Experiences from two sensor network deployments - self-configuration a key to success

Niclas Finne, Joakim Eriksson, Adam Dunkels, Thiemo Voigt
Swedish Institute of Computer Science, Box 1263, SE-164 29
Kista, Sweden
Email: {nfi, joakime, adam, thiemo}@sics.se

2007-08-28

SICS Technical Report T2007:09
ISSN 1100-3154, ISRN:SICS-T–2007/09-SE

***Abstract*—Various experiments have shown that the performance of wireless sensor networks is very hard to predict. It is also acknowledged that deploying sensor networks in real settings is a difficult and tedious task. To contribute to the understanding of wireless sensor network behavior we report on our experience from two recently deployed sensor networks: one in-door surveillance network in a factory complex and a combined out-door and in-door surveillance network. Both networks use advanced sensor network technology such as ad hoc routing and multi hop networking. Our results highlight the need for self-configuration in wireless sensor networks, especially in cases where fast deployment and dynamic environments are important aspects.**

Keywords: wireless sensor network deployment, surveillance, self-configuration

I. Introduction

Wireless sensor networks is a promising technology for many different applications. One of the obvious applications domains are surveillance networks. Wireless sensor networks enable easily deployed surveillance applications in urban terrain. However, experiences with wireless sensor networks have shown that their performance is "fairly pessimistic" [12] which makes the deployment of wireless sensor networks difficult from a communication perspective. Moreover, characteristics of the deployment environments cause additional problems [3], [4], [5]. In particular, Langendoen et al. [3] point out the difficulties posed by, e.g., hardware not working as expected.

In order to contribute to a better understanding of the behavior and problems of real-world sensor network deployments, we report on our experience from two surveillance applications. The first application is an in-door surveillance network in a factory complex. The second application is a combined out-door and in-door surveillance network. Both applications use multi-hop networking.

Our results highlight the need for self-configuration for wireless sensor network deployments, since our experiments show that manual configuration alone is not sufficient. Self-configuration can be used both for configuring the infrastructure as well as individualizing each sensor node for best application performance. An example of the first kind is that nodes self-configure the communication paths in a multi-hop network. This was implemented in our second deployment due to problems with the partly manual configuration of the network in the first deployment. An example where we have experienced the need for self-configuration of individual nodes is when the components used in low-cost sensor nodes behave differently on different nodes. In many of our experiments, radio transmissions triggered the motion detector on some of the nodes. In this case self-configuration mechanism could be implemented by monitoring the transmissions and motion detections combined with a possibility to configure the application to ignore detections during transmissions when a correlation is detected. These are only two examples, out of many, that show the importance of self-monitoring and self-configuration in wireless sensor networks.

The rest of the paper is structured as follows. The setup and measurements for the two deployments are described in Section II. In Section III we present our experiences from the deployments, including unexpected behavior. Finally, we describe related work in Section IV and our conclusions in Section V.

II. Deployments

We have made experiments with sensor network deployment of a surveillance application in two different environments. The first series of experiments were performed in a large factory complex with concrete floors and walls, and the second in a combination of outdoor and indoor urban environment.

During the deployments we measured both sensor data and communication quality. In the first experiment during the first deployment we measured single-hop communication quality in the in-door environment of the factory building. The following experiments all used multi-hop communication and were more focused on application performance.

A. System Setup

In both experiments ESB sensor nodes [8] are used. ESB nodes consist of a MSP430 microprocessor with 2kB RAM, 60kB flash, a TR1001 868 MHz radio and several sensors. The

sensors used during the experiments were the ESB's motion detector (PIR) and its vibration sensor. Furthermore, we have also measured the radio signal strength.

We implemented the applications on top of the Contiki operation system [2] that features the uIP stack, the smallest RFC-compliant TCP/IP stack [1]. All communication uses UDP broadcast and UDP header compression which reduces the size of the UDP/IP header to only six bytes.

The communication in the sensor network mainly consists of three different types of messages:

Measurement messages: used for sending sensor data to the sink. They consist of the source node id and a sequence number followed by ten double byte sensor values. The total message size excluding the UDP header is 22 bytes. The measurement messages are only used in the first experiment with single-hop network because the focus of the other experiments was on performance of the multi-hop surveillance application.

