UNIVERSITY OF SOUTHAMPTON

FACULTY OF ENGINEERING, SCIENCE AND MATHEMATICS

School of Electronics and Computer Science

Energy- and Information-Managed Wireless Sensor Networks: Modelling and Simulation

by

Geoff V. Merrett

Thesis for the degree of Doctor of Philosophy

October 2008

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING, SCIENCE AND MATHEMATICS
SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

ENERGY- AND INFORMATION-MANAGED WIRELESS SENSOR NETWORKS: MODELLING AND SIMULATION

by Geoff V. Merrett

Wireless Sensor Networks (WSNs) allow the remote and distributed monitoring of parameters in their deployed environment. WSNs are receiving increasing research interest, due to their ability to enable a wide range of applications, and their potential to have a major impact on ubiquitous computing. Many research challenges are encountered in retaining a useful network lifetime under constrains imposed by the limited energy reserves that are inherent in the small, locally-powered sensor nodes. This research addresses some of these challenges through the development and evaluation of energy- and information-managed algorithms leading to increased network lifetime.

The first contribution of this research is the development of an Information manageD Energy-aware ALgorithm for Sensor networks with Rule Managed Reporting (IDEALS/RMR). IDEALS/RMR is an application-independent, localised system to control and manage the degradation of a network through the positive discrimination of packets. This is achieved by the novel combination of energy management (through IDEALS) and information management (through RMR) which increases the network lifetime at the possible expense of often trivial data. IDEALS/RMR is particularly suited to applications where sensor nodes are small, energy constrained, embedded devices (particularly those that feature energy harvesting) that are required to report data in an unassisted fashion.

The second contribution of this research is the analysis of various environmental and physical aspects of WSNs, and the effect that they have on the operation of nodes and networks. These aspects include energy components (stores, sources and consumers), sensing devices, wireless communication, and timing; these aspects are independently modelled and, through simulation, their effect on the operation of the network is quantified.

The third contribution of this research is the evaluation of IDEALS/RMR using a simulator that has been specifically developed to integrate both the proposed environmental and physical models, and a novel node architecture that facilitates structured software design. A scenario depicting the use of a WSN to monitor pump temperature in a water pumping station is simulated, and highlights the benefits of the developed algorithms.

Contents

List of Figures

List of Tables

Declaration of Authorship

I, Geoff V. Merrett declare that the thesis entitled 'Energy- and Information- Managed Wireless Sensor Networks: Modelling and Simulation' and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- parts of this work have been published as listed in section 1.5 of this thesis.

Signed: G. V. Merrett

Date: 07 Jan 2009

Acknowledgments

I would like to express my sincere appreciation and gratitude to Bashir Al-Hashimi, Neil White, and Nick Harris for their supervision, guidance and support throughout my Ph.D. I would also like to thank Alex Weddell for regular discussions and his contributions towards collaborative work.

I wish to thank the Engineering and Physical Sciences Research Council (EPSRC) for supporting me by means of a doctoral training award and under grant number EP/D042917/1 Platform: 'New Directions for Intelligent Sensors.' I am also grateful to the Royal Academy of Engineering (RAE) for providing travel support in order for me to publish and present my research to an international audience. I am also thankful for the research facilities and support provided by the School of Electronics and Computer Science (ECS) throughout my time at Southampton. I wish to extend particular thanks to both ECS and the University of Southampton for the exceptional level of support that was provided following the devastating fire that destroyed ECS offices, facilities and research in October 2005.

In addition to the people mentioned above, I also wish to thank the people of the Electronic Systems and Devices (ESD) group, including Andrew, Russel and Cheryl, who have made it such an interesting place to work over the past four years. Finally, I wish to thank my family and friends for their support. This thesis is dedicated to my parents, without whom none of this would have been possible.

Nomenclature

α	Growth Ratio
a	Clock Aging Rate
$\mathrm{ADC_{out}}$	ADC Output
β_{rr}	Recovery Rate
β_{ce}	Charging Efficiency
B	Communication Bandwidth [Hz]
χ	EP/PP Number
c	Speed of Light in a Vacuum $= 3.00 \times 10^{8}\ \mathrm{m/s}$
C	Capacitance [F]
$\mathrm{C_n}$	Rated Capacity [Ah]
$\mathrm{C_{res}}$	Remaining Capacity [Ah]
$\mathrm{C'_{res}}$	Previous Remaining Capacity [Ah]
δ	Separation Amount for Hysteresis in EP Allocation
d	Separation Distance [m]
$\mathrm{d_{max}}$	Maximum Separation Distance Considered for Communication [m]
$\mathrm{d_0}$	Far-Field Reference Distance [m]
$\mathrm{d_f}$	Fraunhofer Distance [m]
D	Largest Linear Dimension of Transmitter Antenna [m]
DC	Duty Cycle
ε	Uniform Random Variable Bounded by $\varepsilon_{\mathrm{min}} \leq \varepsilon \leq \varepsilon_{\mathrm{max}}$
ϵ	Transmit Amplifier Drain Efficiency
E	Energy [J]
$\mathrm{E_b}$	Energy per Bit [J]
$\mathrm{E_{add}}$	Energy Added to Energy Store [J]
$\mathrm{E_{amp}}$	Transmit Amplifier Energy Consumption [J]
$\mathrm{E_{cons}}$	Energy Consumed by Energy Consumer [J]
$\mathrm{E_{elec}}$	Transceiver Circuitry Energy Consumption [J]

$\mathrm{E_{harv}}$	Energy Harvested from Energy Store [J]
$\mathrm{E_{leak}}$	Energy Leakage by Store [J]
$\mathrm{E_{max}}$	Maximum Store Capacity [J]
$\mathrm{E_{rate}}$	Energy Lost or Gained through Rate Capacity Effect [J]
$\mathrm{E_{res}}$	Remaining Energy [J]
$\mathrm{E'_{res}}$	Previous Remaining Energy [J]
$\mathrm{E_{relx}}$	Energy Lost or Gained through Relaxation [J]
$\mathrm{E_{rx}, E_{tx}}$	Receive & Transmit Energy Consumption [J]
$\mathrm{E_{use}}$	Energy Removed from Energy Store [J]
$\mathrm{E_v}$	Illuminance [lx]
$\mathrm{E_v^*}$	Reference Illuminance [lx]
f	Frequency [Hz]
$\mathrm{f_{nom}}$	Nominal Frequency [Hz]
$\Delta\,\mathrm{f_0}$	Initial Frequency Error [Hz]
$\Delta\,\mathrm{f_e}$	Environmental Frequency Variance [Hz]
$\Delta\,\mathrm{f_n}$	Short Term Frequency Instability [Hz]
$\mathrm{F_{scale}}$	Photovoltaic Illuminance Power Scaling Factor
γ	Rate Discharge Factor
$'\mathrm{g}'$	Acceleration due to Gravity $= 9.81$ m $\mathrm{s^{-2}}$
G	Receiver System Gain
$\mathrm{G_R}\,,\ \mathrm{G_T}$	Receiver and Transmitter Antenna Gains [dBi]
$\mathrm{G_r}\,,\ \mathrm{G_t}$	Receiver and Transmitter Antenna Gains
η	Path Loss Exponent
h	Planck's Constant $= 6.63 \times 10^{-34}$ Js
H	Hop Count
$\mathrm{H_T}$	Hard Threshold
I	Current [A]
$\Delta\,\mathrm{I_{frac}}$	Fractional Current Offset (Due to Temperature Dependence)
$\mathrm{I_n}$	Noise Current [A]
$\mathrm{I_P}$	Operating Current [A]
$\mathrm{I_{SC}}$	Short Circuit Current [A]
$\mathrm{I'_{SC}}$	Normalised Short Circuit Current
$\mathrm{I^*_{SC}}$	Short Circuit Current at an Illuminance $\mathrm{E_v^*}$ [A]
$\mathrm{I_{SC-min}}$	Minimum Short Circuit Current at 100fc [A]
$\mathrm{I_{SC-typ}}$	Typical Short Circuit Current at 100fc [A]

$\mathrm{I}_{\mathrm{TC-max}}$	Maximum Short Circuit Temperature Coefficient [%/K]
$\mathrm{I}_{\mathrm{TC-typ}}$	Typical Short Circuit Temperature Coefficient [%/K]
I_{tx}	Transmit Current Consumption [A]
J , J_n	Total Interference Power, Interference Power from Interferer n [W]
k	Boltzmann's Constant = 1.38×10^{-23} J/K
λ	Wavelength [m]
l	Data Length/Number of bits [bits]
l_{enob}	Effective Number of bits [bits]
L	System Loss Factor
M	'C-Rate' [h^{-1}]
υ_{n}	Thermal (Johnson) Noise [V]
nf	Noise Figure
NF	Noise Figure [dB]
N	Noise Power [W]
N_{in}, $\mathrm{N}_{\mathrm{out}}$	Noise Power at Input and Output Stages of the Receiver Circuitry [W]
$\mathrm{N}_{\mathrm{int}}$	Internal Noise in Receiver Circuitry [W]
N_0	Noise Power Spectral Density [W/Hz]
NEP	Noise Equivalent Power [W/√Hz]
ϕ_{χ}	Lower Relative Threshold of EP χ
φ_{χ}	Lower Absolute Threshold of EPχ
Φ	Number of EPs/PPs (i.e. max(χ))
P	Power [W]
$\mathrm{P}_{\mathrm{active}}$, P_{sleep}	Active, Sleep Power Consumptions [W]
$\mathrm{P}_{\mathrm{ADC}}$	ADC Power Consumption [W]
$\mathrm{P}_{\mathrm{harv}}$	Instantaneous Power Harvested from Source [W]
$\mathrm{P}_{\mathrm{leak}}$	Instantaneous Leakage Power [W]
$\mathrm{P}_{\mathrm{max}}$	Maximum Output Power [W]
$\mathrm{P}_{\mathrm{OpAmp}}$	OpAmp Power Consumption [W]
P_{r0} , P_{t0}	Receive Circuitry, Transmit Circuitry Power Consumption [W]
P_{rx} , P_{tx}	Receive, Transmit Power Consumption [W]
P_{R} , P_{r}	Received Power [dB , W]
P_{T} , P_{t}	Radiated Power [dB , W]
PL , $\overline{\mathrm{PL}}$	Path Loss, Mean Path Loss
ρ	Sensor Source Diffusion

r	Communication Bit Rate [bps]
$\Delta\,\mathrm{R}$	Resistor Tolerance
$\mathrm{R_F}$	Feedback Resistance [Ω]
$\mathrm{R_L}$	Load Resistance [Ω]
$\mathrm{R_{on}}$, $\mathrm{R_{off}}$	On , Off Resistance of Switch [Ω]
s	Actual Environmental Value to be Sensed
$\mathrm{s_d}$	Sampled Value
$\mathrm{S_{in}}$, $\mathrm{S_{out}}$	Signal Power at Input and Output Stages of the Receiver Circuitry [W]
$\mathrm{SNR_{in}}$, $\mathrm{SNR_{out}}$	SNR at Input, SNR at Output Stage of the Receiver Circuitry
$\mathrm{S_{R-peak}}$	Peak Radiometric Sensitivity [A/W]
$\mathrm{S_T}$	Soft Threshold
τ	Oscillator Timing Error (expressed in the form ‘1 in τ’)
$\Delta\,\mathrm{t}$	Time Period [s]
t	Time [s]
$\mathrm{t_0}$	Start Time [s]
$\Delta\,\mathrm{t_c}$	A Predefined Time Period [s]
$\Delta\,\mathrm{t_{conv}}$	ADC Conversion Time [s]
$\Delta\,\mathrm{t_d}$	Discharging Time Period [s]
$\Delta\,\mathrm{t_{exp}}$	Expected Time Period [s]
$\Delta\,\mathrm{t_{frac}}$	Fractional Time Offset (Due to Temperature Dependance)
$\Delta\,\mathrm{t_p}$	Preamble Time Period [s]
T	Temperature [K]
$\Delta\,\mathrm{T_{max}}$	Maximum Temperature Error [K]
V	Voltage [V]
$\mathrm{V_{mpp}}$	Maximum Power Point Voltage [V]
$\mathrm{V_{nom}}$	Nominal Store Voltage [V]
$\mathrm{V_{OC}}$	Open Circuit Voltage [V]
$\mathrm{V_P}$	Operating Voltage [V]
$\mathrm{V_{ref}}$	ADC Reference Voltage [V]
$\mathrm{V_s}$	Instantaneous Store Voltage [V]
X_{σ}	Normally Distributed Random Variable with Variance σ^2

Abbreviations

ADC	Analogue to Digital Converter
APP	Application Layer
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
CCA	Clear Channel Assessment
CDMA	Code Division Multiple Access
CSMA	Carrier Sense Multiple Access
CSMA-CA	Carrier Sense Multiple Access with Collision Avoidance
CTS	Clear To Send
DFS	Dynamic Frequency Scaling
DLL	Data Link Layer
DOD	Depth of Discharge
DVS	Dynamic Voltage Scaling
EAN	Energy Analysis Layer
ECO	Energy Control Layer
ENOB	Effective Number of Bits
EP	Energy Priority
EPSN	Energy Priority Sub-Network
FDMA	Frequency Division Multiple Access
FFT	Fast Fourier Transform
GUI	Graphical User Interface
IDEALS	Information Managed Energy Aware Algorithm for Sensor Networks
IEEE	Institute of Electrical and Electronics Engineers
ITU	International Telecommunication Union
Li-ion	Lithium Ion
LOS	Line Of Sight
LPL	Low Power Listening
MAC	Medium Access Control Layer

MCR	‘Minimum Cost’ Routing
MCU	Microcontroller Unit
MPP	Maximum Power Point
MPPT	Maximum Power Point Tracking
NDF	Network Description File
NET	Network Layer
Ni-Cad	Nickel Cadmium
Ni-MH	Nickel-Metal Hydride
OSI-BRM	Open Systems Interconnection – Basic Reference Model
PER	Packet Error Rate
PHY	Physical Layer
PP	Packet Priority
PYE	Physical Energy Layer
PYS	Physical Sensing Layer
RF	Radio Frequency
RFID	Radio Frequency Identification
RMR	Rule Managed Reporting
RMS	Root Mean Square
RTC	Real Time Clock
RTS	Request to Send
SAP	Service Access Point
SEV	Sensor Evaluation Layer
SINR	Signal to Interference Noise Ratio
SNR	Signal to Noise Ratio
SoC	System on Chip
SPR	Sensor Processing Layer
TDMA	Time Division Multiple Access
WSN	Wireless Sensor Network
WSNsim	Wireless Sensor Network Simulator

Chapter 1

Introduction

> *"The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it."*
>
> Mark Weiser, 1991

Mark Weiser's vision for ubiquitous computing envisaged a world instrumented with pervasive computing, that seamlessly and gracefully interacted with human users [1, 2]. Idealistically, ubiquitous computing is a synonym for the evolution of electronic devices that has been historically witnessed, through increased miniaturisation and information/human-centric operation. It is anticipated that by 2015, ubiquitous computing will provide an infrastructure for groundbreaking advances into the information age, with instrumented intelligence built into the fabric of cities and buildings [3]. Ubiquitous computing encompasses a wide range of technologies and applications, including mobile devices (such as mobile phones, mobile media players), smart appliances (such as web-enabled microwaves, fridge-freezers, and ovens), networked home entertainment (such as media centres), radio frequency identification (RFID), and wireless sensor networks (WSNs). Especially relevant to the latter two, and fundamental to the successful implementation of ubiquitous computing, is the development of networked wireless sensing.

1.1 Introduction to Wireless Sensing

Networks of wireless sensors are attractive for both economic and application-enabling reasons. The economic benefits of wireless sensing are driven by the redundancy of cabling between sensors. The cost of installing the wiring for a single sensor is reported to average $200 (as much as 90% of the total installation cost) in an office block [4], or be as much as

$150 per meter for critical applications in hazardous industrial environments [5]. Hence, the introduction of wireless communication in sensing systems enables considerable cost savings; it is suggested that a typical industrial scenario can see a reduction of over 80% in the total system cost (of both materials and installation labour) by using commercially available WSNs [6]. Additionally, wireless sensing enables applications that were previously not achievable with wired sensors, such as the monitoring of environments under extreme conditions or the instrumentation of rotating machinery.

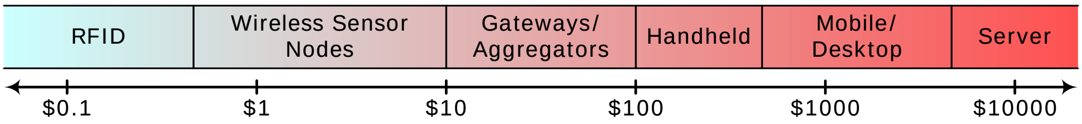


FIGURE 1-1 : Cost spectrum of various computing platforms (adapted from Culler [7]).

As shown in Fig. 1-1, the expected cost of future wireless sensor nodes are anticipated to be between 50¢ and $10 and, as has been seen for RFID as a result of mass manufacture, this is probably achievable if presented with a multi-million node market.

Definition: Wireless Sensor Network (WSN)

"A sensor network is composed of a large number of sensor nodes (small, low-cost, low-power devices with sensing, data processing, and communicating components) that are densely deployed either inside the phenomenon or very close to it." [8]

WSNs (referred to under a multitude of names including embedded networked sensors, wireless integrated network sensors, Sensor Webs, and smart dust) have transitioned from being solely a topic of academic research, to being a commercially attractive, and practically useful technology [9].

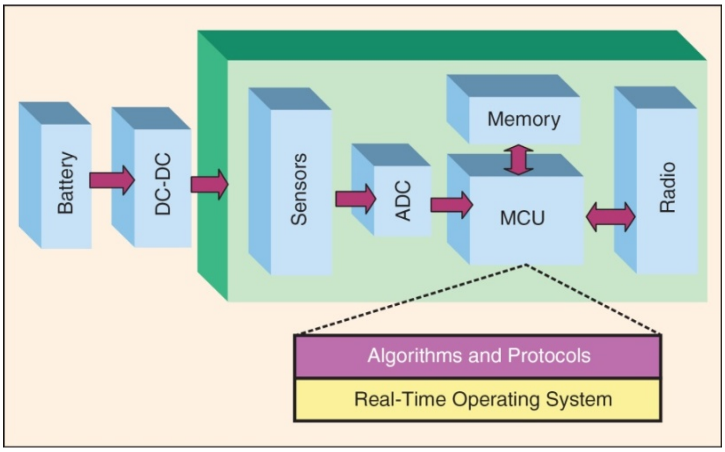


FIGURE 1-2 : Architecture of a typical node (reproduced from Raghunathan *et al.* [10]).

A WSN fundamentally consists of multiple wireless sensor nodes (also known as 'motes', and referred to hereon as 'nodes'), an architecture of which is shown in Fig. 1-2. Nodes are inherently resource constrained, and must usually operate for extended periods of time from their limited and local energy reserves [11]. Nodes sample the surrounding environment using one or more sensors, and these data are locally processed and transmitted wirelessly in accordance with a communication protocol. Multiple sensor nodes (forming a WSN) communicate packets with each other in order to perform packet routing (if a node receives a packet that is destined for a different node, it retransmits it in accordance with the communication protocol).

The range and diversity of WSN applications [12], requirements, designs and platforms are virtually limitless. Hardware platforms range from relatively large, powerful, energy-hungry devices (such as Gumstix, Personal Digital Assistants [PDAs], or PC104-based systems) to smaller, low-power devices using embedded microcontrollers (such as Crossbow motes, smart dust, or bespoke systems often based upon Texas Instruments, STMicroelectronics or ATMEL microcontrollers). While the presented algorithms and techniques may be transferable, the research in this thesis is targeted at the smaller, embedded nodes which perform autonomous operation in industrial and commercial environments. These nodes are expected to survive from local energy reserves (including energy harvesting) for around ten years, and are hence continuously operating on the edge of 'energy' operation, continuously managing their limited energy resources in order to maximise useful operation.

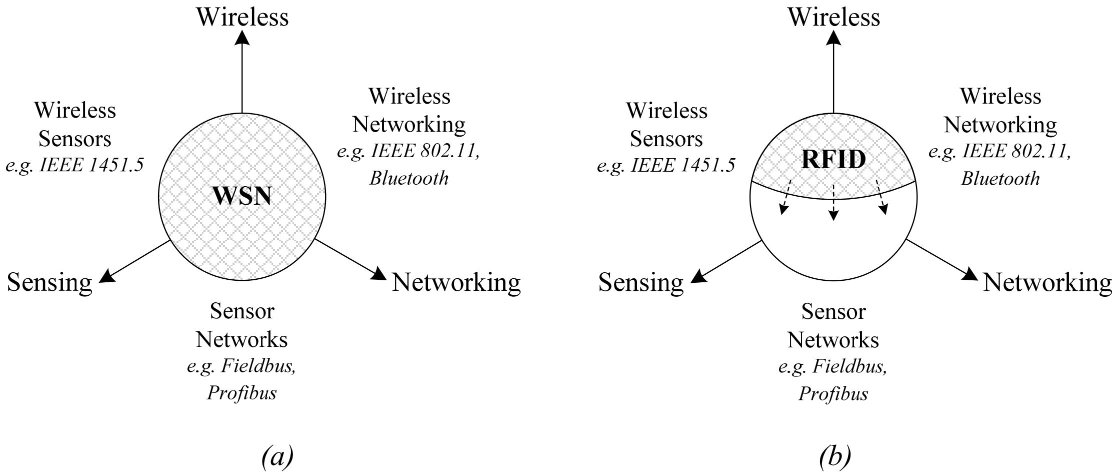
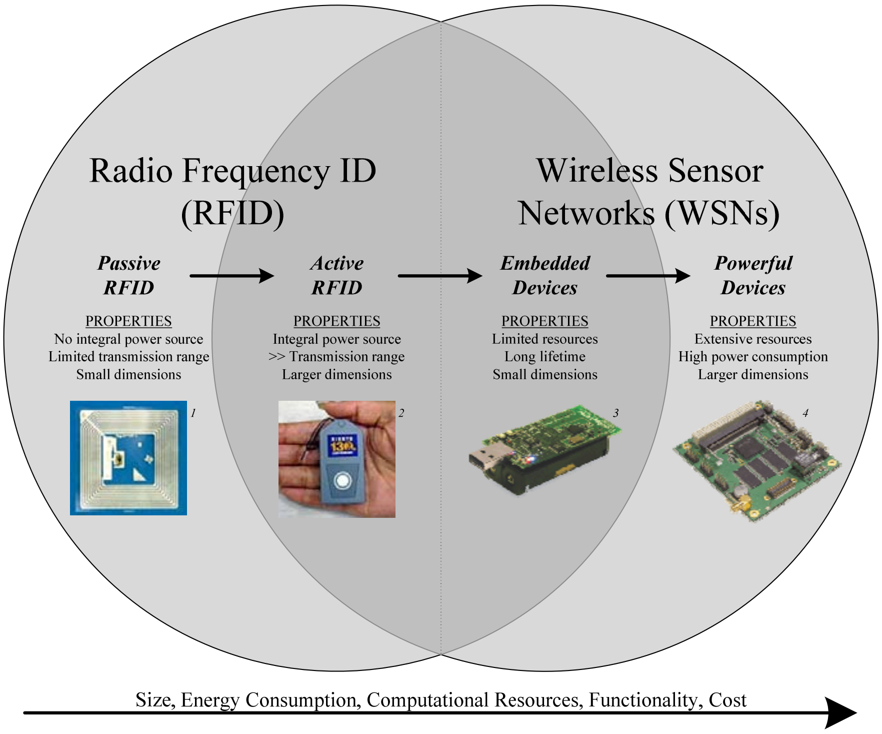


FIGURE 1-3 : The location of a) WSNs, and b) RFID within surrounding technologies.

As shown in Fig. 1-1, WSNs are positioned next to and above RFID in terms of cost. The relationship between the two technologies is not however limited to cost and, in many cases, they also share areas of functionality. RFID, which originated in the identification of aircraft in World War II, is now used in huge volumes for a wide range of applications, including the identification of retail products, medical supplies, baggage, items in supply chain, and personnel (including RFID tags that are embedded in every new UK passport). In 2002, the annual global market for RFID systems was £550m [3], and it is predicted that by 2015, 900 billion food items will be tagged, and stricter livestock legislation will require around 824 million more-sophisticated and expensive tags [13].



[1] Texas Instruments Passive RFID tag, [2] Fujitsu Active RFID tag, [3] Crossbow Telos B Mote, [4] Lippert Cool MoteMaster

FIGURE 1-4 : The overlap between active RFID and WSNs (with small, embedded sensor nodes).

While passive RFID tags (devices containing no local energy reserves, instead ‘harvesting’ energy from the host’s interrogation field) are a more distant relation of nodes, active RFID (devices containing on-board energy stores and a communication range of up to several km) shares similarities with small, embedded nodes [14]. Active RFID has enabled class four and five RFID devices [14], which has promised enhanced functionality including sensing, locationing/positioning, security and networking. Furthermore, research efforts have produced passive transponders that are able to sense [15, 16], which are classified as class two devices. The expansion of RFID from wireless communication (as a wireless identification device) into sensing and networking can be seen in Fig 1-3b. This expansion is resulting in an unclear distinction from WSNs [14, 17]. Indeed, Tully [18] states that *“These devices [small wireless sensor nodes] are indistinguishable from class 4 RFID* *tags… and any distinction is likely to disappear within 5 years"*. Fig. 1-4 shows the overlap that exists between nodes with limited resources and functionality and active RFID tags with enhanced functionality.

While RFID and WSNs may attempt to extend their boundaries, they are both are distinct technologies, defined primarily by their application areas. That is, the primary purpose of any RFID device is in identifying an object (though it may also provide information on the location and surrounding environment of that object), while a WSN is used for monitoring spatially distributed and temporally diverse environments. It is the author's belief that this categorisation should continue to provide the distinction between the two technologies as they continue to evolve.

1.2 Research Justification

WSNs promise to enable a wide variety of applications, providing a revolution in the areas of distributed, remote and wireless sensing. Nodes in a WSN are generally heavily energy, computation, and memory constrained [19]. This creates a requirement for research and development into low-computation resource-aware algorithms for WSNs, targeting small, heavily resource-constrained, embedded sensor nodes.

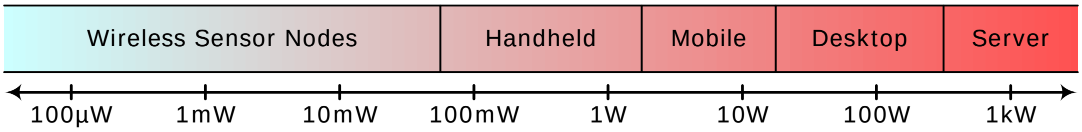


FIGURE 1-5 : Hierarchy of computing based on power consumption (adapted from Intel [20]).

Fig. 1-5 shows the acceptable power consumption of a variety of computing platforms. It can be seen that nodes are expected to consume at most 50mW, with considerably less being a particularly attractive proposition. Energy consumption is of prime importance in WSNs, and algorithms and hardware should be designed with energy-efficiency or energy-awareness as a central constraint:

> *"One of the most important constraints on sensor nodes is the low power consumption requirement… Protocols must focus primarily on power conservation. They must have inbuilt trade-off mechanisms that give the end user the option of prolonging network lifetime at the cost of lower throughput or higher transmission delay."* [19]

In order to extend the lifetime of a sensor network, energy harvesting is a particularly attractive option, theoretically offering indefinitely sustained operation. To maximise its contribution, energy harvesting should be incorporated into a node's energy management scheme. While showing significant promise, energy harvesting management for WSNs has not received research interest comparable to that of other related fields; energy harvesting was virtually unmentioned in the 2005 UK Department of Trade and Industry (DTI) Global Watch mission to the USA [13], and disregarded as a research and development priority for the Office of Communications (OFCOM) in 2007 [21].

Communication is generally regarded as the largest consumer of energy in a sensor node [19]. As such, communicating trivial or redundant data across a network constitutes a waste of resources. Extending this observation, the redundancy of data is not a digital measure, and information-management algorithms should consider the usefulness of data to the end-user:

> *"Disseminated information would have different levels of importance to the end-user… For example, the information of a potential chemical leak is more important than knowing that everything is fine…"* [22]

In the context of this thesis, information-management refers to this process of interpreting the usefulness of data, as opposed to treating all data as having homogeneous usefulness. However, it is important to balance this with maintaining an accurate representation of the network at the sink node/network controller. Allowing a node to *"adapt their overall sensing accuracy to the remaining total resources"* [23] by managing both energy and information content is a primary aim of this research.

In order to develop and evaluate algorithms and protocols, simulation is widely used for WSNs. The realism of the results obtained from such simulations depends fundamentally on the accuracy of the implemented simulation models, representing the hardware, node, and surrounding environment.

1.3 Research Aims

This research aims to investigate a method to reduce the energy consumption of a WSN (and hence increase the network lifetime) through a combination of energy-management and information-management. The target networks are those that consist of small, heavily constrained embedded devices suitable for small-scale industrial and commercial applications. While scalability (the ability of the network to operate efficiently when the number of nodes is dramatically increased) is not a primary concern of this research, the investigation of local algorithms should aid in their scalability. The developed system should be analysed and validated through simulation, giving adequate modelling to aspects that are of direct relevance. For the investigation of energy- and information-management, this includes the adequate modelling of energy components and systems, sensing devices (affecting the accuracy and value of information), communication and timing (both of which heavily affect node operation and data communication).

The themes of this research, and hence this thesis, are:

Energy-Aware Operation: A key theme of this research is considering the energy-aware operation of the nodes, and of the network as a whole. This includes taking advantage of resources such as energy harvesting which, due to the often sporadic and recurring nature of energy availability, provides energy dynamics that require energy-management at the nodes to efficiently exploit.

Information-Aware Operation: This research considers the information-aware operation of the nodes, and of the network as a whole. This involves considering the 'information' that is contained in each packet by quantifying the usefulness of data to the end-user.

Modelling and Simulation: Simulation is used to evaluate the developed system. The usefulness of obtained results depends heavily on the realism and accuracy of the simulation models implemented. As energy and information are of key importance to this research, aspects that directly influence them require adequate modelling.

1.4 Research Contributions

The major contributions of this thesis are:

IDEALS/RMR: The primary contribution of this thesis is IDEALS/RMR (Information Managed Energy Aware Algorithm for Sensor Networks with Rule Managed Reporting), a technique to extend the lifetime of a WSN through the possible sacrifice of low-information packets. RMR is the process of using a system of rules to detect events and subsequently quantify the information content of event data. In isolation this concept is not unique, but when combined with IDEALS (to balance the energy state of the node with the information content of data) presents a novel system that had not been considered elsewhere at the outset of this research. In the presence of energy harvesting, IDEALS/RMR reacts to the dynamically changing networks and environments, and inherently manages energy resources to maximise information throughput and network lifetime. IDEALS/RMR directly contributes to the *energy-aware* and *information-aware* themes of this research.

Modelling WSNs: In order to analyse IDEALS/RMR, the modelling of energy components (including the supercapacitor energy store), sensing, communications and timing has been investigated. This directly contributes to the *modelling and simulation* theme of this research; in addition, by investigating the modelling of aspects relevant to energy and information it also contributes to the *energy-aware* and *information-aware* themes.

WSNsim: While not a primary contribution of this research, WSNsim (WSN Simulator) was designed and implemented as a simulation tool to enable the unobstructed use and analysis of the developed models and algorithms in the desired environments and with the desired metrics. WSNsim contributes to the *modelling and simulation* theme of this research.

A Unified Node Architecture: Central to WSNsim is a novel node architecture which provides interfaces between simulated nodes and the simulated environment specifying (with equal importance) communication, energy and sensing. This architecture aids in the development, analysis, and use of the developed simulation models. By providing a structure for specifying the firmware of a sensor node in simulation, this directly contributes to the *modelling and simulation* theme of this research. In addition, the architecture's consideration of energy and sensing as fundamental node functions means that this also contributes to the *energy-aware* and *information-aware* themes.

1.5 Published Papers

The research in this thesis has contributed in part or full to the following publications:

- G. Merrett, B. M. Al-Hashimi, N. M. White, and N. R. Harris, "Information Managed Wireless Sensor Networks with Energy Aware Nodes," presented at *NSTI Nanotechnology Conference & Trade Show*, Anaheim, California, vol. 3, pp. 367-370, May 2005.

- G. V. Merrett, B. M. Al-Hashimi, N. M. White, and N. R. Harris, “Resource Aware Sensor Nodes in Wireless Sensor Networks,” in *Journal of Physics: Conference Series*, vol. 15, Kent, pp. 137-142, Sep. 2005.
- G. V. Merrett, N. R. Harris, B. M. Al-Hashimi, and N. M. White, “Rule Managed Reporting in Energy Controlled Wireless Sensor Networks (poster),” presented at *Eurosensors XX*, Gothenburg, Sweden, vol. 2, pp. 402-403, Sep. 2006.
- G. V. Merrett, N. R. Harris, B. M. Al-Hashimi, and N. M. White, “Energy Controlled Reporting for Industrial Monitoring Wireless Sensor Networks,” presented at *IEEE Sensors 2006*, Daegu, Korea, pp. 892-895, Oct. 2006.
- G. V. Merrett, N. R. Harris, B. M. Al-Hashimi, and N. M. White, “Energy Managed Reporting for Wireless Sensor Networks,” in *Sensors and Actuators A: Physical*, vol. 142, pp. 379-389, 2008
- A. S. Weddell, G. V. Merrett, N. R. Harris, B. M. Al-Hashimi, “Energy Harvesting and Management for Wireless Autonomous Sensors,” in *Measurement + Control*, vol. 41/4, pp. 104-108, 2008
- G. V. Merrett, A. S. Weddell, A. P. Lewis, N. R. Harris, B. M. Al-Hashimi, N. M. White, “An Empirical Energy Model for Supercapacitor Powered Wireless Sensor Nodes,” presented at *IEEE Int’l Conf. Computer Communications and Networks*, St. Thomas, US Virgin Islands, Aug. 2008.
- G. V. Merrett, A. S. Weddell, N. R Harris, B. M. Al-Hashimi, N. M. White, “A Structured Hardware/Software Architecture for Embedded Sensor Nodes,” presented at *IEEE Workshop Advanced Networking and Communications*, St. Thomas, US Virgin Islands, Aug. 2008.
- G. V. Merrett, A. S. Weddell, L. Berti, N. R. Harris, N. M. White, B. M. Al-Hashimi, “A Wireless Sensor Network for Cleanroom Monitoring,” presented at *Eurosensors XXII*, Dresden, Germany, Sep. 2008.

A selection of these are included in Appendix D.

1.6 Thesis Structure

Fig. 1-6 depicts the structure and flow of this thesis, and shows how the chapters and sections relate to the themes specified in section 1.3. In this figure, the connecting arrows show where individual sections contribute towards subsequent work.

Chapter 2 establishes and critiques the state-of-the-art in WSNs with respect to *energy-aware operation*, *information-aware operation*, and *modelling and simulation*. Chapter 3 introduces IDEALS/RMR, a novel energy and information managed algorithm developed in this research. Chapter 4 presents an investigation of models for sensor network hardware and simulation, which form the basis of WSNsim (outlined in Chapter 5). WSNsim is a simulator developed to analyse IDEALS/RMR in which the nodes' embedded software is represented using a new stack architecture representing multiple interfaces on a node. Chapter 6 presents the results obtained from the simulation of networks incorporating the developed algorithms, while Chapter 7 concludes the report, and outlines activities of future research.

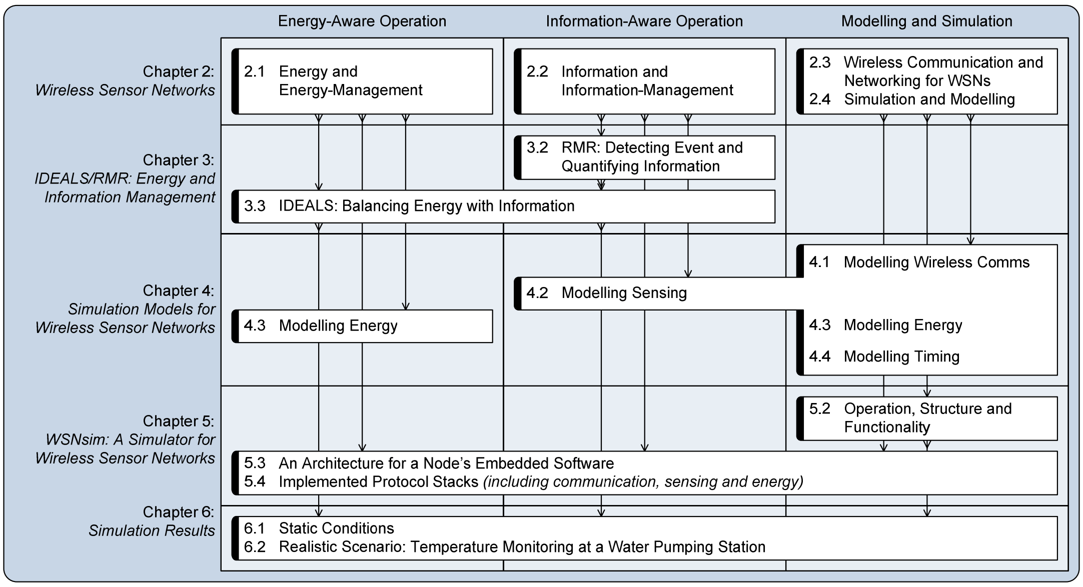

Energy-Aware Operation
Information-Aware Operation
Modelling and Simulation
Chapter 2: Wireless Sensor Networks
2.1 Energy and Energy-Management
2.2 Information and Information-Management
2.3 Wireless Communication and Networking for WSNs
2.4 Simulation and Modelling
Chapter 3: IDEALS/RMR: Energy and Information Management
3.2 RMR: Detecting Event and Quantifying Information
3.3 IDEALS: Balancing Energy with Information
Chapter 4: Simulation Models for Wireless Sensor Networks
4.1 Modelling Wireless Comms
4.2 Modelling Sensing
4.3 Modelling Energy
4.3 Modelling Energy
4.4 Modelling Timing
Chapter 5: WSNsim: A Simulator for Wireless Sensor Networks
5.2 Operation, Structure and Functionality
5.3 An Architecture for a Node's Embedded Software
5.4 Implemented Protocol Stacks (including communication, sensing and energy)
Chapter 6: Simulation Results
6.1 Static Conditions
6.2 Realistic Scenario: Temperature Monitoring at a Water Pumping Station

FIGURE 1-6 : The structure of this thesis, showing where the chapters and sections fit within the themes of energy-aware operation, information-aware operation, and modelling and simulation.

Appendix A provides an investigation into the principles of wireless communication (relevant to the section on the communications model, and the communication process implemented in WSNsim). Appendix B outlines effects found in batteries and supercapacitors (relevant to the section on energy modelling, and the energy process implemented in WSNsim). Finally, Appendix D provides a selection of papers published as a direct result of this research.

Chapter 2

Wireless Sensor Networks

WSNs are receiving considerable interest in both industry and academia due, in part, to the wide range of applications to which they are suited [12]. This chapter provides an overview of the state-of-the-art in WSNs with respect to energy and energy-management, information and information-management, communication protocols, and simulation and modelling.

2.1 Energy and Energy-Management

As presented in Chapter 1, energy is a limited (and hence valuable) resource, both in individual nodes and in the WSN as a whole. A considerable volume of research has been reported investigating a wide range of methods for reducing and controlling the energy consumption [10, 24, 25]; some of which are discussed in this section.

In duty-cycled operation, a node follows a *sleep-wakeup-sample-compute-communicate* cycle, where the majority of the cycle is spent in the low-power sleep state [26]. This process, which relies on hardware support for implementing sleep states, permits the average power consumption of a node to be reduced by many orders of magnitude. An expression for the energy reduction obtainable through duty-cycling can be derived (2.1), where DC is the duty cycle (defined as the fraction of the cycle that the node is active for [26]), and P_{sleep} [W] and P_{active} [W] are the power consumptions of the node in sleep mode and active mode respectively.

$$Energy\ Reduction = (1 - DC)\left(1 - \frac{P_{sleep}}{P_{active}}\right) \tag{2.1}$$

It can be observed from (2.1) that decreasing the DC (thus increasing the proportion of time spent sleeping) increases the reduction in energy. Similarly, an increase in the energy reduction is obtained as P_{sleep} decreases or P_{active} increases; it is paramount therefore in justifying duty-cycled operation that sleep mode power consumptions are orders of magnitudes smaller than that of the active power consumption.

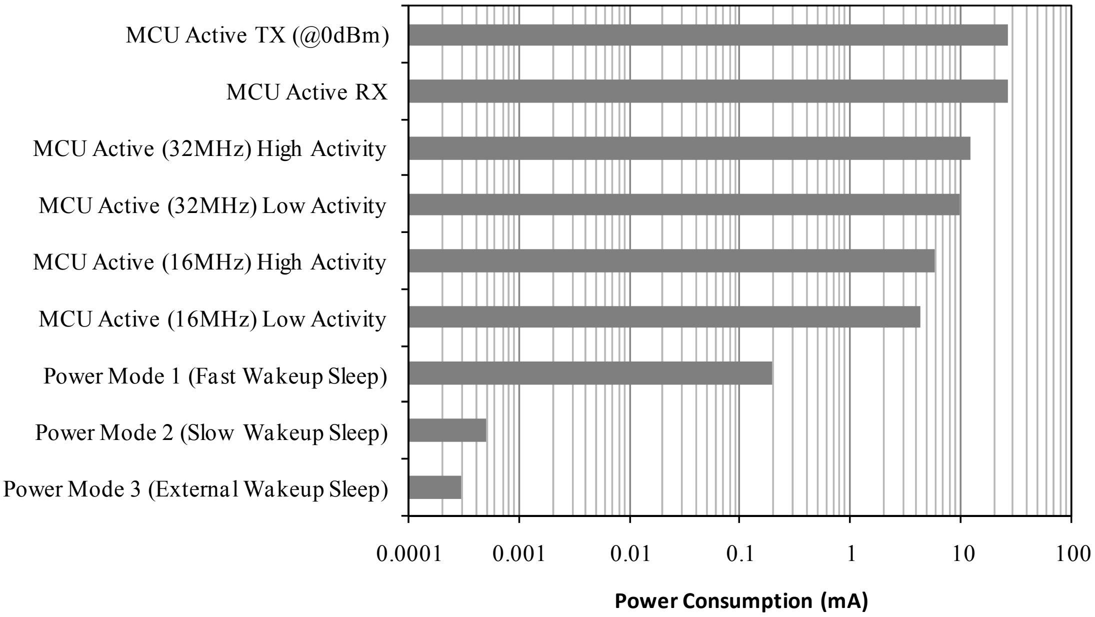


FIGURE 2-1 : The power consumption of the TI CC2430 in different power modes and modes of operation [27].

The different sleep modes offered by a microcontroller unit (MCU) are differentiated by the power consumed, the functionality enabled, the overhead required in entering the mode, and the time required to wake up from the mode [28]. The various power modes available in the Texas Instruments CC2430 System on Chip (SoC) [27] can be seen in Fig. 2-1, where the top six power consumptions represent functionality available in the active mode (power mode zero). From Fig. 2-1, it can be observed that when in power mode two or three, the device consumption is significantly reduced by over four orders of magnitude, thus accentuating the energy reduction permitted through duty-cycled operation.

Duty-cycled operation is usually possible in WSNs due to the common requirement that they do not require continuous sampling or communication (WSNs are often defined by their low data-rate and communication frequency); hence, duty-cycles of one or two percent are common [26]. It is essential that events and packets are not missed while the node is 'asleep' [28], requiring that careful thought is given to duty-cycled operation. Additionally, some applications and sensors require continuous sampling (for example accelerometers) which restricts duty-cycled operation from being easily achievable [29].

The process of duty-cycled operation can also be applied to communications, whereby an active node only receives and transmits for a small proportion of the active time. This form of duty cycling is managed by the Medium Access Control layer (MAC) of the communication protocol. Both MAC and Network layer (NET) algorithms have a significant bearing on the energy-efficient operation of a node, and these are investigated in greater detail in section 2.3.2.

Additional techniques for reducing the energy consumption of a node include Dynamic Voltage Scaling (DVS) and Dynamic Frequency Scaling (DFS), which decrease both the dynamic (switching) and static (leakage) components of a node's energy consumption [24, 25]. Both DVS and DFS trade node performance (a result of lowering the operating voltage or frequency) for energy savings, but this is acceptable in many WSNs. In some implementations, low-frequency operating modes begin to become comparable with the sleep modes (the 32kHz low-frequency operating mode in the ZebraNet project consumes only 50µA more than the sleep mode [30]), and hence permit the node to perform continuous energy monitoring and sensing at a reduced capability.

Through its ability to enable near-perpetual operation (which introduces new requirements, dynamics, and possibilities to energy-aware operation), energy harvesting is receiving increasing research interest for use in WSNs (section 2.1.1). An investigation into algorithms for energy-management and energy-aware operation for WSNs, including those designed for energy harvesting systems, is provided in section 2.1.2.

2.1.1 Energy Harvesting

As discussed in the research justification (section 1.2), the limited energy store inherent in small locally powered sensor nodes causes considerable problems for system designers. Energy harvesting (the process of scavenging ambient energy from sources in the surrounding environment) is an attractive method for overcoming many of these issues. Locally powered nodes that are sustainable through energy harvesting are useful both in applications where no wired energy infrastructure exists (for example, environmental monitoring applications), and also for the easy and economical retrofitting of sensor nodes into existing machinery. Energy harvesting also enables applications that would not otherwise be possible. An example of this is the ZebraNet project [30], which used solar harvesting nodes mounted to Zebra to monitor their behaviour. If energy harvesting had not been utilised, a battery weighing around 10kg would have been required to fulfil the project requirements, which could clearly not be mounted unobtrusively to a Zebra.

There are a wide range of different energy sources suitable for powering wireless sensor nodes [4, 31-34]. Table 2-1 shows the energy/power densities available from a range of these energy sources, and compares them with a range of energy stores by considering the energy that they could provide over a ten year period (perceived as an acceptable node lifetime, and feasible store/source lifetime [35]). Additionally, there is considerable potential for wearable applications of WSNs harvesting energy from the human body (which generates, on average, around 10.5MJ of energy per day [32]), from sources such as blood pressure, body heat, walking or breathing [33].

TABLE 2-1 : A comparison of the typical energy obtainable from a range of different energy stores and sources [4, 32, 34]. The top section of the table is for devices with a fixed amount of energy storage (energy stores), while the bottom part is for devices with a fixed amount of power generation (renewable energy sources).

Energy Source/Store	Power / Energy Density	Energy Obtained over Ten Years
Solar (Outdoors: Direct Sun)	15.0 mW/cm^2	4.73 MJ/cm^2
Solar (Outdoors: Cloudy)	0.15 mW/cm^2	47.3 kJ/cm^2
Solar (Indoors: Office Desk)	6.00 μW/cm^2	1.89 kJ/cm^2
Solar (Indoors: <60W Desk Lamp)	570 μW/cm^2	180 kJ/cm^2
Vibrations	10.0 − 250 μW/cm^3	3.15 − 78.8 kJ/cm^3
Acoustic Noise (75-100dB)	3.00 − 960 nW/cm^3	0.0946 − 303 J/cm^3
Air Flow (5% Efficient at 5m/s)	0.38 mW/cm^3	120 kJ/cm^3
Temperature (10C Differential)	15.0 μW/cm^3	4.73 kJ/cm^3
Human-Powered Systems (Shoe Inserts)	330 μW/cm^3	104 kJ/cm^3
Primary Batteries (Zinc-Air)	3.78 kJ/cm^3	
Primary Batteries (Lithium)	2.88 kJ/cm^3	
Primary Batteries (Alkaline)	1.20 kJ/cm^3	
Secondary Batteries (Li-ion)	1.08 kJ/cm^3	
Secondary Batteries (Ni-MH)	960 J/cm^3	
Secondary Batteries (Ni-Cad)	650 J/cm^3	
Micro-Fuel Cell	3.50 kJ/cm^3	
Heat Engine	3.35 kJ/cm^3	
Radioactive (^{63}Ni)	1.64 kJ/cm^3	

The local behaviour of different energy sources can be characterised by a controllable/predictable metric, as shown below [36]:

Uncontrollable but predictable: The user has no control over the availability of energy, but its behaviour is predictable. Solar energy harvested outdoors is an example of this type of energy source, due to its inherent daily cycle. In some scenarios solar energy harvested indoors can be predictable, for example when the lighting is computer controlled to come on and off between certain hours.

Uncontrollable and unpredictable: The user has no control over the availability of energy, and its behaviour is not easily predictable. Many scenarios using solar energy harvested indoors are unpredictable, for example when the building's users directly control lighting as they sporadically come and go. Additionally, the behaviour of wind energy or vibration sources are unpredictable in many scenarios.

Partially controllable: While the designer has some control over the energy availability, the actual levels are environmentally dependent. While a designer is able to install a Radio Frequency (RF) source to be harvested by nodes, the actual energy harvestable is dependent on the continuously changing propagation losses. Additionally, while energy can be harvested from vibrations on 'always-on' machinery, the magnitude and frequency of the vibrations (affecting the energy harvested) can vary considerably between machines.

Fully controllable: The user is in full control of energy availability. Examples of this type of energy source include wind-up radios and kinetic harvesting wrist watches.

Of particular interest to this thesis are vibration and solar harvesting [31, 37]. Vibration harvesting is often suited to industrial monitoring and building automation applications, as vibrating machinery is plentiful. Other sources of vibration allow a variety of application-enabling systems to be developed, for example it is estimated that a structure monitoring sensor node (with a mass of between 100 and 1000g) can be self powered from vibrations on a bridge under traffic load [38]. The S^5NAP^{TM} [39] is a vibration powered wireless sensor node that buffers harvested energy in a supercapacitor. The node transmits a temperature reading every minute, using energy harvested from a vibration of $2.0m/s^2$ RMS at a frequency of 108Hz. The VIBES wireless microsystem uses a more efficient and considerably smaller generator (less than $1cm^3$) in order to power a node that reports readings from an accelerometer. The node adapts its operation based upon the amount of energy being harvested, and has demonstrated sustained operation transmitting every 12 minutes from a vibration of $0.2ms^{-2}$ RMS at a frequency of 52Hz.

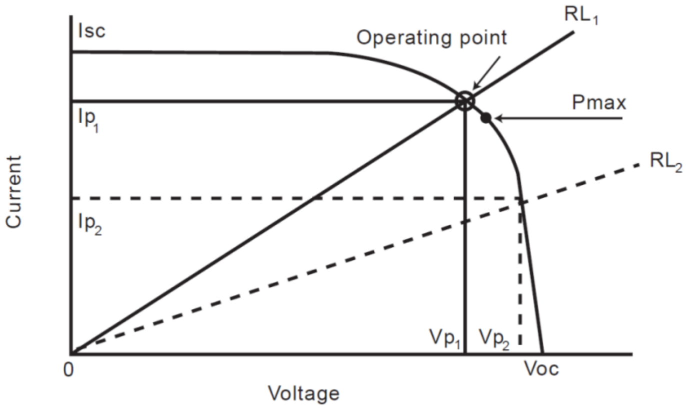


FIGURE 2-2 : The I-V output characteristics of a typical solar cell, where I_{SC} is the short-circuit current, V_{OC}[V] is the open-circuit voltage, I_p [A] is the operating current, V_p [V] is the operating voltage, R_L [Ω] is the load resistance, and P_{max} [W] is the maximum output power (reproduced from [40]).

As shown in Table 2-1, solar harvesting can provide a considerable amount of energy and, due to the often high levels of harvestable light, is well suited to use with WSNs. However, to maximise the energy harvested, it is necessary to match the impedance of the load with that of the photovoltaic. For a given light intensity, the voltage and current level that maximise the output power is called the Maximum Power Point (MPP) [29], as depicted by P_{max} in Fig. 2-2. Maximum Power Point Tracking (MPPT) is the process of monitoring the photovoltaic and dynamically adjusting the load impedance to maintain the source operating at its MPP. A variety of techniques exist for performing MPPT [29], which it is reported can increase the harvesting efficiency by one or two orders of magnitude [41]. However, it is also suggested that as V_p changes very little in some scenarios (such as the dynamic range seen in indoor solar harvesting), *near*-MPP operation (and hence *near*-optimal efficiency) is achievable by voltage clamping the photovoltaic at V_p [9]. This eradicates the need for complex (and power consuming) MPPT circuitry.

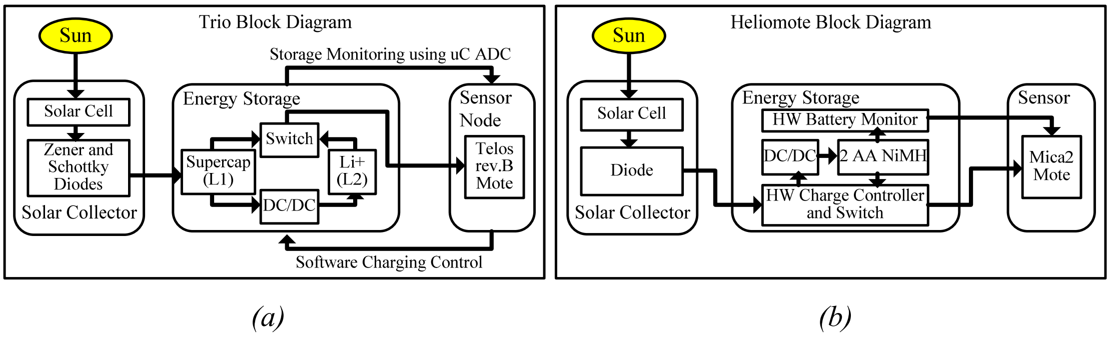


FIGURE 2-3 : Architectures for solar harvesting platforms, a) Trio [42], and b) Heliomote [43] (reproduced from [44]).

The Heliomote system [43] is a solar energy harvesting node where the photovoltaic is connected directly via a diode to a Ni-MH battery (arguing that voltage clamping performs *near*-MPP operation). Due to the direct connection to the photovoltaic, the battery experiences daily recharge cycles. Rechargeable batteries are rated for a few hundred recharge cycles and hence, while the authors disagree, it is claimed by Jiang *et al.* that such a system will have a lifetime limited to no more than two years (a limitation that is overcome by their proposed *Prometheus* subsystem) [45]. Prometheus overcomes the limitations associated with rechargeable batteries by using a supercapacitor as a buffer between the photovoltaic and battery in order to reduce the battery's activity. If the supercapacitor becomes depleted, the MCU switches to the battery to power the node. If the supercapacitor becomes full, the MCU switches to use harvested energy to recharge the battery. The Prometheus subsystem was developed into a complete system and system test-bed in Trio [42]. The energy system diagrams for Trio and Heliomote are shown in Fig. 2-3a and Fig. 2-3b respectively. The AmbiMax node is powered by both solar and wind harvesting, using a separate supercapacitor buffer for each energy source [41]. The switching between the multiple supercapacitor buffers and the single battery is performed without the need for MCU control. Furthermore, AmbiMax provides autonomous analogue MPPT independently on each energy source without MCU control. Another system that harvests energy from multiple sources is presented by Lin *et al.* [46] for wireless medical applications. In this system, the node is primarily powered by a photovoltaic but, in areas of poor light, switches to a reduced functionality mode, and is powered by RF.

While batteries have been the 'traditional' choice for powering nodes, the use of supercapacitors is gaining increasing interest, as demonstrated by the systems shown in this section. A background to the properties of both supercapacitors and rechargeable batteries can be found in Appendix B. Furthermore, the modelling of energy stores and sources is considered in section 2.4.2.2. To maximise the benefits obtained through energy harvesting, management and awareness is required in a node's embedded software; this is discussed in the next section.

2.1.2 Energy-Aware Algorithms

While the majority of energy-aware algorithms constitute routing or media access control protocols (discussed in section 2.3.2), a range of energy-aware algorithms exist for WSNs, including duty-cycle control, energy harvesting management, topology control, and data processing.

The process of transitioning to, and waking-up from, low-power sleep states has an associated latency (reported to vary between 6µs – 10ms for common node hardware [26]) which, if not managed, can result a node's energy consumption actually rising through the use of duty-cycling (for example when used for very short sleep periods). In order to manage this, research has developed algorithms to decide when a node should enter which low-power sleep mode [24, 28]. While the assumption that communication dominates a node's energy consumption is valid for many applications and implementations, this is not the case in some emerging application-driven (and hence sensor-driven) applications. For example, acoustic or seismic sensors often demand power-hungry Analogue-to-Digital Convertors (ADCs) in order to obtain the required temporal and sensing resolution [9]. This can result in the radio transceiver not being the primary energy consumer on a node. Furthermore (as shown in Fig. 2-1), the receive power consumption is often greater than that of the transmit power consumption. Therefore, conventional protocols (and many protocols designed for WSNs) that assume that the receive power is negligible are no longer efficient for WSNs [10].

In order to provide energy-aware operation, it is necessary to monitor the energy resources available to the node. A crude implementation of this is presented in Delin *et al.* [47], whereby if the node's energy store depletes past a preset threshold, the node enters a sleep state to allow the store to be charged back to a sufficient level using solar harvesting. In a similar manner, Cianci *et al.* [48] considers controlling a node's operation based upon whether its residual energy level is above or below a threshold. The result of this controls the node's sleeping behaviour, and hence affects coverage – nodes that have residual energy levels below the threshold will sleep more often.

Kansal *et al.* [49] present a framework for *uncontrolled but predictable* energy sources, which adaptively learns the energy properties of the environment at each node, and predicts future energy availability; for example predicting the daily and seasonal periodicity of solar energy harvesting. The complexity (and hence accuracy) of the learning/modelling functionality can range from a simple exponentially weighted moving-average [36], to spectral analysis (such as a fast Fourier transform) and adaptive filtering [49]. The predictive information is made available for use in energy-aware algorithms such as duty-cycle adjustment (having an increased duty-cycle during the day while the node is harvesting solar energy) [9]. Additionally, the predicted future energy availability can be used to influence packet routing decisions, discussed in more detail in section 2.3.2.2. Furthermore, an extensive 'harvesting theory' is proposed which characterises energy harvesters in order to enable an engineer to select an appropriate energy store and level of performance to provide energy-neutral (sustained) operation [36].

