KTH Technology and Health

Performance Monitoring and Control in Wireless Sensor Networks

Ibrahim Orhan

Licentiate Thesis
June 2012
TRITA-STH Report 2012:3

Department of Computer and Electrical Engineering
School of Technology and Health,
KTH (Royal institute of Technology)

Academic dissertation which with permission from Kungliga Tekniska Högskolan (Royal Institute of Technology) in Stockholm is presented for public review for passing the Licentiate of Technology degree on Tuesday June 5, at 9.15. In Lecture hall 3-221, Alfred Nobels allé 10, Huddinge, Sweden.

TRITA-STH Report 2012:3
ISSN: 1653-3836
ISRN/KTH/STH/2012:3-SE
ISBN 978-91-7501-354-1

©Ibrahim Orhan, Stockholm 2012

Abstract

Wireless personal area networks have emerged as an important communication infrastructure in areas such as at-home healthcare and home automation, independent living and assistive technology, as well as sports and wellness. Wireless personal area networks, including body sensor networks, are becoming more mature and are considered to be a realistic alternative as communication infrastructure for demanding services. However, to transmit data from e.g., an ECG in wireless networks is also a challenge, especially if multiple sensors compete for access. Contention-based networks offer simplicity and utilization advantages, but the drawback is lack of predictable performance. Recipients of data sent in wireless sensor networks need to know whether they can trust the information or not. Performance measurements, monitoring and control is of crucial importance for medical and healthcare applications in wireless sensor networks.

This thesis focuses on development, prototype implementation and evaluation of a performance management system with performance and admission control for wireless sensor networks. Furthermore, an implementation of a new method to compensate for clock drift between multiple wireless sensor nodes is also shown. Errors in time synchronization between nodes in Bluetooth networks, resulting in inadequate data fusion, are also analysed.

Keywords: body sensor networks, wireless sensor networks, performance monitoring, performance control, admission control, Bluetooth performance, clock drift compensation, time synchronization.

Acknowledgments

First and foremost, I would like to thank my supervisor Associate Professor Thomas Lindh for his research initiative in wireless communication services for healthcare and giving me the opportunity to participate in this work. I highly appreciate our discussions, his ideas, support and encouragement. I am sincerely grateful to the former dean of KTH STH, Inge Jovik, for giving me the opportunity to do my research studies besides my ordinary lecturer employment. And I am also sincerely grateful to the dean of KTH STH, Professor Lars-Åke Brodin, for giving me the opportunity to continue my research to a licentiate degree. I would also like to thank my co-supervisor Professor Gunnar Karlsson for all ideas in the start-up of our research. A special thank goes to my colleague and head of the department Magnus Brenning who has supported me all the way up. My colleague, Jonas Wåhslén, who also is co-author of some of the papers; I would like to thank you for all the nice conversions and exchange of ideas we have had in our PhD studies. I would also like to thank all my colleagues at KTH STH for their support.

I owe sincere thanks to Professor Tore J Larsson and Stefan Lundberg and the staff at Centre for Health and Building at KTH for supporting us with equipments but also let us use the full-scale lab for lifelong dwelling in our research. I would also like to thank, Antonio Gonga for his excellent help with TinyOS and how to program in nesC. I also want to thank Jens Burmeister from Dresden Technical University who did an internship at STH for all contribution and ideas you had within our research field.

A special thank goes to my grandfather Ibrahim and my grandmother Hülkiye who took care of me and my sister since we were little kids, and to my father Ali and sister Fatma Gül for their support. Finally, I want to thank my wife Meltem, my son Orkun and my daughter Defne for their unflagging love and support throughout my life.

Table of contents

1 Introduction

Wireless embedded Internet and Internet of things have emerged as important concepts in the recent development in information technology ([1],[2]). The rapid development in micro electronics, radio communications, Internet technology, operating systems and software for embedded systems have paved the way for extensive deployment of wireless sensors, often connected to smart mobile phones. Wireless communication is increasingly used in medical, healthcare, wellness and sports applications. It has also become an important infrastructure for smart homes and ambient assisted living as a way to meet the challenge of the rapidly growing elderly population. Wireless devices with sensors, e.g. ECGs, EEGs, accelerometers, and pulse-oximeters, are expected to be used extensively in at-home healthcare. However, a necessary condition for deployment of wireless sensor communication is that the transmitted medical and health-related parameters are received correctly and can be trusted. Performance monitoring is therefore crucial to verify to provide information about the transmission quality, and to possibly enable feedback control actions.

This thesis addresses performance issues in two different radio communication technologies; contention-based access in the radio layer of the IEEE 802.15.4 standard, and Bluetooth piconets. The IEEE radio standard using CSMA/CA has essentially the same access method as wireless local area networks (IEEE 802.11), whereas Bluetooth uses dedicated timeslots for communication between master and connected slaves. Contention-based systems offer simplicity and utilization advantages, but the drawback is lack of predictable performance. The thesis presents the results of developing a prototype implementation of a performance monitoring and control system in contention-based networks. Bluetooth is so far the dominating radio standard between a mobile phone and wireless sensors. Applications that use fusion of data from multiple sensors require that the actual time for the samples is preserved. The thesis analyzes performance problems in time synchronization using Bluetooth and IEEE 802.15.4 networks. In addition, a new algorithm for compensation of clock drift in the sensor nodes is presented.

1.1 Aim

The main purpose of this thesis work has been to develop and evaluate measurement, monitoring and performance control functions for wireless sensor communication systems. The system has been implemented and evaluated in contention-based access networks (the CSMA/CA radio part in the IEEE 802.15.4 standard). Healthcare in hospitals, ambient assisted living at-home or special housing, wellness and sports are the targeted applications. Bluetooth, which is a different communication technology, has also been analysed regarding time synchronization between multiple sensors. In addition, a method to compensate for clock drift between multiple wireless sensor nodes has been developed and evaluated. Time synchronization is essential for fusion of data from multiple sensors.

The performance system is inspired by a model used in [3] for wireline communication and developed in [4]. It has been adapted to the specifics in wireless sensor networks. An experimental approach has been adopted with a prototype implementation. Repeated measurements have been evaluated and analysed.

1.2 Contributions of the thesis

The major contributions of this thesis are:

- Implementation, testing and evaluation of a performance monitoring method of a light-weight end-to-end performance meter for quality-demanding applications in wireless body sensor network.
- Design and implementation of a method for performance control in wireless body sensor networks based on feedback from a performance meter for healthcare applications.
- A prototype implementation and evaluation of a performance management system using admission control combined with continuous performance control for contention-based wireless sensor networks.
- Analysis and evaluation of time synchronization performance algorithms using Bluetooth and Zigbee/IEEE 802.15.4 networks.
- Design, implementation and evaluation of a method to detect and measure the clock drift in wireless sensor nodes. Furthermore, a new algorithm for clock drift compensation in real time has been developed, implemented and evaluated.

2 Performance in wireless sensor networks

Performance in contention-based wireless networks using CSMA/CA has been studied extensively. Measurements, simulations and theoretical studies show that the loss ratio increases with the traffic load and number of sending nodes. Bianchi [5] has derived an analytical Markov chain model for saturated networks, further developed in [6] and extended to non-saturated networks in [7]. Channel errors due to e.g., external disturbances and obstacles in the environment, can of course increase the loss ratio further. Another related problem, studied in [8], is the reduced throughput in multi-hop networks, with one or several intermediate nodes between sender and receiver.

Performance in low-rate WPAN has been analyzed in several simulation studies ([9] and [10]). Several papers have also addressed congestion and rate control in WLAN and LR-WPAN. CODA (congestion detection and avoidance in sensor networks) is a control scheme that uses an open-loop backpressure mechanism as well as a closed-loop control, where a sink node can regulate a source node's sending rate by varying the rate of acknowledgements sent to the source [11]. CARA (collision-aware rate adaptation) uses the RTS packets in IEEE 802.11 as probes to determine whether losses are caused by collisions (related to CSMA/CA) or by channel errors [12].

It is also possibly to measure the available capacity between two endpoints, or on specific links in a network. Pathrate, Pathload and BART are examples of implementations of such estimation tools ([13], [14] and [15]). SenProbe [16] estimates the maximum achievable rate between two endpoints in wireless sensor networks by injecting packet trains and analyze the dispersion between the packets. Some experimental studies indicate that measurements of available capacity in wireless networks often are inaccurate, especially for multiple hops [17]. Instead of active measurements, the contention-aware admission control protocol (CACP) estimates the available capacity by letting each node measure the amount of time the channel is busy [18]. Perceptive admission control (PAC) is an extension of CACP to encompass node mobility [19].

2.1 A performance manager

A network scenario for the performance management system presented in this thesis is shown in Fig. 1. It consists of wearable sensors, such as ECGs, accelerometers, pulse-oximeters, fixed environment sensors, a coordinator, and intermediate nodes with routing and forwarding capabilities. An application program, running in the coordinator, processes sensor data from the sources and sends the information along with an estimate of the transmission quality to the remote end-user application for presentation and storage. The transmission quality can be expressed in terms of e.g., the statistical uncertainty of estimated parameters and the highest frequency component in a signal to be recovered by the receiver.

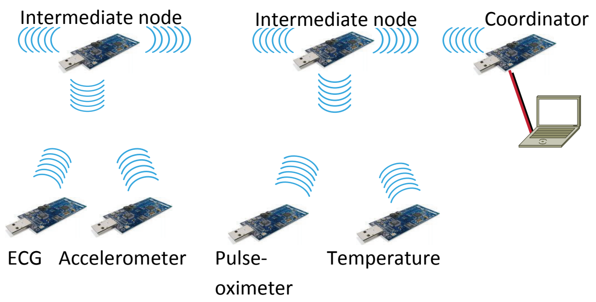

Figure 1. A network scenario where the performance management system is implemented in the coordinator and source nodes.

The performance monitoring and control capabilities can be implemented as add-on functions to be used by applications running in the communicating end-points, e.g., sensor nodes and a coordinator, and not link by link. The ambition has also been to minimize the traffic overhead and energy consumption. The system is targeted to wireless sensor networks that use contention-based access, but can of course also be used in combination with contention-free access, such as guaranteed time slots. The applications, e.g., streaming data from accelerometers and ECGs, require certain levels of throughput and a low loss ratio, however not necessarily zero. The aim is, firstly, to provide quality estimates of the transmitted parameters, and secondly, to reuse this information for admission and performance control of information loss, delays and throughput. This closes the loop between measurements and control.

The performance manager consists of the following functions: a performance meter that collects measurement data; admission control that handles requests to join the network; and performance control that maintains the quality of service for the admitted sensor nodes. The performance meter provides feedback information for admission and performance control. Fig. 2 shows the relationship between these functions. A request from a sensor node to join the network is handled by the admission control based on feedback from the meter. The performance control function is responsible for maintaining the desired quality-of-service once the sensor nodes are allowed to use the wireless channel.

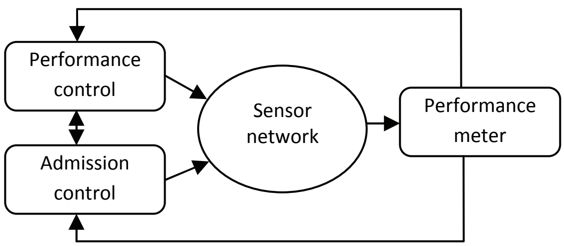

Figure 2. The performance manager consists of performance control and admission control. The performance meter supports the manager with measurement data.

2.2 The performance meter

A light-weight performance meter that combines active and passive meter techniques has been developed and implemented in each wireless sensor node. The meter consists of two counters that keep track of the number of sent and received packets and bytes, and a function that can inject monitoring packets. These dedicated measurement packets are inserted between blocks of ordinary data packets as seen in Fig. 3. They contain a sequence number, a timestamp and the cumulative number of packets and bytes transmitted from the sending node to the receiving node.

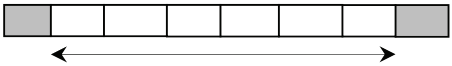

Figure 3. A monitoring block surrounded by two monitoring packets.

The interval between the monitoring packets, i.e. the size of the monitoring block, can be expressed in number of packets or a time interval, constant or varying randomly around a mean value. When a monitoring packet arrives, the receiving node stores a timestamp and the current cumulative counter values of the number of received packets and bytes from the sending node.

Synchronization of the clocks in the participating nodes is not required. The local timestamps are used to calculate the inter-sending and inter-arrival times between pairs of monitoring packets. This means that the arrival time variation is estimated based on the monitoring packets, which represent samples of the ordinary data packet inter-arrival variation. Packet loss, on the other hand, is measured passively and directly using the counters.

2.2.1 Performance metrics

The following metrics can be calculated and estimated based on the collected measurements.

- Packet loss ratio: long-term average and average per monitoring block.
- The length of loss and loss-free periods defined as the number of consecutive monitoring blocks with or without losses. Can be expressed in time units, number of blocks, or number of packets and bytes.
- Inter-arrival jitter, J, is defined as $J=(r_n-r_{n-1})-(s_n-s_{n-1})$, where s is the sending time and r is the receiving time. The monitoring packets provide samples of this delay variation metric, which means that the uncertainty of the estimated statistics (mean value, median, percentiles etc.) is determined by the number of samples, and the variance of the delay process.
- Data throughput between sender and receiver can be calculated as a long-term average and also per monitoring block. The resolution of the peak rate is determined by the ratio between monitoring packets and ordinary data packets. This can also be seen as a measure of utilized capacity.

2.2.2 Meter and monitoring packet implementation

The performance meter is programmed in nesC [20] for TinyOS 2.1. The sensor nodes read samples from the sensors (ECG, accelerometer and temperature), assemble the samples and send them in packets to the coordinator (Fig. 4). The number of bytes and packets are counted. The cumulative number of bytes and packet and a timestamp are inserted into a monitoring packet, which is sent after every n ordinary data packet.

Each time the coordinator receives a data packet, it updates the number of bytes and packets received from each sensor. When the coordinator receives a monitoring packet, it stores a timestamp and the cumulative counter values of the number of received packets and bytes from the sending node. Fig. 4 shows the measurement data sent from the performance meter to the performance manager. The table in the lower left part of Fig. 4 shows the information in each monitoring packet sent from a sensor node: a timestamp, the total number of bytes and the total number of packets sent from the sensor node. The table to the right shows the corresponding information added by the coordinator for each received monitoring packet.

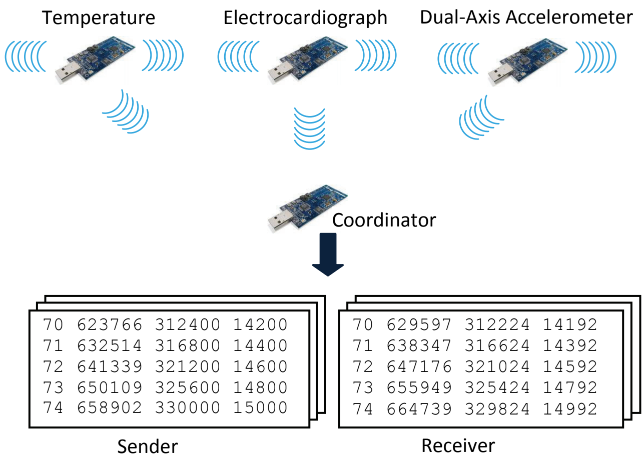

Figure 4. Measurement data from the sender and the receiver nodes. Columns from left to right: monitoring packet sequence no, timestamp (ms), cumulative number of bytes and packets.

2.3 Admission and performance control

A typical application scenario is healthcare at-home with a number of sensors, such as ECGs, pulse-oximeters, accelerometers etc., connected to a coordinator. Fig. 5 shows a scenario with three sensor nodes connected to a coordinator sharing the same wireless channel that applies the CSMA/CA access method. Several hops between the sensor nodes and the coordinator, as well as mobile sensor nodes, are also a feasible scenario.

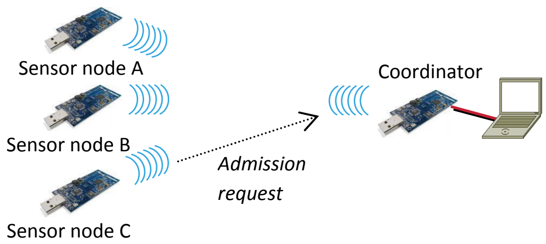

Figure 5. Sensor node C requests to share the wireless channel, already used by sensor node A and sensor node B.

2.3.1 Measurement-based admission control

The idea behind admission control is to accept or reject new sensor nodes to an existing network, while protecting the performance of already admitted nodes. Our purpose is to study whether it is feasible or not to use admission control in contention-based wireless sensor networks. The approach is to found the decision, to accept or reject an admission request, on estimates of real-time measurement data provided by a performance meter. A sensor node that intends to enter the network specifies the sampling rate, the sample size and the performance requirements. The verdict, to accept or reject the request, is determined by the outcome of probe packets transmitted during a test period. The probe packets sent from the requesting node to the coordinator should be of the same kind as the ordinary traffic it will transmit if admitted.

A simple protocol for exchange of messages between the coordinator and the sensor nodes have been defined (Fig. 6). Sensor nodes send requests to join the network for a specified sampling rate, sample size and upper limits on performance parameters. If the coordinator is not busy handling previous requests, it will approve further processing. The sensor node is then instructed to start transmitting probe packets interleaved by monitoring packets. When the test period ends, the sensor node asks the coordinator for the decision. Having received ‘accept’, the sensor node begins transmitting its ordinary data packets to the coordinator. Monitoring packets are inserted between blocks of n data packets or with certain time intervals, to provide the performance meter in the coordinator with real-time updates of the transmission quality.

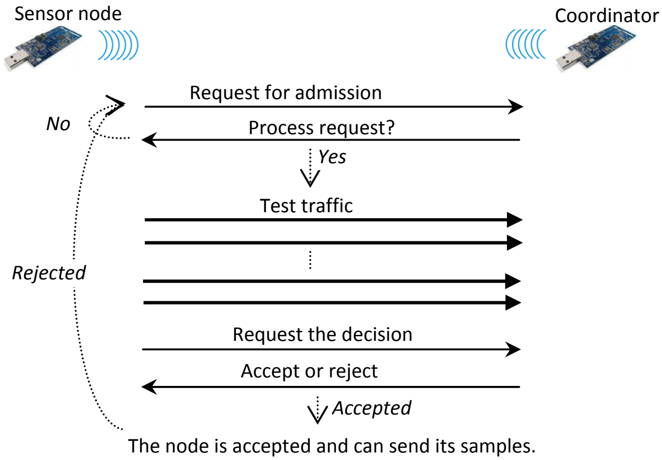

Figure 6. The messages between a sensor node and the coordinator during the admission phase. The arrows in thin lines are signalling messages and the arrows in bold lines represent probe packets in the test traffic phase.

Strict performance guarantees are not feasible in contention-based access networks. However, many applications do not require completely loss-free transmission and are satisfied with soft performance requirements e.g., upper limits on packet loss and delay variation. The need for performance guarantees and predictability in contention-based networks for such applications is addressed in Paper IV in the appendix. The detailed analysis of the test period for admission control and the decision to accept or deny the request for admission is presented in Paper IV in the appendix.

2.3.2 Performance control system

The aim of the performance control system is, firstly, to provide quality estimates of the transmitted parameters, and secondly, to reuse this information for systems management and enable performance control in real-time e.g., to minimize information loss and maintain a desired throughput. The output of the performance control system can also be to change the transmission power, enable or disable acknowledgement, etc. Applications, such as streaming data from accelerometers and ECGs require certain levels of throughput and a low loss ratio, however not necessarily zero.

The performance control system implemented in a coordinator node, bases its decisions on the feedback information it receives from the meter e.g., packet loss, delays and throughput (packet loss in used in our studies). The meter delivers these performance updates for each incoming monitoring block e.g., once a second. The performance monitoring and control method has three main parameters and is presented in Paper IV in the appendix.

2.3.3 Control algorithms

The output of the control algorithm, to decrease or increase the packet frequency, is based on performance data from the current and previous monitoring blocks. The manager sends a request message to a sensor node to either reduce or increase the packet frequency by adding (or subtracting) Δt milliseconds to (or from) the time interval between the transmitted packets.

A performance control system can support quality-of-service by assigning different priority to sensor nodes. Performance control with priority is primarily based on feedback information regarding packet loss and throughput from the respective source nodes. A control algorithm with two priorities, high and low, has been implemented. High priority means that the required throughput (received bits per second) is maintained and the packet loss ratio is kept below a threshold for the prioritized nodes, possibly at the expense of nodes with low priority. If the loss ratio for the high-priority node is above the threshold, the manager will instruct the low-priority sensor nodes, to decrease their transmission rate step by step until the loss ratio for the high-priority node is below the threshold. If the loss ratio still is above the threshold, the sending rate of the high-priority nodes will be decreased as well, and eventually turned off if the loss ratio remains too high.

2.4 Bluetooth performance in data fusion

Time synchronization is a necessary prerequisite for fusion of data from several wireless sensors. It is crucial that data samples from multiple sensors that for example arrive at a mobile phone have correct timestamps or can be estimated correctly. We have observed that errors in time synchronization, resulting in inadequate data fusion depends on the Bluetooth behavior and transmission pattern. This can differ from vendor to vendor depending on the implemented scheduler. Analysis of Bluetooth performance based on recordings of sniffer data is presented in Paper V in the appendix.

2.5 Clock drift compensation

Clock drift of the sensor nodes may lead to unacceptable errors in time synchronization after quite small time periods. Compensation for clock drift, which is therefore a necessary complement to time synchronization, is studied in Paper x in the appendix.

The clock drift is hardware-related and mainly caused by inaccuracy in the clock-crystal frequency. Small variations can occur due to for example power levels, temperature or voltage changes. A straightforward way to estimate the drift is to retrieve timestamps from two sensors at two different time points (separated by a sufficient time interval). If multiple values are retrieved linear regression can be used. Resynchronization or an algorithm to compensate the drift may then be applied. Drift compensation can be implemented either by multiplying the local node by the estimated drift parameter, or by continuously adjusting the node’s local time when necessary.

Clock drift compensation can be achieved by defining a virtual clock that runs with the estimated drift parameter d taken into account. We assume a resolution of 1ms for the local and the virtual clocks in the following. The sensor node clock with slowest clock (the lowest frequency) is used as a reference for the other sensor nodes drift compensation. When the node detects that the virtual clock differs from the local clock more than 1ms, the virtual clock time is subtracted by 1ms.

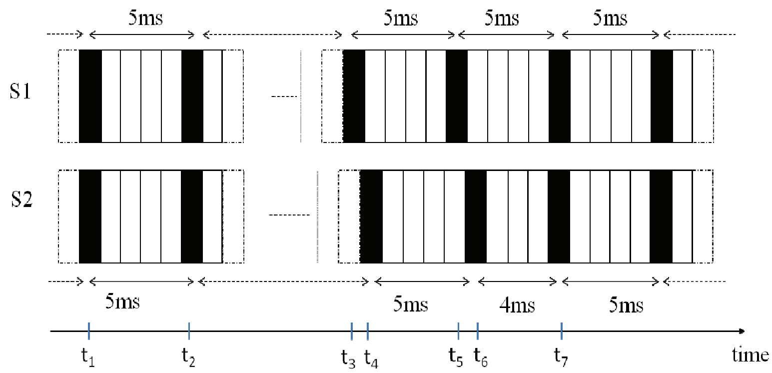

Figure 7. Shows two sensors sampling at 200Hz at the same time (t_1, t_2). Due to clock drift the sampling rate is shifted (t_3, t_4 and t_5, t_6). The drift compensation function decreases the inter-sample time with 1ms for S_2 at t_6 when it detects a drift of at least 1ms. At t_7 the sensors sample at the same time again.

Fig. 7 shows an example of how the drift compensation algorithm works. At t_1 and t_2 both sensor nodes, S_1 and S_2, are reading samples at exactly the same (synchronized) time with the sampling frequency f_s=200Hz (T_s=5ms). At t_3 the sample time for S_1 and S_2 are separated and partly overlapped at t_4. The time difference between the two local clocks at t_5 and t_6 has reached 1ms. The algorithm will now decrease the inter-sampling time T_s for S_2 by 1ms. The result is that the two sensors pick samples simultaneously at t_7. The algorithm will continuously monitor and control the clock drift.

3 Summary of original work

Paper I: NeTraWeb - A web-based traffic flow performance meter

M. Brenning, B. Olander, I. Orhan, J. Wennberg and T. Lindh.
Swedish National Computer Networking Workshop 2006, October 2006, Luleå

This paper presents a web-based traffic flow performance meter system. It combines an active and a passive method for data flow measurements by using so called traffic meters and monitoring packets. The measurement activities are configured, controlled and automated by a management server. The main traffic flow performance parameters such as packet loss, delays, delay variations and throughput, are measured and presented in NeTraMet. The tool can typically be used by end-users between hosts or by a network. Performance monitoring of packet delays, packet loss and throughput for applications such as voice and multimedia interactive services is the main target application of this system.

The system consists of two parts. Firstly, a server which manages and automates the measurement activities based on the information an operator provides via the web user interface. The server also presents the results from the measurements. The second part contains several Perl scripts to simplify the configuration and start-up of the system. The Perl script is also used for creating and sending monitoring packets at a user specified rate based on the actual data throughput of the measured traffic flow.

Contribution: The original idea came from the last author of the paper. The author of this thesis and the first author of the paper have contributed by creating the system that automates measurements of the round-trip time, delays, delay variations, packet loss and throughput of the defined traffic flows. The author of this thesis has also developed and implemented the feature that dynamically adapts the transmission of the monitoring packets.

Paper II: An end-to-end performance meter for applications in wireless body sensor networks

I. Orhan, A. Gonga, and T. Lindh.
International Workshop on Wearable and Implantable Body Sensor Networks, June 2008, Hong Kong.

This paper presents a prototype implementation of an end-to-end light-weight performance meter used for monitoring the transmission quality and to provide online transmission quality feedback to the sending node in wireless sensor networks. The performance meter consists of two counters implemented on the sensors nodes that keep track of the number of sent and received packets and bytes, and a function that can insert monitoring packets. The outcome from the performance meter is estimates of packet loss ratio, delay variation and throughput that can be used for feedback control and management.

Contribution: The author of this thesis participated in the system development and implemented the prototype and performed the testing and evaluation of the system independently. The programming language nesC has been used to build applications for the TinyOS platform.

Paper III: Performance monitoring and control in contention-based wireless sensor networks

T. Lindh and I. Orhan.
International Symposium on Wireless Communication Systems, September 2009, Siena, Italy.

This paper extends Paper II above ("An End-to-End Performance Meter for Applications in WSN") to support performance control based on measurement information from the performance meter. The performance control system minimizes information loss and maintains the desired throughput, based on information about packet loss and throughput gathered from the performance meter. The output of the control algorithm, to decrease or increase the packet frequency, is based on performance data. . Different test scenarios are presented in the paper. The first scenario shows a control scheme with priority which tries to maintain throughput and minimize packet loss for a node with high priority, by punishing nodes with low priority. In the second scenario, where all nodes have the same priority, each node tries to maximize its throughput under the condition that the loss ratio is below a threshold. The last scenario is a combination of the two previous ones. From the beginning both nodes have the same priority. After a certain time, one of the nodes is assigned high priority and higher expected throughput.

Contribution: The original idea came from the first author. The author of this thesis have contributed to the development of the feedback control algorithms, implemented the prototype and the evaluations.

Paper IV: Measurement-based admission control in wireless sensor networks

I. Orhan and T. Lindh.
The Third International Conference on Sensor Technologies and Applications, Sensorcomm 2010, July 2010, Venice. The paper received a best paper award.

In this paper we present a method for admission control in contention-based networks, implemented as a component of a performance management system. The admission control system bases its decision, to accept or reject a request to join the network, on measurements of performance parameters, mainly the packet loss ratio. The decision is done to protect the performance of already admitted nodes. The performance control function, presented in Paper III above, is responsible for maintaining the desired quality of service once the sensors are allowed to use the wireless channel. A typical application scenario is healthcare at-home with ECGs, pulse-oximeters and accelerometers connected to a coordinator. The impact of traffic pattern on packet loss has also been studied.

Contribution: The author of this thesis has developed the admission control system in cooperation with the other author and been responsible for the prototype implementation and testing. This includes testing of the impact of different traffic pattern on packet loss. The paper was written in cooperation with the other author.

Paper V: Local time synchronization in Bluetooth piconets for data fusion using mobile phones

J. Wåhslén, I. Orhan and T. Lindh.
Body Sensor Networks (BSN) 2011 International Conference, May 2011, Dallas, US.

This paper presents a method to synchronize the clocks between the master and the slaves in a Bluetooth piconet in order to perform data fusion of data from multiple sensors with correct timestamps. Performance issues that cause problems for data synchronization between master and slaves in Bluetooth are highlighted.

Contribution: The main contribution of the author of this thesis is the performance experiments and analysis of time synchronization in Bluetooth communication between sensors and mobile phones. The paper was written in cooperation with the co-authors.

Paper VI: Performance evaluation of time synchronization and clock drift compensation in wireless personal area networks

J. Wåhslén, I Orhan, D. Sturm and T. Lindh.
Submitted to 7th International Conference on Body Area Networks, September 2012 Oslo, Norway

This paper presents the result of the evaluations of the performance of algorithms for time synchronization based on three different approaches; one that synchronizes the local clocks on the sensor nodes, and a second that uses a single clock on the receiving node, e.g. a mobile phone and a third that uses broadcast messages. The performances of the synchronization algorithms have been evaluated in wireless personal area networks using Bluetooth piconets and ZigBee/IEEE 802.15.4 networks. A real time implementation of single node synchronization from the mobile phone has been presented and tested. In addition, a new technique to compensate for clock drift to preserve accuracy in data fusion has been developed, implemented and tested.

Contribution: The author of this thesis has presented a method to detect and measure the clock drift of the wireless sensor nodes. A real time drift compensating algorithm has been developed and evaluated in a testbed. The author have also implemented and evaluated the performance of the last two presented time synchronization algorithms in the paper. The paper was written in cooperation with the co-authors.

Journal paper: Measurement-based performance and admission control in wireless sensor networks

I. Orhan and T. Lindh.
International Journal on Advances in Systems and Measurements, SysMea11v4n12, Volume 4, Numbers 1 & 2, pp 32-45, 2011

This invited journal paper summarizes the contributions of Paper II, III and IV above ("An End-to-End Performance Meter for Applications in Wireless Body Sensor Networks", "Performance Monitoring and Control in Contention-Based Wireless Sensor Networks" and "Measurement-Based Admission Control in Wireless Sensor Networks"). The journal paper presents a measurement-based performance management system for contention-based wireless sensor networks. Its main features are admission and performance control based on measurement data from lightweight performance meters in the endpoints.

Contribution: The journal paper is written in cooperation with the other author based on previous work.

