WILDSENSING: Design and Deployment of a Sustainable Sensor Network for Wildlife Monitoring

VLADIMIR DYO, University of Bedfordshire
STEPHEN A. ELLWOOD, DAVID W. MACDONALD, ANDREW MARKHAM, NIKI TRIGONI, and RICKLEF WOHLERS, University of Oxford
CECILIA MASCOLO, BENCE PÁSZTOR, SALVATORE SCELLATO, and KHARSIM YOUSEF, University of Cambridge

The increasing adoption of wireless sensor network technology in a variety of applications, from agricultural to volcanic monitoring, has demonstrated their ability to gather data with unprecedented sensing capabilities and deliver it to a remote user. However, a key issue remains how to maintain these sensor network deployments over increasingly prolonged deployments. In this paper, we present the challenges that were faced in maintaining continual operation of an automated wildlife monitoring system over a one year period. This system analyzed the social co-location patterns of European badgers (*Meles meles*) residing in a dense woodland environment using a hybrid RFID-WSN approach. We describe the stages of the evolutionary development, from implementation, deployment and testing, to various iterations of software optimization, followed by hardware enhancements, which in turn triggered the need for further software optimization. We highlight the main lessons learned: the need to factor in the maintenance costs while designing the system; to consider carefully software and hardware interactions; the importance of rapid prototyping for initial deployment (this was key to our success); and the need for continuous interaction with domain scientists which allows for unexpected optimizations.

Categories and Subject Descriptors: C.3 [**Special-Purpose and Application-Based Systems**]: Real-time and Embedded Systems

General Terms: Algorithms, Design

Additional Key Words and Phrases: Duty Cycling, In-Network Storage, RFID Technology, Wireless Sensor Networks

1. INTRODUCTION

The deployment of sensor networks in a variety of real-world applications is gradually turning from a scientific vision into a reality. A number of systems have already been deployed, ranging from glacier monitoring [Beutel et al. 2009] to real time environmental and wildlife tracking [Zhang et al. 2004; Mainwaring et al. 2002]. Such systems have enabled the collection of spatio-temporal data at unprecedented granularities, and have revolutionized the way in which scientists perform field experiments. At the

An earlier version of this work appeared in Proceedings of the ACM SenSys 2010 [Dyo et al. 2010].
Authors' addresses: V. Dyo, Department of Computer Science and Technology, University of Bedfordshire, Park Square, Luton, LU2 0BL, UK; S.A. Ellwood, D. W. Macdonald, Wildlife Conservation Research Unit, Department of Zoology, University of Oxford, The Recanati-Kaplan Centre, Tubney House, Abingdon Road, Tubney, Abingdon OX13 5QL, UK; A. Markham, N. Trigoni, R. Wohlers, Computer Lab, University of Oxford, Oxford, OX1 3QD, UK; C. Mascolo, B. Pasztor, S. Scellato, Computer Laboratory, University of Cambridge, Cambridge, CB3 0FD, UK. This work was partially carried out when Vladimir Dyo was at Computer Lab, University of Cambridge, UK.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

same time, with the onset of new sensor deployments, the need has come to maintain sensor networks over prolonged deployment periods. Low effort maintenance and self-reconfiguration have become the idealistic selling points of wireless sensor networks. Network maintenance may involve a number of tasks, such as changing batteries, replacing faulty nodes and collecting data from special-purpose storage or gateway nodes. When the maintenance costs exceed user expectations and budget, there is a need to develop the system and make it sustainable. In this paper, we describe one such system and present our experience in building and developing a sustainable wireless sensor network. Our system consisted of a distributed wireless sensor network designed to monitor wildlife and environmental conditions in a dense woodland environment, in Wytham Woods, Oxfordshire, UK. The system was made up of three components. The first consists of active RFID transmitters attached directly to Eurasian badgers (*Meles meles*) as wearable collars. They were monitored by a second component consisting of a collection of fixed detection nodes distributed throughout the woods at key locations close to known badger setts (burrow systems) and latrines. The third component further complemented the assembly by providing a bed of fixed sensor nodes that were deployed within badger foraging areas to monitor micro-climatic conditions and their effect on species movement and mobility patterns.

We first describe the initial 'exploratory' field-deployable prototype designed to understand the domain requirements and the usage patterns. We then describe gradual alterations to initial design based on feedback from the domain scientists (zoologists). In particular, we evaluate each iteration in terms of maintenance cost and show that a series of modification phases to the initial commercial off-the-shelf based design, resulted in ten-fold improvement in maintenance costs, while enabling zoologists to collect unprecedented quantities of high resolution data on wild badger behavior.

In the first phase, we optimized the system at the software level proposing a novel sampling approach for the power hungry animal detection nodes, based on reinforcement learning. The idea was to exploit the behavior patterns of observed animals in order to more efficiently control energy consumption. We also implemented a novel storage management scheme that took into account data urgency and sink mobility to allocate sensor data to carefully selected storage nodes. We observed that these proposed software optimizations had a noticeable effect on the maintenance costs, but the network still required too many hours of hands-on human intervention.

In the second phase, we proceeded to enhance the hardware of the most power-hungry nodes to reduce their energy consumption. Here we provide details of the new platform, and how it drastically reduced the need for labor-intensive field trips to replace depleting batteries. This optimization led to a dramatic improvement in terms of maintenance costs. At the same time it triggered another round of software optimizations - we revisited sampling and in-network storage in the light of the new hardware capabilities. We validate the hypothesis that evolving hardware significantly impacted the performance of algorithms running on the nodes. This prompted us to introduce a more energy-efficient sampling algorithm for detecting badgers, which was not applicable in the old platform. It furthermore impacted the performance of our storage management scheme by altering the patterns of sink mobility. The running costs of the resulting system were reduced to such an extent that it made it realistic for zoologists to envision network expansion. The data collected throughout our deployment have the potential to offer zoologists a deep insight into the social life of badgers and on the correlation of their activities with weather and micro-climatic variations.

The lessons learned in this paper highlight the impact of maintenance costs on system design and the evolution, as well as the interplay between hardware and software optimizations. They also point out the need to take into account domain knowledge and application requirements to enable successful long-term deployments. The remainder

Fig. 1. Eurasian badger wearing one of our collars.

of this paper is organized as follows: Section 2 introduces the characteristics and requirements of the badger monitoring application. Section 3 presents the architecture, design and deployment of our initial monitoring system. Sections 4 and 5 present the two stages of network evolution. Section 6 illustrates the costs incurred by our various stages and existing monitoring techniques. Section 7 analyzes the data collected and presents our main observations of badger behavior. We discuss related work in Section 8 while Section 10 summarizes our findings and concludes the paper.

2. WILDLIFE MONITORING APPLICATION

In this section we describe the challenges and requirements of our badger monitoring application. Badgers (Fig. 1) are nocturnal mammals, spending their days in subterranean multi-entranced burrow systems (so called 'setts'), and foraging at night. In the UK, their habitat is typically mixed wood and farmland. Their active nocturnal period commences when they emerge above ground around dusk. These emergence times thus vary seasonally, therefore correlating with temperature and day length. During their active period, badgers visit specific places, such as 'latrines', which are thought to have an important role in their social behavior (see [Neal and Cheeseman 1996] for an introduction to badger biology).

After foraging, they return to their setts, usually around dawn. Separate, spatially distinct, setts are arranged into social groups. It is thought that badgers move readily between setts within a social group: typically, they would return to their setts of origin each night. Zoologists' understanding of the social bonds between the individuals in a social group remains incomplete. Badgers are apparently territorial, but to what extent they actively or passively establish a home-range is poorly understood. Movements are difficult to observe on a fine temporal scale, but systematic cage-trapping, up to four times per year, has indicated that movement between social groups, at the population level, appears to be minimal [Macdonald et al. 2008].

Zoologists would like to know more about the movements and social interactions of these animals: where and for how long they may meet is especially important. Since GPS receivers function poorly in densely wooded areas, such information is usually gathered by on-site, night time observation, VHF radio telemetry, and more recently by remote video surveillance. All these methods are labor intensive and expensive. For example, VHF tracking requires at least two people to get accurate location information on the animal, and it is not often practical to track multiple animals simultaneously. Despite intensive study of these animals, answers to fundamental questions regarding socio-spatial dynamics, and foraging-patch use remain elusive. The degree of social-

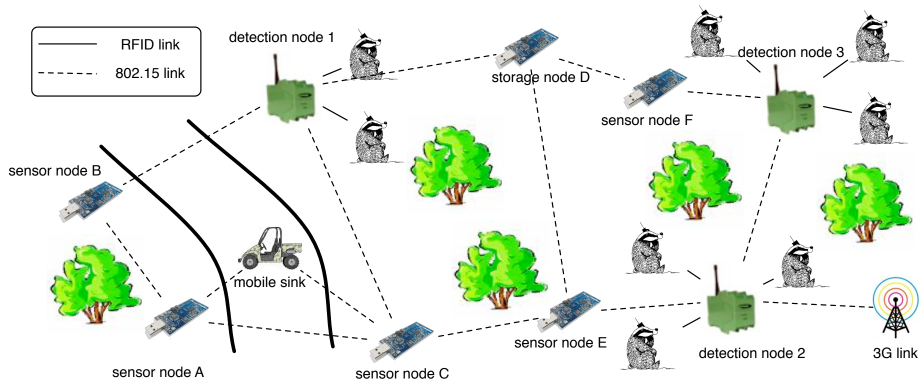


Fig. 2. Heterogeneous network consisting of badgers (equipped with RFID collars), detection nodes (fixed RFID receivers), environmental sensor nodes, zoologists (mobile sinks) and a fixed gateway.

group interaction is only superficially known using current technologies. Given the role badgers are considered to play in the epidemiology of bovine tuberculosis, and the full economic implications of this disease, understanding the temporal distribution of potential disease-carrying contacts, at key resource focal points, such as burrows and food patches (especially where these are shared with domestic animals) [Macdonald et al. 2006], is critical. Badgers are also a protected species, vulnerable to persecution, and emblematic of various conservation organisations. These types of conservation issues are typical for many species, and these risk factors are equally hard to monitor.

Given these requirements we have devised an integrated system for badger monitoring that could further help zoologists understand the social and behavioral implications of badger movements. Fig. 2 shows the heterogeneous nodes and devices that comprised our system. In this wildlife tracking installation we monitored badgers equipped with active RFID tags embedded within a small light-weight collar designed to have minimal impact on badger behavior. RFID receivers, referred to as *detection nodes*, were placed in key locations throughout the woods. In addition, we deployed a number of *sensor nodes* to monitor temperature and humidity in the same area. Sensor nodes and detection nodes were all connected through the same network. Our network also included a single solar powered gateway with cellular connectivity, which was located conveniently for 3G coverage and for its own maintenance. As it had cellular connectivity, it could relay data instantaneously to the end users.

Zoologists also contributed an element to the system as they perform regular trips to the study site to carry out routine observations and equipment maintenance. Thus, they acted as *mobile sinks* and assisted in the task of data collection, relieving the network from part of its communication load.

The data generated by our network fell into three categories: 1) RFID readings that reflected badger observations and were captured by detection nodes, 2) environmental (humidity and temperature) data monitored at regular intervals by fixed sensor nodes; and 3) network health data that indicated battery levels, memory usage and any sensor errors.

Zoologists and network engineers could assign priorities to different data types; a priority value reflects the tolerable delay between generating sensor data and delivering them to the user. For example, the detection of badgers dispersing from their natal setts may be considered very important as it represents potential fission and fusion within badger society, while also generating the potential for the transmission of so-

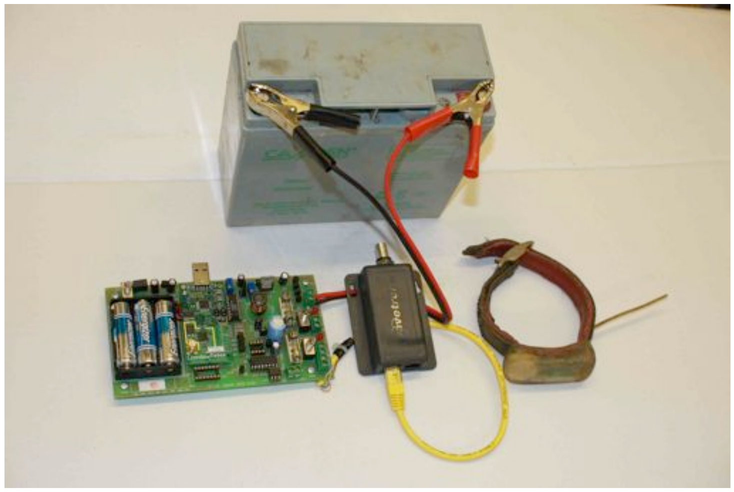


Fig. 3. The badger detection node (left) and the active RFID tag, potted in epoxy and mounted on a collar (right).

cial, genetic, and disease 'information'. Zoologists required prompt notification of these potential dispersal events as soon as they happened, whereas they were able to wait days for summaries of badger activity data, and weeks for raw badger observations and environmental data.