Path messages: used to report route path to the sink. They consist of the message type, the destination node id, a counter for the number of hops to the sink, the source node id, a sequence number, a node visit count, and nine double byte sensor values. This is followed by tuples of visited nodes and their received radio signal strength. Each node adds one such tuple as the message makes its way to the sink. The total message size is 24 + 2 bytes for each node visited.

Alarm messages: used to send alarms about detected activity. They consist of the message type, the destination node id, the number of hops to the sink, the source node id, a sequence number followed by the sensor type and sensor value for the triggered sensor. The total message size is seven bytes. Alarm messages are rather small to allow them to be cached at intermediate nodes but also because smaller messages have a higher probability to arrive at the sink node if we assume a constant bit error rate.

We used two different protocols during the deployment. In the initial experiment we used a single-hop protocol where all nodes broadcast messages to the sink. In the other experiments we used a multi-hop protocol where each node calculate the number of hops to the sink and transmit messages with a limit on hops to sink that forwarding nodes must respect. A node only forwards messages for nodes it has accepted to be relay node for and if it is closer to sink than the hop limit in the message. The forwarded messages are updated with a new decremented hop limit. This means a message can take several paths to the sink and arrive multiple times. During the first deployment only some nodes were configured to forward messages, but in the second deployment any node could configure itself to act as relay node.

After a sensor has triggered an alarm at a node, there is a random delay of up to 0.25 seconds before the alarm message is sent towards the sink node. The node then waits for an acknowledgement for up to two seconds before retransmitting the alarm message. At most three retransmissions are made before giving up on the transmission. The last retransmitted message can arrive six seconds later then the initial message for single hop routes. Only the latest alarm from each node is forwarded. When a new alarm messages catches up with an earlier one while travelling to the sink, the forwarding node discards the earlier message.

B. First Deployment: Factory Complex

The first deployment of the surveillance sensor network was performed in a factory complex in Motala, Sweden. The main building was more than 250 times 25 meters in size and three floors high. Both floors and most walls were made of concrete but there were sections with office-like rooms that were separated by wooden walls. Between the bottom floor and first floor there was a smaller half-height floor. The largest distance between the sink and the most distant nodes was slightly less than 100 meters.

The sensor network we deployed consisted of 25 ESB nodes running a surveillance application. All nodes either were forwarding messages to the sink or monitored their environment using the PIR sensor and the vibration detector. We made several separate experiments ranging from a basic single hop network for measuring communication quality in the factory to a useful multi-hop surveillance network for detecting activity in the factory.

1) Single-Hop Network Experiment: We made the first experiment in order to understand the limitations of communication range and quality in the factory building. The placement of the nodes in this deployment is shown in Figure 1. Node 1 acts as the sink node. All nodes communicate directly with the sink and send measurement packets at regular intervals without retransmitting lost packets. The sink node logged the received messages together with the received radio signal strength.

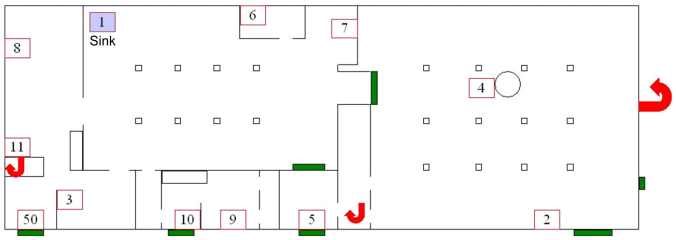

Fig. 1. Placement of the sensor nodes during the first experiment.

Node	Distance (meter)	Walls	Received	Sent (expected)	Sent (actual)	Reception ratio (percent)	Signal strength (avg,max)	
2	65	1 C	92	621	639	15%	1829	2104
3	21	1 W	329	587	588	56%	1940	2314
4	55	1 C	72	501	517	14%	1774	1979
5	33	2 W	114	611	613	19%	1758	1969
6	18	1 W	212	580	590	37%	1866	2230
7	26	2 W	347	587	588	59%	2102	2568
8	15	1 W	419	584	585	71%	2131	2643
9	25	1 W	194	575	599	34%	1868	2218
10	23	2 W	219	597	599	37%	1815	2106
11	17	1 W	331	591	593	56%	2102	2582
50	27	2 W	230	587	594	39%	1945	2334

TABLE I
COMMUNICATION RELATED MEASUREMENTS FOR THE FIRST EXPERIMENT.