Other event: Contention-based wireless sensor networks - a case study for ambient assisted living

J. Burmeister (Dipl.-Ing., Dresden, internship student at KTH, spring 2010), T. Lindh, I. Orhan, L-Å Brodin, S. Lundberg.
Ambient Assisted Living forum 2010, September 2010, Odense, Denmark

This project was done in collaboration with the Centre for Health & Building at KTH. The project focuses on continuous monitoring of the heart activity using a wireless ECG based on the wireless personal area network standard IEEE 802.15.4 but also performance problems in wireless sensor network. We have used wireless sensor nodes in the living lab at the Centre for Health and Building at KTH in a case study to identify and evaluate performance problems. Results from performance tests in the living lab are presented e.g. influence of equipment such as micro wave ovens.

Since contention-based wireless access has no guarantees for the quality of the delivered service it is interesting to determine to what extent the received ECG signal is sensitive to loss of information. ECG records from available reference databases [21] have been exposed to different levels and patterns of packet loss. These records have been analyzed by cardiologists and the result is reported in the paper.

Contribution: The author of this thesis participated in the work to identify and evaluate performance problems in wireless sensor networks. The author has also presented this paper at the forum.

4 Conclusion and future work

In this thesis we have presented a method and a prototype implementation of a performance management system. The control and admission functions are based on measurement data from light-weight performance meters in the endpoints. The test result shows that the implemented performance manager improves the quality and predictability of demanding communication services in contention-based networks. The system can also be used as a tool to dimension and configure services in wireless sensor networks. The thesis also analyses errors in time synchronization due to Bluetooth performance. A new method and a prototype implementation to measure and compensate the clock drift between multiple wireless sensors in real time are presented. The algorithm is implemented and evaluated in an IEEE 802.15.4 compliant sensor node testbed. The prototype implementations of the performance manager and the method for real time clock drift compensation show promising results. However, these are still several open questions, which can be subject for further research.

- Further improvement of the performance meter regarding energy consumption and obtained performance parameter.
- Improvement of the responsiveness of the performance control algorithms.
- Evaluation of other control parameters than varying the packet frequency.
- Further improvement of the algorithms for admission control.
- Develop new methods to automatically configure and tune the system given certain rules.
- Continue and extend the study the loss sensitivity of ECG signals to other medical and healthcare application.
- Further improvement of the algorithm for compensating the clock drift.

References

[1] Interconnecting Smart Objects with IP - The Next Internet, Jean-Philippe Vasseur and Adam Dunkels, Morgan Kaufmann, 2010.

[2] Transmission of IPv6 Packets over IEEE 802.15.4 Networks, RFC 4944.

[3] T. Lindh and N. Brownlee: "Integrating Active Methods and Flow Meters - an implementation using NeTraMet", Passive and Active Measurement workshop (PAM2003), San Diego, April 2003.

[4] M. Brenning, B. Olander, I. Orhan, J. Wennberg, Thomas Lindh: "NeTraWeb: a Web-Based Traffic Flow Performance Meter", Swedish National Computer Networking Workshop 2006, Luleå, Sweden, 2006.

[5] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function", IEEE JSAC, Volume 18, No 3, pp 535 – 547, March 2000.

[6] Hai L. Vu, "Collision Probability in Saturated IEEE 802.11 Networks", Australian Telecommunication Networks and Applications Conference, pp. 21-25, Australia, December 2006.

[7] K. Duffy, D. Malone, and D.J. Leith, "Modeling the 802.11 Distributed Coordination Function in Non-saturated Conditions", Communications Letters, IEEE, Volume 9, Issue 8, pp. 715–717, August 2005.

[8] F. Österlind and A. Dunkels, "Approaching the Maximum 802.15.4 Multi-hop Throughput", HotEmnets, Virginia, June 2008.

[9] D. Cavalcanti et al, " Performance Analysis of 802.15.4 and 802.11e for Body Sensor Network Applications", BSN 2007, Aachen, March 2007.

[10] N. Golmie et al: "Performance analysis of low rate wireless technologies for medical applications" Computer Communications, Volume 28, Issue 10, pp 1266-1275, June 2005.

[11] C.Y. Wan, S.B. Eisenman, and A.T. Campbell, "CODA: congestion detection and avoidance in sensor networks", SenSys, 1st conference on embedded networked sensor systems, pp 266-279, Los Angeles, November 2003.

[12] J. Kim, S. Kim, S, Choi, and D. Qiao, "CARA: Collision-Aware Rate Adaptation for IEEE 802.11 WLANs", INFOCOM, pp 1-11, Barcelona, April 2006.

[13] P. Ramanathan, D. Moore, and C. Dovrolis "What Do Packet Dispersion Techniques Measure", In Proceedings of IEEE INFOCOM, 2001, pp. 905-914, Anchorage, Alaska, USA, April 2001.

[14] M. Jain and C. Dovrolis, "Pathload: a measurement tool for end-to-end available bandwidth", Passive and Active Measurements Workshop, pp 14-25, Fort Collins, USA, March 2002.

[15] S. Ekelin, M. Nilsson, E. Hartikainen, A. Johnsson, J-E Mångs, B. Melander, and M. Björkman, "Real-Time Measurement of End-to-End Available Bandwidth using Kalman Filtering", IEEE NOMS, pp 73-84, Vancouver, Canada, April 2006.

[16] T. Sun, L. Chen, G. Yang, M. Y. Sanadidi, and M. Gerla, "SenProbe: Path Capacity Estimation in Wireless Sensor Networks" SenMetrics 2005, San Diego, USA, July 2005.

[17] D. Gupta, D. Wu, P. Mohapatra, and C-N. Chuah, "Experimental Comparison of Bandwidth Estimation Tools for Wireless Mesh Networks", IEEE INFOCOM Mini-Conference, pp 2891-2895, April 2009.

[18] Y. Yang and R Kravets, "Contention-Aware Admission Control for Ad Hoc Networks" Mobile Computing, IEEE Transactions on Volume 4, Issue 4, pp 363 - 377, July-August, 2005.

[19] Ian D. Chakeres and Elizabeth M. Belding-Royer, "PAC: Perceptive Admission Control for Mobile Wireless Networks", International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine), pp 18-26, Dallas, USA, October 2004.

[20] D. Gay, P. Lewis, R. von Behren, M. Welsh, E. Brewer, and D. Culler, "The nesC language: A holistic approach to networked embedded systems", Proceedings of the ACM SIGPLAN 2003 conference on Programming language design and implementation, pp 1-11, San Diego, USA, June 2003.

[21] Goldberger AL, Amaral LAN, Glass L, Hausdorff JM, Ivanov PCh, Mark RG, Mietus JE, Moody GB, Peng CK, Stanley HE. PhysioBank, Phy-sioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. *Circulation* 101(23):e215-e220 [Circulation Electronic Pages; http://circ.ahajournals.org/cgi/content/full/101/23/e215; 2000.]

List of papers and journals

Paper I
M. Brenning, B. Olander, I. Orhan, J. Wennberg and T. Lindh, “NeTraWeb - A Web-Based Traffic Flow Performance Meter”, Swedish National Computer Networking Workshop 2006, October 2006, Luleå, Sweden.

Paper II
I. Orhan, A. Gonga, and T. Lindh, "An End-to-End Performance Meter for applications in Wireless Body Sensor Networks", International Workshop on Wearable and implantable Body Sensor Networks, June 2008, Hong Kong.

Paper III
T. Lindh and I. Orhan, "Performance Monitoring and Control in Contention-Based Wireless Sensor Networks", International Symposium on Wireless Communication Systems, September 2009, Siena, Italy.

Paper IV
I. Orhan and T. Lindh, "Measurement-Based Admission Control in Wireless Sensor Networks", Sensorcomm 2010, July 2010, Venice, Italy. The paper received a best paper award.

Paper V
J. Wåhslén, I. Orhan and T. Lindh, "Local Time Synchronization in Bluetooth Piconets for Data Fusion Using Mobile Phones", BSN 2011, May 2011, Dallas.

Paper VI
J. Wåhslén, I Orhan, D. Sturm and T. Lindh, ”Performance Evaluation of Time Synchronization and Clock Drift Compensation in Wireless Personal Area Networks”, submitted to 7th International Conference on Body Area Networks, September 2012, Oslo, Norway.

Journal paper
I. Orhan and T. Lindh, "Measurement-Based Performance and Admission Control in Wireless Sensor Networks", "SysMea11v4n12, International Journal On Advances in Systems and Measurements", Volume 4, Numbers 1 & 2, pp 32-45, 2011.

Other event
J. Burmeister, T. Lindh, I.Orhan, L-Å. Brodin and S. Lundberg, “Contention-Based Wireless Sensor Networks – A case study for ambient assisted living”, the project was presented in WSN for Ambient Assisted Living collaboration with Centrum för Hälsa och Byggande (CHB) at KTH at the Ambient Assisted Living forum. September 2010, Odense, Denmark.

Other scientific contributions not included in this thesis

Scientific paper publications

T. Lindh, I. Orhan A. Gonga, “A Performance Monitoring Method for Wireless Sensor Networks”, The 1st International Conference on PErvasive Technologies Related to Assistive Environments, July 2008, Athens.

T. Lindh and I. Orhan, "Performance Control in Wireless Sensor Networks", International Workshop on Wireless Pervasive Healthcare, March 2009, London, UK.

I. Orhan and T. Lindh, " Performance Monitoring and Control in Contention-Based Wireless Sensor Networks ", Swedish National Computer Networking Workshop 2009, May 2009, Uppsala, Sweden.

Scientific journal paper publication

J. Wåhslén, I. Orhan, T. Lindh and M. Eriksson “A Novel approach to multi-sensor data synchronisation using mobile phones“, International Journal of Autonomous and Adaptive Communications Systems, Int. J. Autonomous and Adaptive Communications Systems, Vol. 6, No. 3, pp 289-303, in print, 2013

Other event

I. Orhan and T. Lindh. A showcase/demostration was hold on the conference Body Sensor Networks 2009 with the title "Real-time performance monitoring and control in wireless sensor networks" at San Francisco.

NeTraWeb - A Web-Based Traffic Flow Performance Meter

M. Brenning, B. Olander, I. Orhan, J. Wennberg and T. Lindh
6th Swedish National Computer Networking Workshop
October 2006, Luleå

NeTraWeb

- A Web-Based Traffic Flow Performance Meter

Magnus Brenning[1], Björn Olander[1], Ibrahim Orhan[1], Johan Wennberg[1], Thomas Lindh[1, 2]
[1]*School of Technology and Health,* [2]*Laboratory for communication networks, School of Electrical Engineering KTH, Stockholm.*

Abstract

This paper presents a web-based traffic flow performance meter. The NeTraWeb tool configures and automates the measurement activities, including storage and presentation of the main performance parameters.

1. Introduction

The tool presented in this paper, NeTraWeb, is a web-based traffic flow performance meter system. The measurement activities are configured, controlled and automated by a management server. The main traffic flow performance parameters e.g. packet loss, delays, delay variations and throughput, are presented in graphs and text. The name NeTraWeb is inherited from an extended version of the flow meter NeTraMet ([1], [2]).

The tool can typically be used by end users between hosts or by a network operator between measurement points in the network. Performance monitoring of delay and loss sensitive applications such as voice and multimedia interactive services between the end users or other measurements points in the network is the main application of the tool. The meter agents have also been used in a prototype implementation for monitoring of SIP-based communication, where the approach is to activate performance measurements between clients based on signalling messages to a SIP server [3].

Figure 1 shows a NeTraWeb server and meter agents in two access nodes. The meters could also be deployed in end user hosts or other points in the network where traffic can be observed, e.g. VPN access points or edge nodes.

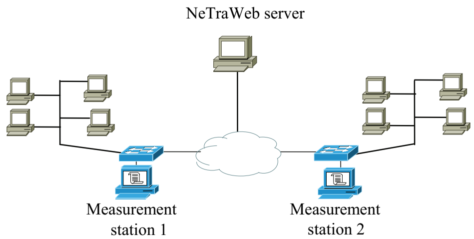

Figure 1. A NeTraWeb server and meter agents at two access points.

The method used by NeTraWeb is explained in Section 2. The functions and structure of the tool are described in Section 3, followed by some measurement results in Section 4 and the conclusions in Section 5.

2. Traffic Meters and Monitoring Packets

2.1. Combining active and passive methods

NeTraWeb uses light-weight traffic meters at the measurement points. An extended version of the traffic flow meter NeTraMet combines active and passive methods [1]. A traditional flow meter counts bytes and packets at a single point. However, to enable packet loss and delay metrics, NeTraWeb uses meters at two points that count the number of packets and bytes for the defined flows observed by the meters.

Monitoring packets are sent between the meters with periodic or random time intervals inserted between block of user data packets (Figure 2). These packets can also be transmitted with a frequency that is adapted to the actual data rate of the monitored traffic flow, and thereby maintaining an approximately constant size of the monitoring block (Figure 2). When a meter detects a monitoring packet a timestamp and the current cumulative counter values of packets and bytes are stored in a flow file by each meter.

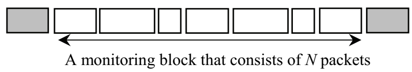

Figure 2. Two monitoring packets enclose a monitoring block that consists of N user packets on average.

The precision, accuracy and resolution of the metrics are determined by the size of the monitoring block, which is the distance between two consecutive monitoring packets (expressed in time intervals or in number of data packets). This is the main parameter of the monitoring method used to adjust the accuracy of the results.

The meter is controlled by a set of rules, using the simple ruleset language (srl), which define the flows and their attributes. A special companion program, NeMaC [4], manages the meter locally. The resulting flow files are then sent to the NeTraWeb server using an automated ssh function in Perl (Section 3).

The modifications to NeTraMet are included in the NeTraMet distribution from version 4.5b8 and later on. We have used version 5.1b11, in which round-trip time as well as one-way delays can be measured [3].

2.2. Measurement results

The method gives the following results per traffic flow (defined as a unique combination of destination and source addresses, destination and source ports, transport protocol and possibly other parameters).

Packet delays and their variation are estimated based on sampling. One-way delay can be measured if the meters have synchronized clocks; otherwise *round-trip delay* and its variation are measured. *Inter-arrival jitter* in both directions can reveal possible asymmetry in delays. The four timestamps used in the round-trip delay estimates are shown in Figure 3.

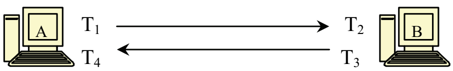

Figure 3. The four timestamps recorded for the monitoring packets.

Round-trip delay is here defined as $T_4–T_1–(T_3–T_2)$ and the inter-arrival jitter (at node B in Figure 3) as $T_2(n+1)–T_2(n)–(T_1(n+1)–T_1(n))$. The definition of inter-arrival jitter is in agreement with RTP/RTCP Internet standards used VoIP measurements [5]. In this case the monitoring packet client in node A sends a packet (of same type and size as the data flow) to node B, which returns the packet to node A. These metrics do not require synchronized clocks. Since GPS synchronization is expensive and sometimes difficult to install, round-trip delay and inter-arrival jitter estimates are an alternative.

The monitoring packets have a dual role. Firstly to provide samples of delays based on the timestamps in Figure 3, and secondly to trigger intermediate readings of the meters, and store these cumulative counter values of packets and bytes for the defined flows.

Packet loss is measured directly. Beside the long-term averages for the entire measurement periods, the distribution of losses per monitoring block is obtained. The *length of loss periods and loss-free periods* can be computed.

Throughput is measured directly. The utilized capacity for the entire measurement period and per monitoring block is obtained.

3. The NeTraWeb System

The system consists of the NeTraWeb server and two meter agents that reside on existing hosts or on dedicated measurement nodes. The *server* manages and automates the measurement activities based on the information an operator provides via the web user interface.

- Attributes of the flow to be monitored.
  The flow is specified using a proper combination of the five attributes: source and destination IP addresses, source and destination ports and layer-4 protocol. The granularity of the flow and sub flows is determined by the number of attributes and their combination.
- The length of the measurement period.
- The rate of the monitoring packets generated between the meter agents: constant time intervals between the packets or a rate that is adapted to the data rate (throughput) of the monitored flows. Monitoring packets generated based on the actual data flow rate can be seen as an adaptive sampling design. This is done by polling the meter MIB for information on the number of user data packets sent in the defined flow [6]. Further issues on systematic and random sampling are discussed in [7].

The *meter agent* is implemented in a Perl program that communicates with the NeTraWeb server on the one hand, and with the local NeTraMet meter and NeMAC on the other hand. The NeTraMet meter uses libpcap to access the network interface.

Based on the information above NeTraWeb automates the measurement activities. This communication scheme between the server and the meter agents is described below.

3.1. The automated measurement activities

The NeTraWeb server creates a configuration file based on user input and sends it to the meter agent in station 1 (Figure 1), using an automated ssh module in Perl. The meter agent in station 1 uses the configuration file to start the NeTraMet meter, the meter manager NeMaC and the monitoring packet generator. Thereafter station 1 sends the configuration file to station 2, which repeats the same procedure as station 1, and sends an acknowledgement to station 1. Both meters and the monitoring packet generators are now active. When the measurement period has ended the meters and monitoring packet generators terminate, and the two resulting flow files are sent to the NeTraWeb server. Further details can be found in [6].

The flow file from meter 1 contains a list of measurement data store by the meter when a monitoring packet is detected: the monitoring packet ID, the current timestamp t (T_1 in Figure 3), the cumulative number of packets and bytes sent and received at time t, and the corresponding measurement data when the monitoring packet returns (T_4 in Figure 3). The flow file from meter 2 consists of measurement data when timestamps T_2 and T_3 are recorded.

The flow files are parsed and stored in different tables in a Postgres DBMS (DataBase Management System). This is done with efficiency and scalability in mind. The DBMS makes it easier to handle multiple simultaneous users and presentations.

The parser is written in ANSI-C with the ecpg library for PostgreSQL support. This allows SQL syntax inside the C code. All web material is written in Hyper Text Markup Language, PHP (PHP: Hypertext Preprocessor) and a few Javascripts.

When the flow files have been imported from the meters to the NeTraWeb system, a user creates a new project and can choose among presentations of the following groups of performance parameters: packet loss , packet round-trip delays, round-trip delay variation, inter-arrival jitter and throughput. It is also possible to import other existing flow files from other sources to NeTraWeb for presentation.

There are two types of graphs; regular time plots and histograms. NeTraWeb uses the Chart PHP4 library to generate graphs. It is covered by GNU GPL (General Public License). The graphs are generated as .PNG pictures. One can change the number of bars and the space between the bars in all histograms. Beside the graphs the most common statistical metric are calculated.

Since the two meters do not start at exactly the same time the flow files have to be synchronized. The useful common part of the flow files are picked out using SQL queries.

The processing and presentation of performance parameter metrics are implemented as SQL queries. This flexible approach means that users can add new metrics, graphs and histograms, or remove existing ones. Further details are found in [8].

4. Measurement Results

For each measurement project NeTraWeb generates more than twenty different graphs. As an illustration some graphs from different measurements are shown in the Figure 4-10.

Some results from a VoIP call between two wireless clients in a local area network are shown in Figure 4-8. The average round-trip delay is 0.62 ms, the maximum value 1.18 ms, the minimum value 0.2 ms, the 99th percentile 1.11 ms, the 97.5th percentile 1.06 ms, the 10th percentile 0.35 ms and the standard deviation 0.22 ms. Figure 4 and Figure 5 show the time plot and distribution of round-trip delays.

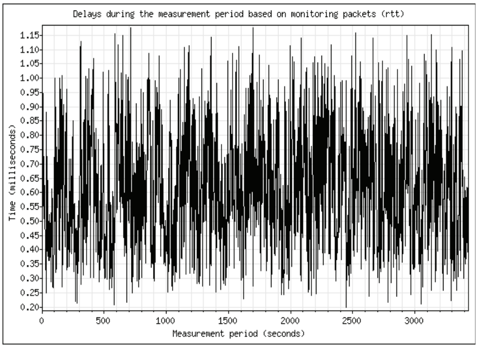

Figure 4. Round-trip delays (in milliseconds) for each monitoring packet during the measurement period.

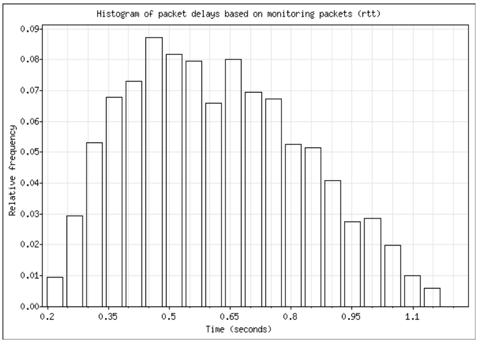

Figure 5. Histogram of the round-trip packet delays in Figure 4.

Figure 6 shows the packet loss ratio per monitoring block. The length of loss-free periods and the corresponding histogram can be seen in Figure 7 and Figure 8. The total loss ratio in this measurement is 0.014.

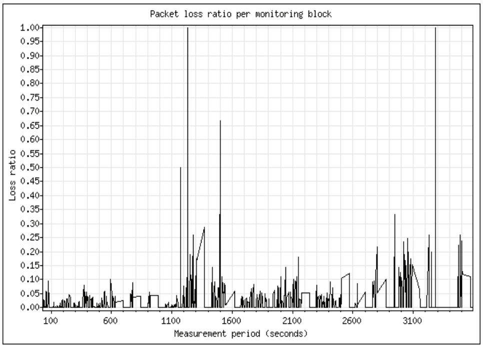

Figure 6. Packet loss ratio per monitoring block during the measurement period.

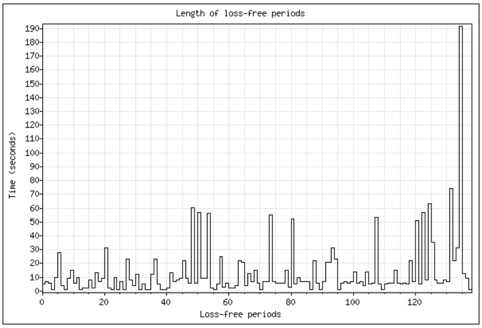

Figure 7. The length of loss-free periods in seconds.

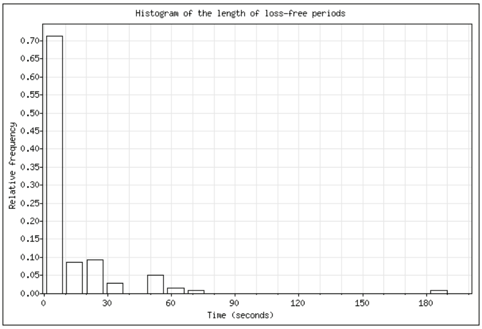

Figure 8. A histogram of the length of loss-free periods shown in Figure 7.

In addition to the graphs presented above, NeTraWeb creates graphs and histograms that cover: round-trip jitter (according to IETF and ITU-T), packet loss (in number of bytes, packets and loss ratio) per monitoring block or in time, the number of losses in loss periods, the length of loss and loss-free periods and throughput.

Inter-arrival jitter from a measurement between VoIP clients located at KTH in Haninge and BTH in Karlskrona can be seen in Figure 9 and Figure 10. The inter-arrival jitter is not symmetric. The variation in arrival times to client B is higher than in the opposite direction. Hence, asymmetric delay variations can be detected without having synchronized clocks using the four timestamps in Figure 3.

5. Conclusions

This paper has presented an open source web-based traffic flow performance meter to be used by end users or network operators. The NeTraWeb system uses a communication procedure between the server and the agents in order to automate the measurement activities. The results are stored in a database. The processing and presentation of performance parameter metrics are implemented as SQL queries. This flexible approach means that it is easy to develop this tool further and adapt it and the users' requirements. A detailed evaluation of the tool will be part of future work.

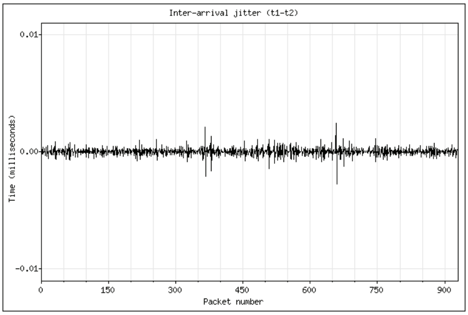

Figure 9. Inter-arrival jitter in the direction from client A to client B. The maximum value is 2.5 ms and the standard deviation is 0.36.

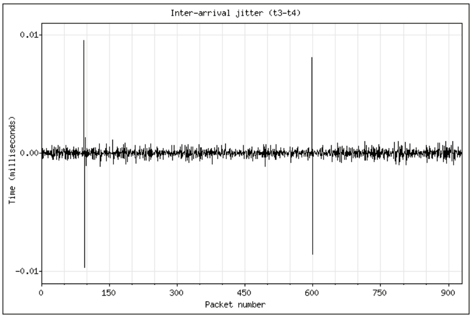

Figure 10. Inter-arrival jitter in the direction from client B to client A. The maximum value is 10.5 ms and the standard deviation is 0.83.

6. References

[1] T. Lindh and N. Brownlee: "Integrating Active Methods and Flow Meters - an implementation using NeTraMet", Passive and Active Measurement workshop (PAM2003), San Diego, April 2003.

[2] NeTraMet - a Network Traffic Flow Measurement Tool , http://www.caida.org/tools/measurement/netramet/

[3] NeTraMet beta versions: ftp://ftp.auckland.ac.nz/pub/iawg/NeTraMet/beta-versions/

[4] NeTraMet & NeMaC Reference Manual v4.3, http://www2.auckland.ac.nz/net/Accounting/ntmref.pdf

[5] RTP: A Transport Protocol for Real-Time Applications, RFC 3550, July 2003.

[6] M. Brenning, I. Orhan: "Performance measurements in IP networks: automation and rate adapted sampling", Master Thesis, Master program Computer networks, School of Technology and Health, KTH, Stockholm, September 2006.

[7] T. Lindh: "Systematic sampling and cluster sampling of packet delays", Passive and Active Measurement workshop (PAM2006), Adelaide, March 2006.

[8] B. Olander, J. Wennberg: "NeTraWeb Management System for Measurements of Performance Parameters", Master Thesis, Master program Computer networks, School of Technology and Health, KTH, Stockholm, September 2006.

An End-to-End Performance Meter for Applications in Wireless Body Sensor Networks

I. Orhan, A. Gonga, and T. Lindh
International Workshop on Wearable and Implantable Body Sensor Networks
June 2008, Hong Kong.

An End-End Performance Meter for Applications in Wireless Body Sensor Networks

Ibrahim Orhan, António Gonga, Thomas Lindh, *Royal Institute of Technology (KTH), Sweden*

***Abstract*—This paper presents a monitoring method and its implementation as a light-weight end-to-end performance meter for quality-demanding applications in wireless body sensor networks. The method is evaluated in a wireless sensor network testbed for healthcare applications.**

I. INTRODUCTION

Wireless sensor networks are rapidly becoming a common infrastructure for exchange of medical and healthcare information. It is crucial that estimates of medical and health-related parameters, transmitted via wireless sensor networks, can be trusted. Correct information about the network performance parameters is therefore important in order to estimate the uncertainty of the measured vital sign parameters. This paper presents a measurement method that provides online transmission quality feedback to the applications. Performance measurements and monitoring in body sensor networks have been studied previously, e.g. in [7] and [8].

II. AN END-TO-END MONITORING METHOD

A. Design Principles

The method is based on the following underlying principles and goals.

- The monitoring is performed end-to-end between the nodes where the applications run.
- A demanding application should be able to include the monitoring function as an add-on service.
- The method should be able be implement on different link layer technologies.
- Minimizing extra energy consumption is important for wireless sensor networks.
- A combination of active and passive measurement methods is applied.

B. Application

Some typical applications that can use the proposed monitoring method are: in-home care, special housing and assistant living, hospital care, wellness and disease monitoring before and after hospital care, emergency care, fitness, sports and athlete training.

Today, there exist a number of different sensors that can be attached to fixed or mobile wireless nodes, e.g. pulse-oximeters, ECGs, EEGs, blood pressure meters, glucose meters, thermometers, EMGs, bio-chemical sensors, accelerometers, gyroscopes and magnetometers, microphones, cameras, IR detectors, RFID tags etc.

The method has been implemented in nesC and Java for an IEEE 802.15.4 low-power platform. Healthcare applications running in cell phones or PDAs communicating with wireless sensor nodes on the one hand, and back-end servers on the other hand, can also use the proposed method.

C. The Performance Meter

The approach is to combine active and passive techniques, inspired by the results from measurements in wireline networks in [1] and [2]. A light-weight performance meter is implemented in each node. The meter consists of two counters that keep track of the number of sent and received packets and bytes, and a function that can insert monitoring packets. These dedicated measurement packets are inserted between blocks of ordinary data packets as seen in Fig. 1. They contain a sequence number, a timestamp and the cumulative number of packets and bytes transmitted from the sending node to the receiving node.

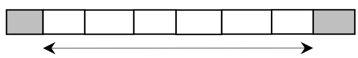

Fig. 1. A monitoring block surrounded by two monitoring packets.

The interval between the monitoring packets, i.e. the size of the monitoring block, can be expressed in number of packets or a time interval, constant or varying randomly around a mean value. A different approach would be to add the monitoring information to samples of the ordinary data packets, which however requires a type indicator in the packet header.

When a monitoring packet arrives, the received node stores a timestamp and the current cumulative counter values of the number of received packets and bytes from the sending node. Observe that for k number of sending nodes, the receiving node maintains k number of separate monitoring functions, one for each sending node.

Synchronization of the clocks in the participating nodes is not required. The local timestamps are used to calculate the

Ibrahim Orhan has a Master's degree in Computer Networks and is a PhD student at the School of Technology and Health at the Royal Institute of Technology (KTH) in Stockholm, Sweden.

António Gonga is student at the Master's program Network Services and Systems at the School of Electrical Engineering, at the Royal Institute of Technology (KTH) in Stockholm, Sweden. He has a Master's degree in Electrical and Computer Engineering at the Technical University in Lisboa, Portugal.

Thomas Lindh has a PhD. He is senior lecturer and researcher at the School of Technology and Health at the Royal Institute of Technology (KTH) in Stockholm, Sweden.

inter-sending and inter-receiving times between pairs of monitoring packets. The inter-arrival jitter can then be calculated in a similar way as for RTP timestamps [3]. This means that arrival time variation is estimated based on the monitoring packets, which represent samples of the data packets' inter-arrival variation. Packet loss, on the other hand, is measured passively and directly using the counters.

D. Performance Metrics

The following metrics can be calculated and estimated based on the collected measurements described in the previous subsection (and in more detail in Section III.B).

- Packet loss ratio: long-term and per monitoring block.
- The length of loss and loss-free periods defined as the number of consecutive monitoring blocks with or without losses. Can be expressed in time unit, number of blocks, and number of packets and bytes.
- Inter-arrival jitter, J, *is* defined as $J = (r_n\text{-}r_{n-1})\text{-}(s_n\text{-}s_{n-1})$, where s is the sending time and r is the receiving time. The monitoring packets provide samples of this delay variation metric, which means that the uncertainty of the estimated statistics (mean value, median, percentiles etc) is determined by the number of samples and the variance of the delay variation process.
- Throughput between sender and receiver: long-term average and also per monitoring block. The resolution of the peak rate is determined by the ratio between monitoring packets and ordinary data packets. This can also be seen as a measure of utilized capacity.