3. INITIAL SYSTEM DESIGN

This section discusses the initial design of our animal monitoring system, whose focus was on strong modularity and portability.

3.1. Sensing

Environmental monitoring: To investigate the potential impact of microclimate on individual badger behavior, we equipped Tmote Sky nodes with two external SHT-71 digital temperature and humidity sensors. One of the sensors was buried 30 cm underground (where it only measured temperature), and the other was mounted at a 1 m height. Ten of these nodes were deployed in the woods and made a measurement every five minutes. Suitable sensor housing was developed by trial and error to protect the sensor and also to allow it to record accurate humidity measurements. Our early packaging resulted in the saturation of the humidity sensor due to local condensation within the enclosure. We found sealing the digital sensor within hot-melt glue and shaping heatshrink to act as a shield to restrict wind chill resulted in the best solution. These devices were configured to either act as standalone data-loggers (which have very low average current consumption - approximately 30 μA) or as normal network nodes.

Badger Monitoring: wildlife tracking presents unique challenges, requiring animal borne tags to be simultaneously small, very reliable, and inexpensive. This influenced a number of design decisions including the use of a commercial 433 MHz Active RFID tag [1] over the alternative of designing a custom miniature mote platform. The selected tags satisfied most design requirements including low cost, miniature size and long lifetime. The small size of the tags was crucial as it allows much smaller animals than badgers to be tracked if necessary. Overall, the selection of commercial low cost tags

[1] http://www.wavetrend.net

Table I. Tag Specifications

Parameter	Value
Transmit frequency	433.92Mhz
Power output	72dbμV/m, 4300μV/m
Modulation	ASK
Bandwidth	$<$1Mhz
Stability	SAW stabilized
Range	0-100m (determined by choice of antenna and environment)
Lifetime	2-6+ years (depending on tag configuration)

also allowed the team to capitalize on the advantages of a fully tested component and focus on ground sensor network design, measurements, data collection and analysis. The tag measured 40x20x3mm in size (without a 123mm external whip antenna), and was powered by an on-board 3V CR2450 coin cell with an expected minimum lifespan of 2 years at 0.4s transmit interval (see Table I for more detailed specifications). The tag uses a Proprietary Signaling Scheme and data protocol (L-Series) patented by Wavetrend. Each RFID tag was hermetically sealed ('potted') in waterproof epoxy resin to protect the tag from environmental and mechanical damage (e.g. chewing by an animal). The collars with potted tags (see Fig. 3) were attached to badgers during routine trapping sessions, approved by institutional ethical review [Macdonald and Newman 2002] (UK Home Office Licence 30/2138; Natural England Licence 200001537). After full recovery from sedation badgers were released at their point of capture.

The presence of tagged animals was registered by 26 RFID detection nodes placed at setts and latrines, covering all main setts in the core study area (see Fig. 4). The detection range of a tagged animal was circa 0-30m, with the selected 433 MHz frequency providing longer communication range and lower obstacle fading through dense vegetation.

Each detection node consisted of an active RFID reader, a Tmote Sky mote and a custom designed mote extension board. For each detected tag the reader provided the following information: tag ID, reader ID, serial counter number, received signal strength (RSSI) and a checksum. The serial counter number facilitated an estimation of the tag age, which is used to notify the system when the tag is nearing the end of its projected life-span, based on the counter value reaching a certain threshold. The threshold varied based on the set beacon frequency. The counter also provided each beacon with a unique ID allowing the system to determine if two readers were detecting the same beacon (indicating tag is in range of both) or a sequential beacon (indicating tag has moved between readers). It could potentially support Real Time Location System functionality and limited accuracy for trilateration. RSSI was provided by the reader and gave an approximate indication of distance from reader. A detailed discussion on the relationship between RSSI and distance is presented in Section 3.3. Each reader had 2 RJ45 connections for communication. Several RFID readers could be daisy chained together via a 2-wire RS485 interface with a maximum of 254 readers in the wired network.

The extension board allowed the interconnection of the mote, RFID reader and peripheral devices to an RS232-TTL converter, MOSFET switches and the voltage regulators. The output voltage ranged from 6V to 12V and was configurable either through potentiometers and switches on board or from the mote via a standard I2C interface. The power management software on the mote duty cycled the peripheral devices including the reader, and monitored both mote and reader voltages to shut down the system should the voltage become low.

It should be noted that animal tracking and static detection and sensor node communication had different requirements in terms of communication range and antenna configuration, so decoupling the two communication systems was desirable. In partic-

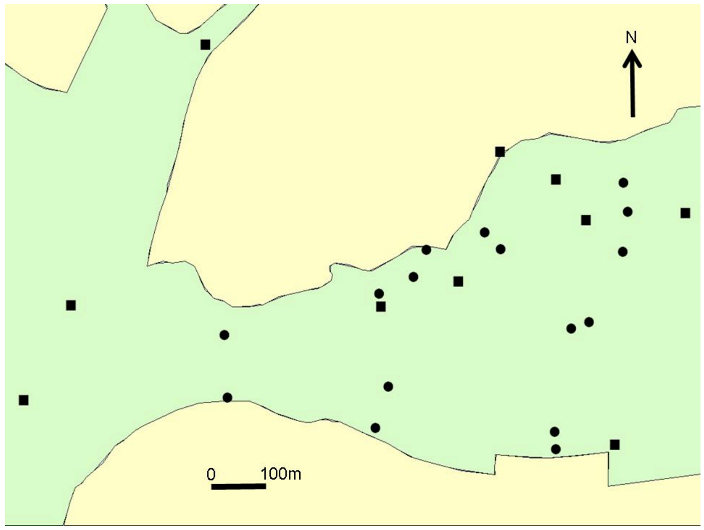


Fig. 4. Map of the study area showing RFID detection nodes: square = setts; circle = latrines. green = wooded areas, yellow = open arable areas.

ular, static node communication required extended communication range with preferably high bandwidth, whereas animal detection required a biologically meaningful communication range, with a high degree of consistency, which required consistent antenna orientation and receiver sensitivity of all detection nodes.

3.2. Data Collection

In our initial system design, we distinguish between two types of data - high-volume data, which consisted of raw badger observations, and low-volume data, which consisted of environmental readings, summaries of badger visits and network status reports.

Compression and local storage: As a result of the large data volumes generated by the network (typically in excess of 400 000 observations per week) we implemented a simple delta based compression technique to allow more data to be stored in the 1Mbyte flash memory of the Tmote Sky. This approach, which is application-specific and computationally lightweight, achieves a 25% higher compression factor than standard compression methods, like gzip. This technique takes advantage of the large degree of similarity between successive RFID readings from the same RFID tag. In essence, the difference between a base RFID observation and subsequent readings is encoded. Each raw reading occupies 10 bytes in its uncompressed form. The difference between an observation and the base record could be stored using only 3 bytes of information. Using this simple scheme, raw data were typically compressed by an average factor of 2.7x. This compares favorably with the resource hungry gzip (LZ77) algorithm which only achieves a compression factor of 2.0x on the same dataset. Thus, by reducing the volumes of data that needed to be buffered within the network, we were able to extend the memory lifetime of the reader node almost threefold. These data could be compressed further using dictionary type compression algorithms such as S-LWZ [Sadler and Martonosi 2006], but the gains would only be marginal and would require additional node resources.

Routing: Low-volume data (such as network status messages) were forwarded to the fixed 3G gateway node using a proactive shortest path routing algorithm. Every node maintained a routing table containing its distance to the gateway node. Initially the gateway advertised beacons with distance 0 to itself, and with increasing sequence (freshness) numbers. The distance from a node to the gateway was evaluated by taking into account the link qualities along the route. Each node maintained a neighborhood table that reflected statistics of outgoing traffic. The expected transmissions etx per message from the current node to a neighbor node N were computed as follows $etx(N) = attempted\, tx(N)/successful\, tx(N)$. The *distance* to the gateway node was defined as the sum of expected transmissions on all links along the route. Note that if all the links along a route had an $etx = 1$, the distance was equal to the number of hops along the route.

Every node broadcasted its distance to the gateway every 30 minutes. Upon receiving an advertisement from a neighbor N, a node compared the advertised distance ($advDist$) to the distance in its local routing table ($rtDist$). If $advDist + etx(N) < rtDist$ then it sets $rtDist := advDist + etx(N)$ and sets neighbor N as its next hop. If the route quality deteriorated significantly, a node simply selected the next best available route.

IPv6 customization and implementation: The multi-priority data collection approach was implemented using the uIP (micro-IP) IPv6 networking stack [Durvy et al. 2008] as well as X-MAC [Buettner et al. 2006] as the networking stack. To the best of our knowledge, the WildSensing project is the first project that used an IPv6 networking stack in the context of a delay tolerant wireless sensor network. Here, we describe our motivation and changes necessary to Contiki's IPv6 networking stack in order to achieve an acceptable network performance.

The choice of using Contiki's uIP networking stack was strongly influenced by the positive findings of Hui et al. [Hui and Culler 2008]. The added flexibility of using the IPv6 standard allows us to easily adapt the network to other tasks later during the network deployment, for example accessing and maintaining individual nodes or allowing near real-time data streaming from specific nodes within the network. Although the overhead incurred in terms of code size is considerable for the T-Mote Sky platform (approximately 16KBytes of additional ROM usage for our implementation), the added modularity and flexibility of the IPv6 network allows us to easily maintain and extend the network with new functions.

Data were disseminated towards storage nodes on a local hop-by-hop basis, instead of an end-to-end basis. Typically, TCP/IP connections are used in IPv6 networks to ensure reliable data transmissions, which require an additional overhead for establishing a connection and requires the end nodes to negotiate costly retransmissions. To avoid this overhead we used local UDP connections to transmit data along each hop towards the storage node. Upon receiving a package from a child node, the parent node returned a UDP ACK message to confirm reliable data transfer. Messages were stored in onboard flash and then only marked for deletion once an ACK message was received from the parent indicating successful custody transfer. In this way, data were relayed reliably, without the energetic expense of establishing and maintaining an end-to-end TCP connection.

We made three noteworthy changes to the original code in order to accommodate the low-power delay tolerant needs of our network. The majority of changes involved optimizing the stack to accommodate the largely static network topology, as expected in the WSN. Firstly, we reduced the frequency of IPv6 processes updating neighborhood variables, from 1000 Hz to 1 Hz, resulting in a reduction in processor power. Secondly, the frequency of neighbor detection messages, namely solicitation and advertising messages, was reduced such that the duty cycled MAC-layer could respond in a timely manner. In a standard IPv6 network, a node attempts to detect new neighbors every

10 ms, which congests the duty cycled MAC which is only able to transmit a few times per second. Consequently the neighbor detection period was reduced to three seconds. Thirdly, the neighborhood cache entries were set to be valid for 24 hours. Typically, a IPv6 networking stack defines the neighbor entry validity as milliseconds, which in our case of infrequent messages would result in new neighbor messages being transmitted for the vast majority of data packages.

While the three modifications to the frequency of updates were not explicitly bound by the IPv6 standard, the variables were specified to be set in milliseconds, rather than seconds. The changes were therefore most likely not conforming to the RFC4861 standard, though they should allow correct interoperation with a standard IPv6 network. Overall, the changes dramatically reduced the typical overhead incurred by an IPv6 network in a delay tolerant setting and allowed a flexible yet energy efficient network deployment.

MAC layer: We decided to use X-MAC [Buettner et al. 2006] at the MAC layer, a preamble based protocol in which senders indicate their intent to send data by frequently transmitting short wake up messages. Nodes periodically woke up, and if they heard a preamble, indicating a packet was addressed to them, they responded with an acknowledgement. This terminated the wakeup phase and the packet was sent. Nodes were configured to wake up every 500 ms and listen for 5.8 ms. This resulted in an effective basic duty cycle of 1.1%.