Table I shows the results of the measurements. The columns from left are node id, distance from the sink in meters, number

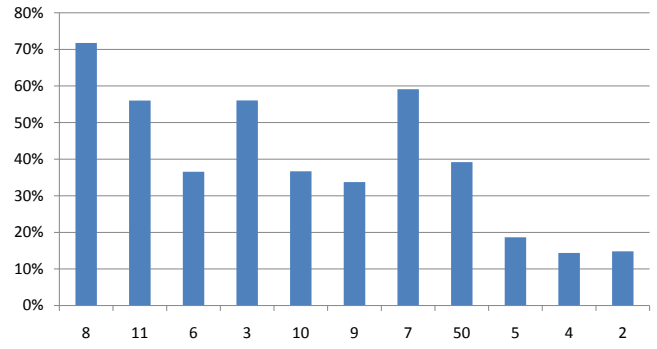

Fig. 2. Receive ratio in percent for each node during the first experiment.

of concrete and wooden walls between node and the sink, number of messages received at the sink from the node, number of messages sent by the node (calculated on sequence number), actual number of messages sent (read from a log stored in each node), percentage of successfully delivered messages, and signal strength (measured at the sink). Figure 2 illustrates the results related to message reception ratio sorted by distance from the sink. As expected the ratio of received messages decreases with increasing distance from the sink.

Full sensor coverage of the bottom floor, or all floors in this factory would not be possible using a single-hop network. The rest of the experiments were made with multi-hop networks.

2) Multi-Hop Network Experiments: We did a number of experiments using a multi-hop sensor network application to understand the performance of a multi-hop sensor network with respect to communication and surveillance coverage. We performed the final experiment in the first deployment during a MOUT (Military Operation on Urban Terrain) exercise with 15-20 soldiers moving up and down the stairs and running in the office complex at the top level of the building.

The response time for alarms in the network are based on how far the node is from the sink, number of hops, and how many retransmissions that are needed for the message to get through. The calculations below assume that the time for processing the detection interrupt and generating the alarm takes no time. This assumption is valid since the radio transmission times and other communication related times are much larger.

The transmission time for the alarm message over the TR1001 radio is calculated based on a bit rate of 19200 using Manchester encoding of the message. This gives an effective transfer rate of 1200 bytes per second. The radio packet overhead for synchronization and checksums is 30 bytes of non Manchester encoded data and the UDP/IP header with header compression is six bytes. This gives the transmission time for an alarm message as

$$t_{tr1001} = \frac{size_{msg}}{speed_{tr1001}} = \frac{(30/2 + 6 + 7)\ bytes}{1200\ bytes/s} = 0.023\text{s}.$$

The best case time for a single-hop alarm can be calculated given this and the time between detection of the motion or vibration and the radio transmission of the message. The

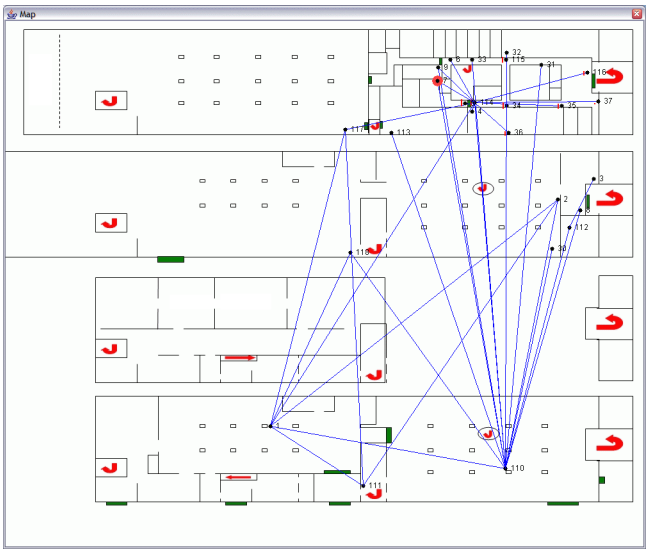

Fig. 3. Screenshot from the final experiment in the factory deployment. All levels of the factory are shown with the top level at the top of the screenshot. Placement of the sensor nodes during the surveillance and the paths used to transfer messages