E. Feedback Information for Control and Management

A management and control system needs the performance meter results for several reasons. One obvious application is to provide statistics for analysis and dimensioning of sensor networks. Another purpose is feedback information for systems management, but also for real-time control. A coordinator or actor node [4] may be responsible for control actions based upon the feedback information, and also to notify and communicate with a central back-end system. The performance meter enables an actor node to determine the effect of the network performance on the statistical uncertainty of the measured medical parameters. To close the loop between feedback information and control actions (e.g. varying the radio power) is perhaps the most important application of the monitoring method.

III. MEASUREMENTS

A. The Testbed

The testbed consists of three different sensors attached to three Tmote Sky motes [5]. Another Tmote Sky mote acts as a base station, connected to a laptop (Fig. 2). The sampled data from the sensor nodes are sent to the base station.

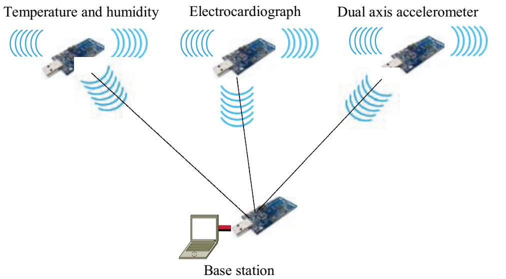

Fig. 2. The measurement testbed with three sensor nodes and a base station.

1) The Temperature and Humidity Sensor

The onboard temperature and humidity sensors are sampled twice a second and the collected samples are sent immediately to the base station. Monitoring packets are inserted between blocks of 100 data packets (each packet is 6 byte).

2) The ECG Sensor

The mote samples the ADC-12 (analog-to-digital converters, 12 bits resolution) interface connected to an ECG at 200Hz. The sampling lasts for 5 seconds, the radio previously switched off, is turned on, and the collected samples are sent to the base station. After sending the data, the procedure is repeated. In total 13 samples of a 16-bit integer is sent in every packet. The idea is to have the radio turned off as long as possible and sending many samples in one packet in order to minimize the power consuming. The node sends 77 packets every five second. A monitoring packet is inserted between blocks of 75 ordinary data packets.

3) The Dual-Axis Accelerometer Sensor

A multi-sensor board (SBT80 from Easysen [6]) is connected to the Tmote Sky mote. It has eight different sensors: visual light, infrared, acoustic, temperature, dual-axis magnetometer and dual-axis accelerometer. The dual-axis accelerometer sensor is connected to two ADCs, one ADC to each axis. The node samples the accelerometer sensor at 100Hz during one second and sends all samples to the base station. The radio is only turned on when sending the samples. Each packet carries 10 samples of 16 bits from the respective axis. The mote sends 20 packets every second. A monitoring packet is sent after every 100 data packets.

4) The Base Station

The base station receives data packets with samples from the three sensors interleaved by monitoring packets. Three separate counters keep track of the number of packets and bytes received from each sensor node. Every monitoring packet is timestamped immediately when it arrives. This receiver information and the monitoring packet data are sent to the laptop and stored in a file to be analyzed when requested.

B. The Meter and Monitoring Packet Implementation

The performance meter is programmed in nesC for TinyOS 2.0. The sensor nodes read samples from the sensors (ECG, accelerometer and temp/humidity), assemble the samples and send them in packets to the base station. The number of bytes and packets are counted. The cumulative number of bytes and packet and a timestamp are inserted into a monitoring packet, which is sent after every *n* ordinary data packet.

A monitoring packet is 17 bytes long and includes the following fields: a start flag, a timestamp when packet is sent, type, a sequence number, number of packets sent, number of bytes sent, and a stop flag. The flags enable the base station to distinguish a monitoring packet from ordinary data packets. The sequence number identifies and keeps track of monitoring packets. The packet and byte fields contain the cumulative number of bytes and packets sent. Finally, the type field enables several sensor data flows from the same node.

Each time the base station receives a data packet, it updates the number of bytes and packets for each sensor. The base station uses the source field in the CC2420 radio header to distinguish the packets from different sources. As soon as the base station receives a monitoring packet, it stores a timestamp and the updated counter values of the number and bytes and packets received from the specific sensor mote. A Java application on the laptop processes the data and creates two files for each sensor. The file with data for the sending node (left in Fig. 3) contains the same information as carried in the monitoring packet: the sequence number, the timestamp, the total number of bytes sent, and the total number of packets sent. The file with the receiving node data contains the corresponding information at the base station side (right side in Fig. 3).

```
70 623766 312400 14200
71 632514 316800 14400
72 641339 321200 14600
73 650109 325600 14800
74 658902 330000 15000
```

Sender

```
70 629597 312224 14192
71 638347 316624 14392
72 647176 321024 14592
73 655949 325424 14792
74 664739 329824 14992
```

Receiver

Fig. 3. Measurement data from the sender and the receiver nodes. Columns from left to right: sequence number, timestamp (ms), cumulative number of bytes and packets.

IV. Results

Table 1-3 and Fig. 4-9 illustrate some results that can be obtained using the monitoring method in wireless sensor networks. The measurement period in the testbed (Fig. 2) was around 80 minutes. The data packet size was maximum 28 bytes. The size of the monitoring block was approximately 100 packets or 75 packets. The sensor sampling rate and the resulting packet rate for the different sensors are described in the previous section. Table 1 shows the throughput (bits per second) between the respective sensors and the base station node. The throughput per monitoring block during the measurement period can be seen in Fig. 4, and the corresponding empirical probability distribution function in Fig. 5. Three distinct events of decreased throughput due to disturbances can be observed.

TABLE 1: THROUGHPUT BETWEEN SENSOR NODES AND BASE STATION

Throughput	Acc–Base stn	ECG–Base stn	Temp-Base stn
Mean value	3.89 kb/s	3.86 kb/s	92 b/s
Maximum	4.03 kb/s	22.3 kb/s	93 b/s
Minimum	2.25 kb/s	2.23 kb/s	90 b/s

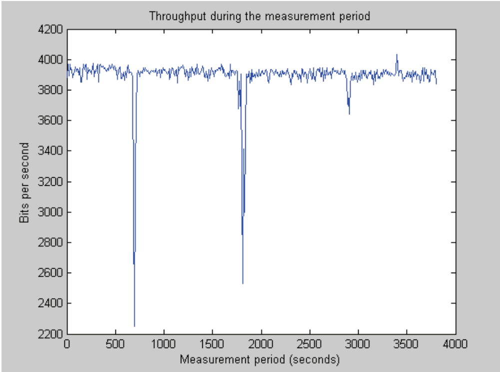

Fig. 4. Throughput per monitoring block for the accelerometer.

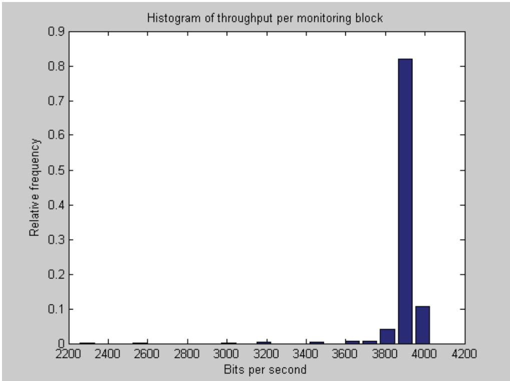

Fig. 5. Histogram of throughput per monitoring block for the accelerometer.

Table 2 shows the loss ratio and mean length of loss periods and loss-free periods for the three wireless links.

TABLE 2: PACKET LOSS BETWEEN SENSOR NODES AND BASE STATION

	Acc–Base stn	ECG–Base stn	Temp-Base stn
Mean loss ratio	1.3%	0.4%	1.4%
Max loss ratio	43%	5.3%	4.0%
Min loss ratio	0.0%	0.0%	0.0%
Loss period mean length	15s	7.4s	197s
Loss-free period mean length	13s	25s	59s

The loss ratio for each monitoring block versus time for the accelerometer is shown in Fig. 6 (compare to Fig. 4). The histogram in Fig. 7 shows the distribution of the losses per monitoring block for the ECG sensor.

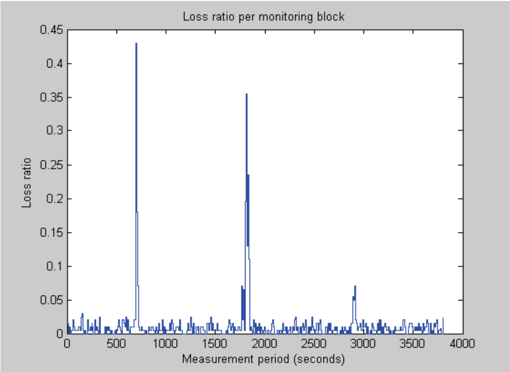

Fig. 6. Loss ratio per monitoring block for the accelerometer.

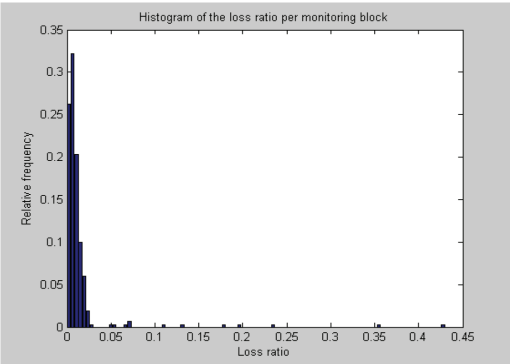

Fig. 7. Histogram of the loss ratio per monitoring block for the accelerometer.

Table 3 shows the inter-arrival jitter. An example for one of the sensors can be found in Fig. 8, and the corresponding distribution in Fig. 9.

TABLE 3: INTER-ARRIVAL JITTER BETWEEN SENSOR NODES AND BASE STATION (MS)

Inter-arrival jitter (ms)	Acc–Base stn	ECG–Base stn	Temp-Base stn
Maximum	13	16	18
Minimum	0	0	0
Standard deviation	4.2	4.1	4.5

The cost for getting the performance information outlined above is in this case around 1% extra energy consumption. It will be lower or higher depending of the size of the monitoring block, which determined the resolution of the metrics. This trade off between energy cost and obtained performance parameter resolution will be studied in more detail.

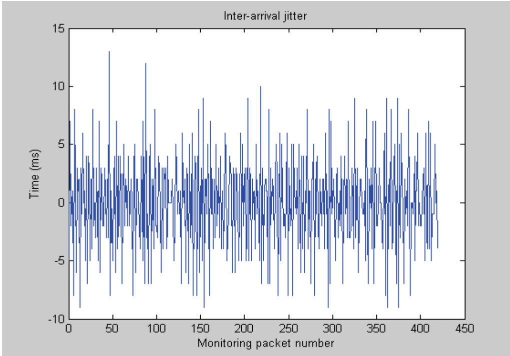

Fig. 8. Inter-arrival jitter for accelerometer.

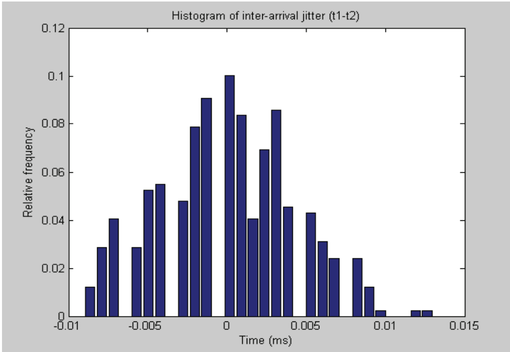

Fig. 9. A histogram of the inter-arrival jitter for the accelerometer.

V. CONCLUSION AND FUTURE WORK

This paper presents a monitoring method and its implementation as a light-weight end-to-end performance meter for quality-demanding applications in wireless sensor networks. The measurement tests have shown promising results. Future work will focus on further improvement of the performance of the meter, analysis of traffic in wireless sensor networks, and to close the loop between monitoring and control based on the feedback information from the meter.

REFERENCES

[1] T. Lindh and N. Brownlee: "Integrating Active Methods and Flow Meters - an implementation using NeTraMet", Passive and Active Measurement workshop (PAM2003), San Diego, April 2003.

[2] M. Brenning, B. Olander, I. Orhan, J. Wennberg, Thomas Lindh: "NeTraWeb: a Web-Based Traffic Flow Performance Meter", SNCNW2006, Luleå, Sweden, 2006.

[3] "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, H. Schultzrinne et al., July 2003.

[4] First IFIP International conference on Wireless Sensor and Actor networks (WSAN2007), Albacete, Spain (www.i3a.uclm.es/wsan07)

[5] TmoteSky – IEEE 802.15.4 compliant sensor module from Sentilla (previously Moteiv).

[6] STB80 – Multi-Modality Sensor Board for TelosB Motes, http://www.easysen.com/SBT80.htm

[7] D. Cavalcanti et al.:"Performance Analysis of 802.14.4 and 802.11e for Body Sensor Network applications", BSN, Aachen. 2007.

[8] N. Golmie et al: "Performance analysis of low rate wireless technologies for medical applications" Computer Communications, June 2005

Performance Monitoring and Control in Contention-Based Wireless Sensor Networks

T. Lindh and I. Orhan
International Symposium on Wireless Communication Systems
September 2009, Siena, Italy.

Performance Monitoring and Control in Contention-Based Wireless Sensor Networks

Thomas Lindh #1, Ibrahim Orhan #2

School of Technology and Health, KTH – Royal Institute of Technology
Marinens vag 30, 136 40 Haninge, Sweden
1 Thomas.Lindh@sth.kth.se
2 Ibrahim.Orhan@sth.kth.se

***Abstract*—This paper presents a method for performance monitoring and control in wireless body sensor networks based on measurement feedback. Test results using a prototype implementation of the method are also analyzed. The method has been evaluated for demanding healthcare related applications in wireless personal area networks.**

I. Introduction

Wireless sensor networks are today being considered for a wide range of demanding applications. One example is wireless body sensor networks that enable continuous monitoring of patients' vital signs parameters in everyday life situations. However, to transmit healthcare related parameters in wireless networks is also a challenge; especially if contention-based access is used. Recipients of data sent in wireless sensor networks need to know whether they can trust the information or not. To address this problem we have developed a performance meter that can measure the performance, and furthermore, to feed a performance control system with real-time measurement data. In Section II we put the results in the context of previous related work. Section III presents the approach and methods for monitoring, feedback and control. Section IV describes the implementation. Section V and Section VI contain the results from test cases.

II. Related work

Measurements, simulations and theoretical studies show that the loss ratio increases with the traffic load and number of sending nodes. Bianchi [1] has derived an analytical Markov chain model for saturated networks, further developed in [2] and extended to non-saturated networks in [3]. Channel errors e.g. due to external disturbances and obstacles in the environment, can of course increase the loss ratio further. Another related problem, studied in [4] is the reduced throughput in multi-hop networks, with one or more intermediate nodes between sender and receiver. Dunkels and Österlind [4] found that the implementation of packet copying in an intermediate forwarding node has significant impact on the throughput.

Performance in LR-WPAN has been analyzed in several studies, often based on simulations ([5],[6]). A performance meter that keeps track of losses, inter-arrival jitter and throughput was presented at BSN 2008 [7]. Several papers have also addressed congestion and rate control in WLAN and LR-WPAN. CODA (congestion detection and avoidance in sensor networks) is a control scheme that uses an open-loop backpressure mechanism as well as a closed-loop control, where a sink node can regulate a source node's sending rate by varying the rate of acknowledgements sent to the source [8]. CARA (collision-aware rate adaptation) uses the RTS packets in IEEE 802.11 as probes to determine whether losses are caused by collisions (related to CSMA/CA) or by channel errors [9].

III. Performance monitoring and control

Fig. 1 shows a configuration with a set of sensor nodes (e.g. a combination of wearable sensors such as ECGs, accelerometers and pulse-oximeters, and fixed environment sensor nodes), a coordinator, one or several intermediate nodes with routing and forwarding capabilities. The application program, running in the coordinator, processes sensor data from the sources and sends the information along with an estimate of the information quality to the remote end-user application or presentation and storage. The information quality can be expressed in terms of e.g. the statistical uncertainty of estimated parameters and the highest frequency component in a signal to be recovered by the receiver.

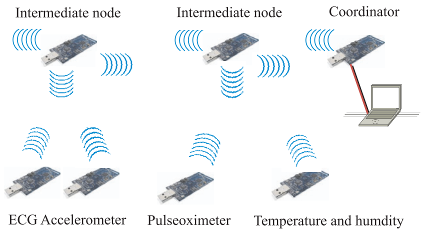

Figure 1. A scenario where performance control is implemented in the coordinator and source nodes.

The system presented in this paper consists of two main parts, a performance meter and a performance manager, described in more detail in the following sections.

The performance monitoring and control capabilities can be implemented as an add-on capability to be used between applications running in the communicating endpoints, e.g. sensor nodes and a coordinator, and not link by link. The ambition has also been to minimize the traffic overhead and energy consumption. The system is targeted to wireless sensor networks that use contention-based access, but can of course also be used in combination with contention-free access, such as guaranteed time slots. The applications, e.g. streaming data from accelerometers and ECGs, require certain levels of throughput and low loss ratio, however not necessarily zero.

The aim is, firstly, to provide quality estimates of the transmitted parameters, and secondly, to reuse this information and enable performance control that minimizes information loss and maintains the desired throughput. This closes the loop between measurements and control.

A. Performance Meter

The performance meter (presented in [7] and inspired by [10]) combines active and passive measurement techniques. It is based on so called monitoring blocks (Fig. 2). The accuracy and resolution of the measurement results is determined by the size of the monitoring block. The meter is implemented in the source nodes and the coordinator. It consists of two counters that keep track of the number of sent and received packets and bytes, and a function that can insert monitoring packets. These measurement packets are inserted between blocks of ordinary data packets as seen in Fig. 2.

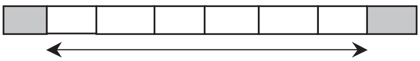

Figure 2. A monitoring block surrounded by two monitoring packets.

They contain a sequence number, a timestamp and the cumulative number of packets and bytes transmitted from the sending node to the receiving node. The interval between the monitoring packets, i.e. the size of the monitoring block, can be expressed in number of packets or a time interval. When a monitoring packet arrives, the receiving node will store a timestamp and the cumulative counter values of the number of received packets and bytes from the sending node.

The following metrics can be calculated and estimated based on the collected measurements: packet loss, inter-arrival jitter and throughput. More detailed information and test results are presented in [7].

B. Measurement-Based Performance Control

The performance manager, implemented in a coordinator node, bases its decisions on the feedback information it receives from the meter, in this case mainly packet loss and throughput. The meter delivers the performance updates for each incoming block of data packets, e.g. 100 packets, from a sensor to the coordinator. The output of the performance manager can e.g. be to increase or decrease the packet frequency, change the transmission power, enable or disable acknowledgement etc. In this study the control actions are limited to varying the packet frequency. Some examples of possible control algorithms are described in Section III-C.

In summary, the monitoring and control method has three main parameters, explained in the coming sections, that can be tuned: the size of the monitoring block (B); the number of previous monitoring blocks (B_n, B_{n-1}, B_{n-2} etc), and their relative weight, that the control algorithm is based on; and, the step size (Δt) that controls the time interval between transmitted packets (or packet frequency).

C. Feedback Control Algorithms

The output of the control algorithm, to decrease or increase the packet frequency, is based on performance data from the current and previous monitoring blocks. The loss ratio and throughput (received bits per second) for a number of the recently received monitoring blocks is kept in memory. The manager sends a request message to a sensor node to either reduce or increase the packet frequency by adding (or subtracting) Δt milliseconds to (or from) the time interval between the transmitted packets. The step size, Δt, is determined by a weighted average of the performance feedback from the current monitoring block and m previous blocks.

Results from test cases with three simple control algorithms are presented in Section V; the first to protect performance of high-priority nodes (Section V-A), the second where all nodes have the same priority (Section V-B), and finally a case where these algorithms are combined and priority is assigned dynamically to one node (Section V-C).

D. Test cases

The purpose of the first control algorithm (Section V-A) is to maintain throughput and minimize losses for a node with high priority, by punishing nodes with low priority. The algorithm works like this. The manager will keep the throughput between a maximum and minimum level. If the throughput drops below the minimum level, the performance manager tells the node to decrease the packet interval by Δt milliseconds. If the throughput rises above the maximum level, the performance manager increases the packet interval by Δt milliseconds.

When monitoring block B_n has arrived at the coordinator, a weighted average of the throughput and the loss ratio based on block B_n, B_{n-1} and B_{n-2} are computed. If the loss ratio for the high-priority node is above the threshold, the coordinator instructs the low priority nodes to increase the packet interval by Δt milliseconds. This is repeated until the loss ratio for the prioritized node is below the threshold.

One reason for a drop in throughput is the contention-based access mechanism. The total time interval between two transmitted packets at the sender side is $t_{ACCESS}+t_{DELAY}+t_{TRANS}$, where t_{ACCESS} is the waiting time due to the access method CSMA/CA, t_{DELAY} is the interval between the time the transmission of packet n is completed and the time the transmission of packet $n+1$ begins, and t_{TRANS} is the frame transmission time (the frame size divided by the bit rate, $250\,\mathrm{kb/s}$ in this case). t_{DELAY} is increased or reduced by the Δt control requests described above. The access time varies depending on how many nodes that are trying to transmit in the same channel. An increase in t_{ACCESS} will lower the packet frequency and throughput.

In the second case, where all nodes have the same priority (Section V-B), each node tries to maximize its throughput under the condition that the loss ratio is below a threshold. The third case (Section V-B) is a combination of the two previous ones. From the beginning both nodes have the same priority. After a certain time, one of the nodes is assigned high priority and higher expected throughput.

IV. System implementation

The testbed used in this work consists of TmoteSky sensor nodes running TinyOS 2.1.0 programmed in nesC. The radio (CC2420) and link layer are compliant with IEEE 802.15.4 LR-WPAN in contention-based access mode. The software system consists of two parts, the performance meter and the performance manager. The meter stores performance data, as described in Section III-A, for each block of received packets (the monitoring block size). This monitoring data is used in two ways: firstly, to estimate the information quality of transmitted sensor data, and secondly, to feed the performance manager function with information for control decision. The performance meter is 60 lines of nesC code in the coordinator and 25 lines of code in a sensor node. The performance manager part is implemented as 65 lines in the coordinator and 5 lines of code in a sensor node.

V. Test results

Fig. 3 shows a test scenario with two sensor nodes that are streaming ECG samples and accelerometer samples to the coordinator through a forwarding intermediate node. Results from two test cases with different control algorithms are presented in the following sections.

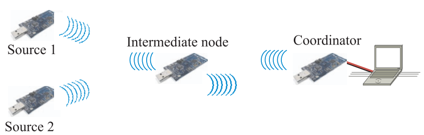

Figure 3. Two source nodes that are sending sensor data to a coordinator via an intermediate node. Source 1 has high priority and source 2 has low priority.

A. Control Scheme with Priority

The control algorithm in our test case means that one of the sensor nodes has high priority and the other one has low priority. The loss ratio threshold is computed as a weighted average of the three recent consecutive monitoring blocks and compared to the threshold 0.02. The required bit rate is 8 kb/s, which corresponds to approximately 250 Hz sampling rate per axis for a two-axis accelerometer or a 500 Hz ECG.

Fig. 4 – Fig. 7 illustrate how the implemented algorithm works in practice. The high priority node starts from 10 kb/s and slows down to the expected bit rate 8 kb/s (Fig. 4). The second node is turned on shortly thereafter ($t \approx 80$ s) at a rate of nearly 16 kb/s (Fig. 5). The received bit rate from the high priority node falls sharply (Fig. 4). The solid lines (blue) show the received bit rate measured at the coordinator. The dotted lines (red) represent the sending bit rate from the sensor node.

The loss ratio for the high priority node peaks at nearly 0.45 (Fig. 6) when the second node starts transmitting. The loss ratio for the low-priority nodes is shown in Fig. 7.

The performance manager reads the performance data provided by the meter for each block of incoming data packets. The monitoring block size is 100 packets in this test case. As soon as the manager detects the increased loss ratio for the high priority node, it will instruct the other node to slow down. The low priority node will directly decrease the transmitting rate (Fig. 5), which results in lower loss ratio (Fig. 6) and higher throughput (Fig. 4) for the prioritized node. As the loss ratio approaches the threshold, the sending rate of the low priority node stabilizes around 3 kb/s (Fig. 5). The performance manager strives to maintain the desired throughput (8 kb/s) for the high-priority during the remaining part of the test, with an average loss ratio below the threshold.

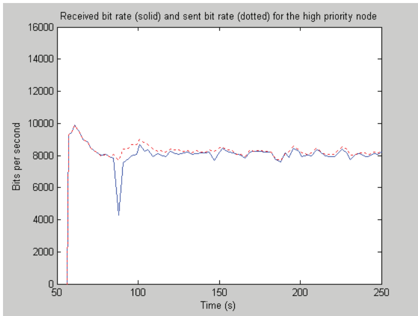

Figure 4. Throughput for the high-priority node.

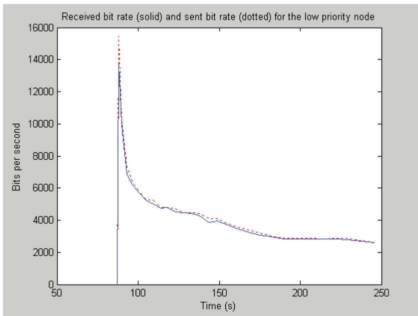

Figure 5. Throughput for the low-priority node.

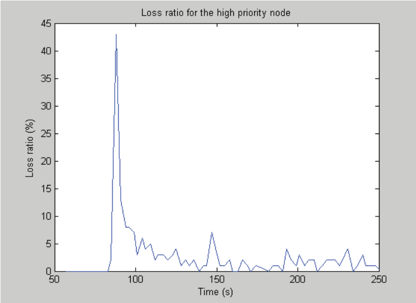

Figure 6. Loss ratio for the high-priority node.

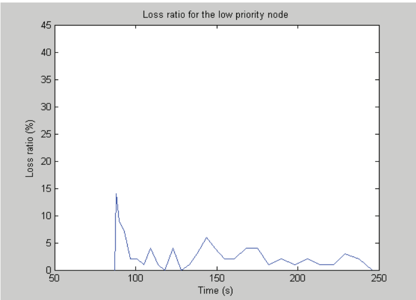

Figure 7. Loss ratio for the low-priority node.

B. Control Scheme without Priority

In this test case a priority scheme is not used. Both sensor nodes are controlled independently by the performance manager under the condition that the loss ratio is below a threshold. If the loss ratio is above the threshold, the sensor node will be instructed to decrease the sending rate (increase the packet interval by Δt milliseconds). No expected throughput is specified.

Both sensor nodes start sending at 18 kb/s as seen in Fig. 8 and Fig. 9. The high loss ratio for both nodes means that the performance manager will order both of them to slow down until the losses fall below the threshold. It can also be observed that the sensor node sometimes maintains the sending rate, even though the loss ratio is significantly higher than the threshold (Fig. 8 and Fig. 10). The explanation is that during heavy loss monitoring packets will also be lost, which delays the decision to decrease the packet frequency.

After a while, the first node's throughput stabilizes around 3 kb/s (Fig. 8) and around 3.5 kb/s for the second node (Fig. 9). Since the control of the sensor nodes is independent of each other, the throughput will normally not be on the same level. One reason is different loss characteristics of the two channels; another may be different starting values. Each sensor tries to find its maximum bit rate without exceeding the loss ratio threshold.

At approximately $t = 180\,\mathrm{s}$ the manager has observed

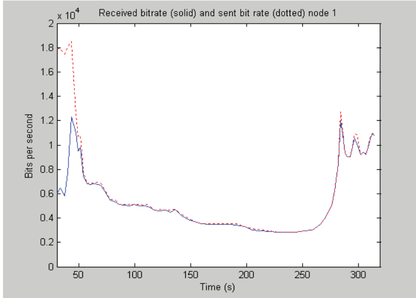

Figure 8. Throughput for node 1 (test case without priority).

that the recent monitoring blocks are loss-free. The packet frequency is therefore increased for node 2 (Fig. 9). At $t = 210\,\mathrm{s}$, sensor node 2 stops transmitting (Fig. 9), which results in approximately zero packet loss for sensor node 1 (Fig. 10). The manager therefore tells the node to increase the packet frequency, up to around 10 kb/s, where the loss threshold forces the node to slow down (Fig. 8).

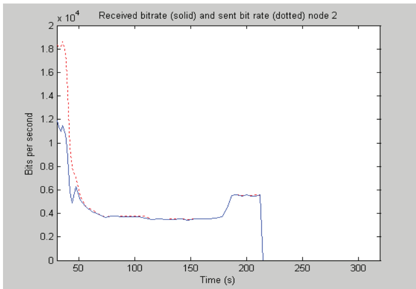

Figure 9. Throughput for node 2 (test case without priority).

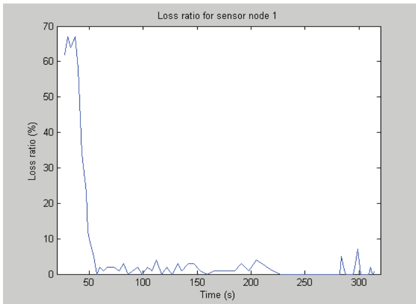

Figure 10. Loss ratio for node 1 (test case without priority).

C. Dynamic Priority Assignment

Fig. 11 and Fig. 12 show a combination of the previous two control algorithms. Both nodes start at bit rate just below

15 kb/s with the upper limit loss ratio 0.02. No node is given priority over the other. The throughput stabilizes between 4 kb/s and 5 kb/s. At $t \approx 300$ seconds, one of the nodes (Fig. 11) is dynamically assigned high-priority, while the other node has to be satisfied with what is left. The reason might be that a higher sampling rate is needed for a sensor.

The bit rate for the high priority node rises to the required 8 kb/s (Fig. 11) and the other sensor node backs off to around 2.5 kb/s (Fig. 12). The step response in Fig. 11 takes around 30s. This time period can be reduced either by allowing larger step sizes (Δt) or decreasing the interval between the monitoring packets).

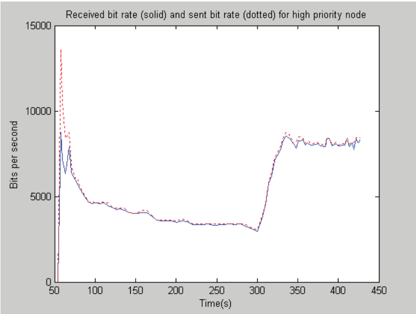

Figure 11. The sensor node is assigned high priority at t=300s and raises the bit rate to 8kb/s.