3.3. Quantifying node detection range

In order to establish the effective range at which an RFID detection node could receive transmissions from animal collars, it was necessary to conduct a number of field experiments. Theoretically, RSSI follows a log-law relationship with distance, but in natural habitats objects such as vegetation absorb or scatter radio signals. In order to quantify the relationship between RSSI and node-tag distance for all records, and so determine the average nodes detection range (all tagged badgers, at all nodes, during our study), we performed ‘walk-tests’ on a sample of tags and nodes specifically chosen to represent the extremes of detection performance of the system. Individual tags were attached to a model badger (8kg of minced meat modeled into the body, neck and head shape of a badger and wrapped in plastic sheeting), carried at badger walking height (suspended on ropes), along 50m transects away from and towards nodes. The models were expected to attenuate tag transmissions in a similar way to live badgers (note models, termed phantoms, are commonly used in human dosimetry trials) and so provide a realistic analogue for deployed tag performance on live badgers.

Old tags, previously worn by badgers, should logically loose capacity to generate a signal due to possible antenna damage and weakening batteries. Therefore, to capture a representative sample of tag affects, two old previously worn tags and two new unworn tags were tested. The more vegetation in the environment the greater the chance that radio signals will be attenuated and so, to capture a representative sample of node effects, we tested tags at 9 (out of 26) node sites ranging from the most open to the most over-grown. Four transects, radiating out at 0°, 90°, 180° and 270° to the node antenna were walked at each site. Node-tag distance was recorded every 2m on a laptop and synchronized with RSSI from the tags as it was recorded simultaneously by the node. General Linear Models (GLMs) were used to investigate/calibrate the effect of individual tags and node sites on the relationship between received signal strength and node-tag distance.

The graph in Fig. 5 shows the results from the walk-tests. In all, over 7500 tag transmissions were recorded. At these large sample sizes the standard GLM output of probability (’p-values’) are uninformative and so instead we emphasize effect sizes in terms of the proportion of variation explained (r^2). The ability of nodes to detect tags

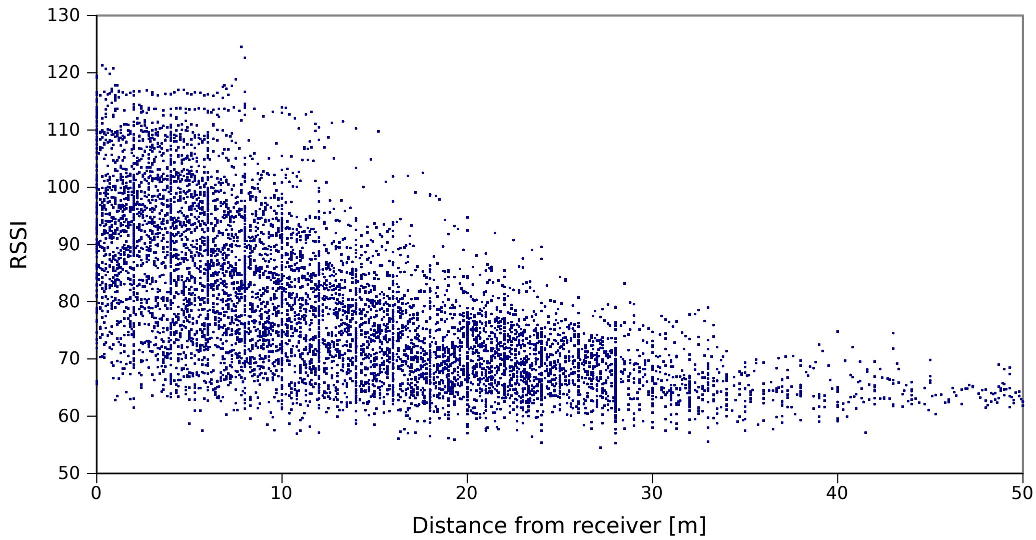


Fig. 5. Variation in RSSI of RFID detection nodes with distance.

did not seem to be biased by individual tags or nodes: there was a negligible effect of individual tag ($F_{3,12}$, r^2=2.7%) and node ($F_{8,12}$, r^2=1.1%) on RSSI. The dominant effect was that of node-tag distance ($F_{1,12}$, r^2=46.0%). That is, as expected, signal strength decreased with node-tag distance. The results from these tests demonstrate that detections do not need to be calibrated for individual badgers or sites. In order to establish a nodes average detection 'range' all data from the tests were pooled with the following results: 95% of all detected transmissions were within 31.5m of a node, 90% within 27.9m and 80% within 22.5m. These thorough tests were necessary to establish the biological meaning of a transmission recorded by a detection node.

3.4. Gateway Link

In order to transfer information from the field to the end-user, a solar powered 3G backhaul link was used, shown in Fig. 6. The emphasis was on making the gateway modular and flexible, thus commercially available (COTS) components were used where possible. The 3G connection was provided by a WiFi router with an external USB 'dongle' modem. A serial port forwarder converted RS-232 serial data from the node to TCP/IP packets. These were then sent over the 3G connection to a remote server where they were parsed and stored in a database.

The gateway node itself was a T-mote SKY which was equipped with a 256 MB SD memory card for buffered storage in the event of link failure. The gateway stored all incoming network packets, as well as any log/error messages that it generated. The node's serial port was forwarded over a TCP connection to a database server.

4. EVOLUTION STAGE 1: IMPROVING SENSING AND DATA COLLECTION

In this section we discuss how we started evolving our initial system design by introducing algorithmic improvements. The main weaknesses of our initial design were the high energy consumption of the badger detection nodes (RFID readers), and the heavy communication load around the fixed gateway. As shown in SectionSection 6, about one visit a week was necessary to change batteries and keep the system running.

Fig. 6. Deployed 3G Link. The solar panel provides power for the link which is housed within the box on the right-hand side of the picture.

4.1. Adaptive Sensing

RFID readers were the major source of power consumption on detection nodes. Despite being powered by a 12V 18Ah battery, without duty cycling they only lasted for one week. Increasing the lifetime of readers was therefore critical for large-scale long-lived deployments.

An obvious way to save energy was to duty cycle the RFID reader by periodically turning it on for a fixed duration of T_{on} seconds every $T_{interval}$ seconds. Nevertheless, setting optimal parameters was not straightforward: a high frequency sampling may have been too wasteful, whereas low frequency sampling may have lost important contacts. Tuning also requires knowledge of badger activity, which may not be known in advance.

We thus devised an adaptive duty cycling approach, which dynamically adapted the parameters T_{on} and $T_{interval}$ taking into account badger activity. We formulated the problem in terms of reinforcement learning [Kaelbling et al. 1996], and suggested a control strategy that adjusted node duty cycles based on animal arrival patterns [Dyo and Mascolo 2008]. The initial values of T_{on} and $T_{interval}$ were set to reflect the target duty cycle and the hardware capabilities of the detection nodes. For example, to achieve a target duty cycle of about 9%, T_{on} was set to 30s with the initial value of $T_{interval}$ at 330s. For efficiency reasons, T_{on} was chosen to be significantly longer than reader boot time T_{boot}, which was 10s.

The approach was composed of two main components: the *short-term* and the *long-term* adaptation components. Short-term adaptation extended the awake time T_{on} of the reader by a fixed short period of T_{ext} seconds each time badger activity was detected (i.e., a tag was in range). The short-term adaptation exploited the temporal burstiness of badger arrivals, as detection of a beacon was usually a good predictor of activity. The drawback of the periodic sampling technique, even in the presence of short-term adaptation, was that it assumed uniform badger activity throughout the day. However, it is rare that animals or humans remain continuously active throughout a day but rather follow a 24-hour circadian rhythm, which may vary depending on the environmental conditions [Aschoff 1965]. Badgers, for instance, are nocturnal animals that are inactive during the day, which means that sampling during the day may be wasteful.

The long-term adaptation component learned daily patterns of badger activity and adapted the interval $T_{interval}$ accordingly. We defined a target daily budget B as the amount of seconds that a badger detection node should spend in active state per day. Each day was divided into N equal time slots. Then, each node computed the expected number of sightings $E(d,t)$ during a day d for timeslot t and assigned a budget $B(d,t)$, proportional to $E(d,t)$, to each timeslot:

$$B(d,t) = B\frac{E(d,t)}{\sum_{i=1}^{N} E(d,i)}. \quad (1)$$

This is the equivalent of 'bidding' more resources in what has been a productive timeslot in previous days. We constrained $B(d,t)$ in the range $[B_{min}, B_{max}]$ in order to still explore all timeslots, even if they had not recently experienced any sighting, and to constrain the maximum number of times the node wakes up within a given time slot. Since in a timeslot of length T the reader was to be active only for $B(d,t)$ seconds, we have that $B(d,t)/T = T_{on}/T_{interval}$ and the node could adjust the duty cycle in each timeslot by setting the interval $T_{interval} = T\frac{T_{on}}{B(d,t)}$ between successive wake ups. On the first day, the budget was spread uniformly throughout all N timeslots, since there was no information about sightings. Then, the expected number of sightings $E(d,t)$ in timeslot t of a particular day d was evaluated as follows:

$$E(d,t) = \alpha \times O(d-1,t) + (1-\alpha) \times E(d-1,t) \quad (2)$$

where $O(d-1,t)$ is the actual number of sightings that were observed in the same timeslot on the previous day and α is a weight in the range $[0,1]$ which controls how rapidly new information is incorporated into the filter. Small values of α gave more weight to past history, but made the adaptation process slow and unable to capture sudden changes, whereas large values will make it very reactive to short term changes and less able to capture long term patterns.

Simulation-based evaluation: The evaluation of the adaptive duty cycling technique has been performed both through simulation and real deployment. Throughout our evaluation, we used $N = 24$ 1-hour timeslots, that is $T = 3600$s. Within each timeslot, the detection node turned the reader on and off to achieve the target duty cycle using Eq. 1. The on-time T_{on} was selected to be 30s, the initial interval $T_{interval} =$ 330s (which corresponded to a budget $B = 7854$s and a duty cycle of about 9%), and the extension time $T_{ext} = 300s$. The $[B_{min}, B_{max}]$ range was set to $[B/120, B/24] \approx [65, 327]$. We used a fixed duty cycling algorithm, where a node woke up and went to sleep at fixed intervals of time, as a baseline. The algorithms were implemented in Tossim 2.0.2 simulator and evaluated by replaying the real data recorded by the always-on node. We made 10 simulation runs for each algorithm with random node offsets.

Fig. 7 shows the performance of the always-on, fixed duty cycling and adaptive algorithms respectively. The always-on node detected all 76707 encounters at 100% duty cycle. The fixed duty cycling node detected 7773 encounters at 9% duty cycle.

The short-term adaptation version detected 50262 (65%) encounters at 10% duty cycle resulting in much higher encounters per duty cycle than always-on and fixed nodes. The duty cycle of the short-term algorithm was 1% higher compared to a fixed algorithm due to extension of the wake-up time when the activity was expected. The combination of short-term and long-term adaptation techniques resulted in slightly fewer encounters (46214) than a short-term version because of longer sleep intervals

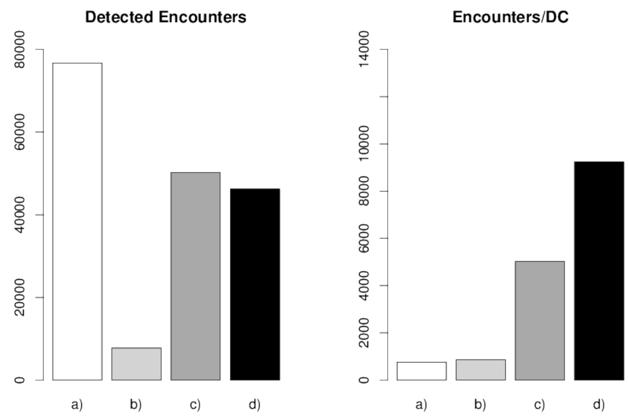


Fig. 7. Simulation results. Comparison of detected encounters and encounters per effective duty cycle for a) always-on b) fixed c) short-term adaptation d) short-term and long-term adaptation.

during inactive periods, causing some encounters to be missed. However, the overall duty cycle reduced to 5% showing the effectiveness of long-term adaptation.

Deployment-based evaluation: In order to evaluate our duty cycling technique in a real deployment, we placed two detection nodes with the same hardware and antenna orientation next to each other. One of the nodes was always on, whereas the other executed our adaptive duty cycling technique. In addition, we processed data from the always-on node to simulate a fixed schedule. The adaptive node was configured to work at 9% duty cycle.

The evaluation was based on 833 hours of summer (July) deployment data from both nodes. Data were periodically retrieved from both nodes manually. The results are summarized in Fig. 8. The fixed duty cycling node captured 7201 sightings while using 10% of the power of the always-on node. The adaptive duty cycled node detected 54568 (73%) of all sightings, while consuming approximately 8.2% of the energy.