Duty-cycling is not the only use for low-power sleep modes, and algorithms (such as ASCENT [50] and SPAN [51]) extend the network lifetime through exploiting redundancy in dense networks [52]. Such algorithms, referred to as topology control algorithms, use a distributed process to select a number of nodes that are currently redundant (as their sensor or communication coverage is duplicated by other nodes) which are put to sleep [10]. The subset of active nodes will change over time so that the overall energy consumption is spread across all sensor nodes.

Energy efficiency can be often traded with the accuracy of complex mathematical calculations – for example an Finite Impulse Response (FIR) filter or Fast Fourier Transform (FFT) – by careful structuring the operations [24]. A common method of implementing this is to use iterative algorithms which provide more accurate results with increased operational time. Therefore, by accepting a less accurate result, the operational time (and hence energy consumption) is reduced. Also, data aggregation (reducing packet transmissions by summarising data from multiple nodes) and fusion (reducing packet transmissions by processing and interpreting data from multiple nodes) techniques can be incorporated [24].

As communication is often the major energy consumer in a WSN, reducing the number of packets communicated results in a significant energy reduction. Algorithms and techniques for energy-aware information extraction are discussed in section 2.2.

2.1.3 Discussion

This section provided an overview of reported research relevant to the energy-management theme of this thesis. As stipulated in the *research justification*, energy-efficiency is of paramount importance to WSNs. As such, low-power techniques should be considered at all stages of the design process, and the possibilities of energy sources (such as vibration and solar harvesting) should be given careful consideration. Such energy sources must however be adequately managed in order to maximise their efficiency and benefits.

Communication has been highlighted as often being a considerable consumer of energy; hence, reducing the number of communicated packets can produce a significant energy reduction. Additionally, this section (and the background to battery and supercapacitor effects presented in Appendix B) has shown that complex interplays and properties exist in components of the energy system.

2.2 Information and Information-Management

While the development of WSNs are dramatically increasing the amount of data that we are able to collect, *"they do not, in themselves, do anything to produce 'information' out of this data"* [53]. Data analysts claim that industrial plants will see a noticeable increase in efficiency once it is figured out how to get the actual information out of data.

As defined in Chapter 1, information-management is the process of interpreting, and adapting the behaviour of nodes as a result of, the usefulness of data. By simply considering all data to be of equal usefulness to the end-user, resources such as energy and bandwidth can be wasted in supporting the transfer of trivial data containing very little actual information. This section provides an overview of research on information-management for WSNs, particularly that which relates to information dissemination.

2.2.1 Information Dissemination

After a sensor node has sampled data from its environment, a strategy is required to dictate how the data is made available to other nodes, a central sink node, or the end-user. This is handled by an information dissemination algorithm (also known in this context as a reporting technique). While information dissemination algorithms are sometimes concerned with the process of packet routing, this is outside the scope of this context, and investigated separately in section 2.3.2.2. Four established techniques for information dissemination are [54, 55]:

> **Continuous/Periodic Dissemination (data 'pushing'):** The sensor node continuously reports data following a periodic schedule; for example a packet is transmitted every *t* minutes containing: *"the temperature at location x is y."* In this way, packets are proactively pushed from the network. Continuous reporting means that packets are transmitted (and an energy cost incurred) even when the sensed parameter has not significantly changed, hence containing little useful information, since the previous transmission.
>
> **Query-Driven Dissemination (data 'pulling'):** The user or application instigates data transfer by querying data from the network [56]; for example a query is propagated into the network requesting: *"what is the temperate in zone 3,"* or *"where in the building is the temperature over 25 degrees?"* Qualifying nodes reply to these

queries with packets similar to: *"the temperature at location x is y."* In query-driven networks, data is pulled from a passive network in response to queries.

Event-Driven Dissemination (data 'pushing'): The 'intelligent' sensor node decides for itself when data should be reported to a sink node; for example, a packet is transmitted containing: *"the temperate is too hot at location x,"* or *"the temperature has unexpectedly increased by ten degrees."* Event-driven networks reactively push data from the network. By locally determining when events occur that are worth reporting, redundant data transmissions can be minimised (using the principle that: "it is not news if one can predict it" [57]).

Hybrid Dissemination: Hybrid dissemination algorithms are an amalgamation of the above techniques. Examples of hybrid dissemination include event-driven techniques proposed by Liu *et al.* [58] (which mixes continuous and query-driven approaches).

In continuous reporting, the choice of period duration has a considerable effect on network performance. If a short period is chosen, a large proportion of the packets are likely to be redundant (containing little useful information), providing little added value to the user while still consuming energy [59]. If a long period is chosen, the network is likely to suffer from latency issues and the missing of events. While the missing of events can be avoided by locally aggregating the average-max-min sensed values (when using a sampling rate that is greater than the dissemination rate), this does not avoid the issues with latency or the 'information smoothing' that aggregation introduces. Hence, while continuous dissemination is suited to applications with random or uncharacterisable signals, in most cases it does not maximise energy consumption or information throughput. Query-driven and event-driven dissemination approaches provide more suitable techniques, and are discussed in the following sections.

2.2.1.1 Query-Driven Dissemination

Query-driven approaches to information dissemination in WSNs are appealing to many system designers and applications, allowing ease of integration due to established database services and methods. Furthermore, considering the network as a distributed database is an attractive research proposition, in line with the vision that *"the network is the sensor"*. TinyDB [60] is an acquisitional query processing system for sensor networks (data is obtained from a network in response to an injected query), designed primarily for use with TinyOS. The system allows a sink node to extract relevant data from the network of sensors by interrogating it with a wide range of SQL-like queries. TinyDB supports a diverse range of queries including in-network aggregation (such as *"what is the average temperature in location x?"*), temporal sampling (such as *"what is the humidity of sensor y every minute for the next hour?"*), event detection (such as *"what locations does the temperature rise above 40ºC over the next ten minutes?,"* and lifetime-based (such as *"report temperature as regularly as is possible in achieving a lifetime of one year"*). Power management exists by optimising and scheduling queries and adjusting reporting rates to minimise consumption.

The Cougar [61] approach also uses query-based dissemination, and considers the sensor network as an abstract distributed database (referred to as a device database system). Cougar supports historic queries (such as *"what was the average rainfall in 1999?"*) which, similarly to the processing of other types of query, are responded to using data stored in-network [56]. While it can be attractive to consider a WSN as having the ability to store historic data it raises a number of critical obstacles. Firstly, in order to provide a versatile and adequate response to a range of possible historic queries, each node is required to have the storage capacity for a considerable amount of data. Secondly, each node in the network is no longer a homogeneous and 'expendable' entity, as the loss of each node now loses a complete set of historic data. The alternative, data warehousing (where data is stored off-network), allows the network to operate as a collection of 'hot-swappable' and fault tolerant nodes, and allows for data integrity provision (through traditional data backup methods).

Query-driven dissemination is suited to applications where the monitoring requirements change often (for example where, on one day, the user is interested in only the temperature in location *x* while, the next day, the user is interested in only the locations at which the humidity level is above *y*), or where users infrequently and irregularly draw information from the network (for example where users wish to find the location of node *z* only when they need to know). Hence, in situations where a clearly defined monitoring application exists (therefore the users' monitoring requirements do not regularly change) and where the network is not 'user-drawn' (that is, instead of the user drawing data from the network, it is proactively provided to them), a more autonomous dissemination algorithm can be more effective in providing the required service while not investing resources in overheads associated with query handling. Such overheads can be seen in TinyDB, which is designed to provide a general solution for a wide range of scenarios. As such, the implementation overheads are reasonably high; for example the code requires 58K of the 128K memory available on a current generation mote.

2.2.1.2 Event-Driven Dissemination

Event detection systems can be crudely classified into two categories: those that report only the 'digital' occurrence of an event (such as smoke and motion detectors), and those that detect the occurrence and magnitude of an event (such as a seismograph which reports the magnitude of any vibrations over a certain threshold) [29]. This form of event-detection can be used in systems to manage the operation of energy-hungry sensors, where low-power (but less accurate) sensors detect the occurrence of an event and subsequently sample power-hungry (or lifetime- and sample-limited) sensors [26]. As previously discussed, event-driven dissemination disseminates data when the node decides that an 'event' has occurred. Event-driven algorithms require a form of intelligence inside the network, however simple, to ascertain when events occur. While a form of event detection is often implemented as part of a query-driven system (such as TinyDB), such systems provide functionality not required for many applications at the expense of considerable overheads.

Rule-based approaches to event detection generate events whenever specific criteria are fulfilled, or features detected, (the details of which are specified in a predefined application-specific rule set) in the sensed environment. These rules can share similarities with those used in query-based event-detection, such as periodic and threshold detection [60]. Manjeshwar *et al.* [62] proposed TEEN, an event-based dissemination technique for WSNs, and is based upon the concept of two thresholds: the Hard Threshold H_T, and the Soft Threshold S_T. Data is sent from the sensor node if the sampled value either exceeds H_T, or changes by more than S_T. A smaller value of S_T maintains a more accurate picture of the network at the expense of power consumption. In this way, TEEN is able to provide a customisable balance between temporal resolution and power consumption (hence network lifetime). However, if neither thresholds are broken, the nodes never communicate meaning that the user will not receive data. This is resolved by the authors in APTEEN [63] where, in addition to hard and soft thresholds, periodic messages are transmitted if the sensor node has not reported for a period Δt_c [s]. Through this, the user is able to efficiently maintain an up-to-date picture of the network, and obtain notification of important events. APTEEN also introduces the possibility of the user querying the network at any node for 'on-the-fly' information retrieval. Hu *et al.* [64] propose an algorithm based upon APTEEN, which adaptively adjusts the value of S_T (essentially controlling the resolution of disseminated data) through attempting to keep the ratio between the number of packets disseminated as a result of Δt_c and S_T constant.

Dhanani *et al.* [59] propose two event-driven dissemination techniques, SCUB and RM, which aim to manage the flow of information in order to reduce the energy consumption. SCUB makes decisions locally at each node as to whether or not data should be disseminated to the user. These decisions are performed by comparing the utility of the energy (which considers the desired network lifetime that is preset by the user) with the utility of the data. The utility of the data is calculated using the assumption that the time since a packet has been disseminated and the change in sampled data are directly proportional to the usefulness of the data. RM (a distributed version of SCUB) uses a single 'resource manager' node in the network to make decisions regarding data dissemination. Nodes in the network transmit a packet to the resource manager and, if the utility of the data dictates so, forwards it on to the user (if the packet is the first of a multiple-packet sequence, the resource manager notifies the source node to transmit the full sequence to the sink).

Werner-Allen *et al.* [65] present a WSN used to monitor the behaviour of an active volcano, using an event based mechanism to control data dissemination. In this system, each node runs an event detection algorithm that computes short-term and long-term exponentially-weighted moving averages over the sampled data. When the ratio between the two averages exceeds a preset threshold, the node transmits an event report to the base station. If the base station receives triggers from at least 30% of the active nodes within a ten second window, the event is considered to be well-correlated and data collection is initiated from all nodes.

Predictive approaches to event-detection use a prediction model at both the node and the sink. Packets are reported whenever sensed data differs from the node's prediction by more than a preset value (as the data will therefore also differ from the sink node's prediction by the same amount). This reduces the packet transmissions (and hence energy consumption) while maintaining information throughput, but requires a powerful sink node that is able to maintain independent prediction models for every node in the network. A variety of models have been suggested for use with predictive event-detection, including Kalman filters [66], autoregressive integrated moving averages [67], and a scheme inspired by MPEG video compression [57]. While techniques have been proposed which do not require the presence of reliable channels [68], basic predictive event-detection relies on packets being successfully delivered. This is because if a packet is dropped between the source and sink nodes, the sink is unable to update its predictive model, and the two models become unsynchronised.

Autonomously operating WSNs are desired for industrial environments, where end users require nodes to push information from the network; event-driven dissemination enables this level of autonomous operation [69]. Predictive approaches to event-detection usually require reasonable computation at the sink node in order to apply and update the prediction model. Furthermore, while predictive techniques ensure a high-level of information throughput (allowing the entire sampled signal to be reconstructed within a certain tolerance at the sink node), this level of reconstruction is often not required or important to the end user. Conversely, rule-based approaches to event detection disseminate packets that the user has identified as being useful and, while they require user input to characterise what constitutes an important event, usually provide robust interpretations of sampled data and event detection [70].

2.2.2 Additional Uses of Information Control

The information dissemination techniques discussed in the previous section use information to control the reporting behaviour of a node's operation. In addition, there are other possible uses of information control, including sample rate adjustment, packet reliability and bandwidth management; these are discussed below.

Considerable overlap exists between information dissemination and adaptive sampling algorithms (which decide when sensors should be energised and sampled). In USAC [71], a node's sampling rate is adjusted based upon a prediction scheme, where data is predicted using a limited-window linear regression model. If the current data falls within a confidence interval, the sampling rate is reduced. However, if the data falls outside the confidence interval, the sampling rate is increased so as to enable the sensor model at both the local node and sink node to be updated. While this form of adaptive sampling adjusts the temporal resolution of samples, the spatial resolution of sampling can also be dynamically controlled. Willet *et al.* [72] propose Backcasting, an adaptive sampling technique that uses feedback from the sink node to control the spatial resolution of samples. In Backcasting, a small subset of the network's nodes sample the environment and disseminate data to the sink node. This 'preview' or initial estimate of the environment is processed by the sink node and 'backcast' into the network. Additional nodes in the network are then sampled in order to refine the initial estimation and meet desired error targets. Using centralised information management which 'feeds back' into the network is also investigated by Chong *et al.* [73], which report a system that translates the data stream received at the sink node into a 'context' (such as *"the room is now empty"*), which is then propagated back to nodes in order to adjust their behaviour (for example by lowering their sampling rates).

As proposed in Chapter 1, different packets usually contain varying amounts of information, which can be proportionately linked to the importance of the packet. Differing levels of service (such as latency or reliability) should be provided to these different packets [74]. In ReInForM, Deb *et al.* [75] propose a method for adjusting the reliability of packet transmissions by transmitting more important packets (the quantification of which is not considered) along multiple routes. Through this technique, the information content of a packet is used to control the redundancy, and hence reliability, of packet transmissions. In the adaptive link layer proposed by Köpke *et al.* [76], the properties of the link layer (such as error correction, acknowledgments, transmission power, and data rate) are adapted to alter the end-to-end reliability for different importance packets.

An information-controlled bandwidth management algorithm is proposed by Hull *et al.* [77], using a rule system to prioritise outgoing data packets depending on their derived importance. The rules allow a node to map sensor data to a traffic rate and traffic class. High importance packets are placed in a high priority queue, while low importance packets are placed in a low priority queue. Through this, the system relates packet latency to packet importance. This concept is also used in TinyDB [60], which can assign an importance to packets in the transmit buffer (calculated using the difference between the last transmitted value and the value in the packet, relying on the assumption that the more a value has a changed the more important it is). This importance value is used to prioritise packets in a node's transmit buffer; the packet with the highest importance is transmitted next and, if the buffer becomes full, the least important packet is discarded. The idea of information controlled bandwidth management is also utilised by Rogers *et al.* [78], whereby nodes determine the information content of sampled data and, if the information gained is worth the global bandwidth usage it will disseminate it through the network. The metric used to evaluate the information content is the Fisher information (an information theoretic metric representing the contained information) and is used to represent the accuracy of the data. For example, a sample (or combination of samples) with a large tolerance contains a lower information content to that of a sample having a small tolerance. This form of information management is suited to applications where the entire network is cooperating in target monitoring, and the sampling tolerances vary throughout the nodes.

2.2.3 Discussion

This section has outlined information management techniques in WSNs, with particular interest given to information dissemination. The disadvantages of using continuous approaches in scenarios that do not have random data have been presented, and query-driven and event-driven dissemination investigated as efficient alternatives. Query-driven dissemination is suited to applications where the monitoring requirements change often, or where users infrequently and irregularly draw information from the network. However, in many applications, networks and scenarios, a query-based system in not required and the overheads incurred waste the nodes resources. Furthermore, the query based system can be unable to fulfil the autonomously and stand-alone demands of the end user. Event-driven dissemination reduces the number of communicated packets by not transmitting packets which are deemed to contain redundant information. These event-based detection algorithms can be broadly split into two categories: rule-based approaches, and predictive approaches. In rule-based approaches, the end-user specifies what they deem to be of interest and of importance to the application. In predictive approaches, the nodes report enough data to enable the sampled environment to be reconstructed (within a tolerance) at the sink node. While predictive methods allow the transfer of more information (under information theory, a measure of uncertainty or unpredictability of data), this is often a different measure to that which the end-user is concerned. Rule-based approaches, while requiring end-user input to define events of importance, often provide the best methods for event detection [70].

The majority of information-dissemination algorithms use any implemented measure of information content as a digital method; a packet either contains information (and should be disseminated) or it does not. By quantifying the level of information content (whether discrete or continuous) the level of information contained in a packet, a node’s behaviour (and that of the network as a whole) can be adapted appropriately. Acceptance for, and use of, a quantified and graduated measure of information content was highlighted in many of the systems reported in section 2.2.2. However, meaningfully quantifying the information content of a packet is a nontrivial task, and can often only be adequately and reliably defined using application specific knowledge and measures [76].

2.3 Wireless Communication and Networking for WSNs

Wireless communication and networking is fundamental to the research, design and operation of WSNs. This section investigates architectures and structures for communication, and presents a summary of wireless networking protocols commonly adopted in WSNs. Following this, the state-of-the-art in WSN MAC and routing algorithms are discussed.

2.3.1 Introduction to Protocol Stacks

A node's radio transceiver has no appreciation of a network, or even the concept of 'communicating' with other devices. These tasks are the role of the communication protocol stack. Communication protocol stacks have been utilised for decades, providing a method of formally structuring the functionality of a networking or communication subsystem through the use of multiple layers. The number, contents and function of layers differs between protocols [79]. Each layer of the protocol stack implements distinct networking tasks, and offers services to the layer above while masking the details and complexity of the actual layer implementation. By utilising protocol stacks, different implementations of the same layer becomes possible, for example the network layer can be replaced to implement an alternative routing algorithm with minimal impact on the surrounding layers.

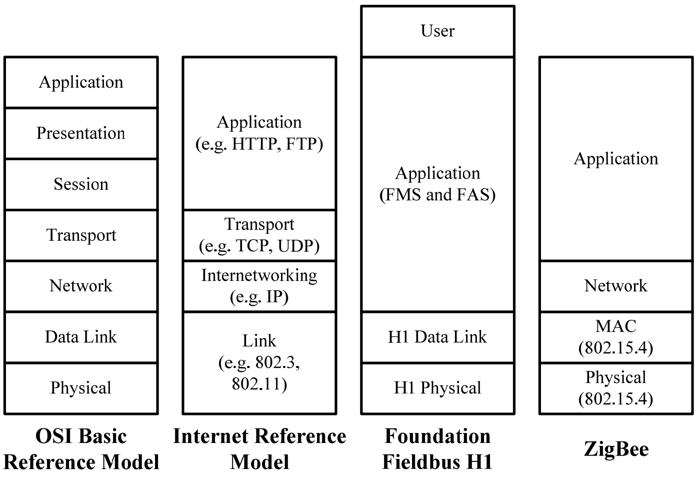


FIGURE 2-4 : The OSI Basic Reference Model (OSI-BRM) [80], Internet Reference Model [81], Foundation Fieldbus H1 Stack [82], and ZigBee Stack [83].

The OSI-BRM (Open Standards Interconnection – Basic Reference Model) [80] was introduced in the mid-1970s, proposing a basic layered structure for communication protocols, shown in Fig. 2-4. The majority of modern communication protocols, stacks and models (for example the Internet Reference Model [81] shown in Fig. 2-4) have added, removed and merged layers from the OSI-BRM to tailor the model to their specific requirements. Additionally, many protocols of relevance to WSNs have absorbed higher layers of the OSI-BRM into their application layer. The functionality of different layers in a protocol stack (such as ZigBee) are described below [79]:

Physical Layer (PHY) is required to perform channel coding and modulation, and manage the timing and communication of bit-level data across the physical medium.

Medium Access Control Layer: performs logical link control (data framing, flow and error checking/correction) and Medium Access Control (controlling multiple accesses to a shared communication medium). The MAC layer is traditionally referred to as the Data Link Layer (DLL) but, due to the prominence of access control in WSN operation, will be referred to in this thesis as the MAC.

Network Layer: manages the operation of the network, and provides functions such as packet routing and congestion control.

Application Layer: contains a range of commonly used high-level protocols, such as HTTP, FTP or SHH. In many protocols, the application layer absorbs the tasks of the Presentation, Session and Transport layers (shown in Fig. 2-4). In smaller devices, the entire application may reside within the application layer.

Fieldbus H1 (one of a number of different fieldbus implementations) [82] was designed as a network architecture for process control applications, such as sensor networks. It can be seen from Fig. 2-4 that the majority of the higher layers of the OSI-BRM have been discarded, as the functionality provided by these layers (for example complex packet routing, flow control and connection management) is not required by the network. Fieldbus adds a User layer to the top of the protocol stack, which allows additional functionality to be provided, such as a user interface. The ZigBee specification [83] defines a low-cost, low-power wireless communication standard that is particularly suited to wireless sensor networks. The ZigBee protocol stack (shown in Fig. 2-4) uses the Physical and Medium Access Control layers of IEEE 802.15.4 [84]. The transport layer is omitted, with the relevant functionality being absorbed by neighbouring layers.

A number of variations on the basic communication stack have been proposed in the literature. Durresi [55] proposes a ‘protocol stack tree’ concept (shown in Fig. 2-5a), an architecture containing a number of different protocols at each layer that satisfy different application requirements. For example, and as shown in Fig. 2-5a, the protocol stack tree would offer a different set of ‘matched’ protocols for real time applications to delay tolerant applications. Protocol stacks need not be constrained to only structuring communications. Graumann *et al.* [85] propose a location stack (shown in Fig. 2-5b) to formalise and structure the process of obtaining a node’s location, and existing structured architectures such as that proposed for intelligent sensing in Karatzas *et al.* [86] could be implemented into a stack architecture. Akyildiz *et al.* [8] proposed a ‘3-dimensional’ stack for use with sensor networks, where three ‘management planes’ were structured in addition to five of the traditional layers from the OSI-BRM (shown in Fig 2-5c). These cross-layer management planes monitor the task, movement and power distribution among the sensor nodes, and provide functionality to all of the stack's layers. Cross-layer design 'breaks the rules' of a protocol stack by, for example, sharing variables between layers in the stack, providing interfaces between non-adjacent layers, and the merging of adjacent layers [87]. While cross-layer design can increase efficiency, it reduces the transferability of developed layers as, by not following the 'standard' rules of the protocol stack, different implementations of the same layer now rely on different interfaces, variables and provided functionality.

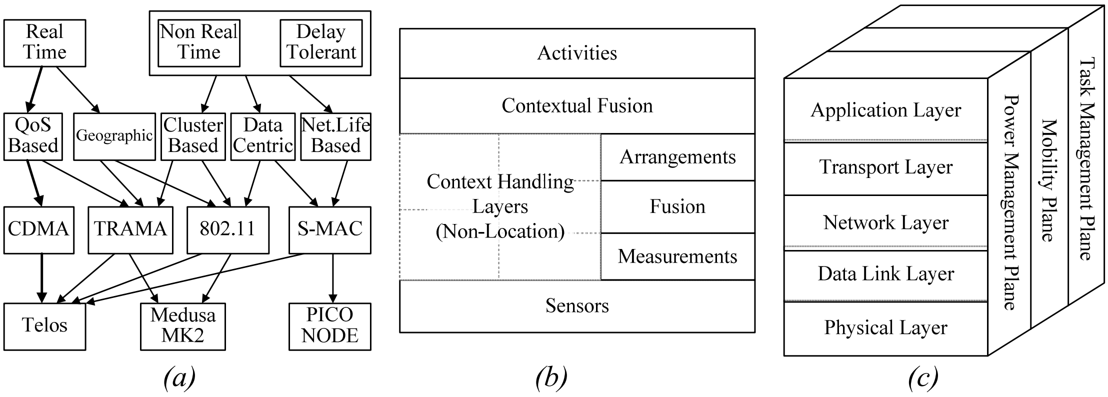


FIGURE 2-5 : Alternative stack architectures for WSNs: a) a protocol stack tree (reproduced from Durresi [55]), b) locationing stack (reproduced from Graumann *et al.* [85]), and c) a '3-dimensional' stack (reproduced from Akyildiz *et al.* [8]).

To provide greater infrastructure and an environment more similar to that of traditional computing devices, a number of operating systems have been designed and proposed for WSNs. TinyOS was the first operating system specifically designed for WSNs, but does not provide functionality offered by a traditional operating system such as scheduling, memory management and multithreading. Instead, TinyOS offers a library of software tools to assist in tasks such as clock synchronisation, power management, radio communications and sensing [60]. Execution in TinyOS is largely event driven (including interrupts from physical hardware), and does not consume clock cycles while watching for events to occur. Individual applications are compiled to include only the software components that they require, hence reducing the storage overhead required of the constrained sensor node [11]. TinyOS is supported by the TinyDB query processing system, and the TOSSIM network simulator. Due to the wide range of nodes that support TinyOS, its popularity and usage makes it the most used operating system for WSNs (used by thousands of developers worldwide) [88]. However, reported problems with TinyOS include space and timing limitations, and the additional overheads incurred by the generic solution can present difficulties in designing ultra-low power algorithms and platforms. Alternative operating systems for WSNs include SOS (improving interrupt handling) and Mantis OS (which operates more like a traditional operating system in providing threaded execution) [89].

2.3.2 Summary of Communication Protocols for WSNs

A number of communication standards have been used with WSNs, including Infrared Data Association (IrDA) [119], HomeRF [119], IEEE 802.11 (also known as WiFi) and Bluetooth (formally standardised as IEEE 802.15.1) [4, 19, 90]. However, most of these have use in only a number of limited applications due to issues regarding transmission distance, network formation and power consumption. Instead, most implemented WSNs use either proprietary protocols or, more recently, solutions built upon the IEEE 802.15.4 standard [84].

IEEE 802.15.4-2006 standardises the design of Low Rate Wireless Personal Area Networks (LR-WPANs), which are particularly suited to sensing applications where a high data rate is not intrinsically required. Contrary to Wireless Local Area Networks (WLANs), a WPAN involves little or no infrastructure, and operates with a low duty cycle; hence WPANs are suited to applications with fixed or mobile devices with very limited energy sources. IEEE 802.15.4 defines only the PHY and MAC layers, leaving other standards to build on top of these (such as ZigBee [83], ISA-100 and Wireless HART [5]). IEEE 802.15.4 supports communication at 868MHz, 915MHz and 2.4GHz; 2.4GHz is the only one of these that supports global use due to differing regulations in the USA and Europe. Communication at 2.4GHz is receiving increased used in WSNs (driven by available silicon, volume sales, and the suitability of the frequency to many environments) [17]. It is reported that nearly half of the nodes deployed last year were based on the IEEE 802.15.4 standard [91].

While the network layers of established protocols such as IEEE 802.15.4 and ZigBee are standardised, established, and commercially available, the design of alternative layers for WSNs receives considerable research interest. An overview of state-of-the-art MAC and NET layers is provided below, with particular interest paid to energy-aware algorithms.

2.3.2.1 Medium Access Control Layer

As mentioned above, the MAC is traditionally a sub-layer of the data link layer, which provides framing, error control, security, and flow control. However, in WSNs the workload of the DLL is considerably reduced, while that of the MAC (regulating the access of a number of nodes to a shared medium [92]) is heavily increased. As such, the MAC is often considered to absorb the tasks of the DLL, and is referred to a distinct layer in its own right:

the MAC layer. As nodes are usually small, cheap and low-power, algorithms are often reasonably simple and onboard clocks not largely accurate. Time synchronisation therefore becomes an issue and algorithms requiring precise time awareness are rendered unfeasible. In WSNs, the primary performance metric is usually energy efficiency, while metrics such as scalability, adaptability to changes, latency, throughput, fairness and bandwidth utilisation are of secondary importance. Energy-efficient MAC design is performed by addressing the following areas of energy waste [93, 94]:

> **Idle Listening:** Radio transceivers typically have four power levels corresponding to the following states: transmitting, receiving, listening (waiting to receive) and sleeping [95]. As idle listening often consumes as much energy as receiving (as reflected in Fig. 2-1), a node that is listening but not receiving wastes significant energy. Therefore, an energy saving can be obtained by duty cycling the transceiver, so that it spends most of its time in a sleep state (as discussed in section 2.1).
>
> **Overhearing:** If a unicast packet (a packet with a single destination) is broadcast by a node, neighbouring nodes waste energy in receiving enough of the packet to realise that it is not destined for them. This is not a problem for broadcast packets where every neighbour is supposed to receive the packet.
>
> **Collisions:** Packets that collide are corrupted, and require retransmission by the source node (incurring an energy cost).
>
> **Overheads:** Energy is consumed in overheads such as MAC headers and footers, control packets, and acknowledgments.
>
> **Over Emitting:** Energy is wasted in a node when a packet is transmitted when the receiving node is not yet ready to receive it (thus requiring a retransmission).

This following subsections provide a review of research into WSN MACs, discussing fixed assignment protocols, random access protocols, and wakeup radios [94, 96, 97].

2.3.2.1.1 Fixed Assignment Protocols

Fixed assignment protocols are those which proactively allocate discrete sections of time (TDMA), spectrum (FDMA), or coding (CDMA) to different nodes, and these nodes only use only these sections to transmit. Generally, these protocols are not particularly suited to WSNs as they generally require a central authority to assign the discrete section. TDMA schemes require good time synchronisation and do not adapt easily to topology changes.

FDMA schemes require additional circuitry to dynamically communicate with different channels. CDMA requires a high computational load, which is often not feasible on a WSN node. Furthermore, the infrequent channel utilisation of WSNs means that collisions occur less often than in their wired counterparts when using random access protocols. For these reasons, fixed assignment protocols are not extensively considered in this report (though more information can be found in [92, 94]).

2.3.2.1.2 Random Access Protocols

In random access protocols, each node competes for communication over a shared channel, and nodes must locally decide when it is best to transmit. This method of channel access can be either slotted (communication can only start at predefined times) or un-slotted (nodes can communicate at any time). ALOHA is a random access protocol for traditional networks that allows a node to transmit whenever it needs to. If a collision occurs, the node retransmits. In slotted ALOHA, a node wishing to transmit waits until the start of the next slot (thus reducing collisions). CSMA (collision sense multiple access) algorithms inspect the channel before transmitting to see if they are going to interfere with other nodes [98]. If the channel is not clear, the node either waits until it is and then transmits immediately (1-persistant CSMA) or 'backs-off' for a random period of time before re-attempting transmission (non-persistent CSMA). Woo *et al.* [98] postulate that this random 'back-off' should be introduced prior to any transmission in order to phase shift the periodicity of the application. CSMA-CD (CSMA with collision detection) is able to detect occurring collisions by listening to the channel while transmitting; this requires a transceiver that can receive while transmitting, and hence is not commonly used in WSNs. CSMA-CA (CSMA with collision avoidance) implements handshaking between a node and its neighbours prior to data transfer using request-to-send (RTS) and clear-to-send (CTS) control packets. This attempts to overcome the hidden-station and exposed-terminal problems (which are a cause of collisions) [79].

The B-MAC (Berkeley MAC) protocol is a simple energy-efficient protocol (and the default in TinyOS) [99]. The concept of B-MAC is that a node wishing to transmit first performs CSMA and, if clear, transmits a preamble for a time Δt_p [s]. After this has elapsed, the packet is transmitted. A node that is not transmitting, periodically (with period Δt_p) wakes up and briefly samples the channel to see if a node is transmitting a long preamble. If it is, it stays on long enough to receive the packet. This is known as low-power listening (LPL), as the receiver node only has to wake up for a short period of time. LPL does not require node synchronisation, and shifts the cost of communicating from the receiver to the transmitter. However, the use of LPL in B-MAC's decreases the effective channel capacity as the transmitting node utilises the channel for > Δt_p per packet. X-MAC attempts to overcome many of the shortcomings of B-MAC (including overhearing, excessive waiting at the receiver for the preamble to finish, and latency) [100]. X-MAC makes these improvements by modifying the preamble sequence. Instead of transmitting a single non-descript preamble for the duration Δt_p, X-MAC repeatedly transmits the destination address. Between repeats, X-MAC also waits for a period of time to allow the receiver to 'acknowledge' and stop the preamble, initiating data transfer. This reduces the latency (as there is not a delay of Δt_p for every packet), reduces the idle listening energy consumption (as the receiver does not have to listen to the preamble even after it has detected it), and reduces overhearing (receiver nodes that hear the preamble but with a different nodes address simply go back to sleep). X-MAC also provides a method for dynamically adapting Δt_p dependent on the traffic load. PicoNet uses an alternative form of LPL, with energy savings based on the assumption that the receive power is less than the transmit power [101]. When a node periodically wakes up to receive, it first broadcasts its address. Nodes that wish to transmit to it remain awake until they receive this broadcast, at which time packet transmission commences.

S-MAC (Sensor-MAC) is a synchronised WSN MAC protocol, which attempts to reduce the energy consumptions associated with idle listening, overhearing, and collisions [93]. Idle listening is reduced by the adoption of duty cycling, consisting of a sleep period and a slotted listen period. The durations of these periods are fixed, but their timing is dependent on the node's schedule. Neighbouring nodes synchronise their schedules in order to wake up and communicate at the same time. To maintain time synchronisation, schedules contain relative times, and listen times are designed to be significantly larger than that of clock drift. Collisions are reduced through the RTS/CTS handshaking process implemented in CSMA-CA. Overhearing is reduced by the address of destination node being signalled during the CSMA-CA handshake sent prior to data transmission. S-MAC was extended in [102] to reduce latency through adaptive listening. In adaptive listening, nodes hearing an RTS or CTS transmission that is not destined for them will wake up after the transmission ends in case the packet is to be forwarded to them. T-MAC (Timeout-MAC) is a variation of S-MAC by Van Dam *et al.* [103], adding adaptive duty cycling into the S-MAC protocol. Under a homogeneous load, T-MAC and S-MAC perform comparably. However, under certain variable load conditions, T-MAC is able to outperform S-MAC by a factor of five.

2.3.2.1.3 Wakeup Radios

An attractive concept for low-energy ’reactive’ networks is that of the wakeup radio, where a receiving node’s primary receiver is ‘woken up’ by a low-power secondary receiver in response to a trigger from the transmitting node [92]. This virtually eliminates idle listening on the primary radio (presuming that only the desired node wakes up), reduces latency (as receivers are woken up when they are needed) and reduces collisions (as transmissions are no longer scheduled into discrete communication periods).

Miller *et al.* [95] propose a wakeup radio MAC protocol that assumes that the secondary low-power wakeup radio is capable of only communicating a busy tone. In the proposed protocol, a wakeup tone causes all neighbouring nodes to wake up their data radios until a filter packet is received (containing the destination address). A tuneable trade-off between full-wakeups (as described above) and triggered-wakeups (using duty cycling on the primary radio) is possible. The duty cycle of the primary channel for triggered wakeups is significantly less than the wakeup radio’s duty cycle. In the MAC proposed for Pico-Radio by Zhong *et al.* [104], the authors assume the presence of a secondary wakeup radio (consuming less than 1μW at full duty cycle [105]) that is able to transmit data as opposed to simply a busy tone. A transmitting node transmits a wakeup packet containing the destination address and channel ID. This causes only the desired node to wake up and switch the receiver to the correct channel (chosen by the transmitting node through a combination of random selection and CSMA).

‘Zero-power’ wakeup radios that, using techniques established for passive RFID, are powered from ambient energy are very attractive in accomplishing the goals of wakeup radio MACs [106, 107]. Gu *et al.* [108] propose that such a virtually zero-power (typically <1μA) wakeup radio is a realistic proposition. The wakeup detection circuitry creates a voltage directly from energy in the received electromagnetic signal, which drives a hardware interrupt on the microprocessor. The authors propose both simple wakeup beacons (with no modulation or encoded data), and also the possibility of selecting only certain nodes to wake up by using multiple frequency transmitters (for example transceivers that are able to concurrently receive or transmit at multiple frequencies) to encode a wakeup ID beacon.

2.3.2.2 Network Layer

The primary function of the NET layer is to perform packet routing, defined by the International Telecommunication Union (ITU) as:

Definition: Packet Routing

> *"the process of determining and using, in accordance with a set of rules, the route for the transmission of a message, ending when the message has reached the destination location"* [109].

With reference to WSNs, routing is the process of delivering a packet from a source node to a sink node. Three distinct communication topologies exist for communicating a packet between its source and destination: single-hop, flat multi-hop, and clustered multi-hop communication (a hop is defined as a single direct transmission between nodes) [110, 111]:

Single-Hop (Direct/Star): source nodes communicate directly with the sink node over a single hop, as shown in Fig. 2-6a. Routing is trivial, but as the diameter of the network increases, so does the required transmission power of nodes (therefore increasing the transmit power consumption).

Flat Multi-Hop (Peer-to-Peer/Mesh): source nodes communicate with the sink node via intermediate nodes, as shown in Fig. 2-6b. The route taken depends on the protocol, but could take a large number of short hops, or a small number of large hops.

Clustered Multi-Hop (Hierarchical): the network is subdivided into clusters (either at placement using heterogeneous nodes, or through self-configuration). Source nodes communicate with their cluster head, which route data to the sink node, as shown in Fig. 2-6c. This is an amalgamation of the single-hop and flat multi-hop topologies.

An analytical investigation of clustered multi-hop routing is presented by Vlajic *et al.* [112], concluding that the optimal choice of flat multi-hop or clustered multi-hop routing depends upon the particular scenario. In this research, clustered multi-hop techniques are not considered, to enable local, reactive, true peer-to-peer networking.

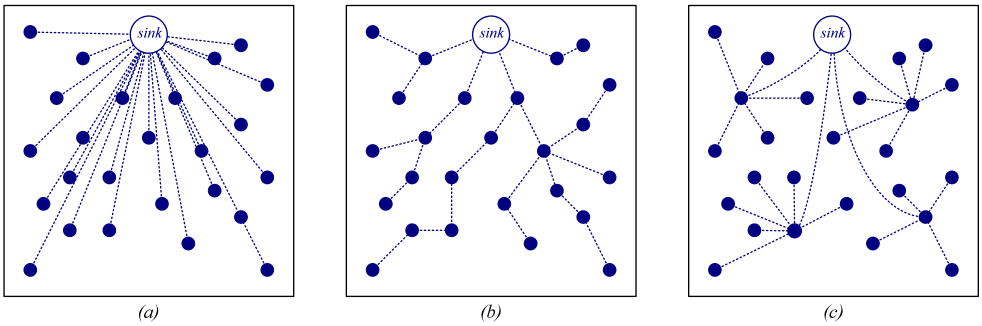


FIGURE 2-6 : Different routing topologies: a) direct, b) flat multi-hop, and c) clustered multi-hop.

Equations (2.2) and (2.3) show the end-to-end power consumed to route a packet through single- and multi-hop techniques, where P_{tx} [W] is the transmission power consumption, P_{rx} [W] is the receive power consumption, d [m] is the separation distance, λ [m] is the wavelength, η is the path loss exponent, and H is the number of hops [47].

$$P_{tx}^{singlehop} \propto P_{rx} \left(\frac{d}{\lambda}\right)^{\eta} \tag{2.2}$$

$$P_{tx}^{multihop} \propto HP_{rx} \left(\frac{d}{H\lambda}\right)^{\eta} \tag{2.3}$$

By combining (2.2) and (2.3), the ratio between the transmit power consumption of single-hop and multi-hop is obtained (2.4). From this, it can be seen that a transmit power saving of $H^{\eta-1}$ is achievable through multi-hop routing; for example, routing over five hops with a path loss exponent of four (feasible for an indoor environment) results in a transmit power consumption reduction of two orders of magnitude over single-hop routing.

$$\frac{P_{tx}^{direct}}{P_{tx}^{multihop}} = H^{\eta-1} \tag{2.4}$$

Communicating via many smaller hops as opposed to single-hop routing reduces the node's required transmit power [113]. Multi-hop routing however, is not necessarily the most efficient or optimum communication technique. However, it is not sufficient to consider only the transmit power consumption, as the power consumed by routing nodes receiving the packet is also significant. Furthermore, multi-hop routing provides a reduction in power consumption only when the path loss dominates the energy consumption of the hardware [114]. In practice, oscillators and bias circuitry often dominate the transmit power, meaning that an increase in separation distance does not necessarily result in an equal rise in power consumption. Multi-hop routing also has the ability to introduce problems including traffic accumulation, latency, and end-to-end reliability [115]. It is however important to note that while the power required to propagate a packet from its source to destination may be higher when using multi-hop routing, this cost is spread across a number of nodes instead of a single node, thus helping to evenly distribute the network degradation over the nodes. This means that nodes in a particularly active area of the network (which is likely to therefore be an area of major interest) will operate for longer before depleting their energy reserves.

It is generally accepted that the optimality of single- or multi-hop routing depends largely on the hardware, scenario and application. Often, a small number of large hops is more efficient than a large number of the smallest possible hops [116]. Multi-hop routing is also particularly useful when operating in hostile environments (the high path loss results in low transmission distances) or when a large separation distance would necessitate significant transmission power levels (which may not even be greater than the maximum permissible by radio communication regulations); these are the scenarios considered by this thesis.

Routing algorithms can also be classified as proactive, reactive, or a mixture of both [4, 111, 117]. In proactive routing, each node maintains routing tables depicting the routes for reaching different nodes in the network. These routes are computed before they are required, and maintained on a periodic basis (introducing overheads). Proactive routing is usually undesirable in WSNs due to its lack of efficient scalability (as the network is scaled, route setup, maintenance, and storage becomes more demanding on resources). In reactive routing, nodes dynamically compute routes to the destination on demand. Reactive routing removes the overhead to setup and maintain the routing tables, but greater effort is required upon propagating packets. Reactive routing is generally favourable in networks with infrequent communication, or with a network that is rapidly changing (where optimal routes continuously change). The level of claimed 'reactivity' varies between algorithms, from those that setup routes per 'session', to those that determine the route on a hop-by-hop basis.

A wide range of different routing algorithms exist (and are the subject of major international research), often optimised for specific applications, scenarios, or network parameters. Detailed surveys of routing in for WSNs are given in [110, 111, 113]. The remainder of this section investigates flooding-based, location-aware, and energy-aware routing algorithms. The consideration and investigation of only this small subset of routing algorithms is due to the particular requirements of the algorithms developed in this research. These requirements are for reactive algorithms (that calculate routes as and when they are required) that are suited to reasonably small energy-aware networks and incur minimal overheads to employ. Furthermore, flooding-based routing algorithms have the additional advantage that all possible routes through the network are attempted (although constrained flooding discounts routes that are particularly inefficient). The reason for these requirements will become clear in later sections (they are dictated by the nature of the developed algorithms).

2.3.2.2.1 Flooding-Based Routing Algorithms

Flooding [118, 119] is a simplistic reactive flat multi-hop routing protocol, where nodes broadcast received packets to all their neighbours unless they have already forwarded the packet, a maximum number of hops has been reached, or the node is the packet's destination. The major advantages of flooding are simplicity, scalability, robustness and minimal overhead [120]. However, disadvantages include implosion (receiving multiple copies of the same packet from its neighbours), overlap (the event will be detected by multiple sensor nodes, which all report the same data in unique packets), resource blindness (resources such as the remaining energy are not taken into account), channel contention and collisions [121, 122]. Flooding is efficient for one-to-many communication but inefficient for one-to-one communication due to the energy wasted to transmit the packet to every node in the network. The process of propagating a packet across a network using packet flooding is shown in Fig. 2-7.

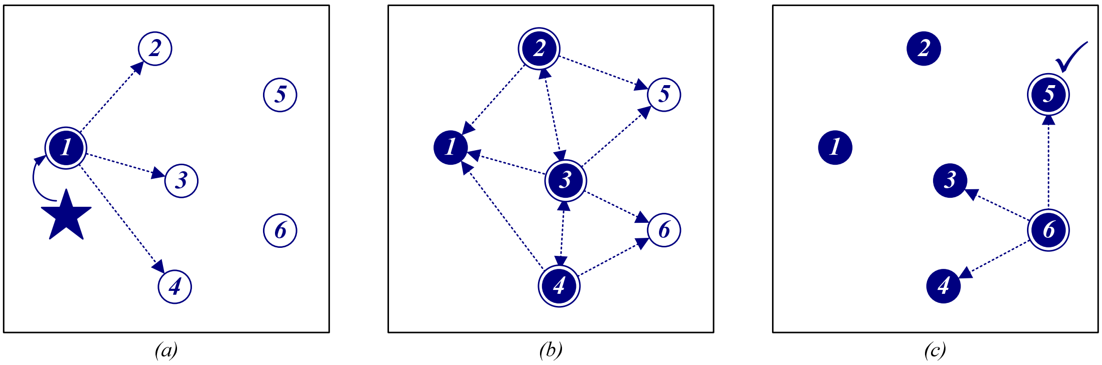


FIGURE 2-7 : The process of flooding a packet from node 1 to node 5. a) Node 1 broadcasts the packet to its neighbours: nodes 2, 3 and 4. b) Nodes 2, 3 and 4 broadcast the packet to their neighbours: nodes 1, 2, 3, 4, 5 and 6. Nodes 1, 2, 3 and 4 have already received the packet, and so ignore it. c) Node 5 (the destination) receives and processes the data. Node 6 continues to broadcast the packet to its neighbours, as it is unaware that it has been successfully received at node 5.

A number of modified flooding algorithms have been presented in the literature. Gossiping [123] is a variation of flooding whereby a node forwards packets to a randomly selected subset of its neighbours (as opposed to all of its neighbours). Gossiping allows the power consumption to be tuned through a set of parameters controlling the number of neighbouring nodes that a packet is forwarded to. Gossiping is more efficient than flooding for one-to-one communication, but inefficient for one-to-many communication (as it takes longer than flooding for a packet to propagate to all nodes). As its name suggests, Flossiping [118] is a combination of both flooding and gossiping, serving both one-to-one and one-to-many communications, whilst keeping the protocol as simplistic as possible. Other modifications to target and constrain the flooding include a framework for constrained flooding (reducing the number of retransmissions) [124, 125], and directional flooding (targeting the flood towards the sink using location-aware nodes) [126].

2.3.2.2.2 Geographical Routing Algorithms

Geographical (or location-aware) routing algorithms are reactive algorithms able to route with little knowledge of the network around it. Geographical routing algorithms generally operate by making a greedy choice for the next-hop, selecting a node that is geographically closest to the sink node (though this is not always the case [127]). Different location-aware algorithms tend to differ by their method of routing around communication holes. A communication hole is encountered when a local maximum is observed during geographic routing (that is, the current node is closer to the sink node than any of its neighbours). The methods used to navigate around communication holes include localised flooding [128], minimum cost [129], right-hand rule navigation [130, 131], and face routing [132]. While most geographical routing algorithms assume that the nodes know their actual location, some techniques have been developed to operate using nodes' relative location. For example, nodes could measure the received signal strength of a packet broadcast at high power from the sink node, and translate this into their approximate distance from it [133].

2.3.2.2.3 Energy-Aware Routing Algorithms

Directed Diffusion [134] is a data-centric routing algorithm for WSNs, designed for query-based networks. Directed diffusion has four main concepts: interests, data, gradients and reinforcement. An interest is the query relating to what the user is interested in, and is flooded through the network from the sink node. Each node that receives an interest constructs a gradient (specifying both data-rate and the route) back to the sink. Data (named using an attribute-value pair, for example *"temperature = 13; location = [13.2 , 1.7]"*) from applicable nodes is returned from the sensor nodes along these gradients. As data is returned to the sink, reinforcement is used to select and 'reinforce' the optimal route from many possibilities. Aggregation may be performed along the route to reduce the energy consumption of the network. While diffusion is not particularly energy-aware in itself, its popularity has resulted in a number of energy-aware algorithms being based upon it [135, 136]. PRIMP [137] is such an algorithm which, when propagating interests throughout the network, marks the interest with a priority. This priority will label the node as either high priority or low priority, determined by the node's residual energy relative to a threshold. Upon returning data back along the gradients, a packet will be propagated via high priority nodes where possible, with the node with the fewest number of hops to the sink being selected first. If no high priority nodes exist, low priority (energy constrained) nodes are selected, with the node with the highest residual energy being selected first. PRIMP is also able to increase reliability by transmitting packets along multiple routes, or to multiple sinks. Shah *et al.* [138] present EAR, which aims to distribute the energy consumed through routing over the network. Unlike diffusion with which is shares many similarities, EAR reinforces multiple routes from the source to sink, each with an associated probability of being selected. The probability is calculated by considering both the residual energy and the energy that is required to communicate using the route. By occasionally routing via sub-optimal routes, EAR spreads the energy cost over the network, minimising the variance in nodes' residual energy levels. Park *et al.* [139] extend EAR by overhearing (eavesdropping on other nodes' communication) to keep route probabilities dynamically updated.

Harvesting-aware routing algorithms have received limited research. Voigt *et al.* [140] proposed a modified version of diffusion which takes the current state and availability of energy harvesting into account when selecting the next hop. The authors have also proposed a similar modification to LEACH (a clustered multi-hop routing algorithm) which takes into account the availability of energy harvesting when selecting a node to become the next cluster head [141]. Research into predicting energy-harvesting availability by Kansal *et al.* [49] has demonstrated the possibility of implementing harvesting-aware routing by considering how much energy is currently being harvested, but also considering how much is likely to be harvested in the future. For example, the authors propose a routing cost metric that considers both the residual energy on the node and the harvesting opportunity of the node (the amount of energy it is likely to harvest in the future).

2.3.3 Discussion

This section has provided an overview of communication protocols and algorithms for WSNs. Protocol stacks provide an established method (which has been arguably fundamental to the success of the internet) to formally structure the tasks of a communication system. These have evolved through time to suit different applications and technologies, but protocol stacks remain heavily targeted towards communication, with other node functions such as sensing and energy management being bundled into the application layer. Current research is beginning to divert away from the strict rules enforced by protocol stacks, towards cross-layer protocols which merge layers, and allow the creation of interfaces between non-adjacent layers. Operating systems for embedded nodes in WSNs provide event-driven or thread-driven execution, and a library of interfaces and functionality for ease of deployment. Protocol stacks are implemented as part of the communication sub-system of an operating system.

MAC protocols control and manage point-to-point communication between two nodes. The major design challenges are involved with reducing energy wastage through idle-listening, overhearing and collisions. LPL techniques transfer the energy cost of communicating from the receiver to the transmitter; this is particularly suited to low frequency event-driven networks. A variety of research challenges are posed in order to obtain energy-efficient reactive MAC protocols that are able to operate in rapidly changing networks. The prospect of 'zero-power' wakeup radios may provide novel and interesting solutions to this, and create a future range of MAC protocols functioning on new and untraditional concepts.

NET protocols control and manage end-to-end communication between a source node and a sink node(s). Research into routing algorithms for WSNs is a rapidly developing area, with hundreds of new algorithms being published every year. This is due in part to the diversity of WSNs, and there is no single solution suitable for all applications. Flooding algorithms are simplistic, reactive, reliable and have low latency. However, they do not scale well for unicast communication as considerable redundancy (hence energy) is introduced through forwarding the packet to every node in the network; constrained flooding algorithms have been proposed to limit these effects. Geographical routing algorithms are suitable for reactive routing in cases where nodes are aware of their own locations, but are susceptible to problems when routing holes are encountered. A range of energy-aware routing algorithms have been developed, which consider a node's residual energy when deciding which route a packet should be sent. Also, a small number of harvesting-aware routing algorithms have been considered where, in addition to a node's residual energy level, its ability to harvest energy also forms part of the routing decision.

2.4 Simulation and Modelling

There are three techniques for analysing networks: analytical methods [142], computer simulations, and practical implementations. The constraints and complexity of WSNs often cause analytical methods to be unsuitable or inaccurate [143]. Additionally, the proportion of algorithms that are analysed through practical evaluation is comparatively low, possibly due to the relative infancy, deployment cost, broad diversity, and application dependence of WSNs. As a result, simulation is currently the most widely adopted method of analysing WSNs, allowing the rapid evaluation, optimisation, and adjustment of proposed algorithms and protocols. A summary of simulators for WSNs is provided in section 2.4.1.

Simulation allows certain areas of network operation to be left out or simplified; for example assuming that packet collisions, interference and noise do not occur, that nodes are always perfectly synchronised with one another, or that particular consumers do not consume any energy. These simplifications often make the process of development and evaluation faster and easier, but can result in algorithms that are not realisable in practice; hence a simulation is only as realistic as the models and assumptions that it is based upon. Existing environmental and physical models for WSN simulation are investigated in section 2.4.2.

2.4.1 Wireless Sensor Network Simulation

While simulation is reasonably well established for Mobile Ad Hoc Networks (MANETs), the simulation of WSNs not only requires the implementation of a radio channel, but also a physical environment and accurate energy models. The design aims and strategies of different simulators result in them each having different strengths and weaknesses; an appreciation of this is essential in either selecting a simulator, or simulation creation. Simulators for use with WSNs can be classified into two predominant categories: those that have been developed as extensions to existing network simulators (such as the SensorSim [144] extension to ns2 [145]), and those that have been designed specifically for the WSNs simulation (such as J-Sim [146]). The models that different simulators employ are often their distinguishing factors; these however are investigated in section 2.4.2. This subsection provides an overview of the design and architecture of some of the major WSN simulators.

Ns-2 [145] – probably the most popular simulation tool for sensor networks – is an object-orientated discrete event network simulator based upon the Real network simulator (ns) that was released in 1989. It is reportedly hard to make changes to and develop extensions for [143]. While this is not such a problem for traditional networks (protocols such as Ethernet and TCP do not require alterations as they are well established), it poses obstacles in the simulation of WSNs. Though ns-2 is relatively complicated to use, researchers are often happy to invest the time in learning how to use it due to its popularity and user-base. The extensibility of ns-2 has been a major contributor to its success, with protocol implementations being widely produced and developed by the research community. Additionally however, ns-2 is limited by its scalability (interdependencies between objects in the object-orientated design do not scale well) and the lack of an application model (sensor networks often require interactions between network and application layers). SensorSim [144] is an extension of ns2 aimed at the simulation of WSNs. SensorSim provides advanced models and the ability to interact with external applications (such as real sensor network hardware). SensorSim is currently withdrawn from release and, as it is built on top of ns-2, suffers from the same scalability problems. Like ns-2, OMNeT++ [147] is a discrete-event general purpose network simulator. OMNeT++ is structured around a modular system: simple modules (such as layers of a protocol stack) contain algorithms, making up the lowest level of hierarchy, while compound modules (such as a sensor node) contain simple modules that interact with each other using messages. OMNeT++ has a versatile Graphical User Interface (GUI) allowing, for example, the user to inspect interconnections between modules and the messages being transferred between them. SenSim [148] is a sensor network extension for OMNeT++. Within the complex module of a node, modules are present to represent each protocol layer, the hardware, and a coordinator (responsible for passing messages around the node). Additional modules outside of the nodes represent a sensor channel and a network channel. However, use of SenSim requires a reasonably high learning curve, which is generally not popular with simulators that are not widely established. Also, due to the lack of a significant user base, there are not many developed protocols available for it. The Castalia [149] simulator is also built upon OMNeT++, and is a model-centric extension for WSNs, providing a range of accurate models to the end-user. The GTSNetS simulator [150] is a sensor network extension to the GTNetS simulator, which aims to provide a scalable, highly extensible and customisable, model-centric simulator to WSN researchers, and also enables the simulation of sensor control networks.

In order to overcome scalability issues inherent in object-orientated structures, J-Sim [146, 151] is designed around a component structure. While its component-oriented structure increases its scalability, the implementation choice of Java (which makes it truly cross-platform) arguably reduces the possible efficiency of the simulator. J-Sim is relatively complicated to use and, due to no real established user base, is not widely adopted. SENSE (Sensor Network Simulator and Emulator) [143] improves on the efficiency of J-SIM by providing a component-orientated architecture programmed in C++, and improving on the inter-communication efficiency of J-Sim. However, SENSE lacks developed extensibility, and does not include functionality such as sensing.

TOSSIM [152] is both a simulator and an emulator for WSNs, in that it simulates TinyOS code for the Mica range of nodes. All nodes in the network must run identical code and, while sensor hardware is modelled, the environment is not. However, TOSSIM provides obvious advantages to projects that are to be implemented on the MICA nodes. The SenQ simulator [153] (extending the GloMoSim and QualNet simulators) continues the concept of providing software emulation, and increases the efficiency, flexibility and modelling accuracy of code emulators such as TOSSIM. ATEMU [154] increases accuracy by providing emulation of the processor on the MICA2 node, and allowing cycle-level emulation of code. This obtains simulation accuracy at the expense of scalability and speed. Avrora [155] also provides cycle-level emulation, but sacrifices continuous synchronisation between nodes in order to improve the scalability of ATEMU.

Many other simulators have been created for investigating specific functions and attributes of WSNs, for example OLIMPO [156] (designed to allow the tracking of specific network traffic through multi-hops), SenSor [157] (a high-level top-down simulator, allowing applications to be incompletely specified), and Sensoria [158] (developed centrally around the GUI to enable easy network design, formation, alteration and simulation).

2.4.2 Environmental and Physical Modelling

To ensure correlation between simulation and practical results, accurate models representing the hardware, node, and surrounding environment are needed [159-161]. However, the level of detail required in order to consider a simulation 'accurate' is largely undefined and, intuitively, it can be seen that WSN simulation at the electron-level is overkill. As stated by Heidemann *et al., "too little detail can produce simulations that are misleading or incorrect, but adding detail requires time to implement, debug, and later change, it slows down simulation, and it can distract from the research problem at hand"* [162]. Algorithm development for WSNs is generally targeted at analysing, improving, or optimising specific performance criteria; hence realistic models should be used for the areas that affect these particular criteria. For example, energy-aware algorithms require realistic energy models in order to ensure close correlation with practical results. Likewise, in order to evaluate information-aware algorithms, realistic sensor models are required. Additionally, because information throughput depends upon dropped and corrupted packets, communication modelling is also required. This section investigates such models for communication, energy, sensing, and timing.

2.4.2.1 Modelling Communication Channels

Radio communication over a wireless channel requires modelling in a WSN simulator in order to determine which nodes are considered to be a transmitting node's neighbours, and to calculate the receipt success of packet transmissions. Two elements should be considered to simulate radio communication: modelling the propagation of signals through the wireless channel, and modelling the effect that these propagation losses have on the link quality.