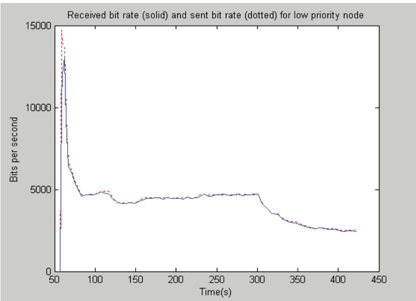

Figure 12. The sensor node is assigned low priority at t=300s and reduces the bit rate to 2.5kb/s

D. Multi-Hop Cases

The bit rate from a sensor to a coordinator will to a large extent depend on the number of hops between the source and destination [4]. The maximum received throughput for the equipment in our testbed (Section IV) using maximum packet length (payload 112 byte) was 50 kb/s for one hop, 35 kb/s for two hops and 20 kb/s for three hops. This is of course a crucial limitation for demanding applications.

VI. Evaluation

This study shows that it is feasible to use the measurement method, based on monitoring blocks, for performance monitoring as well as for feedback control of the performance of applications in sensor networks. The method has been implemented in a network with contention-based (CSMA/CA) access. It can of course also be used for the contention-based part of a super-frame in beacon mode in IEEE 802.15.4, where the contention-free part has guaranteed timeslots for the most demanding applications.

One observation is that avoiding packet loss in situations of buffer saturation by reducing the packet frequency is more straightforward than to handle packet loss due to collisions and possible channel errors.

The size of the monitoring block is an important parameter for the resolution of the performance metrics as well as for the responsiveness of the control function. It deserves a more detailed study. A problem related to control theory is the design of the control algorithm with respect to current and previous estimates of performance data.

To find out, in real-time, what capacity is available for a specified loss ratio, given that a second node transmits 2 kb/s, is another application of the control method.

VII. Conclusions

The presented control method and the prototype implementation can be used to provide quality of service control of applications in wireless sensor networks using contention-based access. A performance meter continuously feeds the performance control function with quality estimates of the transmitted sensor data. The tests results of the implemented algorithms were promising.

References

[1] G. Bianchi, "Performance analysis of the ieee 802.11 distributed coordination function," *IEEE JSAC*, vol. 18, Mar. 2000.

[2] H. L. Vu, "Collision probability in saturated ieee 802.11 networks," in *Proc. Australian Telecommunication Networks and Applications Conference*, Australia, 2006.

[3] K. Duffy, D. Malone, and D. Leith, "Modeling the 802.11 distributed coordination function in non-saturated conditions," *Communications Letters IEEE*, vol. 9, no. 15, Aug. 2005.

[4] F. Österlind and A. Dunkels, "Approaching the maximum 802.15.4 multi-hop throughput," in *Proc. HotEmnets*, Virginia, Jun. 2008.

[5] D. Cavalcanti *et al.*, "Performance analysis of 802.14.4 and 802.11e for body sensor network applications," in *Proc. Body Sensor Networks*, Aachen, Germany, Mar. 2007.

[6] N. Golmie *et al.*, "Performance analysis of low rate wireless technologies for medical applications," *Computer Communications*, Jun. 2005.

[7] I. Orhan, A. Gonga, and T. Lindh, "An end-to-end performance meter for applications in wireless body sensor networks," in *Proc. Body Sensor Networks*, Hongkong, Jun. 2008.

[8] C. Wan, S. Eisenman, and A. Campbell, "Coda: congestion detection and avoidance in sensor networks," in *Proc. 1st conference on embedded networked sensor systems Body Sensor Networks*, Los Angeles, 2003.

[9] J. Kim, S. Kim, S. Choi, and D. Qiao, "Cara: Collision-aware rate adaptation for ieee 802.11 wlans," in *Proc. INFOCOM*, Barcelona, Apr. 2006.

[10] T. Lindh and N. Brownlee, "Integrating active methods and flow meters - an implementation using netramet," in *Proc. Passive and Active Measurement workshop (PAM2003)*, San Diego, Apr. 2003.

Measurement-Based Admission Control in Wireless Sensor Networks

I. Orhan and T. Lindh
The Third International Conference on Sensor Technologies and Applications, Sensorcomm 2010
July 2010, Venice, Italy
The paper received a best paper award.

Measurement-Based Admission Control in Wireless Sensor Networks

Ibrahim Orhan
School of Technology and Health
KTH
Stockholm, Sweden
Ibrahim.Orhan@sth.kth.se

Thomas Lindh
School of Technology and Health
KTH
Stockholm, Sweden
Thomas.Lindh@sth.kth.se

Abstract—**Wireless sensor networks have today emerged as a feasible infrastructure for healthcare applications. This paper addresses the non-trivial performance problems in contention-based wireless networks. We present a method for admission control in contention-based networks, implemented as a component of a performance management system. The test results show that admission control can improve the predictability and level of performance in wireless sensor networks. The system can be used as a tool for dimensioning and configuration as well as for real-time admission control. The often unpredictable dynamics in contention-based access networks means that continuous performance control is needed to maintain a desired quality of service.**

***Keywords*-wireless sensor networks, admission control, performance monitoring and control.**

I. INTRODUCTION

Wireless personal area networks have emerged as an important communication infrastructure in areas such as at-home healthcare and home automation, independent living and assistive technology, as well as sports and wellness. Initiatives towards interoperability and standardization are taken by several players e.g., in healthcare applications. Zigbee Alliance has launched a profile for "Zigbee wireless sensor applications for health, wellness and fitness" [1]. The Continua Health Alliance promotes "an interoperable personal healthcare ecosystem" [2], and at-home health monitoring is also discussed in an informational Internet draft [3]. It shows that wireless personal area networks, including body sensor networks, are becoming more mature and are considered to be a realistic alternative as communication infrastructure for demanding services. However, to transmit data from e.g., an ECG in wireless networks is also a challenge, especially if multiple sensors compete for access as in CSMA/CA. Contention-based systems offer simplicity and utilization advantages, but the drawback is lack of predictable performance. This paper discusses whether admission control in combination with a system for continuous performance management can provide improved and more predictable performance.

Admission control is used in many traditional telecom systems. It is also proposed in new Internet service architectures [4] to provide guarantees for quality of service. In this paper we present a method for measurement-based admission control in wireless personal area sensor networks for contention-based access. It is implemented as a part of an integrated performance management system that also comprises performance monitoring, admission control and performance control.

The rest of the paper is organized as follows: a survey of related work in Section II; performance management in wireless sensor networks in Section III; measurement-based admission control in Section IV; use cases and test results in Section V; and finally the conclusions in Section VI.

II. RELATED WORK

Our approach is to base the decision, to accept or reject a request to join the network, on measurements of performance parameters, mainly the packet loss ratio. A similar probe-based admission control procedure has been suggested for differentiated internet services [4]. Alternatively, one can measure the available capacity between two endpoints or on certain links in a network. Pathrate, Pathload and BART are examples of implementations of such estimation tools ([5], [6] and [7]). SenProbe [8] estimates the maximum achievable rate between two endpoints in wireless sensor networks by injecting packet trains and analysing the dispersion between the packets. Some experimental studies indicate that measurements of available capacity in wireless networks often give inaccurate results especially for multiple hops [9]. Instead of active measurements, Contention-aware admission control protocol (CACP) estimates the available capacity by measurements from each node of the amount of time the channel is busy [10]. Perceptive admission control (PAC) is an extension of CACP to encompass node mobility [11]. We have preferred a straightforward approach where the decision to either accept or reject an admission request is based on direct measurements and estimates of the performance parameters that are decisive for the quality of services.

III. PERFORMANCE MANAGEMENT IN WIRELESS SENSOR NETWORKS

Admission control needs to be seen in the context of other necessary functions, especially performance measurements and control. In this section we briefly present the performance management system that admission control is a part of. The performance manager consists of the following functions: a performance meter that collects measurement data; admission control that handles requests to join the network; and performance control that maintains the quality of service for the admitted sensor nodes. The performance meter provides feedback measurement data for admission control and performance control. Fig. 1 shows the relationship between these functions. A request from a sensor node to join the network is handled by the admission control based on feedback from the meter. The performance control function is responsible for maintaining the desired quality of service once the sensors are allowed to use the wireless channel.

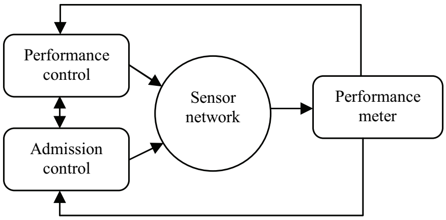

Figure 1. The performance manager consists of admission control and performance control. The performance meter supports the manager with measurement data.

A. *The Performance Meter*

The performance meter, which is based on so called monitoring blocks (Fig. 2), is implemented in the sensor nodes and the coordinator (Fig. 3). It consists of counters that keep track of the number of sent and received packets and bytes, and a function that can insert monitoring packets. These monitoring packets are inserted between blocks of ordinary data packets as seen in Fig. 2. They contain a sequence number, a timestamp and the cumulative number of packets and bytes transmitted from the sending node to the receiving node. The interval between the monitoring packets i.e., the size of the monitoring block, can be expressed in number of packets or a time interval. When a monitoring packet arrives, the receiving coordinator will store a timestamp and the cumulative number of received packets and bytes from the sending node. The following metrics can be calculated and estimated based on the collected measurements: packet loss, inter-arrival jitter and throughput. Detailed information and test results are presented in [12].

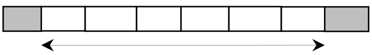

Figure 2. A monitoring block containing data packets surrounded by two monitoring packets.

B. *Feedback Performance Control*

The performance manager, implemented in the coordinator, uses the measurement feedback it receives from the meter: packet loss, delays and throughput. The meter delivers these performance updates for each incoming monitoring block e.g., once a second. The performance manager can order sensor nodes to decrease their transmission to maintain the quality of service for the connected nodes. Sensor nodes can also be assigned different priority. If a high-priority sensor experiences losses above a specified threshold, the manager can instruct low-priority sensors to back-off and decrease their sending rate. The control algorithm has three main tuneable parameters that determine the responsiveness of the control actions: the size of the monitoring blocks; the number of previous monitoring blocks used by the algorithm; and the step size to decrease or increase the packet frequency. More details on the performance manager are presented in [13].

IV. Admission Control in Wireless Sensor Networks

In this section the main idea behind the admission control system is presented.

A. *An Application Scenario*

A typical application scenario is healthcare at-home with a number of sensors such as ECGs, pulse-oximeters, accelerometers etc connected to a coordinator. Fig. 3 shows a case with three sensors connected to a coordinator sharing the same wireless channel that applies the CSMA/CA access method. Several hops between the sensors and the coordinator including mobility are also a feasible. Sensor A and sensor B in Fig. 3 are already connected to the wireless channel transmitting sensor data to the coordinator. The sensors have a priority levels, a specified throughput and an upper limit for the packet loss ratio. Sensor C accesses the channel and requests admission for a specified throughput and related packet loss requirements.

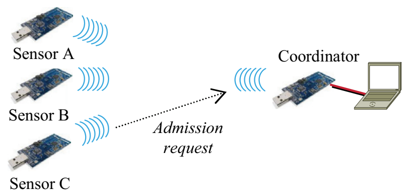

Figure 3. Sensor C requests to share the wireless channel already used by sensor A and sensor B.

B. *Measurement-Based Admission Control for Contention-Based Access*

The purpose of admission control is to accept or reject new sensor nodes to an existing network, while protecting the performance of already admitted nodes. This paper focuses on whether it is feasible or not to use admission control in contention-based wireless sensor networks. Our approach is to found the decision to accept or reject an admission request on estimates of real-time measurement data provided by a performance meter. A sensor node that intends to enter the network specifies the sampling rate, the sample size and the performance requirements. The verdict, to accept or reject the request, is determined by the outcome of probe packets transmitted during a test period. The test traffic sent from the requesting node to the coordinator should be the same as the ordinary traffic it will transmit if admitted. The exchanged messages between a requesting node and the coordinator are described in the next subsection.

In contention-based access networks, such as CSMA/CA for IEEE 802.15.4, strict performance guarantees are not feasible. However, many applications do not require completely loss-free transmission and are satisfied with soft performance requirements e.g., upper limits on packet loss and delay variation. This paper addresses the need for performance guarantees and predictability in contention-based networks for such applications.

1) Messages between the coordinator and sensor nodes

A simple protocol for exchange of messages between the coordinator and the sensor nodes has been defined (see Fig. 4). Sensor nodes send requests to join the network for a specified sampling rate, sample size, priority and upper limits on performance parameters. If the coordinator is not busy handling previous requests, it will approve further processing. The sensor node is then instructed to start transmitting test traffic, including monitoring packets. When the test period is ended, the sensor node asks the coordinator for the decision. Having received 'accept', the sensor node begins transmitting its ordinary data packets to the coordinator. Monitoring packets are inserted between blocks of *n* data packet or with a certain time interval, to provide the performance meter at the coordinator with real-time updates of the transmission quality.

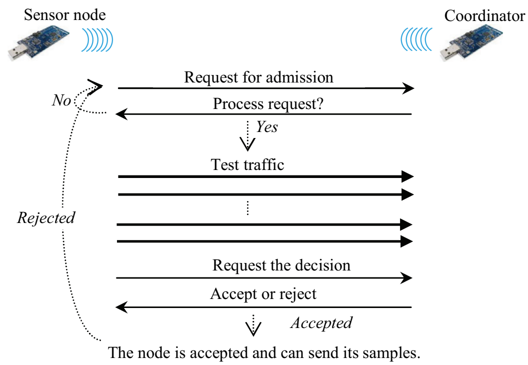

Figure 4. The messages between a sensor node and the coordinator during the admission phase. The arrows in thin lines are signalling messages and the arrows in thick lines represent test traffic.

2) Admission test period

The sensor nodes transmit packets during the test period in the same way as they intends to do if the request is accepted e.g., with a 200Hz sampling rate and 12 bits sample size. The performance meter will report the performance data for traffic between the coordinator and all sensor nodes, including the test traffic from the requesting sensor. Admission is accepted if the average of the performance parameters for any of the already permitted nodes, as well as for the requesting node, is below the threshold value. Admission can be denied to protect the existing nodes from performance degradation.

The length of the test period is a trade-off between retrieving enough information from the probe packets and minimizing the effect on the other sensor nodes' performance. The first priority is to protect the already admitted nodes. The test traffic will be interrupted as soon as the disturbance of the probe packets exceeds the specified thresholds.

The probe packets sent during the test period represent a sampling process of the wireless channel, where the outcome of each sample event is that the packet is lost or not. The probability to lose a packet depends on the total traffic load and the number of nodes transmitting (ignoring radio channel disturbances). The number of samples needed for a given confidence level is determined by the variation of the traffic load. We have assumed that the sampling frequencies of the sensors are stable. This is a reasonable assumption for the kind of the applications the system is intended for. It means that the variance of the traffic load over time is low, and accordingly, that the number of probe packets can be kept low. The experiences from the test cases (Section V) in a normal home environment confirm that a test period of less than 30 seconds is sufficient. The length of the test period is further discussed in Section V.B.

3) Applications

The method and system described in this paper can be used for admission control and continuous real-time performance monitoring and control of operating wireless sensor networks. In addition, it is suitable for dimensioning, configuration and testing of wireless sensor networks prior to operational mode. The system can determine the number of sensors with certain capacity that can share a wireless channel, for given performance requirements. Alternatively, the system can verify the actual performance results for ECGs, accelerometers, pulse-oximeters etc in a wireless sensor network.

V. USE CASES

In this section we present test cases that illustrate the potential performance problems with contention-based access and the need for admission control as well as continuing performance monitoring and control. The first case (Section V.A) illustrates the non-trivial performance problems associated with contention-based access (CSMA). The second case (Section V.B) shows how admission control works in real-time. The length of the test period is also discussed. In the third case (Section V.C), the implemented system is used as an off-line configuration tool to determine how changes of the traffic pattern influence the packet loss ratio. Finally, the alternative to refer a requesting sensor node to a new channel is mentioned in Section V.D.

The testbed in this work consists of TmoteSky sensor nodes running TinyOS 2.1.0 programmed in nesC [14]. The radio (CC2420) and link layer are compliant with IEEE 802.15.4 LR-WPAN in contention-based access mode. The sensor nodes transmit samples from ECGs, pulse-oximeters and accelerometers with sampling rates from 100Hz to 250Hz.

A. *Performance Problems in Contention-Based Access*

Contention-based access is a challenge for applications that require good and predictable performance. Fig. 5 illustrates what can happen when several sensors access a wireless channel. Three sensors (A, B and C) are connected to a coordinator sharing the same channel. The sensors are sampled during a second and the packets are sent back-to-back once a second. The bit rate is 9.6kbps for each sensor node. Fig. 5 shows the loss ratio during a measurement period for sensor A. During the first part (0-70 seconds) only sensor A is active. The loss ratio is almost zero. Between 70-140 seconds sensor B also accesses the channel. The average loss ratio experience by sensor A is 0.03. During the remaining measurement period all three sensors are transmitting on the same channel. The average loss ratio suddenly rises to 0.40. Next time the measurement is repeated the loss ratio might be considerably lower.

For a loss-sensitive application, the performance is unacceptable after sensor B, and especially after sensor C, have accessed the channel. The performance degradation may be avoided if the coordinator applies admission control and also maintains performance monitoring and control to protect the quality of service requirements for the existing nodes.

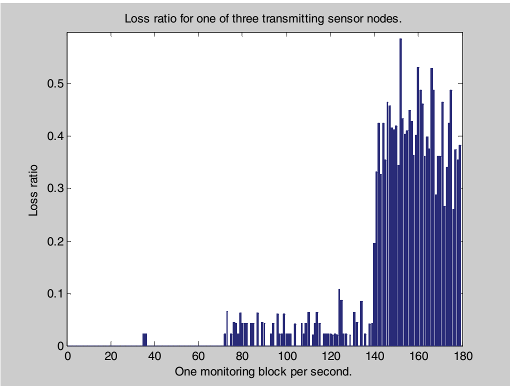

Figure 5. The loss ratio (y-axis) for sensor A node during the measurement period (x-axis in seconds). At approximately t=70s sensor B joins the channel. At t=140s a third node, sensor C, accesses the channel.

B. *Admission Control*

In this test case the coordinator applies admission control when three sensors (accelerometers), one by one, request to join the wireless network (Fig. 6). The sampling rate for the three-axis accelerometer is 200Hz per axis and the resulting average bit rate is 9.6kbps. The upper limit for packet loss for each node is set to 0.02 per monitoring block (the block size is around 1 second). The admission test period is 30 monitoring blocks (30 seconds). The measurement sequence is outlined in Fig. 7.

Sensor A requests admission and begins transmitting probe packets. The loss ratio during this test period is zero. Sensor A's request is accepted and it starts transferring data. The loss ratio for the data traffic from sensor A is almost zero for the period before sensor B requests to join the channel. Table 1 summarizes the loss ratio for each sensor nodes during all test periods and data transfer periods. Losses that exceed the threshold (0.02) are indicated in bold text. It turns out that sensor A and B are accepted, while sensor C is rejected. For a sensor to be rejected it is sufficient that the loss ratio for one of the sensors, including the requesting node itself, exceeds the threshold.

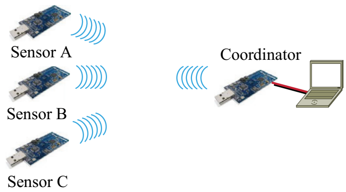

Figure 6. Three sensor nodes connected to the coordinator sharing the same channel.

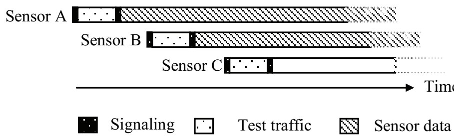

Figure 7. The measurement sequence for the nodes in Fig. 6.

TABLE I. LOSS RATIO FOR TEST PERIODS AND DATA TRANSFER PERIODS FOR SENSOR A, B AND C.

	Sensor A	Sensor B	Sensor C
Test period sensor A	0.0000	--	--
Data transfer	0.0006	--	--
Test period sensor B	0.0012	0.0085	--
Data transfer	0.0019	0.0088	--
Test period sensor C	0.0046	**0.0470**	**0.0250**
Data transfer	0.0051	0.0083	--

The length of the test period is a trade-off between, on the one hand, to minimize the disturbance of existing traffic and reducing the response time for the admission verdict, and on the other hand, to receive sufficient performance data. The drawback of a predetermined fixed length test period is that ongoing traffic may suffer from severe performance deterioration. Fig. 8 shows the impact of test traffic on a sensor node during a 30 second test period. The average loss ratio is almost 0.05, with several peaks around 0.10, which is unreasonable performance deterioration for an existing node during a test period. To avoid this we use an algorithm that calculates the cumulative moving average of the loss ratio for each incoming performance update i.e., for each monitoring packet. The test period is interrupted if the cumulative average exceeds a threshold. The cumulative moving average is defined as $CA_i=(L_1+L_2+L_3+...+L_i)/i$, where L_i is the loss ratio for monitoring block *i*. The algorithm is applied to three examples of test periods in Fig. 8-10. The cumulative averages for the first five blocks in Fig. 8 are $CA_1=0.059$, $CA_2=0.035$, $CA_3=0.032$, $CA_4=0.042$ and $CA_5=0.042$.

If the rule for admittance is to allow maximum three consecutive updates of the loss ratio above the threshold (0.02), the test period will be interrupted after the third block. An additional requirement that the loss ratio for a single block may not exceed 0.05 would in this example mean that the test period is stopped after the first monitoring block.

A slightly different loss pattern is depicted in Fig. 9 (sensor C's loss ratio during a test period). The cumulative average for the first seven blocks are $CA_1=0.0118$, $CA_2=0.0119$, $CA_3=0.0159$, $CA_4=0.0240$ and $CA_5=0.0216$, $CA_6=0.0201$ and $CA_7=0.0206$. In this case, the test period terminates after the 6th monitoring block and the request is rejected.

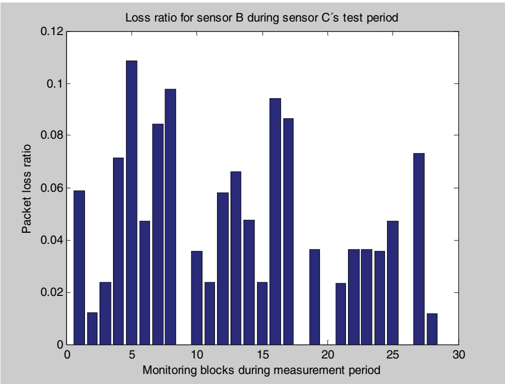

Figure 8. Loss ratio per monitoring block experienced by sensor B during the third sensor's (sensor C) test period. The average loss ratio is 0.047.

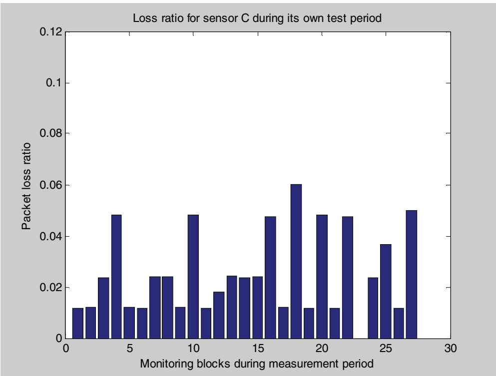

Figure 9. Loss ratio per monitoring block experienced by sensor C during its own test period. The average loss ratio is 0.025.

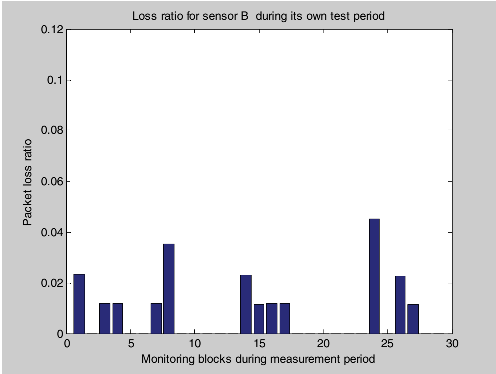

Figure 10. Loss ratio per monitoring block experienced by sensor B during its own test period.

C. *Traffic Patterns and Channel Access*

Packet loss in contention-based wireless networks is sensitive to the traffic pattern from the individual sources. Assume that two nodes collect samples and transmit the samples as a train of packets periodically once a second. If the nodes transmit the packet trains without overlap in time, the risk for losses due to collisions is low. However, the loss probability will increase if the packet trains happen to coincide. The dynamics of the traffic patterns in a network may from time to time lead to losses that exceed the accepted level after the admission test periods. The unpredictability of performance deterioration in wireless contention-based networks means that admission control must be combined with continuous traffic monitoring and control to be able to maintain the desired performance goals.

We have performed tests to study the impact of changes in traffic pattern on packet loss. Sensor node A collects and stores samples during a second. The samples are encapsulated in packets and transmitted back-to-back. The total time to transmit the packet train depends on the sampling rate, the sample size and the packet size. In this case the sensor node sends a packet train of 43 packets with a packet size of 28 bytes, which corresponds to a throughput of 9,6kb/s. The total time to send the packet train was around 500ms. A second node, sensor B, starts transmitting probe packets. It sends a train of packets once a second during the test period. The starting time for each train is shifted 50ms after ten seconds. This is repeated ten times, which means that the total time shift of the packet trains is around 500ms. The basic idea is to let the packet trains from sensor B slide over the packet trains from sensor A. Fig. 11 illustrates this convolution-like procedure.

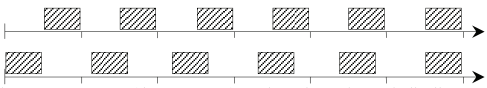

Figure 11. Sensor A (the upper part) sends packet trains periodically every second. The starting times of the trains transmitted by sensor B (lower part) are shifted in time so that they slide over the packet trains from sensor A.

Fig. 12 shows the loss ratio for sensor B. After 10 monitoring blocks (10 seconds) the starting time is shifted 50ms. The average loss ratio for the first half of the measurement period is below 0.01. It rises to 0.10 for block 81-90 and 0.17 for block 91-100. The highest losses occur when the packet trains from the two sensors coincide in time. This convolution-like test might be inappropriate to use in an operating network but is useful for out-of-service configuration and dimensioning tests to estimate a worst case loss ratio.

The traffic pattern for a channel e.g., the starting times of packet trains, is a stochastic process that may result in random losses from zero up to 25% in this case. Due to the unpredictability of contention-based wireless access continuous performance monitoring and control is needed to maintain the desired performance levels.

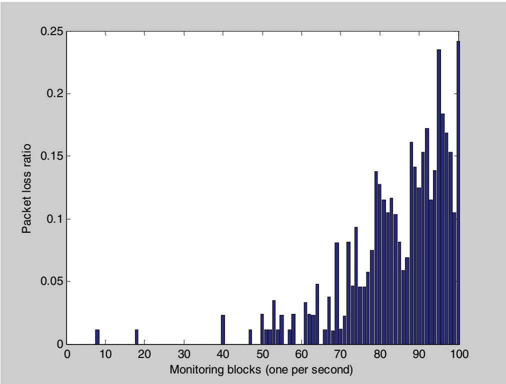

Figure 12. Loss ratio for sensor B. The peak values occur when the packet trains from sensor B coincide in time with the packet trains from sensor A.

D. Redirecting to Another Channel

When a sensor node's request to join the network is rejected there are two alternatives. The node may back off for a while and try once again later. Alternatively, the coordinator may refer the sensor to another radio channel. This feature has been successfully implemented and tested.

VI. CONCLUSIONS

Wireless sensor networks have today emerged as a feasible infrastructure for healthcare applications. This paper has addressed the non-trivial performance problems related to contention-based access wireless channels. We have presented a method for admission control, based on transmission of probe packets during a test period, as a component of a performance management system. This system can be used as a tool for dimensioning and configuration, as well as real-time admission control and continuous performance management of wireless sensor networks.

The length of the test period is a trade-off between minimizing the disturbances on existing traffic, and receiving sufficient performance data for the admission verdict. The proposed algorithm uses a cumulative moving average of the loss ratio for the traffic from each sensor node to decide whether to reject an admission request and interrupt the test traffic, or to allow the sensor use the network. The test results show that admission control can improve the level and predictability of the performance of wireless sensor nodes. A final conclusion is that continuous performance monitoring and control is needed to maintain the desired performance levels.

VII. REFERENCES

[1] "Zigbee wireless sensor applications for health, wellness and fitness", Zigbee Alliance, March 2009.

[2] R. Carroll, R. Cnossen, M. Schnell, and D. Simons, "Continua: an Interoperable Personal Healthcare Ecosystem", IEEE Pervasive Computing, Vol.6, No 4, October-December 2007.

[3] A. Brandt (Zensys Inc) and G. Porcu (Telecom Italia), "Home Automation Routing Requirements in Low Power and Lossy Networks", Internet Draft, September 2009.

[4] I. Más and G. Karlsson, "Probe-based admission control for differentiated-services internet", Computer Networks 51, pp.3902-3918, 2007.

[5] P. Ramanathan, D. Moore, and C. Dovrolis "What Do Packet Dispersion Techniques Measure", In Proceedings of IEEE INFOCOM, 2001, pp. 905-914, April 2001, Anchorage, Alaska, USA.

[6] M. Jain and C. Dovrolis, "Pathload: a measurement tool for end-to-end available bandwidth", Passive and Active Measurements Workshop, March 2002, Fort Collins, USA.

[7] S. Ekelin, M. Nilsson, E. Hartikainen, A. Johnsson, J-E Mångs, B. Melander, and M. Björkman, "*Real-Time Measurement of End-to-End Available Bandwidth using Kalman Filtering*", IEEE NOMS 2006, Vancouver, Canada.

[8] T. Sun, L. Chen, G. Yang, M. Y. Sanadidi, and M. Gerla, "SenProbe: Path Capacity Estimation in Wireless Sensor Networks" SenMetrics 2005, July 2005, San Diego, USA.

[9] D. Gupta, D. Wu, P. Mohapatra and C-N. Chuah, "Experimental Comparison of Bandwidth Estimation Tools for Wireless Mesh Networks", IEEE INFOCOM Mini-Conference, April 2009.

[10] Y. Yang and R Kravets, "Contention-Aware Admission Control for Ad Hoc Networks" Mobile Computing, IEEE Transactions on Volume 4, Issue 4, July-Aug. 2005.

[11] Ian D. Chakeres and Elizabeth M. Belding-Royer, "PAC: Perceptive Admission Control for Mobile Wireless Networks", International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine), October 2004, Dallas, USA.

[12] I. Orhan, A. Gonga, and T. Lindh, "An End-to-End Performance Meter for Applications in Wireless Body Sensor Networks", International Workshop on Wearable and Implantable Body Sensor Networks, June 2008, Hong Kong.

[13] T. Lindh and I. Orhan, "Performance Monitoring and Control in Contention-Based Wireless Sensor Networks", International Symposium on Wireless Communication Systems, September 2009, Siena, Italy.