4.2. Delay-tolerant data collection

The initial design of the data collection algorithm was based on the principle that raw RFID data are high-volume but low-priority, and could be stored locally at sensor nodes. The remaining data had higher priority and were forwarded to the 3G gateway using a tree-based routing algorithm. This initial approach is similar to related work on prioritizing data traffic and taking into account routing costs to determine whether to discard data, store it locally, or forward it to the gateway [Werner-Allen et al. 2008].

Here we added a further step using a delay-tolerant data collection approach, which leveraged the movement of zoologists and other environmental scientists to efficiently collect sensor data. Not only did we prioritize data based on their urgency, but we also prioritized nodes based on the frequency with which mobile sinks visited them. In this way, we forwarded data to carefully selected storage nodes, purely based on data and node priorities.

Data priorities: When data were generated, they were assigned a *data priority* class that represented the latency allowed until they had to be delivered to the end-user. Our network generated observations of tagged badgers captured by the detection

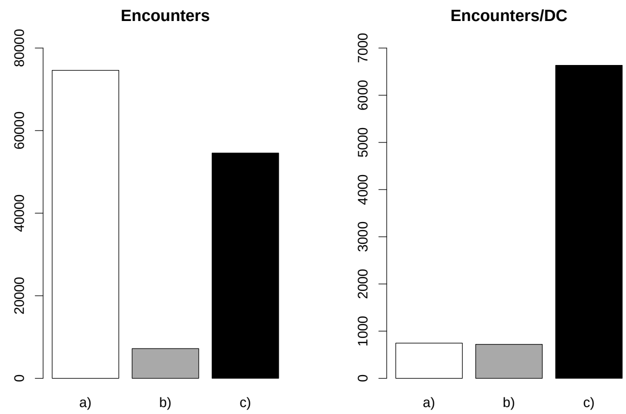


Fig. 8. Deployment results. Comparison of detected sightings, effective duty cycle and the sightings per effective duty cycle for a) always-on b) fixed c) adaptive nodes.

nodes, and environmental sensor data (temperature and humidity). Nodes also created heartbeat messages that reflected their current operational status. This included information such as remaining battery level, memory usage and network statistics. The motivation behind our delay tolerant networking approach was the fact that the majority of the generated data do not have strict latency constraints. It was imperative however, that all data were eventually collected. In order to maximize the battery lifetime of nodes in the network, we used a distributed storage and delivery method, where messages were directed to different destinations based on their tolerable delay. In our system, we offered three priority classes, but this could be extended to any arbitrary number. The three priority classes were as follows:

Priority class 1 represented data with urgent latency requirements (maximum of a few hours delay). These data were forwarded to the 3G-router node for direct access by the researchers. Data of this class could either represent an unusual event or a network status report to ensure the network functioned correctly throughout the deployment.

Priority class 2 represented data with medium latency requirements (maximum of a few days delay). These data were forwarded to frequently visited storage nodes for opportunistic collection. Data of this class could be summaries of badger visits.

Priority class 3 represented data with no latency constraints (delays of weeks are acceptable). All that was required was that they were eventually collected. Data of this class, such as raw sensor data, could remain in memory until collected through a direct download.

Priorities were not only assigned to raw sensor data, but also to composite events or aggregated data. For example, raw badger information may have had priority 3, but when unusually high activity was observed around a certain set this composite event could be assigned priority 1, and would therefore be forwarded to the fixed gateway for immediate delivery. Data priorities could also be either fixed or dynamic, for example, they could vary depending on the zoologists' needs and the data collected from the sensor network.

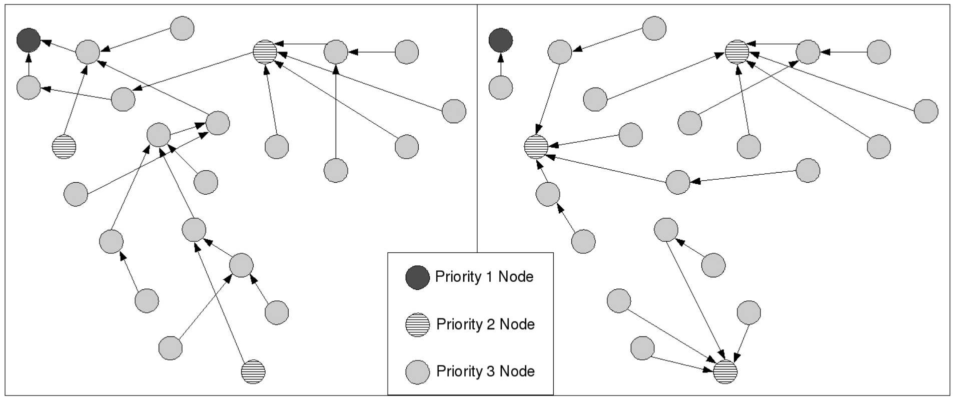
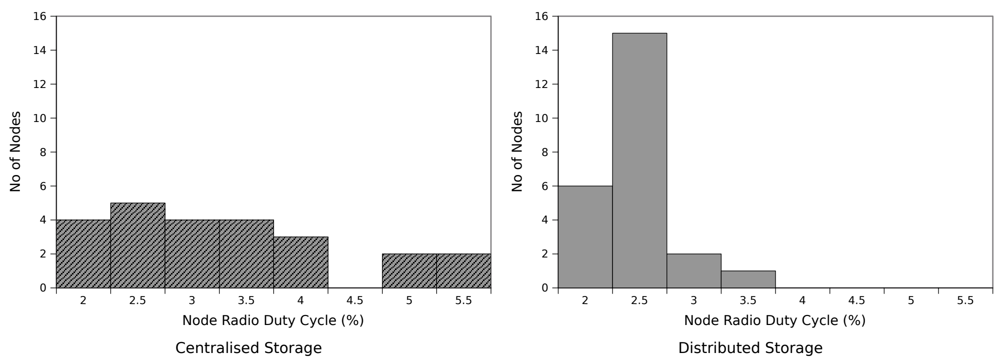


Fig. 9. Example routing trees as found in our deployment: (left) Routing tree for priority 1 data, (right) Routing trees for priority 2 data.



(a) (b)

Fig. 10. Results from the network test: (a) Distribution of node duty cycles for the centralized storage approach. (b) Distribution of nodes duty cycles for the distributed storage approach.

Node priorities: Our priority based in-network storage management approach was very simple and effective. Initially, each node was assigned a priority class P_N based on the frequency it was expected to be visited by mobile sinks for data collection. Some nodes (such as those close to roads and paths) were regularly in contact with a mobile sink and thus contributed a small delay. Other nodes that were placed in rarely visited remote locations were subject to a greater delay.

The more frequently visited a node was, the lower the expected data delivery time, and the lower the assigned node priority class. In our system, the 3G gateway was assigned a priority class 1 as it could offer the lowest data delivery latency. Nodes that were visited at least once every three days by mobile sinks acted as temporary data storage nodes of priority class 2. The remaining nodes in the network had priority class 3.

In our storage management scheme, a data item of priority P_D was stored at the closest node with priority P_N, where $P_N \leq P_D$. Messages with the data priority class of 1 were directed towards the 3G enabled gateway, which allowed users to access them with little delay. Data of priority class 2 were stored at the closest node that had priority 1 or 2. Data of priority class 3 were stored locally at the node where they were generated. Note that node priorities could change dynamically in response to changes in sink mobility. If a node became visited less often, some of the messages that it used to store may then have needed to migrate to another node depending on their priorities.

By asking domain experts to classify data into priority groups, we mapped data to suitable storage nodes, and in this way we ensured that they were delivered on time and with the lowest communication cost. As a data item remained stored at a node, it gradually aged, and its remaining tolerable delay decreased. As a result, it dynamically changed priority and was forwarded to another suitable storage node

Priority- and mobility-aware routing: Once data were assigned a priority and were compressed, they were forwarded to the appropriate destination node, namely 3G gateway nodes of priority 1 or storage nodes of priority 2. Every node maintained a routing table containing the following information for each of the available priority classes:

priority	next hop	seq. no.	dest. node	distance
1	N_A	30	N_E	3
2	N_B	34	N_F	1

The *next hop* simply represented the neighbor to which the data of a certain priority were forwarded. The *sequence number* (seq. no.) and *destination node* (dest. node) fields were used to deal with loops occurring in the network. The sequence number was issued by the destination node and represented the freshness of routing information concerning that node, as in DSDV [Perkins and Bhagwat 1994].

We evaluated the *distance* to a destination node, taking into account the link qualities along the route, in exactly the same way as we evaluated distance to the gateway in Section 3.2. Every node periodically broadcasted its routing table information for each priority class. In our network, we set this broadcast period to 30 minutes. Note that a single advertisement contained routing information for all priority classes. The size of advertisements did not increase with the number of destination nodes, but only in proportion to the number of priority classes. Therefore, the routing overhead of building multiple trees, instead of one, was negligible. Fig. 9 shows the routing trees that were formed in our real deployment for priority 1 and 2 data.

Evaluation: In this section we present results from a 20 day network deployment period with a total of 24 RFID readers. For half of the time, data were collected using the previously described distributed storage approach, and for the other half using a centralized storage approach, as in the initial design. The centralized approach simply forwarded all data to the 3G node; the distributed approach used three additional priority-2 storage nodes at which data were temporarily stored for opportunistic pickup.

In order to have comparable results we utilized a fixed data generation rate for the network evaluation period. Priority 1 data consisted of network status messages generated at each node every 30 minutes, which had to be delivered to the end user within two hours. Priority 2 data consisted of badger activity summaries generated at each node every 15 minutes with a delivery latency of three days. In the centralized approach these data were forwarded to the fixed 3G gateway, whereas in the distributed approach, they were delivered to the nearest storage node that satisfies latency constraints [2].

In both centralized and distributed approaches, a very high delivery ratio was achieved (99.9% of the data were correctly transferred to the appropriate storage or

[2]In our regular network operation, nodes also generate raw badger readings of priority 3, which are stored locally for both approaches, and thus do not incur any network overhead.

Table II. Comparison between the two RFID reader versions.

	Version 1 Reader	**Version 2 Reader**
Node	Tmote Sky	Zigbit Amp
Processor	MSP430	AVR atmega1281V
Node RAM	10 kbyte	8 kbyte
Node Flash	48 kbyte	128 kbyte
External Flash	1 Mbyte	Up to 2 Gb SD
RFID reader power	900 mW	96 mW
Reader turn on time	10 s	0.1 s
Radio range	50 m	1 km
Cost per unit	$590	$320
Mote battery	3 AA	none
Reader battery	18 Ah SLA	18 Ah SLA

3G nodes). Furthermore, this was achieved with an average latency of 14.1 seconds per hop – thus data could be sent over five hops in under 75 seconds on average.

The network status messages, which contained the radio on-time at each node, allowed us to derive the average radio duty cycle of each node over the test period. Fig. 10 shows the distribution of radio duty cycles across the different nodes in the network, with the two storage management schemes. The centralized approach exhibited 46% higher duty cycle than the proposed distributed approach in the average case, and 57% in the worst case at routing hotspots. This shows that by carefully forwarding data of different priorities to suitable storage nodes, we not only reduced the average energy consumption, but also balanced the load more evenly in the network. Our benefits would be much more pronounced if we had forwarded priority-3 data to the gateway in the centralized approach.

5. EVOLUTION STAGE 2: HARDWARE IMPROVEMENTS

Although the algorithms proposed in Section 4 improved the usability of our initial design, our approach was limited by hardware - i.e. the RFID detection node. Experience dictates that rapid field deployment and data gathering are imperative to a system's successful iterative design and deployment - experience in the lab does not translate to success in the field. The detection node was built using off-the-shelf components enabling quick deployment, however these components turned out too general for our specific needs.

5.1. Design of the new node

We incorporated feedback from the users of the system (i.e. the zoologists) in order to make the system more useful. A summary of the major design changes made is shown in Table II, and a photograph of the new node can be seen in Fig. 11. Although the ubiquitous Tmote Sky had enabled us to deploy a prototype system rapidly, its limitations in terms of radio range and usable memory were major constraints. We did not want to design a new custom node from scratch however, rather we wanted to incorporate a more modern and flexible module into the design. The salient criteria were that it should be low cost, power efficient and preferably hand solderable. The cost and power requirements ruled out an advanced node such as the Imote2. Instead, we investigated small, wireless enabled modules that could act as the heart of a generic sensing platform. There were two modules that were a good fit to the application requirements: the Jennic JN5148 and the Meshnetics (now Atmel) Zigbit AMP (ATZB-A24-UFLR). Both of these modules were low power, inexpensive (less than $35 in single quantity) and had an external power amplifier which increased transmission power by +20dBm. They also came in small form-factor packages with numerous peripheral pins that could be used to interface with additional components.