Channel and communication modelling has been heavily researched for wireless network simulation. Wireless channel propagation can be simulated using the models shown in Appendix A.1, resulting in a received signal strength at the receiving node. The realism and complexity of the propagation model used depends largely on the level of accuracy required by the simulator, and also the properties of the application and environment to be simulated. As such, assumptions and approximations (which can reduce simulation and development time) made by the simulator should be carefully considered for the impact that they will have on the simulation accuracy [162, 163]. Some simulators use simplistic free space propagation, while others implement advanced channel and fading models. The SENS simulator [164] models the environment using a system of square tiles, where each tile can have different characteristics (representing different obstacles and surfaces) affecting the signals that propagate across or through them.

To simulate wireless link quality and packet reliability, the simulator must interpret the received signal strength obtained from the propagation model in order to decide what data is received by a neighbouring node. The nodes that are deemed to be neighbouring can be determined using either a disc model (where any node within a predefined radius of the transmitter are deemed to be neighbouring), or a complete model (all nodes in the network are considered to be potential neighbours). Obviously, the latter is more realistic but less efficient. The communication model can provide data to neighbours at different levels of abstraction, including:

Packet Level Communication: Data is transferred between nodes as packets; neighbouring nodes either receive the entire packet correctly, or not at all.

Bit Level Communication: Data is transferred between nodes as individual bits; neighbouring nodes receive individual bits (which may have had bit errors applied).

Different levels of abstraction can be applied to determining whether an error has occurred in the communicated bit or packet:

100% Success (Bounded Signal Strength): If the received signal strength is greater than a predetermined value, the packet or bit is correctly received by the node.

Bit Error Rate Calculation: Having determined the received signal strength using the propagation model, the Signal to Interference Noise Ration (SINR) can be calculated and the Bit Error Rate (BER) deduced, which is probabilistically applied to the bits (more information on SINR and BER is provided in Appendix A.2). This method is used in the recently proposed Castalia simulator [165]. If the simulator is using packet level communication, a single BER is calculated for the entire packet; this can be unrealistic as temporal variations in noise or interference are ignored, and low-probability extremities in shadowing are applied to packets as opposed to bits.

Modulation Scheme Simulation: The simulation of modulation and encoding is performed. Binary data is modulated, and analogue signal levels are propagated across the channel, being affected by noise and interference. The signal levels are demodulated by the receiving node, and interpreted as binary signals (it is during this quantisation process that channel errors are introduced). This method models the source of bit errors as opposed to relying on a probabilistic BER but, for the majority of simulations, this method is far more complex than required.

Using the methods detailed in this section, neighbouring nodes are presented with either a packet or a series of bits. The receiving nodes must then interpret (and possibly reconstruct) the packet in accordance with its communication protocol stack.

2.4.2.2 Modelling Energy Components

A comprehensive energy model generally consists of a model for an energy store (for example a battery), and energy consumers (for example the radio transceiver).

Energy models to represent battery effects (such as those described in Appendix B.1) are becoming more widely adopted in WSN simulators. However, many existing simulators fail to implement an energy model with anything more than an ‘ideal’ (or linear) battery model, which does not provide an accurate representation of practical hardware. The simple battery model proposed in Park *et al.* [166] (also used in SenSim [148]) describes an ideal battery (2.5), where C_{res} [Ah] is the remaining capacity after discharging for a period Δt_d [s], C_n [Ah] is the rated capacity of the store (effective at time t_0 [s]), and I [A] is the current being drawn at time t [s].

$$C_{res} = C_n - \int_{t=t_0}^{t_0+\Delta t_d} I(t)\, dt \tag{2.5}$$

This can also be expressed in terms of energy as opposed to capacity, and this is widely used as a simple model in WSN simulation (2.6) [144, 150, 165], where E_{res} [J] is the remaining (or residual) energy in the battery, E_{max} [J] is the energy capacity of the store, P [W] is the consumed power, and V_S [V] is the store voltage. Note that in this non-integral form, (2.6) holds true only if the power or current drain remains constant.

$$E_{res} = E_{max} - P\Delta t_d = IV\Delta t_d \tag{2.6}$$

These simple battery models, as shown in Appendix B.1, do not accurately represent the discharge of a battery, and a range of analytical methods, electrical circuit models, stochastic models, and electro-chemical models have been proposed for modelling them [167-169]. However, in WSN simulation, where a large number of nodes have current drains that change very frequently, many of these techniques are too computationally intense to be used and provide an unnecessary level of detail [162].

In order to provide a simple but more realistic representation of a battery, WSN simulators are beginning to model rate discharge and relaxation effects (discussed in Appendix B.1) [166]. The modelling of relaxation is difficult to implement due to its complex nature, being dependent on many device-dependant electrochemical and physical parameters. In the SENSE simulator [143], rate discharge is modelled using (2.7), where E'_{res} [J] is the residual energy Δt_d seconds ago, γ determines the relationship between the current and the discharge rate (this is dependent on the device and battery technology used).

$$E_{res} = \frac{E'_{res}}{1 + \gamma I} - P\Delta t_d \tag{2.7}$$

Relaxation is also modelled in SENSE (2.8), where β_{rr} is the recovery rate, and α is the growth ratio that can eventually be reached). Relaxation only occurs if the current is drawn from the battery for at least a predetermined period of time.

$$E_{res} = \alpha E'_{res}\left(1 - e^{-\beta_{rr}(t - t_0)}\right) \tag{2.8}$$

J-Sim [151] provides models for rate discharge in both Coin-Cell and Li-NR cell types, and uses a lookup table to obtain the capacity that is seen for different discharge currents. The SenQ simulator [153] provides an advanced battery model to represent both rate discharge and relaxation through the use of an optimised version (which obtains an increase in performance of four orders of magnitude) of that proposed by Rakhmatov *et al.* [170].

The energy model must also give consideration to the consumers of energy in a node. A commonly quoted energy model is that proposed by Heinzelman *et al.* [171], which considers the energy cost incurred through transmitting a packet as being a direct function of the length of the packet l [bits]. This model provides equations for the energy required to transmit (2.9) and receive (2.10) a packet, where E_{elec} [J] is the energy required for the circuitry to transmit or receive a single bit, E_{amp} [J] is the energy required for the transmit amplifier to transmit a single bit a distance of one meter.

$$E_{tx}(l,d) = E_{elec}l + E_{amp}ld^2 \tag{2.9}$$

$$E_{rx}(l) = E_{elec}l \tag{2.10}$$

While this system permits simplistic modelling, it is largely optimistic in terms of the energy consumption. This is because it does not factor costs associated with having the transceiver enabled but not receiving or transmitting data (for example, the energy consumed through idle-listening, where the receiver is enabled but not receiving data). Additionally, this model can make the developer assume that the energy cost per packet is equal to the energy per bit multiplied by the packet length, ignoring MAC layer overheads (such as acknowledgments, headers, and control packets), collisions, and overhearing [162]. Therefore, it is generally more realistic to consider the time that the radio transceiver is enabled for, and let the communication protocol layers handle how long this has to be in order to successfully communicate. Wang *et al.* [172] propose a model for the transmit (2.11) and receive (2.12) power consumption of a radio transceiver, where ϵ is the drain efficiency of the transmit amplifier, P_t [W] is the radiated transmit power, P_{t0} [W] is the power consumed by the transmit circuitry, and P_{r0} [W] is the power consumed by the receive circuitry. Alternatively, some simulators use a look-up-table for current or power consumptions of the radio in different states (for example, receiving, idle, transmitting, and sleeping) [143].

$$P_{tx} = P_{t0} + \frac{P_t}{\epsilon} \tag{2.11}$$

$$P_{rx} = P_{r0} \tag{2.12}$$

While some simulators choose to consider the radio transceiver as the only energy consumer, this can be an unrealistic assumption to make in many situations [143]. In some scenarios where the radio transceiver spends a large proportion of its time in a sleep mode, the node's microcontroller can use a significant proportion of its available energy. Furthermore, if high-powered sensors or other peripherals are used, these can dominate consumption [9]. A number of simulators use look-up-tables in order to model the current consumption of the node's hardware. This includes considering the different power states of the radio transceiver and microcontroller [148, 151], and even sensing [164, 165]. The GTSnetS simulator [150] provides two models for calculating the sensor energy consumption that are not simply obtained from a look-up-table. The first model considers that the energy cost incurred is directly proportional to the number of bits obtained from the sensor; hence the number of bits obtained are multiplied by a predefined constant. The second model considers that the sensor's range can be traded with energy, and hence the energy per bit from the first model is scaled appropriately with distance.

The TOSSIM and AVRORA emulators have each had energy consumer models developed for them, named PowerTOSSIM [173] and AEON [174] respectively. Both of these models operate upon empirical measurements of the current consumptions of their emulated device hardware (including the radio transceiver, microcontroller, and sensors). These current consumptions are used by the model, which inspects the emulated code for hardware state changes and uses this to calculate the overall power consumption.

2.4.2.3 Modelling Sensing

Accurate sensing and environmental parameter models are important to WSN simulation as it is the environmental phenomena and the sensors that sense them that drives operation, controls traffic, and affects the behaviour of algorithms such as data aggregation and fusion [153]. Despite this, sensing is often neglected in WSN simulation, and is often limited to feeding nodes with random numbers or giving each node a static value [165].

J-Sim [146] uses the SensorSim [144] architecture for sensing, providing a sensor channel in the environment and a sensor stack in the node. The sensor channel, similar to the wireless propagation channel for communication, is used to supply data to the different sensor modules. The sensor stack in the simulated nodes contains two layers: the physical layer and the sensor layer. The physical layer detects stimuli in the sensor channel and uses a propagation model to detect the strength at which it was detected. The sensor layer then extracts and computes application-relevant data, which are passed to the node's application.

The SenQ simulator [153] models two types of sensed phenomena: diffusive channels and mobility channels. Diffusive channels model environmental parameters such as temperature, which diffuse throughout the environment. Mobility channels model properties of discrete objects (detectable by parameters such as light intensity and acoustics) though considering wave-based propagation. The SENS simulator [164] uses the environmental 'tile' concept (as used for its communication model) in order to model the sensing of wave-based objects such as acoustics. Castalia [165] claims that diffusive parameters (such as temperature) can be modelled by mobility techniques, by considering a number of discrete sources that are diffused over space and additively applied to a sensor, as shown in (2.13), where $s(t)$ is the value of the physical process at time t [s], $s_d(t)$ is the detected (or sensed) value at time t

[s], $d_n(t)$ [m] is the separation distance from the nth source at time t [s], and ρ_1 and ρ_2 are source diffusion parameters. Alternatively, Castalia allows a simpler representation whereby can a script file dictates the values that each sensor receives.

$$s_d = \sum_{all\ sources\ n} \frac{s(t)}{(\rho_1 d_n(t) + 1)^{\rho_2}} \tag{2.13}$$

Additionally, Castalia observes that *"issues like sensing device noise or bias are rarely taken into account,"* and applies to each sensor a normally distributed bias (representing difference in the manufacturing process), normally distributed noise, and the future implementation of drift. The GTSnetS simulator [150] includes the modelling of errors and accuracy in sensed data by providing three different models. First, the sensing process can be affected by a bounded random error as shown in (2.14), where s_d is the sampled value, s is the real value, and ε is a uniform random value bounded by ε_{min} and ε_{max}.

$$s_d = \left(1 + \frac{\varepsilon}{100}\right) s \ , \qquad where\ \varepsilon \sim U(\varepsilon_{min}, \varepsilon_{max}) \tag{2.14}$$

Second, a distance dependent sensing accuracy can be incorporated; where instead of placing constant bounds on the random value ε, the bounds are a function of the distance (2.15).

$$-\varepsilon_{min} = \varepsilon_{max} \propto d^{\eta} \tag{2.15}$$

Third, the error value can be an additive normally distributed error X_σ with zero mean and variance σ^2, applied as shown in (2.16).

$$s_d = s + X_\sigma \ , \qquad where\ X_\sigma \sim N(0, \sigma) \tag{2.16}$$

2.4.2.4 Modelling Timing

The modelling of timing is often overlooked in WSN simulation. WSNs are usually temporally sensitive (delivering a spatial and temporal representation of the surrounding environment) and, due to the low-cost low-power nature of nodes, clock drift results in different nodes having a different appreciation of the 'current time'. Time modelling is important for simulating algorithms such as duty-cycled MAC layers (which require temporal synchronisation), and time-sensitive algorithms requiring data to be marked with accurate time-stamps [153]. A mathematical analysis of clock drift is presented by Symmetricom [175] which identifies the frequency of an oscillator as (2.17) where f [Hz] is the oscillator frequency, f_{nom} [Hz] is the nominal frequency of the oscillator, Δf_0 [Hz] is the initial frequency error, a is the aging rate, Δf_n [Hz] is the short-term frequency instability (due to noise) and Δf_e [Hz] is the environmental frequency variance (due to parameters such as temperature).

$$f(t) = f_{nom} + \Delta f_0 + a(t - t_0) + \Delta f_n(t) + \Delta f_e(t) \tag{2.17}$$

Providing a consideration for time modelling is of importance to simulation; however, explicit timing models are virtually non-existent in WSN simulators. The SENS simulator [164] provides delay modelling whereby signal delays (affecting both the propagation of radio waves and environmental signals that are sensed) vary depending on the environment that the signal is passed through. The recently proposed SenQ simulator [153] implements both clock skew (error in the time when the clock was initially set) and clock drift (different nodes have different definitions for the length of a second), calculated using (2.17). In addition, the recently proposed Castalia simulator [165] allows different nodes to 'turn on' at different times, hence implicitly introducing initial synchronisation errors between nodes.

2.4.3 Discussion

This section has addressed the modelling and simulation themes of this research. Simulation is often used in the evaluation of WSNs, due to the ability to quickly and easily evaluate, optimise, and adjust proposed algorithms and protocols. A wide range of simulators are available, both for generic networks, and those specific to WSNs. These simulators all have strengths and weaknesses that tailor their use to different applications, platforms, simulation requirements, and design decisions. The correlation between simulation and practice depends fundamentally on the realism and range of models implemented in the simulator.

Communication models are heavily researched for traditional wired networks, wireless networks, and WSNs. The required level of detail in the communication model needs to be considered to trade simulation accuracy with development and simulation time. Due to the volume of research into energy-aware WSN algorithms, sufficient energy modelling is paramount to WSN simulation. The modelling of energy stores has improved over recent years to include battery models which consider rate discharge and relaxation effects, but 'ideal' stores and batteries are generally the only types offered. The modelling of energy consumers varies considerably between simulators but, at best, is modelled empirically from measurements of the current drain for different devices and peripherals on the node. Sources of energy harvesting are not explicitly considered in any of the WSN simulators investigated. While the operation of sensors and the sensed phenomena have a major influence on the operation of a WSN, they are often simulated with little consideration. Some simulators have modelled sensed phenomena through the implementation of a sensor channel, and the interpretation of these data through a limited sensor stack. While some recently proposed simulators have suggested the modelling of sensor accuracy and error, these models are general noise models, or specific to tracking scenarios. Recent simulators are also beginning to consider timing models, which consider the effect of clock drift on the operation of nodes. Fundamentally, careful consideration should be given to all possible areas of a sensor node, in order to adequately model those that can affect the performance of the algorithm under evaluation, and hence offset the results that are being obtained.

2.5 Summary

This chapter has established the state-of-the-art in the energy-aware, information-aware, and modelled and simulation themes of this research. Specific assumptions, limitations and capability have been identified, and these form the basis of this thesis.

Due to the constraints imposed on nodes, energy efficiency is of paramount importance in WSNs. Energy-aware algorithms should be considered at all aspects of hardware development and algorithm design. The operation of the network should be controlled in order to maximise the lifetime (and hence minimise the collective energy consumption) of the network. Such energy-aware algorithms should also consider the nodes energy resources, including the presence of any energy harvesting sources.

The primary purpose of most WSNs is to disseminate information from and around the network. To date, most networks consider all data to be of equal importance to the end-user. By considering the information content of a transmitted packet, its propagation through the network can be controlled. This is the concept behind information-aware algorithms. Furthermore, the potential for information-aware event-based dissemination techniques in the operation of autonomous nodes has been highlighted. The use of rule-based approaches to define event detection often proves to be the most accurate method.

Protocol stacks are widely used in both practical implementations and network simulation. While protocol stacks give structured consideration to different tasks of the communication stack, other node functions such as energy management, intelligent sensing, actuation and locationing are often overlooked, and bundled into an overflowing application layer. This chapter has provided an overview of MAC and NET layers, a background of which is relevant to the decisions made regarding network simulation in Chapter 5. A wide variety of simulators have been proposed for evaluating WSNs. These are distinguished from each other by factors such as scalability, code emulation, and extensibility or, more often, through the range of models that they offer. While the modelling of communication has received significant attention, aspects such as energy, sensing and timing are often neglected.

Chapter 3

IDEALS/RMR: Energy and Information Management

In Chapter 2, the importance of both information- and energy-management in WSNs was presented, and the need for energy-aware information dissemination algorithms based upon non-predictive techniques highlighted. This chapter presents IDEALS/RMR (Information manageD Energy-aware ALgorithm for Sensor networks with Rule Managed Reporting), which extends the lifetime of a WSN through a combination of information-management and energy-management. Section 3.1 gives an overview of the IDEALS/RMR system as a whole, while sections 3.2 and 3.3 provide a detailed description of the operation of the RMR and IDEALS components respectively. A set of guidelines for the configuration of IDEALS/RMR are given in section 3.4.

3.1 Overview

IDEALS/RMR is an application independent, localised technique to extend the lifetime of a WSN. IDEALS/RMR functions alongside the traditional wireless sensor node system architecture presented in Fig. 1-2 and offers significant energy savings and managed information throughput. IDEALS makes independent decisions at each node, which follow a global network-wide energy policy. The extension in the network lifetime is achieved at the possible sacrifice of low information packets as a result of a union between:

> **Information Management:** the process of discriminating between packets based upon their information content (the usefulness of communicated data). Rule Managed Reporting (RMR) determines when events occur in the sampled environment, and the information contained in subsequently generated packets (described in section 3.2).

Energy Management: the process of discriminating between packets based upon the state of a node's energy resources. IDEALS balances a packet's information content obtained from RMR with the state of the node's energy stores (which are supplemented by, and lever the benefits of, energy harvesting) to control the node's network involvement (described in detail in section 3.3).

This union results in a node's behaviour and operation being determined independently from its neighbouring nodes, as a function of its local energy state and the importance of events occurring in its sensed environment; this has not been explicitly considered before. The primary concept of IDEALS/RMR is that a node with a high energy reserve acts for the good of the network by generating packets from all locally detected events, and by participating fully in packet routing. However, a node with a near-depleted energy reserve acts selfishly, by only generating or routing packets that are deemed to have a high information content. If a node does not wish to participate in the routing of a packet, it appears invisible by not responding to its neighbours' requests. By doing this, IDEALS/RMR extends the network lifetime for important packets, through the possible loss of low importance packets. Under normal conditions, it is anticipated that a network should be designed to harvest as much as energy as it depletes, resulting in the transparent operation of IDEALS/RMR. However, for short periods, a large influx of traffic can be witnessed as the network responds to significant events (known as an event shower [54]). During these, IDEALS/RMR manages the decline of the network to maintain its usefulness and maximise information throughput.

3.2 RMR: Detecting Events and Quantifying Information

While a wide range of techniques for WSN information extraction exist that use predictive techniques (discussed in section 2.2), these are usually conducted with an aim of completely reconstructing the sampled parameter at the sink node to within a certain tolerance. An example of this is the method proposed by Kho *et al.* [176], which evaluates the information in sampled data (through mathematically evaluating the uncertainty, and hence information, contained in a sample) in order to optimally adjust the sample rate and maintain an 'acceptable' reconstruction at the sink. However, it was noted through discussion with end-users that, often, they are not concerned with obtaining a reconstruction of the data (which then has to be further processed on this sink node to detect notable events) but are instead interested in the occurrence of particular events [69]. Therefore, in this research RMR is proposed as a system providing event detection, implemented through the use of designer-created rules. The purpose of RMR is to determine when events have occurred in the sensed environment, and to ascertain the respective information content of such data. RMR is a local process occurring at each node, and requires no negotiation or knowledge about other nodes. As such, RMR (and subsequently IDEALS) could be applied to a single wireless sensor, as a method of energy and bandwidth management through the discrimination of sensed data.

The operation of RMR relies on the ability of the designer to decide upon a set of rules that describe the importance of different events that could occur during the operation of the network (guidelines for the creation of rules are given in section 3.4). For each rule created by the designer, a packet priority (PP) is assigned, which relates to how important a subsequently generated packet is. For example, in a scenario using five PPs (PP1-PP5), a high packet priority (PP1) relates to a high information packet (for example, a car tyre pressure sensor detecting a fast puncture requiring urgent action). Conversely, a low packet priority (PP5) relates to a low information packet (for example, a routine 'everything is ok' packet). Intermediate priorities PP2-PP4 are allocated to packets whose information content lies between these two extremes (for example, PP3 could relate to a slow puncture). As justified in Chapter 2, it is a feasible assumption that events with differing importance and usefulness can be identified in many application scenarios. Furthermore, high importance packets can often be used in order to identify specific 'events', while low importance packets can be used to ensure a reasonably up-to-date picture of the network is provided to the end-user when sufficient energy is available to support it.

The RMR system is implemented in a node's embedded firmware, located between the sensors and the communication stack in order to detect events (and quantify their importance) before packets are sent; the RMR system diagram is shown in Fig. 3-1. The location and integration of RMR into a sensor node's embedded firmware is discussed in greater detail in section 5.3.3 (and shown in Fig. 5-5). In Fig. 3-1, data are periodically sampled from multiple sensors (monitoring parameters in the environment), and processed into 'meaningful' values (such as the temperature in degrees Celsius, or incident light level in lux). These values are passed to the *RMR controller* to generate a packet payload (if it decides that an event has occurred) with an associated PP. RMR performs this by using *rule compliance* to ascertain if any of the rules in the *rule table* have been triggered, using the *history* where necessary. A list of triggered rules is returned to the *RMR controller*, which passes them to *payload assembly* to generate the packet payload. A packet payload is returned to the *RMR controller*, which outputs the packet payload and PP to the next stage (in this case, IDEALS). The individual components of RMR are considered below.

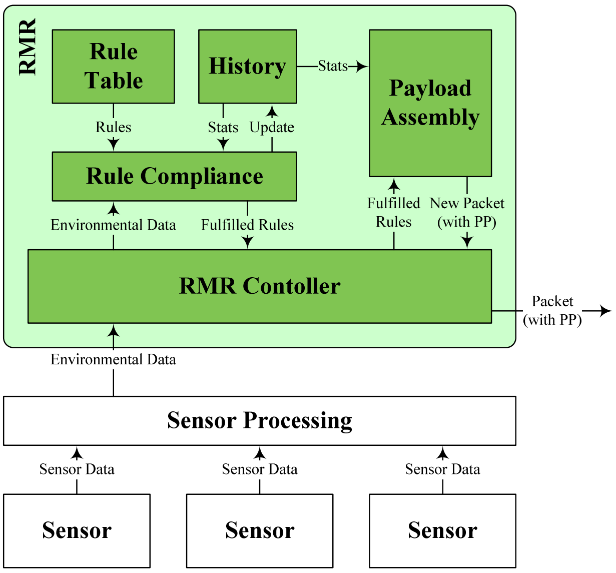


FIGURE 3-1 : The RMR system diagram

3.2.1 The Rule Table

RMR uses a rule table to describe what constitutes an event in the sensed environment, and the respective PP of a subsequently generated packet. Each rule in the rule table consists of the following fields:

Name: a descriptive and unique identifier for the rule.

Sensor: the sensor or sensors to which the rule applies.

Type: the action of the rule, selected from the list presented in this subsection (for example threshold, differential, periodic, routine, or feature).

Value: the rule trigger value, such as the differential magnitude, or time period.

Packet Priority: the packet priority (PP) that is allocated to a packet generated on fulfilment of this rule.

Parameter: additional rule-specific parameter.

Direction: specifies whether the rule applies to ‘rising’ or ‘falling’ signals (or ‘both’).

The rule table is predefined by the designer, but there is obvious scope for allowing this to be updated during network operation by broadcasting control packets through the network. The number of rules that can be defined by the designer is limited only by the available program memory (each rule is small, consisting of only the fields shown above) and execution time. Rules describe differing events that can be detected in the sensed environment, including but not limited to threshold, differential, periodic, routine and feature rules, which are individually detailed for the remainder of this subsection.

3.2.1.1 The Threshold Rule

A threshold rule is triggered whenever the current and last values sampled from a sensor are on opposite sides of the threshold (therefore the threshold has been crossed). An example of a threshold rule could be *"if the temperature rises above or drops below 60 degrees"*. Threshold breaches that occur as a result of local troughs and peaks occurring in the sensed parameter between samples will not be detected. Threshold event detection can be used to detect useful events in a very wide range of practical scenarios including the monitoring of water depth, structural strain, office temperature, and transport embankment tilt [69]. For the fields in an RMR threshold rule, *value* represents the threshold level, *parameter* controls hysteresis, and *direction* is enabled. The algorithm for threshold triggering is shown below:

```
' identifies if a threshold rule has been triggered
function threshold(rule, current_sample)
    ' check to see if hysteresis is currently locking the rule triggering
    trigger_locked = history.get_hysteresis_locked(rule)
    if trigger_locked then
        if absolute(current_sample - rule.value) > rule.parameter then
            history.set_hysteresis_locked(rule, false)
            trigger_locked = false
    ' if rule triggering is not locked, check if the rule is triggered
    rule_triggered = false
    if not trigger_locked then
        last_sample = history.get_last_sampled_value(rule.sensor)
        if last_sample = nothing then
            rule_triggered = true
        if rule.direction != "falling" then
            if (current_sample > rule.value) && (last_sample <= rule.value) then
                rule_triggered = true
        if rule.direction != "rising" then
            if (current_sample < rule.value) && (last_sample >= rule.value) then
                rule_triggered = true
        if rule_triggered then history.set_hysteresis_locked(rule, true)
    ' return whether or not the rule has been triggered
    return rule_triggered
```

Direction specifies if the rule will be triggered upon the sampled value 'rising' through the threshold, 'falling' through the threshold, or both. The direction that the sampled value is moving in is included in reported packets, aiding in the reconstruction of data at the sink (and occupying only 1 bit).

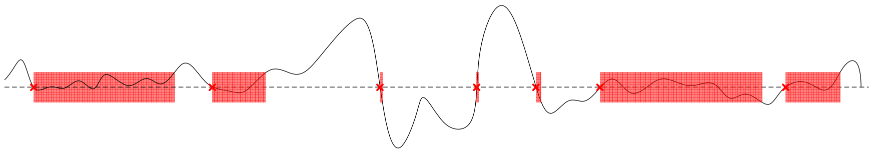


FIGURE 3-2 : Hysteresis in the threshold triggering mechanism.

Parameter is used in order to introduce hysteresis into triggering operation (representing a required displacement from the threshold), and helps to suppress spurious packet transmissions due to noise-induced signal oscillations about the threshold. If *parameter* is zero, no hysteresis will be used on threshold triggering, and hence the rule will be triggered whenever the sensed value cross *value* (subject to the state of *direction*). If *parameter* is nonzero, after the rule is triggered, the sensed value will have to displace from the threshold by more than the value of *parameter* before it can be triggered again. This is illustrated graphically in Fig. 3-2, where the dotted line represents the threshold level set by *value*, the crosses signify points where the rule is triggered, and the shaded areas (their height controlled by *parameter*) highlight where hysteresis is currently 'locking' rule triggering (hence the rule is not triggered even when the sensed data crosses the threshold). In this example, *direction* is set to 'both', hence the rule is triggered when the sensed value crosses the threshold in either direction.

3.2.1.2 The Differential Rule

A differential rule is triggered whenever the current and last values sampled from a sensor differ by more than a certain amount. An example of a differential rule could be *"if the temperature changes by more than five degrees between samples"*. Local troughs and peaks (for which the gradient exceeds that of the differential) that occur in the sensed parameter between samples will not be detected. Differential event detection is suggested to be useful in detecting long term trends such as the increase in office temperature or gradual eroding of transport embankments [69]. For the fields in a threshold rule, *value* represents the magnitude of the differential, *parameter* is unused, and *direction* is enabled. As with the threshold rule, *direction* specifies if the rule will be triggered upon the sampled value 'rising' by an amount greater than the differential, 'falling' by an amount greater than the threshold, or both. The direction that the sampled value is moving in is included in reported packets, aiding in the reconstruction of data at the sink (and occupying only 1 bit). The algorithm for differential triggering is shown below:

```
' identifies if a differential rule has been triggered
function differential(rule, current_sample)
    ' check (and return) if the rule is triggered
    last_sample = history.get_last_sampled_value(rule.sensor)
    if last_sample = nothing then
        return true
    if rule.direction != "falling" then
        if (current_sample – last_sample) >= rule.value then
            return true
    if rule.direction != "rising" then
        if (current_sample – last_sample) <= rule.value then
            return true
    return false
```

3.2.1.3 *The Periodic Rule*

A periodic rule is triggered when the rule has not triggered for a certain period of time. An example of a periodic rule could be *"every 10 minutes"*. For the fields in a periodic rule, *value* represents the required time period, and *parameter* and *direction* are unused. The algorithm for periodic triggering is shown below:

```
' identifies if a periodic rule has been triggered
function periodic(rule)
    ' check (and return) if the rule is triggered
    time_last_periodic = history.get_time_last_periodic(rule.name)
    if time_last_periodic = never then
        history.set_time_last_periodic(current_time())
        return true
    elseif absolute(time_last_periodic - current_time()) >= rule.value then
        history.set_time_last_periodic(current_time())
        return true
    else
        return false
```

Periodic rules are required in order to satisfy an end-user desire to maintain a reasonably up-to-date impression of the network. The frequency and importance of the periodic allows the energy consumed to be traded with the disseminated temporal resolution.

3.2.1.4 The Routine Rule

A routine rule is an alternative to the periodic rule, which is usually more energy and bandwidth efficient. A rule is triggered if a packet of the rule's PP or higher has not been reported for a certain period of time. An example could be *"if no packet of >PP3 has been sent for over 15 minutes"*. In a routine rule, *value* represents the required time period, and *parameter* and *direction* are unused. The algorithm for routine triggering is shown below:

```
' identifies if a routine rule has been triggered
function routine(rule)
    ' check (and return) if the rule is triggered
    time_pp_last_passed = history.get_time_last_passed(rule.sensor, rule.pp)
    if time_pp_last_passed = never then
        history.set_time_last_passed(rule, current_time())
        return true
    elseif time_diff(time_pp_last_passed, current_time()) >= rule.value then
        return true
    else
        return false
```

3.2.1.5 The Feature Rule

A feature rule is a more advanced analysis of sampled data (for example using FFTs or pattern detection), and hence is strictly a family of rules as opposed to a specific rule. Feature rules are triggered when a particular pattern or feature is detected.

3.2.2 The RMR History

The RMR history object contains information on previously occurring events and data, for use by rule compliance in ascertaining if rules have been triggered. For the majority of rules, only the 'previous sample' or 'time since' needs to be stored in memory; this requires very minimal overheads (feature rules may require additional storage and computation to detect complex patterns). The history is required to provide the following information:

> **Last Sampled Value –** maintained for use by threshold and differential rules which, in order to calculate if the threshold has been crossed or the differential exceeded, require knowledge of the last value that was sampled by each sensor on the node. Therefore, a separate record is stored for each sensor.

Hysteresis Locked – maintained for use by threshold rules, which use it to implement the hysteresis feature discussed above by storing whether or not triggering is currently locked due to hysteresis. A record of whether or not triggering is locked is stored for each threshold rule in the rule table.

Time Last Period– maintained for use by periodic rules, which need to calculate the time difference between the current time and the last-triggered time. A record of the last period time is stored for each periodic rule in the rule table.

Time Last Passed– maintained for routine rules, which require knowledge of the time a rule of PPχ or above was last triggered. A record is stored for each sensor and PP.

3.2.3 Rule Compliance

Rule compliance checks samples received from a sensor against each rule in the rule database (using the RMR history for information about previous packets and samples). The rule compliance algorithm in RMR is shown below:

```
' identifies which rules have been triggered
function rule_compliance(sensor, current_sample)
    ' inspect each rule in the rule table
    for each rule in rule_table
        ' if this rule relates to the current sample...
        if rule.sensor = sensor
            rule_triggered = false
            switch(rule.type)
                case "threshold"
                    rule_triggered = threshold(rule, reading)
                case "differential"
                    rule_triggered = differential(rule, reading)
                case "periodic"
                    rule_triggered = periodic(rule)
                case "routine"
                    rule_triggered = routine(rule)
                case "feature"
                    rule_triggered = feature(rule, reading)
            ' if the rule has been triggered, add it to the triggered rules list
            if rule_triggered then
                triggered_rules.add(rule)
    next
    ' update the history with the current sample from the sensor
    history.set_last_sampled_value(sensor, current_sample)
    ' return the list of triggered rules
    return triggered_rules
```

Any rules that are fulfilled are passed back to the RMR controller to determine the information content of the packet.

3.2.4 Payload Assembly

After rule compliance has identified which rules (if any) have been triggered, they are passed to payload assembly along with the data obtained from the sensor. The purpose of payload assembly is to create the payload to be transmitted in a packet, and to assign a PP to the packet. In the version of RMR that is presented later in this thesis, the following fields are assembled in the payload:

> **Time Of Sample –** The time at which the sample was made. Four bytes allow years between 2000 to 2136 to be represented with a one second resolution.
>
> **Sensor –** An ID number that allows the sensor type to be identified. One byte allows up to 256 different sensor types to be represented.
>
> **Sample Confidence –** A measure of the confidence in the sampled value (obtained from intelligent sensors). One byte permits 0-127% confidence with 0.5% resolution.
>
> **Signal Direction –** A single bit that is used to denote whether the sampled parameter has increased or decreased since the last sample.
>
> **Value Length –** The length in bytes of the data value in the payload (which follows this field). Seven bits allow data values of up to 128 bytes.
>
> **Value –** Contains the sample obtained from the sensor(s).

Multiple rules can often be triggered by a data sample, resulting in rule compliance passing multiple rules to payload assembly. This is handled by allocating the payload the highest PP out of all of those that have been triggered. The PP is not integrated into the payload, and is passed separately to IDEALS.

3.3 IDEALS: Balancing Energy with Information

IDEALS works alongside RMR to form the IDEALS/RMR system (shown in Fig. 3-3), which receives data and its associated PP, and ascertains whether or not transmitting the packet is worth the energy cost incurred.

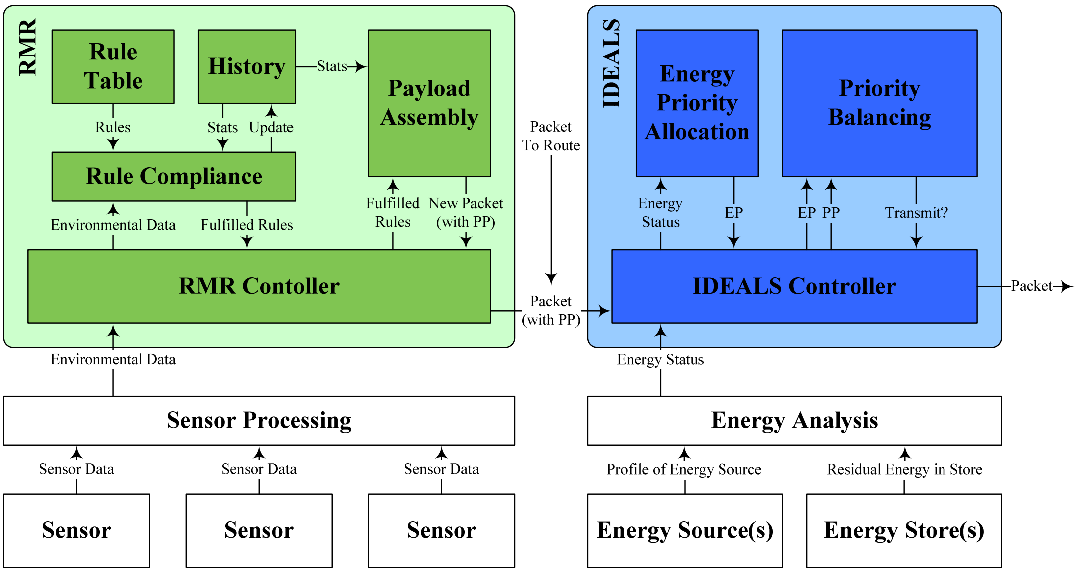


FIGURE 3-3 : The IDEALS/RMR system diagram

In addition to receiving data from RMR, the *IDEALS controller* also receives information on the energy status (which is obtained from the energy source and energy stores through energy analysis). The energy status is passed to *energy priority allocation*, which assigns the node's energy state with an energy priority (EP). The IDEALS controller passes the EP and PP to *priority balancing*, which decides whether or not the packet should be transmitted. If it should, the *IDEALS controller* passes the packet to the communication stack for transmission. Additionally, packets that require routing are passed to IDEALS from the communication stack (the packet from the source node contains its PP), and are handled by the IDEALS controller in the same way as for new packets.

3.3.1 Energy Priority Allocation

The node's energy resources are characterised by EP allocation, which assigns an EP based on the state of the energy resources. For example, in a five EP scenario (EP1–EP5), nodes with high energy reserves are allocated a high energy priority (EP5), while near depleted energy stores are allocated a low energy priority (EP1). Intermediate priorities EP2–EP4 relate to the energy levels which lie between these extremes. Additionally, a fraction of the energy is allocated to EP0, maintaining a reserve to manage the node when the energy store is heavily depleted. In this state, no sensing or communications takes place.

The relationship between the energy priority and the residual energy level is dictated by a number of energy thresholds, which map the EPs to different residual levels of the energy store. These are specified as an energy priority (depicted $EP\chi$), and its lower threshold.

There are two methods for specifying thresholds in EP allocation, using a) relative thresholds, or b) absolute thresholds. Relative thresholds (the lower relative threshold of $EP\chi$ is depicted by φ_χ) specify the threshold as a fraction of the energy store's total energy capacity. In this way, if the nodes in the network have heterogeneous energy store capacities, their thresholds are located at different absolute positions (as shown in Fig. 3-4a). This results in all nodes originally being in the same energy priority, and hence the nodes with the smaller energy stores consuming a higher fraction of their energy. Therefore, the operation of IDEALS/RMR will affect each node differently, as opposed to being a network wide policy. With relative thresholds, the EP that the node is currently assigned is given in (3.1), where Φ is the number of EPs (i.e. the maximum EP is $EP\Phi$).

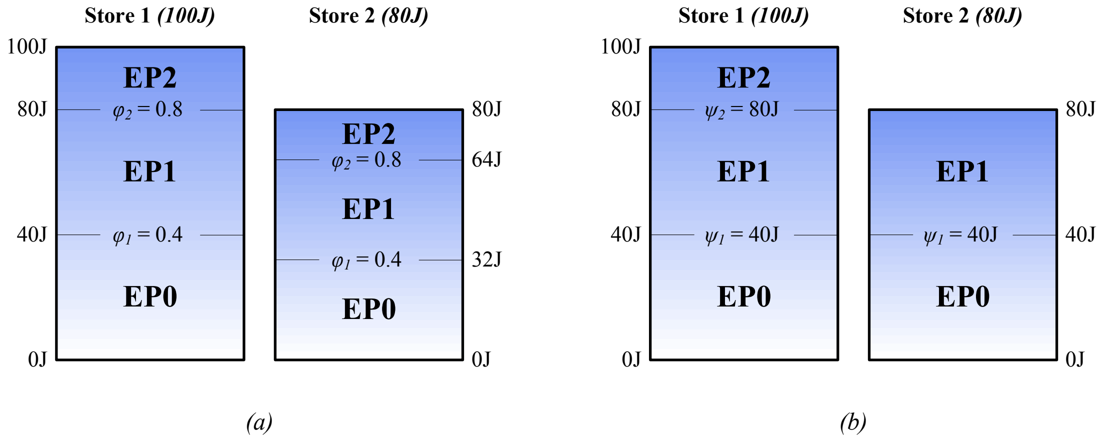


FIGURE 3-4 : Examples of the a) relative, or b) absolute energy threshold specification techniques in IDEALS.

$$\varphi_{\chi-1} \le \frac{E_{res}}{E_{max}} < \varphi_\chi \ , \qquad 0 \le \chi \le \Phi \tag{3.1}$$

Absolute thresholds (the lower absolute threshold of $EP\chi$ is depicted by ψ_χ) specify the threshold as an absolute energy value. In this way, nodes with different capacity energy stores have their thresholds located at the same absolute positions (as shown in Fig. 3-4a). This might mean however, that nodes with lower energy capacities never participate in the communication of low-importance packets. With absolute thresholds, the EP that the node is currently assigned is given by (3.2).

$$\psi_{\chi-1} \le E_{res} < \psi_\chi\big|_{\psi_\chi = E_{max}\varphi_\chi} \ , \qquad 0 \le \chi \le \Phi \tag{3.2}$$

The choice of relative or absolute thresholds is dependent on the application, network configuration, and node hardware, and hence the choice is left to the designer. It should however be consistent throughout the set of EP thresholds.

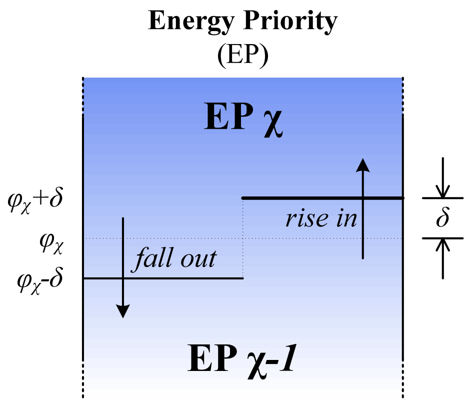


FIGURE 3-5 : Hysteresis in the IDEALS Energy Priority process.

The EP thresholds φ_χ or ψ_χ have hysteresis applied to them in order to reduce oscillations between EPs. This is implemented by specifying a separation δ which is superimposed onto the EP threshold, as shown in Fig. 3-5. For example, if the node is currently assigned EP3, in order to 'fall out' to EP2, the energy level must fall below $\varphi_\eta - \delta$. Likewise, in order to 'rise in' to EP4, the energy level must rise above $\varphi_\eta + \delta$. Note, that due to the inclusion of hysteresis, equations (3.1) and (3.2) are only valid for the condition $\delta = 0$.

3.3.2 Priority Balancing

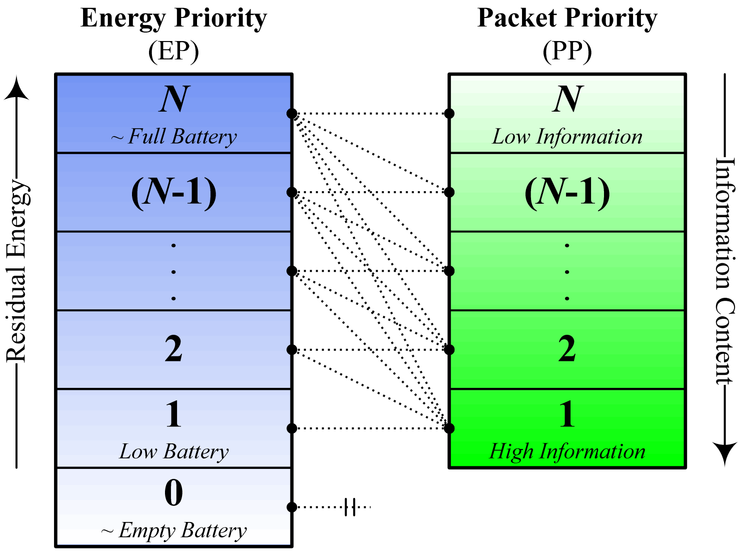


FIGURE 3-6 : Priority balancing in IDEALS

Priority balancing in IDEALS is the process of trading the node's available energy with a packet's information content in order to determine whether or not the packet should be transmitted. This is performed by comparing the packet priority (PP) obtained from RMR with the energy priority (EP) obtained from energy priority allocation, and decides whether or not the packet should be transmitted. A packet will be sent if EP ≥ PP. Therefore, as the node's energy resources deplete, packets will be selectively discarded in order of their deemed information content. The priority allocation and balancing process can be seen in Fig. 3-6. For example, if the energy store is full (EP5), packets with any information content (PP1–PP5) will be transmitted. However, if the energy store is near depleted (EP1), only packets with a high information content (PP1) will be transmitted.

IDEALS is also used during the packet forwarding process. When a sensor node receives a packet that requires forwarding, IDEALS makes the same comparison between the PP (embedded in the received packet), and the EP. If the node does not have the required resources to forward the packet, it is simply discarded. For routing protocols that require a handshaking process, the PP can be embedded in the handshake data. In this way, the receiver node can decide whether or not to respond to the request.

3.3.3 Energy Priority Sub-Networks (EPSNs)

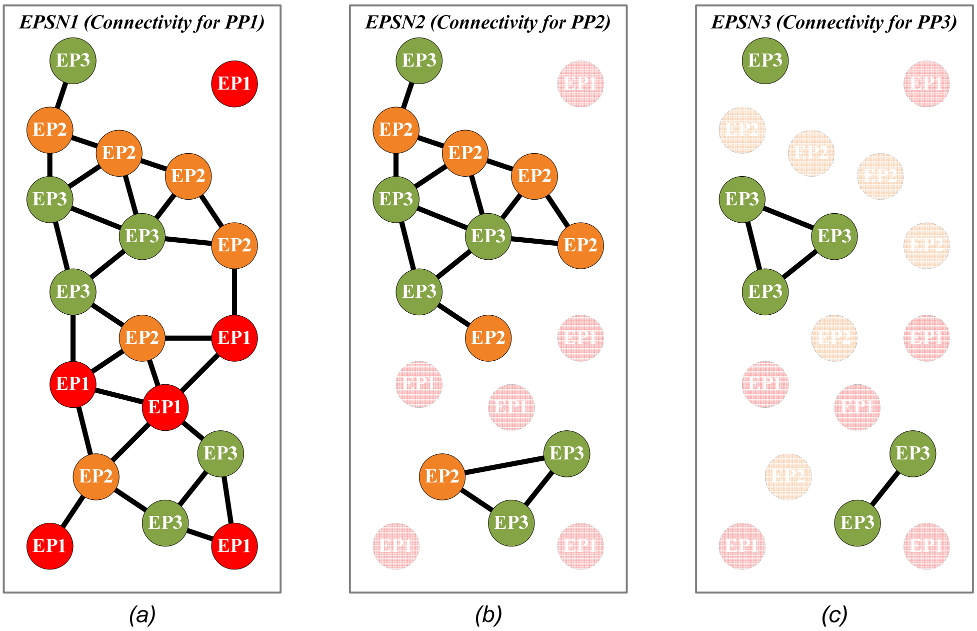

EPSN1 (Connectivity for PP1)
EPSN2 (Connectivity for PP2)
EPSN3 (Connectivity for PP3)
EP1
EP2
EP3
(a)
(b)
(c)

FIGURE 3-7 : Energy Priority Sub-Networks (EPSNs) inherently created through the use of IDEALS. The network connectivity a) for a packet of importance PP1, b) for a packet of importance PP2, and c) for a packet of importance PP3, is shown by the connecting lines.

Because IDEALS/RMR selectively discards packets during the routing process based upon the perceived information content, a number of energy priority sub-networks (EPSNs) are inherently introduced; EPSNs are temporally dynamic sub networks that provide different levels of connectivity for different PPs. For example, a packet of the highest priority will be handled by any node in the network (save those in EP0), and so packet routing can use a fully connected network. However, a packet of low priority will generally only be handled by a small subset of the nodes in the network, and so packet routing will see a considerably smaller sub network. This is depicted graphically in Fig. 3-7.

The concept of EPSNs is central to the operation of, and the energy savings and information control associated with, IDEALS/RMR. By having only a sub-set of the network available for the communication of less-important data, connectivity is maintained for important data.

3.4 Guidelines for Configuring IDEALS/RMR

This section provides generic guidelines to configure IDEALS/RMR for a monitoring application. It is the author's belief that it should be possible for an end-user or application designer to select a set of rules simply, provided that they have knowledge of the environment being sensed, and are aware of what constitutes important and useful data. It can be argued that in many applications, even if it the designer is unaware of what the sampled data is usually likely to be, they are still be able identify the events that they are interested in knowing about.

The guidelines are shown below:

1) The Application, Environment and Scenario. It is necessary that the designer has an understanding of the application, environment, and scenario. The designer should have an appreciation of what it is that they are sensing, how this is likely to vary, and what is considered to be a noticeable event in such a scenario.

2) Event Detection. The designer must identify events that they (or the end-user) consider to be useful information. Once they have been identified, their importance requires quantification through the allocation of a PP.

a) *What constitutes an event?* From an application perspective, a list of events should be collated that are of interest to the end-user, for example 'pump overheating', 'subject moving', or 'sensor failure'.

b) *How important is each event?* Once the event list has been collated, the designer needs to quantify the usefulness of the data to the end-user by allocating a PP to each event. Each event does not need to have a distinct PP, for example both 'pump overheating' and 'sensor failure' may be quantified as being of equal importance to the end-user. Through this, the designer has the option of creating any number of PPs. At this stage, the designer has the ability to set the maximum number (Φ) of PPs and EPs. Φ will at most equal the number of rules in the rule set (as more than this will include redundant EPs that serve no purpose and incur overheads). At its simplest, $\Phi = 1$, and all events are deemed to be of equal importance to the end-user (IDEALS is disabled).

c) *How can the events be detected?* Now that the events and their deemed importance have been designed, thought needs to be given to how they can be detected using the nodes onboard sensors, and RMR's threshold and differential rules. If these rules cannot be used to detect the event, it is likely that a feature rule could be used; feature rules are not discussed further in this thesis.

3) Dissemination Resolution. In addition to detecting events in the environment, the designer must also give consideration to the creation of rules that are used to provide the end-user with a 'reasonably' up-to-date representation of the network. This aids in remote signal reconstruction, and also gives the user the 'reassurance' that the network is still operational [63].

a) *What is the desired dissemination resolution?* A trade-off exists between saving energy (by transmitting few packets) and the temporal resolution of disseminated data, and should be given consideration by the designer.

b) *What routine or periodic rules should be allocated?* Periodic rules are generally redundant (instead replaced by routine rules) except for when a fixed temporal periodic is required, or where the application requires backward compatibility with a previous systems. Routine rules are more efficient and permit 'energy-aware' periodic reporting. It is suggested that a range of periodic or routine rules with different PPs are allocated; an example of this is the creation of a high importance routine rule with a period of 24 hours, and a low importance routine rule with a period of 1 hour.

c) An alternative to using periodic or routine rules to disseminate a reasonably up-to-date network representation is to use differential rules with a low differential (as suggested in [63]). This is not suggested though as it is susceptible to noise, and essentially introduces a random variable into this section of the rule set.

d) In energy harvesting systems, the energy required to generate packets from all of the rules should be less than the 'nominal' energy harvested when the node is harvesting (for example the nominal energy available from a solar cell during the day, or the nominal energy available from a vibration harvesting device while the machinery is vibrating). It is recommended that, at least, PP1 does not contain any periodic or routine rules, so that this EP is dominated by events.

4) Energy Profile. RMR is now configured as the rule set has been created. IDEALS must be configured by deciding upon the EP thresholds:

a) *Should relative or absolute thresholds be used?* If the nodes in the network have homogeneous energy stores, this decision is trivial as both types will cause identical network operation. If the nodes have heterogeneous energy stores, the two threshold types will cause differing operation, and the choice depends on application-specific parameters. Use of relative thresholds provides a simple and transferable EP allocation, and is suited to most applications and networks. Absolute thresholds allow energy depletion to be uniform amongst nodes and provides more consistent operation, especially in energy harvesting networks. However, care needs to be given by the designer to make sure that the low importance packets are able to be disseminated, avoiding the case where nodes simply waste energy through needless redundant dissemination.

b) The maximum EP ($EP\Phi$) is equal to the maximum PP ($PP\Phi$) in order for priority balancing to operate correctly.

c) *Where should EP thresholds be placed?* It is suggested that they are initially equally distributed throughout the range of the largest energy store. Following this, inspection and consideration of the rule set may dictate adjustment.

5) Feedback. Once the network is operational, the data that is being disseminated should be monitored to ensure that it is successfully detecting the desired events, and effectively balancing energy with information. If the network implementation allows the in-network update of the RMR rule table, this could be performed in order to tune the operation of the WSN.

While these guidelines are generic, they can be followed for a wide range of implementation scenarios.

3.5 Discussion and Summary

This chapter has investigated the energy- and information-aware operation themes of this thesis through the proposal of IDEALS/RMR, an application independent, localised system to control and manage the degradation of a network through the positive discrimination of packets. This is achieved through the novel combination of information management (assessed through RMR) and energy management (controlled by IDEALS). Packets that are deemed to not be important enough for delivery can be dropped at a number of stages: event generation, local priority balancing, and packet routing. While it is not essential, the operation of IDEALS is enhanced in the presence of energy harvesting, using sources such as those outlined in section 2.1.1. When nodes harvest energy, they have cyclic lifetimes whereby they come back to life after their energy stores deplete. Therefore, it is possible to deviate from a traditional assumption that a WSN has a fixed lifetime after which nodes the dying from depleted energy stores cause the network to become useless. The computational costs introduced by IDEALS/RMR are low, as the mathematical operations required are simple comparisons and increments. Furthermore, when IDEALS is used in packet routing, the PP does not need to be re-evaluated as it is embedded in the received packet. While IDEALS and RMR are presented as a single entity in this thesis, both IDEALS and RMR could be used as individual entities. For example, IDEALS could be coupled with a predictive algorithm for quantifying information. Concordantly, instead of controlling energy consumption and selective packet discarding, RMR could prioritise a queue of outgoing packets, or provide varying levels of service for packets of differing importance.

It is the author's belief that the union of information and energy management utilised by IDEALS/RMR has not been considered elsewhere, and is distinct from the algorithms presented in Chapter 2. RMR can be classified as a rule-based information dissemination algorithm, due to its ability to detect events and quantify the contained information. This is different to the predictive techniques highlighted in Chapter 2, which consider the information contained to be related to its unpredictability (disseminating data that enables better reconstruction of the sampled data) as opposed to what is useful to the end-user.

The RMR rules could be construed as pre-defined queries of a system such as TinyDB [60]; however, by definition, queries are defined and posed at runtime. RMR is an embedded, localised approach to dissemination, where a minimal set of rules are stored on the node to autonomously 'push' information to the end-user(s). Furthermore, RMR assigns events and subsequently generated packets with different priorities (the packet priority, or PP), which then allows IDEALS to adjust the node's operation based upon its energy state. While rules similar to RMR's differential, threshold and routine rules are described in TEEN [62] and APTEEN [63] (under the names of 'soft-threshold', 'hard-threshold' and 'adaptive periodic' respectively), RMR is distinct in a number of ways. In RMR, the designer is free to choose from any number of more diverse rules, and provides an infrastructure for simple future rule development. Furthermore, TEEN's hard-threshold operates by considering it as a greater than operator as opposed to it detecting the crossing of the threshold. This results in a very large number of packets being generated whenever a threshold is exceeded. The most significant difference to, and improvement over, APTEEN is the association of a packet priority with each rule, which specifies the information content (and hence importance) of subsequently generated packets. Such an information content can then be used by systems such as IDEALS to further exploit information management.

While IDEALS influences the routing decisions made by a node, it does not perform any routing itself. While it implicitly introduces a form of topology control through the introduction of EPSNs, this is a result of the nodes' energy-aware operation and is not the primary control in IDEALS. IDEALS can therefore be classified as a hybrid energy-aware and information-aware algorithm that enhances the packet routing process and topology control. ReInForm [75] and Hull *et al.* [77] use an externally determined information metric to provide varying service levels by controlling packet success and managing bandwidth. While energy management is considered in PRIMP [137] (favouring nodes with less depleted energy stores), only a binary distinction is inferred, with no concept of packet differentiation (all packets are treated equal). Dhanani *et al.* [59] propose algorithms to balance the utility of messages and energy resources (proposed after the publication of IDEALS/RMR); these are distinct from IDEALS/RMR in the methods and operation.

Naturally, it must be possible to quantify differing levels of information or usefulness for different events detected by the node; the system is not suited to applications where all data are equally important such as systems that have a binary event trigger, for example a system where light switches in a building are replaced by a network of wireless switches. However, it is the author's belief that this is possible in the majority of applications. The method of selective packet discarding used by IDEALS may seem inefficient in some scenarios. For example, a packet being discarded when it is a single hop away from the sink node (even though it has travelled along ten hops) may seem counterintuitive. In order to rectify this, it has been suggested that IDEALS is modified to not only consider the node's energy resources, but also the energy that has been invested by the network in propagating it this far. However this is essentially equivalent to 'weighting' information that is generated from nodes further away from the sink, and implicitly declaring that these data are more important. However, this is clearly not the case, and hence such a modification deviates from the design aims of IDEALS/RMR.

Chapter 6 presents the evaluation of IDEALS/RMR through the simulation of a network under controlled conditions, and a network representing a realistic scenario.

Chapter 4

Simulation Models for Wireless Sensor Networks

Simulation results are only as realistic as the models that they are built around. As investigated in Chapter 2, WSNs are often simulated using incomplete, inaccurate, or overly simplified models to represent the hardware, node operation, and surrounding environment. To ensure a close correlation between simulation and practical results, accurate models are needed; in particular, areas that directly affect the performance of the algorithm or protocol under evaluation should be modelled. Of particular importance to this thesis are the areas of energy- and information-awareness; therefore, sufficient energy modelling is of obvious importance. Information awareness and quantification depends upon the accuracy of sensed data, and so this should also receive adequate modelling. Furthermore, the accuracy of the information conveyed through the network also depends on the successful reception of packets, hence communication needs to be adequately modelled which, in turn, requires timing to be considered. Therefore, this research has developed communication, sensing, energy and timing models; these are presented individually in this chapter. In this thesis, other aspects such as location-awareness, device-level instruction accuracy and computation/memory resource utilisation are of lesser importance in obtaining comparative results and hence are not investigated.