[14] D. Gay, P. Lewis, R. von Behren, M. Welsh, E. Brewer, and D. Culler, "The nesC language: A holistic approach to networked embedded systems", Proceedings of the ACM SIGPLAN 2003 conference on Programming language design and implementation, San Diego, USA.

Local Time Synchronization in Bluetooth Piconets for Data Fusion Using Mobile Phones

J. Wåhslén, I. Orhan, T. Lindh
Body Sensor Networks 2011 International Conference
May 2011, Dallas.

2011 International Conference on Body Sensor Networks

Local Time Synchronization in Bluetooth Piconets for Data Fusion Using Mobile Phones

Jonas Wåhslén
School of Technology and Health
KTH
Jonas.Wahslen@sth.kth.se

Ibrahim Orhan
School of Technology and Health
KTH
Ibrahim.Orhan@sth.kth.se

Thomas Lindh
School of Technology and Health
KTH
Thomas.Lindh@sth.kth.se

Abstract - **This paper presents a method to synchronize the clocks in a Bluetooth piconet from the application layer in a mobile phone. It adapts algorithms for time synchronization of distributed systems and the Internet to Bluetooth networks. The performance issues that cause problems for data synchronization between master and slaves in Bluetooth are highlighted. The tests show that the synchronization error is limited to one sampling time.**

I. Introduction and Related Work

Applications in mobile phones that collect and analyze data from multiple wireless sensors have become wide-spread today e.g., in sports, leisure and for physical activities in general. In this paper we present a method to synchronize the clocks between the master and the slaves in a Bluetooth piconet in order to perform data fusion of data from multiple sensors with correct timestamps. One example is collecting data from several inertial sensors placed along the legs of a cyclist to measure the ankle and knee angles and provide feedback to optimize performance [1]. Another application, also estimating the knee angle from wireless inertial sensors, is gait analysis e.g., for recovery and rehabilitation after a knee surgery.

The main problem studied in this paper is to associate samples from different sensors that are taken at approximately the same time in order to perform data fusion. One approach is use the Bluetooth clock to synchronize the slaves to the master [3] or other layer-2 techniques. Synchronized time division access protocols can also be used [1]. A comprehensive survey of time synchronization in wireless sensor networks can be found in [4]. Our approach is to achieve data synchronization from the application layer, without having access to the internal Bluetooth stack. It is sometimes not possible or even desirable, to program the wireless sensor nodes. In previous work we have presented an algorithm on the mobile phone for data synchronization without any time information in the data samples sent from the sensor nodes [5]. In this paper we propose a method to synchronize the sensor nodes' clocks using a combination of methods for distributed systems (see brief overview in [6]) and the Network Time Protocol (NTP) [7] for the Internet, adapted to Bluetooth and mobile phones.

The paper is organized as follows. Section II introduces our method for time synchronization and the theory behind it. Section III highlights some of the performance issues in Bluetooth that make synchronization of data from multiple sensors on a mobile phone difficult, as described in Section IV. The implementation, tests, discussion of results and conclusions are found in Section V to Section VII.

II. Time Synchronization from the Mobile Phone

The main idea in this paper is to synchronize the slaves' clocks to the master's clock in a Bluetooth piconet. As can be seen in Fig. 1, a master places a timestamp, T_1, in a synchronization request message, $m_{request}$, sent to a slave that hosts a sensor. After receiving $m_{request}$, the sensor node inserts its local time, T_2, into the return message, m_{reply}, before returning it to the master. Timestamp T_3 is stored by the master when m_{reply} is received. The master can, based on these timestamps, determine the offset between its own clock and the slave's clock, and accordingly synchronize the two clocks. When the master receives a data sample from the slave, including its local timestamp, it adds the estimated offset to obtain the common synchronized time.

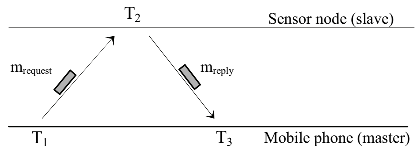

Fig. 1. The figure shows a synchronization message sent between a mobile phone and a sensor node, and the timestamps set by the master and slave.

Instead of letting the slave act as a client and request the master's time, compute the offset and set its clock, the master performs all calculations and implements the synchronization. We have also preferred to use only one timestamp on the slave (T_2) for the same reason, to save processing and energy for the slaves.

Let T_1 and T_3 be timestamps set by the master, T_2 be a timestamp set by the slave, T_{round} be the estimated round-trip time (RTT), and D_{min} be the minimum delay time for $m_{request}$ from the master to the slave. The earliest time $m_{request}$ can reach the slave is T_2+D_{min}, and the latest time $m_{request}$ can reach the slave is T_2+T_{round}-D_{min}. The maximum error in estimating the time for $m_{request}$ to reach the slave is ±(T_{round}/2-D_{min}). The original idea in Cristian's algorithm [8] for probabilistic clock synchronization in distributed systems is that a server sends its time, t_{server}, as a reply to a request from a client. The client sets its clock to t_{server}+T_{round}/2. Choosing T_{round}/2 as the delay be-

978-0-7695-4431-1/11 $26.00 © 2011 IEEE
DOI 10.1109/BSN.2011.11

IEEE computer society

tween the readings of T_1 and T_2 minimizes the maximum error. Since the maximum error depends on T_{round}, Cristian suggests discarding measurements of the round-trip time above a threshold, $T_{round}>D_{min}+\varepsilon$, where ε is the maximum error. To discard the largest values and take the average of the remaining RTT measurements is also a possibility.

In our case (Fig. 1), $T_{round}=T_3-T_1$, and the estimated offset, o, between the clocks is the difference between T_1 and T_2 plus the delay between the readings of the clocks (half of the round-trip time). Hence, the offset o is

$$o=(T_1-T_2)+T_{round}/2=T_1-T_2+(T_3-T_1)/2=(T_1-T_2+T_3-T_2)/2 \qquad (1)$$

The master adds this offset to the local timestamp inserted by the sensor node along with the data samples. Cristian also takes the clock drift rate into account, which can be compensated for by repeating the synchronization process with certain intervals.

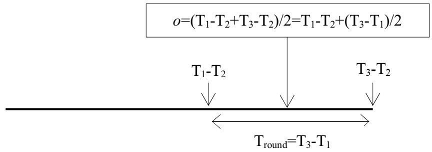

Fig. 2. The figure shows how the offset of the slave's clock relative to the master's clock and the round-trip time is estimated. The error limit for the offset is $\pm(T_{round}/2-D_{min})$.

As a measure of the data synchronization error we define the error distance, $\varepsilon_{distance}$, as the number of sampling intervals that separates samples from sensors taken at the same time, where "same time" means a distance less than a half sampling interval. Samples from sensors that are correctly synchronized have a maximum distance of half a sampling interval of the sensor with the highest sampling frequency. In NTP, a client applies the Marzullo algorithm [9], or refined versions of it, to select the optimal estimate of offset and round-trip delay from a number of time servers. In our case, Marzullo's algorithm can be used to find an optimal offset, o, with the uncertainty margin RTT/2 (Fig. 2) among a set of estimates. In practice it means to find the maximum T_1-T_2 and the minimum T_3-T_2 among a number of overlapping offset intervals (see Fig. 3). We define this smallest RTT as

$$T_{round\text{-}min}=\min(T_3-T_2)-\max(T_1-T_2), \qquad (2)$$

where $T_{round\text{-}min}/2$ is an estimate of D_{min}.

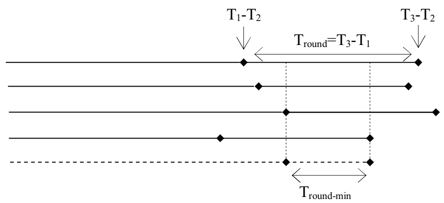

Fig. 3. The figure illustrates Marzullo's algorithm to select an offset, o, with the smallest uncertainty interval, $o\pm T_{round\text{-}min}/2$, where o is the centre point of the interval $T_{round\text{-}min}$.

III. Bluetooth Performance Issues

Bluetooth is today the dominating wireless standard to connect sensors to mobile phones. We have observed several performance issues in Bluetooth that makes synchronization of data from an application in a mobile phone unreliable and complicated. In this section we present results using the testbed in Fig. 4. Trace files captured by a Bluetooth sniffer (Frontline [10]) have been used in our analysis.

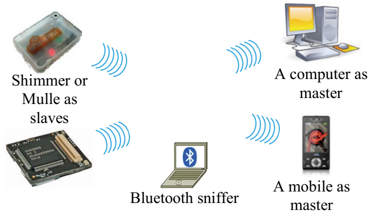

Fig. 4. The testbed consists of a Shimmer or a Mulle sensor node as slaves and a computer or a mobile phone as master. A Bluetooth sniffer is placed between the master and the slaves to capture the traffic including the setup phase and the control packets.

A. Sleep Time

In Bluetooth a master or a slave can refrain from sending in order to save energy. Slaves that have data to send may therefore be forced to buffer packets until they receive a PULL or NULL packet from the master, which permits them to transmit data. We have used USB Bluetooth dongles from two different vendors (dongle A and B) connected to a computer as master and a Shimmer [11] sensor node as slave. The traffic between the slave and dongle A as master is shown in Fig. 5 and the traffic between dongle B as master and the slave is shown in Fig. 6. The Shimmer sensor node, running TinyOS-2.1, sends samples at 100Hz to the Bluetooth chip via the UART.

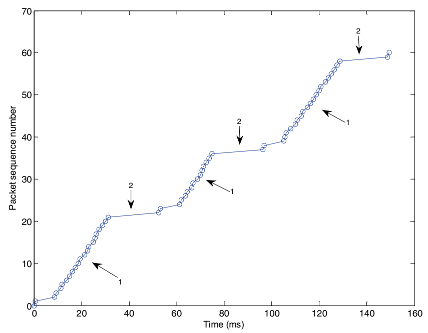

Fig. 5. Packets sent between a master and a slave. The Bluetooth USB dongle A acts as master and the Shimmer sensor as slave. The figure shows periods (labeled 2) when no packets are sent.

A Java application on the computer receives the samples from the sensor node. The circles in the sections labeled 1 in Fig. 5 and Fig. 6 represent packets sent between the master and the slave. However, in the periods labeled 2, no packets are transmitted. Our explanation for these gaps in the traffic pattern in Fig. 5 and Fig. 6 is that the master is going to sleep in order to save energy. This occurs when the master has sent a number of NULL packets to the slave without any response from the slave.

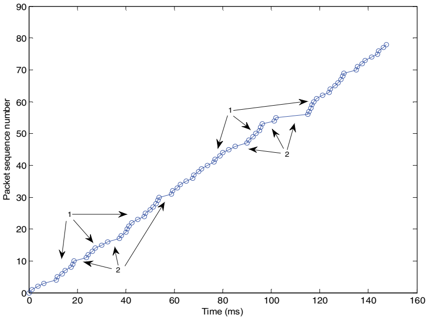

Fig. 6. Packets sent between a master and a slave. The Bluetooth USB dongle B acts as master and the Shimmer sensor as slave. The figure shows periods (labeled 2) when no packets are sent.

The observed sleep time for Bluetooth dongle A (Fig. 5) is around 20ms. Since the sampling rate is 100Hz; the slave's buffer will contain two samples. The other master (Fig. 6) has the same behavior but the idle periods are shorter. Even though the slave has data to transmit during the sleep time, it has to wait for a NULL or POLL packet from the master to be able to transmit again. The samples are sent by the operating system to the UART every 10ms (f_s=100Hz), but the transmission times are not periodic. The variation in inter-sending times using the dongle A can be seen in Fig. 7.

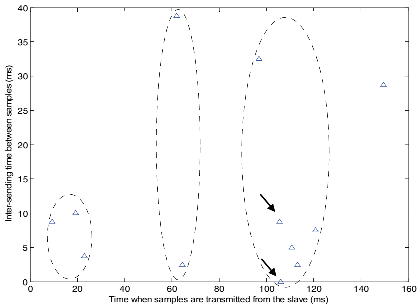

Fig. 7. The sending times for samples on the x-axis and the time from the previous sample on the y-axis using the BT dongle A. The sampling time is constant (T_s=10ms, f_s=100Hz) but the slave transmits the samples in bursts separated by gaps in sleep time periods. The ellipses correspond to those parts in Fig. 5 labeled 1. The gaps between these bursts occur when the master sleeps to save energy (labeled 2 in Fig. 5).

The Bluetooth sniffer also verifies that a sample from a sensor can be transmitted in different packets. The slave may send the first fragment of a sample in a 3-slots packet, and the remaining fragment of the sample and next sample in the following 3-slots packet. This happens when the last packet is sent before the master enters an idle period. Very rarely, the entire sample is sent in a 3-slots packet, even though it fits into the payload. The Bluetooth chip in the Shimmer device does not fill the entire payload before transmission. The rest of the sample will be sent in the next packet after sleep time period.

B. *Scheduling of Multiple Sensors*

In the previous section we showed that the master's behavior during the idle period leads to unpredictable delays between the samples' transmission times. Another reason for stochastic delays between the actual sampling times and the sending times is the number of slaves that are active. In this section we present the traffic pattern and inter-sending sample times of one and two Mulle [12] sensor nodes connected to a mobile phone. A single Mulle sensor gives a rather stable and periodic transmission pattern of samples to the mobile phone (blue line of triangles in Fig. 8). The inter-sending times of the samples are the same as the sampling intervals, 10ms (100Hz sampling rate). Furthermore, the master sends POLL packets all the time without any sleep time and the client is sending high data rate (DH) packets or NULL packets as reply to the POLL packets. The gaps in Fig. 5 and Fig. 6 due to the sleep time do not exist at all in Fig. 8 (blue triangles). The master sends POLL packet continuously and the slave responds with DH packets that contain the entire samples and not fragments, which was the case for the Shimmer sensor node described in Section III.A. However, if a second slave is added to the piconet, the traffic pattern between the master and slave changes (x-marks in Fig. 8). We can now identify idle periods and the constant inter-sending times are not maintained. The transmission times become more scattered and samples are often sent in bursts, interleaved by gaps. The inter-sending times will vary depending on how master's scheduler is implemented by the Bluetooth vendor. If multiple slaves are connected, a master will not necessarily run a round-robin algorithm when requesting data from the slaves. Therefore, some samples will be buffered by a slave before transmitted to the master. Fig. 9 shows sequence numbers and timestamps for every sample represented by triangles and squares from the two respective slaves. The encircled samples show that a sensor can send several samples in a row before the other sensor transmits its sample(s). It is apparent that a round-robin scheduling is not utilized.

Beside the idle periods (Section III.A) and the scheduling of multiple sensors discussed above, there are several other performance issues that result in data synchronization problems. One example is that a slave can refrain from transmitting a packet even it has received a POLL packet from the master. A slave avoids sending small packets in order to save energy. If the buffer is not sufficiently filled, the slave postpones the transmission. Another examples is that the preferred packet format (medium data rate (DM) or high data rate (DH) using 1, 3 or 5 timeslots) affects the performance. A DH packet is not protected by forward error correction (FEC), which leads to retransmissions and increased synchronization problems in a noisy or congested channel.

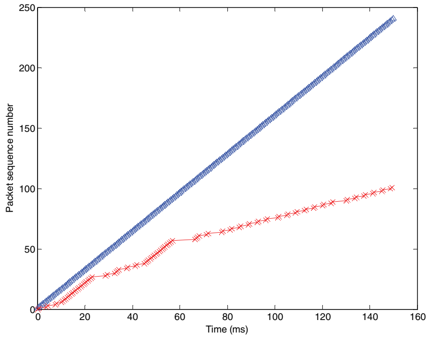

Fig. 8. Packets sent between a mobile phone and one (blue triangles) or two (red x-marks) Mulle sensor node(s) as slave(s).

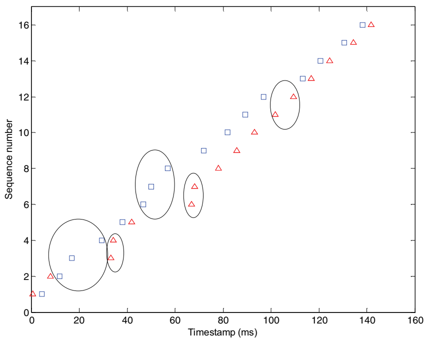

Fig. 9. Timestamps and sequence numbers for a number of samples from two sensors (blue squares and red triangles) connected to the mobile phone during 160ms.

IV. Data Synchronization Problems in Mobile Phones

Synchronization of data from multiple sensors using timestamps in mobile phones is often unreliable. The fundamental reason is the stochastic delay, Δt, between the sampling time on the sensor node and the time the sample is read by the application from the buffer on the mobile phone. Fig. 10 shows the two buffers where the packets may be delayed.

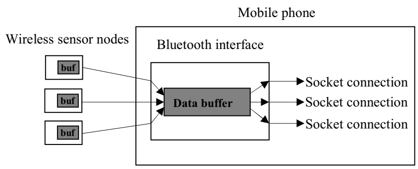

Fig. 10. A block diagram of how a mobile phone handles data input from wireless sensors. Data from the sensors pass two buffers; the outgoing data buffer on the sensor, and the incoming data buffer on the mobile phone.

We have found that the main contribution to Δt is the variable waiting time in the outgoing buffers on the sensor nodes. High sampling frequency and many slaves can however lead to waiting times in mobile phones with low capacity. As described in Section III, the stochastic waiting time in the sending buffers is due to the behavior of the master and slave in the piconet e.g., sleeping times and scheduling. Fig. 11 shows the un-calibrated acceleration data from two sensor nodes with 300Hz sampling rate during 500ms at the top, and a blow up of the encircled part during 50ms at the bottom. All samples are timestamped when the application on the mobile phone reads them from the incoming buffer. Both sensor nodes are tied together in order to experience exactly the same acceleration when moved. The anticipated periodic intervals of around 3ms between the samples are replaced by a more random distribution. Data fusion based on these timestamps will fail. The synchronization error distance, $\varepsilon_{distance}$ defined in Section II, is approximately 6 sampling intervals (there are 6 samples from one sensor between two consecutive samples from the other sensor). Furthermore, this error distance will increase if the number of sensor nodes or the sampling frequency increases. In previous work, we have developed an algorithm that reduces the synchronization error to around 1-2 sampling intervals [5]. However, the error increases if the sensors have different sampling frequencies. In this paper we propose an alternative approach where the sensor nodes' clocks are synchronized to the master's clock locally in a Bluetooth piconet by an application program.

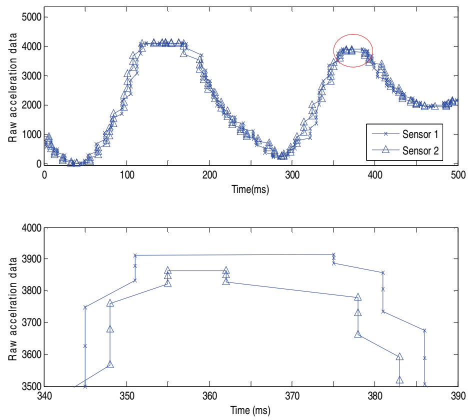

Fig. 11. On top the un-calibrated acceleration value (y-axis) during 500ms (x-axis) from two wireless sensor nodes that are tied together. A blow up of the encircled part is shown in the lower part of the figure. The sampling rate for both sensors is 300Hz. The samples are timestamped on the mobile phone.

V. Implementation and Tests

We have implemented a local Bluetooth piconet time synchronization from an application program on a mobile phone based on the method described in Section II, illustrated in Fig. 1 - Fig. 3. The synchronization application program, 30 lines of code written in Java Micro Edition, on the mobile phone (Ericsson k800i) sends the $m_{request}$, receives the m_{replay} and performs the algorithm to calculate the offset o for each connected sensor node. The synchronization code in the Shimmer sensor nodes, running TinyOS-2.1, consists of 7 lines of code in nesC that primarily receives $m_{request}$ and inserts a timestamp in the response m_{reply}. The algorithm consists of the following steps.

i) The mobile phone sends $m_{request}$ to the sensor node and stores its timestamp T_1.

ii) The sensor node receives $m_{request}$ and inserts its local timestamp T_2 in the response m_{reply}.

iii) The mobile phone receives m_{reply} and stores its timestamp T_3.

iv) This is repeated to get a sufficient number of measurements (30-50 in our tests).

v) The mobile phone calculates the offset o according to Equation (1) in Section II and Fig. 2 and stores the average value (after deleting the highest outliers).

vi) Step *i* to step *v* is repeated for each connected sensor node.

vii) The estimated offset value is added by the mobile phone to the timestamp in each sample received from the respective sensor. Hence, all incoming samples will have a synchronized timestamp.

Every loop in step *iv* takes less than 50ms, which means that a run of 30 loops in completed in less than 1.5s. The result is showed in Fig. 12. The test configuration is the same as for the test resulting in Fig. 11. The only difference is that our synchronization algorithm is used in Fig. 12. We have repeated the test a large number of times. The results clearly verify that the maximum error distance, $\varepsilon_{distance}$, defined in Section II as the number of sampling intervals that separates samples from sensors taken at the same time, is in the order one sampling time T_s. Note that it is not possible to obtain a maximum $\varepsilon_{distance}$ below $T_s/2$ even though the synchronization is perfect. The start of the sampling process in each sensor is not coordinated. The sampling times will therefore have a maximum phase shift of a half sampling interval (of the highest sampling frequency).

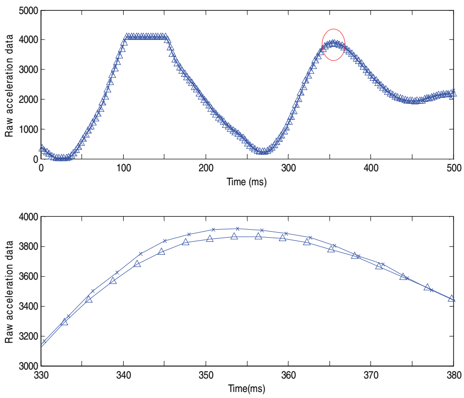

Fig. 12. On top the un-calibrated acceleration value (y-axis) during 500ms (x-axis) from two wireless sensor nodes that are tied together. A blow up of the encircled part is shown in the lower part of the figure. Both sensor nodes are sampling in 300Hz. The sensors' clocks are synchronized using our method.

For comparison, the Marzullo approach of estimating the offset value and the smallest $T_{round\text{-}min}$, Equation (2) in Section II, is also implemented. This approach also has the advantage of saving processing time and capacity on the mobile phone. Instead of calculating the average offset from 30-50 loops it suffices to store two values, $\min(T_3\text{-}T_2)$ and $\max(T_1\text{-}T_2)$, during the entire calculation. Fig. 13 shows 15 estimates of the offset, the RTT and how the $T_{round\text{-}min}$ is determined.

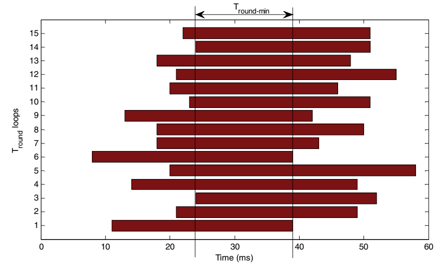

Fig. 13. Offset times between a master and a sensor node for 15 test loops. The bars along the x-axis show the T_{round} part of the offset. The right end of the bars is T_3-T_2, and the left end is T_1-T_2. The middle point of each bar represents the estimated offset $o=(T_1\text{-}T_2)+T_{round}/2$. See also Fig. 2. The interval $T_{round\text{-}min}$ is an estimate of the smallest T_{round}.

The validation shows a difference in estimated offset of around 1/3ms in between taking the average of the measured offsets and $T_{round\text{-}min}$ approach. As the difference is smaller than the sampling period $1/f_s$ (300Hz in this case) it is difficult to evaluate the accuracy of the two algorithms. However, it can be observed the offset using $T_{round\text{-}min}$ approach were on average 2ms smaller than taking the average. The probable explanation is that it takes on average 2ms for the sensor node to process $m_{request}$, insert a timestamp and send m_{reply} to the master.

VI. Discussion

The uncertainty in estimating the offset o is $\pm(T_{round}/2\text{-}D_{min})$, where D_{min} is the shortest delay for $m_{request}$ from the master to the slave. This round-trip time consists of four timeslots plus processing and waiting time in the master and the slave. The minimum delay that cannot be reduced is four 1-slot timeslots for sending messages: $m_{request}$ in the first timeslot from the master to the slave, next timeslot from slave to master cannot be used since the reply message is not ready, a POLL or NULL packet from the master in the third timeslot from the master, and finally is m_{reply} sent in the fourth timeslot from the slave. Four 1-slot timeslots in Bluetooth takes around 2.5ms (4×625μs). The time for the sensor node to process $m_{request}$, insert a timestamp and place m_{reply} in the outgoing buffer is less than a timeslot in our tests. The main contribution to the delay, in addition to the four timeslots and the processing time, is the varying waiting times in the outgoing and incoming buffers. As can be seen in Fig. 14, the RTT varies depending on the number of other sensors that transmit simultaneously. The background traffic from these additional nodes consists of packets with 22 byte payload sent at 300Hz. The bars in Fig.

14 (from left to right) represent: T_{round} average, the confidence interval (0.99 level) of the offset, the confidence interval (0.99 level) of T_{round}, and the average and the confidence interval (0.99 level) of $T_{round\text{-}min}$.

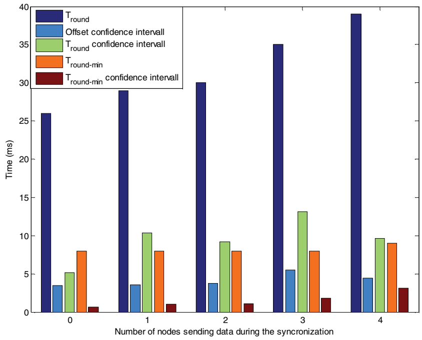

Fig. 14. The measured RTT (T_{round}) for the synchronization messages between a master and a slave when zero, one, two, three and four other sensor nodes are transmitting data. The bars on the x-axis show: T_{round} average, the confidence interval (0.99 level) of the offset, the confidence interval (0.99 level) of T_{round}, and the average and the confidence interval (0.99 level) of $T_{round\text{-}min}$.

$T_{round\text{-}min}/2$ can be used as an estimate of D_{min}. The maximum uncertainty in estimating the offset o is ± ($T_{round}/2$-D_{min})., as described in Section II. If the round-trip time is 20ms and $T_{round\text{-}min}$ is 10ms, then the worst case error is ±5ms. For round-trip times that are nearly symmetric the maximum error becomes small.

The clock drift was negligible during our relatively short test periods. However, in normal operation it is necessary to recalibrate the clock offsets between the master and the slaves with certain intervals. Resynchronization takes less than 1.5 second.

As seen in Fig. 14, T_{round} will increase when several other nodes transmit simultaneously. The accuracy of our test results has not been affected by this.

In future work we plan to use the algorithm to synchronize on-body sensors in sports applications where data from multiple sensors are processed to provide feedback to athletes [13].

VII. CONCLUSIONS

We have presented a method for local time synchronization in a Bluetooth piconet from an application program in a mobile phone. The algorithm, based on ideas from distributed computer systems and network synchronization, is implemented for wireless sensors connected to a mobile phone. The test result accuracy is better than one sampling interval. One application is processing of data from multiple sensors worn by athletes to provide feedback for improved training and competition performance.

REFERENCES

[1] P. Havinga, R. Begg, M. Palaniswami, R. Marin-Perianu, M. Marin-Perianu, D. Rouffet and S. Taylor, “Body Area Wireless Sensor Networks for the Analysis of Cycling Performance”, BodyNets’10, Corfu, Greece, September 2010.

[2] S. Chen, J.S. Brantley, T. Kim and J. Lach, “Characterizing and Minimizing Synchronization and Calibration Errors in Inertial Body Sensor Networks”, BodyNets’10, Corfu, Greece, September 2010.

[3] M. Ringwald and K. Römer, “Practical Time Synchronization for Bluetooth Scatternets”, BROADNETS 2007, Raleigh, North Carolina, USA, September 2007.

[4] K. Römer, P. Blum and L. Meier, 2Time Synchronization and Calibration in Wireless Sensor Networks”, in Ivan Stojmenovic (Ed.): Handbook of Sensor Networks: Algorithms and Architectures. John Wiley & Sons, ISBN 0-471-68472-4, pp. 199-237, September 2005.

[5] J. Wåhslén and T. Lindh, “A Novel Approach to Multi-Sensor Data Synchronization Using Mobile Phones”, BodyNets’10, Corfu, Greece, September 2010.

[6] G. Coulouris, J. Dollimore and T. Kindberg, “Distributed Systems: Concepts and Design”, Chapter 11, 4th edition, Addison-Wesley, 2006.

[7] D. Mills, J. Martin, J. Burbank and W. Kasch, “Network Time Protocol Version 4: Protocol and Algorithms Specification”, RFC 5905, Internet Engineering Task Force (IETF), June 2010.

[8] F. Cristian, “Probabilistic clock synchronization”, Distributed computing (1989): pp. 146-158.

[9] K. Marzullo and S. Owicki, “Maintaining the time in a distributed system”, ACM Operating Systems Review 19, 3 (July 1985), 44-54.

[10] Frontline, http://www.fte.com

[11] Shimmer Research, http://www.shimmer-research.com

[12] Mulle sensor node, http://staff.www.ltu.se/~jench/mulle.html

[13] D. Sturm, K. Yousaf and M. Eriksson, “A wireless, Unobtrusive Kayak Sensor Network Enabling Feedback Solutions”, International Conference on Body Sensor Networks (BSN 2010), Singapore, June 7 - 9, 2010.

Performance Evaluation of Time Snchroniation and Clock Drift Compensation in Wireless Personal Area Networks

J. Wåhslén, I Orhan, D. Sturm and T. Lindh

Submitted to 7th International Conference on Body Area Networks
September 2012, Oslo, Norway

Performance Evaluation of Time Synchronization and Clock Drift Compensation in Wireless Personal Area Networks

Jonas Wåhslén Ibrahim Orhan Dennis Sturm Thomas Lindh
School of Technology and Health
KTH
Email:{jonas.wahslen; ibrahim.orhan; dennis.sturm; thomas.lindh}@sth.kth.se

ABSTRACT
Efficient algorithms for time synchronization, including compensation for clock drift, are essential in order to obtain reliable fusion of data samples from multiple wireless sensor nodes. This paper evaluates the performance of algorithms based on three different approaches: one that synchronizes the local clocks on the sensor nodes, and a second that uses a single clock on the receiving node, (e.g. a mobile phone), and a third that uses broadcast messages. The performances of the synchronization algorithms are evaluated in wireless personal area networks, especially Bluetooth piconets and ZigBee/IEEE 802.15.4 networks. A new approach for compensation of clock drift and a realtime implementation of single node synchronization from the mobile phone are presented and tested. Finally, applications of data fusion and time synchronization are shown in two different use cases: a kayaking sports case, and monitoring of heart and respiration of prematurely born infants.

Keywords
Time synchronization, clock drift, data fusion, Bluetooth, IEEE 802.15.4.