Although the Jennic module had a number of advantages, such as a low power 32 bit processor as opposed to the 8 bit Atmega1281V in the Zigbit AMP, we used the latter as it had better community support, especially in Contiki. This allowed us to port our existing code rapidly from the Tmote SKY platform to the AVR platform, with minor modifications to the existing RF230 radio driver. The radio range of the new modules was improved to be in excess of 1 km in woodland at maximum power, a great improvement that increases the span of the network considerably (note that this is the transmission range of the radio, and not the detection range of the RFID reader, which is unchanged). The drawback of transmitting at the highest power level is that this increases the current consumption from 17 mA to 50 mA.

As the Zigbit AMP is essentially a microcontroller with an embedded radio, we needed to add additional components in order to satisfy application requirements. Firstly, we added external memory to the board in order to remove the constraints present in the initial system. The board is equipped with a 4Mbyte serial dataflash chip and also a removable mini-SD memory card. At present, this allows the addition of up to 2 Gbytes of SD based flash, but larger capacities could be supported with modifications to the SD driver software, allowing high capacity cards to be used. We also added an RTC with battery backup to allow nodes to maintain their time when batteries were changed. Currently, nodes are unsynchronized – this is an issue that will be addressed in subsequent firmware iterations. One problem with the Tmote Sky is that the onboard sensors are not removable. This is not a problem in an indoors laboratory setting, but in a real deployment, sensors must be placed externally to the protective housing. Thus, we incorporated light and temperature sensors, which could be detached from the main board.

A major change in this version was the switch from the RS-485 version of the RFID reader to an OEM board. The RS-485 version was a suitable choice for the initial deployment, as it allowed us great flexibility in daisy chaining multiple readers together and had a simple serial interface. However, the power consumption and slow turn-on time were issues. These high power requirements necessitated the use of a separate reader and mote batteries, so that the mote would remain powered even if the reader exhausted its supply. Switching to the OEM version of the RFID reader negated these problems. It has a simple synchronous serial TTL interface and a pin that could be used to trigger an interrupt on the microcontroller when a tag was read. This allowed us to power down the microcontroller while the reader was active, whereas in the previous version, we had to maintain the clock for the UART. Furthermore, in the Tmote Sky, the radio and the UART were multiplexed, which led to a lot of problems with hardware locking to prevent concurrent access to the peripherals. In the new version, the RFID reader had its own dedicated pins. The turn-on time for the OEM reader is under 100ms, and it uses 96 mW when active.

Lastly, we used a simpler power distribution system, with 3V as a common rail. A small charge pump was used to generate the 5V required for the OEM RFID board. The nodes can be powered either from a 3V battery or from a 12V battery using a switching regulator. We also included a small prototyping area on the board, as our prior experience had shown us that there were often instances where we would want to connect an additional device (such as a moisture sensor) to a node.

In summary, the new version of the detection node has dropped the power consumption by nearly an order of magnitude. The storage space has been increased to such an extent that it allows for 40 years of storage at the current generation rates, as opposed to one week. This will allow us to gather more information and sample environmental sensors at a much higher resolution. The communication range has also been increased greatly, which allows the network to cover a much larger area with

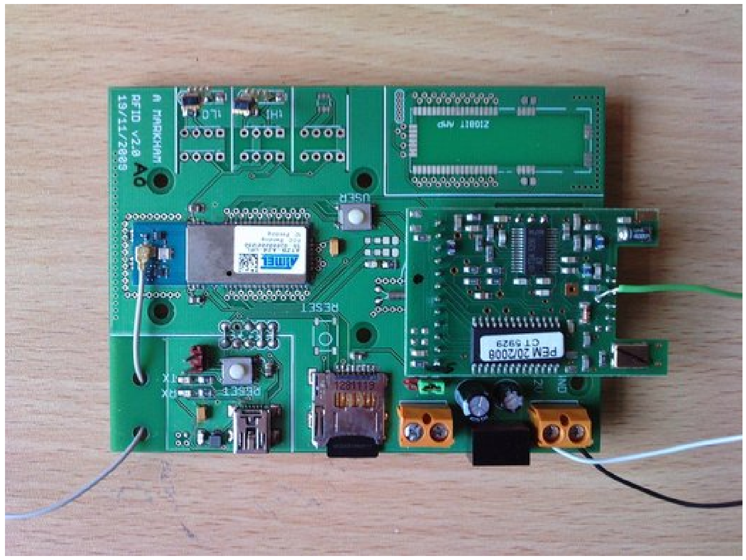


Fig. 11. The second version of the node.

fewer devices. However it must be stated that it was our experience garnered from the prototype deployment that allowed us to design a well optimized successor.

5.2. Duty Cycling Revisited

Given that the RFID reader on the new node could be powered up in 0.1 s, as opposed to the 10s for the previous version, the parameters for the learning algorithm presented in Section 4.1 could be modified. The original $T_{interval}$ was set to 330s, with a duty cycle of 9%. Although this saved a large amount of power, allowing the node to operate for longer, it had the drawback of not being able to react to the presence of animals outside of the normal predicted times, as the off time could be quite long (up to an hour). In order to address this, we modified T_{on} to be 1s, with $T_{interval}$ to be 11s. This still resulted in a 9% duty cycle, but the short term adaptability could react to the presence of unusual events, for example a badger emerging during the day. The longest time for which the reader was off was reduced to less than a minute, which increased the chances of detecting animals, while still accounting for their nocturnal behavior.

Fig 12 shows the simulation results for the same set of data as in Section 4.1, with new parameters. The shorter wake up interval resulted in both higher encounter detection and energy efficiencies. The short-term adaptation algorithm detected 91% of all encounters while working at 9% duty cycle. The combination of short-term and long-term algorithm resulted in 89% detection rate at much lower 5% duty cycle. The deployment results conducted with the same parameters are shown in Fig 13.

5.3. Data Collection Revisited

The hardware improvements introduced in Stage 2 had a dual effect on the data collection process. Change in sink mobility patterns: Recall that mobile sinks are domain scientists that roam through the woods and opportunistically collect data from storage nodes. Some of them are zoologists visiting the network for maintenance purposes, whereas others are from other disciplines visiting the woods for their own purposes unrelated to our sensor network. The visits of the former were reduced because hardware optimizations made the change of batteries less frequent. With fewer mobile sink

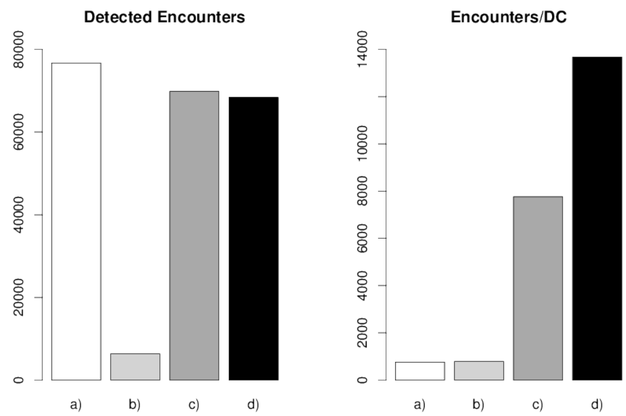


Fig. 12. Simulation results. Comparison of detected encounters and encounters per effective duty cycle for a) always-on b) fixed c) short-term adaptation d) short-term and long-term adaptation.

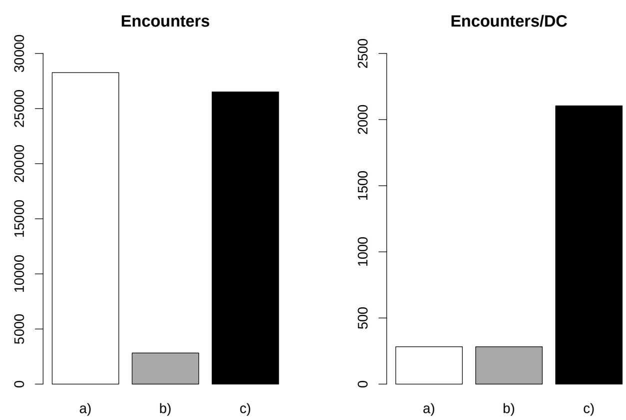


Fig. 13. Deployment results with new hardware. Comparison of detected encounters and encounters per effective duty cycle for a) always-on b) fixed c) adaptive algorithms.

visits, one would expect an increase in the data propagated over multiple hops through the fixed network, and thus an increase in the average and worst-case communication cost.

Change in communication range: the effect of reduced sink mobility was, however, offset by the significant increase in the communication range of fixed nodes. Recall that the hardware optimizations introduced in the second stage dramatically increased the communication range of sensor and badger detection nodes from 50m to 1km. As a result, all nodes had one-hop connectivity to the fixed 3G gateway, and no longer needed to make use of mobile sinks.

Hence, in the second evolution stage, the hierarchy of nodes (based on priorities) collapsed and the use of mobile sinks to collect data efficiently and relieve the fixed network, became less relevant. This shows that the benefits of software-level optimizations, such as the priority-based delay-tolerant data collection, are tightly dependent on the hardware used. The priority-based delay-tolerant scheme proposed in Section 4.2 yielded significant benefits in the first version of the system, but proved of little use to the second one. However, we expect it to become relevant again in the near future, when we proceed to the third evolution stage of the system. Our short-term plan is to scale-up the network to cover a larger area. The extended network of badger and environmental sensor nodes will again become a multi-hop network, and the data collection scheme will be reinstated.

6. NETWORK MAINTENANCE COSTS

In this section, we will describe the evolution of our system in terms of the costs involved. As a baseline, we will also show the approximate cost of conventional VHF tracking [Kenward 2001]. This involves collaring animals with VHF tags that emit periodic radio signals. VHF tags are analogue devices achieving individual identification by frequency separation, and limiting the number of IDs available. On the other hand, active RFIDs are digitally encoded allowing more IDs in a given band without the need for a receiver to scan multiple channels. The VHF tags are detected by receivers carried by field-workers at a range of tens to thousands of meters (depending on environmental conditions). Using triangulation (requiring at least two people on the ground), the approximate location of the animal can be found. Our RFIDs transmitted at much lower power than VHF tags increasing battery life, while limiting range (c. 30m), so giving a more precise location estimate for tagged animals. VHF tracking has been a popular method since the late 1960s because it was, and still is in many circumstances, the only way of tracking wild animals.

Note, although we are comparing the costs between VHF and our system, the data collected by the two methods are rather different - although they collect the same information (i.e. the location of a specific animal), our system logs an animal about twice a second when it is within range of a detection node, while this is not the case for VHF tracking. The more animals tracked by VHF, the more human trackers are required on the ground up to the point where the number of trackers risks disturbing the animals being tracked. Our system instead offers continuous automatic detection (presence/absence) of the animals at specific locations with minimal interference.

From previously tracking studies of badgers, using VHF, we know that at least one person is needed to work for about 10 hours a night to track one animal. If we assume we have enough people to work for 28 days, this would result in 280 hours per person, costing 2,030 USD using a 7.25 USD/h wage. It is easy to see how this is not feasible in the long run, especially, because one person can often only track one animal at a time. It is also not possible to provide continuous tracking (i.e. 24/7) without considerable costs and man-hour overhead, not to mention the fact that the more people there are in the woods, the more the animals are disturbed.

Importantly, there are several other methods of animal tracking, such as the GPS and ARGOS satellite-based systems that we do not include in the direct comparison. They are inappropriate because of inferior spatial resolution (ARGOS) and reliability (GPS performs poorly in woodland). Furthermore, ARGOS tags can cost over 1500 USD each and a badger-sized GPS tag lasts for only a few months, whereas our RFID tags cost in the order of 60 USD each and last for ca. 2 years. Our RFID readers and sensor motes also contributed to the total cost of our system, however the price of each detection node was around 300 USD, thus still less than a comparable ARGOS system.

Table III. Breakdown of the average cost incurred to maintain each stage of the system for 4 weeks. Costs are normalized with respect to the number of animals being monitored.

	Visits (HRs)	Battery cost (USD)	Total cost (USD)	Detection per animal	Cost per detection per animal
Stage 1 (HW only)	29.7	156.76	372.5	56107	0.006
Stage 1 (HW & SW)	10.8	52.9	131.4	40958	0.002
Stage 2 (HW only)	2.7	1.04	20.615	56107	0.0003
Stage 2 (HW & SW)	1.3	0.56	10.3	56107	0.0001

We deployed 74 RFID tags and 26 detection nodes, summing to 4440 + 7800 = 12 240 USD, while buying 74 ARGOS tags would have cost us approximately 111 000 USD.