As discussed in section 2.4.2, the level of modelling (ranging from electron-level device models to abstract system-level models) can vary considerably. Significant time and resources could be invested in modelling a device or parameter at a high level of detail, which never renders any differences in obtained results. Conversely, models that are too simplistic can result in inaccurate and unrealistic results that are not practically realisable. The ‘required’ level of detail in models can often not be foreseen; in these cases it is beneficial to include redundant detail and obtain realistic results rather than producing results that do not correlate with practical experiences. In the models presented in this chapter, a level of detail based around either empirical results or quoted errors and tolerances has been used, and it is the author's belief that this renders a sufficient and necessary level of detail to render the obtained simulation results useful. The effect of the various models can be seen in the simulation results presented in Chapter 6.

4.1 Modelling Wireless Communication

A communication model is required to both model the propagation through a wireless channel and the reception of loss of packets at neighbouring nodes; these have received considerable attention. In order to evaluate IDEALS/RMR the possibility of lost packets and collisions is important, while the perfect modelling of wireless communication is not. Hence the communication model proposed in this research aims to model propagation in a reasonably generic environment, and consider packet reception in a way that enables the consideration of interference and packet loss. Relating to the distinctions presented in section 2.4.2.1, the proposed communication model uses an empirical propagation model and communicates at the byte-level (allowing bytes to be transmitted at discrete intervals, and interference to be calculated for each) with BER calculation. Further background information on wireless communication is provided in Appendix A.

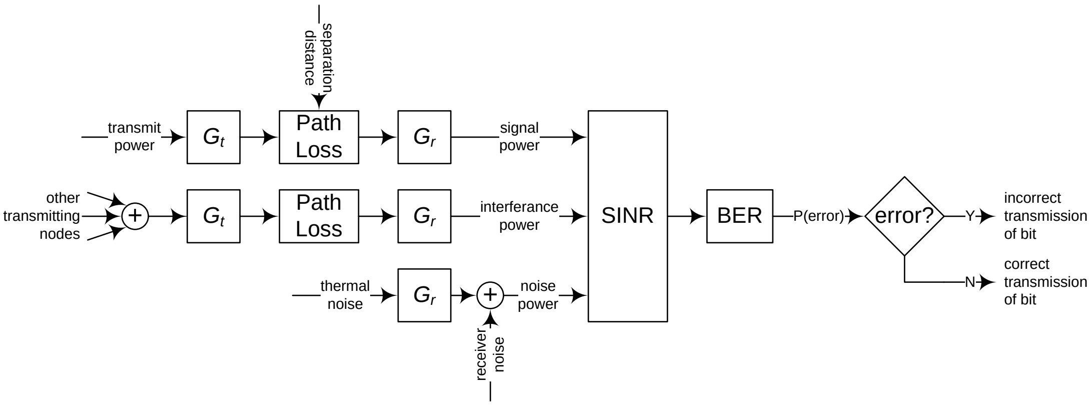


FIGURE 4-1 : The process of deciding if a byte was correctly received by a neighbouring node.

While every node in the network could be considered as a potential receiver of communicated data, this is restricted in the model to only nodes within a radius d_{max} [m] of the transmitting node in order to increase efficiency. The value of d_{max} is chosen to be large enough so that it only removes nodes that would be too far away to correctly receive packets. Each node where $d \leq d_{max}$ uses the process shown in Fig. 4-1 to determine if the byte was correctly received. In this process, the propagation model proposed by the IEEE 802.15.4 working group [177] (presented in Appendix A.1) is used to calculate the received signal strength at the node (4.1), (4.2), where P_T [dB] and P_R [dB] are the transmit and receive powers respectively and G_T [dBi] and G_R [dBi] are the transmitter and receiver antenna gains respectively.

$$P_R = \begin{cases} P_T + G_T + G_R - 50.2 & , d < 0.5 \\ P_T + G_T + G_R - 56.2 - 20\log_{10}(d) + X_{\sigma(d)} & , 0.5 \leq d \leq 4 \\ P_T + G_T + G_R - 56.2 - 20\log_{10}(d) - 0.7(d-4) + X_{\sigma(d)} & , d > 4 \end{cases} \tag{4.1}$$

$$X_{\sigma(d)} \sim N\big(0, \sigma(d)\big), \qquad where\ \sigma(d) = 2.5 + 0.0065d \tag{4.2}$$

The interference power is calculated by summing the total power received by a node that does not originate at the source node. This is an approximation as, while the BER model used (4.4) assumes that any interference is Additive White Gaussian Noise (AWGN) [84], our interference model considers ‘interfering’ transmissions from other nodes as interference. The noise power at the receiver is calculated using (4.3), where N [dB] is the noise power, k [J/K] is Boltzmann’s Constant, T [K] is the temperature, B [Hz] is the bandwidth, and NF [dB] is the receiver’s noise figure. Using the received signal power, interference power, and noise power, the SINR can be calculated.

$$N = 10^{\frac{kTB}{10}} + G_R + NF \tag{4.3}$$

The SINR is used to calculate the BER; the IEEE 802.15.4 model for the BER [84] is reasonably complex, expanding to contain fifteen exponential terms. However, it was noticed that the IEEE 802.15.1 specification [178] provides a significantly simpler model, containing only a single exponential. This model was adjusted to approximate the IEEE 802.15.4 BER, with a linear term added to reduce the error in high-noise situations where the SINR < 0.19 (4.4). The error in this approximated model is always within ±0.01 of the actual 802.15.4 model.

$$BER = \begin{cases} -1.7SINR + 0.5 & , \quad SINR < 0.19 \\ 0.9e^{-8SINR} & , \quad SINR \geq 0.19 \end{cases} \tag{4.4}$$

$$\mathrm{PER} = 1 - (1 - \mathrm{BER})^{\mathrm{l}} \tag{4.5}$$

Finally, the BER is applied to the received byte to introduce communication errors where appropriate. The ability of the proposed model to introduce communication errors can be seen in Fig. 4-2, which shows a plot of the packet error rate against separation distance for two different models. The PER is given by (4.5), where l is the length of the packet. The simple model considers all transmissions within a preset distance to be perfectly received, and all transmissions greater than this distance to always be erroneous. The BER model (described above) removes this binary acceptance method, the effect of which is furthered by the presence of path loss shadowing (represented by the cross in Fig. 4-2).

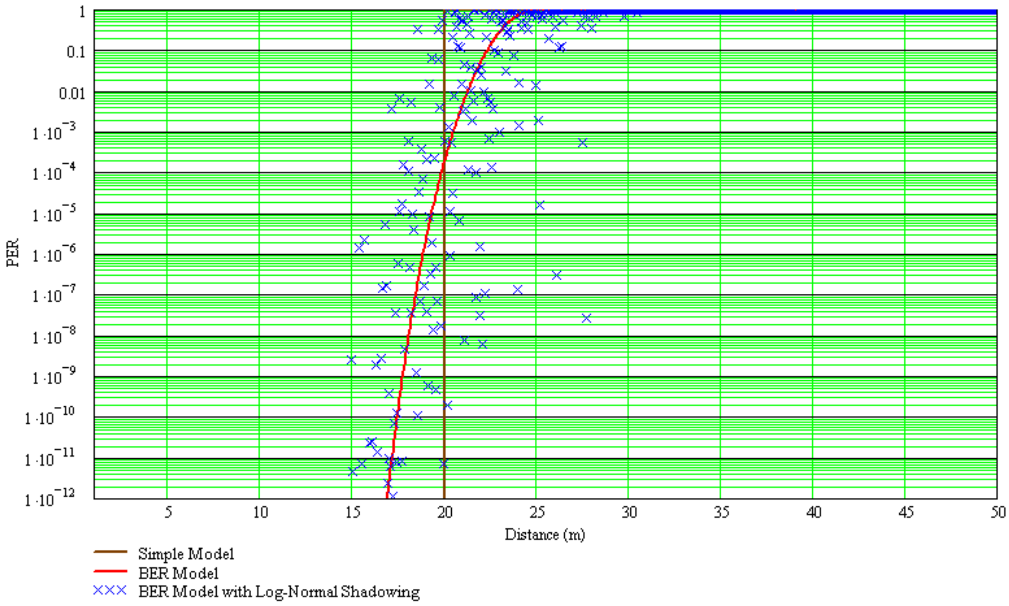


FIGURE 4-2 : Effect of considering BERs in communication as opposed to assuming a simple disc model (whereby a node separation of greater than a predefined distance always results in incorrectly received packets). The crosses show the variance in the Packet Error Rate (PER) as a result of log-normal shadowing.

This communication model presented in this research shares many similarities with the channel model of the Castalia simulator [165] that has since been released. This is encouraging, as one of the primary aims of the Castalia simulator is the provision of an advanced channel/radio model for WSNs.

4.2 Modelling Sensing

As discussed in Chapter 2, sensing models are required in order to describe how distant and distributed parameters are sensed by spatially distributed nodes, and the errors and inaccuracies that are introduced when these phenomena are subsequently sampled and converted into an electrical signal. If sensing models are not implemented in simulation, it is assumed that the node is always in possession of a ‘perfect’ understanding of the environmental phenomena under inspection, which has a direct effect on the operation of information-aware algorithms including those used for data dissemination, data fusion, and data reconstruction.

In this section, a model for the environment is presented (section 4.2.1) which models the variation of physical parameters in the sensed environment; this model is queried by sensor models at each node. It is proposed that sensor models should consider sensor hardware in order to model inaccuracies, tolerances, errors, saturation, and control. As an example, two models are presented: a digital temperature sensor (section 4.2.2), and a light sensing circuit (section 4.2.3) featuring two software selectable sensitivity ranges. In section 6.1, sensor nodes featuring the proposed temperature and light sensor models will be simulated, and the impact of their detail and granularity evaluated. However, it is important to note that these specific models are only two examples of models for sensor devices, providing an overview of the way in which they are implemented. The methods used for evaluating an individual sensor device or sensing circuit based upon quoted errors and inaccuracies can be transferred to other devices and circuits.

4.2.1 The Environment

The environment model models the variation of physical parameters in the environment, for example light, temperature or mechanical vibrations. It is a simulation element responsible for representing the variation of a physical parameter, enabling spatially distributed simulated sensor nodes to their environment. This should not be considered as the opposite of the practical reconstruction of a physical parameter based upon spatially discrete sensor readings, as this introduces a number of differing and additional challenges. When a node samples its sensors, the sensor models in turn query the environment model to obtain the value of the parameter at their location and the current time. Existing simulators (such as those presented in Chapter 2) often place heavy restrictions on the methods that can be used to model the parameter; for example by stipulating that every sensed parameter is modelled as a propagating wave. The environment model proposed in this research simply provides an interface for obtaining a value when presented with a spatial location and time, allowing the individual physical parameters to be modelled using a number of different techniques:

> **Continuous Field:** A mathematical representation of a spatially and temporally distributed continuous field $f(x, y, z, t)$ is specified. This is often a simple method of implementing a correlated variation in a scenario where actual data does not exist. This is used to represent the environmental temperature and light distributions in the simulations presented in section 6.1.

Discrete Interpolation: A continuous field is empirically sampled at spatially and temporally discrete intervals. These discrete samples are then interpolated to provide estimates of the parameter at any position and time. This is a suitable method for implementing continuous fields (such as light and temperature) with sampled data.

Source Diffusion: The physical parameter can be located to a limited number of point sources, and the parameter spatially diffuses from these. This is suitable for parameters where the predominant sources are local and easily identifiable. This is used to represent the pump temperature and vibration distributions in the simulations presented in section 6.2.

The designer is free to model the variation of a physical parameter using any of the above techniques (or any other methods provided they interface correctly with the environment model), enabling a far less restrictive sensing model.

The environment model is not used only by the sensing model. The value of physical parameters has an effect on many other models, including solar energy harvesting (dependent on the incident light level), vibration energy harvesting (dependent on the level of vibration), and timing accuracy (dependent on the ambient temperature).

4.2.2 Temperature Sensing

Temperature sensors are commonly used in WSNs, largely due to their default inclusion on most node hardware, used in applications including industrial and environmental monitoring. The purpose of the temperature sensing model is to consider the range and accuracy of sampled data, and the conversion time and power consumption. The proposed temperature sensor model is based upon the Analog Devices AD7415 digital temperature sensor [179], which communicates directly with the node's microcontroller via an I^2C interface. A summary of the relevant specifications from the datasheet is given in Table 4-1.

TABLE 4-1 : AD7415 digital temperature sensor specification (adapted from [179]).

Parameter	Value
Temperature Range	-40ºC to +125ºC
Resolution	0.25ºC (10 bits)
Accuracy	±0.85ºC at +40ºC ±1.5ºC at -40ºC to +70ºC ±2.0ºC at +40ºC to +85ºC ±3.0ºC at +40ºC to +125ºC
Typical Conversion Time	29µs
Typical Conversion Current	1.2mA

The datasheet also shows that the accuracy (at 40ºC) is normally distributed with ~99% of values falling within three standard deviations of the zero mean. For the purposes of this model, it is assumed that this distribution holds true throughout the temperature range of the device. The maximum temperature error is shown in Fig 4-3, where the points shown in Table 4-1 are modelled by a simple 2-line piecewise approximation (4.6).

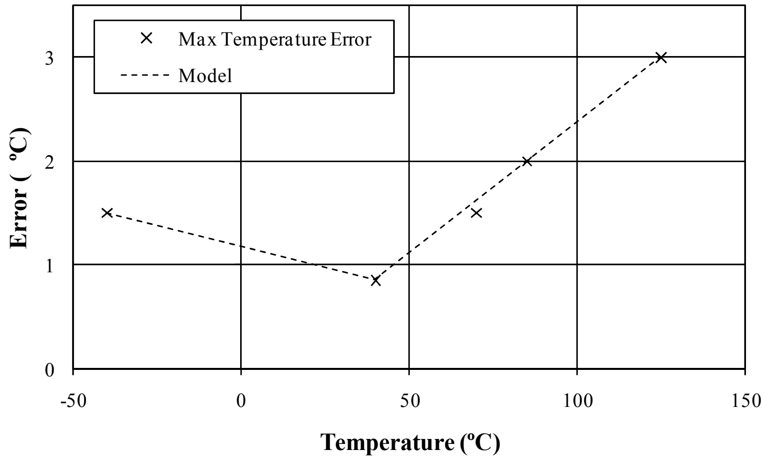


FIGURE 4-3 : The temperature dependence of the AD7415 [179].

$$\Delta T_{max}(T) = \begin{cases} 0.85 - 0.00813(T - 313) & , \quad T < 313 \\ 0.85 + 0.02529(T - 313) & , \quad T \geq 313 \end{cases} \tag{4.6}$$

In addition to being offset by a temperature dependent error, the output ADC_{out} of the digital temperature sensor is also quantised to a 0.25ºC resolution. If the temperature exceeds the operating range of the sensor, it outputs a reading of zero. The overall accuracy is given by (4.7), where X_σ is a per-device constant that is normally distributed with zero mean and standard deviation 0.333.

$$ADC_{out}(T) = \frac{\lfloor 4(T - 273 + X_\sigma \Delta T_{max}(T)) \rfloor}{4} \quad , \qquad where\ \alpha \sim N\left(0, \frac{1}{3}\right) \tag{4.7}$$

This section has proposed models considering errors and inaccuracies in the sensing process for a digital temperature sensor.

4.2.3 Light Sensing

Light sensors are used in a number of applications, including environmental monitoring, and as control sensors for MPPT in solar harvesting systems. The light sensing circuit that is considered in this section has two (software controlled) sensitivity ranges, and could be easily expanded to handle a larger number of sensitivity ranges. The purpose of this model is to handle the various sensitivity ranges, and to provide consideration for the various inaccuracies and errors that are introduced through the sampling process. The sensing circuit (shown in Fig. 4-4) uses a photodiode with an operational amplifier configured in a transimpedance configuration (which operates as a current to voltage amplifier). For good operation, an operational amplifier with a high input impedance and low input bias current is required. In this circuit, the voltage (marked V [V] in Fig. 4-4) is the inverse of the product of the input current and the feedback resistance R_F [Ω]. The feedback resistance is controlled by the *range* input (which is connected to a port on the MCU and managed by the node's embedded software). The two ranges are tuneable through the selection of R1 and R2 (which are calculated by using the graph of I_{SC} vs. lx in the photodiode's datasheet in combination with Ohm's law); the values shown in Fig. 4-4 provide a low-range of 0-1000 lx (corresponding to indoor lighting and overcast solar lighting) and a high-range of 0-100000 lx (satisfying strong direct sunlight).

A summary of the relevant specifications from the datasheets is given in Table 4-2.

TABLE 4-2 : Light sensing circuit component specifications (adapted from [27, 180-182]).

Parameter			Value
Photodiode [180]	Operating Temperature	T	-20ºC to +85ºC
	$\mathrm{I_{SC}}$ Typical	I_{SC-typ}	60μA at 100fc
	$\mathrm{I_{SC}}$ Minimum	I_{SC-min}	35μA at 100fc
	Peak Radiometric Sensitivity	S_{R-peak}	0.5 A/W
	TC $\mathrm{I_{SC}}$ Typical	I_{TC-typ}	0.12 %/ºC
	TC $\mathrm{I_{SC}}$ Maximum	I_{TC-max}	0.23 %/ºC
	Noise Equivalent Power (NEP)	NEP	$2.40 \cdot 10^{-14}$ W/√Hz
OpAmp [181]	Power Consumption	P_{OpAmp}	160μA
	Gain-Bandwidth Product	B	0.35MHz
Analogue Switch [182]	On Resistance	R_{on}	0.25Ω
	Off Resistance	R_{off}	15GΩ
Resistors	Tolerance	ΔR	±1%
ADC [27]	Conversion Speed	Δt_{conv}	20μs at l = 7bits 36μs at l = 9bits 68μs at l = 10bits 132μs at l = 12bits
	Effective Number of Bits (ENOB)	l_{ENOB}	6.5bits at l = 7bits 8.3bits at l = 9bits 10.0bits at l = 10bits 11.5bits at l = 12bits
	Reference Voltage	V_{ref}	1.25V
	Power Consumption	P_{ADC}	1.2mA

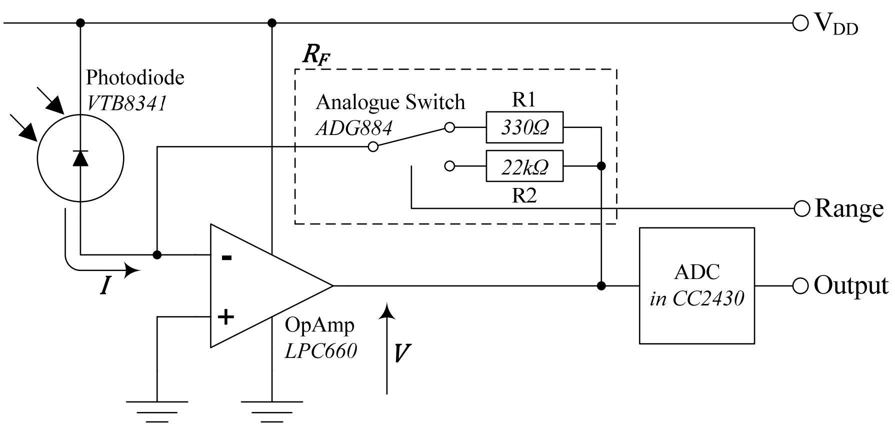


FIGURE 4-4 : Proposed light sensing circuit.

The light sensing model considers the inaccuracies and errors introduced at various locations by considering the process of sampling in Fig. 4-4. To do this, the model first ascertains the current through the photodiode at a particular light level at a particular node, following which the voltage at the output of the operational amplifier is calculated. Finally, the output of the ADC is derived, and it is this value that is subsequently passed to the simulated node. The light sensing model outputs zero if the temperature is outside of its operating range.

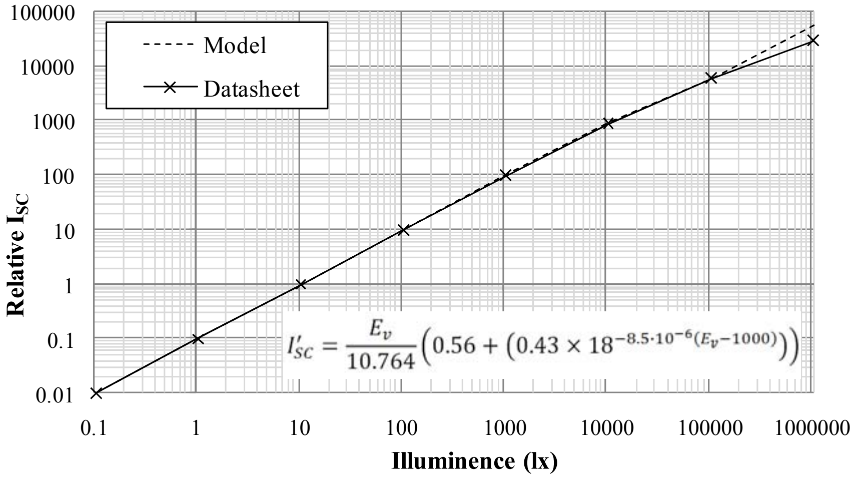


FIGURE 4-5 : Empirical values and the modelled relative short circuit current vs illuminence [180].

To ascertain the current through the photodiode, the relative short circuit current I'_{SC} (a normalised current that can be applied to different devices of the same technology) is calculated for the instantaneous illuminance level (obtained from the environment model), shown in Fig. 4-5. From this, the short circuit current is calculated (4.8), where X_σ is a per-device constant that is normally distributed with zero mean and standard deviation 0.333. The current produced by the photodiode is susceptible to noise and temperature dependence.

The noise current I_n [A] is given by (4.9), where ε is a uniformly distributed random value between zero and one.

$$I_{SC} = I'_{SC}\left(I_{SC-typ} + X_\sigma\left(I_{SC-typ} - I_{SC-min}\right)\right) \ , \qquad where\ X_\sigma \sim N\left(0, \frac{1}{3}\right) \tag{4.8}$$

$$I_n = \varepsilon S_{R-peak}\sqrt{B} \cdot NEP \ , \qquad where\ \varepsilon \sim U(0,1) \tag{4.9}$$

The photodiode's temperature dependence is modelled using the I_{TC-typ} and I_{TC-max} values shown in Table 4-2 (4.10), and the instantaneous temperature is obtained from the environment model.

$$\Delta I_{frac}(T) = (T - 298)\frac{X_\sigma\left(I_{TC-max} - I_{TC-typ}\right) + I_{TC-typ}}{100} \ , \qquad where\ X_\sigma \sim N\left(0, \frac{1}{3}\right) \tag{4.10}$$

The current at the input to the operational amplifier is hence given by (4.11).

$$I = \left(I_{SC} + I_n\right)\left(1 + \Delta I_{frac}\right) \tag{4.11}$$

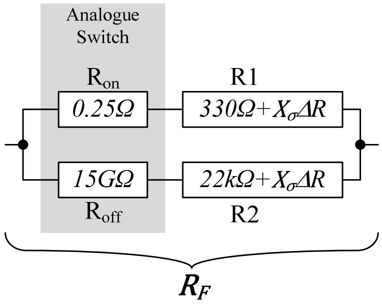


FIGURE 4-6 : The resistor network created by the on and off resistance of the analogue switch, shown when it is in high range mode.

In order to calculate the voltage at the output of the operational amplifier, the feedback resistance R_F must be calculated. This calculation should take the tolerance on the resistors' rated value into account (the tolerance is multiplied by a normally distributed value with zero mean and standard deviation of 0.333). Furthermore, the internal resistance in the analogue switch should be considered in the resistor network (as shown in Fig. 4-6).

$$V = (IR_F) + v_n \ , \qquad \text{where } v_n \sim N\left(0, \sqrt{kTBR_F}\right) \tag{4.12}$$

Once the feedback resistance R_F is known, the voltage at the output of the operation amplifier is given by (4.12) with superimposed Johnson (or thermal) noise in the resistors v_n [V].

Following the calculation of the output voltage, it is quantised by the ADC and passed to the node as a sensor sample. To account for error due to noise and distortion in the ADC's amplifier, it is quoted with an 'effective number of bits' (ENOB) which is a measure of the actual 'usable' resolution that can be obtained. This light sensing model uses the ENOB (often a non-integer value) in the quantisation process in order to model these sources of error (4.13). Additionally, the output is clipped to the range of the ADC (4.14).

$$ADC_{out} = \left| \frac{V}{2 * V_{ref}} 2^{l_{ENOB}} \right| \cdot \frac{2^{l}}{2^{l_{ENOB}}} \tag{4.13}$$

$$\frac{-2^{l}}{2} \leq ADC_{out} \leq \frac{2^{l}}{2} \tag{4.14}$$

This section has proposed models considering errors and inaccuracies in the sensing process for a light sensing circuit.

4.3 Modelling Energy

As discussed in Chapter 2, adequate energy modelling is required in order to obtain realistic energy-aware simulation results. Failure to do this can result in unrealistic lifetime predictions, the unfaithful operation of energy-aware algorithms, and incorrect analysis of energy efficiency. The energy subsystem of a node can usually be considered and modelled as having three distinct components (shown in Fig. 4-7), namely:

Energy Store: components that store energy; for example batteries or supercapacitors

Energy Consumer: components that consume energy from the energy store; for example the radio transceiver, microcontrollers, sensors, and other peripherals.

Energy Source: components that provide energy to an energy store; for example photovoltaics, vibration harvesters, and mains electricity.

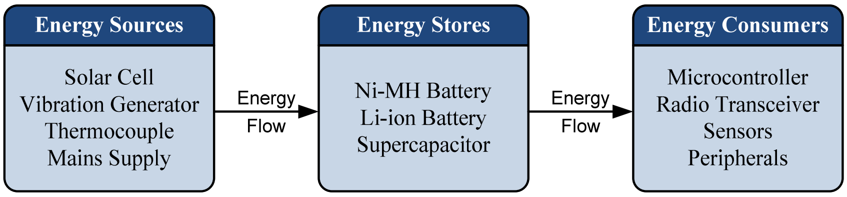


FIGURE 4-7 : The components of a node's energy subsystem, where energy is created by sources, is buffered by stores, following which it is used by consumers.

These components are considered in turn for the remainder of this subsection.

4.3.1 Energy Stores

Energy stores play a crucial role in WSNs, receiving, storing and distributing energy in the node. The energy store model is required to consider the behaviour of the energy store, with respect to its voltage profile, how much energy E_{use} [J] is removed when E_{cons} [J] is dissipated by consumers, and how much energy E_{add} [J] is added when E_{harv} [J] is provided by energy sources. As many energy-aware algorithms require a knowledge of the node's residual energy, the model should also consider the accuracy and operation of store monitoring. The energy model used in the research considers three different energy stores: the ideal store, the battery and the supercapacitor. For background information on energy stores, see Appendix B.

4.3.1.1 'Ideal' Store Model

As described in Appendix B.1, an ideal energy store (a purely theoretical device) provides a constant voltage until the energy stored within it is depleted, at which time it provides 0V. Two intuitive rules define the operation of an ideal energy store, a) it is not able to store more energy than it has capacity to store, and b) once the store is depleted, no more energy is available until some is added; hence $0 \leq E_{res} \leq E_{max}$.

An ideal store is defined by its operating voltage, its maximum capacity, and its initial level (expressed as a percentage equal to E_{res}/E_{max}). Idealistically, it can be modelled by considering that energy usage and energy addition are perfectly efficient and hence lossless processes. This is hence governed by (4.15) and (4.16), where P_{harv} [W] is the power harvested from an energy source.

$$E_{use} = E_{cons} = IV_s\Delta t = \int IV_s\,dt \tag{4.15}$$

$$E_{add} = E_{harv} = P_{harv}\Delta t = \int P_{harv}\,dt \tag{4.16}$$

A node is able to monitor the residual energy in an ideal energy store without error, and with perfect accuracy.

4.3.1.2 Battery Store Model

While often used in WSN simulation, ideal stores are theoretical devices. The most common energy store used on nodes is the battery, the effects of which are presented in Appendix

B.1. The operation of the battery store is based upon the two rules of the ideal store, satisfying the inequality $0 \leq E_{res} \leq E_{max}$. The maximum capacity of the cell is calculated using (B.1). Unlike an ideal energy store, as the energy in a battery decreases its operating voltage also decreases (this is shown in Fig. B-1).

The energy subtracted from the store is equal to the sum of the energy consumed by the circuit, energy leaked by the store since the last operation (E_{leak} [J]), and energy lost and gained through the rate capacity (E_{rate} [J]) and relaxation (E_{rlx} [J]) effects (4.17). Further information on these dynamic effects is provided in Appendix B.1. Batteries suffer from self discharge, with Ni-MH batteries losing around 13% of their capacity in the first 24 hours, and then 13% per month thereafter. This leakage energy is considered in the battery energy store model.

$$E_{use} = \gamma(E_{cons} + E_{leak} - E_{rlx}) \tag{4.17}$$

As presented in Chapter 2, simulators usually model the rate capacity effect by considering it to scale the available or remaining capacity of the battery. However, in order to provide a simple and energy-based store model, the model used in this research considers it to effect the amount of energy removed from the cell. When a cell is discharged at a low rate, the energy removed from the battery is virtually equal to the energy consumed by the node. However, when a cell is discharged at a high rate, more energy is removed from the battery than is consumed by the node. The amount of additional energy that is removed from the battery is determined by the rate factor γ, and is a function of the C-rate. The rate factor can be approximated for different battery technologies from the literature, an example of which is presented for Ni-MH and Li-ion batteries in Freeman *et al.* [183]. Using this energy-based representation of the rate capacity effect not only allows for intuitive operation, but also allows for the simplistic, albeit approximated, modelling of the relaxation effect.

The operation of the relaxation effect model is controlled by the discharge C-rate in relation to a preset relaxation threshold. If the discharge current is higher than the relaxation threshold, a proportion of the energy ‘wasted’ as a result of the rate capacity effect is put into a ‘relaxation pot’. When the discharge current drops below the relaxation threshold, the energy is given back to the cell using a temporally exponential relationship (approximating empirical data observed in the literature). If the discharge current was to again rise above the threshold, any energy in the ‘relaxation pot’ is cleared.

As summarised in Appendix B.1, the process of charging a battery is not an efficient operation, and the efficiency varies between battery technologies. The energy added to the store is equal to the energy provided by energy harvesting multiplied by the charging efficiency β_{ce} for the specific technology.

$$E_{add} = \beta_{ce} E_{harv} \tag{4.18}$$

A node is able to monitor the energy residual in its battery energy store through the use of an ADC on the node's microcontroller (modelled in the same way as the light sensing circuit in section 4.2.3). The node's microcontroller then approximates the energy remaining in the battery, using empirical measurements of the energy-voltage relationship and the instantaneous voltage sampled across the battery.

4.3.1.3 Supercapacitor Store Model

Supercapacitors (discussed in Appendix B.2) usually have low maximum operating voltages, typically 2.3V. Hence, in order to obtain an operating voltage in the range 2.0-3.6V (as required by the Texas Instruments CC2430), two supercapacitors are required in series. A supercapacitor energy store is defined by the capacitance of each capacitor (C [F]), the nominal store voltage V_{nom} [V], and the initial level (expressed as a percentage equal to E_{res}/E_{max}). A normally distributed random value is applied to the rated capacitance in line with the tolerance specified in the datasheet; for the supercapacitors investigated, this is quoted as -20/+40% [184]. As with the ideal energy store, the energy added to the store is equal to the amount of energy provided by energy harvesting; this is shown in (4.15). However, as supercapacitors exhibit significant leakage, the energy subtracted from the store is the sum of the energy consumed by circuitry and the energy leaked by the store since the last operation; this is represented by (4.19).

$$E_{use} = E_{cons} + E_{leak} \tag{4.19}$$

The energy in the supercapacitor store is given by (4.20). Therefore, the energy stored in two 4.7F supercapacitors in series with a store voltage of 3.6V is 15.228J (the tolerances on the rated capacitance relate to a considerable range of 12.2J-21.3J).

$$E_{res} = \frac{1}{4} C V_s^{\,2} \tag{4.20}$$

Fig. 4-8 shows how the voltage across the supercapacitor store drops as the stored energy decreases (shown by the solid line). As the CC2430 has a minimum operating voltage of 2.0V, the energy store cannot power the sensor node once the store voltage drops below this (shown by the shaded region). Therefore, the 'usable' stored energy is lower than the actual stored energy (shown by the dotted line).

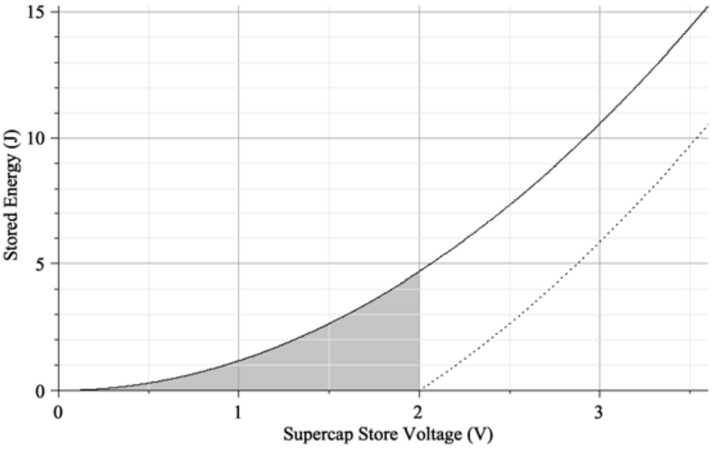


FIGURE 4-8 : The voltage across a supercapacitor store as its stored energy decreases. The solid line represents the E-V relationship, while the shaded area shows the energy that is unusable by devices requiring a 2V minimum operating voltage (such as the CC2430). The dashed line shows the stored 'usable' energy that is therefore available.

As described in Appendix B.2, supercapacitors usually suffer from high levels of leakage. In order to quantify and model this, the self discharge characteristics of 4.7F, 10F and 50F supercapacitors [184] were obtained through experimentation. By monitoring the voltage across the supercapacitors over a period of many days using a high impedance data logger, the energy leaked (and hence the leakage power) can be calculated.

By inspecting the energy reduction due to leakage (shown in Fig. 4-9), it would seem that the 50F supercapacitor provides a node with the most energy; this would also seem intuitive. However, upon inspection of the voltage reduction due to leakage (shown in Fig. 4-10), it can be seen that the voltage of the 50F supercapacitor drops at a considerably faster rate than that of the 10F supercapacitor, and hence could stop a node operating sooner. Hence, the lower leakage of the 10F supercapacitor may result in it outperforming the 50F supercapacitor. The observation that the 10F supercapacitor exhibits leakage that is orders of magnitude lower that both the 4.7F and 50F supercapacitors (an observation that has been repeated and verified for multiple devices of the same type) interestingly go against the claim made by Jiang et al. [45] that *"the larger the capacity, the greater the leakage current"*. While, in comparison, the 4.7F supercapacitor exhibits higher leakage, it is suited to energy harvesting nodes due to its smaller dimensions and faster charging ability (the greater the capacity, the more energy is required to charge it to a usable voltage). Hence, the 4.7F supercapacitor is considered for the energy model in this research.

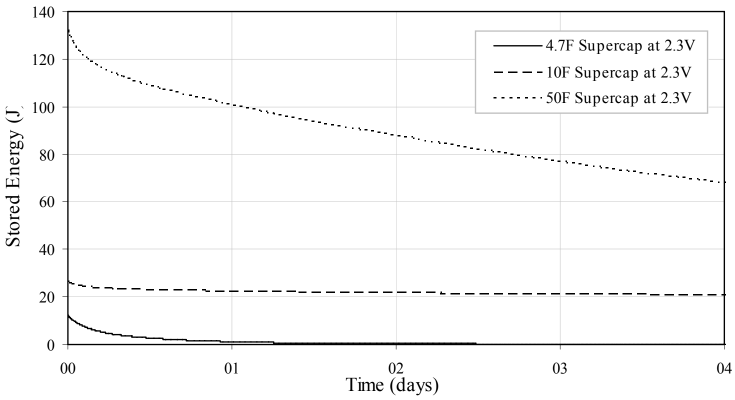


FIGURE 4-9 : The reduction in stored energy in a 4.7F, 10F and 50F supercapacitor [184] due solely to leakage (also known as self-discharge).

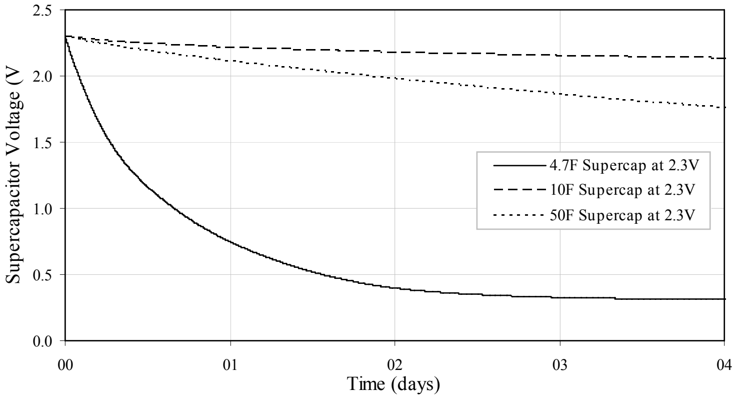


FIGURE 4-10 : The reduction in voltage of a 4.7F, 10F and 50F supercapacitor [184] due solely to leakage (also known as self-discharge).

Through experiments, it was found that the leakage of a supercapacitor is dependent on the period of time that it is held at its ‘start’ voltage; this can be seen in Fig. 4-11. In these experiments, a 4.7F supercapacitor was charged (from fully depleted) to 2.3V and held there for 1 hour, 10 hours, and 100 hours. It can be seen that the leakage power reduces as the hold time increases, and all three traces eventually converge to the same relationship. This relationship was also found for the experiments on the 10F and 50F supercapacitors. Due to the dynamic nature of WSNs, and the often intermittent and variable presence of energy harvesting, the shortest hold duration (one hour) was chosen for the leakage power model.

In order to create an empirical model for the leakage power as a function of the residual energy $P_{leak}(E_{res})$ [W], a polynomial is fitted to the ‘1hr’ curve in Fig. 4-11. Subsequently, the energy store model can use this function for the energy leaked over a period of time (4.21), where Δt [s] is the time period that has elapsed since the leakage was last calculated.

$$E_{leak}(\Delta t, E_{res}) = \Delta t P_{leak}(E_{res}) = \int P_{leak}(E_{res})\, dt \tag{4.21}$$

While the leakage power decreases with residual energy (and store voltage), once the voltage decreases past a certain point (2.0V for the CC2430), the store will be useless. However, even once the store is unable to provide a high enough voltage to power the node, leakage will still be present; therefore when energy is later provided through energy harvesting, it will first need to return the store voltage to a usable voltage before the node can operate.

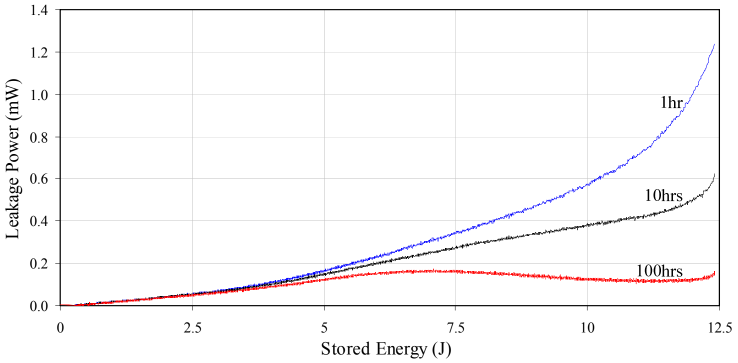


FIGURE 4-11 : A plot of the leakage power against stored energy for a 4.7F supercapacitor, having been held at 2.3V for 1 hour, 10 hours and 100 hours.

The process of monitoring the instantaneous voltage of supercapacitor energy store is modelled in the same way as for the battery energy store.

4.3.2 Energy Consumers

Energy consumers are components of a sensor node that require energy in order to operate. This section considers which consumers require modelling, and the properties of their consumer behaviour. Chapter 2 reported that the way in which different simulators, simulations, and algorithm evaluations consider energy consumption varies considerably. Some assume that a node’s only energy consumer is the radio transceiver; some use only a simplistic model of this, considering only the energy cost incurred for the transfer of each bit of data. Others provide additional consideration for the microcontroller, sensors, or additional peripherals.

In the consumer model developed as part of this research, it is accepted that a sensor node has many energy consumers which can have a significant effect on simulation results with only a minor change in the simulation conditions. The energy consumptions of all major consumers are considered, even if it is only to find that under the specific simulation conditions they have negligible effect. This is because, due to the ease of changing the simulation configuration and parameters, it is easy for a previously negligible consumer to become predominant. While this 'multiple-consumer' approach has been suggested elsewhere, this energy model is used in a higher level simulator (hence does not require machine or device-dependent code), and considers the behaviour of the energy consumers.

To model the individual consumers, their current drain is considered (the energy consumption can be subsequently calculated by multiplying by the operating voltage and integrating through time). The current drains are obtained from device datasheets, as shown in Table 4-3. While some existing consumer models use experimental results to obtain the current drawn by different consumers, it is arguable that these data have already been measured by device manufacturers in datasheets. Naturally, these data could be directly replaced with experimental results if available.

TABLE 4-3 : Power consumption of various power modes, tasks and peripherals.

Task		Current Drain
Microcontroller [27]	PM3 (full sleep)	0.3µA
	PM2 (slow wakeup sleep)	0.7µA
	PM1 (fast wakeup sleep)	190µA
	PM0 (medium activity, 32MHz XOSC)	10.5mA
Radio Transceiver [27]	Receive	16.2mA
	Transmit [172]	7.6mA + $P_t/(V_s\epsilon)$
Sensing (section 4.2)	Temperature (*section 4.2.2*)	1.2mA
	Light (*section 0*)	1.36mA
	Monitor Energy Store Voltage	1.2mA
RTC (*section* 4.4)	Standby	0.2µA
	Signal	0.4mA

In Table 4-3, the radio transceiver's transmit power consumption is based upon the drain efficiency model discussed in section 2.4.2.2. In order to derive an expression for the transmit amplifier's drain efficiency ϵ, the transmit current drain I_{tx} [A] and radiated power P_t [W] quoted for the Texas Instruments CC2430 [27] can be entered into the equation $\epsilon = P_t[mA]/V_s(I_{tx} - 7.6mA)$, and the resultant values plotted; this plot and an approximated expression for the drain efficiency is shown in Fig. 4-12.

As shown in Table 4-3, the energy consumer model considers the power consumed in different microcontroller power modes. A benefit of incorporating this is that consideration has to be given to when the node can enter the sleep states; neglecting this can result in algorithms that cannot be practically implemented.

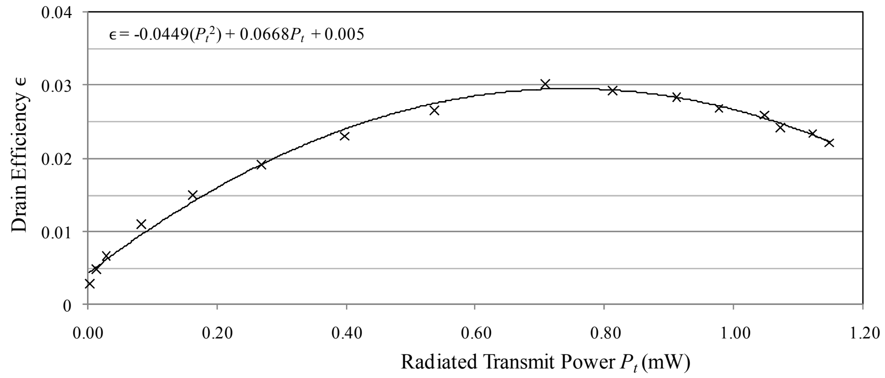


FIGURE 4-12 : The drain efficiency for the Texas Instruments CC2430 SoC across a range of radiated power levels (using data quoted in [27]).

The energy model in this research also considers the behaviour of consumers. Virtually all simulators consider energy consumers as constant-power drains, and hence are unaffected by changes in the instantaneous store voltage. As the voltage changes considerably during discharge of a supercapacitor, consideration of the discharge behaviour is paramount. Investigations performed by Weddell *et al.* [35] on a CC2430 showed that the behaviour of the node is current dominated, exhibiting a variance of only 20% across the complete operating voltage range. In contrast, the power consumed across this voltage range varies by over 100%. Hence, for the CC2430, it is more realistic to model an energy consumer as a constant-current drain as opposed to a constant-power drain; this is the relationship assumed by this energy model.

In the proposed energy model, the consumer model is notified when a peripheral is enabled or disabled, or when a device's power state is changed. Subsequently, the consumer model requests that the store model reduce its energy level (calculated by multiplying the instantaneous store voltage with the total current consumption and time period). This operation is performed both periodically (to ensure that any change in store voltage tends to zero), and when consumers are turned off (meaning that the calculated total current drain is always temporally accurate).

4.3.3 Energy Sources

In the energy model used in this research, the source model calculates the energy harvested over a period of time (which is added to the energy store) based upon the variation of physical parameters in the environment model. As energy harvesting has been virtually unconsidered in WSN simulation, algorithms that are dependent on it generally implement only very simple models. This may consider simply proportionately scaling the physical parameter in order to obtain the power output, for example assuming that the power provided by a photovoltaic cell is the illuminance multiplied by a factor F_{scale}; this is referred to hereon as the simple solar source model.

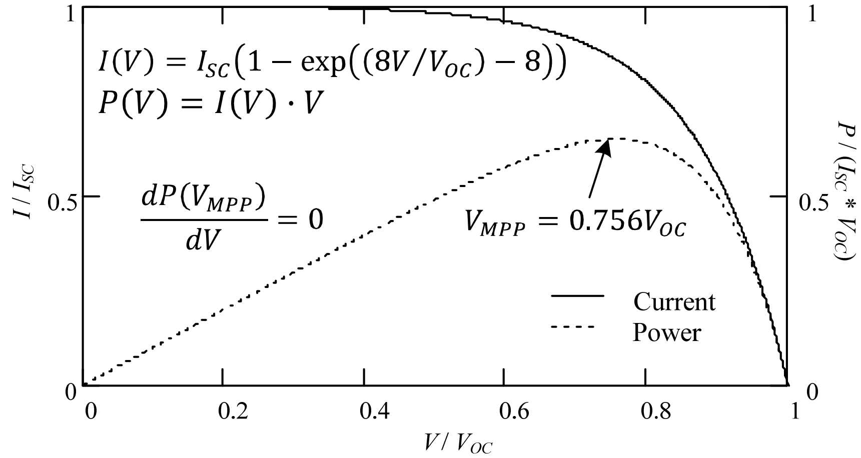


FIGURE 4-13 : The normalised I-V (and resultant P-V) characteristics of a typical photovoltaic cell, with the MPP located at the voltage $V_{MPP}V_{OC}$.

As presented in Chapter 2, the output current (and hence output power) of a photovoltaic cell is dependent on the operating voltage. If the photovoltaic is connected directly to a supercapacitor energy store via only a diode, the maximum power output will rarely be obtained. In order to better model this voltage dependence, the improved solar source model considers the I-V characteristic of a typical photovoltaic cell. Fig. 4-13 shows the normalised I-V (and the resultant P-V) characteristics of the modelled photovoltaic cell where I_{SC} [A] is the short circuit current and V_{OC} [V] is the open circuit voltage; this relationship is approximated from the properties of a Schott indoor photovoltaic cell [185]. It can be seen that the MPP is obtained when the cell voltage V [V] (in this model, the cell voltage is equal to the store voltage minus the voltage drop across the diode) is 0.756 times the photovoltaic's open circuit voltage.

TABLE 4-4 : Parameter values used in the improved solar cell model.

Parameter	Value
Reference illuminance E_V^*	100lx
Short Circuit Current at an illuminance E_V^*, I_{SC}^*	60μA
Open Circuit Voltage V_{OC}	4.4V
Scaling Factor F_{scale}	1.44

Using the I-V relationship shown in Fig. 4-13, the reference short circuit current I_{SC}^{*} [A] can be obtained, which is the short circuit current produced under an illuminance of E_V^{*} [lx]. Through the inspecting of data obtained by Weddell *et al.* [35], a scaling factor F_{scale} was found that represents how the short circuit current scales with illuminance; the operation of the model is based upon the assumption that the I-V characteristics (and hence the MPP) remain constant as the illuminance varies. Hence, the power harvested from the photovoltaic P_{harv} [W] is given by (4.22), where the values used in this model are provided in Table 4-4 (calculated based upon empirical data obtained by Weddell *et al.* [35]).

$$P_{harv}(E_v, V) = \frac{F_{scale}E_v}{E_v^{*}} I_{SC}^{*} V \left(1 - e^{\left(\frac{8V}{V_{OC}} - 8\right)}\right) \tag{4.22}$$

In addition to modelling solar energy harvesting, the energy source model considers vibration harvesting. However this is reasonably simplistic, and considers the output power to be directly proportional to the level of vibration [186]. This outputted power is constant throughout the voltage range of the store; this is a reasonable approximation to make provided that the relevant interfacing circuitry is in place.

4.3.4 Energy Models Outside of Simulation

The previous subsections have presented energy models for use in WSN simulation. Energy models are, however, also required in a node's embedded firmware in order to permit energy-aware operation. While this is not a central role of this thesis, it is briefly considered here for completeness.

An energy-aware node (and the energy aware algorithms that they execute) usually requires a knowledge of the residual energy in its store(s). As presented above, this is usually performed through obtaining a measurement of the store's voltage. In order to translate this voltage into a measure of the remaining energy, a store model is required. In the example of a dual supercapacitor store, this can be simply performed using (4.20). More complicated models are usually required for batteries, often being implemented as a piecewise approximation [35]. Additional models considering effects such as leakage and energy harvesting are required if an accurate prediction of the remaining lifetime is desired.

The voltage across a supercapacitor store usually varies considerably through its operation. The behaviour of consumers has a considerable effect on the calculation of the remaining lifetime; the effect of consumer behaviour on simulation has already been discussed in section 4.3.2. Consider the case where a dual supercapacitor store powers a node (harvesting no energy) operating between 2-3.6V, and quoted as having an average power consumption of P at 3.6V. The node could be considered as one of three types of consumer:

> **Constant-Power:** The node consumes P regardless of the instantaneous operating voltage. When the residual energy level is 50%, the node has used 50% of its lifetime.
>
> **Constant-Current**: The node consumes a constant current equal to $P/3.6$; therefore the power consumed is proportional to the operating voltage, and hence the consumption decreases with time. When the residual energy level is 50%, the node has used only 43% of its lifetime.
>
> **Constant-Resistance:** The node has a constant resistance equal to $3.6^2/P$; therefore the power consumption is proportional to the square of the operating voltage, and hence the consumption decreases with time. When the residual energy level is 50%, the node has used only 36% of its lifetime.

This shows the importance of considering, and modelling, the energy consumer behaviour in a node's firmware, if it is to provide realistic lifetime estimates. This is shown by the results obtained by Weddell *et al.* [35] (discussed in section 4.3.2), which identified the CC2430 as being a current-dominated energy consumer.

4.4 Modelling Timing

As described in Chapter 2, the operation of WSNs is often time-sensitive, requiring synchronisation between nodes and a knowledge of 'real' time. For example, in many applications it is necessary to transmit a timestamp alongside sensor data to aid in signal reconstruction. If a node's interpretation of time is inaccurate, the timestamps it appends will also be inaccurate and the reconstructed data offset. It is unlikely that this inaccuracy will be consistent between nodes, causing timestamped data at different nodes to be uncorrelated. Additionally, neighbouring nodes that utilise duty-cycled MAC protocols are usually required to wake up at the same time to communicate. If nodes do not compensate or accommodate for clock drift, wakeup periods will become offset and MAC operation fails. These effects can be overcome through the use of synchronised MAC [94] and time synchronization [187] algorithms, but should be considered in simulation to avoid potentially unseen effects rendering results unrealistic. This section presents timing models required for simulating delays, clock drift, and real time clocks (RTCs).

4.4.1.1 Delays

The process of performing operations and switching functionality (for example changing the processor or radio transceiver state, performing a computation, taking a sensor reading, or communicating) takes a period of time to undertake. While emulators consider such delays, simulators (and the algorithms and protocols they simulate) often neglect them.

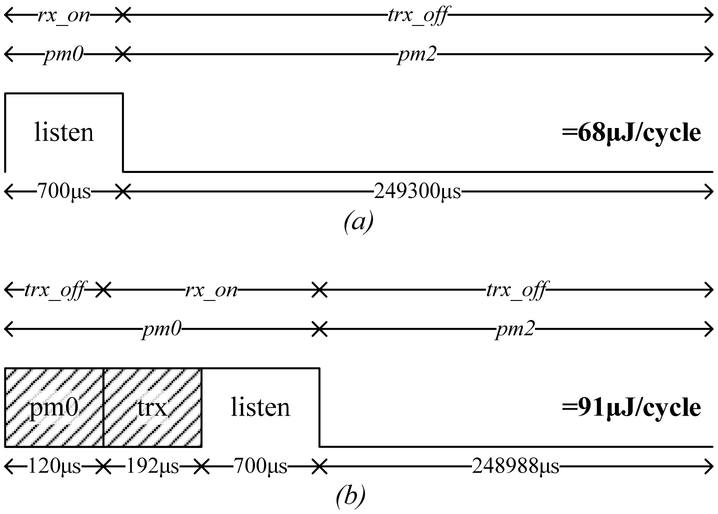


FIGURE 4-14 : The impact of considering delays in a MAC LPL cycle, where transceiver and microcontroller state change delays are a) not considered, and b) considered.

An example of the effect that delays can have on a simulation is that of transceiver state changing. The Texas Instruments CC2430 specifies a 192µs transceiver turnaround period to change from transmitting to receiving, and a 120µs period to change from sleep mode to active mode. Fig. 4-14a shows a node's MAC receive cycle (waking up every 250ms to see if a neighbour wishes to communicate) where state change delays are not considered. At the start of the cycle, the node enters active mode, briefly turns on its receiver, and then re-enters the sleep mode for the remaining time; this consumes 68µJ during the cycle. In Fig. 4-14b, the state change delays have been considered. At the start of the cycle, the node enters active mode, waits, turns on its radio receiver, waits, and then briefly listens after which time it re-enters the sleep mode; this consumes 91µJ during the cycle. This equates to an energy increase of over 33%, which has a considerable bearing on simulation results. When additional delays are considered, this difference will increase.

In the proposed timing model, the delays required for different tasks are stored in a lookup table, as shown in Table 4-5. Note that delays due to the execution of instructions on the microcontroller are not considered, due primarily to the difficulty in implementing this in simulation. However, this is approximated by introducing a random delay for Service Access Point (SAP) communication (communication between adjacent layers of a protocol stack), which aids in the provision of execution delays.

TABLE 4-5 : Delays required for various power modes, tasks and peripherals.

Task	Time Delay Δt (μs)
CPU/Transceiver PM3 Wakeup [27]	120
CPU/Transceiver PM2 Wakeup [27]	120
CPU/Transceiver PM1 Wakeup [27]	4
Transceiver State Change [27]	192
Sense Temperature (*section 4.2.2*)	29
Sense Light (*section 0*)	*dependent on ADC resolution*; $20 \leq \Delta t \leq 132$
Monitor Store Voltage (*section 4.3.1*)	*dependent on ADC resolution*; $20 \leq \Delta t \leq 132$
RTC Signal (*section 4.4.1.3*)	150
SAP Communication	*uniform random value*; $1 \leq \Delta t \leq 5$

4.4.1.2 Clock Drift

In practice the clocks used in WSNs are rarely accurate, which adds error to the duration of delays controlled by internal timers. The Texas Instruments CC2430 has a low-power 32kHz RC oscillator that is used as a clock while the device is in a sleep mode. WSNs occupy sleep modes for the majority of their operation, hence the accuracy of this oscillator is the most significant in controlling the node's appreciation of 'real' time. The Texas Instruments CC2430 datasheet [27] quotes the RC oscillator as having a calibrated accuracy of ±0.2%, equating to a loss or gain of around one day per year (further extenuated by a 3%/V and 0.4%/ ºC drift).

$$\Delta t = \Delta t_{exp} \times \left(1 + \frac{X_\sigma}{\tau}\right) \;, \qquad where\; X_\sigma \sim N\left(0, \frac{1}{3}\right) \tag{4.23}$$

The proposed timing model accounts for clock drift by considering it as a constant error at each node (the magnitude of the error is different for each node, but constant for the duration of the network). Whenever a time delay is requested (either because hardware is switching or transitioning, or because a timer has been set), the timing model applies an error to the delay that is actually performed (4.23), where Δt is the time period that has actually elapsed, Δt_{exp} is the delay that is requested or expected, X_σ is a normally distributed random value that is fixed on run-time for each node (with mean 0 and standard deviation 0.333), and τ is the timing error in the node's crystal oscillator (expressed as a '1 in τ' value). The use of a normally distributed value with a standard deviation of 0.333 results in 99.7% of nodes having clock drift within the rated tolerance τ. The CC2430 quotes an accuracy of ±40ppm for the crystal oscillator, which equates to a τ of 25000.

4.4.1.3 Real Time Clock

In addition to the onboard clock which provides the facility for implementing timers on a microprocessor, some nodes also contain a real time clocks (RTC) to keep track of the 'absolute' time; this is used for implementing functions such as the time stamping of sensor data. The proposed timing model includes the modelling of a RTC, while the model is generic, it has been implemented considering the properties of the Dallas/Maxim DS1302 RTC [188]. When the time is requested from the simulated RTC, the model looks at the actual time (simulation time) elapsed since the RTC was last queried, applies an offset to this, and then adds it onto the last queried time. This is shown in (4.24), where $\Delta t_{frac}(T)$ is the additional error that is introduced at an operating temperature T; the effect of which is specified in the datasheet, and shown in (4.25). The datasheet also specifies a crystal tolerance of ±20ppm, which equates to a β of 50000.

$$\Delta t = \Delta t_{exp} \times \left(1 + \frac{X_\sigma}{\tau} + \Delta t_{frac}(T)\right) \;, \qquad where\ X_\sigma \sim N\left(0, \frac{1}{3}\right) \tag{4.24}$$

$$\Delta t_{frac}(T) = \frac{-0.043(T-25)^2}{1 \cdot 10^6} \tag{4.25}$$

This method of discretely updating the accuracy (which considers only the temperature at the time that the time is requested) is valid provided that the temperature variation is at a low frequency compared with that of the RTC sampling rate.