1. INTRODUCTION
Correctly performed data fusion is crucial for applications in sports, medicine, health and many other fields. Samples of data from multiple sensors are received by a coordinating node such as a mobile phone. Data fusion requires that samples from multiple sensors have a time reference that is synchronized and comparable for all connected nodes. However, even small differences in clock frequency among the nodes may lead to unacceptable errors after some time. Compensation for clock drift is therefore a necessary component in time synchronization. Today, the main radio technologies for communication between wireless sensors and coordinating nodes are Bluetooth and IEEE 802.15.4. In this paper we evaluate and compare the performance of different time synchronization algorithms in Bluetooth piconets as well as ZigBee/IEEE 802.15.4 networks. In addition, we develop and test a new technique to compensate for clock drift to preserve accuracy in data fusion.

The paper is organized in the following way. Section 2 covers different methods for time synchronization and clock drift compensation. Performance evaluation results from Bluetooth piconets and IEEE 802.15.4 networks are shown in Section 3. Section 4 discusses applications in a kayaking sports use case and a neonatal care use case.

2. ALGORITHMS
This section briefly describes the algorithms for synchronization (Section 2.1) and clock drift compensation (Section 2.2) that are evaluated in Section 3.

2.1 Synchronization Algorithms
Two different approaches for time synchronization in wireless sensor networks are described in this section: firstly to coordinate the local clocks in each of the sensor nodes, and secondly to use the clock of a single central node, e.g. a mobile phone, to synchronize data. A comprehensive survey of time synchronization in wireless sensor networks can be found in [1].

2.1.1 Synchronizing the Clocks in Sensor Nodes
Algorithms for synchronization of local clocks in wireless sensor networks often apply a combination of methods for distributed systems [2] and the Network Time Protocol [3] for the Internet. The algorithm used in the performance evaluation in Section 3 is described in detail in [4]. The main idea is to synchronize the sensor nodes' (slaves') clocks to a mobile phone (master) clock in a Bluetooth piconet. Fig. 1 illustrates the algorithm: The master stores a timestamp, t_1, and sends a synchronization request message, $m_{request}$, to a slave that hosts a sensor. After receiving $m_{request}$, the sensor node inserts its local time, t_2, into the return message, m_{reply}, before transmitting it back to the master. The master generates timestamp t_3 when m_{reply} is received. The master can, based on these timestamps, determine the offset between its own clock and the slave's clock, and accordingly synchronize the two clocks. When the master during consecutive data acquisition receives a sample from the slave, including its local timestamp, it adds the estimated offset to obtain the common synchronized time. The master performs all offset calculations and implements the synchronization, slaves are passive and simply respond to requests.

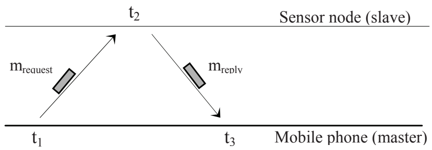

Fig. 1. The figure shows a synchronization message sent between a mobile phone and a sensor node, and the timestamps set by the master and slave.

Timestamps t_1 and t_3 are set by the master and timestamp t_2 is set the by slave. Let t_{round}=t_3-$t_1$be the estimated round-trip time and D_{min} the minimum delay time for $m_{request}$ from the master to the slave. The earliest time $m_{request}$ can reach the slave is t_2+D_{min}, and the latest time $m_{request}$ can reach the slave is $t_2+t_{round}-D_{min}$. The maximum error in estimating the time for $m_{request}$ to reach the slave is $\pm(t_{round}/2-D_{min})$. The original idea in Cristian's algorithm [5] for probabilistic clock synchronization in distributed systems is that a server responds with its time, t_{server}, as a reply to a request from a client. The client sets its clock to $t_{server}+t_{round}/2$. Choosing

$t_{round}/2$ as the delay between the readings of t_1 and t_2 minimizes the maximum error.

Instead of letting the slave act as a client and request the master's time, compute the offset and set its clock, the master performs all calculation and implements the synchronization. The estimated offset, *o*, between the clocks is the difference between t_1 and t_2 plus the delay between the readings of the clocks (half of the round-trip time), as shown in Fig. 2. The master adds this offset to the local timestamp inserted by the sensor node along with the data samples.

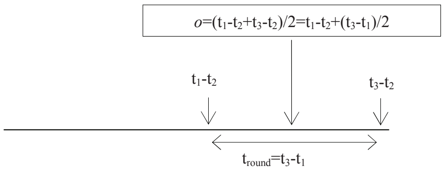

Fig. 2. The figure shows how the offset of the slave's clock relative to the master's clock and the round-trip time is estimated. The error limit for the offset is ±(t_{round}/2-D_{min}).

An alternative method to coordinate the local clocks in sensor nodes is to use broadcast messages. This approach is further developed and evaluated in Section 3.2.

2.1.2 *Single Clock Synchronization*

Instead of coordinating the clocks in the sensor nodes, an alternative approach is to rely on the single clock in the receiving node, such as a mobile phone ([1], [6] and [7]). Fig. 3 shows an application in a mobile phone that reads samples from wireless nodes.

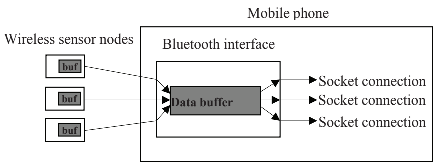

Fig. 3. An application in a mobile phone reads samples from three sensor nodes. Data from the sensors pass two buffers; the outgoing data buffer on the sensor, and the incoming data buffer on the mobile phone.

The accuracy of data synchronization based on timestamps by the mobile phone (Fig. 3) depends on a variable time period, Δt, between the actual sampling time on the sensor node and the time the sample is read from the buffer on the mobile phone. Let t_{mobile} be the time set by the application program on the mobile phone, and t_{sensor} be the time when the sample was acquired by the sensor node. If the clocks on the sensor node and the mobile phone are synchronized in time, then Δt for sample k will be $\Delta t(k) = t_{mobile}(k) - t_{sensor}(k)$. The main idea is to identify samples with the smallest Δt, and use the result to recalculate the timestamps for all samples. The method estimates the minimum Δt that consists of the fixed part of the delay i.e. processing time, sending time, propagation time etc. An algorithm that identifies samples with minimum Δt has been implemented [7]. It uses the observation that the last sample in the incoming queue has spent the shortest time in a buffer. The algorithm used in the performance evaluation in Section 3 can be summarized in the following steps for each connected sensor.

i) Let the samples be timestamped when read by the application program from the incoming buffer on the mobile phone.
ii) Identify a set of samples that is likely to have a minimum Δt.
iii) Calculate the average offset from this set of samples to correct the timestamps of the remaining samples.

The original algorithm in [7], which applied linear regression, has been improved to satisfy real-time requirements.

2.2 Clock Drift Compensation

The relation between two nodes local time, $t_{sensor1}$ and $t_{sensor2}$ can be modelled as a linear equation defined as

$$t_{sensor1} = o + dt_{sensor2} \qquad (1)$$

where *o* is the clock offset and *d* is clock drift. This clock drift is hardware-related and mainly caused by inaccuracy in the clock-crystal frequency. Small variations can occur due to for example power levels, temperature or voltage changes. A straightforward way to estimate the drift parameter *d* in Equation 1 is to retrieve timestamps from two sensors at two different time points (separated by a sufficient time interval). If multiple values are retrieved linear regression can be used. Resynchronization or an algorithm to compensate the drift may then be applied. Drift compensation can be implemented either by multiplying the local node by the estimated drift parameter, or by continuously adjusting the node's local time when necessary.

Clock drift compensation can be achieved by defining a virtual clock that runs with the estimated drift parameter *d* taken into account. We assume a resolution of 1ms for the local and the virtual clocks in the following. The sensor node clock with slowest clock (the lowest frequency) is used as a reference for the other sensor nodes drift compensation. When the node detects that the virtual clock differs from the local clock more than 1ms, the virtual clock time is subtracted by 1ms.

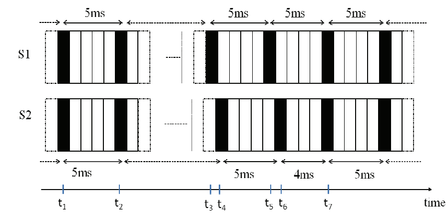

Fig. 4. The figure shows two sensors sampling at 200Hz at the same time (t_1, t_2). Due to clock drift the sampling rate is shifted (t_3, t_4 and t_5, t_6). The drift compensation function decreases the inter-sample time with 1ms for S_2 at t_6 when it detects a drift of at least 1ms. At t_7 the sensors sample at the same time again.

Fig. 4 shows an example of how the drift compensation algorithm works. At t_1 and t_2 both sensor nodes, S_1 and S_2, are reading samples at exactly the same (synchronized) time with the sampling frequency f_s=200Hz (T_s=5ms). At t_3 the sample time for S_1 and S_2 are separated and partly overlapped at t_4. The time difference between the two local clocks at t_5 and t_6 has reached 1ms. The algorithm will now decrease the inter-sampling time T_s for S_2 by 1ms. The result is that the two sensors pick samples simultaneously at t_7. The algorithm will continuously monitor and control the clock drift.

3. PERFORMANCE EVALUATION RESULTS

This section presents results of performance evaluation of the algorithms outlined in the previous section. The testbed is described in Section 3.1 and results from tests in Bluetooth piconets and ZigBee/IEEE802.15.4 networks are shown in Section 3.2 and Section 3.3. Finally the algorithm for clock compensation is evaluated in Section 3.4.

3.1 Experimental Setup

The results presented in this section are obtained from a common testbed with a coordinator (master) and one up to six sensor nodes (slaves) from Shimmer Research [8]. These nodes are programmed in nesC (TinyOS 2.1.0 operating system) and are able to switch between a Bluetooth radio transceiver and a ZigBee/IEEE 802.15.4 radio transceiver.

3.2 Time Synchronization in Bluetooth Piconets

In this section we evaluate the performance of the two algorithms for synchronizing sensor nodes to a mobile phone (master) in Bluetooth piconets described in Section 2.1.

3.2.1 Synchronizing the Clocks in Sensor Nodes

The algorithm is evaluated by varying two parameters. Firstly, the number of connected sensor nodes, and secondly, the number of repeated loops of $m_{request}$ and m_{reply} (in Fig. 1), where each loop results in a set of timestamp t_1, t_2 and t_3. Two different cases are analysed regarding the way the sensor nodes are connected. Fig. 5 shows the synchronization error when nodes are simultaneously connected to the master during the synchronization procedure. In Fig. 6, each sensor node is synchronized to the mobile phone one at a time. The figures show the 99th percentile of the synchronization error in milliseconds for 1 up to 50 loops, where each test run is repeated a large number of times.

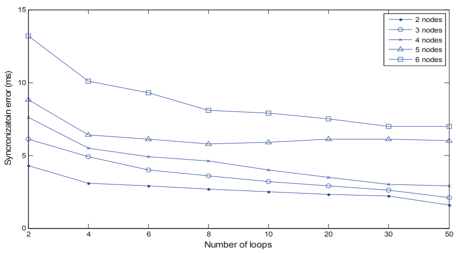

Fig. 5. Synchronization error in milliseconds when all nodes in a Bluetooth piconet are connected at the same time. The error decreases as the number of loops increases. The error increases when several nodes are connected.

From Fig. 5 and Fig. 6 it is obvious that the synchronization error decreases as the number of loops increases. No substantial improvement is gained by repeating the procedure more than 50 times, which takes approximately 1 second. It can often be interrupted after fewer loops. It is also evident that the error increases as the number of connected nodes increases. Furthermore, a substantially higher synchronization error can be observed when all nodes are connected at the same time (Fig. 5).

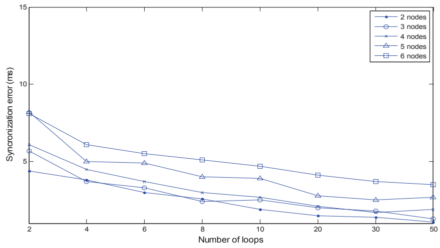

Fig. 6. Synchronization error in milliseconds when one node at a time is connected in a Bluetooth piconet. The synchronization error is approximately one half compared to Fig. 5, when the nodes all nodes are connected during the synchronization process.

This is a direct effect of the way Bluetooth controls multiple connections, where the master determines when a slave is permitted to transmit data by sending a null or pull packet to the slave. When a single sensor node is connected, the master only needs to check if that node has data to send. However, when multiple nodes are connected, the master will schedule the order that the nodes are permitted to transmit. A sensor node may have to wait before sending its sampled data. One reason is that the master visits every connected sensor (sends POLL or NULL packets) regardless of whether they have data to send or not. Another reason is that the scheduling is necessarily done in a round-robin fashion [4]. The additional error when all sensors are simultaneously connected during the synchronization procedure (Fig.5) is approximately twice the error when the sensors are synchronized one at a time (Fig. 6). For example, when 6 nodes are connected the time interval between every poll request from the master to a slave will be approximately 7ms.

3.2.2 Single Clock Synchronization from the Mobile Phone

The method described in Section 2.1.2 relies solely on the clock in the mobile phone for data synchronization and therefore does not require coordination of the sensor nodes' clocks. The synchronization error after 50 loops ranges from 1.1ms to 3.2ms (99th percentiles) depending on the numbers of nodes connected. This can be compared to 7ms error if all six nodes are connected simultaneously (Fig. 5) and 3.5ms error when the six nodes are connected one at a time (Fig. 6). The tests clearly show that the single clock synchronization approach (Section 2.1.2) results in significantly higher accuracy compared to synchronizing the clocks in multiple sensor nodes that are connected simultaneously. This is due to the fact that the response message m_{reply} (Fig. 1) with t_2 is blocked in the slaves' outgoing buffer until a later timeslot assigned by the master. The single clock synchronization method is of special interest when the sensor nodes' clocks are not accessible and if resynchronization is needed, e.g. due to clock drift. In the original algorithm [7], linear regression based on the complete set of samples is used. This study shows that subsets of the samples can feed the algorithm, which decreases the processing time considerably.

3.3 Time Synchronization in ZigBee/IEEE 802.15.4 Networks

In this section we present the result of time synchronization in IEEE 802.15.4 networks based on the method in Section 2.1.1 and an alternative method using broadcast messages.

3.3.1 *Synchronizing the Sensor Nodes' Clocks*

Fig. 7 shows the synchronization errors using the same test setup as in Section 3.2 with the exception that the radio standard is IEEE 802.15.4, configured for contention-based access (CSMA/CA), instead of Bluetooth.

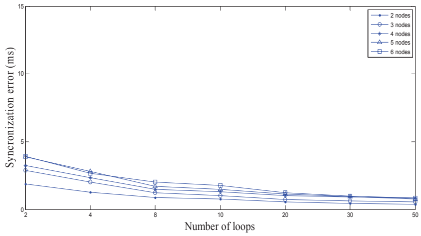

Fig. 7. The synchronization error in milliseconds for 1 to 6 participating sensor nodes and increasing number of loops in the synchronization procedure.

Clock synchronization according to the method in Section 2.1.1 gives considerably lower errors in IEEE 802.15.4 radio communication than in Bluetooth piconets after a few synchronization loops. Contrast to Bluetooth, contention-based access without a central scheduler means that a node is not forced to wait for a master poll request before transmitting the m_{reply} (Fig. 1). Applying CSMA/CA node merely checks if the media are idle prior to transmitting. The tiny synchronization messages will not result in major waiting times due to other nodes occupying the wireless channel. After 50 synchronization loops the error is approximately below 1ms even though all six nodes are active.

3.3.2 *Broadcast Message Synchronization*

The IEEE 802.15.4 radio standard supports broadcast messages multiple sensor nodes at the same time, which may be a feasible method for time synchronization. Our repeated tests have validated that measuring the time synchronization error using broadcast messages gives the same accuracy as using a hardware reset.

An algorithm inspired by the Flooding Time Synchronization Protocol [9] for broadcast synchronization was implemented as following.

i) The coordinating node sends a broadcast message to all sensor nodes.

ii) A sensor node stores its local time when receiving the broadcast message.

iii) The synchronized time can be obtained, by just subtracting the sensor actual time with the time stored in the variable. All slave sensors will then have the same time.

Even though the coordinating node may have to wait for the wireless channel to become idle, the sensor nodes will receive the broadcast message whenever it is sent. Our tests show that every sensor node will have the same synchronized time with a resolution of at least 1 millisecond. The coordinating node's clock can also be included by sending a broadcast message from one of the sensor nodes. Since a broadcast solution provides accurate time synchronization and is straightforward to implement there is no need for the methods described in Section 2.1 in ZigBee/IEEE 802.15.4 networks.

3.4 Results of Clock Drift Compensation

The clock drift has been measured and the algorithm for drift compensation has been evaluated in a testbed with IEEE 802.15.4 compliant Tmote Sky motes [13] that have the same microcontroller, MSP430, as in the Shimmer nodes. A sensor node, connected to a PC, is acts as coordinator for five sensor nodes. The coordinator sends a broadcast message to reset the sensor nodes' local clocks. The sensor nodes are requested to send their local time periodically (every 30 seconds), used by the coordinator as input for monitoring and control of the clock drift. The clock drift compensation method in Section 2.2 is implemented. Fig. 8 shows the implementation of drift compensating function when sampling. A periodic timer is implemented for how often the sensor node should sample. Variable x is used for inter sample time and y is used for compensating drift. When the timer expires, a request is sent to read the AD-channel sample value, using the HplAdc12 interface [14] available in MSP430, and it will be stored. If a clock drift is calculated, the variable y will be set to 1 otherwise 0. Setting the value y to 1 compensates the drift by letting the periodic timer run for 1ms less.

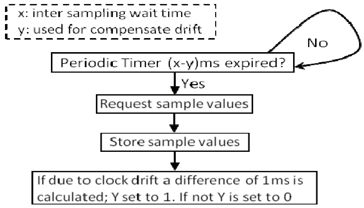

Fig. 8. A flow diagram for the drift compensating sampling algorithm.

Fig. 9 shows the maximum time difference between the two most drifting sensors (the sensor node with fastest and slowest clock) without drift compensation. The maximum time difference between 5 connected sensor nodes was approximately 5ppm. After 1 hour the time difference is 19ms.

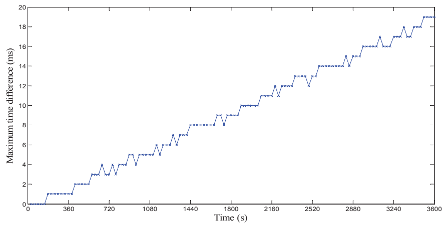

Fig. 9. The maximum time difference in ms between the two most drifting sensors (slowest and fastest clock) without drift compensating function.

Fig. 10 shows the effect of the drift compensating function. A maximum difference of 1ms is measured between two most drifting sensors measured at the testbed. The time from the two sensor node were transmitted every 30s during 1hour. The implementation of the algorithm results in a maximum clock drift of 1ms, which will preserve data fusion for sampling rates up to 1kHz.

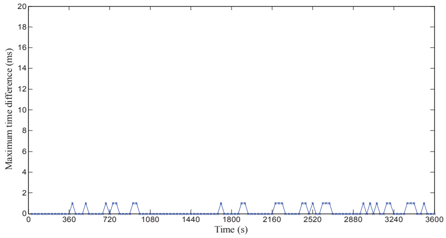

Fig. 10. The maximum time difference in ms between the two most drifting sensors (slowest and fastest clock) with drift compensating function.

3.5 Summary of Results

The performance evaluations in the previous sections show that the synchronization error is limited to around 1ms in ZigBee/IEEE 802.15.4 networks when a time synchronization protocol (Section 3.3.1) or broadcast messages (Section 3.3.3) are used. For Bluetooth, which so far is the prevailing communication technique between a mobile phone and wireless sensors, the errors increase as the number of connected nodes in a piconet increases. Our tests show that synchronization solely based on the clock in the mobile phone exhibits better performance than synchronizing the individual clocks in the sensor nodes (which sometimes is impossible). The algorithm to compensate for clock drift, due to variations in clock frequencies among the sensor nodes especially in low cost consumer equipment, has proven to be efficient (Section 3.4). Our study has resulted in performance limits (Fig. 5-7), in terms of synchronization errors, for prototype implementations of the algorithms in Section 2.

4. APPLICATIONS

Two real world use cases from two different fields, where time synchronization is important, are presented in this section.

4.1 Feedback in Kayaking Based on Synchronization

Flat water kayaking is an Olympic discipline in which a long and slim boat is propelled through the water by paddle equipped athletes. The motion in which the paddle transfers an athlete's force onto the water requires a complex, highly dynamic motor activity of the arms, shoulders, trunk and legs. Training supervision is ideally done by a coach, who primarily uses theoretical and personal practical knowledge and observation to derive subjective analysis and recommendations. Existing objective quantitative methods include time, distance measurement, heart rate measurement, GPS velocity tracking and video capturing. In a lab (e.g. on an ergometer) additional quantitative data can be accumulated by lactate and oxygen uptake measurement and motion analysis. All these methods are stand-alone technologies and can rarely be synchronized for a sophisticated data collection and analysis. Furthermore, there is only a small range of devices that can tolerate the conditions of on-water paddling; a desire for sophisticated on-water biomechanics measurement has been postulated by various researchers and coaches [10]. Based on the Mulle platform [11] a wireless sensor platform for on-water performance measurement on a kayak has been developed (Fig. 11).

Fig. 11. A Kayak system in action (left). Two paddle nodes (center, top) and the footstretcher (right, bottom) with an electronics box link via Bluetooth to a Java enabled mobile phone (right, top).

The system is designed to measure paddle and footstretcher force as well as motion parameters for real-time feedback (opportunity for motor learning through knowledge of performance) as well as post-processing. This platform as a successor of [12] consists of two Bluetooth enabled, battery powered paddle force measurement units with built-in accelerometer and gyro data acquisition as well, as a Bluetooth node that includes nine axis motion sensors, a 10Hz GPS and six force transducers for point-of-pressure measurement for the individual foot on the foot stretcher.

The Bluetooth radio transmitted data is received on a regular mobile phone running custom J2ME software for data storage, visualization and analysis. Data from each of the three nodes is sampled at 100Hz and transmitted in packages of 32 byte (hull node), respectively 23 bytes (paddle nodes) to the phone. Since a crucial component of athletic performance is the coordination, the timing, of different movements, it is absolutely essential that knowledge of the temporal resolution and the time instance, at which a data sample on one sensor node with respect to the other two sensor nodes' clocks is obtained, is accounted for. Before the data is stored or analyzed all three sensor nodes' samples have to be aligned with respect to a common valid timeline.

One goal is to detect the timing between the force on paddle and by the legs on the footstretcher. This parameter can be used to analyse the performance of an elite athlete and to give feedback on synchronized or unsynchronized activation of upper trunk (paddle) and leg musculature (footstretcher). Furthermore, the very crucial parameter of boat velocity is determined by the GPS supported by an accelerometer (Fig. 12), both situated on and connected to the hull node.

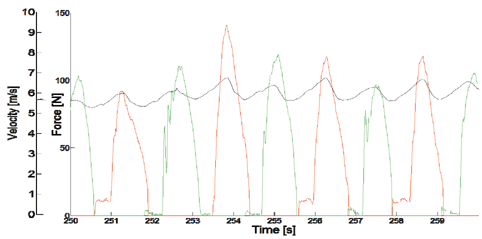

Fig. 12. The figure shows the forces from the right paddle node (green), the force from the left paddle node (red) and the velocity (black) measured on the hull node using the synchronization algorithm.

Synchronization is achieved by an algorithm executed on the mobile phone based on the method described in Section 2.1.1. The precision of this method has been determined to have a maximum synchronization error of 0.4±0.2ms, with a confidence grade of 99%. This has been deemed as sufficient for the current sampling rate as it is below 1/10 of the sampling period.

4.2 Monitoring in Neonatal Care

Cardiorespiratory monitoring in a Neonatal Intensive Care Unit (NICU), is one of the most vital monitoring system to presents secure the health of the infants [15]. Normally an ECG and a pulse-oximeter both connected via cables to the monitoring system are used. Fig. 13 shows information from a cardiorespiratory monitor (from the top); the ECG (green), the respiration (red), and the oxygen level from the pulse-oximeter (yellow).

Fig. 13. A photo of a cardiorespiratory monitor for an infant in neonatal care, which displays ECG, respiration and the oxygen level.

However the cables that connect the ECG and pulse oximeter to the cardiorespiratory monitor, complicate a natural and frequent skin-to-skin contact between the baby and the parents. Multiple studies have showed that kangaroo care, skin-to-skin contact between the infant and its parents, offers an environment that is as natural as possible for the infant to mature in. It improves the wellbeing not only for the infant but for the parents too. A wireless monitoring system can combine the goal with the requirement that an infant in NICU needs 24/7 monitoring of heart rate, respiration, oxygen level.

Since the monitoring is continuous over long time periods and the data rate is low, IEEE 802.15.4 radio communication seems to be an appropriate solution. A premature born baby may need extra oxygen for months after birth and therefore also monitoring of the oxygen level with a pulse-oximeter. False positive alarms, due to movements that lead to pulse-oximeter artefacts, are not uncommon and complicate a calm environment for the baby and the family. It is possible to minimize these alarms if the pulse-oximeter and the ECG are synchronized with each other [16]. Pulse-oximeter artefacts due to movements can then be identified and ruled out as true alarms. The tests presented in Section 3.3 show accurate time synchronization between two wireless sensor nodes can be maintained for hours.

5. CONCLUSIONS

Accurate time synchronization is essential for data fusion based on samples from multiple sensors. This paper has presented results of performance evaluation of major methods for time synchronization of sensor nodes in wireless personal area networks, mainly Bluetooth and ZigBee/IEEE 802.15.4. A new method for clock drift compensation is presented and evaluated. The performance results in this study are valuable guidelines when implementing synchronization algorithms for multi-sensor fusion, e.g. in sports and healthcare applications.

6. REFERENCES

[1] K. Römer, P. Blum and L. Meier, “Time Synchronization and Calibration in Wireless Sensor Networks”, in Ivan Stojmenovic (Ed.): Handbook of Sensor Networks: Algorithms and Architectures, pp. 199-237, September, 2005.

[2] G. Coulouris, J. Dollimore and T. Kindberg, “Distributed Systems: Concepts and Design”, Chapter 11, 4th edition, Addison-Wesley, 2006.

[3] D. Mills, J. Martin, J. Burbank and W. Kasch, “Network Time Protocol Version 4: Protocol and Algorithms Specification”, RFC 5905, Internet Engineering Task Force (IETF), June, 2010.

[4] J. Wåhslén, I. Orhan and T. Lindh, “Local Time Synchronization in Bluetooth for Data Fusion Using Mobile Phones”, International Conference on Body Sensor Networks (BSN 2012), Dallas, May, 2012.

[5] F. Cristian, “Probabilistic clock synchronization”, Distributed computing (1989): pp. 146-158.

[6] M. Jadliwala, Q. Duan, S. Upadhaya, J. Xu, “Towards a Theory for Securing Time Synchronization in Wireless Sensor Networks”, WiSec ’09.

[7] J. Wåhslén, T. Lindh, M. Eriksson and I. Orhan, “A Novel Approach to Multi-Sensor Data Synchronization Using Mobile Phones”, IJAACS, in print.

[8] Shimmer Research, http://www.shimmer-research.

[9] M. Maróti, B. Kusy, G. Simon and A. Lédeczi, “The Flooding Time Synchronization Protocol”, SensSys 2004, Balitmore, USA.

[10] J.S. Michael, R. Smith and K.B. Rooney, “Determinants of kayak paddling performance”, Sports Biomechanics, Vol. 8, No. 2, 2009.

[11] Mulle sensor node, http://staff.www.ltu.se/~jench/mulle.html

[12] D. Sturm, K. Yousaf and M. Eriksson, “A wireless, Unobtrusive Kayak Sensor Network Enabling Feedback Solutions”, (BSN 2010), Singapore, June 7 - 9, 2010.

[13] Tmote Sky – IEEE 802.15.4 compliant sensor module from Sentilla (previously Moteiv).

[14] HplAdc12 interface for direct AD-sampling in the MSP430 MCU: http://tinyurl.com/7l7d54r.

[15] I. Murkovic, M.D. Steinberg and B. Murkovic, “Sensors in neonatal monitoring: Current practice and future trends”, Technology and Health Care 11, 2003.

[16] L.K.L. Lum and P.W. Cheung, “Evaluation of Pulse oximetry with EKG Synchronization, IEEE Engineering in medicine and biology society 10th International conference, 1988.

Measurement-Based Performance and Admission Control in Wireless Sensor Networks

I. Orhan and T. Lindh

International Journal On Advances in Systems and Measurements
Volume 4, Numbers 1 & 2, pp 32-45, 2011

Measurement-Based Performance and Admission Control in Wireless Sensor Networks

Ibrahim Orhan
School of Technology and Health
KTH
Stockholm, Sweden
Ibrahim.Orhan@sth.kth.se

Thomas Lindh
School of Technology and Health
KTH
Stockholm, Sweden
Thomas.Lindh@sth.kth.se

Abstract—**This journal paper presents a measurement-based performance management system for contention-based wireless sensor networks. Its main features are admission and performance control based on measurement data from lightweight performance meters in the endpoints. Test results show that admission and performance control improve the predictability and level of performance. The system can also be used as a tool for dimensioning and configuration of services in wireless sensor networks. Among the rapidly emerging services in wireless sensor networks we focus on healthcare applications.**

Keywords - wireless sensor network, admission control, performance monitoring and control.

I. INTRODUCTION

Wireless personal area networks have emerged as an important communication infrastructure in areas such as at-home healthcare and home automation, independent living and assistive technology, as well as sports and wellness. Initiatives towards interoperability and standardization are taken by several players e.g., in healthcare services. Zigbee Alliance has launched a profile for "Zigbee wireless sensor applications for health, wellness and fitness" [2]. The Continua Health Alliance promotes "an interoperable personal healthcare ecosystem" [3], and at-home health monitoring is also discussed in an informational Internet draft [4]. It shows that wireless personal area networks, including body sensor networks, are becoming more mature and are considered to be a realistic alternative as communication infrastructure for demanding services. However, to transmit data from e.g., an ECG in wireless networks is also a challenge, especially if multiple sensors compete for access as in CSMA/CA. Contention-based systems offer simplicity and utilization advantages, but the drawback is lack of predictable performance. Recipients of data sent in wireless sensor networks need to know whether they can trust the information or not. To address this problem we have developed a performance meter that can measure the performance [5], and furthermore, feed a performance control system with real-time measurement data [6]. This paper also discusses whether admission control in combination with a system for continuous performance management can provide improved and more predictable performance. Admission control is used in many traditional telecom systems. It is also proposed in new Internet service architectures [7] to provide guarantees for quality of service. In this paper we present a method for measurement-based admission control in wireless personal area sensor networks for contention-based access. It is implemented as a part of an integrated performance management system that comprises performance monitoring, admission control and performance control.