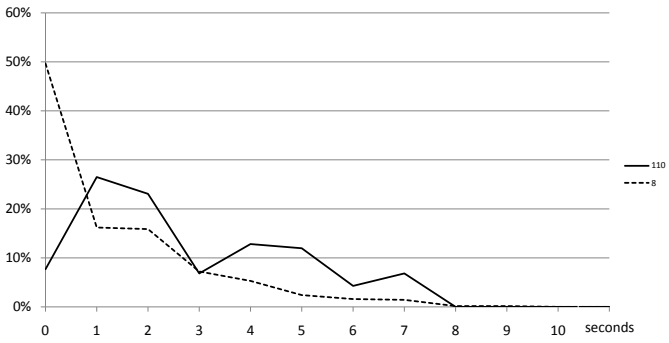

Fig. 4. Time spread in seconds for messages from each node as the messages arrived at the sink node. Times measured from the initial message to the last retransmission.

radio transmission can occur immediately after a detection but might also be delayed up to 0.25 seconds. The absolute best case response time (assuming that the sink has a very fast processing from reception until display of the alarm) is approximately 0.023 seconds. The worst case response time for an alarm to arrive at the sink in the single hop case is based on the retransmission cycles that are two seconds, and assuming that the message arrives at the last retransmission is

$$t_{single-hop,worst} = 0.023\text{s} + 0.25\text{s} + 3 \times 2\text{s} = 6.27\text{s},$$

which is much worse than the best case. The worst case time calculation assumes that no other node is communicating when the communication starts. Otherwise there can be a very short delay when other nodes are communicating because the radio is using a CSMA (Carrier Sense Multiple Access) mechanism to delay transmissions during transmissions by other nodes.

This delay can be neglected since the probability for it is low, and the delay is very short compared to six seconds.

For multi-hop messages the best case is when the network currently is completely empty, i.e. no relay node has any messages buffered. The best case is essentially two single-hops with shortest possible random delays plus the processing time of the arrived message before it is ready to be forwarded which is assumed to be less than one millisecond. The resulting two-hop best case time is

$$t_{two-hop,best} = 2 \times t_{single-hop,best} + t_{processing}$$
$$= 2 \times 0.023\text{s} + 0.001\text{s} = 0.047\text{s}.$$

The worst case two-hop alarm message time is when the message has to be retransmitted the maximum number of times at each hop. In the two-hop case there can also be other messages in the relay node's buffer which are going to be sent before. Each of these messages causes a delay of half a second.

Based on the experiments we can assume that there are at most eight other messages to relay, which gives a four seconds delay in the relay node per round of sending. Each message is forwarded at most four times before the relay node gives up. The total time will then be

$$t_{two-hop,worst} = 2 \times t_{single-hop,worst} + 4 \times t_{buffer}$$
$$= 2 \times 6.27\text{s} + 4 \times 4\text{s} = 28.5\text{s}.$$

This assumes that the messages reach the sink at the final retransmission. This is however a very unlikely case. Three and more hops can easily be calculated using the same assumptions.

Based on the measurements from the final experiment during the deployment it is possible to understand the communication quality of the network and the time it takes for alarm messages to reach the sink.

Node 110 was the most heavily loaded forwarding node in the network. It had a direct connection to the sink and forwarded 10270 messages from other nodes during the three hour long experiment. During the experiment the sink received 604 alarms generated by node 110, and 27 percent of these alarms were received several times due to retransmissions. Only four percent of the retransmitted messages arrived later than six seconds after the first message.

The worst case response time for node 110 is the same case as the worst case two-hop response time without the response time of the hop to the relay node. This is because the alarm is triggered on the relay node itself. The worst case response time for node 110 is

$$t_{node110,worst} = t_{two-hop,worst} - t_{single-hop,worst} = 22.3\text{s}$$

Another interesting node is node 8 that had seven paths to the sink, one direct connection with the sink, and six paths via different forwarding nodes. The sink received 1270 unique alarms from node 8 and 957 duplicates. Figure 4 shows the distribution of the delay between a message arriving at the sink the first time and the last time for both node 110 and 8. The causes of the multiple reception of the same message are due to multiple paths and retransmissions. Even if node 110 is communicating with the sink, the difference between first and last message for both nodes are very similar and no messages have a delay above eight seconds.

Most of the other nodes were relaying their messages via relay nodes and had a few alternative paths to the sink, with similar or better delays than node 110 and 8. This indicates that the network was reliable and that most of the alarms got to the server, and in many cases via several paths.