4.5 Discussion and Summary

This chapter has presented research relevant to the modelling and simulation theme of this thesis. Energy, sensing, communication and timing models have been highlighted as key areas that require adequate modelling in order to effectively simulate energy- and information- managed algorithms such as IDEALS/RMR. A byte-level empirical communication model has been presented, which calculates packet reception through the modelling of BERs. Sensing models have been proposed for a digital temperature sensor and a light sensing circuit. Energy models have been presented for energy stores (with particular attention to supercapacitors), energy consumers, and energy sources. Timing models have been proposed to consider the accuracy of oscillators, and the temperature dependence of RTCs. Furthermore, the importance of considering various time delays in WSN simulation has been expressed.

While models specific to particular hardware have been proposed in this section (such as specific supercapacitors, microcontrollers, and sensors), the methods used in this section for sensing and timing models (which evaluate an individual device or circuit based upon quoted errors and inaccuracies) can be transferred to other devices and circuits. This also applies to the developed empirical models (for energy sources, stores and consumers), whereby the ‘black box’ approach to considering sensor nodes and energy stores can be easily transferred to alternative hardware platforms and devices.

The models presented in this section are used in the developed simulator, WSNsim (discussed in Chapter 5), which is used to obtain the simulation results presented in Chapter 6. The results presented in Chapter 6 evaluate and highlight the impact of the various models developed in this section.

Chapter 5

WSNsim: A Simulator for Wireless Sensor Networks

A number of simulators have been proposed and developed for the evaluation of WSNs, an overview of which was given in Chapter 2. As each of these has strengths and weaknesses in differing areas, the inclusion of IDEALS/RMR, environmental and physical models (in particular energy harvesting), and support for the required results is a nontrivial task requiring considerable modifications and extensions. Hence, it was decided that while the use of an existing simulator could be used, this could not be appropriately accomplished in an effective and time efficient manner. Therefore, WSNsim (Wireless Sensor Network Simulator) was developed to debug, evaluate and improve the algorithms developed as part of this research. WSNsim is an in-house object orientated discrete-event simulator for WSNs, developed by the author using Microsoft Visual Studio .net 2005. While the simulator is not the primary subject of this thesis, it is outlined here in order to show how the environmental and physical models are interacted with, how energy and information are handled, and to put the obtained results into context. As it has been developed as an in-house tool to evaluate only this research, providing it as an open source simulator to the wider research community would be premature as it was not designed to be general purpose, has limited documentation, and does not currently have the personnel required to support it. This will however be considered in the future, as it has received reasonable interest.

5.1 Aims and Requirements

At the outset of this research, an initial version of WSNsim was conceived in order to validate the concepts of the research. However, the use and evaluation of this highlighted a number of key issues that were fixed and improved in the second major version of the software. The initial version was a procedural discrete event simulator, structured and operated with the GUI as the central control element. The GUI was utilised by the user to define and manage the entire network simulation including node and event placement, energy resources, harvesting environments, and was also used to control simulation execution. Furthermore, the simulation of each node's embedded software was not implemented in a structured manner, resulting in the extensibility and customisation being inherently limited. The simulator contained minimal environmental modelling in terms of communication, sensing, energy and timing; in addition, most areas of the simulator used arbitrary units, with energies specified relative to the capacity of the energy store, distances specified relative to the dimensions of the simulated environment, and times specified as 'timesteps' (a unit of discrete time equivalent to a MAC or sensing cycle). Following the initial validation of IDEALS/RMR, it was imperative to revise WSNsim in order to obtain more relevant results with a simulation environment that could be easily and adequately used to obtain them. The primary aims of the revised (and current) version of WSNsim are:

Design & Structure: WSNsim is an object orientated discrete event simulator with a rational structure, thus increasing extensibility. As opposed to being GUI-centric, a simulation controller manages, coordinates and controls execution using a discrete packet scheduler to pass data between objects. WSNsim uses standard units, for example distances are expressed in meters, energies in Joules, and times in seconds (although the discrete packet scheduler has a temporal resolution of 1μs).

Simulated Node Structure: The simulated nodes execute code of a similar structure in the simulator as they do in a real system (for example implementing communication functionality in a protocol stack formed of distinct layers); this simplifies the process of comparing experimental and simulation results. WSN is able to accommodate heterogeneous nodes which operate using different protocols and hardware resources.

Environmental Models: WSNsim is built around the various environmental models, including all of those researched and presented in Chapter 4, covering communication, sensing, energy, and timing models.

Network Specification: The network topology, simulation configuration, GUI selection, and node configuration is specified via a network setup file allowing for an emphasis on simulation as opposed to network design. This also enables the running of concurrent automated simulations.

GUI and Logging: A wide range of different non-interactive GUIs can be enabled in WSNsim, each showing different aspects of the network. Additionally, the logging system allows for easier extraction of key data.

5.2 Structure, Operation and Functionality

This section presents the organisational and operational structure of WSNsim, the methods by which it is configured, and the various visualisation and logging facilities that are made available to the user for evaluating a simulation's performance.

5.2.1 Simulator Structure

The structure of WSNsim is shown in Fig. 5-1, which highlights the level of importance that is given to environmental modelling and the simulated node structure. Operation is coordinated by a central controller, which uses a packet scheduler to manage the discrete event simulation. The packet scheduler operates with a resolution of one microsecond and, below this, allows up to one million different packets to be queued for the same microsecond. In this context, the term packet describes not only communication packets transmitted between simulated nodes, but also any data transferred between the layers in the simulated node's embedded software, and the various objects of the simulator; any form of time delayed process is implemented via the packet scheduler.

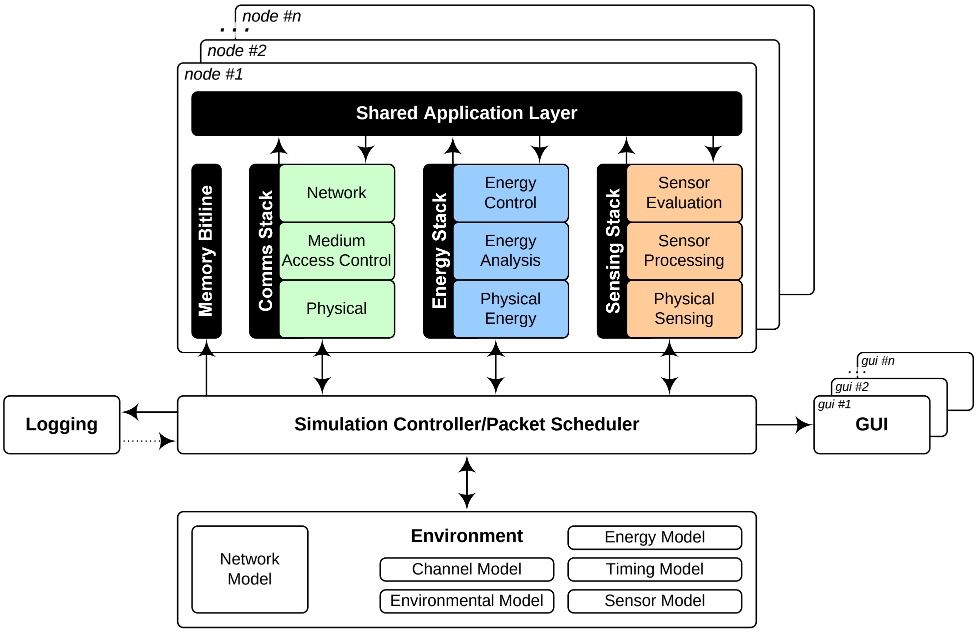


FIGURE 5-1 : The structure of WSNsim.

The packet scheduler also manages which objects can communicate with which other objects; for example, the physical layer of the communication stack is not allowed to communicate directly with the network layer. The network controller uses the node collection (an array of node objects depicting every node in the network), logging objects, GUI objects, and the environment object in order to perform network simulation.

5.2.2 The Network Description File

When a simulation begins, the network model imports a network description file (NDF) that contains information regarding the simulation (including the simulation duration and environment dimensions), nodes, GUIs, and logging configurations. After importing the NDF, the network model makes it easily available to the simulation controller on demand. The node parameters configurable for each node in the NDF include the MAC address, stack layer selection (each node can have one of three different implementations of each layer of the embedded software stack), position (this can be scripted in the NDF to change through time), power-on time (this can also be scripted in the NDF, including the ability to specify a random time within a certain range), and hardware properties for attached energy stores, sources, antennas and sensors. Certain parameters related to the hardware attached to the node (such as the presence of energy sources and sensors) are indicated to the simulated embedded software of each node using a 'memory bitline' object on the node; this object represents a set of jumpers on the node signifying the hardware setup of each.

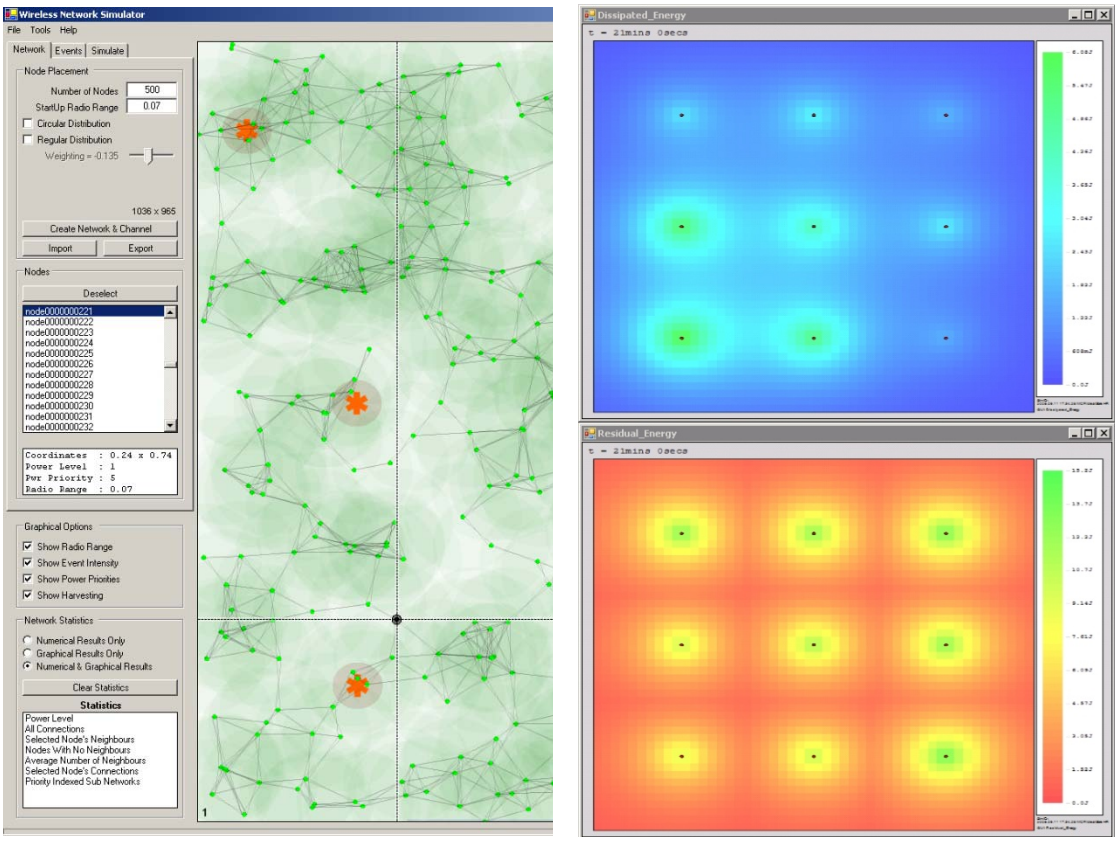
TABLE 5-1 : The effect of enabling and disabling IDEALS, RMR and energy harvesting on the operation of the node.

IDEALS	RMR	Harvesting	The Effect on Node Operation
×	×	×	Periodic sampling without harvesting
		✓	Periodic sampling with harvesting
	✓	×	Event-based sampling without harvesting
		✓	Event-based sampling with harvesting
✓		×	Event-based sampling with IDEALS, but no harvesting
		✓	Event-based sampling with IDEALS and harvesting

In order to effectively evaluate the system, IDEALS, RMR, and energy harvesting need to be independently enabled and disabled; these are also specified in the network description file. The operation of the node as a result of these various settings is shown in Table 5-1.

5.2.3 Visualisation and Logging

As mentioned above, the operation of the initial version of WSNsim was coordinated around an interactive GUI element (shown in Fig. 5-2a) providing network setup, configuration and simulation control. However, through moving coordination to a separate controller element and defining the network configuration in a separate NDF, the revised version of WSNsim exhibits GUI elements (shown in Fig. 5-2b) with a sole purpose of conveying the operation of the network. The types of GUIs available include communications (radio energy in the channel), sensing (variation of the environmental parameter or sensor output), and energy (residual, harvested, or dissipated). Any number of GUI elements can be displayed during simulation, and are configured in the NDF file to describe their type, parameters, dimensions, resolution, colours, update frequencies, and whether or not the GUI should also be saved to a video file. In addition to the 'plan view' of the network and the visualisation of the parameter in question, each element also contains an indication of current simulation time, a unique identifier, and a scale relating colours to practical values. This allows a constructive understanding of the state of the network during or after execution.



Number of Nodes 500
StartUp Radio Range 0.07
☐ Circular Distribution
☐ Regular Distribution
Weighting = -0.135
1036 x 965

node0000000221
node0000000222
node0000000223
node0000000224
node0000000225
node0000000226
node0000000227
node0000000228
node0000000229
node0000000230
node0000000231
node0000000232

```
Coordinates  : 0.24 x 0.74
Power Level  : 1
Pwr Priority : 5
Radio Range  : 0.07
```

☑ Show Radio Range
☑ Show Event Intensity
☑ Show Power Priorities
☑ Show Harvesting

○ Numerical Results Only
○ Graphical Results Only
● Numerical & Graphical Results

Power Level
All Connections
Selected Node's Neighbours
Nodes With No Neighbours
Average Number of Neighbours
Selected Node's Connections
Priority Indexed Sub Networks

t = 21mins 0secs

t = 21mins 0secs

(a) *(b)*

FIGURE 5-2 : Screenshots from the a) initial, and b) the current version of WSNsim.

In WSNsim, logging (recording statistics during simulation execution) to comma separated value (CSV) files provides the fundamental means for evaluating the operation of the network; the level of logging that is performed is configured in the NDF. A number of different log files are created to record a wide range of different occurrences, including the activity of energy stores, sources and consumers, packet transmissions, sensor sampling, and IDEALS/RMR decisions. After the simulation has completed, a separate application is used to extract and compute key data from the log files, and is subsequently displayed using a graphics package such as SigmaPlot. A selection of the results that are obtainable can be seen in Chapter 6.

One metric of use in evaluating the performance of IDEALS/RMR is 'information throughput' (a term coined as part of this research). As this research considers the information content of data to be based upon a user's perception of the importance of the data as opposed to the information theoretical value, the quantification of 'information throughput' is a non-trivial task. However, the user quantifies the importance of data through the selection of PPs, where a large PP relates to low information content and a small PP relates to high information content. Therefore, a measure of the information contained in a packet is the 'inverse' of the PP (5.1).

$$Information = \Phi - \chi \tag{5.1}$$

Information can be measured at a variety of stages in the network. Firstly, the 'information generated' can be calculated by considering all packets generated by RMR (packets that are generated, but not necessarily transmitted). Secondly, the 'information transmitted' can be calculated by considering all packets that a node transmits. Finally, the 'information received' is calculated by considering all packets that are received at the sink node.

5.2.4 The Environmental Models

The environment object in WSNsim contains the environmental and physical models for each node. For efficiency and coding simplicity, each node's modelled hardware (for example its energy store or sensors) are contained within the environment object as opposed to each node object. The models proposed in Chapter 4 and implemented in the environment object are stochastic, and require the availability of normally distributed random variables; these are obtained using the Box-Muller method [189].

5.3 An Architecture for a Node's Embedded Software

As specified in section 5.1, an aim of WSNsim is to provide a structured approach to algorithm design and network simulation. Chapter 2 illustrated how protocol stacks form the basis of the majority of communication protocols and algorithms; the distinctions that were conceived by the OSI-BRM still dominate design, with algorithms developed for specific layers of the protocol stack such as the network layer (routing algorithms). While protocol stacks formally structure the functionality of a communication subsystem through the use of layers, other aspects of a sensor node (and the interfaces between them) are left largely unstructured. This is not ideal, as sensor processing and energy management are arguably as important as communication in the operation of a sensor node. IDEALS/RMR requires aspects of both energy-management and information-management to operate; hence a method of structuring these additional interfaces in a node's embedded software is required.

This section proposes an architecture for specifying and structuring multiple interfaces in a node's embedded software, a concept that is fundamental to the structure and operation of WSNsim. This is performed by allowing sensor nodes to incorporate a number of distinct stacks. Each of these stacks which implement distinct functionality such as communication, sensing, energy management, actuation or locationing, is derived from a basic template stack. The individual stacks are combined into a 'unified' stack through a shared application layer which is the highest layer of each stack. As discussed in Chapter 2, a range of variations to the communication stack (including cross-layer optimisations and additional 'planes' for energy management), stacks specifying functionality such as locationing [190] and sensing [29], and simulation structures to specify energy channels and sensing channels have been reported.

The use of this architecture translates the benefits of a communication stack across all major node tasks, including structuring functionality, permitting efficient code reuse, and promoting standardised, interchangeable protocol design.

5.3.1 The Basic Template Stack

Stacks complying with the proposed architecture are derived from the basic template stack shown in Fig. 5-3. The layer boundaries are placed with the aim of accommodating a wide range of implementations, while retaining definite limits on the required functionality of each layer to permit the interchanging of individual layers. The functions of the layers are:

Interface Layer: Interfaces directly with the appropriate hardware, and hides the complexity of the circuitry from the higher layers. For communications, this layer represents the physical layer of the OSI-BRM.

Medium Layer: Provides low-level processing such as error checking, and masks the complexity of the medium (such as the sensed phenomenon or wireless channel). For communications, this layer represents the data link layer of the OSI-BRM.

Management Layer: Provides high-level processing (such as event detection, network formation and maintenance), and masks the complexity of object groups (for example a network of nodes, number of energy sources, or collection of sensors) from the higher layers. For communications, this layer represents the network, session, transport, and presentation layers of the OSI-BRM (the majority of which are often neglected in practice).

Shared Application Layer: Contains cooperative functionality that passes data between stacks. For communications, this layer represents the application layer of the OSI-BRM.

The decision to use three layers (logically placed for interfacing, processing, and controlling) plus a shared application layer is a result of both the generic structure that this logic compartmentalisation provides, and also the observation of modern communication protocols such as ZigBee [83].

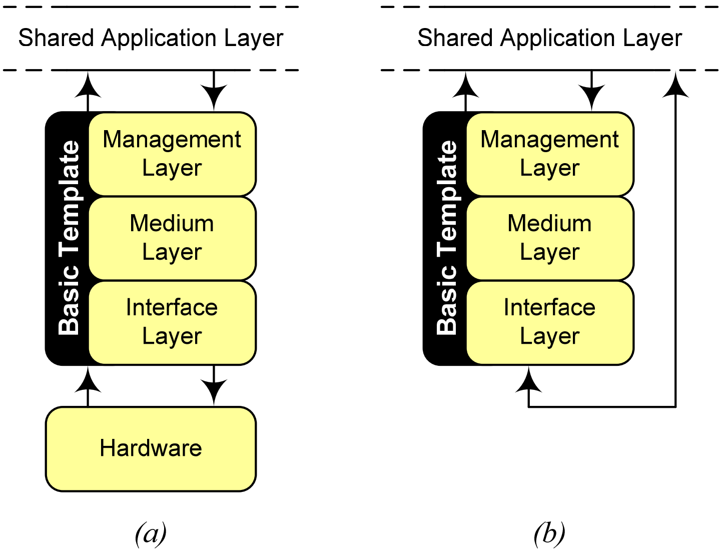


FIGURE 5-3 : The proposed architecture's basic template stack for a) hardware interfaces, and b) software interfaces.

It is envisaged that the proposed architecture is not restricted to only structuring software/hardware interfaces. Software/software interfaces can also be structured using the basic template stack shown in Fig. 5-3b. In this, the raw data for processing is passed via the shared application layer from another stack, processed by the stack, and returned to the shared application layer. The investigation of software stacks are outside the scope of this research, but are an interesting area of future research.

5.3.2 Creating a 'Unified' Stack

Once a number of individual stacks (each structuring a different node function or interface) have been derived from the basic template stack, a 'unified' stack is formed by combining these distinct stacks via the shared application layer. Stacks use the shared application layer to communicate and pass data between each other. The structure of a 'unified' stack specifying communications, energy management, and intelligent sensing is shown in Fig. 5-4; this is the unified stack implemented in WSNsim (see Fig. 5-1). The functionality and implementation of each layer of this 'unified' stack is presented in section 5.4.

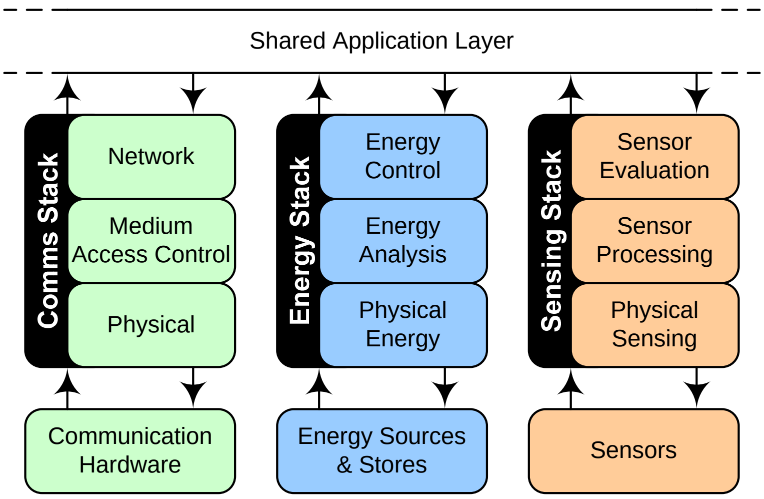


FIGURE 5-4 : The 'unified' stack implemented in the nodes in WSNsim to specify interfaces for communications, energy management, and intelligent sensing.

5.3.3 Use of the Architecture

The proposed architecture is particularly useful where a node has non-trivial functionality in areas other than communication; IDEALS/RMR is a good example of such a case. The incorporation of IDEALS/RMR into the 'unified' stack shown in Fig. 5-4 can be seen in Fig. 5-5. In this diagram, RMR is located solely within the intelligent sensing stack (which is intuitive considering that RMR's event detection and packet assembly techniques operate independently from the energy status of the node or its communication behaviour). IDEALS however is split between the energy management stack and the communications stack (due to its interaction with both the energy state of the node, and the transmission and forwarding of packets). The energy management stack (further information is provided in section 5.4.3) monitors the state of energy resources, and uses IDEALS to allocate the node an EP. Whenever this EP changes, the stack notifies the network layer of the communication stack via the shared application layer. The intelligent sensing stack (further information is provided in section 5.4.2) samples the environment and uses RMR to detect events; any resulting data (and their associated PP) are passed to the network layer of the communication stack via the shared application layer. Upon receiving data to transmit, the network layer uses IDEALS to perform priority balancing with to decide if a packet should be transmitted. Furthermore, if the network layer determines that a packet requires forwarding, it performs priority balancing using the PP embedded in the received packet.

With certain devices, a number of the lower layers of a stack may become very lightweight as functionality is transferred into hardware. An example of this could be a temperature sensor that performs event and fault detection, and provides a reading of the temperature in degrees Celsius (with an associated confidence) transmitted via a digital interface.

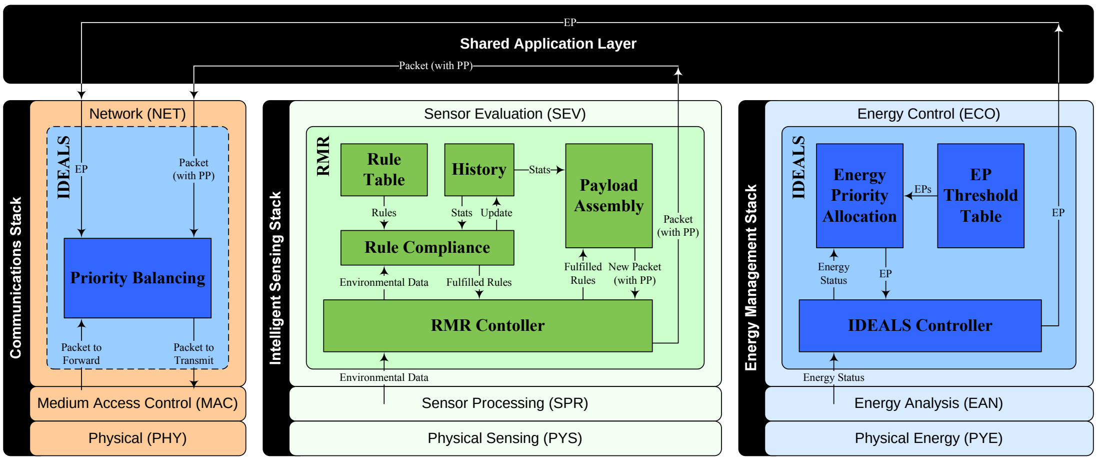


FIGURE 5-5 : The location of IDEALS/RMR in a unified stack specifying interfaces for communications, intelligent sensing and energy management.

Another example of a system that has obvious scope for being structured using the proposed architecture is Prometheus [45], discussed in Chapter 2. Prometheus has considerable additional onboard functionality for energy management, performing store monitoring and switching. If this functionality were translated into an energy management stack, the physical energy (interface) layer would perform the ADC readings to monitor the battery and supercapacitor voltages, and control the switching circuitry to direct the flow of energy. The energy analysis (medium) layer would subsequently provide temperature compensation for the battery voltage, and perform any linearisation or scaling of the ADC output in order to convert the reading into a voltage. Finally, the energy control (management) layer would make the high level decisions regarding whether to use the supercapacitor or battery to power the node, and whether or not to charge the battery. Additionally, this layer could make decisions on the energy state of the node in order to control the duty cycle of sensing and reporting tasks.

Use of the architecture has a number of limitations, restrictions, and costs. While it can also be seen as a positive constraint, the designer is forced to consider their implementation within the bounds of a stack structure (although the presence of multiple stacks arguably makes this process more flexible). The introduction of additional stacks into a node's embedded software is likely to introduce overheads in terms of space, computational time, and power consumption. This is an obvious cost incurred through the use of the architecture, but it is believed that the benefits listed in the previous sections outweigh these costs; numerical quantification will be considered as an area of future research. Recently, cross-layer protocols and algorithms (which claim to improve efficiency through the merging of some or all layers of the OSI-BRM) have received considerable research interest. However, it is arguable that the modularity (lost through cross-layer design) that is gained for all interfaces on a node through our proposed architecture is worth the minor loss in efficiency. Cross-layer optimization techniques could still be applied to the multiple stacks of the architecture as part of the protocol development phase.

Under this PhD, the proposed architecture has been considered as a simulation entity, structuring the node's simulated embedded software in WSNsim (as shown in Fig. 5-1). Through its use to structure sensor node simulation in this PhD, I have found that the proposed architecture provides a number of benefits. Of most significant is the way in which the architecture modularises a problem, and encourages the designer to consider many different aspects of embedded software development that would otherwise either not be attempted, or else complicated to overcome. It is likely that the advantages that I have witnessed through the architecture use in simulation are transferable to its use with real nodes; this is however outside the scope of this thesis. Alex Weddell is however taking this research forward through the further development and implementation of 'unified' stacks for use in a practical node's embedded firmware [191]. Though the implementations presented in the next section have been developed individually by me, the concept of the intelligent sensing and energy management stacks was conceived jointly with Alex Weddell.

5.4 Implemented Protocol Stacks

Through the use of the architecture presented in the previous section, WSNsim allows different layers to be altered or replaced to change functionality. As WSNsim has been designed as a tool specifically for the evaluation of IDEALS/RMR, it has been created to support the three-stack architecture shown in Fig. 5-4. The contents of the individual layers in each stack (used in the simulations presented in Chapter 6), are discussed in this chapter.

5.4.1 The Communication Stack

As depicted in Fig. 5-6, the communication stack implemented in the simulated node's embedded software consists of three layers: the PHY, MAC, and NET.

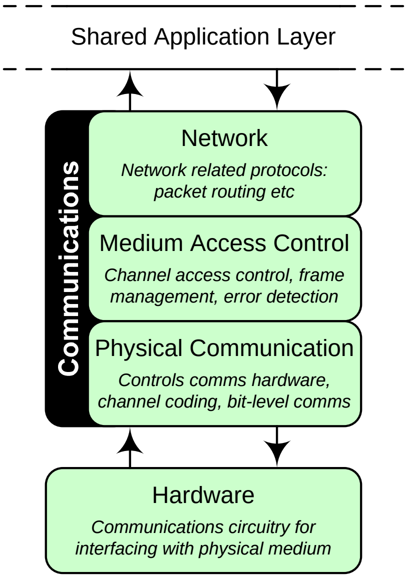

Shared Application Layer
Communications
Network
Network related protocols: packet routing etc
Medium Access Control
Channel access control, frame management, error detection
Physical Communication
Controls comms hardware, channel coding, bit-level comms
Hardware
Communications circuitry for interfacing with physical medium

FIGURE 5-6 : The communication stack used in the nodes in WSNsim.

5.4.1.1 The Physical Layer

The physical layer implemented in WSNsim is based upon the 2.4GHz air interface specified in IEEE 802.15.4-2006 [84], and implements the complete range of PD-SAP (physical layer data service access point) and PLME-SAP (physical layer management entity service access point) primitives, and the PHY-PIB (physical layer personal area network information base). Further information can be found in the IEEE 802.15.4-2006 specification. The PHY has been extended with primitives to support long preambles in order to accommodate the LPL techniques that are required by Broad-MAC; this is discussed in the following subsection.

5.4.1.2 The Media Access Control Layer

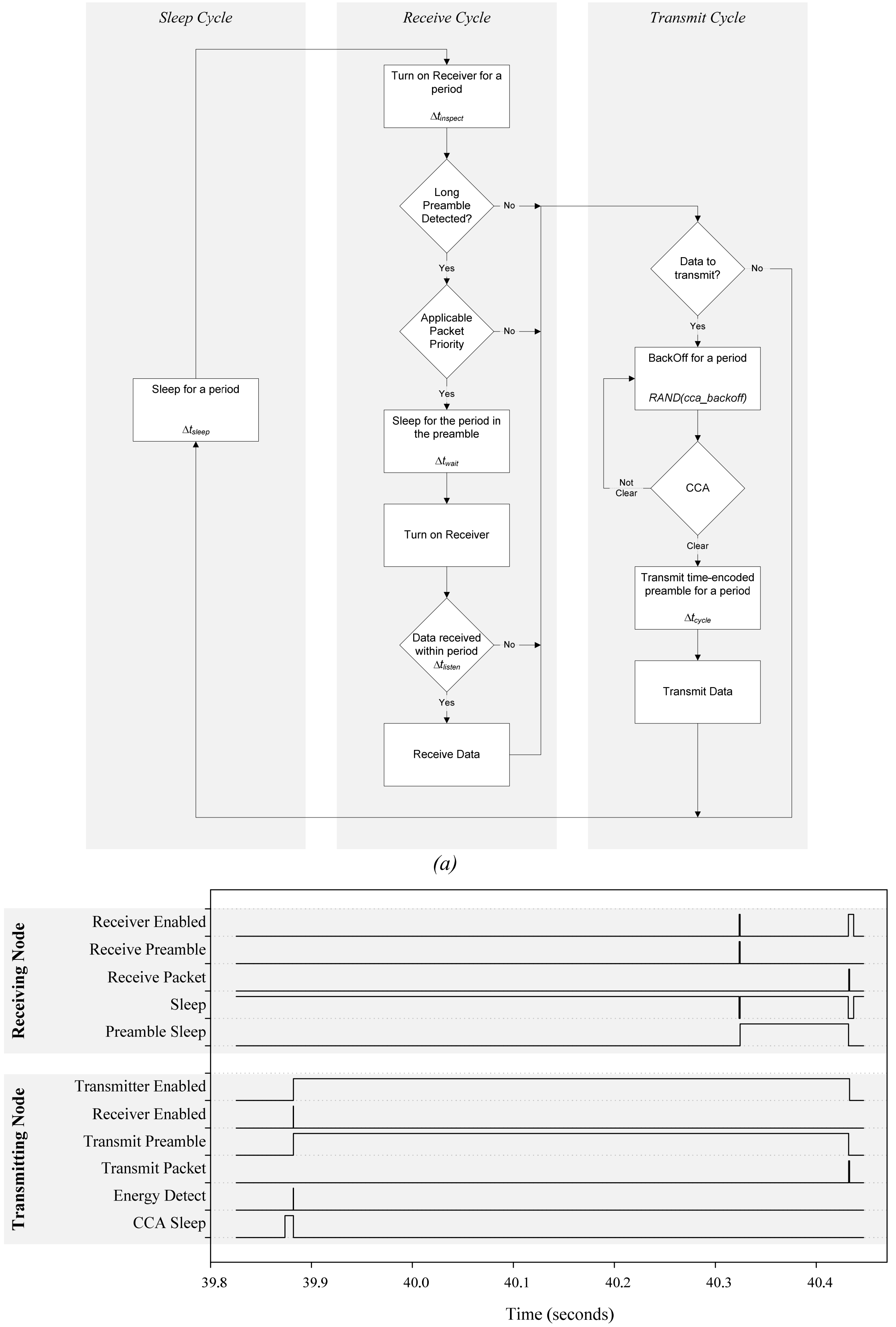
The MAC implemented in WSNsim is a LPL carrier sense protocol based heavily on the concepts of B-MAC [99] and X-MAC [100] (discussed in section 2.3); it shall be referred to as Broad-MAC (Broadcast MAC). Broad-MAC uses the same LPL technique proposed in B-MAC to transfer energy away from the receiver to the transmitter. This is useful in scenarios where data is communicated infrequently; hence the reporting period is significantly lower than the MAC cycle period. X-MAC improves this by embedding the destination address in the long preamble to reduce costs associated with idle-listening, and allows the receiving node to prematurely terminate the long preamble by 'acknowledging' it.

TABLE 5-2 : The Broad-MAC long preamble frame structure.

4 bytes	1 byte	1 byte	1 byte	1 byte	1 byte	1 byte
Preamble	SFD	Length	Type	Remaining Count	PP	XSum

As Broad-MAC is designed for use with broadcast routing algorithms (such as packet flooding), premature terminations and destination address embedding cannot be used to reduce idle-listening. Instead, Broad-MAC embeds the packet's PP in the long preamble so that nodes with a lower EP can ignore the preamble. Additionally, the long preamble contains a count of the remaining number of preambles that are to be sent before packet transmission commences. These can be seen in the frame structure shown in Table 5-2. Because the receiving node knows the length of a preamble frame, it can calculate the time that this process is going to take. Hence, following the detection of a long preamble frame, a receiver node returns to a sleep state until just before the data packet is to be transmitted.

Fig. 5-7a presents a flow diagram representing the operation of Broad-MAC. At the beginning of the receive cycle, the node awakes, and enables its receiver for a period equal to $\Delta t_{inspect}$ [s]. If a long preamble sequence is not detected during this period (or if the PP embedded in the preamble is higher than the node's EP), the node moves to the transmit cycle. Otherwise, after sleeping for a period Δt_{wait} [s] (calculated from data in the long preamble), the node turns on its receiver and receives the packet (timing out if no data is received for a period Δt_{listen} [s]) and enters the transmit cycle. If the node has data to transmit, it waits for a random period of time, following which it performs a Clear Channel Assessment (CCA). If the channel is not clear, the wait-CCA process is repeated; otherwise the node transmits a long preamble for a period Δt_{cycle} [s], following which the packet is transmitted. The node then enters the sleep cycle for a period Δt_{sleep} [s]. The time periods implemented in WSNsim (used for the simulations in Chapter 6) are shown in Table 5-3.



(a)

(b)

FIGURE 5-7 : a) The Broad-MAC flow diagram, and b) a timing diagram showing the packet communication process in Broad-MAC (obtained from WSNsim).

TABLE 5-3 : Broad-MAC timings used in WSNsim.

Time Period	Value
$\Delta t_{inspect}$	700μs
Δt_{listen}	5ms
min cca_backoff	1ms
max cca_backoff	10ms
Δt_{sleep}	500ms
Δt_{cycle}	550ms

The operation of Broad-MAC can be seen from the timing diagram shown in Fig. 5-7b, produced as an output from a simulation in WSNsim. In this diagram, the bottom traces are those of the transmitting node, while the upper traces are those of the recipient.

As shown in Fig. 5-7b, upon wanting to transmit data, the transmitting node first enters a CCA Sleep (or BackOff) state, following which it performs a CCA by enabling the radio receiver and performing a physical layer 'energy detect' operation. Upon detecting that the channel is clear, the transmitting node enables its radio transmitter, and transmits the long preamble. During the transmission of the long preamble, the receiving node awakes from its sleep state, enables its radio receiver, and detects the long preamble transmission. The node decodes Δt_{wait} [s] from the preamble, and returns to sleep for this period of time. Upon waking up again, the node enables its receiver at the same time as the transmitting node stops transmitting the preamble and transmits the packet. Both nodes then return to sleep.

5.4.1.3 The Network Layer

The NET layer implemented in WSNsim performs packet routing. Two different NET layers have been incorporated: packet flooding and 'minimum cost' routing. The different layers can be easily integrated into WSNsim by developing them as separate layer implementations, and then selecting which one is desired in the simulation NDF.

5.4.1.3.1 Packet Flooding

As discussed in Chapter 2, packet flooding operates by a node broadcasting any packet that requires routing. Through this technique, all of the nodes in the network that are in the same connected graph as the source node will receive the packet with the minimum latency. Each receiving node maintains a small circular queue containing the source address and sequence number of each received packet; if a received packet matches any of those in this queue, it is not rebroadcast. Packet flooding was chosen for use with this research for a number of reasons. First, flooding is a simplistic and flexible algorithm which allows for easy integration, understanding, and interpretation of results. Furthermore, while flooding does not scale gracefully, this research is concerned primarily with relatively small networks. If the use of IDEALS/RMR is required with considerably larger networks, the constraining of packet flooding using one of the techniques outlined in Chapter 2 may prove a valid option. The final and most significant reason for using packet flooding is due to its applicability to IDEALS. IDEALS selectively discards packets based upon their deemed importance, which creates a set of EPSNs (Fig. 3-7). Reactive routing algorithms are necessary as these sub-networks are very dynamic in nature, due to nodes in the network changing EPs as their energy levels fluctuate. By using packet flooding, all of the possible routes in the network are made available to the packet, hence maximising packet throughput. This therefore allows IDEALS to be evaluated (as the source of dropped packets) without being concerned with complications in single-path routing algorithms. It is also interesting to note that through packet discrimination, IDEALS effectively 'constrains' the packet flooding process.

5.4.1.3.2 'Minimum Cost' Routing

As packet flooding is an epidemic algorithm suffering from scalability issues, and widely considered to not be energy-efficient, the inclusion of an alternative routing algorithm in WSNsim is desirable in order to provide comparative results. Due to the considerable range of different routing algorithms for WSNs, and no clear winner in terms of the most energy-efficient for all scenarios, the selection of a best-case algorithm is outside the scope of this thesis. However, in order to compare results with a single-path routing algorithm, a 'minimum cost' routing (MCR) algorithm is also implemented in WSNsim. MCR is not designed to be practically realisable, requiring a complete knowledge of the locations and states of all the nodes in the network and is hence implemented using knowledge that would not usually be present in a sensor node.

Upon wishing to transmit a packet, MCR selects the next hop by recursively considering all of the possible routes to the sink node (possible routes are defined as those using nodes that will accept a packet of the current PP), and selecting the one with the smallest hop count. If multiple routes with the same hop count exist, one is randomly selected in order to distribute energy consumption. Only next hops that are within a preset distance are considered; a distance is selected that results in a good chance of successful receipt in order to minimise energy consumption due to retransmissions. Because the algorithm uses a single-path, per-hop acknowledgements are used. The use of acknowledgments requires the modification of

Broad-MAC in order to manage the transmission and receipt of acknowledgement packets, and the retransmission of data packets if no acknowledgement is received.

5.4.2 The Sensing Stack

As depicted in Fig. 5-8, the sensing stack implemented in the simulated node's embedded software is formed of three layers: the physical sensing layer (PYS), sensor processing layer (SPR), and sensor evaluation layer (SEV). As defined in Chapter 4, WSNsim contains environmental sensing models for both a digital temperature sensor and a light sensing circuit. As such, these two sensors have been considered in the sensing stack.

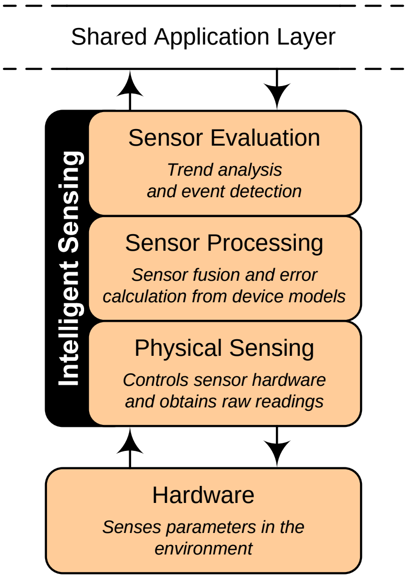

Shared Application Layer
Intelligent Sensing
Sensor Evaluation
Trend analysis and event detection
Sensor Processing
Sensor fusion and error calculation from device models
Physical Sensing
Controls sensor hardware and obtains raw readings
Hardware
Senses parameters in the environment

FIGURE 5-8 : The intelligent sensing stack used in the nodes in WSNsim.

5.4.2.1 The Physical Sensing Layer

The PYS layer interfaces with sensor hardware in order to energise sensors, and obtain raw readings from the node's digital interface (in the case of the digital temperature sensor) or ADC (in the case of the light sensing circuit). The implemented PYS layer handles two requests, *sense* and *range*, from the higher layer. The *sense* request causes the parsed sensor to be sampled, and after the necessary sensing period has passed, the reading is passed to the higher layer. The *range* request updates the current setting of the light sensor's sensitivity range to the value parsed, and returns the new value to the higher layer.

5.4.2.2 The Sensor Processing Layer

The purpose of the SPR layer is to convert raw sensor readings obtained from the PYS into high-level interpreted, error corrected values with associated error bounds where appropriate (of considerable use when performing multi-sensor fusion [192]). This conversion process is performed through the use of embedded sensor models. The implemented SPR layer handles one request, *sense*, from the higher layer, which is simply passed on to the lower layer. In order to convert the raw values obtained from the PYS layer into meaningful values, temperature and light sensing models are implemented based upon the 'inverse' of the models presented in Chapter 4. Obviously, the effects of the random variables in the sensor models cannot be reversed, and hence error exists in the sampled data. The bounds of this error can be calculated through either inspection, or through analysis of the error sources. Additionally, if the reading obtained from the light sensor is nearing saturation, the SPR layer sends a *range* request to the lower layer in order to change the sensitivity range.

5.4.2.3 The Sensor Evaluation Layer

The SEV layer is responsible for coordinating sensing, evaluating the significance of sensed data and performing event detection, and identifying faults in the sensor hardware. Upon waking up from a sleep state (coinciding with the MAC's sleep cycle), the shared application layer (APP) notifies the SEV layer. The SEV sends a *sense* request to the SPR for each attached sensor; the sensors attached to a particular node are specified in the NDF, which are stored in each node's *bitline* at runtime. Upon waking, the shared APP also obtains the current time from the RTC hardware; this reading is provided to the SEV using the *time* command. Upon receiving sensor data from the SPR layer, the SEV processes it using RMR (contained in the SEV layer) to provide event detection. As specified in Chapter 3, on the detection of an event RMR returns an assembled payload which, in addition to the sensed data, contains a timestamp, confidence, signal direction, and sensor descriptor. A payload is then passed from the SEV to the shared APP, accompanied by a respective PP.

5.4.3 The Energy Stack

As depicted in Fig. 5-9, the simulated energy stack is formed of three layers: the physical energy layer (PYE), energy analysis layer (EAN), and energy control layer (ECO). As defined in Chapter 4, WSNsim contains physical energy store models for 'ideal', dual supercapacitor, and secondary battery stores, and these are considered in the energy stack.

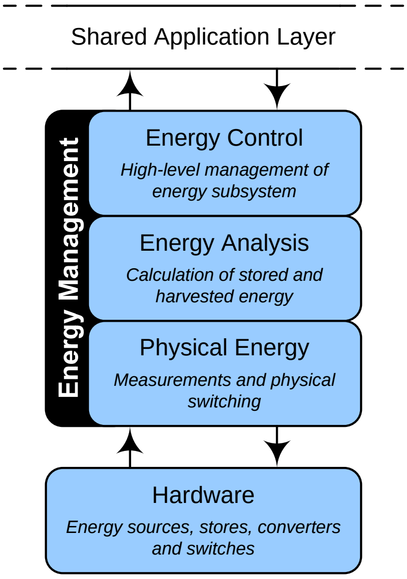

Shared Application Layer
Energy Management
Energy Control
High-level management of energy subsystem
Energy Analysis
Calculation of stored and harvested energy
Physical Energy
Measurements and physical switching
Hardware
Energy sources, stores, converters and switches

FIGURE 5-9 : The energy management stack used in the nodes in WSNsim.

5.4.3.1 The Physical Energy Layer

The PYE layer is responsible for monitoring and controlling a node's energy hardware in order to manage the supply and demand of energy. To monitor the residual energy in an energy store, the PYE layer uses the onboard ADC to obtain a reading of the voltage across the energy store; this is performed on receipt of a *measure* request from the higher layer and, when obtained, the resultant measurement is sent to the higher layer.

5.4.3.2 The Energy Analysis Layer

The EAN layer converts voltage measurements obtained by the PYE into meaningful statistics such as the residual energy, residual energy fraction, or remaining lifetime. This is performed through the use of models for the dual-supercapacitor and battery (Ni-MH, Ni-Cad, and Li-ion) stores that convert the voltage into the residual energy; these models are based upon the 'inverse' of the models presented in Chapter 4. Once calculated, the residual energy level – expressed as both a relative and absolute value – is sent to the higher layer.

5.4.3.3 The Energy Control Layer

The ECO layer takes a high-level view of the energy subsystem, and controls the energy-aware operation of the node. This includes the determination of the node's EP based upon the data received from the EAN layer. Upon waking up from a sleep state (which coincides with the MAC's sleep cycle), the shared APP notifies the ECO layer which subsequently sends a *measure* request to the EAN layer to monitor the store voltage. Upon receiving a measurement of the energy residual in the store, the ECO layer calculates an EP that is assigned to the node.

5.5 Discussion and Summary

This chapter has outlined research that has been central to the modelling and simulation themes of this thesis. WSNsim has been presented as a simulator developed for the purpose of enabling the evaluation of IDEALS/RMR with the range of desired environmental and physical models. Designed for the purpose of simulating relatively small networks, scalability was not an aim in the development of WSNsim; scalability is sacrificed through both the object-orientated structure, and the depth incorporated in some of the models. While certain aspects such as the 'unified' architecture are new and novel concepts, the development of WSNsim was not the primary purpose of this research. Instead, the simulator is a means to both house the collection of developed environmental and physical models, and to provide a method for the analysis and development of IDEALS/RMR.

This chapter has also discussed a 'unified' architecture suitable for independently structuring multiple functions on a senor node. Use of the architecture enables the process of developing software and algorithms for sensor nodes to be better structured, and devolving the focus of algorithm development away from communication. Descriptions of the implemented layers for the communications, intelligent sensing and energy management stacks have been provided in order to place the simulation results presented in Chapter 6 in context. The communication stack uses a MAC layer optimised for use with IDEALS/RMR (based heavily upon the concepts of B-MAC and X-MAC), and network layers for both packet flooding and 'minimum cost' routing. The sensing stack interfaces with the sensing models to process sensed data, control sensor hardware, and uses RMR to perform event detection, assign PPs, and assemble payloads. Finally, the energy stack interfaces with the energy models in order to monitor store voltages, calculated residual energy levels, and assign EPs. While in the context of this thesis, the 'unified' architecture has only been considered as a simulation structure, it has obvious applications to practical embedded systems, and this is being researched further by Alex Weddell [191].

Chapter 6

Simulation Results

Chapter 3 presented IDEALS/RMR, an energy- and information-managed algorithm for WSNs. Subsequently, Chapter 4 investigated communication, energy, sensing and timing models which were integrated into WSNsim (presented in Chapter 5). This chapter presents simulation results obtained from WSNsim which demonstrate the advantages that can obtained through the use of IDEALS/RMR. Results from two different simulation scenarios are presented: a network operating under static conditions (section 6.1), and a scenario depicting a network monitoring temperature in a water pumping station (section 6.2).

As highlighted in Chapter 2, the union of energy and information management in WSNs is a novel concept, thus restricting its comparative evaluation. It will therefore be compared to the case where nodes transmit packets based only upon RMR controlled event detection.

6.1 Static Conditions

The static conditions network is simulated to ascertain the benefits of IDEALS under constant conditions, testing the network to destruction under controlled decay. Additionally, this section evaluates the various simulation models proposed in Chapter 4.

6.1.1 Simulation Scenario, Setup and Parameters

In this simulation, a small WSN of nine nodes is organised in a regular 3x3 grid with a single sink node in the bottom left of the network that is the destination of all packets; a snapshot of this is shown in Fig. 6-1. In this snapshot, the small dots represent sensor nodes, the lines between them represent links between nodes where the separation distance is under

20 meters, and the shading shows the amount of energy remaining in the node (as this snapshot is taken at zero seconds, all of the nodes have full energy stores).

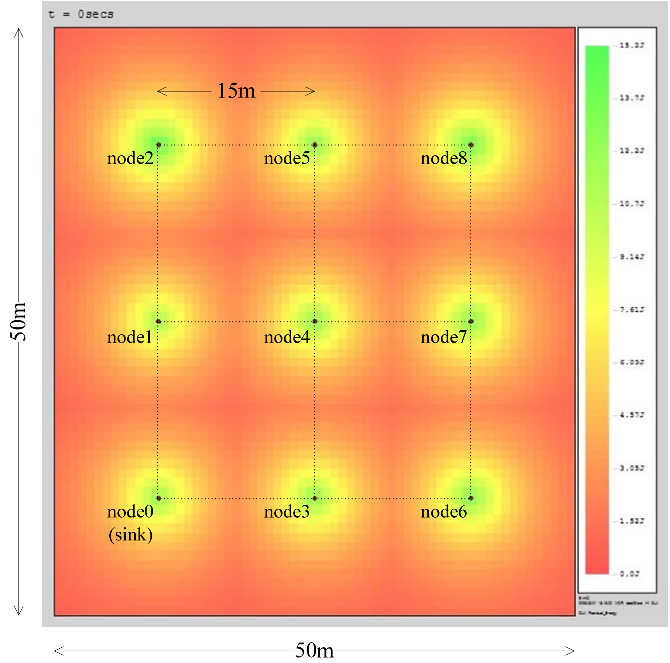


FIGURE 6-1 : A snapshot of the static conditions network under simulation in WSNsim.

During simulation, the sink node turns on at zero seconds, while the remaining nodes turn on at random times within the first minute. The simulations run for a period of 24 hours. During this period, the temperature sweeps linearly from -25ºC to 80ºC in order to evaluate the temperature dependence of the simulation models. The node separation distance of 15m (21.2m across the diagonal) was selected in order to enforce multi-hop routing using the grid structure. 'Minimum cost' routing was configured to consider only separation distances of under 20m, which was identified as providing a good chance of packet success. Under packet flooding, some packets will be transmitted via diagonal neighbours (not shown in Fig. 6-1), but this will be achieved with a lower packet success rate.

The energy stores used in these simulations have a maximum (and initial) nominal capacity of 15.228J (chosen to equal the capacity of a dual 4.7F supercapacitor store) operating at 3.6V (the maximum operating voltage of the CC2430). Three different energy stores (ideal, battery and dual supercapacitor) are individually simulated, each with a capacity (and initial energy level) of 15.228J. This is a very limited energy store when compared with that provided by a traditional AA size battery, with the supercapacitor store providing only enough energy to power the node's radio receiver for 140 seconds if left permanently on. When energy harvesting is enabled, a static level of 500µW is harvested (for example, from an ‘always-on’ vibration energy harvester); this equates to 30mJ in a minute or 43.2J in a day. Therefore, due to the very limited store capacity, the network relies heavily on the presence and management of energy harvesting. The configuration of the EP thresholds (where IDEALS was enabled) is shown in Table 6-1.

TABLE 6-1 : EP thresholds used in the simulation of the static conditions networks.

Energy Priority	Lower Bound	Upper Bound	Hysteresis
EP0	0%	5%	2%
EP1	5%	30%	
EP2	30%	50%	
EP3	50%	70%	
EP4	70%	90%	
EP5	90%	100%	

The nodes in the network operate with a duty cycled MAC algorithm with a sleep period of 500ms (the total length of the duty cycle depends on the activity of the node during the awake period). The sampling of the temperature sensor, and inspection of the energy store are coordinated with this MAC duty cycle. For the purposes of comparison, the timing model is configured to not reset the RTC when node runs out of energy (equivalent to assuming that the node has a separate backup battery to maintain the RTC). This was implemented to allow the harvesting only network to be objectively compared (as when the RTC resets, the timestamp on the data packets also resets). Alternatively, the node could request the current time from other nodes when it turned back on, but this is beyond the scope of this work.

When IDEALS/RMR is disabled, each node in the network reports data to the sink node every two minutes. When IDEALS/RMR is enabled, the temporal resolution of the network is traded with the nodes energy; the rules used to achieve this are shown in Table 6-2, whereby the reporting frequency halves every time the EP drops. Note that in this simulation, RMR controls only the reporting frequency of the network using routine rules. In the next section, considering an actual application scenario, complete event-based operation is considered.

TABLE 6-2 : RMR rules used in the simulation of the static conditions network.

Name	Type	PP	Value		Parameter
R(2M)pp5	Routine	5	2	mins	n/a
R(4M)pp4		4	4		
R(8M)pp3		3	8		
R(16M)pp2		2	16		
R(32M)pp1		1	32		

The static conditions network was simulated under four configurations:

i) no energy harvesting or IDEALS/RMR

ii) IDEALS/RMR only

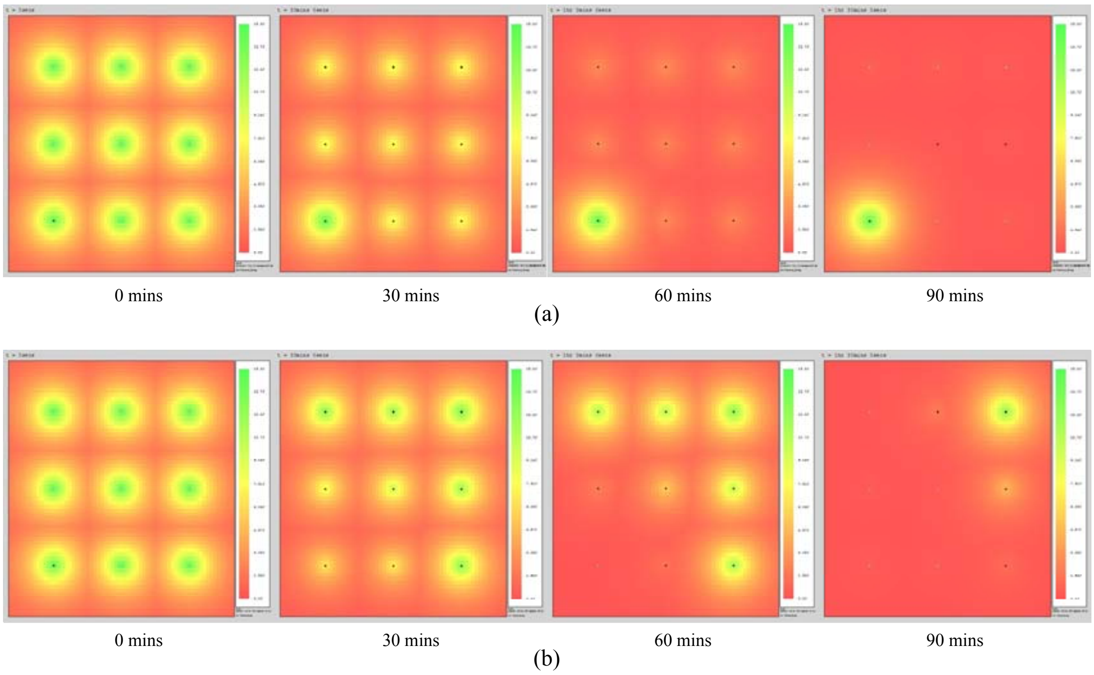
iii) energy harvesting only

iv) energy harvesting and IDEALS/RMR.

The remainder of this section presents results using a variety of different metrics relevant to the investigated areas of energy and information.

6.1.2 Energy

This section presents an analysis of the developed algorithms and models from the perspective of energy.

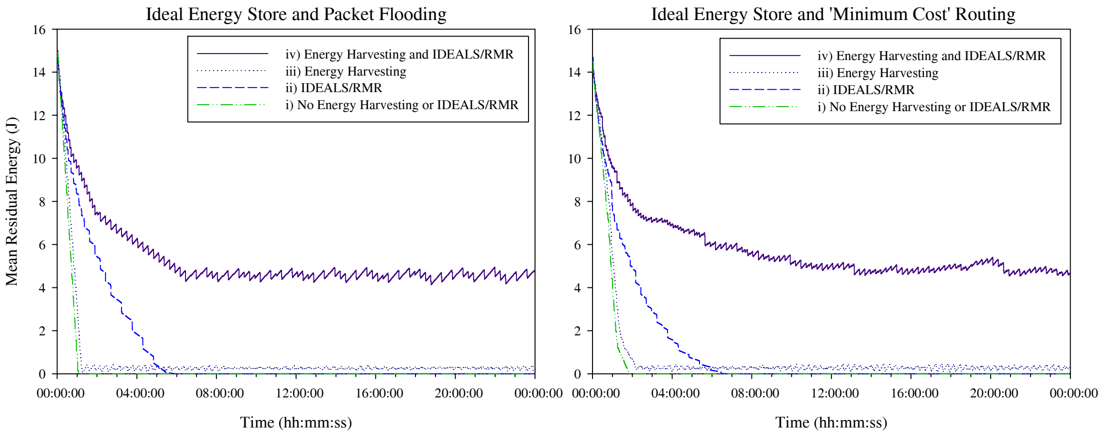


(a)

(b)

FIGURE 6-2 : Deterioration of the nodes' energy levels during a) packet flooding, and b) 'minimum cost' routing. The nodes in both simulations utilised energy harvesting and no IDEALS/RMR.