Table III shows the summary of the costs involved in maintaining our system. We consider the number of man-hours needed, as well as the battery costs for each stage. The total cost includes the price of monthly up-keep of the system. We also include how many animal detections we had recorded in a month and how much each of these recordings cost. In our deployment we had only two main sources of costs, maintenance visits to the woods by the zoologists and the costs involved in battery consumption and charging. By maintenance visits, we mean the regular visits to download the data and change the batteries on the detection nodes. Early in the deployment, we had to go to the study area more frequently to fix bugs and make small improvements on the devices. Due to the lack of a remote reprogramming feature, we actually had to go and manually re-flash the devices - this added extra cost, but was not included in this evaluation, as it did not majorly contribute to the total maintenance cost of the system. However, it emphasized the need for a remote reprogramming feature in our future deployments. Developing the new software and hardware for each stage also added to the total cost, however this was excluded from our evaluation. There were two PhD students and two post-doctoral researchers working on the project for 3 years, however it is difficult to estimate accurately the number of working hours spent developing the system.

Stage 1 is our initial hardware node deployed. We have logs of how much money we spent on batteries and how much time we spent in the woods. Each detection node was made up of an RFID reader and a Tmote Sky. The Tmotes were powered by AA batteries, while the readers were powered by an 18Ah 12V batteries. We spent about 147 USD on AA batteries and about 8.9 USD (4 times a month, using 0.4 kWh for 20c/kWh) on recharging the reader batteries on all 26 detection nodes. From the logs, we also see that about 30 hours per month were spent in the woods, summing to 372 USD (again, using 7.25 USD hourly wage). From our database, we collated the total number of active tags per month during the deployment, as well as the number of detections per month; thus on average, one animal generated 56,107 records per month, giving a single detection cost of around 0.6 cents. At this point, the bottleneck became the 1 MB storage on the detection node - without compression, this became full (depending on activity) within a week, however, using our data compression technique, we were able to extend this to double the lifetime of the nodes, requiring only two field visit per month, totaling 10 hours. The adaptive duty cycling approach allowed the battery costs to be reduced to about 53 USD, or 131 USD per month. Slightly fewer records were generated, but a single record still cost less than in the previous stage i.e. 0.2 cent.

In stage 2 we introduced new hardware that radically increased the lifetime of the detection node, while yielding the same number of sightings as in stage 1. In our first stage 2 deployment, we put the hardware out for testing, without any software enhancement (such as duty cycling either the radio or the reader). The node lasted for 2 months on the same battery, and due to its extensive memory capacity, did not require data download. Since our new hardware used one large, rechargeable, deep cycle battery, this negated the need to buy AA batteries for the motes. The charging costs of the large batteries amounted to 0.2 c/kWh x 0.4 kWh x 26 x 0.5 (once in two months) = 1.04 USD per month. On average, one visit lasted for about 5.4 hours, so one visit for two months resulted in 2.7 hours per month. Since we needed to visit the nodes once in 2 months, our monthly cost was 2.7 hr x 7.25 USD + 1.04 = 20.615 USD. The cost of a single detection was reduced to 0.03 cent. It is worth noting, however, that at this point, the cost of getting to the woods or tagging the animals is actually higher than the maintenance cost.

The introduction of the enhanced software in Stage 2 (described in Section 5) further extended the lifetime of our new hardware. We obtained a 2-fold increase in the lifetime of the node, hence only one visit in every 4 months became necessary. This resulted in a maintenance cost of 10.3 USD per month, and the cost of a single detection thus became negligible.

7. DATA ANALYSIS

We have collected over 29 million records since the system became fully operational in March 2009. This section analyses a subsection of these data (from 14 March 2009 to 19 September 2009) for illustration only, to demonstrate the utility of the system in generating biologically useful data. In doing so it is important to note that we do not attempt to infer biological significance from any of our observations, instead our analyses are purely descriptive. The full dataset, including microclimatic correlates gathered from sensor nodes, will be subject to zoological analysis elsewhere.

7.1. Data Gathered

Badgers were trapped up to four times a year for a concomitant research project [Macdonald and Newman 2002]. This provided an opportunity to put RFID tags on the animals. There have been 9 trapping sessions since June 2008, during which 74 animals were tagged. Animals were able to remove 12 tags (collars), which were found on the ground. More tags were similarly lost, but not found. Whenever possible, these animals were retagged. Over the year, a lot of attention was given to keeping the system running uninterrupted, i.e., always replacing the batteries and downloading the data before the nodes stopped functioning. We set up a database where all the records were uploaded.

7.2. A Window into Badger Movement Patterns

One of the advantages of our automatic monitoring system has been that we were able to capture data with high temporal resolution from our fixed detector sites. This allowed us to produce records of daily badger activity for future zoological analyses.

A density plot of badger ‘sightings’ is shown in Fig. 14(a). The horizontal axis shows the time in 24-hour format. The vertical axis shows the day of year. The intensity of each dot represents the average amount of time that badgers were observed at the detection nodes.

In the evening, badgers exited their setts (indicated by the strong dark line at dusk in Fig. 14(b)). They then visited the latrine nodes probably foraging for food in between. At the end of the night, they returned to their setts, producing a high density of activity on the right side of Fig. 14(b).

Regarding seasonal trends, and as expected, it can be seen that the length of time that badgers were out of their setts decreased, reaching a minimum around day 170

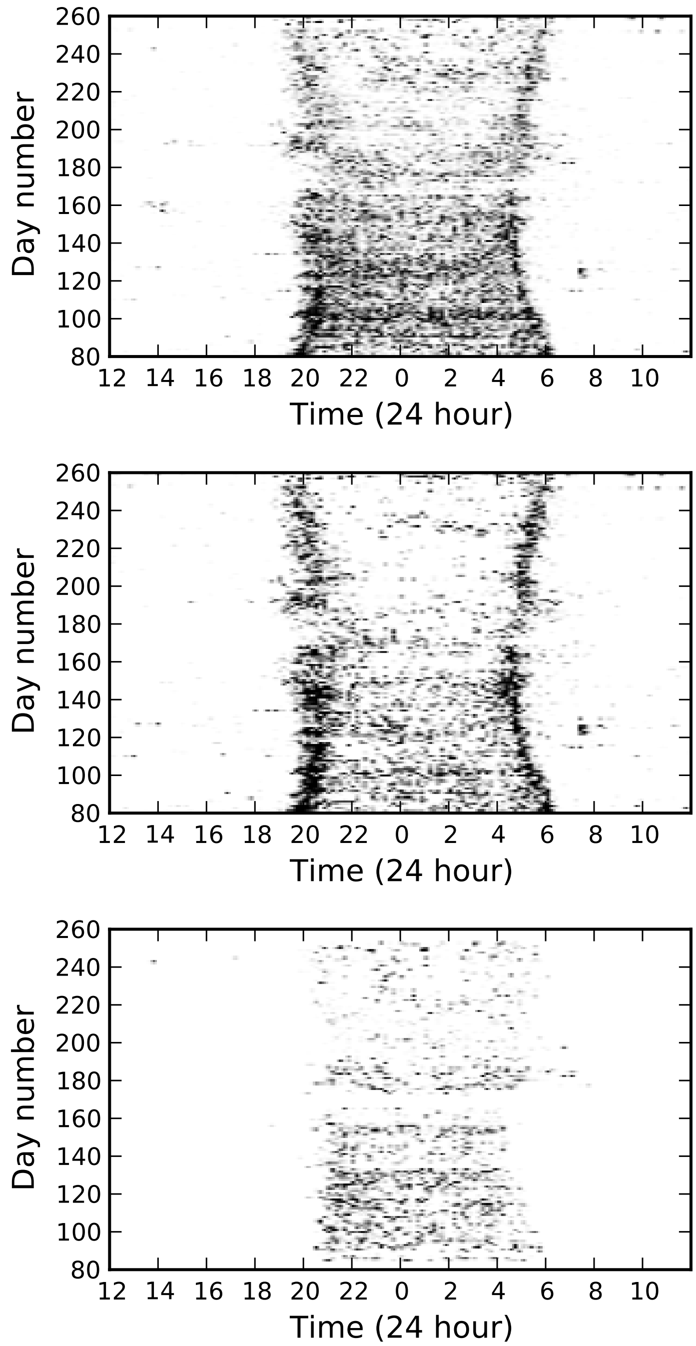


Fig. 14. Badger activity captured at detection nodes. Horizontal axis is time of day and vertical axis is day of year. (a) Badgers detected at any detection node. (b) Badgers detected at nodes placed near setts. (c) Badgers detected at nodes placed near latrines.

(corresponding to June 18). From this point on, the average trip time starts to increase again, with decreasing day-length.

7.3. Correlation of badger activity with night-length

As badgers are nocturnal mammals, it was expected that there would be a seasonal variation in their behavior correlating with the number of hours of darkness per night. A correlation analysis was conducted on 372 nights of data in order to determine whether indeed there was a significant relationship. We defined a badger trip-length as the difference in time between when it first emerged (and was observed at any detection node) and when it was last seen within the system for each night. The median trip-length was then used as a metric, as it is robust to outliers. Fig. 15 shows the variation in median trip-length and night-length with the day of the year. The four gaps in data reflect periods when trapping was undertaken. As the badgers are disturbed, these data have been removed from the analysis. It can be seen that our data suggest a cyclical trend in badger activity, with a minimum occurring in the middle of the year. As expected, badger activity appears to have peaked in October and mid-February with decreases towards the beginning and end of the year. In order to demonstrate this relationship more clearly, a scatter diagram between trip-length and night-length was plotted and is shown in Fig. 16. There is a highly significant correlation between the two variables ($\rho(372) = 0.301$, $p < 0.05$). Interestingly, when the data from January and December are removed from the data series, the correlation becomes even stronger ($\rho(318) = 0.605$, $p < 0.05$). This hints that there is possibly another effect occurring during the winter months; this is currently under investigation.

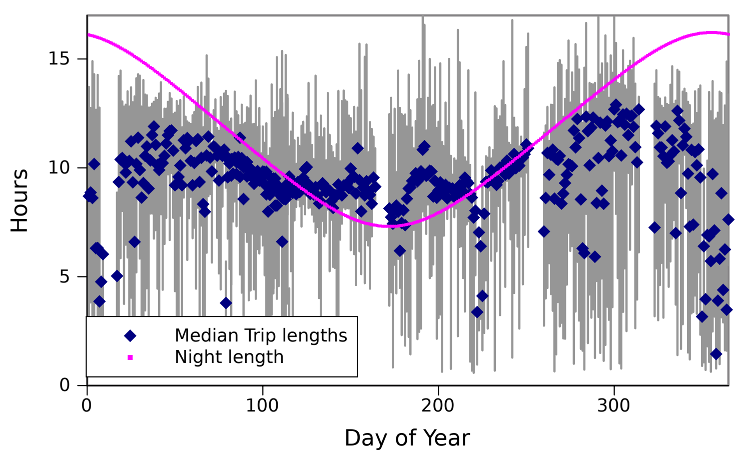


Fig. 15. Variation in badger trip-length with day of year. The error bars show the interquartile range of trip-lengths. Also shown is the number of hours of darkness.

7.4. Badger Co-location

We extracted pairwise co-locations between badgers from the detection node records: our assumption was that two animals were within 0-60m of each other if they were recorded contemporaneously by the same detection node. Because we do not have any indication of the type, if any, of social interaction between the animals, we must be cautious in any assumptions we infer. Nonetheless, this methodology allows us to provide a high-level description of co-location patterns.

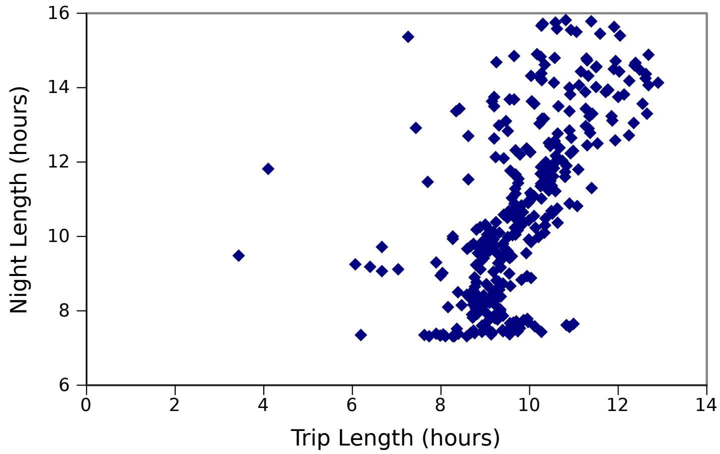
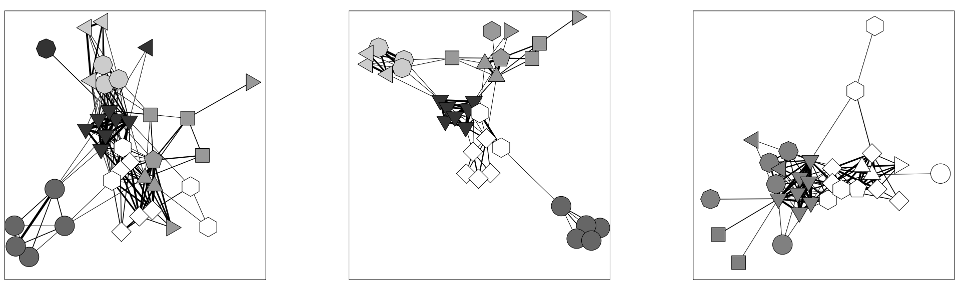


Fig. 16. Scatter diagram showing the relationship between trip-length and night-length.