The rest of the paper is organized as follows: a survey of related work in Section II; performance management in wireless sensor networks in Section III; measurement-based performance and admission control in Section IV; use cases and test results in Section V; and finally the conclusions in Section VI. This journal paper is an extension of a paper on admission control presented at a conference [1]. It provides a more detailed view of the other parts of the system, as well as the entire system for performance management.

II. RELATED WORK

Performance in contention-based wireless networks using CSMA/CA has been studied extensively. Measurements, simulations and theoretical studies show that the loss ratio increases with the traffic load and number of sending nodes. Bianchi [8] has derived an analytical Markov chain model for saturated networks, further developed in [9] and extended to non-saturated networks in [10]. Channel errors due to e.g., external disturbances and obstacles in the environment, can of course increase the loss ratio further. Another related problem, studied in [11], is the reduced throughput in multi-hop networks, with one or several intermediate nodes between sender and receiver. Dunkels and Österlind [11] found that the implementation of packet copying in intermediate forwarding nodes has significant impact on the throughput.

Performance in low-rate WPAN has been analyzed in several simulation studies ([12] and [13]). A performance meter that keeps track of losses, inter-arrival jitter and throughput has been developed [5]. Several papers have also addressed congestion and rate control in WLAN and LR-WPAN. CODA (congestion detection and avoidance in sensor networks) is a control scheme that uses an open-loop backpressure mechanism as well as a closed-loop control, where a sink node can regulate a source node's sending rate by varying the rate of acknowledgements sent to the source [14]. CARA (collision-aware rate adaptation) uses the RTS packets in IEEE 802.11 as probes to determine whether losses are caused by collisions (related to CSMA/CA) or by channel errors [15].

Our implementation of admission control, to accept or reject a request to join the network, is based on measurements of performance parameters, mainly the packet loss ratio. A

2011, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

similar probe-based admission control procedure has been suggested for differentiated Internet services [7]. Alternatively, one can measure the available capacity between two endpoints, or on specific links in a network. Pathrate, Pathload and BART are examples of implementations of such estimation tools ([16], [17] and [18]). SenProbe [19] estimates the maximum achievable rate between two endpoints in wireless sensor networks by injecting packet trains and analyze the dispersion between the packets. Some experimental studies indicate that measurements of available capacity in wireless networks often are inaccurate, especially for multiple hops [20]. Instead of active measurements, the contention-aware admission control protocol (CACP) estimates the available capacity by letting each node measure the amount of time the channel is busy [21]. Perceptive admission control (PAC) is an extension of CACP to encompass node mobility [22]. We have preferred a straightforward approach where the decision to either accept or reject an admission request is based on direct measurements and estimates of the performance parameters that are decisive for the quality of services.

III. Performance Management in Wireless Sensor Networks

A network scenario for the performance management system in this paper is depicted in Fig. 1. It consists of wearable sensors, such as ECGs, accelerometers, pulse-oximeters, fixed environment sensors, a coordinator, and intermediate nodes with routing and forwarding capabilities. An application program, running in the coordinator, processes sensor data from the sources and sends the information along with an estimate of the transmission quality to the remote end-user application for presentation and storage. The transmission quality can be expressed in terms of e.g., the statistical uncertainty of estimated parameters and the highest frequency component in a signal to be recovered by the receiver.

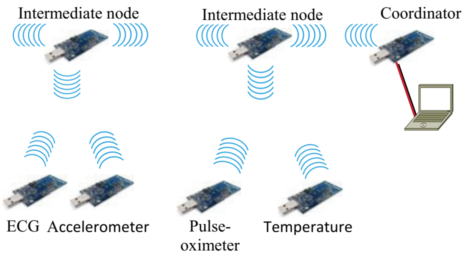

Figure 1. A network scenario where the performance management system is implemented in the coordinator and source nodes.

The performance monitoring and control capabilities can be implemented as add-on functions to be used by applications running in the communicating endpoints, e.g., sensor nodes and a coordinator, and not link by link. The ambition has also been to minimize the traffic overhead and energy consumption. The system is targeted to wireless sensor networks that use contention-based access, but can of course also be used in combination with contention-free access, such as guaranteed time slots. The applications, e.g., streaming data from accelerometers and ECGs, require certain levels of throughput and a low loss ratio, however not necessarily zero. The aim is, firstly, to provide quality estimates of the transmitted parameters, and secondly, to reuse this information for admission and performance control of information loss, delays and throughput. This closes the loop between measurements and control.

Admission control needs to be seen in the context of other necessary functions, especially performance measurements and control. The performance manager consists of the following functions: a performance meter that collects measurement data; admission control that handles requests to join the network; and performance control that maintains the quality of service for the admitted sensor nodes. The performance meter provides feedback information for admission and performance control. Fig. 2 shows the relationship between these functions. A request from a sensor node to join the network is handled by the admission control based on feedback from the meter. The performance control function is responsible for maintaining the desired quality-of-service once the sensor nodes are allowed to use the wireless channel. The performance meter is described in the following subsection (III.A) and admission and performance control in Section IV.

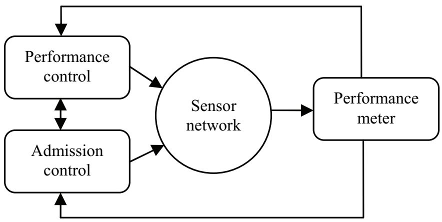

Figure 2. The performance manager consists of performance control and admission control. The performance meter supports the manager with measurement data.

A. *Performance Meter*

The approach is to combine active and passive techniques, inspired by the results from measurements in wired networks ([23] and [24]). A light-weight performance meter is implemented in each node. The meter consists of two counters that keep track of the number of sent and received packets and bytes, and a function that can inject monitoring packets. These dedicated measurement packets are inserted between blocks of ordinary data packets as seen in Fig. 3. They contain a sequence number, a timestamp and the cumulative number of packets and bytes transmitted from the sending node to the receiving node.

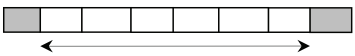

Figure 3. A monitoring block surrounded by two monitoring packets.

The interval between the monitoring packets, i.e. the size of the monitoring block, can be expressed in number of packets or a time interval, constant or varying randomly around a mean value. When a monitoring packet arrives, the receiving node stores a timestamp and the current cumulative counter values of the number of received packets and bytes from the sending node. Observe that for *n* sending nodes, the receiving node maintains *n* separate monitoring functions, one for each sending node.

Synchronization of the clocks in the participating nodes is not required. The local timestamps are used to calculate the inter-sending and inter-arrival times between pairs of monitoring packets. The inter-arrival jitter can then be calculated in a similar way as for RTP timestamps [25]. This means that the arrival time variation is estimated based on the monitoring packets, which represent samples of the ordinary data packet inter-arrival variation. Packet loss, on the other hand, is measured passively and directly using the counters.

1) Performance metrics

The following metrics can be calculated and estimated based on the collected measurements described in the previous subsection.

- Packet loss ratio: long-term average and average per monitoring block.
- The length of loss and loss-free periods defined as the number of consecutive monitoring blocks with or without losses. Can be expressed in time units, number of blocks, or number of packets and bytes.
- Inter-arrival jitter, J, is defined as $J=(r_n-r_{n-1})-(s_n-s_{n-1})$, where s is the sending time and r is the receiving time. The monitoring packets provide samples of this delay variation metric, which means that the uncertainty of the estimated statistics (mean value, median, percentiles etc.) is determined by the number of samples, and the variance of the delay process.
- Data throughput between sender and receiver can be calculated as a long-term average and also per monitoring block. The resolution of the peak rate is determined by the ratio between monitoring packets and ordinary data packets. This can also be seen as a measure of utilized capacity.

2) Meter and monitoring packet implementation

The performance meter is programmed in nesC [26] for TinyOS 2.1. The sensor nodes read samples from the sensors (ECG, accelerometer and temperature), assemble the samples and send them in packets to the coordinator (Fig. 4). The number of bytes and packets are counted. The cumulative number of bytes and packet and a timestamp are inserted into a monitoring packet, which is sent after every *n* ordinary data packet. A monitoring packet is 17 bytes long and includes the following fields: a start flag, a timestamp when packet is sent, type, a sequence number, number of packets sent, number of bytes sent, and a stop flag. The flags enable the coordinator to distinguish a monitoring packet from ordinary data packets. The sequence numbers identify and keep track of the monitoring packets. The packet and byte fields contain the cumulative number of bytes and packets sent. Finally, the type field enables measuring several sensor data flows from the same node.

Each time the coordinator receives a data packet, it updates the number of bytes and packets received from each sensor. The coordinator uses the source field in the CC2420 radio header to distinguish the packets from different sources. When the coordinator receives a monitoring packet, it stores a timestamp and the cumulative counter values of the number of received packets and bytes from the sending node. Fig. 4 shows the measurement data sent from the performance meter to the performance manager. The table in the lower left part of Fig. 4 shows the information in each monitoring packet sent from a sensor node: a timestamp, the total number of bytes and the total number of packets sent from the sensor node. The table to the right shows the corresponding information added by the coordinator for each received monitoring packet.

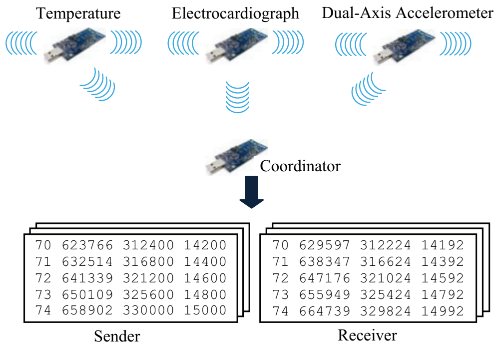

Figure 4. Measurement data from the sender and the receiver nodes. Columns from left to right: monitoring packet sequence no, timestamp (ms), cumulative number of bytes and packets.

IV. Admission and Performance Control in Wireless Sensor Networks

In this section the main idea behind the admission control (Section IV-A) and performance control (Section IV-B) system is presented. A star topology network controlled by a coordinator is used.

A. Measurement-Based Admission Control for Contention-Based Access

A typical application scenario is healthcare at-home with a number of sensors, such as ECGs, pulse-oximeters, accelerometers etc., connected to a coordinator. Fig. 5 shows a scenario with three sensor nodes connected to a coordinator sharing the same wireless channel that applies the CSMA/CA access method. Several hops between the sensor nodes and the coordinator, as well as mobile sensor nodes, are also a feasible scenario. Sensor node A and sensor node B in Fig. 5 are already connected to the wireless channel transmitting sensor data to the coordinator. The sensor nodes have a specified throughput and an upper limit for the packet loss ratio. Sensor node C requests admission to join the network for a specified throughput and packet loss ratio.

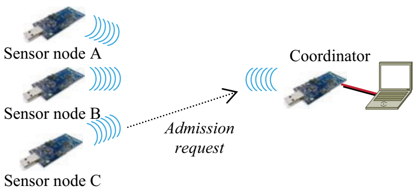

Figure 5. Sensor node C requests to share the wireless channel, already used by sensor node A and sensor node B.

The idea behind admission control is to accept or reject new sensor nodes to an existing network, while protecting the performance of already admitted nodes. Our purpose is to study whether it is feasible or not to use admission control in contention-based wireless sensor networks. The approach is to found the decision, to accept or reject an admission request, on estimates of real-time measurement data provided by a performance meter. A sensor node that intends to enter the network specifies the sampling rate, the sample size and the performance requirements. The verdict, to accept or reject the request, is determined by the outcome of probe packets transmitted during a test period. The probe packets sent from the requesting node to the coordinator should be of the same kind as the ordinary traffic it will transmit if admitted. The exchanged messages between a requesting node and the coordinator are described in the next subsection.

Strict performance guarantees are not feasible in contention-based access networks. However, many applications do not require completely loss-free transmission and are satisfied with soft performance requirements e.g., upper limits on packet loss and delay variation. The need for performance guarantees and predictability in contention-based networks for such applications is addressed in one of the use cases in Section V-C.

1) Messages between the coordinator and sensor nodes

A simple protocol for exchange of messages between the coordinator and the sensor nodes have been defined (Fig. 6). Sensor nodes send requests to join the network for a specified sampling rate, sample size and upper limits on performance parameters. If the coordinator is not busy handling previous requests, it will approve further processing. The sensor node is then instructed to start transmitting probe packets interleaved by monitoring packets. When the test period ends, the sensor node asks the coordinator for the decision. Having received 'accept', the sensor node begins transmitting its ordinary data packets to the coordinator. Monitoring packets are inserted between blocks of n data packets or with certain time intervals, to provide the performance meter in the coordinator with real-time updates of the transmission quality.

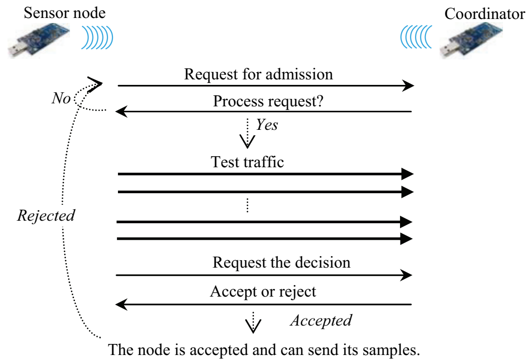

Figure 6. The messages between a sensor node and the coordinator during the admission phase. The arrows in thin lines are signalling messages and the arrows in bold lines represent probe packets in the test traffic phase.

2) Admission test period

The sensor nodes transmit probe packets during the test period in the same way as they intend to do if the request is accepted. The performance meter will report performance data for traffic between the coordinator and all sensor nodes as well as the test traffic from the requesting sensor node. Admission is accepted if the averages of the performance parameters for any of the already permitted nodes, including the requesting node, are below the threshold value. Admission can be denied to protect the existing nodes from performance degradation. The length of the test period is a trade-off between retrieving enough information from the probe packets and minimizing the effect on the other sensor nodes' performance. The first priority is to protect the already admitted nodes. The test traffic phase will be interrupted as soon as the probe packets have the effect that e.g., the loss ratio threshold is exceeded. The probe packets sent during the test period can be seen as a sampling process of the wireless channel, where the outcome of each sampling event is that the packet is lost or succeeds. The probability to lose a packet depends on the total traffic load and the number of nodes that are transmitting (ignoring radio channel disturbances). The number of samples needed for a given confidence level is determined by the variance of the traffic load. We have assumed that the sampling frequencies of the sensors are stable. This is a reasonable assumption for the kind of the applications the system is intended for. It means that the variance of the traffic load over time is low, and accordingly, that the number of probe packets can be kept small. The experiences from the use cases (Section V-C) in a normal home environment confirm that a test period of less than 30 seconds is sufficient. The length of the test period is further discussed in Section V-C.

B. Performance Control System

The aim of the performance control system is, firstly, to provide quality estimates of the transmitted parameters, and secondly, to reuse this information for systems management and enable performance control in real-time e.g., to minimize information loss and maintain a desired throughput. The output of the performance control system can also be to change the transmission power, enable or disable acknowledgement, etc. Applications, such as streaming data from accelerometers and ECGs in Fig. 7, require certain levels of throughput and a low loss ratio, however not necessarily zero.

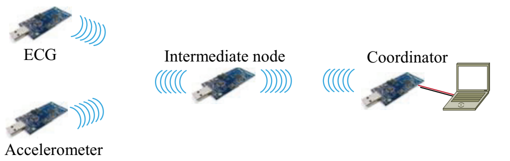

Figure 7. Two sensor nodes transmitting data to a coordinator via an intermediate node.

The performance control system (Fig. 8), implemented in a coordinator node, bases its decisions on the feedback information it receives from the meter e.g., packet loss, delays and throughput (packet loss in used in our cases). The meter delivers these performance updates for each incoming monitoring block e.g., once a second. The performance monitoring and control method has three main parameters. Firstly, the size of the monitoring block that determines the resolution of the performance metrics as well as the response time for the control actions. Secondly, the number of previous monitoring blocks (B_n, B_{n-1}, B_{n-2} etc), and their relative weight. The performance measurement results are calculated per each received monitoring block. To which degree the control method can rapidly adapt to changes is determined by these parameters. Thirdly, a step size (Δt) controls the time interval between transmitted packets (and thereby the packet frequency). This step size determines the response time and also the stability of the system.

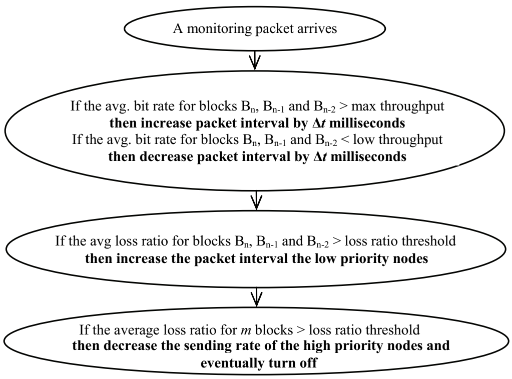

Figure 8. Flow diagram of the implemented control algorithm for prioritized nodes.

1) Algorithm to control throughput and loss ratio

The output of the control algorithm, to decrease or increase the packet frequency, is based on performance data from the current and previous monitoring blocks. The loss ratio and throughput (received bits per second) for a number of the recently received monitoring blocks are kept in memory. The manager sends a request message to a sensor node to either reduce or increase the packet frequency by adding (or subtracting) Δt milliseconds to (or from) the time interval between the transmitted packets.

2) Control algorithm with priority

A performance control system can support quality-of-service by assigning different priority to sensor nodes. Performance control with priority is primarily based on feedback information regarding packet loss and throughput from the respective source nodes. A use case with two levels of priority is described in more detail in Section V-B. High priority means that the required throughput (received bits per second) is maintained and the packet loss ratio is kept below a threshold for the prioritized nodes, possibly at the expense of nodes with low priority. If the loss ratio for the high-priority node is above the threshold, the manager will instruct the low-priority sensor nodes, to decrease their transmission rate step by step until the loss ratio for the high-priority node is below the threshold. If the loss ratio still is above the threshold, the sending rate of the high-priority nodes will be decreased as well, and eventually turned off if the loss ratio remains too high.

V. USE CASES

In this section, we present use cases where the performance meter, performance control and admission control is used. Section V-A shows how the performance meter is used for online transmission quality feedback. Examples of parameters and statistical uncertainty are presented. Section V-B contains two cases: performance control to maintain throughput and keep packet loss below a threshold; and control with different and dynamically assigned priority. Section V-C illustrates the potential performance problems with contention-based access and the need for admission control, as well as continuing performance monitoring and control. The sensor node platform TmoteSky [28], running TinyOS 2.1 and programmed in nesC, is used in all cases below. The radio (CC2420) and link layer are compliant with IEEE 802.15.4 LR-WPAN [27] in contention-based access mode.

A. Performance Meter – Online Transmission Quality Feedback

1) The testbed and measurement scenarios

Two different network scenarios are studied. In Fig. 9 the sensor nodes are attached to the coordinator in a star topology.

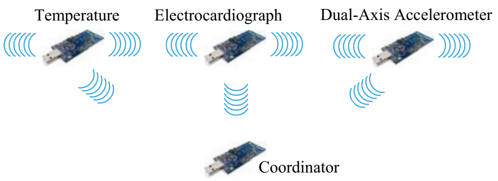

Figure 9. A network scenario with three sensor nodes and a coordinator.

In the second scenario (Fig.10), a sensor node is placed two hops away from the coordinator. The intermediate node forwards the packets from the sensor node to the coordinator. The buffer size in the intermediate node is 20 packets. In both scenarios, samples of sensor data are sent from the sensor node to the coordinator.

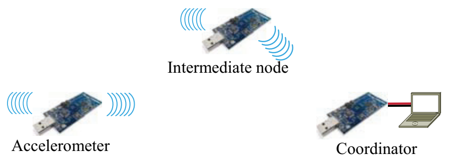

Figure 10. A network scenario with two hops between the sensor node and coordinator.

a) *The temperature sensor*

The temperature sensor in Fig. 9 is sampled twice a second and the collected samples are sent immediately to the coordinator. Monitoring packets are inserted between blocks of 100 data packets (6 byte payload).

b) *The ECG sensor*

One of the sensors in Fig.9 reads samples at 200Hz from the ADC-12 (analog-to-digital converters, 12 bits resolution) connected to an ECG. The samples are collected during five seconds. The radio is switched on, the samples are sent to the coordinator, and then switched off. This procedure is then repeated. Each packet contains 13 samples. The idea is to keep the radio turned off as long as possible and send several samples in each packet, in order to minimize the power consumption. In this case, the sensor node transmits 77 packets back-to-back every five seconds. A monitoring packet is inserted between blocks of approximately 100 ordinary data packets.

c) *The dual-axis accelerometer sensor*

A multi-sensor board (SBT80 from Easysen [29]) with a dual-axis accelerometer sensor is connected to two ADCs, one ADC for each axis. The accelerometer is sampled at 100Hz. The radio is only turned on during transmission. The sensor node sends 20 packets per second to the coordinator. Each packet carries 10 samples, 5 samples from each axis. Monitoring packets are inserted between blocks of 200 data packets, i.e. with approximately 10 seconds intervals.

2) Results and discussion

In this section some results using the performance meter in the two scenarios in Fig. 9and Fig.10 are presented.

a) *Loss periods and loss-free periods*

The loss ratio per monitoring block during the measurement period for the accelerometer data is illustrated in Fig. 11. The distinct loss events in the beginning of the measurement period are caused by radio interferences. Table I shows the loss ratio per monitoring block and the mean length of loss periods and loss-free periods for the three wireless links in Fig. 9.

TABLE I. PACKET LOSS RATIO BETWEEN SENSOR NODES AND COORDINATOR

	Acc–Coord.	ECG–Coord.	Temp-Coord.
Mean loss ratio	0.038	0.002	0.006
Max loss ratio	0.935	0.040	0.100
Min loss ratio	0.000	0.000	0.000
Loss period mean length (s)	37s	6s	11s
Loss-free period mean length (s)	15s	40s	165s

The loss ratio during the three loss periods is between 0.8 and 0.9 (Fig. 11). The length of the loss-periods (consecutive monitoring blocks that contain at least one lost packet) is shown in Fig. 12.

b) *Inter-arrival delay variation*

Table II shows the inter-arrival delay variation (jitter) for the scenario in Fig. 10 with two hops between the sensor node and the coordinator compared to one hop. The sensor node transmits 20 packets per second. The radio communication is not exposed to disturbances in this case. Fig. 13 and Fig. 14 show that the inter-arrival jitter is several times higher with an intermediate node then without it. Packet loss for two hops is also considerably higher compared to one hop. The high levels of inter-arrival jitter and packet loss in the two-hop case is due to the intermediate node's receiving and forwarding capabilities.

TABLE II. INTER-ARRIVAL JITTER (MS)

Inter-arrival jitter (ms)	One hop	Two hops
Maximum	13	59
Minimum	0	0
Standard deviation	4.0	12.5

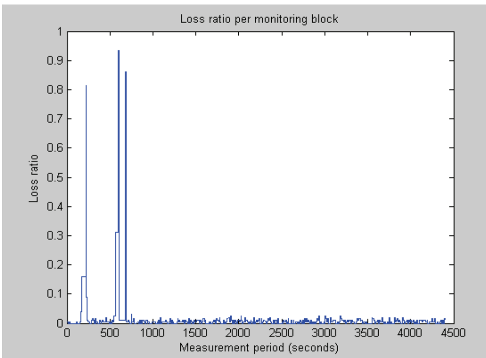

Figure 11. The loss ratio per monitoring block for accelerometer data in Fig. 10.

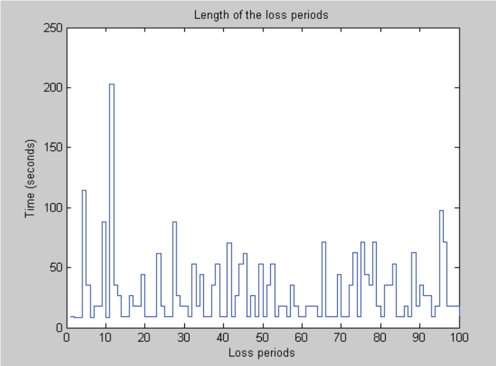

Figure 12. The length of loss periods in seconds.

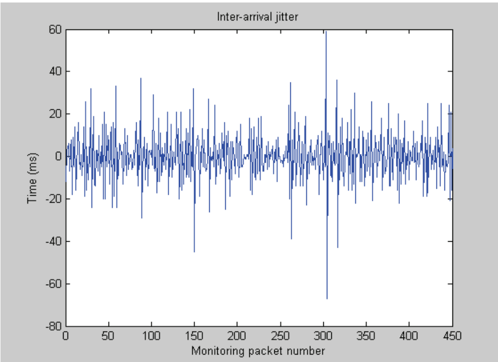

Figure 13. Inter-arrival jitter for a two-hop case.

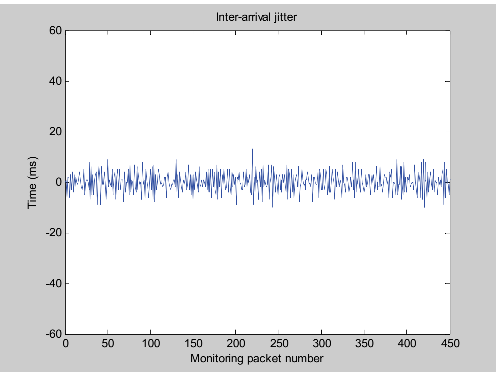

Figure 14. Inter-arrival jitter for a one-hop case.

c) *Uncertainty in parameter estimation*

The result from the performance meter can be used for calculating the statistical uncertainty of the parameter estimates based on samples from sensors. Table III shows how the confidence interval increases and highest frequency component in a received signal decreases when the loss ratio increases due to network performance degradation.

TABLE III. UNCERTAINTY IN ESTIMATION OF LOSS RATIO

Monitoring block duration (s)	Loss ratio	Conf. interval (0.99 level, stdev=4)	Highest frequency component in received signal
10s	0.025	0.93	50Hz
20s	0.313	1.11	35Hz
40s	0.935	3.62	3Hz
10s	0.010	0.93	50Hz
80s	0.861	2.47	18Hz
10s	0.030	0.94	50Hz
10s	0.005	0.93	50Hz
10s	0.005	0.93	50Hz
10s	0.010	0.93	50Hz

Details from the longest loss period in Fig. 12 are shown in Table III. The entire loss period consists of 20 monitoring blocks and lasts for 200 seconds. The monitoring block size in this case is 10 seconds. However, several blocks are longer e.g., the second, third and fifth row in Table III. The explanation is that monitoring packets, as well as data packets, may disappear before they arrive at the destination during a loss period. If one or several monitoring packets in a row are lost, the original monitoring blocks are merged into a larger block. Row 5 in Table III is a concatenation of 8 original blocks, where 7 monitoring packets were lost.

The loss ratio in Table III stretches from 0.005 to 0.935. The increased statistical uncertainty in estimating the mean value as the losses increase is shown in the third column. The standard deviation is around 4 units and the confidence level is chosen to be 0.99. The resulting confidence interval for an ideal communication channel without losses will be 0.92. A loss ratio of 0.313 (second row) leads to a confidence interval of 1.11, and a loss ratio of 0.935 gives a four times wider confidence interval (3.62). The number of samples, n, for a certain confidence interval, d, and confidence level (z=2.58 for 0.99 confidence level), and standard deviation, s, is given by, $n = \dfrac{z^2 \cdot s^2}{(d/2)^2}$.

The highest frequency component that can be recovered by the receiver for a 100Hz sampling rate is 50Hz (the sampling theorem). In this case the actual highest frequency component in the received signal is as low as 3Hz during the 36 seconds long period with a loss ratio of 0.935.

B. *Performance Control Algorithms*

Three examples of performance control are presented in this section. The purpose of the first control algorithm is to maintain throughput and minimize losses for a node with high priority by punishing nodes with low priority (Section V-B.1). In the second case, where all nodes have the same priority (Section V-B.2), each node tries to maximize its throughput under the condition that the loss ratio is below a threshold. The third case (Section V-B.3) is a combination of the two previous ones. From the beginning both nodes have the same priority. After a certain time, one of the nodes is dynamically assigned high priority and higher throughput. Finally, we show how the number of hops between a sensor node and the receiving coordinator determine the end-to-end throughput (Section V-B.4).

Fig. 15 shows the network scenario for the first case with two sensor nodes that are streaming ECG samples and accelerometer samples to the coordinator through a forwarding intermediate node.

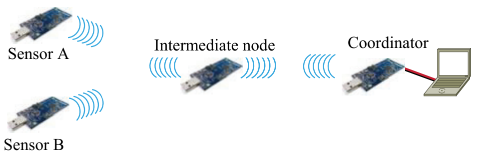

Figure 15. Two sensor nodes transmitting data to a coordinator via an intermediate node. Sensor node A has high priority and sensor node B has low priority.

1) *Control with priority*

The control algorithm in this case means that one of the sensor nodes is assigned high priority. The goal is to maintain throughput and keep loss ratio below an upper limit (0.02) for the high-priority node. The loss ratio threshold is computed as a weighted average of the three recent consecutive monitoring blocks and compared to the threshold 0.02. The required bit rate is 8kb/s, which corresponds to approximately 250Hz sampling rate per axis for a two-axis accelerometer or a 500Hz ECG.

Fig. 16 to Fig. 19 illustrate how the implemented algorithm works in practice. The high-priority node starts from 10kb/s and slows down to the expected bit rate 8kb/s (Fig. 16). The second node is turned on shortly thereafter (t≈80s) at a rate of nearly 16kb/s (Fig. 17). The received bit rate from the high-priority node falls sharply (Fig. 16). The solid lines (blue) show the received bit rate measured at the coordinator. The dotted lines (red) represent the sending bit rate from the sensor node. The loss ratio for the high-priority node peaks at almost 0.45 (Fig. 18), when the second node starts transmitting. The loss ratio for the low-priority nodes is shown in Fig. 18.

The performance manager reads the performance data provided by the meter for each block of incoming data packets. The monitoring block size is 100 packets in this test case. As soon as the manager detects the increased loss ratio for the high-priority node, it will instruct the other node to slow down. The low-priority node will directly decrease the transmitting rate (Fig. 17), which results in lower loss ratio (Fig. 18) and higher throughput (Fig. 15) for the prioritized node. As the loss ratio approaches the threshold, the sending rate of the low-priority node stabilizes around 3kb/s (Fig. 17). The performance manager strives to maintain the desired throughput (8kb/s) for the high-priority during the remaining part of the test, with an average loss ratio below the threshold.

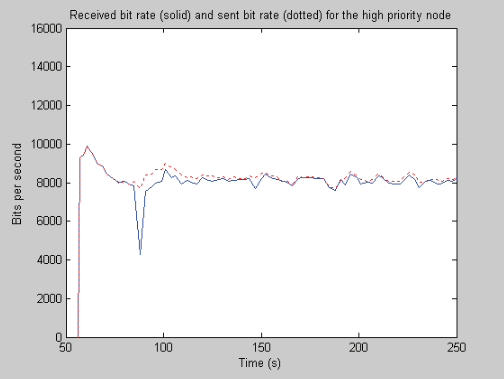

Figure 16. Throughput for the high-priority node.