Fig. 6-2 shows a sequence of snapshots during the simulation showing the energy state of the nodes using a) packet flooding, and b) 'minimum cost' routing. In these snapshots, the small dots represents sensor nodes (a dark dot depicting that the node is turned on), and the colour map represents the node's residual energy (where green equals a full store, and red equals a depleted store). When packet flooding is utilised (Fig. 6-2a), all nodes expend approximately the same energy except for the sink node that needs only to receive packets (it is the destination of all packets, and hence is not required make any transmissions). However, with 'minimum cost' routing, nodes closer to the sink node deplete their energy reserves first, as they must cooperate in the routing of all packets in the network. Furthermore, the sink node has to acknowledge receipt of every packet, and hence depletes its energy reserve quickly.



(a) (b)

FIGURE 6-3 : Mean residual energy levels for the nodes in the network utilising ideal energy stores and a) packet flooding, and b) 'minimum cost' routing.

Fig. 6-3 shows the mean residual energy level for the different network configurations using ideal energy stores with a) packet flooding, and b) 'minimum cost' routing. The mean residual energy for packet flooding does not include energy expended by the sink node as, due to the reasons mentioned above regarding its receive-only operation, it would provide a bias to the mean results. It can be seen that the graphs for the different routing algorithms are very similar, with only minor differences (the most noticeable of which is that 'minimum cost' routing adds variability due to its less uniform operation. As expected, networks that do not feature energy harvesting or IDEALS are the first to deplete their energy reserves (network 'i'). If energy harvesting is added (network 'iii'), the rate of depletion is slightly reduced (as the nodes are obtaining a comparatively small energy increase). Once the energy level of a node in network 'iii' drops below 5% (approximately 0.75J), it begins to locally oscillate as the nodes toggle between EP0 and EP1. If IDEALS is enabled (network 'ii'), the rate of depletion decreases at specific points (determined by IDEALS' EP thresholds) as the energy level drops, resulting in a network lifetime extension. If energy harvesting is added to IDEALS (network 'iv'), the effect is emphasised, and the network lifetime is extended indefinitely, with the node settling to oscillate between EP1 and EP2 (the boundary is at approximately 4.5J).

The results shown in Fig. 6-3 were obtained from a single-run of WSNsim. They are however a representative example of the results that are obtained for this simulation scenario, with repeated experiments expected to produce very similar results. This is because there are no stochastic parts to the ideal energy store model, or the energy consumer and 'simple' energy source models. The only variability that would affect these results through repeated simulation would be as a result of packet success in the communication model, which would have only minor affect on the results for 'Minimum Cost' Routing (by requiring retransmissions due to lost packets). Variability in the sensing models will not have an effect on residual energy levels in this simulation scenario, as nodes are configured to report packets every two minutes regardless of the data sensed.

Due to the concept of deliberately discarding packets, the energy gained by energy harvesting (network 'iii') increases the network lifetime considerably more if coupled with IDEALS (network 'iv'). To explain this, consider that at the beginning of the simulation (when the node is in EP5), an average of 4.5 packets are transmitted through the network every minute, with a fixed amount of energy being harvested. However, when the nodes are in EP1, an average of only 0.3 packets are transmitted through the network every minute, while the same amount of energy is harvested. These results also show that for networks of this size, flooding is not particularly inefficient, and provides benefits including minimum latency and good reliability.

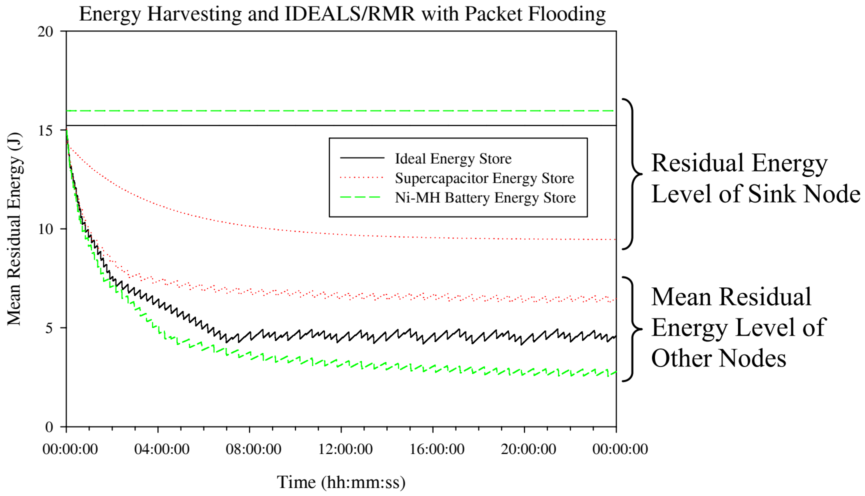
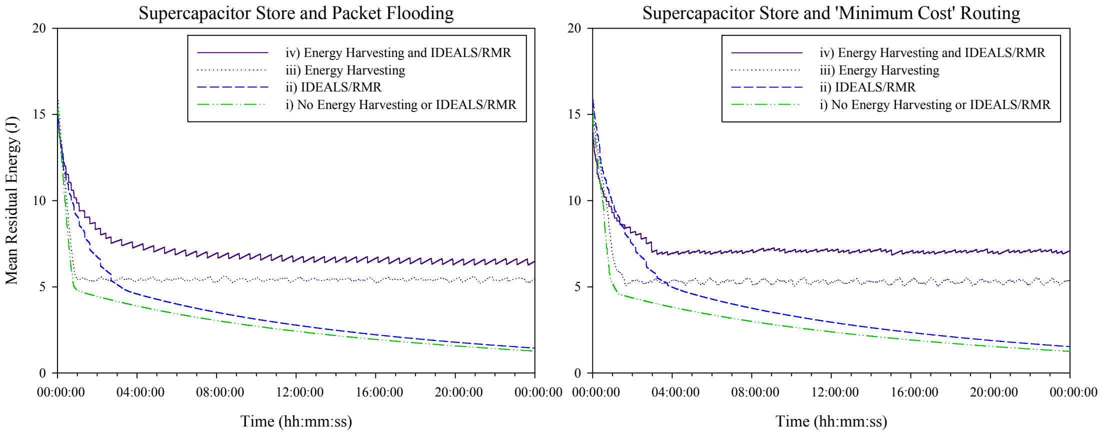


FIGURE 6-4 : Residual energy levels for the nodes in the network using various energy stores with packet flooding.

The results presented in Fig. 6-3 are for nodes using ideal energy stores. Fig. 6-4 compares different energy store models for nodes using packet flooding with energy harvesting and IDEALS/RMR (network 'iv'), and demonstrates the importance of store modelling. In this figure, both the residual energy level of the sink node and the mean residual energy level in the remaining nodes can be observed. The sink nodes using ideal energy stores and Ni-MH batteries are able to remain fully charged by using harvested energy to overcome the energy requirements of the MAC cycle, receiving data, powering peripherals, and any leakage. The leakage present in the sink node using the supercapacitor store is, however, too great to be countered by energy harvesting, and hence the residual energy drops until the leakage is balanced by energy harvesting. As seen in Fig. 6-3, the non-sink nodes in network 'iv' achieve energy-neutral operation after approximately seven hours. However, the nodes with a Ni-MH energy store are not able to achieve this, as their charging efficiency reduces the effective energy harvested (shown in Fig. 6-4). It should be remembered that while the supercapacitor energy store seems to be more efficient than the ideal energy store in Fig. 6-4, the node is not able to use 5J of the energy in the store, as it is available at a voltage below the operating voltage of the node. When this is subtracted, it can be seen that the nodes are tending towards just achieving energy-neutral operating, albeit operating in the lowest EP; this is primarily a result of the leakage present in the supercapacitors.



(a) (b)

FIGURE 6-5 : Mean residual energy levels for the nodes in the network utilising dual supercapacitor energy stores and a) packet flooding, and b) 'minimum cost' routing.

Due to the obvious differences between the ideal energy store and the practical energy stores, the supercapacitor energy store will be considered for the remaining results in this section. The mean residual energy level for the different network configurations using supercapacitor energy stores with a) packet flooding, and b) 'minimum cost' routing are given in Fig. 6-5. In these figures, the presence of energy harvesting can be seen retaining the node above its usable residual energy level, while on the two non-harvesting networks, the energy store continues to leak after the node has turned off due to an insufficient store voltage. While intuitive, it is interesting to note that if energy harvesting later provided energy to the store, the node would require enough energy to return the store to >2V (at a residual level of 5J) before the node would begin to operate. As shown in the previous results, IDEALS/RMR controls the degradation of the network (networks 'ii' and 'iv').

Fig. 6-5 shows only the mean residual energy level, hence the deviation of individual nodes about the mean cannot be seen. Observing Fig. 6-2, it can be seen that, especially with 'minimum cost' routing, the variation in energy is quite considerable. Through inspection, it is found that IDEALS/RMR manages the state of the entire network through independent and locally greedy decisions, spreading the energy cost over the network and maintaining usability.

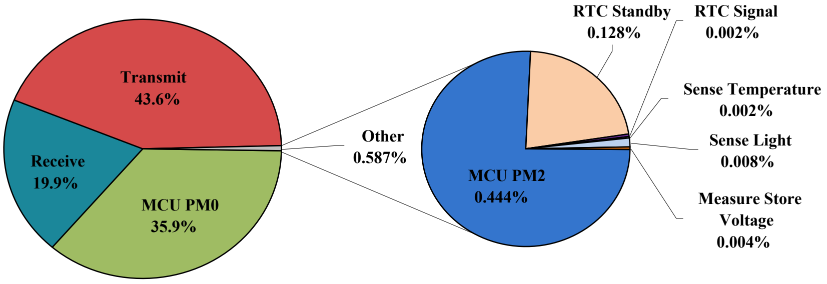
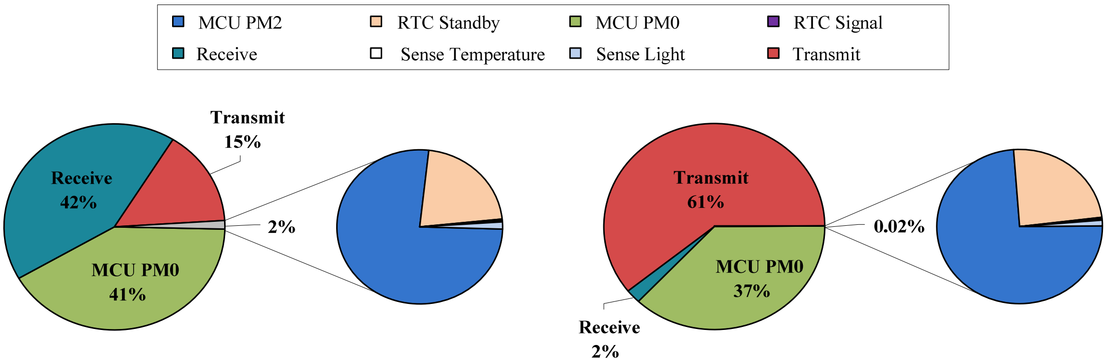


FIGURE 6-6 : The mean distribution of energy consumption for nodes using packet flooding with energy harvesting and IDEALS/RMR (network 'iv').

In addition to providing information on the residual energy level of nodes, WSNsim also produces statistics regarding where energy is consumed within nodes, as shown in Fig. 6-6. It can be seen that in this simulation, the radio transceiver dominates the consumption of the node (constituting almost two thirds of the energy consumption). However, it is also interesting to note that the receive power constitutes a large part of this, and hence if only the transmit power were considered, significantly longer (and unrealistic) lifetime predictions would be produced. Although a LPL MAC is used (designed to shift the energy consumption from the radio receiver to the transmitter), transmissions are only occurring approximately once every two minutes, whereas the receiver is listening (albeit for a considerably shorter period of time) twice a second. If a non LPL MAC was used, receiving would constitute a considerably larger portion of the energy consumption. Additionally, the microcontroller in a non-sleep mode is shown to be a predominant consumer, consuming around a third of the energy. In this simulation, other consumers constitute a negligible proportion of the energy consumption. However, as proposed in Chapter 4, the distribution of energy consumption is able to change considerably with a small change in the network setup. To illustrate this, the same simulation setup is used but with the nodes adopting different reporting frequencies (reporting every 30 seconds, or every two hours). The energy consumed by various consumers for these two reporting frequencies is shown in Fig. 6-7. These simulations use unconstrained energy stores in order to highlight only the difference in consumption. It is important to consider that the pie charts provide relative and not absolute values; in fact, the nodes in ‘a’ use an average of 20J, whereas those in ‘b’ use an average of 1kJ. When the reporting frequency is increased from two hours to 30 seconds, the percentage of energy used in receiving is reduced, as the node is transmitting considerably more often. Furthermore, the proportion of energy consumed by ‘other’ consumers reduces by two orders of magnitude, as these remain constant as the energy budget is heavily increased. By comparing the charts in Fig. 6-7, the effect on energy consumption as a result of a simple change in the network configuration can be seen, and hence the importance of modelling all consumers appreciated.



a) two hour reporting frequency

b) thirty second reporting frequency

FIGURE 6-7 : A comparison of energy consumption with networks utilising 1) 120 minute, and b) 0.5 minute reporting frequencies (both simulations were run without energy harvesting or IDEALS/RMR, but with unlimited energy reserves).

In addition to providing information on how much energy each consumer has used, WSNsim also outputs statistics on how long each consumer was enabled for. Fig. 6-8 shows this for the simulation of network ‘iv’. Of particular interest are the times spent communicating and processing. The CC2430 is a System-on-Chip solution integrating a microcontroller and radio transceiver. In the conducted simulations, the energy consumption for the microcontroller was separated from that of the transceiver (often quoted and simulated as combined figures), as it is not necessary for the transceiver to be enabled while the microcontroller is. Reinforcing this decision, it can be seen that in this simulation the radio transceiver is on for a total of 419 seconds, whereas the microcontroller is active for 446 seconds. This means that for 6% of the time, the microcontroller is on without the radio transceiver, and causing an increase in the energy consumption of the node.

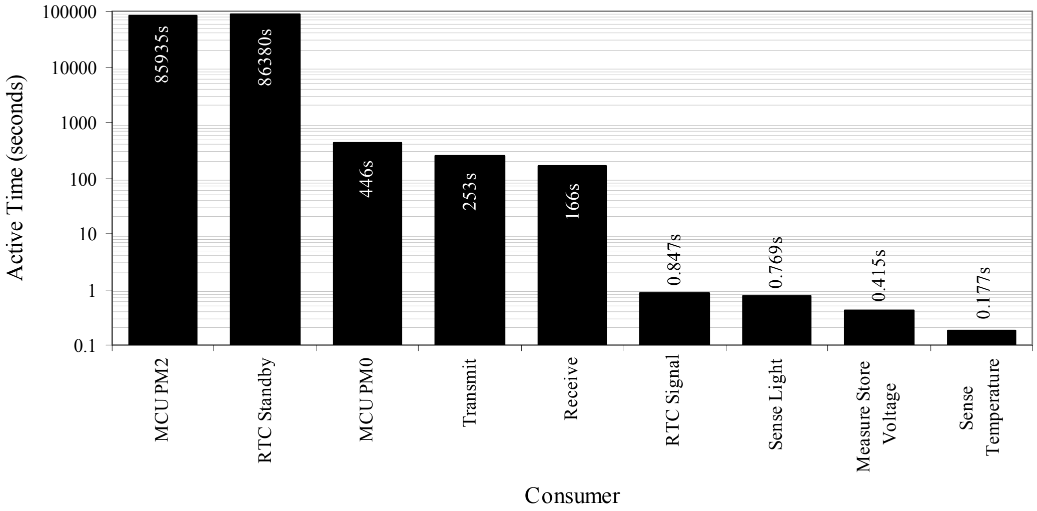


FIGURE 6-8 : The time periods for which different energy consumers remained active.

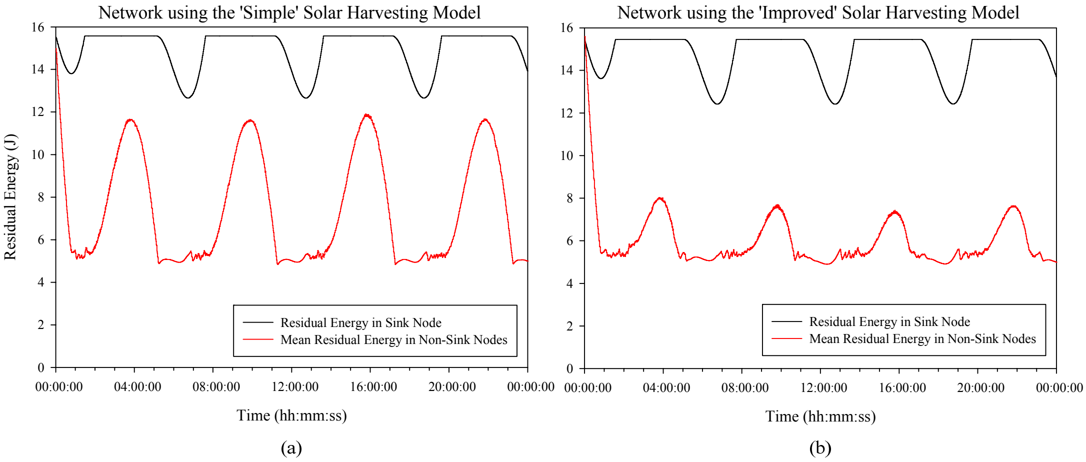


FIGURE 6-9 : Residual energy levels for the nodes in a network utilising solar energy harvesting with an ideal energy stores, simulated with a) the 'simple' solar harvesting model, and b) the 'improved' solar harvesting model.

When energy harvesting has been enabled in the above results, a continuous and static power has been provided to the nodes. To complete the energy results in this section, the effects and differences between the two solar harvesting models is considered. In these results, the environmental light level varies between one lx and 2000 lx four times a day (in order to demonstrate the operation). The nodes are powered from supercapacitor energy stores, and operate using packet flooding and IDEALS/RMR. As explained in detail in Chapter 4, the major difference between the two solar harvesting models is that the simple model considers the photovoltaic to provide a output power directly related to the incident light level, while the 'improved' model considers also the I-V characteristics of the photovoltaic, resulting in the power obtained to be dependent on the store voltage. These results are presented in Fig. 6-9, showing the residual energy in the sink node and the mean residual energy in the remaining nodes for both the 'simple' and 'improved' solar harvesting models.

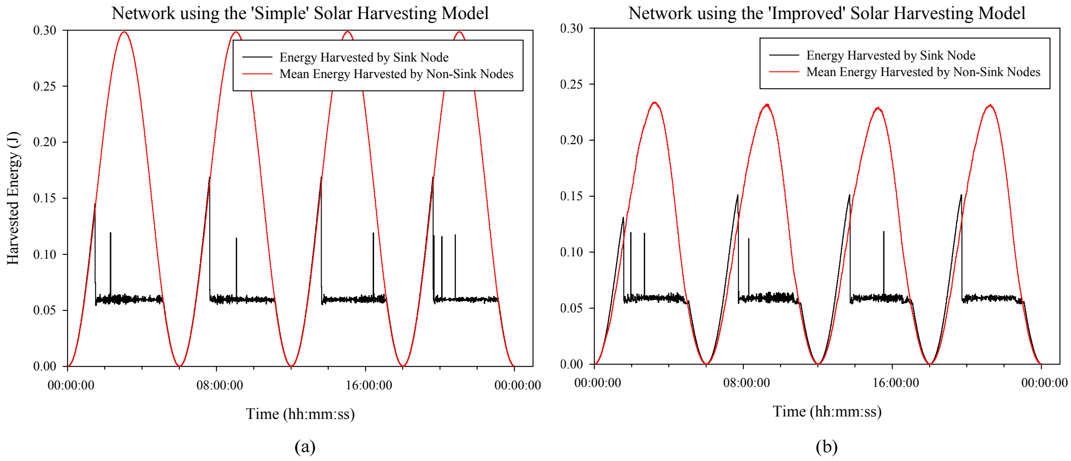


FIGURE 6-10 : Energy harvested by the nodes in a network utilising solar energy harvesting with an ideal energy stores, simulated with a) the 'simple' solar harvesting model, and b) the 'improved' solar harvesting model.

From Fig. 6-9, it can be seen that the energy level of the sink node is similar in both models, as the store voltage is near maximum and hence the I-V characteristics of the photovoltaic are not limiting the energy harvested. However, considering the energy in the non-sink nodes, it can be seen that a significant reduction in harvested energy is obtained with the 'improved' model due to the nodes operating at a considerably lower store voltage of approximately 2V. This can also be seen in Fig. 6-10, which shows the harvested energy that is successfully stored in the energy store. Both the sink node and the non-sink nodes using the simple source model (Fig. 6-10a) harvest energy at the same rate; this is seen as both traces overlap until the sink node's store becomes saturated, at which time only enough energy to sustain operation is harvested. However, in the 'improved' model, the sink node harvests energy at a faster rate than the non-sink nodes (the two traces do not overlap), as the lower operating voltage means that a lower power is harvested. It can also be seen that the individual curves are asymmetric, due to the store voltage increasing during harvesting.

6.1.3 Information

This section presents an analysis of the developed algorithms and models from the perspective of information.

Fig. 6-11 shows the results for packet (plots ‘a’ and ‘c’) and information (plots ‘b’ and ‘d’) throughput from the simulations featuring packet flooding and no energy harvesting. Comparing graphs ‘a’ and ‘b’ with ‘c’ and ‘d’, it can be seen that IDEALS/RMR enables the network to survive and hence report packets for a long period of time (extending the lifetime by a factor of four). This supports the residual energy results presented in Fig. 6-5 and, also explains the non-linear energy depletion that could be seen in these results, a result of IDEALS/RMR reducing packet transmissions as the nodes’ energy reserves deplete.

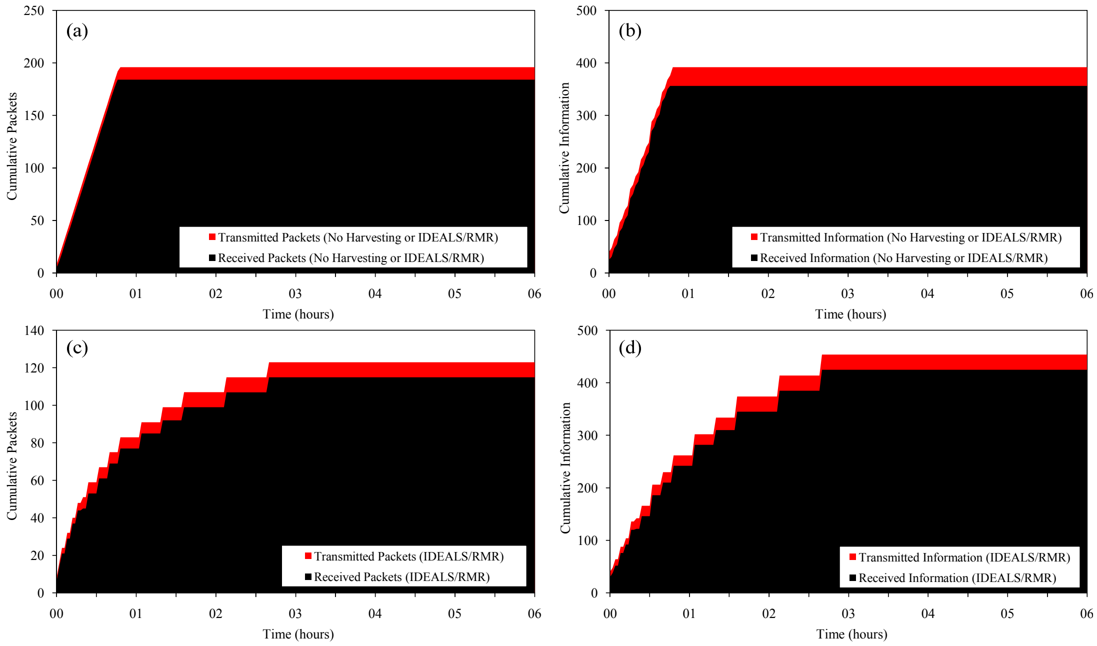


FIGURE 6-11 : Packet and information throughput for networks not featuring energy harvesting.

The network featuring IDEALS/RMR (plot ‘c’) communicates fewer packets to the sink node than that not featuring IDEALS/RMR (plot ‘a’) before the nodes’ energy stores are depleted (115 packets as opposed to 184). This is because the nodes featuring IDEALS/RMR have a longer lifetime; therefore, they expend a greater proportion of their energy budget on static energy consumers such as the MAC cycle and energising of peripheral devices. However, it is important to note that although fewer packets are throughput, IDEALS/RMR allows the transfer of greater information (plot ‘d’ compared to plot ‘b’) communicating 425 ‘units’ of information as opposed to 356. The loss of packets (packets that are transmitted but not received, highlighted in red in Fig. 6-11) are due to the nodes in the network turning on at random times during the first minute of network simulation; therefore, a route to the sink node is not necessarily present during this time.

Fig. 6-12 shows the results for packet (plots ‘a’ and ‘c’) and information (plots ‘b’ and ‘d’) throughput for the simulations featuring packet flooding and energy harvesting. Comparing graphs ‘a’ and ‘b’ with ‘c’ and ‘d’, it can be seen that IDEALS/RMR causes considerably fewer packets (only 449 as opposed to 1329) and information (2035 as opposed to 2807) to be throughput. This is because IDEALS/RMR is using a defined ‘policy’ to control the operation of the node and the network in order to strategically save energy while still communicating a high level of information. Conversely, the network not featuring IDEALS/RMR is transmitting packets at a high rate whenever the node has energy. Therefore, when a path is not present between the source and sink nodes (due to nodes not being instantaneously operational), packets are not successfully received (shown by the packet success rate of 59% as opposed to IDEALS/RMR’s 96%). Furthermore, if the average information per packet is considered, IDEALS/RMR transfers an average of 4.5, as opposed to 2.1, hence making better use of the available energy. The operation and benefits of this are more obvious in the application-based results presented in section 6.2.

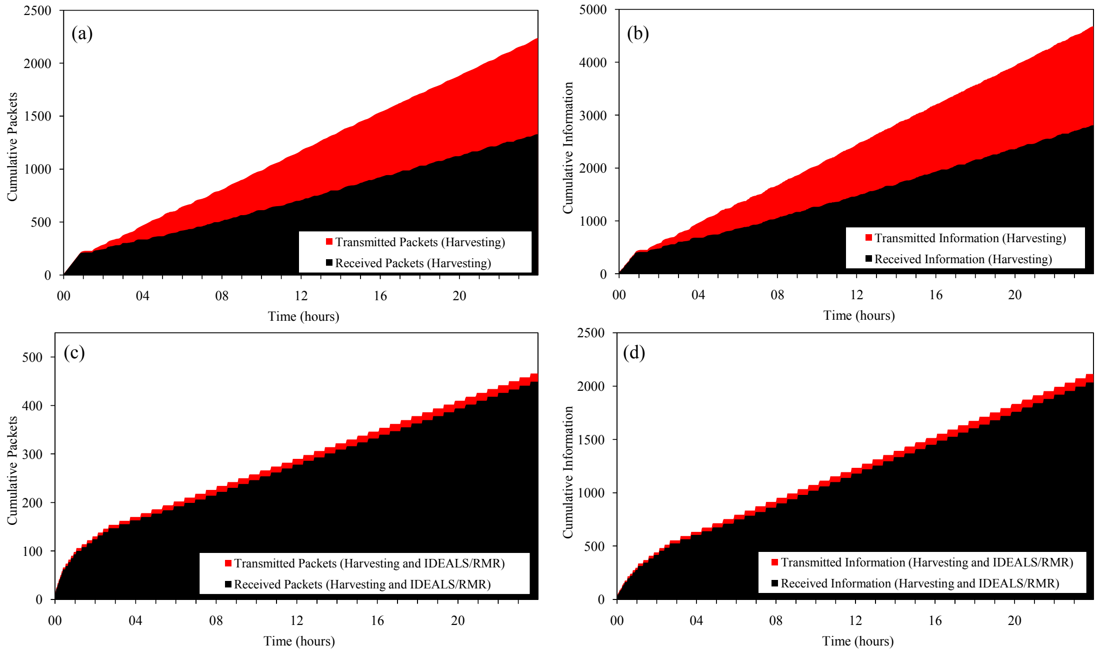


FIGURE 6-12 : Packet and information throughput for networks featuring energy harvesting.

Fig. 6-13 shows the possible reconstruction of light level data at the sink node from two nodes in the network, node 01 (only one hop from the sink node) and node 08 (furthest distance from the sink node). Energy harvesting is present in each simulation. A scaling difference can be observed in every plot between the reconstructed traces and the actual (environmental) variation; this is because the stochastic sensor model is providing different values on the different simulation runs, resulting in the traces being scaled (shown below in Fig. 6-14). Furthermore, it can be seen that the reported data is zero for the first hour of operation; this is due to the environmental temperature sweep that is in operation during this simulation to test the sensor models causing operation outside the temperature bounds of the sensor models. These properties however do not affect the analysis of the results in this section.

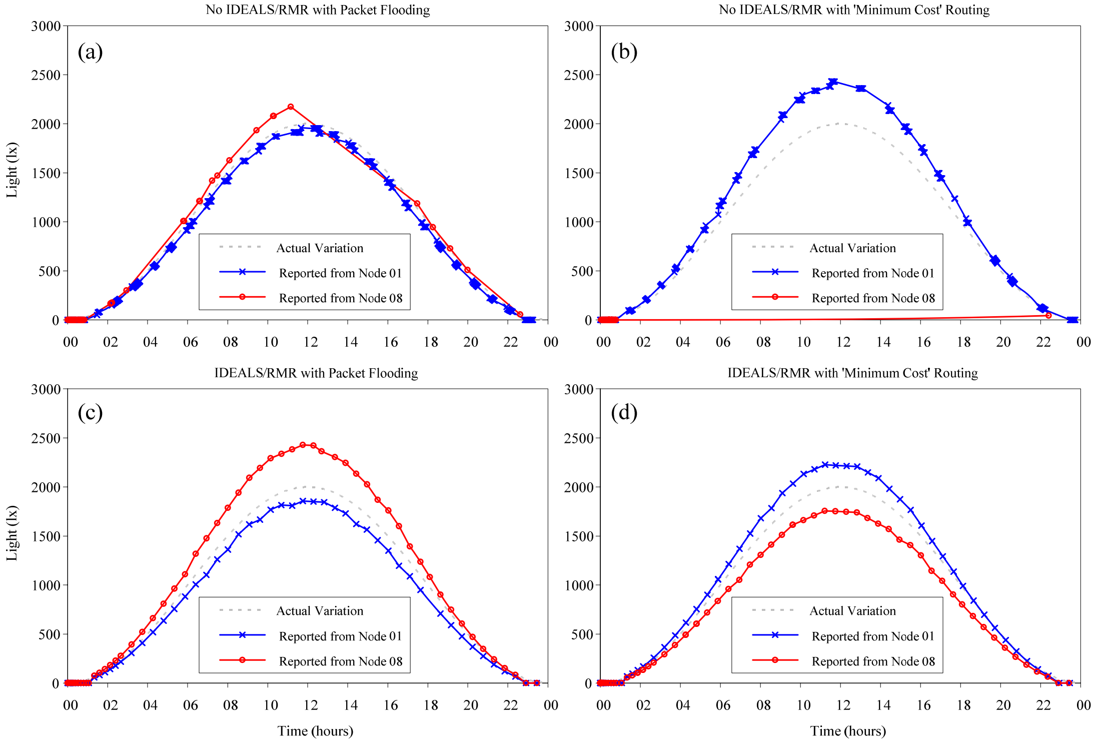


FIGURE 6-13 : Data reconstruction performed at the sink node for a node one hop from the sink node (node 01) and a node four hops from the sink node (node 08).

In Fig. 6-13a, the reconstruction of the sensed data can be seen when no IDEALS/RMR is present. Node 01 is able to sustain operation and intermittently report to the sink node when enough energy is harvested (this can be seen by the ‘groups’ of reports) However, due to no coordination between nodes, the reconstruction of data from node 08 is limited, due to large periods of missed data points when a route didn’t exist across the network. With ‘minimum cost’ routing (Fig. 6-13b) this effect is extenuated, with few packets making it across the network from node 08. In Fig. 6-13c, IDEALS/RMR manages reporting and permits good data reconstruction. The operation of IDEALS/RMR can be seen by the gradual reduction in reported packets during the first few hours of operation, and also the removal of the groups of reports, instead spreading the energy over a greater period of time. These benefits of IDEALS/RMR can also be transferred to ‘minimum cost’ routing, as shown in Fig. 6-13d.

Finally, Fig. 6-14 provides a plot of the environmental light level against the sensed light level; this is shown for all nine nodes in a single simulation, and shows the effect of the light sensing model on the accuracy of sensed data. In this figure, if the modelled sensor contained no error, a linear trace with unity gradient would be plotted for all nine traces on the graph. However, due to the stochastic model used for the light sensor circuitry, and the quantisation performed by the node's ADC, this behaviour is not observed. This highlights the result of the many sources of error introduced by the model (as discussed in Chapter 4). The modelling of these errors are particularly relevant in WSNs, where often cheap and uncalibrated sensors are deployed in order to reduce size, cost and time.

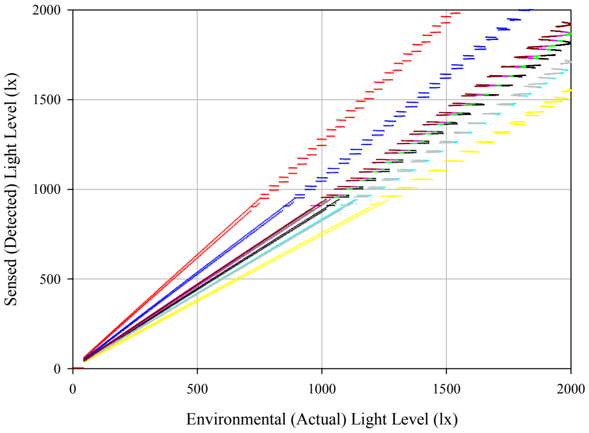


FIGURE 6-14 : Relationship between the environmental light level, and that detected by the sensor node.

In Fig. 6-14, the stochastic nature of the light sensor circuit can be seen by the variation in slope on the various traces (each trace represents the response of a different node's light sensing circuit). Additionally from this graph, it can be seen that there are two output (sensed) results for every input (environmental) value; this demonstrates the temperature dependence of the model as a result of the temperature sweep that was simulated across the simulated environment (each input value is sampled twice, as shown in 6-13). Furthermore, the switching of the light sensing circuit's sensitivity range when it reaches 1000lx can also be seen by the increase in spacing (a result of the ADC's digital quantisation).

6.1.4 Discussion

These simulation results have shown that a significant extension in the network lifetime can be obtained for important packets at the possible expense of less important packets. Obviously, the numerical results obtained in this section are only correct for this specific simulation. However, the general trends and observations are common for the majority of applications and, with careful planning, and parameter selection, it is believed that it is possible to retain a connected and operational network of 100% for the most important packets.

This section has also allowed a comparison between the packet flooding and 'minimum cost' routing algorithms. The obtained results show that, for networks of this scale, packet flooding is not too inefficient. This is because, with flooding, all nodes incur one packet transmission per packet communicated through the network. MCR however has approximately two nodes (on average) transmitting two packets per packet communicated through the network (plus any retransmissions). 'Minimum cost' routing also depletes the energy reserves of nodes that are closest to the sink faster than outlying nodes. Furthermore, packet flooding offers minimum latency communication and introduces redundancy. Due to the nature of IDEALS/RMR, it operates more effectively in this scenario with packet flooding.

6.2 Realistic Scenario: Temperature Monitoring at a Water Pumping Station

The simulations of the static network conditions shown above allowed a controlled and comparable investigation of IDEALS/RMR. However, to investigate the operational functionality of IDEALS/RMR with event-based reporting, a more realistic scenario is required. For this reason, this section presents the results from a realistic [186, 193], simulated scenario of temperature monitoring in a water pumping station.

6.2.1 Simulation Scenario, Setup and Parameters

In this industrial scenario, a WSN is used to monitor the temperature of 20 water pumps (a relatively small network). The locations of the nodes were fixed for all simulations after being initially randomly generated, and the nodes in the network turn on at random times during the first minute of simulation. A snapshot of the network under simulation is shown in Fig. 6-15. In this snapshot, the small dots represent sensor nodes, the dotted lines represent links between nodes where the separation distance is less than 20m (this is the distance considered by the 'minimum cost' routing algorithm, and the colour map depicts the energy residual in the nodes' energy stores (all nodes have full energy reserves in this snapshot as it was taken at zero seconds). Hops across distances further than 20m may be possible when using packet flooding, but will be obtained with a lower packet success rate (due to a higher PER). An environment with dimensions of 70m square was chosen to enforce multi-hop routing throughout the network (when using the propagation model proposed in Appendix A). Naturally, this effect could have been obtained using smaller dimensions and scaling the transmit power accordingly. In an industrial monitoring scenario such as this, it could be thought that ~100% reliability in packet transmissions is important; hence, the deliberate discarding of packets may cause concern. However, no locally powered system is capable of meeting this requirement under all conditions and, by selectively discarding low importance packets, it is hoped that IDEALS/RMR increases the reliability of these packets that are deemed to be important.

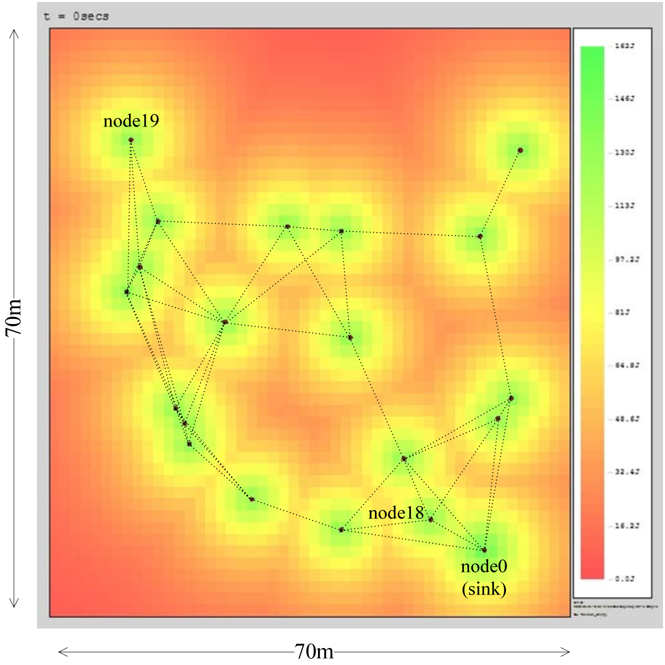


FIGURE 6-15 : A snapshot of the water pumping station network under simulation.

All nodes report packets to a single sink node (node 0), which is labelled in Fig. 6-15. In addition to considering the mean behaviour of the network, some of the results in this section also consider the operation of nodes 18 and 19 in order to evaluate a node a single hop from the sink node and at the network extreme respectively.

In the simulated scenario, every pump in the station operates once daily for a random period of time (between one and 12 hours), with an ambient temperature of 25ºC and a normal operating temperature of 50ºC. It takes around two hours for the sensed temperature of the pump to change from ambient temperature to within 10% of the normal operating temperature. Sensors are duty-cycled, following the same cycle as the MAC algorithm, and are hence sampling with a temporal resolution of around one second. The simulation is run over a period of six days. During pump operation, mechanical vibrations occur locally at a fixed frequency, allowing a maximum of 4.5mW of harvested power to supplement the nodes' energy stores. This is calculated from a pump vibrating at 0.1'**g**' (where **g** is 9.81m/s^2), which equates to a harvested power of around 4.5mW [31, 186]. While the pump is inactive, no vibrations occur, and no energy can be harvested (this consumes negligible energy). This means that a node could harvest a maximum of 194J in a day; however due to the random nature of the pump duration it may only harvest 9.7J, and then nothing for a subsequent period of nearly 48 hours. This variation allows the investigation of the operation of the network while it operates for intermittent periods with little harvesting.

The simulated nodes contain two 50F supercapacitors in series, equating to an energy capacity of 162J; this is considerably smaller than the capacity of a standard AA size battery. The 50F dual supercapacitor store was selected as nodes must operate for up to 48 hours without being recharged. The 4.7F supercapacitor store was unsuitable as it leaks to below a usable 2V within a 24 hour period. The 10F supercapacitor store was not used as through the empirical experiments made it did not demonstrate supercapacitor leakage and would not highlight the ability of IDEALS/RMR to manage this (leakage is usually apparent).

TABLE 6-3 : RMR rules used in the water pumping station scenario.

Function	Name	Type	PP	Value		Parameter
Routine Rules	R(10M)pp7	Routine	7	10	mins	n/a
	R(1H)pp6		6	1	hrs	
	R(3H)pp5		5	3		
	R(6H)pp4		4	6		
	R(12H)pp3		3	12		
Operational Event Detection	T(37.5)pp3	Threshold	3	37.5	ºC	1.0ºC
	T(27.5)pp2		2	27.5		
	T(47.5)pp2			47.5		
	T(20)pp2			20		
	T(55)pp2			55		
Deviation Event Detection	T(10)pp1	Threshold	1	10	ºC	1.0ºC
	T(70)pp1			70		
	T(80)pp1			80		
	T(90)pp1			90		
	T(100)pp1			100		
	D(5)pp1	Differential		±5		n/a

As the data reporting frequency is reduced in this network compared to the static network conditions, the MAC duty cycle is also reduced to a sleep period of one second (the active time, and hence total MAC cycle time, is dependent on the node's activity). While this has an effect on the latency of the network, it should remain in the order of seconds assuming good channel utilisation. Considering latency as a metric is outside the scope of this research. It is important to note that communications and sensing are duty cycled whilst energy harvesting occurs continuously when the pump is operating; hence although the harvested power is less than the power consumption of the node, it equates to a greater energy over a period of time.

Reporting rules are defined as shown in Table 6-3. These rules were created by inspecting the normal operating conditions of the pump, and anticipating various possible faults of varying importance. The rules are segregated into three functions to provide varying levels of operation depending on the energy state of the node. Routine rules (with the lowest importance, PP7-3) aim to provide a reasonably up-to-date and accurate picture of the network when abundant energy is available (these are chosen so that as the node's EP decreases the period of the routine rules increases, hence reducing the load on the node as it's residual energy decreases). Operation events are detected by threshold rules (with medium importance, PP2-3) to capture events that are expected to occur during normal operation of the pump. Finally, deviation events are detected by threshold and differential rules (with maximum importance, PP1) to capture faults and errors in the pumps and sensors of which the user requires notification. Temperature thresholds are given 1ºC hysteresis in order to suppress spurious detected events due to noise. While successful operation relies on the careful selection of parameters such as reporting rules and PP/EP threshold, it is believed that it should be a simple process to a designer having knowledge of the target application.

TABLE 6-4 : EP thresholds used in the simulation of the water pumping station.

Function	Energy Priority	Lower Bound	Upper Bound	Hysteresis
Node Maintenance	EP0	0%	5%	2%
Emergency Operation	EP1	5%	25%	
Normal Operation	EP2	25%	50%	
	EP3	50%	60%	
Routine Operation	EP4	60%	70%	
	EP5	70%	80%	
	EP6	80%	90%	
	EP7	90%	100%	

The EP thresholds are defined as shown in Table 6-4. These are designed to operate with the RMR rules shown in Table 6-3 to provide varying classifications of operation. Four EPs (EP4-7) represent the upper 40% of the node's energy reserve, which is used to cater for routine operation (operation when the node has an abundance of energy). EP2-3 use the subsequent 35% of the node's energy reserve to allow normal operation (operation that it is hoped can be sustained). Finally, the next 20% of the node's energy is used for emergency operation (operation to communicate problems with the network), and the final 5% for performing node maintenance in order to allow it to return to a higher EP. As in the static condition simulations (section 6.1), four configurations were simulated:

i) No energy harvesting or IDEALS/RMR

ii) IDEALS/RMR only

iii) Energy harvesting only

iv) Energy harvesting and IDEALS/RMR.

As the benefits of using event detection in this situation are obvious, RMR is enabled in all simulations; hence, rules will be triggered when events occur in all simulation. However, when IDEALS is disabled, the importance of the sensed data is ignored. The environmental and pump data were designed to reflect expected conditions in the scenario. Two different sets of data were simulated for the pumping station (outlined in sections 6.2.1.1and 6.2.1.2):

Normal Operating Conditions: Representing the normal and expected operating conditions and the pumping station, operating with no faults.

Fault Conditions: A range of faults occur in the pumping station.

6.2.1.1 Normal Operating Conditions Data

The normal operating conditions data represent the case where all of the pumps and nodes are operating correctly. The data were generated using random pump on times and durations (generated between bounds of one and 12 hours), with exponential growth and decay. Uniform noise with a maximum amplitude of 0.5ºC was added to the sensed temperature to represent temperature fluctuations. The data generated for a node for the duration of the simulation can be seen in Fig. 6-16, where the shaded areas represent the occurrence of vibration energy harvesting at the node (when enabled).

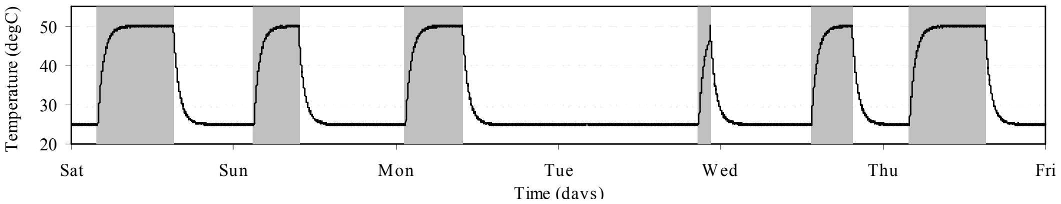


FIGURE 6-16 : Generated pump data for a node under normal operating conditions.

These data were simulated to investigate the magnitude of the network lifetime extension possible through the use of IDEALS/RMR under normal (no fault) conditions.

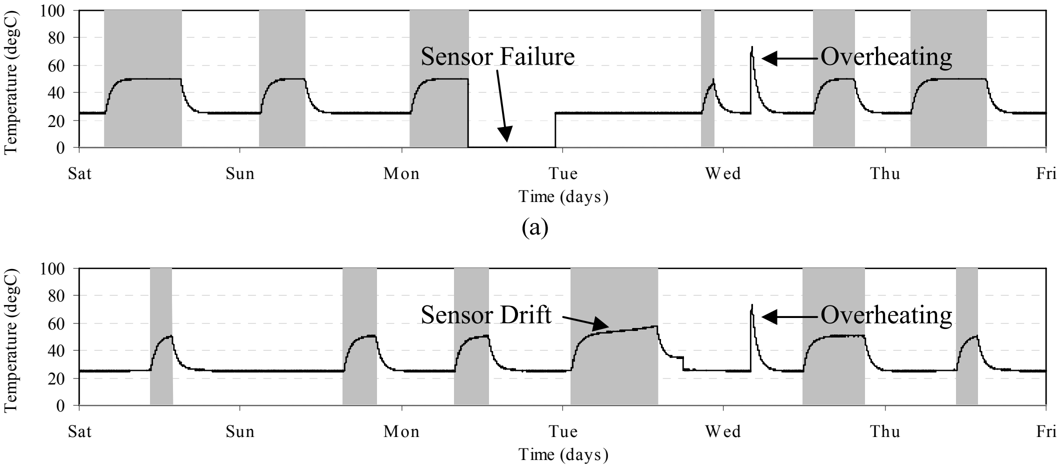
6.2.1.2 Fault Operation Data

The fault operation data includes multiple pump and sensor faults occurring during the simulation. The data were generated in the same way as for the normal operating conditions. However, faults (detailed in Table 6-5) were added to investigate how the network reacts to influxes of packets (and IDEALS/RMR's ability to maintain a controlled degradation of the network), and to see if RMR portrayed an accurate representation of the events at the sink.

TABLE 6-5 : Faults occurring in the water pumping station 'fault condition' simulations.

Node/Pump	Fault	Observed As	Time Span (hrs)
Node 03	Sensor Failure	Reads 0ºC	21-49
Node 06	Rapid Sensor Drift	Reading Decreases	25-49
Node 08	Sensor Failure	Reads 0ºC	58-71
Pump 10	Bearing Failure	Temperature Rise of 8ºC	32-55
Node 13	Rapid Sensor Drift	Reading Increases	72-90
All Pumps	Fire	Temperature Rise of 50ºC	100-110

Fig. 6-17 shows fault data generated for a) node 08 (showing sensor failure and overheating), and b) node 13 (showing sensor drift and overheating).



(a)

(b)

FIGURE 6-17 : Generated pump data for a) node08, and b) node13, for fault conditions.

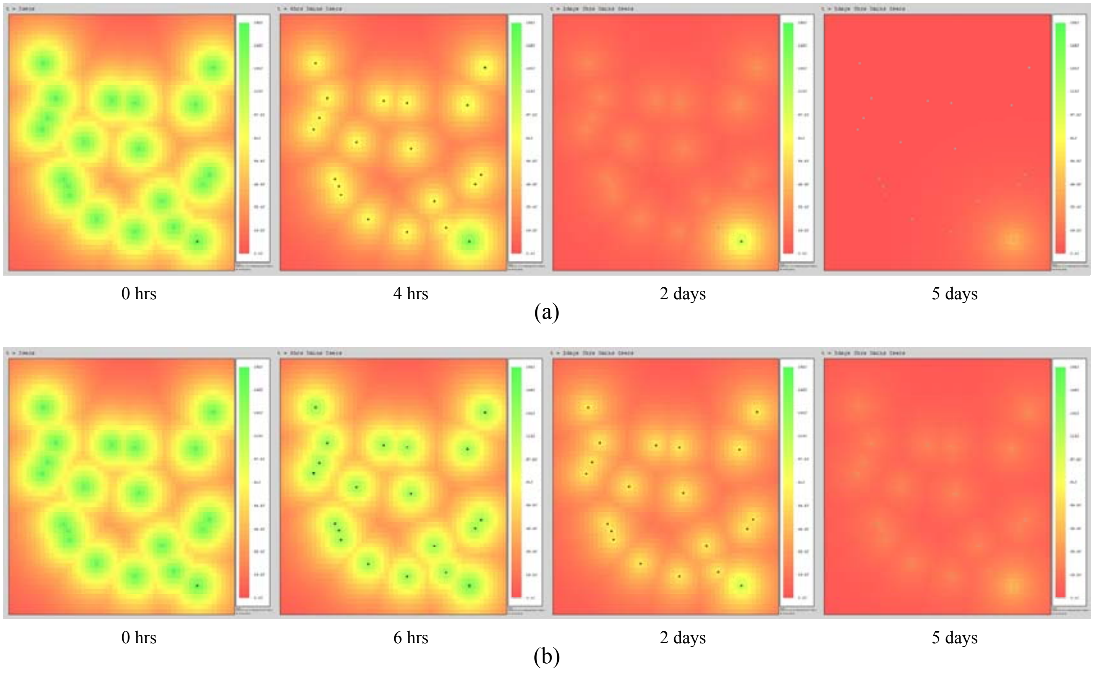
The remainder of this section presents results using a variety of different metrics relevant to the investigated areas of energy and information.

6.2.2 Energy

This section presents an analysis of IDEALS/RMR from the perspective of energy.

6.2.2.1 Normal Operating Conditions

In order to understand the behaviour of the network, this section first presents pictorial graphics depicting the decline of the network under different conditions. These graphics (such as those shown in Fig. 6-18) show a plan view of the network at different times during simulation. The colour map shows the nodes' residual energies (green depicts full energy stores, while red depicts no energy), and node state (a dark dot represents a node that is turned on, while a light dot represents a node that has turned off).



0 hrs 4 hrs 2 days 5 days

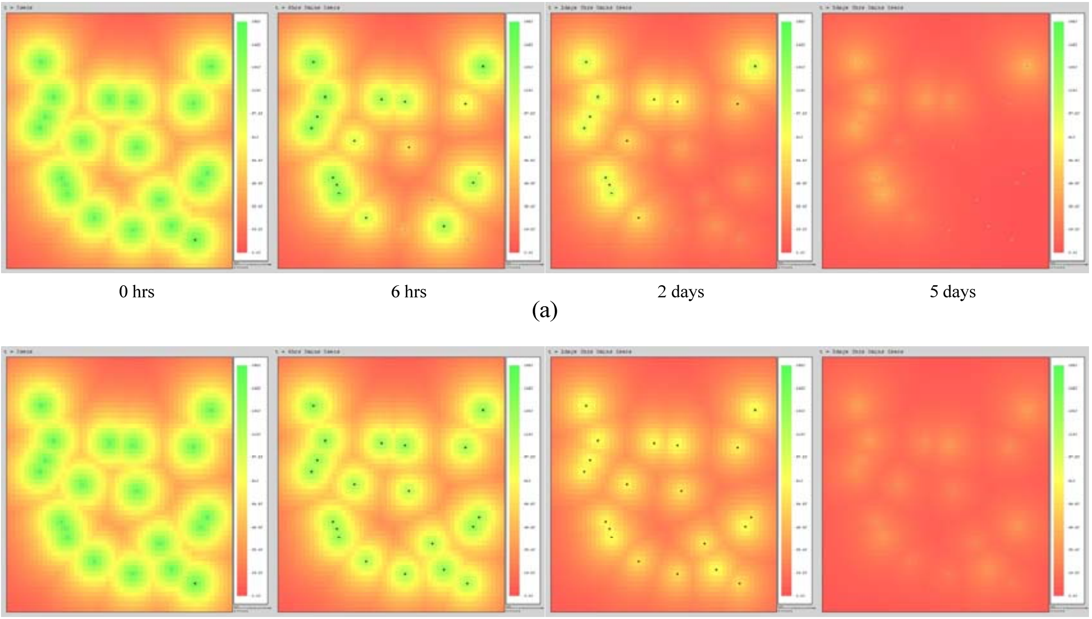
(a)

0 hrs 6 hrs 2 days 5 days

(b)

FIGURE 6-18 : Deterioration of nodes' energy levels at different stages in the network's operation using packet flooding and no energy harvesting with a) no IDEALS/RMR, and b) IDEALS/RMR.

In Fig. 6-18, it can be seen that all the nodes in the network (except the sink node, which does not incur any energy consumption due to packet transmissions) decline at a similar rate regardless of the presence of IDEALS/RMR; this is due to the inherent nature of packet flooding which means that every node is forwarding every packet. In Fig. 6-18a, all of the nodes in the network except the sink have depleted their energy stores (the store voltage has dropped below 2V) by two days and have turned off, although the supercapacitor energy store continues to leak. However, with IDEALS/RMR (Fig. 6-18b), the decline of the network has been managed, and all nodes remain operational at two days; this is achieved by 'dropping' less important packets, which are, in this case, 'routine' packets.



(a)

(b)

FIGURE 6-19 : Deterioration of nodes' energy levels at different stages in the network's operation using 'minimum cost' routing and no energy harvesting with a) no IDEALS/RMR, and b) IDEALS/RMR.

In Fig. 6-19a, it can be seen that 'minimum cost' routing results in the nodes around the sink node depleting their energy reserves quickly; indeed the sink node and the majority of its neighbours (that provide routes to the rest of the network) have all turned off. However, with the inclusion of IDEALS/RMR (Fig. 6-19b), this effect is controlled, and all nodes in the network are still operational after 2 days of operation.

Fig. 6-20 shows the mean residual energy level in the network under both packet flooding (neglecting the energy in the sink node due to the reasons previously mentioned) and 'minimum cost' routing. It can be seen that in case i), where nodes are not harvesting of using IDEALS/RMR, the network lifetime is under half a day using packet flooding and under quarter of a day using 'minimum cost' routing. Nodes in the network using 'minimum cost' routing deplete their energy stores quicker through loss of connectivity due to nodes closest to the sink consuming more energy than outlying nodes. With IDEALS/RMR enabled, the lifetime using either routing algorithm is extended to around three days as a result of the controlled degradation of the network. Likewise, with the presence of intermittent energy harvesting, after around half a day (or quarter of a day in the case of ‘minimum cost’ routing), operation becomes intermittent in line with the presence of harvesting. Although it looks as though these nodes have around 75J in their energy stores, this is the average and, in fact, some nodes have full reserves (150J) while others have ‘depleted’ reserves (<50J).

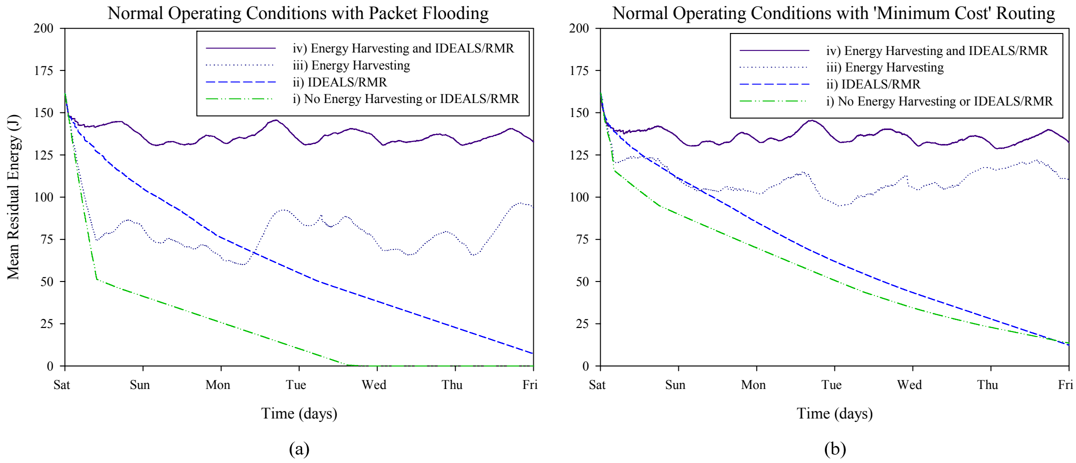


FIGURE 6-20 : Mean residual energy levels for the nodes in the network utilising a) packet flooding, and b) ‘minimum cost’ routing under normal operating conditions.