(a) (b) (c)

Fig. 17. Badger social networks: The shade of a node (node = badger) represents the social community it belongs to while its shape denotes the sett it lives in. Different networks are created by using (a) all co-locations between animals at setts and latrines, (b) only co-locations at setts, and (c) only co-locations at latrines.

Since setts and latrines have different social functions for badgers, co-locations are divided into three datasets according to where they take place: (a) setts and latrines together; (b) setts only; (c) latrines only. To investigate the broad social structure we create a weighted social graph for each co-locations dataset where nodes represent badgers and the weight of each link is proportional to the amount of time for which the two animals were co-located.

Fig. 17 illustrates the resulting graphs for each dataset, where communities have been detected using the algorithm described in [Blondel et al. 2008]. The network described by all co-locations in Fig. 17(a) depicts 5 discrete algorithm defined 'communities', but each interlinked with one-another. This is not as evident for the network defined from sett co-locations (Fig. 17(b)) where the 5 similar communities are more discrete with fewer links between them, giving greater separation. Conversely, for the network defined from latrine co-locations (Fig. 17(c)), only two communities were in evidence.

8. RELATED WORK

Wildlife and Environmental Monitoring A number of other wildlife monitoring deployments also exist like Zebranet [Zhang et al. 2004], DuckIsland [Szewczyk et al. 2004] and TurtleNet[Gorlick 2007]. Sikka et al. [Sikka et al. 2006] discuss the deployment of a hybrid network consisting of mobile sensors mounted on farm animals and fixed sensors measuring soil moisture and weight of food and water consumed by animals. Selavo et al. [Selavo et al. 2007] describe the deployment of wireless sensor network for measuring complex light environment in thickets and also use delay tolerant networking, fault-tolerant distributed storage and custom hardware. A number of modified Mica2 motes were deployed by Gilman et al. [Tolle et al. 2005] to monitor the microclimatic conditions and solar radiation in a redwood tree for 44 days. [Naumowicz et al. 2008] describe the design and deployment of a pilot sensor system for monitoring seabirds using passive RFID technology. The sensor nodes were based on a modular MSB platform with a custom extension board. The data were collected by a base station PC located in the centre of a network deployed in a star topology. [Rutishauser et al.] describe the design and field testing of a sensor system to monitor physiology and behavior of wildlife animals. The system consisted of animal-borne sensor nodes equipped with 3-axis accelerometer, GPS and a short range radio. The network used delay tolerant routing and fixed relaying nodes for data collection. Barro Colarado Island ARTs [Kays and Wikelski 2007] is a long standing system for large scale animal tracking wearing radio-transmitters. The system consists of a wireless network of seven tall Automated Radio Telemetry System towers (ARTs) deployed on hilltop towers above the forest canopy. Their system is, however, not as low cost and easy to deploy as Wildsensing is. Finally, Encounternet is an ongoing project for tracking small animals [Burt et al. 2010] and uses 38 base stations deployed in a 2-3km area to monitor tagged birds.

With respect to these we have developed a highly integrated heterogeneous deployment which enabled us to gather very large quantities of data. Moreover, the system was able to customize the distribution of the data depending on the urgency of the delivery required. The combination of low cost tags and energy efficient ground sensor network enabled an autonomous and non-intrusive monitoring of wildlife animals at much larger scale than previously possible.

Duty cycling: [Mainland et al. 2005] propose a machine learning based approach for adaptive resource allocation for sensor networks. Here, the sensors were modeled as self-interested agents that attempted to maximize their profit and a simulation based evaluation was presented.

Data collection: The MRME algorithm [Ekici et al. 2006] scheduled mobile sinks to visit static nodes before data delays expired. When data were close to expiration, multi-hop routing was used to guarantee timely data delivery. Unlike our approach, the MRME algorithm assumed both control over sink mobility and homogeneous data latency requirements for all data. The SensorScope project [Barrenetxea et al. 2008] described the deployment of a low duty-cycle sensor network in which a central base station gathered data. SensorScope used a non-standardized networking stack that was designed for remote areas that cannot be frequently accessed. Hui et al. [Hui and Culler 2008] demonstrated the usability of the IPv6 standard for sensor networks as a flexible networking layer whilst maintaining a very low duty cycle - our choice of network stack was strongly influenced by their findings.

In Lance [Werner-Allen et al. 2008], each data unit had an associated value, as well as a cost for multi-hop data delivery to a single basestation. Values and costs were taken into account to determine download scores, i.e. the priorities of data units for data delivery. Unlike Lance [Werner-Allen et al. 2008], we not only use priorities to rank data units, but also to rank storage nodes. In addition, we sent data of different priorities to different storage nodes immediately, instead of delaying their delivery to a single node (the basestation). Jiang et al. [Jiang et al. 2007], propose EMA, an energy management architecture that enables prioritized enforcement of policy directives. If, for example, there were sufficient energy resources in the network, a sample-and-send directive was used, whereas the system gracefully degrades to sample-and-store when energy resources become scarce. Their framework could be combined with ours to offer a greater variety of policies. For example, a directive could suggest that when energy resources are scarce, a class of data must be demoted to a lower priority. As a result, these data could be delivered to a closer but less frequently visited storage node, and will incur a lower energy cost. Unlike existing systems, in which prioritization results in a binary decision (store vs. download), our system uses data priorities to select among a wide variety of data delivery options.

Evolution: With any design, it is very difficult to 'get it right' the first time, despite a lot of planning and effort. We have shown how our systems developed over time, and how we have managed to reduce the maintenance cost to a tenth of the initial costs, while still collecting substantial quantities of data.

Similarly, the authors of the ZebraNet project [Zhang et al. 2004] describe the different stages of hardware upgrade they went through in their deployment. They deployed 3 different sensors, each improving on the capability of their previous ones. The improvements included solar panels, changing the radio to a more energy efficient one and increasing the on-board memory. The Glacsweb Project [Martinez et al. 2009] was developed to monitoring glacial dynamics through the use of WSN. They have had yearly deployments from 2001 to 2008 in different regions and countries (including Norway and Iceland). Their deployments relied on a number of 'probes' embedded in the ice, and a base station, relaying data back from the sensors to the scientists. The base station turned out to be their single point of failure, they redesigned it from deployment to deployment to improve on reliability and robustness. [Barrenetxea et al. 2008] designed a wireless sensor system for environmental monitoring and deployed it in a number of environments over the period of two years. The first deployments tested the hardware, with subsequent deployments focusing on networking layer and in-situ maintenance.

Although we detail similar evolutions to the aforementioned projects, our overall aims were different. Here we not only focus on the long-term maintenance of our system and general improvement in its reliability, but we also reconcile the inherent relationships between the necessity for specific software and hardware evolutions and the resulting cost savings and benefits from such actions.

9. LESSONS LEARNED

Although there is currently a lot of work on building real sensor systems, very few attempts have been made to deploy them in the field and then maintain and develop them. In this paper, we provide details of the first distributed active RFID-WSN hybrid system for wildlife tracking. We undertook an iterative process of software and hardware designs and developments, while still maintaining backwards compatibility.

Maintenance Costs. We gained invaluable experience from our deployment. System maintenance is a key to a long-term deployment, and the costs associated with it should be factored in from the initial design stages. Though our first stage was very successful in collecting large quantities of high quality data, maintaining it turned out to be more expensive than initially expected. For a wildlife monitoring application, continuous operation is essential therefore maintenance is unavoidable.

Software and Hardware Interaction. With software enhancements, we were able to increase the lifetime of the system, and thus decreased the necessary maintenance,

Fig. 18. A sheep wearing an original RFID tag.

however this resulted in the hardware becoming our limiting factor, hence our second lesson: to achieve maximum power efficiency, application-specific hardware is often necessary. With our second stage, we were able to decrease the maintenance costs to a fraction of what they were before, while collecting the same amount of data. Optimizations that work on some hardware, however, might not perform as well on a different system, i.e. software optimizations need to take into account the capabilities and the characteristics of the hardware. The introduction of the new hardware resulted in fewer visits to the woods by zoologists, which affected the in-network-storage. Moreover, once the new hardware was in place the detection node duty cycling could be improved with finer grain parameters which would not have been possible on the earlier version of the hardware.

Rapid Initial Prototyping and Deployment. One of our most pertinent results was the realization that no initial deployment would satisfy all our design requirements. This suggests that the best approach for long term monitoring systems is to design a prototype that can be rapidly deployed using commercial off-the-shelf technology. This is especially important in applications like ours, where no prior data had been collected on a similar scale in the same environment. Although our initial prototype suffered from a lot of practical problems, it was easy to get the system working in the field. Before deploying the system in its final location, the woods, we went through a series of test-phases. Once the devices proved to be reliable within the lab, they were put on sheep for further testing. Sheep are much more accessible and easier to handle than badgers, while providing an environment similar to the final scenario. The original (i.e. before being prepared for badgers) tags were put on collars, worn by sheep to see how their bodies influence the detection range, as well as how environmental factors affect the tags and detection nodes. One of the sheep is shown on Fig. 18, wearing a tag. Further to the sheep testing, we tested the final tags on model badgers, as explained in Section 3.3. Rapid prototyping allowed us to collect suitable data to understand how we could improve the system as a whole. These observations then guided the evolution of the system, allowing us to dramatically reduce the cost of system maintenance by

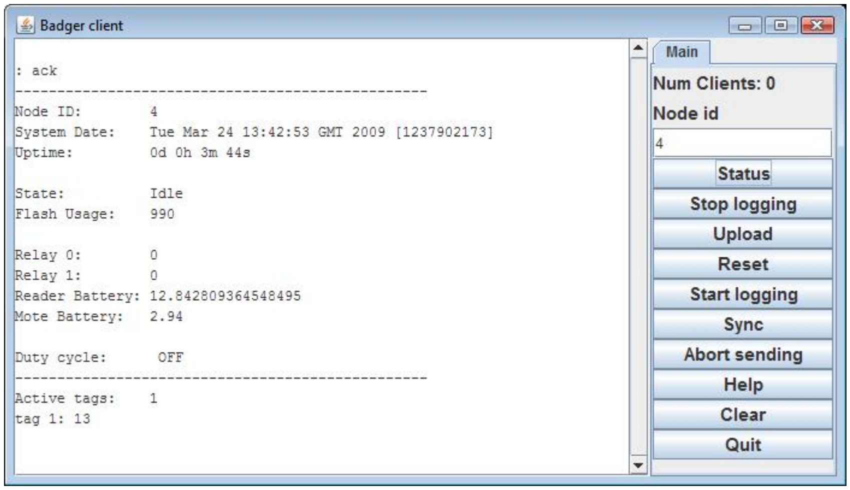


Fig. 19. Simple graphical user interface for the system.

increasing the runtime of devices. Note, no amount of simulation or laboratory testing is equivalent to problem solving in real deployments. Failures are common, and some failures, such as animals interfering with equipment are unquantifiable until the system is actually deployed. Thus we suggest that researchers deploy an initial version (even if it is a datalogger) as soon as possible, so that knowledge can be gained about practical problems.

Gradual versus step-change improvements. In the evolution of a system, a choice has to be made whether to improve it gradually or to switch over to a new system entirely. The choices made here were influenced by the needs of the application. In our case, we had to slowly incorporate new improvements, testing them over a period of months in the field, so as to gather a continuous record of data. This was because any gaps in the data could significantly reduce their biological significance. Other applications can tolerate interruptions that allow for all effort to be concentrated on designing and deploying a new and improved version. This leads naturally to step-changes in capability and functionality, with all components of the system being upgraded simultaneously. This is an important lesson, as it dictates the type of evolutionary strategy that can be adopted.

