchi, Leandro, 12

Roedig, Utz, 34

Römer, Kay, 25, 29, 36

Rosello, Victor, 88

Rothenpieler, Peter, 38

Ruggiero, Wilson Vicente, 75

S

Sachidananda, Vinay, 46

Saukh, Olga, 10

Selavo, Leo, 31

Shih, Chia-Yen, 6

Simek, Milan, 16, 48, 73

Sinha, Bhabani, 57

Sivieri, Alessandro, 27

Soleymani, Ramin, 6

Spieß, Patrik, 25

Steenhaut, Kris, 23

Strübe, Moritz, 64

Suri, Neeraj, 46

Suzuki, Makoto, 42

T

Tezeghdanti, Miled, 20

Thiele, Lothar, 10

Tiete, Jelmer, 23

Tiloca, Marco, 69

Touhafi, Abdellah, 23

Tranquillini, Stefano, 25

Trigui, Sahar, 20

Tuininga, Herman, 95

Turau, Volker, 82

U

Ulrich, Tamara, 10

V

Vaccari, Francesco P., 12

van Gijsel, Lex, 93

Van Laerhoven, Kristof, 66

van Velzen, Joost, 95

Viñuela, Juan Pablo, 97

Vittorioso, Antonio, 44

Voigt, Thiemo, 25, 36

Vuckovic, Dusan, 84

W

Wilke, Joachim, 80

Wiskot, Thomas, 80

Wolisz, Adam, 62

X

Xiao, Jian-Hua, 77

Z

Zúñiga, Marco Antonio, 36