6.2.2.2 *Fault Conditions*

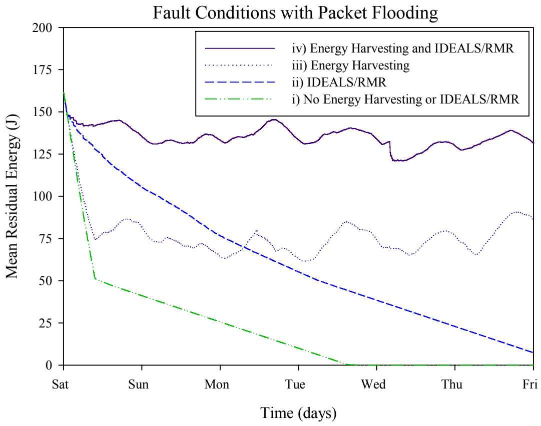


FIGURE 6-21 : Mean residual energy levels for the nodes in the network utilising packet flooding under fault conditions.

Fig. 6-21 shows the mean residual energy level in the network for the network featuring packet flooding (again neglecting the energy in the sink node), and simulated with the fault conditions dataset. When compared to Fig. 6-20, it is virtually identical (as would be expected), although most noticeable is the sharp drop in trace ‘iv’ on Wednesday morning.

This is due to the influx of packets that are transmitted as a result of the simulated fire that all nodes in the network are reporting. This occurrence demonstrates that, not only is the network able to respond to and report this fault but, also, IDEALS/RMR handles the subsequent operation of the nodes in order to regain the state of the network; this can be seen by the similarity in energy levels in Figs. 6-20 and 6-21 during Thursday.

6.2.3 Information

This section presents an analysis of IDEALS/RMR from the perspective of information.

6.2.3.1 Normal Operating Conditions

In order to visualise the usefulness of the data, or the 'information' transmitted through the network, this section presents the packet success under different network configurations. In these graphics (such as Fig. 6-22), the black line shows the actual variation of the temperature in the environment. The vertical red lines (and the crosses at the top of them) represent values that are received from this node at the sink node. Finally, the shaded areas represent areas where the pump is active (and hence energy harvesting is possible when enabled).

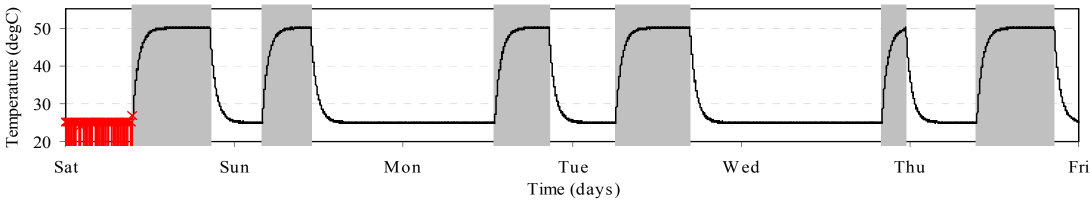


FIGURE 6-22 : Success of packets transmitted from node 19 in a 'normal operating conditions' network utilising packet flooding with no IDEALS or energy harvesting.

In Fig. 6-22, packets are sent every 10 minutes (through routine rule "R(10M)pp7") until the node has depleted its energy store. As the node is transmitting at high frequency without harvesting energy or managing energy, it depletes its energy store without reporting anything of real interest to the sink node. In Fig. 6-23, the network features IDEALS/RMR, and hence is able to manage the node's energy. A number of packets are sent during the first hour of operation as the energy in the nodes' stores deplete. The normal operation rules then come into effect to allow the monitoring of event operation (through various threshold rules). This allows two days of useful data to be communicated to the sink node before the network dies.

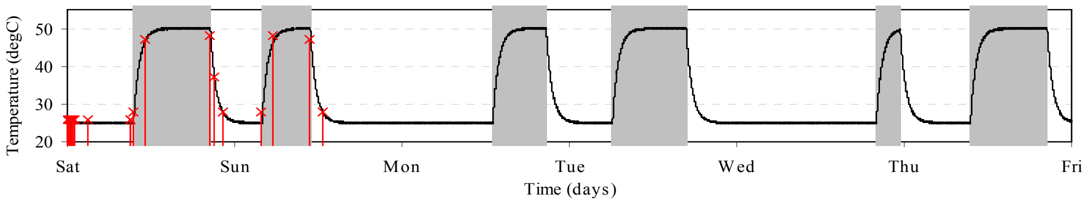


FIGURE 6-23 : Success of packets transmitted from node 19 in a 'normal operating conditions' network utilising packet flooding with IDEALS, but no energy harvesting.

The above results (packet success for networks not featuring energy harvesting) are not included for the other network configurations shown in this section, as the energy budget is insufficient to adequately visualise operation. However, it should be mentioned that similar results are obtained in each case, where the node transmits continuously and depletes before the pump starts (in the case of no IDEALS/RMR), or successfully detects the first two days of pump operation with declining resolution before depleting (in the case of IDEALS/RMR).

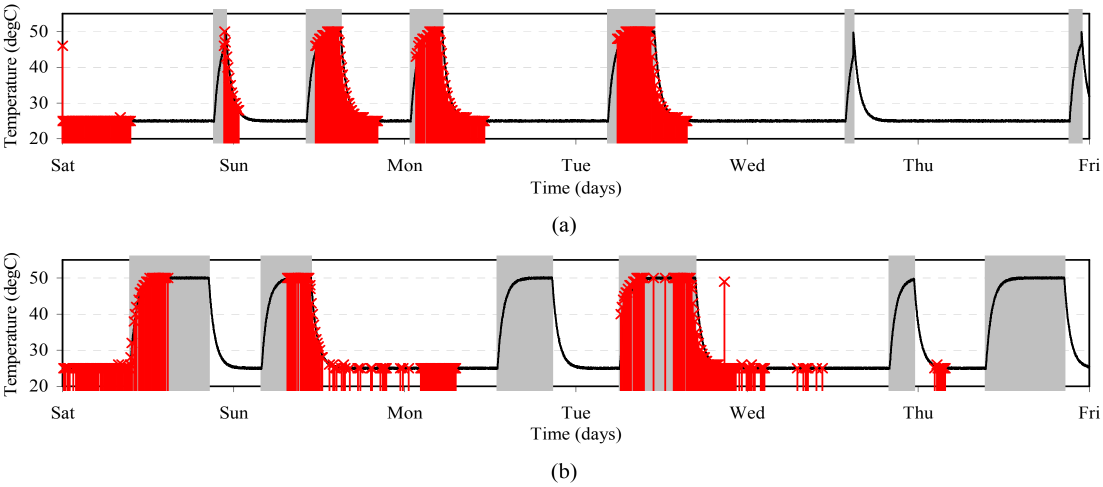
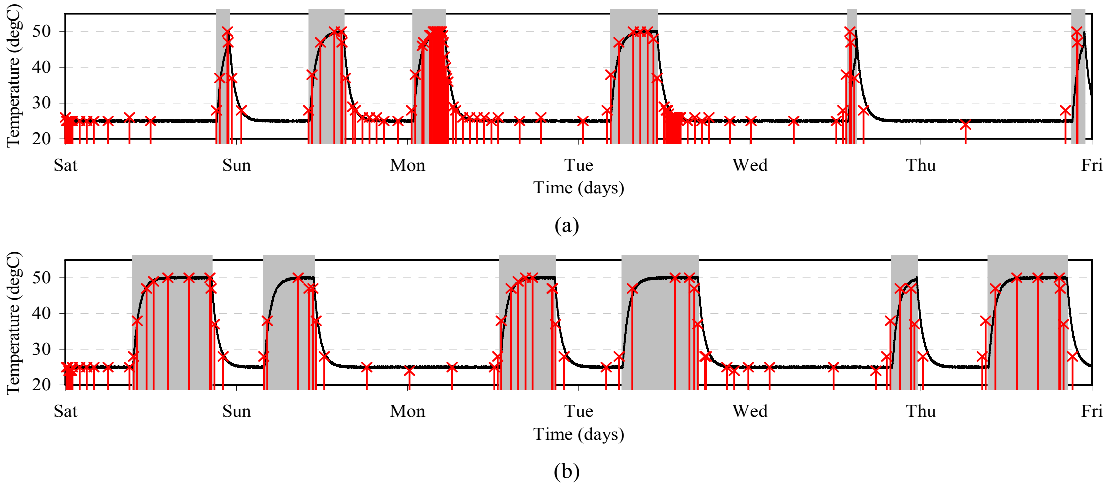


FIGURE 6-24 : Success of packets transmitted from a) node 18 (close to sink), and b) node 19 (distant), in a 'normal operating conditions' network utilising packet flooding with energy harvesting but no IDEALS.

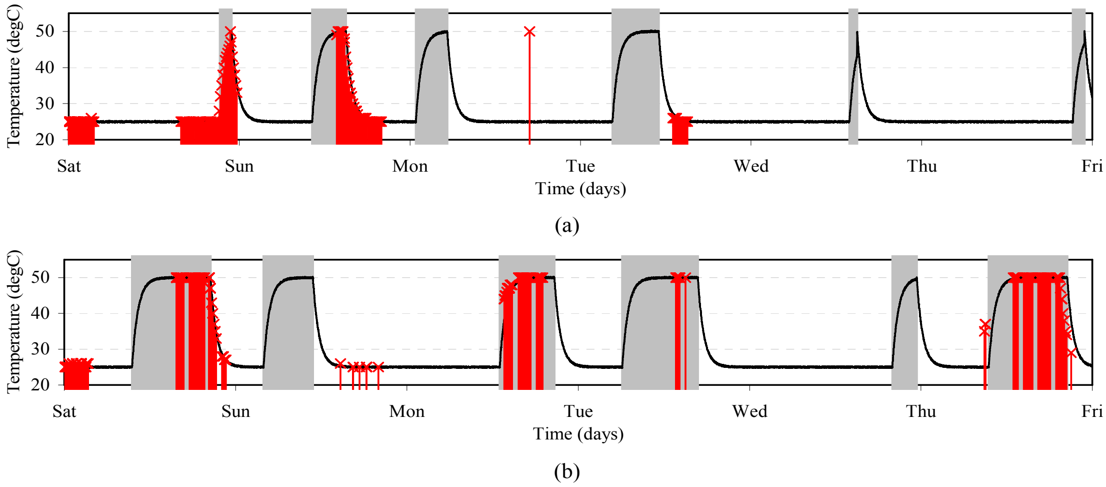
In Fig. 6-24, the nodes are continuously sending packets every 10 minutes (through routine rule "R(10M)pp7"), though this is happening intermittently as and when nodes harvest energy through energy harvesting. It can be seen that, while node 18 (only one hop from the sink node) successfully communicates all packets to the sink node, node 19 (a distant node) drops packets as they are not able to find a route through the network. Furthermore, an undetected packet error (one that does not result in an invalid checksum) can be noticed just before Wednesday. In Fig. 6-25, IDEALS manages the network operation. Node 18 (Fig. 6-25a) successfully reports pump operation for the entire simulation and, when there is abundant energy due to energy harvesting, reports expends more energy in reporting. Node 19 (Fig. 6-25b) is able to successfully communicate as many packets, as the connectivity is not present in the network to send low importance packets on demand. However, the network is still able to successfully report pump operation for the entire simulation with good resolution.



(a)

(b)

FIGURE 6-25 : Success of packets transmitted from a) node 18 (close to sink), and b) node 19 (distant), in a 'normal operating conditions' network utilising packet flooding with IDEALS and energy harvesting.

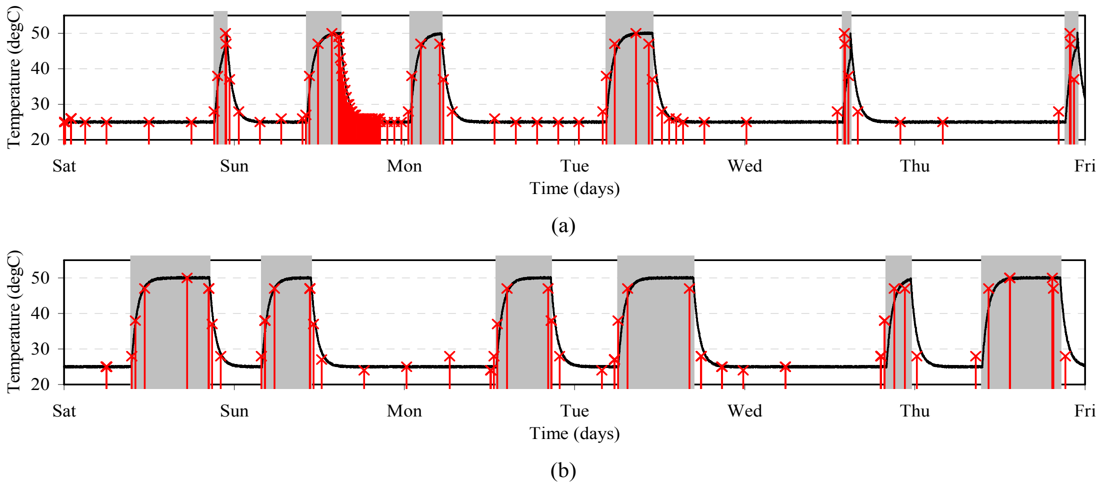


(a)

(b)

FIGURE 6-26 : Success of packets transmitted from a) node 18 (close to sink), and b) node 19 (distant), in a 'normal operating conditions' network utilising 'minimum cost' routing with energy harvesting but no IDEALS.

Fig. 6-26 shows the packet success of a network featuring energy harvesting but not IDEALS with 'minimum cost' routing. Comparing this with Fig. 6-24, it can be seen that packet success is considerably reduced. This is due to 'minimum cost' routing losing connectivity quicker than packet flooding by depleting the energy stores of nodes closest to the sink node first. Fig. 6-27, however, shows the improvement that can be obtained through the use of IDEALS, which manages the energy of all nodes in the network. It can be seen that the addition of IDEALS results in 'minimum cost' routing successfully reporting pump operation for the entire simulation period, and the behaviour is similar to that of packet flooding (shown in Fig. 6-25).



(a)

(b)

FIGURE 6-27 : Success of packets transmitted from a) node 18 (close to sink), and b) node 19 (distant), in a 'normal operating conditions' network utilising 'minimum cost' routing with IDEALS and energy harvesting.

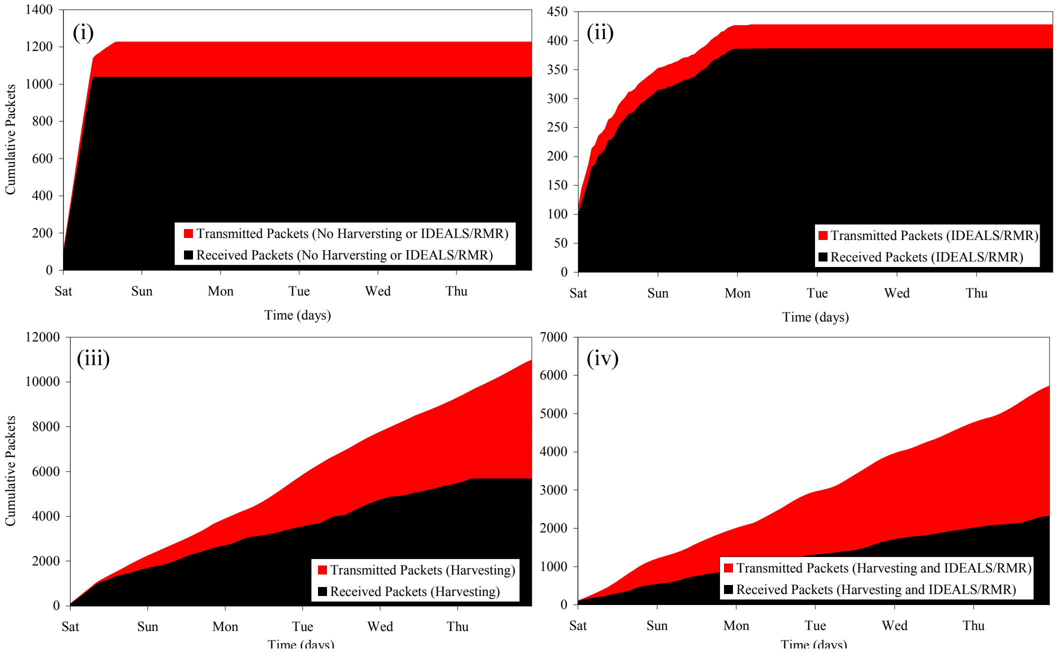


FIGURE 6-28 : Cumulative packet throughput under 'normal operating conditions' with packet flooding.

Fig. 6-28 shows the cumulative packet throughput of the 'normal operating' conditions under packet flooding for the four network configurations. These graphics show the total number of packets that have been sent and received by the sink node during the simulation. It can be seen that when no energy harvesting or IDEALS is present (Fig. 6-28a), 1040 packets are received before the network dies. When IDEALS is enabled (Fig. 6-28b), a considerable extension in the lifetime of the network is achieved, but this is at the expense of only 387 packets being communicated. The operation and benefit of this reduction can be seen in Fig. 6-29, which shows the average information that is transmitted per packet. Comparing traces 'i' and 'ii' in Fig. 6-29, it can be seen that while only a third of the number of packets are transmitted, those that are contain considerably higher information content.

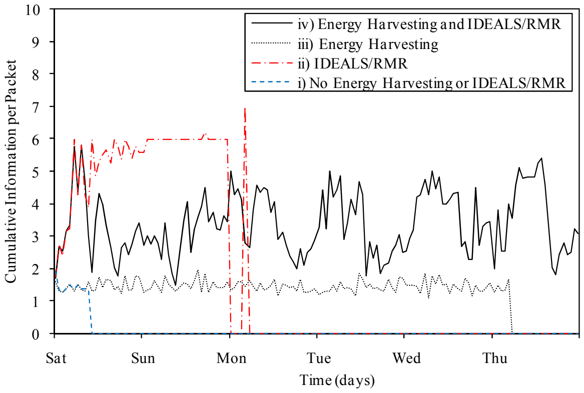
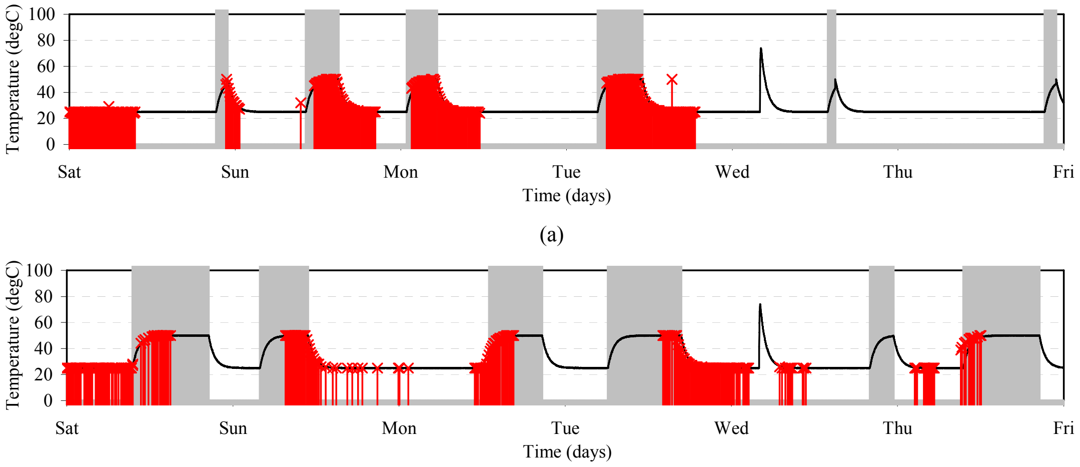


FIGURE 6-29 : Average information per packet under 'normal operating conditions' with packet flooding.

When energy harvesting is present but IDEALS is not enabled, 5685 packets are successfully received by the sink node during the simulation. However, when IDEALS is enabled, only 2328 packets are communicated. Again, by comparing traces 'iii' and 'iv' in Fig. 6-29, it can be seen that while under half the number of packets are transmitted, those that are transmitted contain a higher information content.

6.2.3.2 Fault Conditions

Under fault conditions, the network featuring energy harvesting but no IDEALS (6-30) achieves only intermittent and sporadic reporting of pump operation. Many operational features, and faults (in this case the simulated fire) go undetected and unreported. This is as expected when the results are compared with 6-24, which exhibited identical results under normal operating conditions.

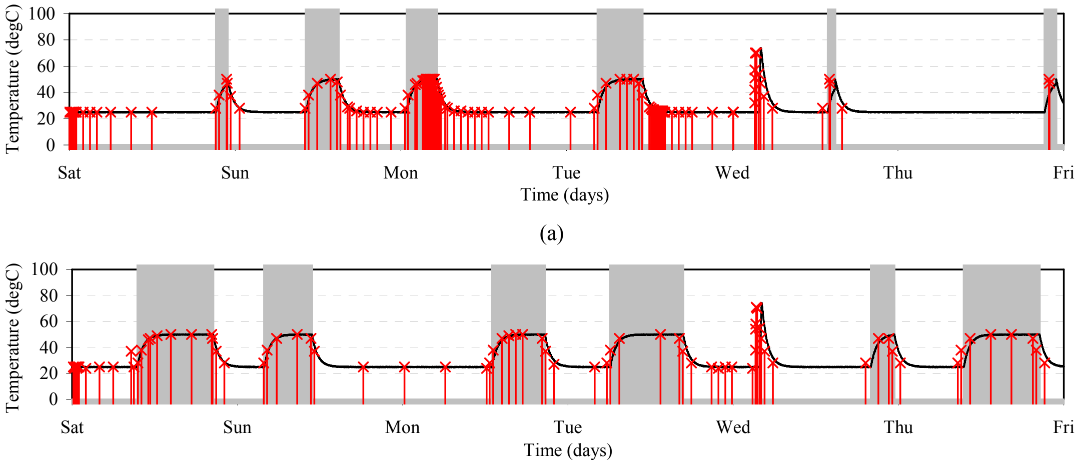


(a)

(b)

FIGURE 6-30 : Success of packets transmitted from a) node 18 (close to sink), and b) node 19 (distant), in a ‘fault condition’ network utilising packet flooding with energy harvesting but no IDEALS.

When IDEALS and energy harvesting are present in the network (6-31), the operation of the network is coordinated in such a way that all operational events and faults of interest and relevance to the end-user are successfully reported. Again, as expected (when comparing results to those presented in 6-25), the node closest to the sink achieves a greater temporal reporting resolution as it does not rely on the availability of intermediate nodes. However, the distant node reports all occurring pump state changes and faults, highlighting the ability of IDEALS/RMR to successfully manage a node’s energy and information resources.



(a)

(b)

FIGURE 6-31 : Success of packets transmitted from a) node 18 (close to sink), and b) node 19 (distant), in a ‘fault condition’ network utilising packet flooding with energy harvesting and IDEALS.

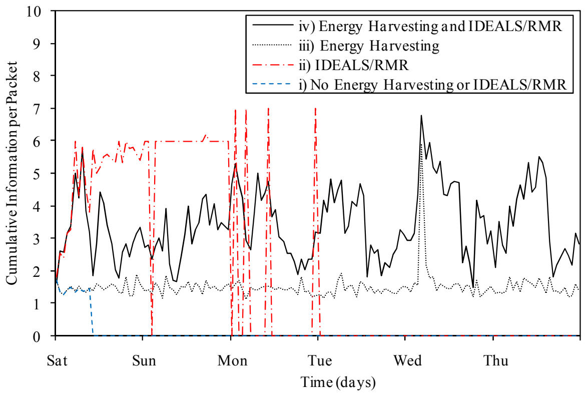


FIGURE 6-32 : Average information per packet under 'fault conditions' with packet flooding.

Fig. 6-32 shows the mean information per packet that is transmitted under the fault conditions network. Comparing these results with Fig. 6-29, the most notable difference is the peak that occurs for the energy harvesting networks (and therefore the networks that are still operational) at the time of the simulated fire occurring on Wednesday. This, and other spikes that are clearly visible in network 'ii', are due to faults occurring in the environment causing the generation of high importance packets.

6.2.4 Discussion

This section has presented the results obtained for the water pumping station scenario. In agreement with the results obtained in section 6.1, these results have shown that IDEALS/RMR permits a significant extension in the network lifetime for important packets at the possible expense of less important packets. Furthermore, these results have reiterated that applicability and benefits of packet flooding to network operation, in particular its ability to evenly distribute the energy cost while providing redundancy and reduced latency.

IDEALS/RMR is able to manage the reporting activities of the node in order to conserve energy and expend it on conveying information that is off particular use to the end-user. This has been demonstrated in this section, which has highlighted the ability of IDEALS/RMR to report only specific events and, as a result, extend the network lifetime. When coupled with energy harvesting, IDEALS/RMR manages these additional energy resources, and controls the operation of the network to sustain the requirements of the end-user.

Chapter 7

Conclusions

7.1 Summary of Work

Wireless Sensor Networks typically consist of a number of small, inexpensive, locally powered sensor nodes that communicate detected events wirelessly via multi-hop routing. WSNs are continuing to receive increasing research interest, largely due to the wide range of applications to which they are suited. A key research area is concerned with extending the limited network lifetime, inherent to the sensor nodes, which is being attempted from all areas of WSN development. This thesis has addressed the issue of energy-efficient operation by considering both energy-management and information-management, and the developed algorithms have been simulated using a custom simulator that is structured around environmental and physical models.

Chapter 3 investigated the energy- and information-aware operation themes of this thesis through the proposal of IDEALS/RMR, an application independent, localised system to control and manage the degradation of a network through the positive discrimination of packets. IDEALS/RMR achieves this through the novel combination of information management (assessed through RMR) and energy management (controlled by IDEALS), which increases the network lifetime at the expense of the possible sacrifice of low information packets. The IDEALS/RMR system is particularly suited to applications where sensor nodes are small, energy constrained (particularly those that feature one or more sources of energy harvesting) embedded devices that are required to report data in an unassisted fashion. It is envisaged that under normal conditions, the network will harvest as much energy as it depletes, and so IDEALS will appear transparent. However, if an influx of packets occurs as a result of a significant event, IDEALS/RMR will manage the decline of the network to maintain its usefulness, and return it to full functionality once the event has passed.

Chapter 4 investigated environmental and physical models for WSN simulation, considering communication, energy, sensing and timing models. While communication models have received significant research interest, the proposed model in this research pays particular attention to packet receipt in order to consider packet losses that affect the node's reporting performance. Energy models have been proposed for energy stores (in particular, the supercapacitor energy store has received little consideration elsewhere for WSN simulation), energy consumers, and energy sources. Sensing models have been developed for a digital temperature sensor and a light sensing circuit, and consider errors and inaccuracies in the sensing process. Finally, the developed timing model addresses issues associated with timing delays, oscillator error and temperature dependence. These four areas of modelling have been considered in order to address and evaluate the areas of energy-aware and information-aware operation; if alternative metrics were being considered, it is likely that other models would be required in order to provide accurate and realistic results.

Chapter 5 has outlined WSNsim, a simulator for WSNs that has been developed to evaluate IDEALS/RMR using the environmental and physical models developed in Chapter 4. WSNsim is structured around a novel embedded architecture for sensor nodes, which uses multiple layered stacks to structure the node's functionality. This architecture, which can be translated from simulation to deployment, aims to permit efficient code reuse, and promote the design of standardised, interchangeable protocols for all aspects of a sensor node.

Chapter 6 has presented results obtained from WSNsim to evaluate IDEALS/RMR under both controlled conditions and as a realistic simulated scenario (temperature monitoring in a water pumping station). These results have shown that IDEALS/RMR provides a considerable extension to the network lifetime and, when coupled with energy harvesting, can permit perpetual operation. IDEALS/RMR has received interest external to this research, and it is encouraging that their practical evaluation has confirmed its ability to provide an increase in the network lifetime [194, 195].

This thesis has highlighted the importance of information management in the energy-aware operation of a sensor network. Through IDEALS/RMR, a method for reducing the energy consumption of a WSN has been investigated and proposed, using a novel combination of information-management and energy-management. This thesis has also considered the effect and highlighted the importance that various environmental and physical aspects of WSNs, including energy components (stores, sources and consumers), sensing devices, timing and wireless communication, can have on the operation of sensor nodes and networks. The effect of these aspects is evaluated and quantified through simulation, using a simulator (WSNsim) that has been developed as part of this research to integrate environmental and physical models, and a novel node architecture. The developed algorithms have been evaluated through simulation, and the obtained results demonstrate that significant energy-savings and beneficial energy-management can be obtained by considering the value of information.

7.2 Future Work

The research that has been undertaken for this thesis has successfully met the *research aims* proposed in section 1.3. However, the investigated research area of energy- and information-managed algorithms has been highlighted as an interesting and diverse aspect of WSNs, and there are many avenues along which additional research could be conducted to further this research. This section outlines some of these areas of future work.

7.2.1 Energy-Aware Operation

In IDEALS, the process of allocating energy priorities is currently performed by thresholding the residual energy level of the energy store(s). It is likely that the energy-aware operation of the node could be enhanced by also considering the amount of energy that is likely to be available (or not available) in the near future due to the dynamics of the energy harvesting environment. For example, if the energy store is depleting but the node predicts that it will shortly be harvesting plentiful energy, the node could adjust the EP thresholds in order to invest greater resources in current operation. This would extend the concept of environmental energy prediction techniques proposed by Kansal *et al.* [36, 49] (discussed in Chapter 2) by developing general purpose resource-light algorithms that intelligently influence the EP allocation process. The implementation of such functionality would be located within the energy management stack of a ‘unified’ stack, with the ECO layer performing harvesting prediction and dynamic EP allocation.

In this thesis, EPs have been used only to control the process of packet generation and forwarding. The energy-aware operation of the nodes could be further enhanced by using the EP to control other operational aspects, such as the sampling rates of energy-intensive sensors, or the duty cycles and sleep periods of MAC algorithms, essentially extending the concept of offering differing service levels to different EPSNs. Furthermore, it would provide the application designer with control and management over the energy consumption that is not related to packet transmissions and data reporting.

While the development of IDEALS originally aimed to remain largely application and algorithm independent, a number of requirements and constraints are introduced due to the inherent nature of the algorithm. IDEALS' combination of localised operation and dynamically changing EPSNs results in a demand for communication protocols that are completely reactive. The behaviour of the EPSNs is operationally similar to a highly mobile or faulty network, whereby nodes continuously disappear and reappear from the EPSN as they change EP. As such, dissemination routes must be calculated reactively and on demand, requiring highly reactive energy-aware routing algorithms. Additionally, MAC protocols that can provide varying levels of service to different EPSNs (as described in the previous paragraph) are required. Alternatively, the use of wakeup radios removes the need for idle listening; hence reducing both transmit and receive power consumptions. Further research should also evaluate the operation of IDEALS/RMR with a wider range of applications, routing algorithms and MAC layers.

7.2.2 Information-Aware Operation

In order to extend and develop RMR, investigation is required into extending the range and capability of the available rules. Many possibilities for this can be learnt from query-based systems, for example by allowing a rule to depend on the combined outcome of other rules; for example, a high priority rule that is triggered when two other rules of a lower priority are triggered. The operation and implementation of the feature rule also requires investigation, which is required to provide advanced pattern detection through a computationally inexpensive algorithm (a key attribute of the IDEALS/RMR system).

As an alternative to the rule-based RMR system which requires an understanding of the environment and setup by the end-user, alternative methods of information quantification could be used with IDEALS. An overview of these predictive techniques is presented in Chapter 2. Furthermore, the use of these techniques would allow an information theoretic evaluation of the algorithms, whereby the true transfer of information could be analysed in simulation.

Information-aware operation could be extended through the investigation of appending data onto packets. If it is assumed that the length of a transmitted packet is not directly proportional to the consumed energy (for example, transmitting ten packets of ten bytes is more expensive than five packets of 20 bytes), the information throughput can be increased at little energy cost by appending many sampled data into a single packet. When IDEALS authorises a packet for transmission, data from every sensor on the node could be appended into the subsequently generated packet. Furthermore, instead of just adding the most recent data for each sensor, multiple history values could be reported per sensor; this creates a trade-off between the history length and the energy consumed. IDEALS could also append previously dropped packets onto transmitted packets; this could operate as a local process (whereby only packets generated at the node are buffered) or a network-wide process (whereby the node also buffers packets that IDEALS rejects forwarding for).

7.2.3 Modelling and Simulation

This thesis has highlighted the modelling of supercapacitor energy stores as a non-trivial operation, due primarily to their complex leakage characteristics and the variation between devices. Due to both the considerable differences between the operation of supercapacitors and batteries, and the increasing number of deployments that are using supercapacitors, the further modelling of supercapacitor energy stores is paramount in obtaining useful and dependable simulation results. Additionally, the behaviour of various consumers as the store voltage drops is needed, due to its particular relevance to supercapacitor powered nodes. Furthermore, a more detailed and generalised modelling of energy sources will allow simulation to evaluate and utilise the advances that are being made in source interfacing.

This thesis has considered the modelling of communications, sensing, energy (stores, sources and consumers) and timing, due to the influence that these had on the metrics that were being evaluated. Different simulations and scenarios however require the modelling of additional interfaces and parameters in order to effectively evaluate them; these could include locationing hardware, memory and computational resource limitations, and other physical phenomenon (such as those experienced in harsh environmental conditions).

WSNsim has provided an invaluable in-house tool for the evaluation of IDEALS/RMR and the developed environmental models. Following a number of enquiries regarding the public availability of WSNsim, its development and refinement in order to make it available as an open-source research platform is a future task; this would also help to highlight to end users the benefits of the proposed 'unified' architecture in algorithm design and implementation.

Appendix A

Principles of Wireless Communication

A.1 Modelling Radio Frequency Propagation

As its name would suggest, radio frequency communication between nodes requires the use of a wireless channel. Therefore wireless channel modelling must be, at the very least, rudimentarily understood in order to design, simulate or implement a WSN. Compared to their wired counterparts, propagation through wireless communication channels are notoriously difficult to predict and model because of their apparent randomness. This is due largely to objects in the environment causing obstructions, reflections, refraction and scattering [196]; for example, it is reported in Kumagai *et al.* [197] that the propagation distance improved by a factor of ten from the lab environment to the outdoor deployment location. Propagation models are often tailored for a specific environment and protocol, and are generated largely from experimental results and statistical modelling.

Signals propagating across a communication channel are affected by a distance dependant loss of power, referred to as the path loss. Path loss is defined as the ratio of the radiated power (the power radiated from the radio transmitter) to the received power [92]. This relationship is shown in (A.1), where PL is the path loss, P_T [dB] is the radiated power, P_R [dB] is the received power, and d [m] is the separation distance.

$$PL(d) = P_T - P_R(d) \tag{A.1}$$

In (A.1), PL, P_T and P_R are all expressed in decibels. Decibels are logarithmic units of relative measurement, that allow values to be added and subtracted in place of multiplying and dividing [198]. In this document, parameters expressed in decibels are denoted by an uppercase subscript (for example the radiated power P_T), while parameters in linear units have lowercase subscripts (P_t). Decibel values can be either 'relative' or 'absolute'. 'Relative' decibel values are dimensionless; the path loss (PL) is a 'relative' decibel value, specifying the ratio between the transmit and receive powers. 'Absolute' decibel values are relative to a fixed reference; the transmit and receive powers (P_T and P_R respectively) are 'absolute' decibel values, specifying powers relative to a reference power of 1mW (depicted by units of dBm). Values can be easily converted between W and dB using (A.2).

$$P[dB] = 10 \cdot \log_{10}(P[W]) \tag{A.2}$$

It can be seen that the maximum separation distance between two communicating nodes is limited by the maximum radiated power that can be transmitted by the transmitting node, and the minimum received power that can be successfully decoded by the receiving node. The maximum transmit power of a device is usually limited by the nominal output power of the radio transceiver, and by regulations (the maximum radiated power is limited in order to reduce interference with other devices and control associated health risks) made by national or international bodies such as the FCC. The receiver sensitivity and saturation power of the radio transceiver specifies the smallest and largest signal strengths respectively that can be received at the receiver to achieve a specific performance (normally related to a minimum acceptable bit error rate). Bit Error Rates (BER) are discussed further in section A.2.3. Once the nominal output power and receiver sensitivity are known, the maximum transmission distance between two nodes can be calculated using the path loss equation from an appropriate propagation model.

The remainder of this section investigates the modelling of radio propagation using Friis free space propagation, log-distance propagation, and two path loss equations tailored for modelling the propagation of IEEE 802.15.4 signals.

A.1.1 Free Space Propagation

In 1946, Harald T. Friis published his findings on the power transfer between two antennas in free space [199]. Friis' free-space equation is derived from Maxwell's equations, and describes electromagnetic propagation in free space; this therefore assumes line-of-sight (LOS) propagation and no reflections or obstructions [200]. The Friis free-space equation is given in (A.3), f [Hz] is the carrier frequency, c [m/s] is the speed of light in a vacuum, and G_t and G_r are the transmitted and receiver antenna gains respectively [92]. Generally, antenna gains are measured with reference to an isotropic radiator (an ideal antenna that radiates uniformly with unity gain). L is the system loss factor (representing losses in the transceiver circuitry [201]), and is always greater than or equal to unity.

$$P_r = \frac{P_t G_t G_r}{L}\left(\frac{c}{4\pi d f}\right)^2 \tag{A.3}$$

Alternatively, and often of more use, the received power can be expressed with decibel units (A.4).

$$P_R = P_T + G_T + G_R - 20\log_{10} f - 20\log_{10} d + 147.55 - L \tag{A.4}$$

Using (A.4) and (A.1), the path loss under free-space propagation can be calculated as (A.5).

$$PL = -G_T - G_R + 20\log_{10} f + 20\log_{10} d - 147.55 + L \tag{A.5}$$

The Friis free-space equation is only valid for distances greater than the Fraunhofer distance [196]. The Fraunhofer distance d_f [m] marks the minimum distance from the antenna to the far-field region (the area at which antenna effects can be neglected, and the propagating electromagnetic field behaves like an electromagnetic wave [202]). The Fraunhofer distance is given by (A.6), where D [m] is the largest linear dimension of the transmitter antenna, and λ [m] is the wavelength of the electromagnetic wave. From (A.6), it should be noted that for a given wavelength, the Fraunhofer distance (and hence the near-field radius) increases quadratically with the antenna dimension.

$$d_f = \frac{2D^2}{\lambda}\ , \qquad d_f \gg D\ , d_f \gg \lambda \tag{A.6}$$

In general, the Friis free-space propagation model is too simplistic for general use where obstructions and reflections dramatically effect propagation. However, more advanced models – such as the log-distance model (section A.1.2) – are often fundamentally based on the Friis free-space model.

A.1.2 The Log-Distance Model

Theoretical and experimental propagation models have shown that the transmitted signals decrease logarithmically with distance. A widely used and adapted propagation model is the log-distance model (A.7), where $\overline{PL}$ is the mean path loss, d_0 [m] is a reference distance in the far-field of the transmitting antenna, and η is the path loss exponent.

The reference path loss $\overline{PL}(d_0)$ is usually obtained via measurements, or by using the Friis free space model (section A.1.1) [196]. The path loss exponent η represents the rate at which the path loss increases with distance. Constructive interference can result in a path loss exponent that is smaller than that in free-space (η < 2) [92].

$$\begin{aligned}\overline{PL}(d) &\propto 10\eta \log_{10}\left(\frac{d}{d_0}\right) \\ &= \overline{PL}(d_0) + 10\eta \log_{10}\left(\frac{d}{d_0}\right) , \qquad d \geq d_0\end{aligned} \tag{A.7}$$

Due to channel fading, the path loss between two points will fluctuate around the mean path loss. This can be accommodated for in the log-distance model by using log-normal shadowing [196]. This is included through the introduction of a shadowing variance, a normally distributed random variable X_σ with variance σ^2 (A.8).

$$\begin{aligned}PL(d) &= \overline{PL}(d) + X_\sigma \\ &= \overline{PL}(d_0) + 10\eta \log_{10}\left(\frac{d}{d_0}\right) + X_\sigma , \qquad d \geq d_0\end{aligned} \tag{A.8}$$

Experimental measurements for both the path loss exponent and standard deviation for a variety of environments are given in Table A-1 [196].

TABLE A-1 : Path loss exponent and standard deviation measurements for various propagation environments (adapted from Rappaport *et al.* [196]).

Environment		Freq (MHz)	η	σ
Office	Hard Partitions	1500	3.0	7.0
	Soft Partitions	900	2.4	9.6
		1900	2.6	14.1
Factory (Line Of Sight)	Textile/Chemical	1300	2.0	3.0
		4000	2.1	7.0
	Metalworking	1300	1.6	5.8
Factory (Obstructed)	Textile/Chemical	4000	2.1	9.7
	Metalworking	1300	3.3	6.8

This model can be extended to model wireless propagation through floors of buildings. This can be included by providing values for η and σ that are a function of the number of floors, or by using a set of floor attenuation factors (FAF) that are added to the path loss [203, 204].

A.1.3 IEEE 802.15.4 Path Loss Models

As previously discussed, the IEEE 802.15.4 standard [84] is particularly relevant to WSNs. This section outlines two different path loss models for IEEE 802.15.4 at 2.4GHz, both of which are based upon the log-distance path loss model.

A.1.3.1 Path Loss Model 1

The IEEE 802.15.4 specification [84] specifies a path loss model for communication at 2.4GHz (A.9). The lower bound of 0.5m is added from the IEEE 802.15.2 specification [205] (upon which the model is derived). Through inspection this model can be seen to specify free space LOS propagation for the first eight meters (using Friis free-space propagation), and the log-distance path loss model with d_0 = 8m and η = 3.3 (with no log-normal shadowing) for distances greater than this.

$$PL(d) = \begin{cases} -G_T - G_R + 40.2 + 20\log_{10}(d) & ,0.5m \le d \le 8m \\ -G_T - G_R + 58.5 + 33\log_{10}\left(\frac{d}{8}\right) & ,d > 8m \end{cases} \tag{A.9}$$

A.1.3.2 Path Loss Model 2

Alternatively, a report from the IEEE 802.15.4 working group suggests an indoor propagation model for IEEE 802.15.4 at 2.4GHz [177]. This model specifies free space LOS propagation for the first four meters, with additional loses of 0.7dB/m for distances greater than this. Additionally, a fading margin of 16dB is specified, and log-normal shadowing with σ = 3dB and 9dB at 8m and 100m respectively. The path loss equation is shown mathematically in (A.10); σ is derived through linear interpolation to the expression $\sigma(d) = 0.065d + 2.48$.

$$PL(d) = \begin{cases} -G_T - G_R + 56.2 + 20\log_{10}(d) + X_{\sigma(d)} & ,0.5m \le d \le 4m \\ -G_T - G_R + 56.2 + 20\log_{10}(d) + 0.7(d-4) + X_{\sigma(d)} & ,d > 4m \end{cases} \tag{A.10}$$

A.1.3.3 IEEE 802.15.4 Path Loss Model Comparison

Both models are plotted in Fig A-1, and log-normal shadowing with a variance of σ = 8dB (chosen to represent 2.4GHz in an office environment, inferred from Table A-1). It can be seen through inspection that the indoor model (A.10) predicts a higher path loss than that specified in IEEE 802.15.4 (A.9).

By inspecting both models and considering results that have been empirically observed through practical experimentation, the second path loss model seems more reasonable for an indoor environment (giving a mean transmission distance of around 15m when used with a receiver sensitivity of -90dBm).

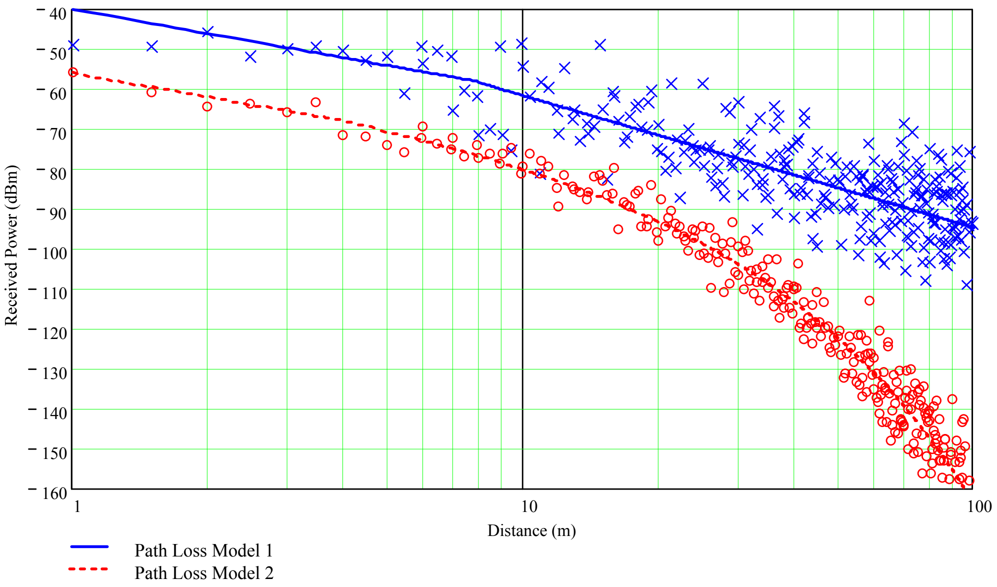


FIGURE A-1 : Comparison of the two IEEE 802.15.4 2.4GHz propagation models (model 1: specified in IEEE 802.15.4-2006 [84], and model 2: indoor model proposed in [177]) with P_T = 0dBm, $G_R = G_T$ = 0dBi.

A.2 Radio Frequency Communication

As a signal propagates through a communication channel, its power reduces due to attenuation factors that were discussed in the previous section. However, receiving these signals would be possible at infinite separation distances (through amplification of the signal) if it were not for the presence of noise and interference in the system [198]. This section provides a discussion of signal to noise ratios, interference, noise, and bit error rates.

A.2.1 Signal to Noise Ratios and Interference

The Signal to Noise Ratio (SNR) is a convenient measure of the ratio between the average signal power and noise power at a particular point in a communication link [198]. The signal to noise ratio is given by (A.11), where N [W] is the noise power, E_b [J] is the energy per bit, r [bits/s] is the communication bit rate, N_0 [W/Hz] is the noise-power spectral density (the noise power in one Hz of bandwidth), and B [Hz] is the communication bandwidth.

$$SNR = \frac{P_r}{N} = \frac{E_b r}{N_0 B} \tag{A.11}$$

The signal to interference plus noise ratio (SINR) is given by (A.12), where J [W] is the total interference power, and J_n [W] is the interference power from interferer n [92].

$$SINR = \frac{P_r}{N + J} = \frac{P_r}{N + \sum_n (J_n)} \tag{A.12}$$

In these definitions, *noise* is defined as Additive White Gaussian Noise (AWGN), and is discussed further in section A.2.2. The *signal* refers to the signal of interest being communicated between the transmitter and receiver. An *interferer* is an unwanted signal from an external source (that contain meaningful information, and hence are not random noise) that obstructs or masks the signal of interest [92]. However, in the IEEE 802.15.4 standard, interference is defined as “AWGN in the same bandwidth” [84], explaining why SINR and SNR are used interchangeably in this specification.

A.2.2 Communication Noise

Thermal noise is a result of the thermal motions of electrons present in conducting materials [92]. The thermal noise power spectral density is given in (A.13), where h [Js] is Planck’s constant, k [J/K] is Boltzmann’s constant, and T [K] is the temperature [198].

$$N_0 = \frac{hf}{\exp\left(\frac{hf}{kT}\right) - 1} \tag{A.13}$$

At radio frequencies, we can say that $kT >> hf$, and so (A.13) can be reduced to (A.14).

$$N_0 = kT \ , \qquad kT \gg hf \tag{A.14}$$

Using the relationship between the noise power and the noise power spectral density, the equation for the noise power can be derived (A.15).

$$N = N_0 B = kTB \tag{A.15}$$

Using (A.15), we can see that for an IEEE 802.15.4 network operating at 2.4GHz (having a bandwidth of 2MHz [84]), the thermal noise present is -111dBm at room temperature. However, to find the noise floor of the receiver, the internal noise in the receiver must also be accounted for. The sources and flow of noise in the receiver are shown in Fig. A-2, where SNR_{in}, S_{in} [W], and N_{in} [W] are the signal to noise ratio, signal power and noise power at the input of the receiver respectively, SNR_{out}, S_{out} [W], and N_{out} [W] are the signal to noise ratio, signal power and noise power at the output of the receiver respectively, G is the receiver system gain, and N_{int} [W] is the power of the internal noise in the receiver.

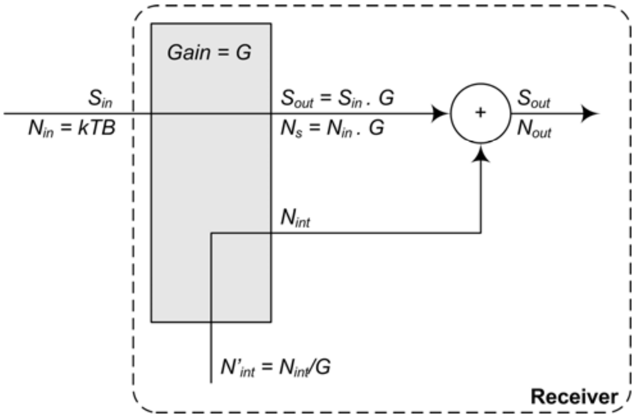


FIGURE A-2 : Sources of noise present in a receiver (adapted from Pearson *et al.* [206]).

The noise figure is a measure of the noise added to the signal by the receiver [206]; in this document, NF and nf are used to represent the noise figure in decibel and linear units respectively. The noise figure is a ratio between the noise power output of the receiver, and the noise power output of the receiver if it was noiseless (A.16).

$$nf = \frac{N_{out}}{N_{in}G} \tag{A.16}$$

Using (A.16) and Fig. A-2, an absolute value for the internal noise added by the receiver N_{int} can be derived (A.17).

$$nf = \frac{N_{in}G + N_{int}}{N_{in}G} = 1 + \frac{N_{int}}{N_{in}G} \tag{A.17}$$

Interestingly, (A.16) can also be rearranged to show the noise figure equal to the ratio of the input and output SNRs (A.18).

$$nf = \frac{N_{out}}{N_{in}G} = \frac{N_{out}S_{out}}{N_{in}S_{in}} = \frac{SNR_{in}}{SNR_{out}} \tag{A.18}$$

A.2.3 Bit Error Rates

The bit error rate (BER) gives the probability of a single bit being incorrectly transmitted across a communication channel. BERs are usually quoted or graphed against the SNR. Usually, as the SNR decreases (that is, the noise component becomes more influential), the BER increases.

$$BER = \frac{8}{15} \cdot \frac{1}{16} \sum_{k=2}^{16} (-1)^k \binom{16}{k} \mathrm{e}^{20 \cdot SINR \cdot \left(\frac{1}{\mathrm{k}} - 1\right)} \tag{A.19}$$

The IEEE 802.15.4 specification [84] gives the BER for 2.4GHz operation as shown in (A.19), where $\binom{16}{k}$ is the binomial coefficient (also notated as ${}_{16}C_k$).

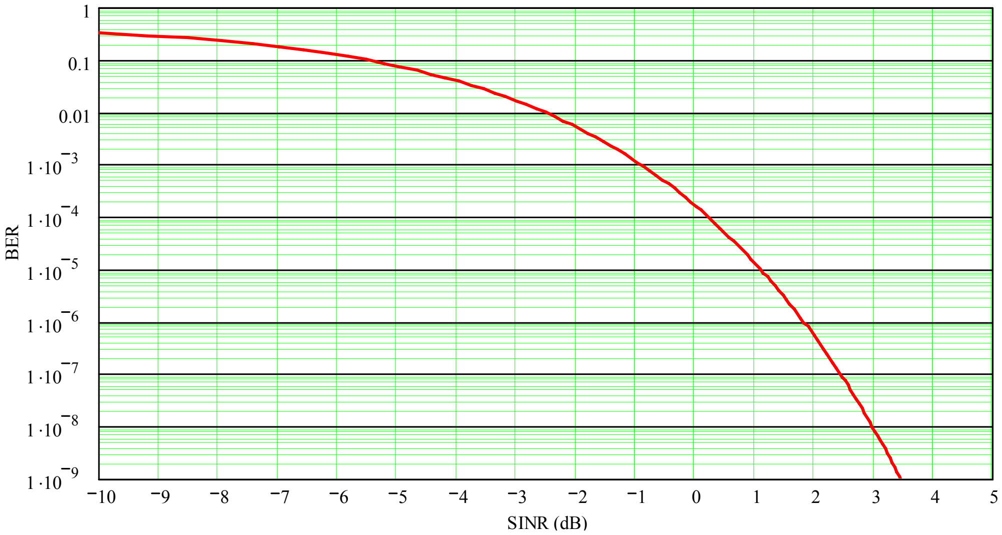


FIGURE A-3 : Bit Error Rate (BER) against Signal to Noise Interference Ratio (SINR) specified for IEEE 802.15.4 [84].

The BER defined in (A.19) is plotted in Fig. A-3. In Fig. A-3, it can be seen that IEEE 802.15.4 performs well in noisy environments, as even with equal noise and signal powers (SINR of 0dB) the resulting error rate is less than one bit error in 6000.

Appendix B

Battery and Supercapacitor Effects

B.1 Battery Effects

Batteries are classified into two categories; primary batteries (such as zinc-carbon and alkaline) are non-rechargeable, while secondary batteries (such as nickel cadmium [Ni-Cd], nickel-metal hydride [Ni-MH], and lithium-ion [Li-ion]) are rechargeable but with comparatively lower energy densities [207]. In this thesis, only secondary batteries are considered as primary batteries are of little use in energy-harvesting systems. Ni-MH cells have a higher energy density than Ni-Cd cells and are less toxic (containing no Cadmium), but are not as intolerant to repeated cycling. Li-ion cells have higher energy densities and are more efficient than Ni-MH cells. However, Li-ion cells require complex charging circuitry and contain potentially unsafe materials (the transport of lithium-based batteries is now subjected to heavy restrictions on airlines). A comparison of the properties of different secondary battery technologies is given in Table B-1.

TABLE B-1 : Properties of secondary battery technologies [207].

Battery Technology	**Ni-Cd**	**Ni-MH**	**Li-ion**
Energy Density (Wh/kg)	30-37	75	150
Operating Voltage	1.25-1.00	1.25-1.00	4.0-3.0
Self Discharge Rate (% loss/month at 20°C)	10	15-25	2
Charging Efficiency (Ah, %)	65-70	65-70	99
Cycle Life	500-2000	300-600	1000+

It is often assumed that energy stores such as batteries discharge with ‘ideal’ characteristics, providing the nominal voltage until all of the energy within the cell is depleted, at which time the voltage drops to zero (shown the ideal curve in Fig. B-1) [207]. In practice, the cell voltage drops as the energy in them becomes depleted (shown by Curve 1 in Fig. B-1). In order to calculate the usable energy stored in the battery, (B.1) is often used [208], where E_{max} [J] and V_{nom} [V] are the maximum capacity and nominal voltage of the battery respectively, C_n [Ah] is the rated capacity, which is rated for a discharge over n hours.

$$E_{max} = C_n \cdot V_{nom} \cdot 3600n \tag{B.1}$$

This relationship is however only an approximation, as it assumes that the battery is able to operate at the maximum voltage for the entire operation, and that the energy from it is usable by the load up until the rated cut-off voltage. In many applications, the load (such as the microprocessor) does not operate up to this voltage, resulting in either a reduction of the usable capacity, or requiring DC-DC step-up conversion [208].

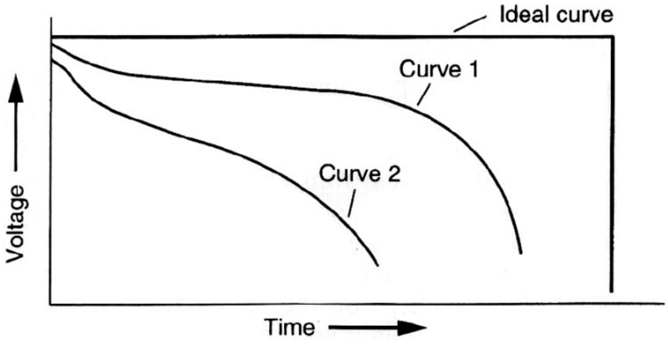


FIGURE B-1 : Characteristics of battery discharge, showing both the ideal and actual characteristics. Curve 2 is discharged at a higher current (hence a higher C-rate) than that of Curve 1 (reproduced from [207]).

While a battery may have a rated capacity C_n of 1500mAh, this does not mean that it could supply 1500mA for one hour. Instead, battery manufacturers usually specify a 'standard' discharge rate, specified as a 'C-rate'. The 'C-rate' is a charge/discharge rate relative to the rated capacity of the battery, and is generally expressed as M [h^{-1}] in (B.2), where I [A] is the discharge current, and M is the multiple or fraction of C_n.

$$I = MC_n \tag{B.2}$$

For example, GP batteries state that the rated capacity of their Ni-MH batteries is obtained through *"discharging at 0.2C to an end-voltage of 1.0V after fully charging at 0.1C"* [209]. The reason for quoting battery capacities like this is because, for batteries designed for lifetimes of many years at low discharge currents, the battery would deplete very quickly over a period of one hour due to the rate capacity effect. The rate capacity effect (also known as rate discharge) is the dependence of the battery's effective capacity being a function of the discharge current. If a battery is discharged at a current higher than its rated discharge current leads to a significant reduction in its lifetime, seen as a reduction in capacity [10, 167, 210]. This is because of the diffusion of active material (chemicals which give rise to electro-chemical reactions to generate electrical energy) through the battery. When discharging at low currents, active material is able to diffuse through the electrolyte at a sufficient rate. However, when the discharge current is high, the active ingredients are unable to diffuse through the electrolyte at the same rate as that of which they are being consumed at the cathode (Fig. B-2b). If the high discharge rate is sustained, the battery becomes exhausted even though active material remains in the cell (Fig. B-2d). If the high discharge rate is stopped or reduced, the active material is able to diffuse through the electrolyte allowing the battery to recover a proportion of it lost capacity. This effect is known as relaxation (seen in the progression from Fig. B-2b to B-2d).