Continuous interaction with domain scientists. Our system was built as an *experimental tool*, as opposed to a proof of concept. The design of smart protocols and algorithms to reduce message overhead or energy consumption is only useful if it complies with the requirements of the eventual users of the system. Such interactions are not only useful to make sure the system works as expected, but also to provide interesting ideas for optimizations. One such key observation made when discussing system requirements with the domain scientists was that not all data had real-time requirements. In response, we formulated a priority based routing approach that reduced traffic load, particularly around network hotspots, by forming multiple routing trees. However, another application specific factor we took advantage of allowed data to be collected opportunistically by zoologists working in the woods. To reduce these data volumes, a simple lossless compression algorithm was devised. Further to the improvements in the routing design, much simpler enhancements improved the usability of our system. In our first version, we controlled the devices using a simple command-line interface. For computer scientists, this is sufficient, however, typing commands in

the forest turned out to be very uncomfortable and troublesome. Our users had to type commands under harsh weather conditions - strong wind, rain and cold. This problem was ameliorated by creating a simple java-based GUI (shown in Fig. 19). This produced a major improvement, and halved the time it took to deal with each device. By understanding the needs of the users, we were able to tailor our system design, extending the lifetime of devices in the network, whilst still satisfying application requirements.
Remote reprogramming and re-tasking Initially, high level system requirements were defined from the beginning, however detailed specifications were missing due to lack of enough information about the problem. It is not uncommon to have limited information before deployment - many important system parameters become apparent once data starts flowing in. In our wildlife monitoring scenario, many of the issues (such as tag-beacon period, radio frequency used, quantities of data stored, etc) became clear only after the first few weeks of the deployment. Our strategy was to try to prepare for the unknown, and collect everything as frequently as possible. This, however, makes it very difficult to optimize many parameters (such as duty cycling or memory usage), and may introduce redundancy. The point is, we had no way of changing any system parameter on the fly, remotely. What is more, we had no way of fixing bugs or reinstalling applications on the sensors without actually visiting the devices in the deployment area, and re-flashing them manually - costing money and time. Although parameter tuning and debugging should ideally happen in the testing phase, it is inevitable that some things need adjustments or fixing. In a more general case, the feature of remote reprogramming or re-tasking is essential to fully utilize a deployed system. Generally, sensor networks (including ours) are mainly application specific - all the functionalities are hard-coded before deployment. This is acceptable during the first iteration of a system, since the aim is to get the system working reliably. However, once sensors (and the software running on them) are mature enough so that developers do not need to spend most of their time fixing low-level, operating system or driver related issues, the first priority is not *making it work*, but rather *how to make the most out of the system we have*. We iteratively achieved a huge improvement on our initial system, however, there is always room for smaller improvements. Furthermore, considering the second version of our detection nodes had the potential to out-live the tags attached to the animals highlights the chance that system requirements might completely change with time - the sensors could be used for something completely different. This emphasizes the importance of a remote re-tasking or reprogramming scheme that can re-use an already deployed system. Remote reprogramming and re-tasking is a hot topic for WSN research[Wang et al. 2006] due to the challenges and requirements described above. In the scope of this work, we did not focus on this issue, however this is one of the next problems to address in our future iterations.

10. CONCLUSIONS

We learned a number of interesting lessons. First, network maintenance should not be an afterthought, but a key consideration in the original design of the system. If not, then maintaining a sensor network can become far more expensive than building it. Second, before delving into algorithmic improvements and strenuous testing of new software, it is important to carefully consider hardware limitations. Sometimes it is more cost-efficient to replace the hardware platform than to design and test new software for an existing platform. Third, the benefits of software optimization (e.g. improving sampling, storage and data collection algorithms) largely depends on the hardware. An algorithmic improvement that yields significant benefits on one platform may be less efficient or even not applicable to another. Fourth, engineering sustainable sensor networks is an iterative process that alternates between hardware and software changes. Last, these changes must be performed in a controlled manner so that they do not disrupt the data collection process.

We believe the results and conclusions in this paper will provide an important insight into the workings of a long-lived outdoor wireless sensor network deployment.

11. ACKNOWLEDGMENTS

The Wildsensing project was funded by EPSRC. We also acknowledge the support through project EPSRC CREAM. The Wytham badger project is funded by the People's Trust for Endangered Species. Special thanks to Chris Newman (CN) and Christina Buesching and team for help with badger trapping and CN for comments on this paper. Wavetrend provided advice and support in the use of their RFID technology.

REFERENCES

ASCHOFF, J. 1965. *Circadian Clocks*. North Holland Press.

BARRENETXEA, G., INGELREST, F., SCHAEFER, G., VETTERLI, M., COUACH, O., AND PARLANGE, M. 2008. SensorScope: Out-of-the-Box Environmental Monitoring. In *Proceedings of IPSN '08*. IEEE Computer Society, Washington, DC, USA, 332–343.

BEUTEL, J., GRUBER, S., HASLER, A., LIM, R., MEIER, A., PLESSL, C., TALZI, I., THIELE, L., TSCHUDIN, C., WOEHRLE, M., AND YUECEL, M. 2009. PermaDAQ: A Scientific Instrument for Precision Sensing and Data Recovery in Environmental Extremes. In *Proceedings of IPSN '09*. IEEE Computer Society, Washington, DC, USA, 265–276.

BLONDEL, V. D., GUILLAUME, J.-L., LAMBIOTTE, R., AND LEFEBVRE, E. 2008. Fast Unfolding of Communities in Large Networks. *Journal of Statistical Mechanics 2008,* 10, P10008.

BUETTNER, M., YEE, G. V., ANDERSON, E., AND HAN, R. 2006. X-MAC: A Short Preamble MAC Protocol for Duty-cycled Wireless Sensor Networks. In *Proceedings of SenSys '06*. ACM, New York, NY, USA, 307–320.

BURT, J., OTIS, B., MENNILL, D., DOUCET, S., MAYNARD, D., AND WARD, K.-A. 2010. Use of Wireless Smart Tags to Monitor lek Visitation by Female Long Tailed Manakins (Chiroxiphia linearis). Tech. rep. May.

DURVY, M., ABEILLÉ, J., WETTERWALD, P., O'FLYNN, C., LEVERETT, B., GNOSKE, E., VIDALES, M., MULLIGAN, G., TSIFTES, N., FINNE, N., AND DUNKELS, A. 2008. Making Sensor Networks IPv6 Ready. In *Proceedings of SenSys'08*. ACM, New York, NY, USA, 421–422.

DYO, V., ELWOOD, S. A., MACDONALD, D. W., MARKHAM, A., MASCOLO, C., PÁSZTOR, B., SCELLATO, S., TRIGONI, N., WOHLERS, R., AND YOUSEF, K. 2010. Evolution and Sustainability of a Wildlife Monitoring Sensor Network. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*. SenSys'10. ACM, New York, NY, USA, 127–140.

DYO, V. AND MASCOLO, C. 2008. Efficient Node Discovery in Mobile Wireless Sensor Networks. In *Proceedings of DCOSS '08*. Santorini, Greece, 60–78.

EKICI, E., GU, Y., AND BOZDAG, D. 2006. Mobility-based Communication in Wireless Sensor Networks. *IEEE Communications Magazine 44,* 7, 56–62.

GORLICK, A. 2007. Turtles to Test Wireless Network.

HUI, J. W. AND CULLER, D. E. 2008. IP is Dead, Long Live IP for Wireless Sensor Networks. In *Proceedings of SenSys '08*. ACM, New York, NY, USA, 15–28.

JIANG, X., TANEJA, J., ORTIZ, J., TAVAKOLI, A., DUTTA, P., JEONG, J., CULLER, D., LEVIS, P., AND SHENKER, S. 2007. An Architecture for Energy Management in Wireless Sensor Networks. *SIGBED Review 4,* 3, 31–36.

KAELBLING, L. P., LITTMAN, M. L., AND MOORE, A. P. 1996. Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research 4*, 237–285.

KAYS, R. W. AND WIKELSKI, M. 2007. Automated radio telemetry system initiative. `http://www.princeton.edu/~wikelski/research/index.htm` (Accessed 21/03/2011).

KENWARD, R. E. 2001. *A Manual for Wildlife Radio Tagging (Biological Techniques)* 2 Ed. Academic Press.

MACDONALD, D. W. AND NEWMAN, C. 2002. Population Dynamics of Badgers (*Meles meles*) in Oxfordshire, U.K.: Numbers, Density and Cohort Life Histories, and a Possible Role of Climate Change in Population Growth. *Journal of Zoology 256,* 01, 121–138.

MACDONALD, D. W., NEWMAN, C., BUESCHING, C. D., AND JOHNSON, P. J. 2008. Male-biased Movement in a High-density Population of the Eurasion Badger (*Meles meles*). *Journal of Mammalogy*, 1077–1086.

MACDONALD, D. W., RIORDAN, P., AND MATHEWS, F. 2006. "Biological Hurdles to the Control of TB in Cattle: A Test of Two Hypotheses Concerning Wildlife to Explain the Failure of Control". *Biological Conservation 131,* 2, 268 – 286. Infectious Disease and Mammalian Conservation.

MAINLAND, G., PARKES, D. C., AND WELSH, M. 2005. Decentralized, Adaptive Resource Allocation for Sensor Networks. In *Proceedings of NSDI '05*. USENIX Association, Berkeley, CA, USA, 315–328.

MAINWARING, A., CULLER, D., POLASTRE, J., SZEWCZYK, R., AND ANDERSON, J. 2002. Wireless Sensor Networks for Habitat Monitoring. In *Proceedings of WSNA '02*. ACM, New York, NY, USA, 88–97.

MARTINEZ, K., HART, J. K., AND ONG, R. 2009. Deploying a Wireless Sensor Network in Iceland. In *Proceedings of GSN '09*. Springer-Verlag, Berlin, Heidelberg, 131–137.

NAUMOWICZ, T., FREEMAN, R., HEIL, A., CALSYN, M., HELLMICH, E., BRÄNDLE, A., GUILFORD, T., AND SCHILLER, J. 2008. Autonomous Monitoring of Vulnerable Habitats using a Wireless Sensor Network. In *Proceedings of the workshop on Real-world wireless sensor networks*. REALWSN '08. ACM, New York, NY, USA, 51–55.

NEAL, E. AND CHEESEMAN, C. 1996. *Badgers*. Poyser Books.

PERKINS, C. E. AND BHAGWAT, P. 1994. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. *SIGCOMM Comput. Commun. Rev. 24*, 234–244.

RUTISHAUSER, M., PETKOV, V., OBRACZKA, J. B. K., MANTEY, P., WILLIAMS, T. M., AND WILMERS, C. C. CARNIVORE: A Disruption-Tolerant System for Studying Wildlife. *EURASIP Journal on Wireless Communications and Networking 2011*.

SADLER, C. AND MARTONOSI, M. 2006. Data Compression Algorithms for Energy-constrained Devices in Delay Tolerant Networks. In *ACM Conference on Embedded Network Sensor Systems (SenSys)*.

SELAVO, L., WOOD, A., CAO, Q., SOOKOOR, T., LIU, H., SRINIVASAN, A., WU, Y., KANG, W., STANKOVIC, J., YOUNG, D., AND PORTER, J. 2007. LUSTER: Wireless Sensor Network for Environmental Research. In *SenSys'07: Proceedings of the 5th international conference on Embedded networked sensor systems*. ACM, New York, NY, USA, 103–116.

SIKKA, P., CORKE, P., VALENCIA, P., CROSSMAN, C., SWAIN, D., AND BISHOP-HURLEY, G. 2006. Wireless Adhoc Sensor and Actuator Networks on the Farm. In *Proceedings of IPSN '06*. 492–499.

SZEWCZYK, R., POLASTRE, J., MAINWARING, A., AND CULLER, D. 2004. Lessons From A Sensor Network Expedition. In *Proceedings of EWSN '04*. 307–322.

TOLLE, G., POLASTRE, J., SZEWCZYK, R., CULLER, D., TURNER, N., TU, K., BURGESS, S., DAWSON, T., BUONADONNA, P., GAY, D., AND HONG, W. 2005. A macroscope in the redwoods. In *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*. ACM, New York, NY, USA, 51–63.

WANG, Q., ZHU, Y., AND CHENG, L. 2006. Reprogramming wireless sensor networks: challenges and approaches. *IEEE Network 20,* 3, 48–55.

WERNER-ALLEN, G., DAWSON-HAGGERTY, S., AND WELSH, M. 2008. Lance: Optimizing High-resolution Signal Collection in Wireless Sensor Networks. In *Proceedings of SenSys '08*. 169–182.

ZHANG, P., SADLER, C. M., LYON, S. A., AND MARTONOSI, M. 2004. Hardware Design Experiences in ZebraNet. In *Proceedings of SenSys '04*. ACM, New York, NY, USA, 227–238.

Received February 2007; revised March 2009; accepted June 2009