Using the available 25 sensor nodes and the alarm application it is possible to get sensor coverage of the most important passages of the factory complex. In our case the important passages were doors, stairs, and corridors.

C. Second Deployment: Combined in-door and out-door urban terrain

The second deployment was made in an artificial town made for MOUT exercises. It consists of a main street and a crossing with several wooden buildings on both sides of the streets. At the end of the main street there are a few concrete buildings. In this deployment the distance between the sink and the nodes at the edge of the network was about 200 meters, making the network more than twice as long as in the first deployment.

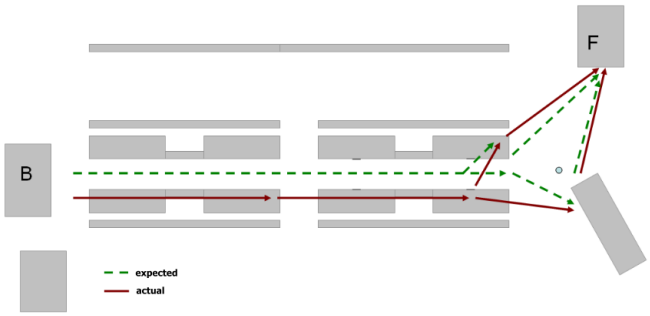

Fig. 5. Map over the artificial urban terrain with a main street and several buildings. Expected and actual path of the soldiers during the MOUT exercise differed a lot.

The surveillance system was improved in two important ways. First the network was more self-configuring in that there was no need for manually configuring which role each node should have (relay node or sensor node). Secondly the tracing of data messages was included in all messages, so even alarm messages included the path information so that information of the current configuration of the network was constantly updated as messages arrived to the sink node. This made the message size slightly larger, but since the network was at most four hops long this was not a big issue.

Even with the added path information in the alarm messages, the response times for alarms in the network are similar to the response times in the first deployment. The major difference is that the distant nodes were three or four hops away from the sink rather than two or three. Figure 7 shows

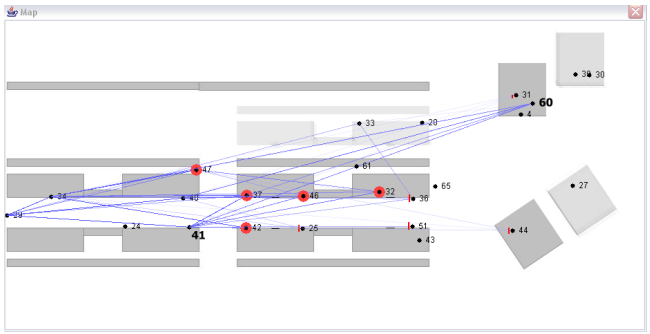

Fig. 6. Screenshot of the surveillance application showing several alarms on the main street and the recently active network paths.

the distribution of message delays of two different nodes. Node 60 is an edge node with many paths to the sink but still manages to get 10 percent of its transmitted messages directly to the sink almost 200 meters away. Only one percent of the duplicated messages arrived ten seconds after the first message. Node 41 is a relay node that forwards many messages, and it has a similar distribution of delays between first and last message. The results of the added distance and hop-count between the sink and the edge nodes are that we get slightly higher worst-case delays, but not much more delays in the average case.

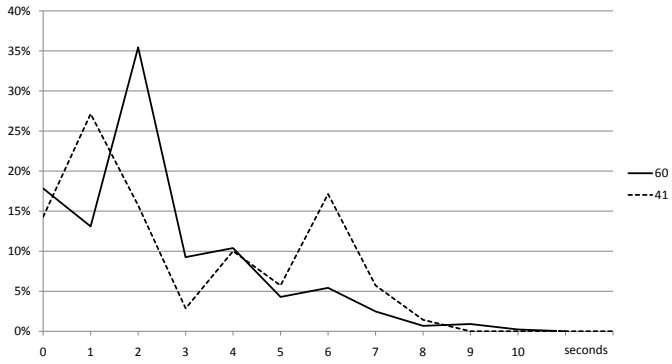

Fig. 7. Times between first and last duplicated messages of nodes 41 and 60 in the second deployment.

Using the 25 nodes in this deployment gave us fairly good sensor coverage of the most important areas of the urban like terrain.

III. Experiences

When we deployed the sensor network application we did not know what to expect in terms of deployment speed, communication quality, applicability of sensors, etc. Both deployments were made in locations that we did not have any previous experience from. This section reports on the various experiences we made during the deployments.