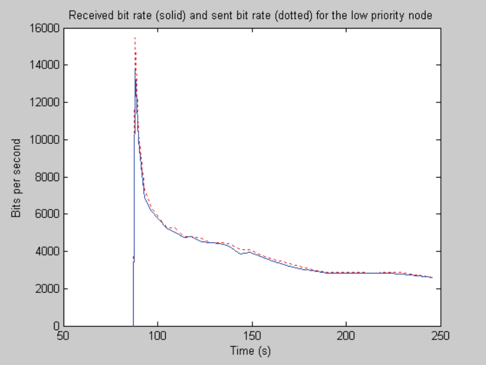

Figure 17. Throughput for the low-priority node.

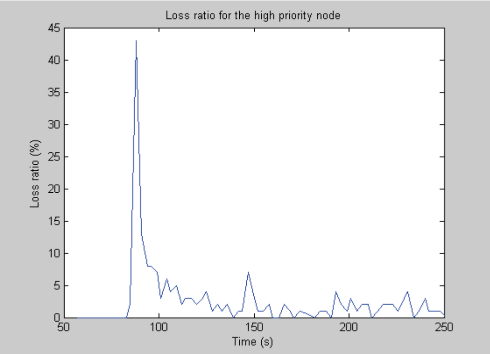

Figure 18. Loss ratio for the high-priority node.

2011, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

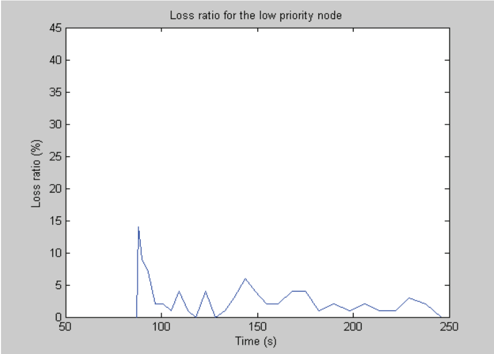

Figure 19. Loss ratio for the low-priority node.

2) Control without priority

In this case priority is not used. Both sensor nodes are controlled independently by the performance manager under the condition that the loss ratio is below a threshold. If the loss ratio exceeds the threshold, the sensor node will be instructed to decrease the sending rate (increase the packet interval by Δt milliseconds). No expected throughput is specified. Both sensor nodes start sending at 18kb/s as seen in Fig. 20 and Fig. 21. The high loss ratio for both nodes means that the performance manager will order both of them to slow down until the losses fall below the threshold. It can also be observed that the sensor node sometimes maintains the sending rate, even though the loss ratio is significantly higher than the threshold (Fig. 20 and Fig. 22). The explanation is that during heavy loss, monitoring packets will be lost as well, which delays the decision to decrease the packet frequency. After a while, the first node's throughput stabilizes around 3kb/s (Fig. 20) and around 3.5kb/s for the second node (Fig. 21). Since the control of the sensor nodes is independent of each other, the throughput will normally not be on the same level. One reason is different loss characteristics of the two channels; another may be different starting values. Each sensor node tries to find its maximum bit rate without exceeding the loss ratio threshold.

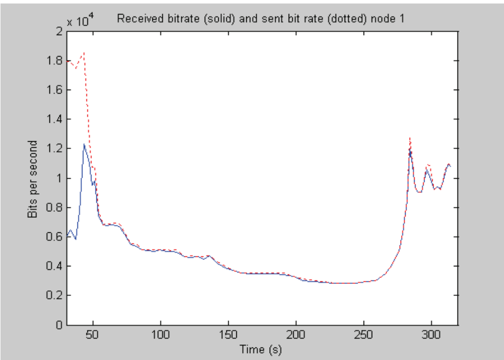

Figure 20. Throughput for node 1 (test case without priority).

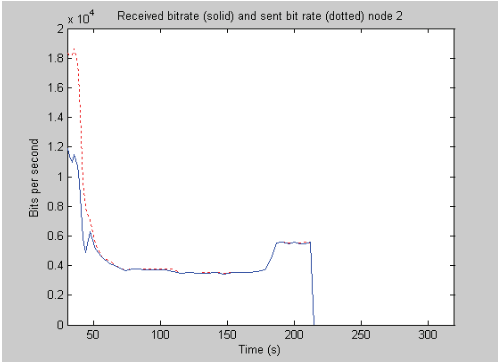

Figure 21. Throughput for node 2 (test case without priority).

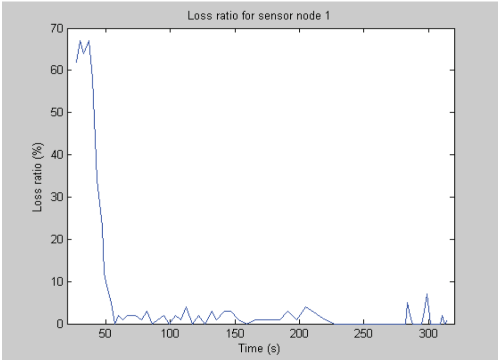

Figure 22. Loss ratio for node 1 (test case without priority).

At approximately t=180s, the manager has observed that the recent monitoring blocks are loss-free. The packet frequency is therefore increased for node 2 (Fig. 21). At t=210s, sensor node 2 stops transmitting (Fig. 21), which results in approximately zero packet loss for sensor node 1 (Fig. 22). The manager therefore tells the node to increase the packet frequency, up to around 10kb/s, where the loss threshold forces the node to slow down (Fig. 20).

3) Dynamic priority control

Fig. 23 and Fig. 24 show a combination of the previous two control algorithms. Both nodes start at a bit rate just below 15kb/s with 0.02 as the upper limit for the loss ratio. No node is given priority over the other. The throughput stabilizes between 4kb/s and 5kb/s. At t≈300 seconds, one of the nodes (Fig. 15) is dynamically assigned high priority, whereas the other node has to be satisfied with what is left. The reason might be that a higher sampling rate is needed for a sensor. The bit rate for the high-priority node rises to the required 8kb/s (Fig. 23) and the other sensor node backs off to around 2.5kb/s (Fig. 24). The step response in Fig. 23 takes around 30s. This time period can be reduced either by allowing larger step sizes (Δt) or decreasing the interval between the monitoring packets).

2011, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

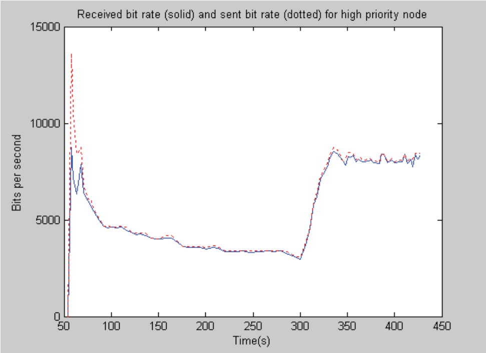

Figure 23. The sensor node is assigned high priority at t=300s and raises the bit rate to 8kb/s. The solid line represent received bit rate and the dotted line show sent bit rate.

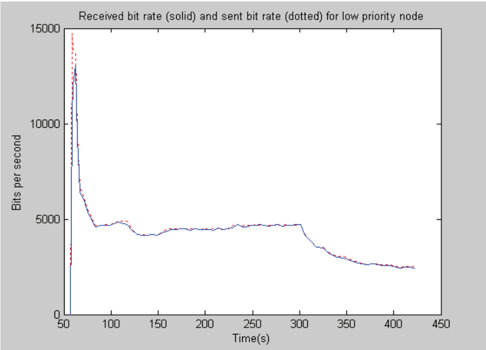

Figure 24. The sensor node is assigned low priority at t=300s and reduces the bit rate to 2.5kb/s. The solid line represent received bit rate and the dotted line show sent bit rate.

4) *Multi-hop cases*

The bit rate from a sensor node to a coordinator will to a large extent depend on the number of hops between the source and destination [4]. We have measured throughput between a sensor node and the receiving coordinator for zero, one and two intermediate nodes. The maximum received throughput for the equipment in our testbed using maximum packet length (payload 112 byte) was 50kb/s for one hop, 35kb/s for two hops and 20kb/s for three hops. This is of course a crucial limitation for demanding applications.

5) *Results and discussion*

Our analysis shows that it is feasible to use the measurement method, based on monitoring blocks, for performance monitoring as well as for feedback control of the performance of applications in wireless sensor networks. The results of the priority control algorithms are promising. The method has been implemented in a network with contention-based (CSMA/CA) access. It can of course also be used for the contention-based part of a super-frame in beacon mode in IEEE 802.15.4, where the contention-free part has guaranteed timeslots for the most demanding applications. One observation is that it is more straightforward to avoid packet loss in situations of buffer saturation by reducing the packet frequency, than to handle packet loss due to collisions and channel errors.

The monitoring and control method has three main parameters, that can be tuned for optimal results: the size of the monitoring block (B); the number of previous monitoring blocks (B_n, B_{n-1}, B_{n-2} etc) and their relative weight and, the step size (Δt) that controls the time interval between transmitted packets (or packet frequency). A more systematic study of these aspects related to control theory is for future work. To find out, in real-time, what capacity is available for a specified loss ratio, given that a second node transmits at a certain bit rate, is another example of application for the performance control method.

C. *Admission Control and Performance Issues*

In the following section we present some of the potential performance problems with contention-based access and the need for admission control, as well as continuing performance monitoring and control. Section V-C.1 illustrates the non-trivial performance problems associated with contention-based access (CSMA). Section V-C.2 shows how admission control works in real-time. The length of the test period is also discussed. In the third case (Section V-C.3), the implemented system is used as an off-line configuration tool to determine how changes of the traffic pattern influence the packet loss ratio. Finally, the alternative to allocate a new radio channel to a requesting sensor node is mentioned in Section V-C.4. The testbed consists of sensor nodes transmitting samples from ECGs, pulse-oximeters and accelerometers with sampling rates from 100Hz to 250Hz to a coordinator.

1) *Performance problems in contention-based access*

Contention-based access is a challenge for applications that require good and predictable performance. Fig. 25 illustrates what can happen when several sensors access a wireless channel. Three sensor nodes (A, B and C in Fig. 26) are connected to a coordinator sharing the same channel. The sensors are sampled during a second and the packets are sent back-to-back once a second. The bit rate is 9.6kbps for each sensor node. Fig. 25 shows the loss ratio during a measurement period for sensor node A. During the first part (0-70 seconds) only sensor node A is active. The loss ratio is almost zero. Between 70-140 seconds, sensor node B also accesses the channel. The average loss ratio experience by sensor node A is 0.03. During the remaining measurement period all three sensor nodes are transmitting on the same channel. The average loss ratio suddenly rises to 0.40.

For a loss-sensitive application, the performance is unacceptable after sensor node B, and especially after sensor node C, has joined the channel. The performance degradation may be avoided if the coordinator applies admission control and also maintains performance monitoring and control to protect the quality of service requirements for the existing nodes.

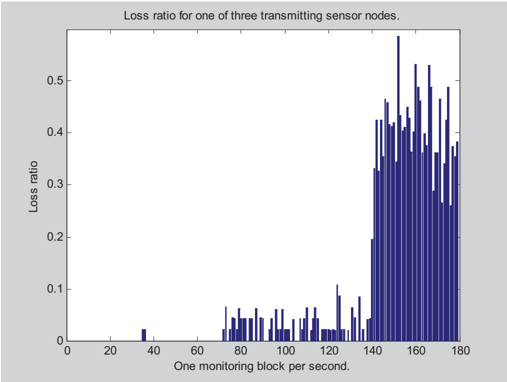

Figure 25. The loss ratio (y-axis) for sensor node A during the measurement period (x-axis in seconds). At approximately t=70s sensor B joins the channel. At t=140s a third node, sensor node C, accesses the channel.

2) Admission control

In this test case, the coordinator applies admission control when three sensor nodes with accelerometers, one by one, request to join the wireless network (Fig. 26). The sampling rate for the three-axis accelerometer is 200Hz per axis and the resulting average bit rate is 9.6kbps. The upper limit for packet loss for each node is set to 0.02 per monitoring block (the block length is around 1 second). The admission test period is 30 monitoring blocks (30 seconds). The measurement sequence is outlined in Fig. 27. Sensor node A requests admission and begins transmitting probe packets. The loss ratio during this admission test period is zero. Sensor node A's request is accepted and it starts transferring data. The loss ratio for the data traffic from sensor node A is almost zero before sensor node B requests to join the channel. Table IV summarizes the loss ratio for each sensor node during every test and data transfer period. Loss ratios exceeding the threshold (0.02) are indicated in bold text. It turns out that sensor node A and B are accepted, while sensor node C is rejected. For a sensor node to be rejected it is sufficient that the loss ratio for one of the sensor nodes, including the requesting node itself, exceeds the threshold.

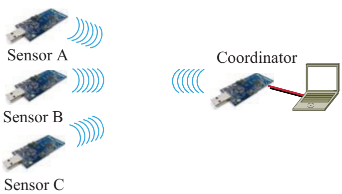

Figure 26. Three sensor nodes connected to the coordinator sharing the same channel.

The length of the test period is a trade-off between, on the one hand, to minimize the disturbance of existing traffic and reducing the response time for the admission verdict, and on the other hand, to receive sufficient performance data. The drawback of a predetermined fixed length of the test period is that ongoing traffic may suffer from severe performance deterioration. Fig. 28 shows the impact of test traffic on a sensor node during a 30 seconds test period. The average loss ratio is almost 0.05, with several peaks around 0.10, which is unacceptable performance deterioration for an already admitted node during a test period. To avoid this, we use an algorithm that calculates the cumulative moving average of the loss ratio for each incoming performance update i.e., for each monitoring packet. The test period is interrupted if the cumulative average exceeds a threshold. The cumulative moving average is defined as $CA_i=(L_1+L_2+L_3+...+L_i)/i$, where L_i is the loss ratio for monitoring block i. The algorithm is applied to the three test periods in Fig. 28 – Fig. 30. The cumulative averages for the first five blocks in Fig. 28 are CA_1=0.059, CA_2=0.035, CA_3=0.032, CA_4=0.042 and CA_5=0.042.

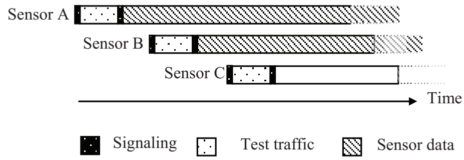

Figure 27. The measurement sequence for the nodes in Fig.26.

TABLE IV. LOSS RATIO FOR TEST PERIODS AND DATA TRANSFER PERIODS FOR SENSOR NODE A, B AND C.

	Sensor A	Sensor B	Sensor C
Test period sensor A	0.0000	--	--
Data transfer	0.0006	--	--
Test period sensor B	0.0012	0.0085	--
Data transfer	0.0019	0.0088	--
Test period sensor C	0.0046	**0.0470**	**0.0250**
Data transfer	0.0051	0.0083	--

If the rule for admittance is to allow maximum three consecutive updates of the loss ratio above the threshold (0.02), the test period will be interrupted after the third block. With an additional requirement that the loss ratio for a single block cannot exceed 0.05, this example means that the test period is interrupted after the first monitoring block.

A slightly different loss pattern is depicted in Fig. 29 (sensor node C's loss ratio during a test period). The cumulative average for the first seven blocks are CA_1=0.0118, CA_2=0.0119, CA_3=0.0159, CA_4=0.0240 and CA_5=0.0216, CA_6=0.0201 and CA_7=0.0206. In this case, the test period terminates after the 6^{th} monitoring block and the request is rejected.

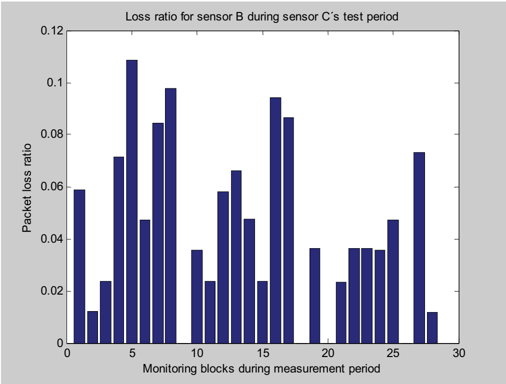

Figure 28. Loss ratio per monitoring block experienced by sensor node B during the third sensor's (sensor node C) test period. The average loss ratio is 0.047.

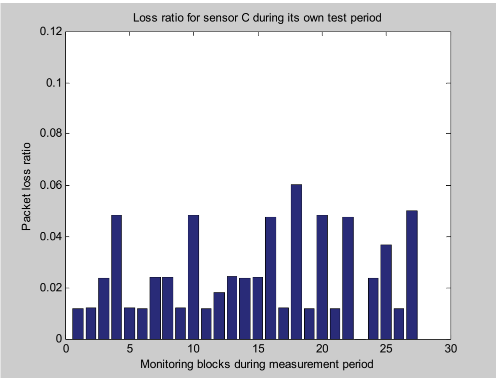

Figure 29. Loss ratio per monitoring block experienced by sensor node C during its own test period. The average loss ratio is 0.025.

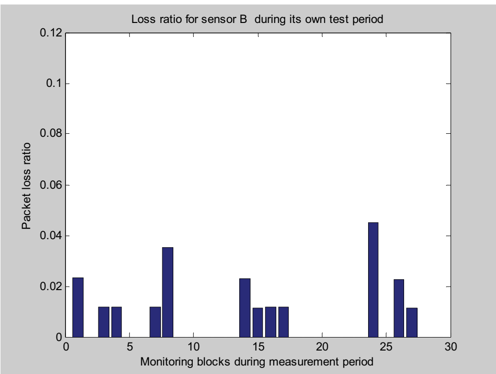

Figure 30. Loss ratio per monitoring block experienced by sensor node B during its own test period.

3) *Traffic patterns and channel access*

Packet loss in contention-based wireless networks is sensitive to the traffic pattern from the individual sources. Assume that two nodes collect samples and transmit the samples as a train of packets periodically once a second. If the nodes transmit the packet trains without overlap in time, the risk for losses due to collisions is low. However, the loss probability will increase if the packet trains happen to coincide. The dynamics of the traffic patterns in a network may from time to time lead to losses that exceed the accepted level after the admission test periods. The unpredictability of performance deterioration in wireless contention-based networks means that admission control must be combined with continuous traffic monitoring and control to be able to maintain the desired performance goals.

We have performed tests to study the impact of changes in traffic pattern on packet loss. Sensor node A collects and stores samples during a second. The samples are encapsulated in packets and transmitted back-to-back. The total time to transmit the packet train depends on the sampling rate, the sample size and the packet size. In this case, the sensor node sends a packet train of 43 packets with a packet size of 28 bytes, which corresponds to a throughput of 9.6kb/s. The total time to send the packet train was around 500ms. A second node, sensor node B, starts transmitting probe packets. It sends a train of packets once a second during the test period. The starting time for each train is shifted 50ms after ten seconds. This is repeated ten times, which means that the total time shift of the packet trains is around 500ms. The basic idea is to let the packet trains from sensor node B slide over the packet trains from sensor node A. Fig. 31 illustrates this convolution-like procedure.

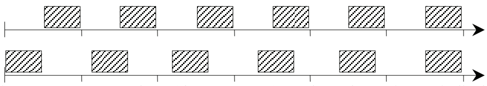

Figure 31. Sensor node A (the upper part) sends packet trains periodically every second. The starting times of the trains transmitted by sensor node B (lower part) are shifted in time so that they slide over the packet trains from sensor A.

Fig. 32 shows the loss ratio for sensor node B. After 10 monitoring blocks (10 seconds), the starting time is shifted 50ms. The average loss ratio for the first half of the measurement period is below 0.01. It rises to 0.10 for block 81-90 and 0.17 for block 91-100. The highest losses occur when the packet trains from the two sensors coincide in time. This convolution-like test might be inappropriate to use in an operating network but is useful for out-of-service configuration and dimensioning tests to estimate a worst case loss ratio. The traffic pattern for a channel e.g., the starting times of packet trains, is a stochastic process that may result in random losses from zero up to 0.25 in this case. Due to the unpredictability of contention-based wireless access continuous performance monitoring and control is needed to maintain the desired performance levels.

2011, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

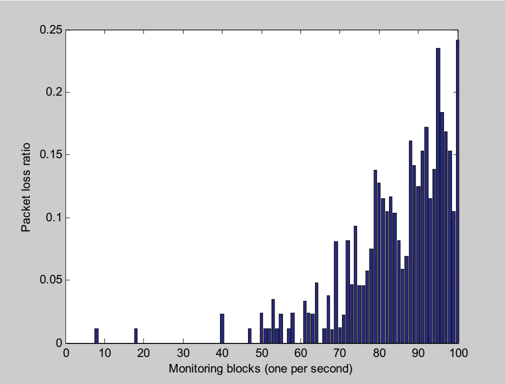

Figure 32. Loss ratio for sensor node B. The peak values occur when the packet trains from sensor node B coincide in time with the packet trains from sensor node A.

4) Redirecting to another channel

When a sensor node's request to join the network is rejected there are two alternatives. The node may back off for a while and try once again later. Alternatively, the coordinator may refer the sensor node to another radio channel. This feature has been successfully implemented and tested.

5) Results and discussion

The length of the test period is a trade-off between minimizing the disturbances on existing traffic, and receiving sufficient performance data for the admission verdict. The proposed algorithm uses a cumulative moving average of the loss ratio for the traffic from each sensor node to decide whether to reject an admission request and interrupt the test traffic, or to permit the sensor node to use the network. The test results show that admission control can improve the level, and predictability, of the performance of wireless sensor nodes. In addition, the method is also suitable for dimensioning, configuration and testing prior to operational mode. It can determine the number of sensor nodes that can share a wireless channel, for given performance requirements. A final conclusion is that continuous performance monitoring and control is needed to maintain the desired performance levels.

VI. CONCLUSIONS

Wireless sensor networks have today emerged as a feasible infrastructure for demanding applications e.g., in healthcare. This paper has addressed the non-trivial performance problems related to contention-based access to wireless channels. We have presented a measurement-based system for admission and performance control in wireless senor networks. The measurements are provided by a distributed light-weight performance meter. The test result shows that the implemented admission and performance control functions improve the quality, and predictability, of demanding services. The system can also be used as a tool for dimensioning and configuration of services in wireless sensor networks.

REFERENCES

[1] T. Lindh and I. Orhan, "Measurement-Based Admission Control in Wireless Sensor Networks", Sensorcomm, pp. 426–431, Venice/Mestre, July 2010.

[2] "Zigbee wireless sensor applications for health, wellness and fitness", Zigbee Alliance, March 2009.

[3] R. Carroll, R. Cnossen, M. Schnell, and D. Simons, "Continua: an Interoperable Personal Healthcare Ecosystem", IEEE Pervasive Computing, Vol. 6, No. 4, October-December 2007.

[4] A. Brandt (Zensys Inc) and G. Porcu (Telecom Italia), "Home Automation Routing Requirements in Low Power and Lossy Networks", Internet Draft, September 2009.

[5] I. Orhan, A. Gonga, and T. Lindh, "An End-to-End Performance Meter for Applications in Wireless Body Sensor Networks", BSN 2008, pp. 330–333, Hongkong, June 2008.

[6] T. Lindh and I. Orhan, "Performance Monitoring and Control in Contention-Based Wireless Sensor Networks", International Symposium on Wireless Communication Systems, pp. 507-511, Siena, Italy, September 2009.

[7] I. Más and G. Karlsson, "Probe-based admission control for differentiated-services internet", Computer Networks 51, pp.3902-3918, September 2007.

[8] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function", IEEE JSAC, Volume 18, No 3, pp 535 - 547 March 2000.

[9] Hai L. Vu, "Collision Probability in Saturated IEEE 802.11 Networks", Australian Telecommunication Networks and Applications Conference, pp. 21-25, Australia, December 2006.

[10] K. Duffy, D. Malone, and D.J. Leith, "Modeling the 802.11 Distributed Coordination Function in Non-saturated Conditions", Communications Letters, IEEE, Volume 9, Issue 8, pp. 715–717, August 2005.

[11] F. Österlind and A. Dunkels, "Approaching the Maximum 802.15.4 Multi-hop Throughput", HotEmnets, Virginia, June 2008.

[12] D. Cavalcanti et al, " Performance Analysis of 802.15.4 and 802.11e for Body Sensor Network Applications", BSN, Aachen, March 2007.

[13] N. Golmie et al: "Performance analysis of low rate wireless technologies for medical applications" Computer Communications, Volume 28, Issue 10, pp 1266-1275, June 2005.

[14] C.Y. Wan, S.B. Eisenman, and A.T. Campbell, "CODA: congestion detection and avoidance in sensor networks", SenSys, 1st conference on embedded networked sensor systems, pp 266-279, Los Angeles, November 2003.

[15] J. Kim, S. Kim, S, Choi, and D. Qiao, "CARA: Collision-Aware Rate Adaptation for IEEE 802.11 WLANs", INFOCOM, pp 1-11, Barcelona, April 2006.

[16] P. Ramanathan, D. Moore, and C. Dovrolis "What Do Packet Dispersion Techniques Measure", In Proceedings of IEEE INFOCOM, 2001, pp. 905-914, Anchorage, Alaska, USA, April 2001.

[17] M. Jain and C. Dovrolis, "Pathload: a measurement tool for end-to-end available bandwidth", Passive and Active Measurements Workshop, pp 14-25, Fort Collins, USA, March 2002.

[18] S. Ekelin, M. Nilsson, E. Hartikainen, A. Johnsson, J-E Mångs, B. Melander, and M. Björkman, "Real-Time Measurement of End-to-End Available Bandwidth using Kalman Filtering", IEEE NOMS, pp 73-84, Vancouver, Canada, April 2006.

[19] T. Sun, L. Chen, G. Yang, M. Y. Sanadidi, and M. Gerla, "SenProbe: Path Capacity Estimation in Wireless Sensor Networks" SenMetrics 2005, San Diego, USA, July 2005

[20] D. Gupta, D. Wu, P. Mohapatra, and C-N. Chuah, "Experimental Comparison of Bandwidth Estimation Tools for Wireless Mesh Networks", IEEE INFOCOM Mini-Conference, pp 2891- 2895, April 2009.

[21] Y. Yang and R Kravets, "Contention-Aware Admission Control for Ad Hoc Networks" Mobile Computing, IEEE Transactions on Volume 4, Issue 4, pp 363 - 377, July-August, 2005.

[22] Ian D. Chakeres and Elizabeth M. Belding-Royer, "PAC: Perceptive Admission Control for Mobile Wireless Networks", International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine), pp 18-26, Dallas, USA, October 2004

[23] T. Lindh and N. Brownlee: "Integrating Active Methods and Flow Meters - an implementation using NeTraMet", Passive and Active Measurement workshop (PAM2003), San Diego, April 2003.

[24] M. Brenning, B. Olander, I. Orhan, J. Wennberg, and T. Lindh: "NeTraWeb: a Web-Based Traffic Flow Performance Meter", SNCNW2006, Luleå, Sweden, October 2006.

[25] "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, H. Schultzrinne et al., July 2003.

[26] D. Gay, P. Lewis, R. von Behren, M. Welsh, E. Brewer, and D. Culler, "The nesC language: A holistic approach to networked embedded systems", Proceedings of the ACM SIGPLAN 2003 conference on Programming language design and implementation, pp 1-11, San Diego, USA, June 2003.

[27] IEEE Standard 802.15.4 - 2006.

[28] Tmote Sky – IEEE 802.15.4 compliant sensor module from Sentilla (previously Moteiv).

[29] STB80 – Multi-Modality Sensor Board for TelosB Mote, www.easysen.com, May 2011.

2011, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Contension-Based Wireless Sensor Networks - A case study for ambient assisted living

J. Burmeister, T. Lindh, I.Orhan, L-Å. Brodin and S. Lundberg. A project was presented in WSN for Ambient Assisted Living collaboration with CHB at the Ambient Assisted Living forum.
September 2010, Odense, Denmark

Contention-based wireless sensor networks - a case study for ambient assisted living

J. Burmeister (Dipl.-Ing., Dresden, internship student at KTH, spring 2010), T. Lindh, I. Orhan, L-Å Brodin, S. Lundberg.

Ambient Assisted Living forum 2010, September 2010, Odense, Denmark

Abstract of the presentation in the track "AAL in Research".

Wireless personal area networks have emerged as an important communication infrastructure in areas such at-home healthcare and home automation, independent living and assistive technology. Initiatives towards interoperability and standardization are taken by several players. Zigbee Alliance has launched a profile for "Zigbee wireless sensor applications for health, wellness and fitness" [1]. The Continua Health Alliance promotes "an interoperable personal healthcare ecosystem" [2]. These examples show that wireless personal area networks, including body sensor networks, are becoming more mature and are considered to be a realistic alternative as communication infrastructure for demanding services.

However, to transmit vital sign parameters from e.g. ECGs, pulse oxymeters and accelerometers in wireless networks is also a challenge, especially if multiple sensors compete for access. Contention-based access networks offer simplicity and utilization advantages, but the drawback is unpredictable performance due to loss of transmitted packets. We have used the SHIMMER wireless sensor platform [3] in the living lab at the Centre for Health and Building at KTH in a case study to identify and evaluate performance problems. The full-scale living lab consists of two apartments especially equipped with modern technique for healthcare at home and assisted living.

Our paper focuses on continuous monitoring of the heart activity using a wireless ECG based on the Wireless personal area network standard IEEE 802.15.4 (WPAN). Results from performance tests in the living lab will be presented e.g. influence of equipment such as micro wave ovens. Since contention-based wireless access has no guarantees for the quality of the delivered service it is interesting to determine to what extent the received ECG signal is sensitive to loss of information. ECG records from available reference databases [4] have been exposed to different levels and patterns of packet loss. These records have been analyzed by cardiologists and the result is reported in the paper. One interesting conclusion is that a diagnosis is fully possible for ECGs with packet loss ratio up to at least 5%. This project is part of research at the School of Technology and Health at KTH.

References

[1] "Zigbee wireless sensor applications for health, wellness and fitness", Zigbee Alliance, March 2009.
[2] R. Carroll, R. Cnossen, M. Schnell and D. Simons, "Continua: an Interoperable Personal Healthcare Ecosystem", IEEE Pervasive Computing,
Vol.6, No 4, October-December 2007.
[3] SHIMMER (Sensing Health with Intelligence, Modularity, Mobility and Experimental Reusability), http://www.shimmer-research.com
[4] Goldberger AL, Amaral LAN, Glass L, Hausdorff JM, Ivanov PCh, Mark RG, Mietus JE, Moody GB, Peng CK, Stanley HE. PhysioBank, PhysioToolkit,
and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals.Circulation 101(23):e215-e220 [Circulation Electronic Pages; http://circ.ahajournals.org/cgi/content/full/101/23/e215; 2000 (June 13th).]