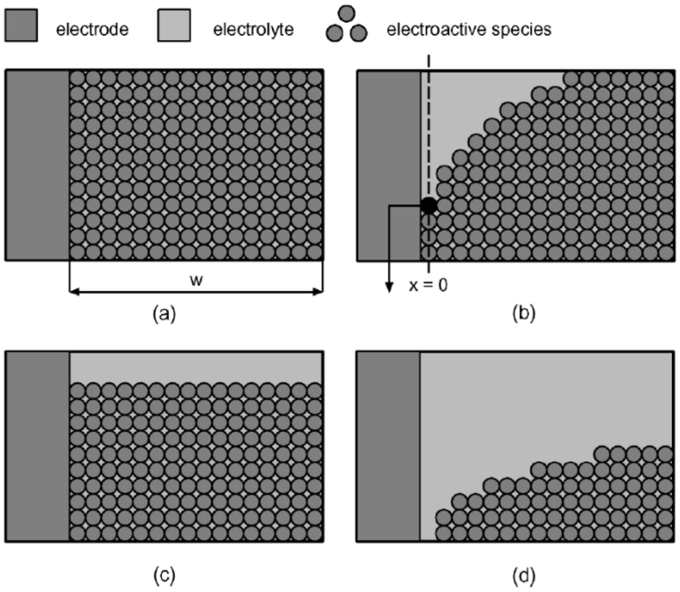


FIGURE B-2 : The rate-discharge and recovery effects seen in batteries: a) charged state, b) before recovery, c) after recovery, and d) discharged state (adapted from [170]).

All batteries suffer from self-discharge (or leakage) to a certain extent. Nickel based batteries tend to suffer from relatively high levels of self discharge, losing around 10% of its capacity in the 24 hours after charging, dropping to around 10% per month after this. Self-discharge is particularly sensitive to temperature (increasing with increased temperature), as shown for Ni-MH batteries in Fig. B-3.

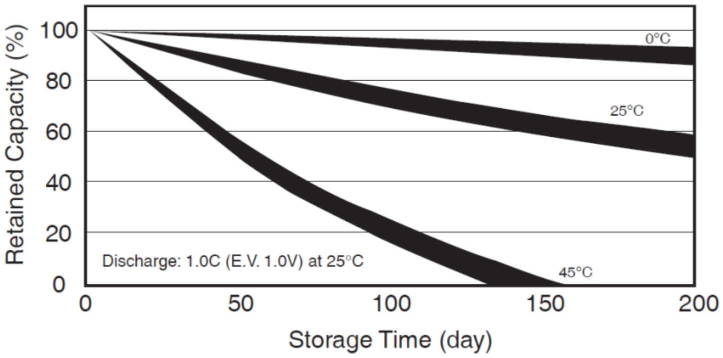


FIGURE B-3 : Temperature dependence of self-discharge in Ni-MH batteries (reproduced from [211]).

Battery cycle-life (the number of times a battery can be recharged before it is considered to have reached the end of its usable life) is an important factor, with Ni-MH batteries lasting for 500 cycles under 100% depth-of-discharge (DOD). However, the cycle-life increases as the DOD decreases, reducing it to 10% extends the cycle-life to around 5000 cycles [212]. The memory effect (experienced by Ni-Cd batteries) is the apparent drop in capacity if a battery is repeatedly used with a shallow discharge pattern [209]. The memory effect is not exhibited in Ni-MH and Li-ion cells. Batteries are not charged with perfect efficiency (that is, not all of the energy that is put into the cell is stored), as shown in Table B-1. For further information on batteries, their effects and technologies can be found in Buchmann [213] and Linden *et al.* [207].

B.2 Supercapacitor Effects

Unlike most batteries, capacitors are not affected by over charging or over discharging, and hence complicated charging and interface circuitry is not required. Additionally, capacitors are able to absorb and dissipate energy very quickly, and can sustain a virtually infinite number of recharge cycles [45]. The nature of most Wireless Sensor Network (WSN) applications dictate an intermittent energy consumption, and many energy harvesting sources used on sensor nodes inherently introduce a daily charging cycle. This makes capacitors attractive for use as energy stores on sensor nodes. However, the capacitance of most readily available conventional aluminium electrolytic capacitors is under 4700μF, equating to a stored energy of around 20mJ at 3V. Clearly this is not comparable to a battery, resulting in such capacitors not being suitable to supply the energy demands of sensor nodes.

Supercapacitors (also known as ultracapacitors or electrochemical dual-layer capacitors) [214, 215] differ from aluminium electrolytic capacitors by, in place of a dielectric, using a physical mechanism to generate a double electric field [184]. Supercapacitors share many of the benefits of aluminium electrolytic capacitors (such as those previously mentioned) [216] but have considerably higher energy densities, hence achieving capacities ranging from millifarads to hundreds of farads [45]. The energy stored in a capacitor (also applicable to supercapacitors [214]) is given by (B.3), where E_{res} [J] is the residual energy stored in the capacitor, C [F] is the rated capacitance, and V_s [V] is the instantaneous store voltage (the voltage across the capacitor). Using this, the capacity of a 100F supercapacitor with a maximum voltage of 2.5V (such as that used in [217]) is 313J, providing a feasible and attractive alternative to secondary batteries in implementing a sensor node's energy store.

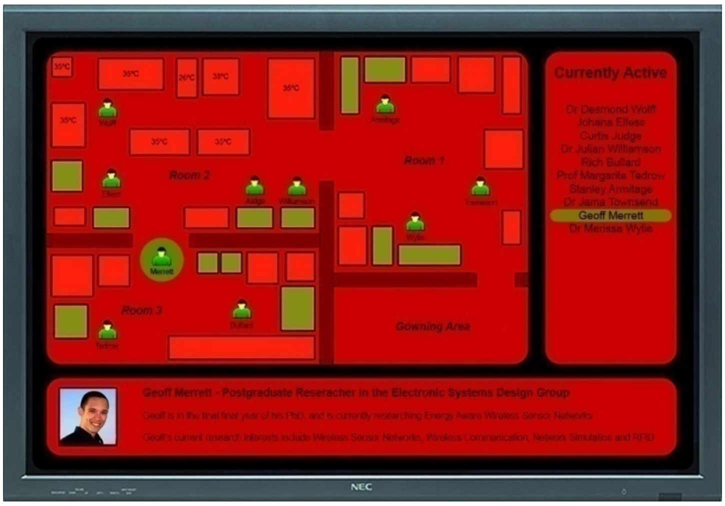
$$E_{res} = \frac{1}{2} C {V_s}^2 \tag{B.3}$$

Supercapacitors are often believed to be ideal, non-degrading energy stores, exhibiting only the effects specified above; however, while their characteristics are generally more favourable than most secondary batteries, supercapacitors exhibit a number of usually undesirable properties and effects. They are not able to operate indefinitely, and have a limited lifetime after which time they display a significant drop in capacitance. The actual lifetime is heavily dependent on temperature, humidity and operating voltage, but by operating at a reduced voltage and at room temperature, the lifetime can be considerably longer than ten years [35]. Supercapacitors are usually specified with wide tolerances on their rated capacitance (for example -20/40%) and a low maximum voltage rating (2.3V) [184]. Additionally, the supercapacitor's quadratic energy/voltage relationship shown in (B.3) causes the voltage to drop significantly as the stored energy depletes. This means that before the capacitor is fully depleted, it will be unable to provide the voltage required by external circuitry. One of the most significant problems with supercapacitors is the high leakage (or self-discharge) that they exhibit. This leakage is difficult to determine [184], and is generally only crudely specified in datasheets [45]. The leakage of a supercapacitor is dependent on the length of time that it was charged for [184]. The leakage is heavily reduced as the voltage across the supercapacitor drops, and so it is recommended that two supercapacitors are combined in parallel (halving the voltage of each) [45, 216]. This also increases the life of the supercapacitor, as the lifetime doubles for every 0.1V reduction in operating voltage [35].

Appendix C

The Mountbatten Cleanroom Wireless Sensor Network

The old Mountbatten cleanroom at the University of Southampton (destroyed by fire in 2005) did not interface well with its surroundings. Staff and students would leave the University with little or no idea of what was inside the cleanroom, and what work was being undertaken. Users of the cleanroom would often waste time going to the cleanroom and getting gowned, only to find that the machinery they required was in use. One of the architectural design aims of the new Mountbatten complex is to 'open up' the cleanroom, to show staff, students and visitors what is undertaken inside. In order to further evaluate the performance of IDEALS/RMR and the accuracy of the developed simulation models, a practical implementation has been proposed that, at the time of writing, is in the early stages of development.



Geoff Merrett - Postgraduate Reseracher in the Electronic Systems Design Group

Geoff is in the final final year of his PhD, and is currently researching Energy Aware Wireless Sensor Networks

Geoff's current research interests include Wireless Sensor Networks, Wireless Communication, Network Simulation and RFID

FIGURE C-1 : Prototype public display for the Mountbatten cleanroom WSN.

This implementation proposes a WSN for implementation into the new Mountbatten cleanroom; the primary aims of the cleanroom WSN are:

> **Publicity and Awareness:** The WSN will help to show what is happening inside the cleanroom, presenting information that could not be seen though existing methods. This will publicise what happens in the cleanroom to visitors, prospective personnel, and existing personnel. The information will be made readily available, via displays outside the cleanroom, and via the intranet/internet.
>
> **Tool for Staff and Students:** Users of the cleanroom will be able to see which machines are in use prior to visiting the cleanroom and getting changed. Therefore, making information on machine usage available over the intranet has the potential to increase productivity. Additionally, the WSN will help to remove further barriers by providing messaging service, whereby personnel outside the cleanroom can send short messages to colleagues inside the cleanroom via the intranet.
>
> **Assisting and Providing Infrastructure:** Automated long-term analysis of machine usage statistics will allow technicians to identify machines that are not often used, and turn them off (saving resources) or ultimately replace them.
>
> **WSN Research Platform:** The WSN will provide an effective research platform for energy-aware WSNs, allowing further development once the initial project is installed.

The majority of these aims can be seen graphically in the display screen (located outside the cleanroom) prototype shown in Fig. C-1. The currently active panel (top right of the screen) shows a list of the people that are currently using the cleanroom. The plan panel (top left of the screen) is a graphical bird's eye view of the cleanroom, showing who is working where, which machines are in use (depicted by the bench/machine colour), and parameter distribution across the cleanroom. The information panel (bottom of the screen) shows information about the currently selected person (highlighted in the currently active and plan panels), including their photograph, position, and research interests (drawn from publically available information). This information brings a personal element to cleanroom usage.

The network will be a mesh network with all nodes participating in multi-hop packet routing; the network architecture is shown in Fig. C-2. The network consists of a number of *Mobile Tracker Nodes (MTNs)* which are associated with users of the cleanroom (the association process takes places when the users pick up a node from a *Registration Device [RD]* in the gowning areas). The *MTNs* trilaterate their own position using packets or signals emitted from static nodes located on machinery and benches. The static nodes also implement machine usage statistics (using a switch on the nodes) and distributed parameter sampling (such as temperature). Network operation is coordinated from an out-of-cleanroom *Network Controller (NC)*, which provides user-*MTN* association and data management (displays, intranet and internet interfaces are all controlled by the *NC*).

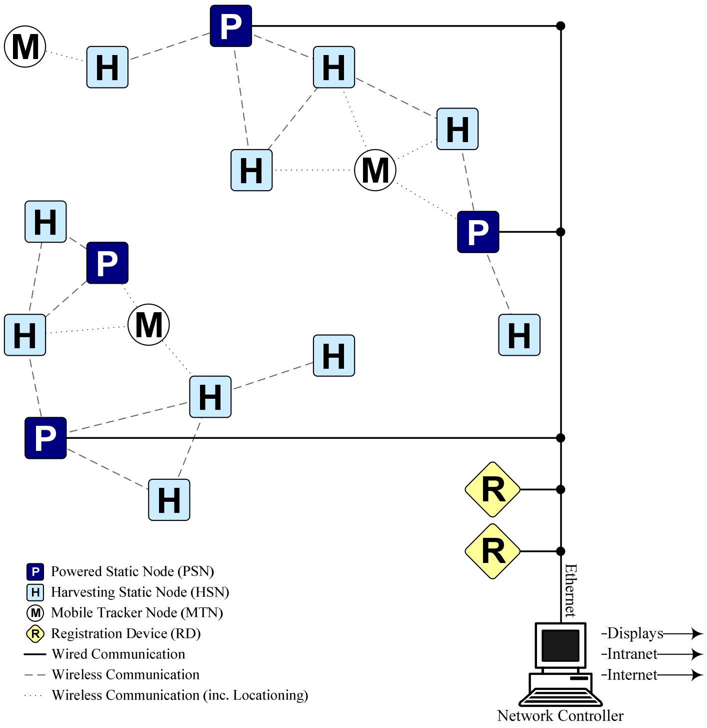

P
H
M
R
P Powered Static Node (PSN)
H Harvesting Static Node (HSN)
M Mobile Tracker Node (MTN)
R Registration Device (RD)
Wired Communication
Wireless Communication
Wireless Communication (inc. Locationing)
Ethernet
-Displays
-Intranet
-Internet
Network Controller

FIGURE C-2 : The architecture of the Mountbatten cleanroom WSN.

IDEALS/RMR will be used to make nodes' network involvement dependent on their local resources. A primary challenge of the cleanroom project is the ability of the network to be energy-efficient, while providing an up-to-date and accurate impression of its monitored environment; IDEALS/RMR is critical in accomplishing this. The ability to infer packet importance depends on the packet type, and possible metrics are:

Parameter Distribution: Parameters such as temperature or ambient light can be controlled by threshold and differential rules such as those presented above.

Machine Usage: These packets are unable to have information–management applied to them, as there is no distinction between important and unimportant packets. Therefore, they will always be transmitted as important packets.

Locationing Trilateration Packets: These packets are transmitted by static nodes in order for *MTNs* to self-determine their location. By allocating these packets a medium importance, the number of participating static nodes (and hence the accuracy of the trilateration process) is traded for energy.

Location Dissemination: These packets report the location of the *MTNs* to the *NC*. The importance of can be proportional to the distance that the node moved since the last report (i.e. a small location change is not as important as a room change).

Appendix D

Selected Publications

G. V. Merrett, N. R. Harris, B. M. Al-Hashimi, and N. M. White, “Energy Managed Reporting for Wireless Sensor Networks,” in *Sensors and Actuators A: Physical*, vol. 142, pp. 379-389, 2008.

G. V. Merrett, A. S. Weddell, A. P. Lewis, N. R. Harris, B. M. Al-Hashimi, N. M. White, “An Empirical Energy Model for Supercapacitor Powered Wireless Sensor Nodes,” presented at *IEEE Int’l Conf. Computer Communications and Networks*, St. Thomas, US Virgin Islands, Aug. 2008.

G. V. Merrett, A. S. Weddell, N. R Harris, B. M. Al-Hashimi, N. M. White, “A Structured Hardware/Software Architecture for Embedded Sensor Nodes,” presented at *Workshop Advanced Networking and Communications*, St. Thomas, US Virgin Islands, Aug. 2008.

Bibliography

[1] M. Weiser, "The computer for the 21st century," *Scientific American (International Edition)*, vol. 265, pp. 66-75, Sep. 1991.

[2] M. Satyanarayanan, "Pervasive computing: vision and challenges," *IEEE Personal Communications*, vol. 8, pp. 10-17, Aug. 2001.

[3] DTI, "Electronics 2015 - Making A Visible Difference," Department of Trade & Industry, London, Business Sector Report, 04/1812, 2004

[4] J.M. Rabaey, M.J. Ammer, J.L. da Silva, Jr. et al., "PicoRadio supports ad hoc ultra-low power wireless networking," *Computer*, vol. 33, pp. 42-48, Jul. 2000.

[5] N.V. Dierdonck, "Wireless Standards Demystified," GreenPeak Technologies, 20080114 Standards overview.doc, Feb. 2008

[6] Towson, "Ubiquitous--Wireless Sensor Networks: An Introduction," presented at *WSN Training: Intro to Wireless Sensor Networks*, Nov. 2005

[7] D. Culler, "Wireless Sensor Networks – the next IT Revolution," presented at *35th Korea Electronics Show*, Korea, Oct. 2004

[8] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *Communications Magazine*, vol. 40, pp. 102-114, Aug. 2002.

[9] V. Raghunathan, S. Ganeriwal, and M. Srivastava, "Emerging techniques for long lived wireless sensor networks," *Communications Magazine*, vol. 44, pp. 108-114, Apr. 2006.

[10] V. Raghunathan, C. Schurgers, S. Park, and M.B. Srivastava, "Energy-aware wireless microsensor networks," *Signal Processing Magazine*, vol. 19, pp. 40-50, Mar. 2002.

[11] D. Culler, D. Estrin, and M. Srivastava, "Guest editors' introduction: overview of sensor networks," *Computer*, vol. 37, pp. 41-49, Aug. 2004.

[12] T. Arampatzis, J. Lygeros, and S. Manesis, "A survey of applications of wireless sensors and wireless sensor networks," in *Proc. Mediterranean Conf. Control and Automation*, Limassol, Cyprus, pp. 719-724, Jun. 2005

[13] I. Smith, A. Furness, C. Bullock et al., "Radio Frequency Identification (RFID) - A Mission To The USA," DTI, London, May 2006

[14] M. Ward, R.v. Kranenburg, and G. Backhouse, "RFID: Frequency, standards, adoption and innovation," JISC, May 2006

[15] A.D. DeHennis and K.D. Wise, "A wireless microsystem for the remote sensing of pressure, temperature, and relative humidity," *Microelectromechanical Systems*, vol. 14, pp. 12-22, Feb. 2005.

[16] J.R. Smith, A.P. Sample, P.S. Powledge et al., "A wirelessly-powered platform for sensing and computation," in *Proc. 8th Int'l Conf. Ubiquitous Computing (UbiComp 2006)*, Orange County, CA, vol. 4206, pp. 495-506, Sep. 2006

[17] S. Methley and C. Forster, "Predicting Take up and Spectrum in Wireless Sensor Networks," presented at *5th Wireless Sensing Interest Group (WiSIG) Meeting*, Teddington, UK, March 2008

[18] A. Tully, "Pervasive tagging, sensors and data collection: A science and technology review for the foresight project on intelligent infrastructure systems," *IEE Proceedings: Intelligent Transport Systems*, vol. 153, pp. 129-146, Jun. 2006.

[19] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, pp. 393-422, Mar. 2002.

[20] Intel(R) Mote, http://www.intel.com/research/exploratory/motes.htm; last accessed 18th April 2008.

[21] OFCOM, "Technology Trends Workshop Report," London, Dec. 2007

[22] B. Deb, S. Bhatnagar, and B. Nath, "Information Assurance in Sensor Networks," in *Proc. 2nd ACM Int'l Workshop Wireless Sensor networks and Applications (WSNA'03)*, San Diego, CA, United States, pp. 160-168, Sep 2003

[23] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: scalable coordination in sensor networks," in *Proc. 5th Ann. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MOBICOM'99)*, Seattle, WA, pp. 263-270, Aug. 1999

[24] R. Min, M. Bhardwaj, S.-H. Cho et al., "Low-power wireless sensor networks," in *Proc. 14th Int'l Conf. VLSI Design*, pp. 205-210, Jan. 2001

[25] A. Chandrakasan, R. Min, M. Bhardwaj et al., "Power Aware Wireless Microsensor Systems," in *Proc. Solid-State Device Research Conference, 2002. Proceeding of the 32nd European*, pp. 37-44, Sep. 2002

[26] P.K. Dutta and D.E. Culler, "System software techniques for low-power operation in wireless sensor networks," in *Proc. Int'l Conf. Computer Aided Design (ICCAD'05)*, San Jose, CA, USA, pp. 925-32, Nov. 2005

[27] TI, "2.4 GHz IEEE 802.15.4 ZigBee low cost, low power System-on-Chip (SoC) solution (rev. 2.1)", 2006.

[28] A. Sinha and A. Chandrakasan, "Dynamic power management in wireless sensor networks," *IEEE Design & Test of Computers*, vol. 18, pp. 62-74, Apr. 2001.

[29] P.H. Chou and C. Park, "Energy-efficient platform designs for real-world wireless sensing applications," in *Proc. Int'l Conf. Computer Aided Design (ICCAD'05)*, San Jose, CA, USA, pp. 913-20, Nov. 2005

[30] P. Zhang, C.M. Sadler, S.A. Lyon, and M. Martonosi, "Hardware design experiences in ZebraNet," in *Proc. Int'l Conf. Embedded Networked Sensor Systems*, Baltimore, MD, pp. 227-238, Nov. 2004

[31] S.P. Beeby, M.J. Tudor, and N.M. White, "Energy harvesting vibration sources for microsystems applications," *Measurement Science and Technology*, vol. 17, Oct. 2006.

[32] S. Roundy, D. Steingart, L. Frechette et al., "Power Sources for Wireless Sensor Networks," in *Proc. 1st European Workshop Wireless Sensor Networks (EWSN'04)*, vol. 2920/2004, pp. 1-17, Jan. 2004

[33] T. Starner and J.A. Paradiso, "Human Generated Power for Mobile Electronics," in *Low Power Electronics Design*, vol. Chapter 35, C. Piguet, Ed.: CRC Press, pp. 1-35, 2004.

[34] H. Ren, M.Q.-H. Meng, and X. Chen, "Physiological information acquisition through wireless biomedical sensor networks," in *Proc. IEEE Int'l Conf. Information Acquisition*, Jun. 2005

[35] A.S. Weddell, N.R. Harris, and N.M. White, "Alternative Energy Sources for Sensor Nodes: Rationalized Design for Long-Term Deployment," in *Proc. Int'l Instrumentation and Measurement Technology Conf.*, Victoria, Canada, pp. 1370-1375, May 2008

[36] A. Kansal, J. Hsu, S. Zahedi, and M.B. Srivastava, "Power Management in Energy Harvesting Sensor Networks," *ACM Transactions on Embedded Computing Systems (in revision)*, pp. 35, May 2006.

[37] S. Roundy, P.K. Wright, and J. Rabaey, "A study of low level vibrations as a power source for wireless sensor nodes," *Computer Communications*, vol. 26, pp. 1131-1144, Jul. 2003.

[38] N.G. Elvin, N. Lajnef, and A.A. Elvin, "Feasibility of structural monitoring with vibration powered sensors," *Smart Materials and Structures*, vol. 15, pp. 977-986, Aug. 2006.

[39] S.E. George, M. Bocko, and G.W. Nickerson, "Evaluation of a vibration-powered, wireless temperature sensor for health monitoring," in *Proc. 2005 IEEE Aerospace Conference*, Big Sky, MT, USA, pp. 3775-81, Mar. 2005

[40] Panasonic, "Panasonic Solar Cells Technical Handbook '98/99," Techincal Handbook, Aug. 1998

[41] C. Park and P.H. Chou, "AmbiMax: Autonomous Energy Harvesting Platform for Multi-Supply Wireless Sensor Nodes," in *Proc. Sensor and Ad Hoc Communications and Networks (SECON'06)*, vol. 1, pp. 168-177, Sep. 2006

[42] P. Dutta, J. Hui, J. Jeong et al., "Trio: enabling sustainable and scalable outdoor wireless sensor network deployments," in *Proc. Int'l Conf. Information Processing in Sensor Networks*, Nashville, TN, pp. 407-15, Apr. 2006

[43] V. Raghunathan, A. Kansal, J. Hsu et al., "Design considerations for solar energy harvesting wireless embedded systems," in *Proc. Int'l Symp. Information Processing in Sensor Networks (IPSN'05)*, Los Angeles, CA, vol. 2005, pp. 457-462, Apr. 2005

[44] Understanding Solar Powered Wireless Sensor Nodes at Systems Level, http://www.eecs.berkeley.edu/Research/Projects/Data/102077.html; last accessed March 2007.

[45] X. Jiang, J. Polastre, and D. Culler, "Perpetual Environmentally Powered Sensor Networks," in *Proc. 4th Int'l Conf. Information Processing in Sensor Networks (IPSN'05)*, Los Angeles, CA, Apr. 2005

[46] K.-Y. Lin, T.K.K. Tsang, M. Sawan, and M.N. El-Gamal, "Radio-triggered solar and RF power scavenging and management for ultra low power wireless medical applications," in *Proc. IEEE Int'l Symposium on Circuits and Systems*, Island of Kos, Greece, pp. 4, May 2006

[47] K. Delin and S. Jackson., "The Sensor Web: A New Instrument Concept," in *Proc. SPIE Symp. Integrated Optics*, San Jose, CA, Jan. 2001

[48] C.M. Cianci, V. Trifa, and A. Martinoli, "Threshold-based algorithms for power-aware load balancing in sensor networks," in *Proc. Swarm Intelligence Symposium*, Pasadena, CA, pp. 349-56, Jun. 2005

[49] A. Kansal and M.B. Srivastava, "An Environmental Energy Harvesting Framework for Sensor Networks," in *Proc. Int'l Symp. Low Power Electronics and Design (ISLPED'03)*, pp. 481-486, Aug. 2003

[50] A. Cerpa and D. Estrin, "ASCENT: Adaptive Self-Configuring sEnsor Networks Topologies," in *Proc. 21st Conf. IEEE Computer and Communications Societies (INFOCOM'02)*, vol. 3, pp. 1278-1287, Jun. 2002

[51] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," in *Proc. 7th Int'l Conf. Mobile Computing and Networking*, Rome, Italy, pp. 85-96, Jul. 2001

[52] L. Wang and Y. Xiao, "Energy saving mechanisms in sensor networks," in *Proc. Int'l Conf. Broadband Networks*, pp. 777-785, Oct. 2005

[53] W. Boyes, "Data into information," *Control*, vol. 17, pp. 77-81, Sep. 2004.

[54] D. Chen and P.K. Varshney, "QoS support in wireless sensor networks: A survey," in *Proc. Int'l Conf. Wireless Networks*, Las Vegas, NV, vol. 1, pp. 227-233

[55] A. Durresi, "Architectures for Heterogeneous Wireless Sensor Networks Invited Paper," in *Proc. 16th IEEE Int'l Symp Personal, Indoor and Mobile Radio Communications (PIMRC'05)*, vol. 2, pp. 1289-1296, Sep. 2005

[56] P. Bonnet, J. Gehrke, and P. Seshadri, "Querying the physical world," *IEEE Personal Communications*, vol. 7, pp. 10-15, Oct. 2000.

[57] S. Goel and T. Imielinski, "Prediction-based monitoring in sensor networks: Taking lessons from MPEG," *Wireless Extensions to the Internet Computer Communication Review*, vol. 31, pp. 82-98, Oct. 2001.

[58] X. Liu, Q. Huang, and Y. Zhang, "Combs, needles, haystacks: balancing push and pull for discovery in large-scale sensor networks," in *Proc. Int'l conf. Embedded networked sensor systems*, Baltimore, MD, USA, pp. 122-133, 2004

[59] S. Dhanani, J. Arseneau, A. Weatherton et al., "A comparison of utility-based information management policies in sensor networks," in *Proc. Systems and Information Engineering Design Symposium*, Charlottesville, VA, pp. 84-9, Apr. 2006

[60] S.R. Madden, M.J. Franklin, J.M. Hellerstein, and W. Hong, "TinyDB: an acquisitional query processing system for sensor networks," *ACM Transactions on Database Systems*, vol. 30, pp. 122-73, Mar. 2005.

[61] Y. Yao and J. Gehrke, "The Cougar approach to in-network query processing in sensor networks," *SIGMOD Record*, vol. 31, pp. 9-18, Sep. 2002.

[62] A. Manjeshwar and D.P. Agrawal, "TEEN: a routing protocol for enhanced efficiency in wireless sensor networks," in *Proc. 15th Int'l Parallel and Distributed Processing Symp. (IPDPS'01)*, pp. 2009-2015, Apr. 2001

[63] A. Manjeshwar and D.P. Agrawal, "APTEEN: a hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks," in *Proc. 16th Int'l Parallel and Distributed Processing Symp. (IPDPS'02)*, pp. 195-202, Apr. 2002

[64] N. Hu, D. Zhang, and F. Wang, "Dynamic threshold scheme used in directed diffusion," in *Proc. Int'l Conf. Embedded Software and Systems*, Xi'an, China, pp. 220-9, Dec. 2005

[65] G. Werner-Allen, K. Lorincz, J. Johnson et al., "Fidelity and yield in a volcano monitoring sensor network," in *Proc. 7th USENIX Symp. Operating Systems Design and Implementation (OSDI'06)*, Seattle, WA, USA, pp. 381-96, Nov. 2006

[66] A. Jain, E.Y. Chang, and Y.-F. Wang, "Adaptive stream resource management using Kalman Filters," in *Proc. Int'l Conf. on Management of Data, (SIGMOD'04)*, Paris, France, pp. 11-22, Jun. 2004

[67] C. Liu, K. Wu, and M. Tsao, "Energy efficient information collection with the ARIMA model in wireless sensor networks," in *Proc. IEEE Global Telecommunications Conf.*, St. Louis, MO, pp. 5, Dec. 2005

[68] G. Wang, H. Wang, J. Cao, and M.G.A.-M. Guo, "Energy-Efficient Dual Prediction-Based Data Gathering for Environmental Monitoring Applications," in *Proc. Conf. Wireless Communications and Networking (WCNC'07)*, pp. 3513-3518, Mar 2007

[69] M. Britton, "Emerging Applications for Intelligence in Wireless Sensor Networks in Industrial Environments," presented at *5th Wireless Sensing Interest Group (WiSIG) Meeting*, Teddington, UK, March 2008

[70] M. Wenz and H. Worn, "Event-based Production Rules for Data Aggregation in Wireless Sensor Networks," in *Proc. IEEE Int'l Conf. Multisensor Fusion and Integration for Intelligent Systems*, pp. 59-64, Sep. 2006

[71] P. Padhy, R. Dash, K. Martinez, and N. Jennings, "A Utility based sensing and communication model for a glacial sensor network," in *Proc. Int'l Conf. Autonomous Agents and Multiagent Systems (AAMAS'06)*, Hakodate, Japan

[72] R. Willett, A. Martin, and R. Nowak, "Backcasting: adaptive sampling for sensor networks," in *Proc. Int'l Symp. Information Processing in Sensor Networks*, Berkeley, CA, USA, pp. 124-33, Apr. 2004

[73] S.K. Chong, S.W. Loke, and S. Krishnaswamy, "Wireless sensor networks: from data to context to energy saving," in *Proc. Int'l Workshop Ubiquitous Data Management (UDM-05)*, pp. 33-40, Apr. 2005

[74] S. Bhatnagar, B. Deb, and B. Nath, "Service Differentiation in Sensor Networks," in *Proc. 4th Int'l Symp. Wireless Personal Multimedia Communications*, Sept. 2001

[75] B. Deb, S. Bhatnagar, and B. Nath, "ReInForM: reliable information forwarding using multiple paths in sensor networks," in *Proc. 28th Conf. Local Computer Networks*, Bonn, Germany, pp. 406-415, Oct. 2003

[76] A. Kopke, H. Karl, and M. Lobbers, "Using energy where it counts: protecting important messages in the link layer," in *Proc. 2nd European Workshop on Wireless Sensor Networks*, pp. 226-235, Feb. 2005

[77] B. Hull, K. Jamieson, and H. Balakrishnan, "Bandwidth management in wireless sensor networks," in *Proc. Int'l Conf. Embedded Networked Sensor Systems*, Los Angeles, CA, pp. 306-307, Nov. 2003

[78] A. Rogers, R.K. Dash, N.R. Jennings et al., "Computational Mechanism Design for Information Fusion within Sensor Networks," in *Proc. Int'l Conf. Information Fusion*, Florence, Italy, July 2006

[79] A.S. Tanenbaum, *Computer Networks*, 4 ed. Upper Saddle River, New Jersey: Pearson Education, 2003.

[80] "Information technology - Open Systems Interconnection - Basic Reference Model: The Basic Model", 1994.

[81] D. Meyer and G. Zobrist, "TCP/IP versus OSI," *IEEE Potentials*, vol. 9, pp. 16-19, Feb. 1990.

[82] S. Kolla, D. Border, and E. Mayer, "Fieldbus networks for control system implementations," in *Proc. Electrical Insulation Conf. & Electrical Manufacturing & Coil Winding Technology Conf.*, pp. 493-498, Sep. 2003

[83] ZigBee Alliance, "ZigBee Specification Document 053474r17", 2007.

[84] IEEE, "IEEE Standard for Information Technology - Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)", 2006.

[85] D. Graumann, W. Lara, J. Hightower, and G. Borriello, "Real-world implementation of the location stack: the universal location framework," in *Proc. Workshop on Mobile Computing Systems and Applications*, Monterey, CA, pp. 122-8, Oct. 2003

[86] D. Karatzas, A. Chorti, N.M.White, and C.J.Harris, "Teaching Old Sensors New Tricks: Archetypes of Intelligence," *IEEE Sensors: Special Issue on Intelligent Sensing*, 2006.

[87] V. Srivastava and M. Motani, "Cross-layer design: a survey and the road ahead," *Communications Magazine, IEEE*, vol. 43, pp. 112-119, Dec 2005.

[88] D. Culler, "TinyOS: Operating System Design for Wireless Sensor Networks," in *Sensors*, 2006.

[89] M. Healy, T. Newe, and E. Lewis, "Power Management in Operating Systems for Wireless Sensor Nodes," in *Proc. IEEE Sensors Applications Symposium (SAS'07)*, pp. 1-6, Feb. 2007

[90] M. Honkanen, A. Lappetelainen, and K. Kivekas, "Low end extension for Bluetooth," in *Proc. IEEE Radio and Wireless Conference*, pp. 199-202, Sep. 2004

[91] C. Koumpis and s. Aliwell`, "Energy Harvesting and Power Management Technologies enabling Wireless Sensing: An Action Group Study," presented at *5th Meeting of the Wireless Sensing Interest Group (WiSIG)*, Teddington, UK, Mar 2008

[92] H. Karl and A. Willig, *Protocols and Architectures for Wireless Sensor Networks*. Chichester: John Wiley & Sons Ltd, 2005.

[93] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proc. IEEE Conf. Information Communications (INFOCOM'02)*, New York, NY, USA, vol. 3, pp. 1567-76, Jun. 2002

[94] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC protocols for wireless sensor networks: a survey," *IEEE Communications Magazine*, vol. 44, pp. 115-121, Apr. 2006.

[95] M.J. Miller and N.H. Vaidya, "Minimizing energy consumption in sensor networks using a wakeup radio," in *Proc. IEEE Wireless Communications & Networking Conf. (WCNC'04)*, vol. 4, pp. 2335-2340, Mar. 2004

[96] G.P. Halkes, T. Van Dam, and K.G. Langendoen, "Comparing energy-saving MAC protocols for wireless sensor networks," *Mobile Networks and Applications*, vol. 10, pp. 783-91, Oct. 2005.

[97] M. Ali, U. Saif, A. Dunkels et al., "Medium access control issues in sensor networks," *Computer Communication Review*, vol. 36, pp. 33-6, Apr. 2006.

[98] A. Woo and D.E. Culler, "A transmission control scheme for media access in sensor networks," in *Proc. Int'l Conf. Mobile Computing and Networking*, Rome, pp. 221-235, Jul. 2001

[99] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proc. Int'l Conf. Embedded Networked Sensor Systems (SenSys'04)*, Baltimore, MD, United States, pp. 95-107, Nov. 2004

[100] M. Buettner, G.V. Yee, E. Anderson, and R. Han, "X-MAC: A short preamble MAC protocol for duty-cycled wireless sensor networks," in *Proc. Int'l Conf. Embedded Networked Sensor Systems (SenSys'06)*, Boulder, CO, United States, pp. 307-320, Oct 2006

[101] F. Bennett, D. Clarke, J.B. Evans et al., "Piconet: embedded mobile networking," *IEEE Personal Communications*, vol. 4, pp. 8-15, Oct. 1997.

[102] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM Trans. Networking*, vol. 12, pp. 493-506.

[103] T. Van Dam and K. Langendoen, "An adaptive energy-efficient MAC protocol for wireless sensor networks," in *Proc. Int'l Conf. Embedded Networked Sensor Systems (SenSys'03)*, Los Angeles, CA, United States, pp. 171-180

[104] L.C. Zhong, R. Shah, C. Guo, and J. Rabaey, "An Ultra-Low Power and Distributed Access Protocol for Broadband Wireless Sensor Networks," in *Proc. IEEE Broadband Wireless Summit*, Las Vegas, Nevada, May 2001

[105] C. Guo, L.C. Zhong, and J.M. Rabaey, "Low power distributed MAC for ad hoc sensor radio networks," in *Proc. Global Telecommunications Conference (GLOBECOM'01)*, vol. 5, pp. 2944-2948, Nov. 2001

[106] S.A. Bhalerao, A.V. Chaudhary, R.B. Deshmukh, and R.M. Patrikar, "Powering wireless sensor nodes using ambient RF energy," in *Proc. IEEE Int'l Conf. Systems, Man and Cybernetics*, Taipei, Taiwan, vol. 4, pp. 2695-2700, Oct. 2006

[107] P. Kolinko and L.E. Larson, "Passive RF Receiver Design for Wireless Sensor Networks," in *Proc. Int'l Microwave Symposium*, pp. 567-570, Jun 2007

[108] L. Gu and J.A. Stankovic, "Radio-triggered wake-up for wireless sensor networks," *Real-Time Systems*, vol. 29, pp. 157-182, Mar. 2005.

[109] "ITU-T Recommendation Q.9: General Recommendation on Telephone Switching and Signalling, Vocabulary of Switching and Signalling Terms", 1988.

[110] J.N. Al-Karaki and A.E. Kamal, "Routing techniques in wireless sensor networks: a survey," *Wireless Communications*, vol. 11, pp. 6-28, Dec. 2004.

[111] Q. Jiang and D. Manivannan, "Routing protocols for sensor networks," in *Proc. 1st Consumer Communications and Networking Conf. (CCNC)*, pp. 93-98, Jan. 2004

[112] N. Vlajic and D. Xia, "Wireless sensor networks: to cluster or not to cluster?," in *Proc. Int'l Symp. A World of Wireless Mobile and Multimedia Networks*, Buffalo-Niagara Falls, NY, pp. 9, Jun 2006

[113] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad Hoc Networks*, vol. 3, pp. 325-49, May 2005.

[114] R. Min and A. Chandrakasan, "Top five myths about the energy consumption of wireless communication," in *Proc. ACM SIGMOBILE Mobile Computing and Communications Review*, pp. 65-67, Jan. 2003

[115] M. Haenggi, "Twelve reasons not to route over many short hops," in *Proc. 60th IEEE Vehicular Technology Conf. (VTC'04)*, Los Angeles, CA, vol. 5, pp. 3130-3134, Sep. 2004

[116] Z. Shelby, C. Pomalaza-Raez, H. Karvonen, and J. Haapola, "Energy optimization in multihop wireless embedded and sensor networks," *Int'l J. Wireless Information Networks*, vol. 12, pp. 11-21, Jan. 2005.

[117] Q. Zhao and L. Tong, "Energy efficiency of large-scale wireless networks: proactive versus reactive networking," *IEEE J. Selected Areas in Communications*, vol. 23, pp. 1100-1112, May 2005.

[118] Y. Zhang and L. Cheng, "Flossiping: a new routing protocol for wireless sensor networks," in *Proc. IEEE Int'l Conf. Networking, Sensing and Control*, vol. 2, pp. 1218-1223, 2004

[119] D. Ganesan, B. Krishnamachari, A. Woo et al., "An Empirical Study of Epidemic Algorithms in Large Scale Multihop Wireless Networks," Intel Research, IRB-TR-02-003, Mar. 2002

[120] P. Downey and R. Cardell-Oliver, "Evaluating the impact of limited resource on the performance of flooding in wireless sensor networks," in *Proc. 2004 Int'l Conf. Dependable Systems and Networks*, Florence, Italy, pp. 785-794, Jun. 2004

[121] J. Kulik, W. Heinzelman, and H. Balakrishnan, "Negotiation-based protocols for disseminating information in wireless sensor networks," *Wireless Networks*, vol. 8, pp. 169-185, 2002.

[122] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proc. 5th Ann. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MOBICOM'99)*, Seattle, WA, vol. 8, pp. 153-167, May 2002

[123] Z.J. Haas, J.Y. Halpern, and L. Li, "Gossip-based ad hoc routing," in *Proc. Conf. IEEE Computer and Communications Societies (INFOCOM'02)*, vol. 3, pp. 1707-1716, Jun. 2002

[124] Y. Zhang and M. Fromherz, "Constrained flooding: a robust and efficient routing framework for wireless sensor networks," in *Proc. Int'l Conf. Advanced Information Networking and Applications*, Vienna, Austria, pp. 6, Apr. 2006

[125] P. Gburzynski, B. Kaminska, and W. Olesinski, "A tiny and efficient wireless ad-hoc protocol for low-cost sensor networks," in *Proc. Design, Automation and Test in Europe Conference and Exhibition*, Nice, France, Apr. 2007

[126] Y.-B. Ko, J.-M. Choi, and J.-H. Kim, *A New Directional Flooding Protocol for Wireless Sensor Networks*, 3090 ed, 2004.

[127] S. Wu and K.S. Candan, "GPER: geographic power efficient routing in sensor networks," in *Proc. 12th IEEE Int'l Conf. Network Protocols*, Berlin, Germany, pp. 161-172, Oct. 2004

[128] G.G. Finn, "Routing and addressing problems in large metropolitan-scale internetworks," USC/ISI, ISI/RR-87-180, Mar. 1987

[129] Y. Yu, R. Govindan, and D. Estrin, "Geographical and Energy Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks," UCLA, May 2001

[130] B. Karp and H.T. Kung, "GPSR: Greedy Perimeter Stateless Routing for wireless networks," in *Proc. 6th Ann. Int'l Conf. Mobile Computing and Networking (MOBICOM'00)*, Boston, MA, pp. 243-254, Aug. 2000

[131] H. Cho and Y. Baek, "Location-based routing protocol for energy efficiency in wireless sensor networks," in *Proc. Embedded and Ubiquitous Computing*, Nagasaki, Japan, pp. 622-31, Dec. 2005

[132] J. Li, L. Gewali, H. Selvaraj, and V. Muthukumar, "Hybrid greedy/face routing for ad-hoc sensor network," in *Proc. Euromicro Symp. Digital System Design (DSD'04)*, pp. 574-578, Aug. 2004

[133] V.C. Barbosa and R.C. Dutra, "Finding routes in anonymous sensor networks," *Information Processing Letters*, vol. 98, pp. 139-144, May 2006.

[134] C. Intanagonwiwat, R. Govindan, D. Estrin et al., "Directed diffusion for wireless sensor networking," *Trans. Networking*, vol. 11, pp. 2-16, Feb. 2003.

[135] T. Kwon, M. Chen, and Y. Choi, "Energy-efficient differentiated directed diffusion (EDDD) in wireless sensor networks," *Computer Communications*, vol. 29, pp. 231-45, Jan. 2006.

[136] V. Handziski, A. Köpke, H. Karl et al., "Improving the Energy Efficiency of Directed Diffusion Using Passive Clustering," in *Wireless Sensor Networks*, vol. 2920 / 2004, *Lecture Notes in Computer Science*, H. Karl, A. Wolisz, and A. Willig, Eds., pp. 172-187, 2004.

[137] Y. Liu and W.K.G. Seah, "A scalable priority-based multi-path routing protocol for wireless sensor networks," *Wireless Information Networks*, vol. 12, pp. 23-33, Jan. 2005.

[138] R.C. Shah and J.M. Rabaey, "Energy aware routing for low energy ad hoc sensor networks," in *Proc. IEEE Wireless Communications and Networking Conf. (WCNC'02)*, Orlando, FL, vol. 1, pp. 350-355, Mar. 2002

[139] G. Park, S. Yi, J. Heo et al., "Energy aware routing with dynamic probability scaling," in *Proc. Int'l Conf. Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, Regina, Canada, pp. 662-70, Aug. 2005

[140] T. Voigt, H. Ritter, and J. Schiller, "Utilizing solar power in wireless sensor networks," in *Proc. 28th IEEE Int'l Conf. Local Computer Networks (LCN'03)*, pp. 416-422, Oct. 2003

[141] T. Voigt, A. Dunkels, J. Alonso et al., "Solar-aware clustering in wireless sensor networks," in *Proc. 9th Int'l Symp. Computers and Communications (ISCC'04)*, Alexandria, Egypt, vol. 1, pp. 238-243, Jun. 2004

[142] V. Prasad and S.H. Son, "Classification of Analysis Techniques for Wireless Sensor Networks," in *Proc. Int'l Conf. Networked Sensing Systems (INSS'07)*, pp. 93-97, Jun 2007

[143] G. Chen, J., M.J. Branch, L.Z. Pflug, and B. Szymanski, "SENSE: A Sensor Network Simulator," in *Advances in Pervasive Computing & Networking*, B. Szymanksi and B. Yener, Eds., pp. 249-267, 2004.

[144] S. Park, A. Savvides, and M.B. Srivastava, "SensorSim: A simulation framework for sensor networks," in *Proc. Int'l Workshop Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Boston, MA, pp. 104-111, Aug. 2000

[145] The Network Simulator - ns2, www.isi.edu/nsnam/ns; last accessed Mar. 2008.

[146] A. Sobeih, J.C. Hou, L.-C. Kung et al., "J-Sim: a simulation and emulation environment for wireless sensor networks," *IEEE Wireless Communications*, vol. 13, pp. 104-19, Aug. 2006.

[147] A. Varga, "The OMNET++ discrete event simulation system," in *Proc. European Simulation Multiconference*, Prague, Czech Republic, pp. 319-25, Jun. 2001

[148] C. Mallanda, A. Suri, V. Kunchakarra et al., "Simulating Wireless Sensor Networks with OMNeT++," *submitted to IEEE Computer*, 2005.

[149] Castalia - A Simulator for WSNs; last accessed Dec. 2007.

[150] E.M. Ould-Ahmed-Vall, G.F. Riley, and B.S. Heck, "Large-Scale sensor networks simulation with GTSNetS," *Simulation*, vol. 83, pp. 273-290, 2007.

[151] A. Sobeih and J.C. Hou, "A Simulation Framework for Sensor Networks in JSim," University of Illinois at Urbana-Champaign, UIUCDCS-R-2003-2386, Nov. 2003

[152] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and scalable simulation of entire TinyOS applications," in *Proc. Int'l Conf. Embedded Networked Sensor Systems (SenSys'03)*, Los Angeles, CA, pp. 126-137, Nov. 2003

[153] M. Varshney, D. Xu, M. Srivastava, and R. Bagrodia, "SenQ: a scalable simulation and emulation environment for sensor networks," in *Proc. 6th Int'l Symp. Information Processing in Sensor Networks*, Cambridge, MA, USA, pp. 196-205, Apr. 2007

[154] J. Polley, D. Blazakis, J. McGee et al., "ATEMU: a fine-grained sensor network simulator," in *Proc. Conf. Sensor & Ad Hoc Communications & Networks (SECON'04)*, pp. 145-152, Oct. 2004

[155] B.L. Titzer, D.K. Lee, and J. Palsberg, "Avrora: scalable sensor network simulation with precise timing," in *Proc. Int'l Symp. Information Processing in Sensor Networks (IPSN'05)*, pp. 477-482, Apr. 2005

[156] J. Barbancho, F.J. Molina, C. Leon et al., "OLIMPO, An ad-hoc wireless sensor network simulator for public utilities applications," in *Proc. 2nd European Workshop on Wireless Sensor Networks (EWSN'05)*, Istanbul, Turkey, vol. 2005, pp. 419-424, Feb. 2005

[157] S. Mount, R. Newman, E. Gaura, and J. Kemp, "SENSOR: an algorithmic simulator for wireless sensor networks," in *Proc. Eurosensors XX*, Gothenburg, Sweden, vol. 2, pp. 410-411, Sep. 2006

[158] J.N. Al-Karaki and G.A. Al-Mashagbeh, "Sensoria: a fully-fledged simulator for wireless sensor networks," in *Proc. Int'l Symp. Personal, Indoor and Mobile Radio Communications*, Helsinki, Finland, pp. 6, Sep. 2006

[159] D. Cavin, Y. Sasson, and A. Schiper, "On the accuracy of MANET simulators," in *Proc. Int'l Workshop Principles of Mobile Computing*, Toulouse, France, pp. 38-43, Oct 2004

[160] T.R. Andel and A. Yasinsac, "On the credibility of manet simulations," *Computer*, vol. 39, pp. 48-54, Jul 2006.

[161] M. Varshney and R. Bagrodia, "Detailed models for sensor network simulations and their impact on network performance," in *Proc. Int'l Symp. Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Venice, Italy, pp. 70-77, 2004

[162] J. Heidemann, N. Bulusu, J. Elson et al., "Effects of Detail in Wireless Network Simulation," in *Proc. Conf. Communication Networks and Distributed Systems Modeling and Simulation*, Phoenix, AZ, pp. 1-10, Jan 2001

[163] C. Newport, D. Kotz, Y. Yuan et al., "Experimental evaluation of wireless simulation assumptions," *Simulation*, vol. 83, pp. 643-661, Sep 2007.

[164] S. Sundresh, W. Kim, and G. Agha, "SENS: A sensor, environment and network simulator," in *Proc. Annual Simulation Symposium*, Arlington, VA, pp. 221-228, Apr. 2004

[165] Castalia - A Simulator for WSNs, http://castalia.npc.nicta.com.au/; last accessed Mar. 2008.

[166] S. Park, A. Savvides, and M.B. Srivastava, "Battery capacity measurement and analysis using lithium coin cell battery," in *Proc. Int'l Symp. Low Electronics and Design (ISLPED'01)*, Huntington Beach, CA, pp. 382-387, Aug. 2001

[167] K. Lahiri, A. Raghunathan, S. Dey, and D. Panigrahi, "Battery-driven system design: a new frontier in low power design," in *Proc. Design Automation Conf. 2002*, pp. 261-267, 2002

[168] M. Handy and D. Timmermann, "Simulation of Mobile Wireless Networks with Accurate Modeling of Non-linear Battery Effects," in *Proc. Int'l. Conf. Applied Simulation and Modeling*, Marbella, Spain, pp. 532-537, Sep. 2003

[169] L. Gao, S. Liu, and R.A. Dougal, "Dynamic lithium-ion battery model for system simulation," *IEEE Trans. Components and Packaging Technologies*, vol. 25, pp. 495-505, Sep. 2002.

[170] D. Rakhmatov and S. Vrudhula, "Energy management for battery-powered embedded systems," *Trans. Embedded Computing Sys.*, vol. 2, pp. 277-324, Aug 2003.

[171] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc. 33rd Int'l Conf. System Sciences*, Hawaii, Jan. 2000

[172] Q. Wang, M. Hempstead, and W. Yang, "A Realistic Power Consumption Model for Wireless Sensor Network Devices," in *Proc. Conf. Sensor and Ad Hoc Communications and Networks*, vol. 1, pp. 286-295, Sep. 2006

[173] V. Shnayder, M. Hempstead, B.-r. Chen et al., "Simulating the power consumption of large-scale sensor network applications," in *Int'l Conf. Embedded networked sensor systems*. Baltimore, MD, USA: ACM, pp. 188-200, 2004.

[174] O. Landsiedel, K. Wehrle, and S. Gotz, "Accurate Prediction of Power Consumption in Sensor Networks," in *Proc. 2nd IEEE Workshop Embedded Networked Sensors (EmNetS-II)*, pp. 37-44, May 2005

[175] Symmetricom, "Stochastic model estimation of network time variance," Symmetricom, White Paper, WP/SMENTV/D/0403/PDF, Apr 2003

[176] J. Kho, A. Rogers, and N. Jennings, "Decentralised Control of Adaptive Sampling in Wireless Sensor Networks," *ACM Transactions on Sensor Networks*, vol. 5, 2009 (In Press).

[177] F. Martin, P. Gorday, J. Adams, and H.v. Leeuwen, "IEEE 802.15.4 PHY Capabilities," presented at *IEEE P802.15 Working Group for WPANs*, May 2004

[178] "IEEE Standard for Information technology - Part 15.1: Wireless medium access control (MAC) and physical layer (PHY) specifications for wireless personal area networks (WPANs)", 2005.

[179] Analog Devices, "±0.5°C Accurate, 10-Bit Digital Temperature Sensors", 2005.

[180] PerkinElmer Optoelectronics, "VTB8341: VTB Process Photodiodes", 2000.

[181] National Semiconductor, "LPC660 Low Power CMOS Quad Operational Amplifier", 2004.

[182] Analog Devices, "ADG884 0.5 Ohm CMOS, Dual 2:1 MUX/SPDT Audio Switch", 2006.

[183] D. Freeman, "Battery management tackles alternative battery technologies in advanced portable systems," in *Proc. WESCON/94*, pp. 303-308, Sept. 1994

[184] Panasonic, "Gold Capacitors Technical Guide", 2005.

[185] Schott Solar GmbH, Putzbrunn, Germany, "ASI-OEM Solarmodules for Indoor", 2006.

[186] Perpetuum Ltd, http://www.perpetuum.com/; last accessed May. 2008.

[187] J. Elson and K. Romer, "Wireless sensor networks: a new regime for time synchronization," in *Proc. First HotNets Workshop, Computer Communication Review*, Princeton, NJ, USA, vol. 33, pp. 149-54, Jan. 2003

[188] Maxim, "DS1302 Trickle-Charge Timekeeping Chip", 2005.

[189] A.C. Atkinson and M.C. Pearce, "The Computer Generation of Beta, Gamma and Normal Random Variables," *J. Royal Statistical Society, Series A (General)*, vol. 139, pp. 431-461, 1976.

[190] A. Boukerche, H.A.B.F. Oliveira, E.F. Nakamura, and A.A.F. Loureiro, "Localization systems for wireless sensor networks," *IEEE Wireless Communications*, vol. 14, pp. 6-12, Dec 2007.

[191] A.S. Weddell, G.V. Merrett, N.R. Harris, and B.M. Al-Hashimi, "Energy Harvesting and Management for Wireless Autonomous Sensors," *Measurement + Control*, vol. 41, pp. 104-108, May 2008.

[192] P.J. Boltryk, C.J. Harris, and N.M. White, "Intelligent sensors-a generic software approach," *Journal of Physics: Conference Series*, vol. 15, pp. 155-60, 2005.

[193] L. Nachman, R. Kling, R. Adler et al., "The Intel mote platform: a Bluetooth-based sensor network for industrial monitoring," in *Proc. Int'l Symp. Information Processing in Sensor Networks*, Los Angeles, CA, pp. 437-42, Apr. 2005

[194] A. Ortiz, T. Olivares, and L. Orozco-Barbosa, "A heterogeneous role-based sensor network," in *Proc. Workshop Performance monitoring and measurement of heterogeneous wireless and wired networks*, Chania, Crete Island, Greece, pp. 61-67, Oct. 2007

[195] J.C. Castillo, T. Olivares, and L. Orozco-Barbosa, "Implementation of a rule-based routing protocol for wireless sensor networks," in *Proc. Workshop Performance monitoring and measurement of heterogeneous wireless and wired networks*, Chania, Crete Island, Greece, pp. 19-25, Oct. 2007

[196] T.S. Rappaport, *Wireless Communications: Principles & Practice*: Prentice Hall, 1996.

[197] J. Kumagai, "Life of birds," *Spectrum*, vol. 41, pp. 42-49, Apr. 2004.

[198] I. Otung, *Communication Engineering Principles*: Palgrave, 2001.

[199] H.T. Friis, "A Note on a Simple Transmission Formula," *Proc. IRE*, pp. 254-256, May 1946.

[200] D.C. Hogg, "Fun with the Friis free-space transmission formula," *Antennas and Propagation Magazine*, vol. 35, pp. 33-35, Aug. 1993.

[201] T.S. Rappaport, R. Muhamed, and V. Kapoor, "Propagation Models," in *The Communications Handbook*, J. D. Gibson, Ed.: CRC Press, pp. 1182-1196, 1997.

[202] K.Y. Yazdandoost and R. Kohno, "Antenna, Wave Propagations, and Field Regions for Body Area Network (IEEE 802.15-06-0397-00-0ban)," presented at *IEEE P802.15 Working Group for WPANs*, Sep. 2006

[203] J.B. Andersen, T.S. Rappaport, and S. Yoshida, "Propagation measurements and models for wireless communications channels," *Communications Magazine*, vol. 33, pp. 42-49, Jan. 1995.

[204] S.Y. Seidel and T.S. Rappaport, "914 MHz path loss prediction models for indoor wireless communications in multifloored buildings," *Trans. Antennas and Propagation*, vol. 40, pp. 207-217, Feb. 1992.

[205] "IEEE Recommended Practice for Information Technology - Part 15.2: Coexistence of Wireless Personal Area Networks with Other Wireless Devices Operating in Unlicensed Frequency Bands", 2003.

[206] J. Pearson, *Basic Communication Theory*. Trowbridge: Prentice Hall, 1992.

[207] D. Linden and T.B. Reddy, "Handbook of Batteries," 3 ed: McGraw-Hill, 2002.

[208] P. Corke, P. Valencia, P. Sikka et al., "Long-duration solar-powered wireless sensor networks," in *Proc. Workshop Embedded Networked Sensors*, Cork, Ireland, pp. 33-37, June 2007

[209] GP Batteries, "Nickel Cadmium Technical Hand Book", 2002.

[210] C. Park, K. Lahiri, and A. Raghunathan, "Battery discharge characteristics of wireless sensor nodes: an experimental analysis," in *Proc. Sensor and Ad Hoc Communications and Networks, 2005. IEEE SECON 2005. 2005 Second Annual IEEE Communications Society Conference on*, pp. 430-440, Sep. 2005

[211] GP Batteries, "Nickel Metal Hydride Technical Handbook", 2006.

[212] A. Kansal, J. Hsu, M. Srivastava, and V. Raghunathan, "Harvesting Aware Power Management for Sensor Networks," in *Proc. Design Automation Conference, 2006 43rd ACM/IEEE*, pp. 651-656, Jul. 2006

[213] I. Buchmann, *Batteries in a Portable World*, 2 ed: Cadex Electronics Inc., 2001.

[214] A. Burke, "Ultracapacitors: why, how, and where is the technology," *J. Power Sources*, vol. 91, pp. 37-50, Nov 2000.

[215] L. Zubieta and R. Bonert, "Characterization of double-layer capacitors for power electronics applications," *IEEE Trans. Industry Applications*, vol. 36, pp. 199-205, Feb. 2000.

[216] S. Mahlknecht and M. Roetzer, "Energy supply considerations for self-sustaining wireless sensor networks," in *Proc. European Workshop on Wireless Sensor Networks*, pp. 397-399, Feb 2005

[217] F. Simjee and P.H. Chou, "Everlast: Long-life, Supercapacitor-operated Wireless Sensor Node," in *Proc. Int'l Symposium Low Power Electronics and Design (ISLPED'06)*, pp. 197-202, Oct. 2006