1) Network Configuration: During the first deployment the configuration needed to make a node act as a relay node was done manually. This made it very important to plan the network carefully and make measurements on connectivity at different locations in order to get a fairly optimal number of forwarding nodes. This was one of the largest problems with the first deployment, and it was a requirement that the network should be self-configuring during the next deployment. During the second deployment the network's self-configuration made deployment a much faster and easier task.

The importance of self-configuration of the network routing turned out to be higher than we expected since we suddenly needed to move together with the sink node to a safer location during the second deployment, where we did not risk being fired at. This was while the network was deployed and active.

Another problem was that the relay nodes have some constraints of the communication quality that need to be satisfied before the relay nodes started forwarding messages for other nodes. These constraints were set during tests at our SICS office but the environment in the factory required other settings, and the deployment would greatly benefit if the nodes themselves adjusted the constraints based on communication with each other. A mechanism for coordinated self-configuration of forwarding acceptance level would be interesting to study for this problem.

2) Unforeseen Hardware Problems: During radio transmissions some of the sensor nodes triggered unexpected sensor readings which cause unwanted false alarms. This is due to hardware related problems and affects sensors differently. Analog sensors are very hard to use during communication and even digital sensors can be affected. Since this was detected and understood during the first deployment, the only option was to rewrite the application so that it was possible to turn off the sensing on the nodes that had this behavior. A long term solution would be to develop software that can detect this problem automatically and stop reading sensor values while communicating. There is a need for both self-monitoring and self-configuration here.

3) Parameter Configuration: In the implementation of the communication protocols and surveillance application there are a number of parameters with static values set during early testing with small networks. Many of these parameters need to be optimized for best performance of the application. This optimization can only partly be done before deploying the network since the environment for different deployments vary much.

Some of the parameters we have identified for optimization are retransmission timers, alarm triggering delays, radio transmission power level, and time before refreshing a communication link.

4) Misplaced Surveillance Network: During the second deployment the sensor network was deployed for surveillance of a street. Nodes were placed on the outside walls of buildings to get the best position for the surveillance. When the operation started it turned out that the soldiers were moving forward inside the buildings instead of on the street as we had expected. Most of the sensor nodes actually detected the movements with their vibration detector instead of the PIR since opening door and moving quickly inside buildings made enough vibrations

for the sensor to detect. This was unexpected since the vibration sensors are not very sensitive on the ESB nodes.

5) Monitoring of the surveillance application: An important feature for understanding the performance of a sensor network deployment is that there are some independent indications of activity of the area that the sensor network monitors. We did not have an explicit installation of a parallel monitoring system, but in both deployments we did have some parallel feedback.

During the first deployment the sensor networks alerted us of movements in various parts of the factory but since we did not have any information about the soldiers' current locations it was difficult to estimate the time between detection by the sensor nodes and the alarm at the sink. At some times the soldiers used grenades which were powerful enough to trigger the vibration sensors on the nodes. This gave us an opportunity to estimate the time between we heard the grenade explosions until the vibration alarm reached the sink node.

During the second deployment we had the opportunity to see a real time feed from a wireless camera and use it to compare the soldiers' path with the alarms from the sensor network.

6) Radio Transmission: During the first deployment we placed routing nodes in places where we expected good radio signal strength (less walls, and floors). We expected the most used route to the sink to be through routing nodes in the stairwells. However the actual path taken for most messages turned out to be straight through two concrete floors via a node placed at the ground floor below the office rooms where the sensors were deployed.

Another radio related issue was that the WLAN camera that we intended to use for a video stream did not work at all. Even though many sensor nodes could communicate through the concrete floors, the WLAN did not work at all. This is partly due to the higher radio frequencies used in WLAN which results in shorter transmission range.

7) Instant feedback: One important feature of the application during deployment was that when started, a node shows connection status with the leds. It starts with blinking the red led as long as the node is not connected to the network. As soon as it is connected, the node beeps and flashes all leds, and then goes silence. Without this feature we would have been calling the person at the sink all the time just to see if the node had connected to the network. Another indirect and useful feature related to this was that we got an approximation of how good the connection was for that node. The longer time for the node to connect to the network, the worse the connection was. Nodes that required more than 5 - 10 seconds to connect, were usually moved to a position with better connectivity.

IV. Related Work

During recent years several wireless sensor networks have been deployed the most prominent being probably the one on Great Duck Island [4]. Other efforts include glacier [5] and volcano monitoring [10]. For an overview on wireless sensor network deployments, see Römer and Mattern [7].

In order to understand the combination of the wireless medium and the behavior of the transmitters on current sensor nodes, researchers have studied radio characteristics in different environments. Zhao and Govindan have observed a grey area within the transmission range of radios where transmission reliability varies significantly [12]. Also using a straight line configuration, Reijers et al. have studied link layer performance [6]. Both efforts used the same radios as on the ESB nodes used in our deployments. Turau et al. [11] as well as Sun and Cardell-Oliver [9] have performed measurements outside with a focus on packet delivery and link quality.

While we also present a number of measurement results our results have been obtained using a deployed application rather than a dedicated experimental setup. Measurements have also been performed both inside and outside, while our networks had both an outdoor and indoor part in a harsh military environment where, for example, one node was destroyed by a grenade. Furthermore, while we also present measurement results our focus is on the experiences and lessons learned in particular with relation to self-configuration.

V. Conclusions

In this paper we have reported results and experiences from two sensor network surveillance deployments. The experiences indicate that it is feasible to use COTS sensor network for surveillance. Our results highlight the need for self-configuration of both the network and surveillance application since we have experienced that manual configuration is not sufficient for real sensor network deployments, especially in situation when deployment time is very important.

From the experiences in our deployments we have identified several areas where self-configuration would make the deployment easier and the application performance higher. Some of these are: routing and node roles, radio signal strength threshold for relaying messages, filters for noisy sensors, retransmission delays and transmission power.

Acknowledgments

This work was funded by the Swedish Defence Materiel Administration.

References

[1] A. Dunkels. uIP - a TCP/IP stack for 8- and 16-bit microcontrollers. Web page. 2003-10-21. `http://dunkels.com/adam/uip/`

[2] A. Dunkels, B. Grnvall, and T. Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. In *Proceedings of the First IEEE Workshop on Embedded Networked Sensors (IEEE Emnets '04)*, Tampa, Florida, USA, November 2004.

[3] K.G. Langendoen, A. Baggio, and O.W. Visser. Murphy loves potatoes: Experiences from a pilot sensor network deployment in precision agriculture. In *14th Int. Workshop on Parallel and Distributed Real-Time Systems (WPDRTS)*, apr 2006.

[4] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. In *First ACM Workshop on Wireless Sensor Networks and Applications (WSNA 2002)*, Atlanta, GA, USA, September 2002.

[5] P. Padhy, K. K. Martinez, A. Riddoch, H. Ong, and J. Hart. Glacial environment monitoring using sensor networks. In *Proc. of the Workshop on Real-World Wireless Sensor Networks (REALWSN'05)*, Stockholm, Sweden, June 2005.

[6] N. Reijers, G. Halkes, and K. Langendoen. Link layer measurements in sensor networks. In *Proceedings of the 1st IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, Ft. Lauderdale, USA, October 2004.

[7] K. Römer and F. Mattern. The design space of wireless sensor networks. *IEEE Wireless Communications*, 11(6):54–61, December 2004.

[8] J. Schiller, H. Ritter, A. Liers, and T. Voigt. Scatterweb - low power nodes and energy aware routing. In *Proceedings of Hawaii International Conference on System Sciences*, Hawaii, USA, January 2005.

[9] Jingbo Sun and Rachel Cardell-Oliver. An experimental evaluation of temporal characteristics of communication links in outdoor sensor networks. In *Proc. of the ACM Workshop on Real-World Wireless Sensor Networks (ACM REALWSN'06)*, Uppsala, Sweden, June 2006.

[10] Geoff Werner-Allen, Konrad Lorincz, Jeff Johnson, Jonathan Lees, and Matt Welsh. Fidelity and yield in a volcano monitoring sensor network. In *Symposium on Operating Systems Design and Implementation (OSDI)*, Seattle, USA, 2006.

[11] Volker Turau; Christian Renner; Marcus Venzke; Sebastian Waschik; Christoph Weyer; Matthias Witt. The heathland experiment: Results and experiences. In *Proceedings of the REALWSN'05 Workshop on Real-World Wireless Sensor Networks*, Stockholm, Sweden, June 2005.

[12] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *The First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003)*, Los Angeles, California, November 2003.