Using Local and Global Knowledge in Wireless Sensor Networks

Christopher Gwilliams
Cardiff University

January 3, 2014

Contents

List of Figures

List of Tables

Abstract

To Write...

Chapter 1

Introduction

A wireless sensor network (WSN) consists of a collection of heterogeneous nodes with sensing and, typically, wireless capabilities. These sensing nodes can be extremely complex and powerful devices with the ability to sense multiple phenomena simultaneously, or they can be simple motes that have limited processing power and are tasked with sensing one thing in their environment.

Upon deployment, these nodes use their wireless capabilities to form links with their neighbours, where a neighbour is any node that is within transmission range. The way that nodes discover, and communicate with, their neighbours is defined by their routing protocol. Routing protocols vary based on the purpose of the WSN, the requirements of data transmission as well as the characteristics of the nodes. Communication between nodes is expensive and drains the available power faster than any other action that a node performs. For example, if a WSN is deployed in a building with consistent power available, then the routing protocol does not need to be adapted to ensure the nodes maximise their battery life by transmitting as little as possible. However, not every WSN has unlimited resources at their disposal and these protocols, as well as the underlying structure of the network, are used to ensure the network is able to perform well for as long as possible.

Each WSN is different and each will have different constraints, a WSN that monitors traffic along a busy road may experience memory limitations,

whereas a WSN that is deployed in the middle of a desert may experience power issues. Typically, however, all WSNs do the same thing: sense one or more characteristics of their environment and forward that data on to a specified endpoint.

1.1 The Local Knowledge Problem

The majority of WSNs do not know what data they are sensing, or have any knowledge of their environment. This means that, unless fixed by a routing protocol or human that deployed the network, data is delivered on a chronological basis and is then filtered at the base station, usually manually. Some WSNs store all of the data on the node and users of the network must use a 'pull' model to query for data from nodes, but this requires some technical knowledge and, while it does increase the battery life of the nodes, it is a manual process again.

The environment that a WSN is deployed is usually rich and the data sensed often contains patterns that can be used to improve the performance of the network. For example, if a node knows that it is has only been triggering between the hours of 6pm and 5am for the past few weeks, it can enter a deep sleep outside of those hours or use that time to transmit data it has been storing while it knows it will be inactive. Alternatively, this knowledge can be used to prioritise data throughout the network so that the most important data is received first, instead of the most recent. An example of this could two camera nodes deployed facing the entry and exit of a building, tasked with looking for intruders between 5pm and 8am. If the camera facing the exit is triggered at 5:01pm and the camera on the entrance is triggered at 5:05pm, then the knowledge that the security guard leaves through the exit between 5:01pm and 5:08 pm will allow the entrance camera to prioritise its capture as more important, as it is an irregular occurrence.

This knowledge can be categorised into *local* and *global*. Local knowledge (LK) is the knowledge of an area that has been gained through experience or experimentation and global knowledge (GK) is knowledge that is generally available to everybody. An example of this is someone who has been tasked

with deploying a WSN in the Amazon rainforest would use readily available sources, such as the Internet or prior research, to determine the humidity and weather patterns in order to use a node that could withstand such conditions. This would be classed as GK. However, a native to the Amazon may know that three of the locations that the nodes are to be deployed in are flooded for two weeks of the year, rendering their readings useless for that time period and increasing their risk of failure. This is LK, as it cannot be gained without experiencing the flooding in that area, or experimenting with water levels.

We believe that the use of this knowledge can increase the efficiency of the network, as well as prioritise sensed data by its value instead of the time it was recorded. To show this, we have developed a network architecture for WSNs that utilises knowledge from the data it senses, as well as its deployed environment. It is called the Knowledge-based Hierarchical Architecture for Sensing (K-HAS) and this thesis will show how K-HAS addresses the problem of delivering the most important data first and improving the overall efficiency of the network.

1.2 Motivation

Throughout this thesis, we focus on a scenario motivated by our collaboration with Cardiff University School of Biosciences, who run a research centre in the Malaysian rainforest, in Sabah, known as Danau Girang (DG) . Located on the banks of the Kinabatangan river, DG has been running for more than six years and holds Masters, PhD and Undergraduate students from around the world, studying the ecology and biodiversity of the unique region.

The reason that the rainforest that DG is set in is so unique is that the area was heavily logged until the late 1970s and now serves as a corridor, between large Palm Oil plantations, connecting two separate rainforest lots. The area is now secondary rainforest (rainforest that has grown since being destroyed) and is experiencing a large variety of wildlife using the area as a habitat, or as a path. Some of this wildlife is unique to this area of the world and DG has had sightings of animals that have not been seen in many years.

There is a variety of research projects currently underway in the field

centre, looking into fish population, crocodile attacks, hornbill habitats or the movement patterns of small mammals. One project that has been running almost since DG opened, is the *corridor monitoring programme*, a programme that consists of dozens of wildlife cameras deployed in various areas around DG and triggered whenever an animal triggers a break in their infrared (IR) sensor.

The Kinabatangan is a very humid place, with thick forest, making it very difficult to walk through and even more difficult for hardware to survive the conditions. Cameras are placed along the river and up to 1km into the forest, recording triggers onto SD cards. These SD cards are collected and stored at the field centre, where the images are manually collated and processed. The cameras are designed to have a battery life of three months but, due to the humidity, a battery life of 3 weeks is more realistic. In 2010, twenty cameras were deployed and half of them were inspected every two weeks, on a rotating basis. In that time, each cameras can record more than a thousand pictures and the dynamic nature of the rainforest, such as the sun through leaves, falling trees and reflections in the water can cause the camera to trigger when an animal has not walked past; we call these *false triggers*. False triggers can make up to 70

We have used this scenario to test our hypothesis and implement a WSN that automates the collection, transmission, processing and storage of images, using LK to classify the data and prioritise the flow of information through the network, making more efficient use of the limited power and bandwidth available.

1.3 Research Contributions

1.4 Thesis Structure

The rest of this thesis is structured as follows. Chapter 2 provides some background on wireless sensor networks and the use of knowledge. Chapter 3 explains some technical decisions we made and the findings when running

experiments in the Malaysian rainforest. Chapter 4 introduces the K-HAS architecture we have developed and explains the purpose of each tier. Chapter 5 details the ontology we have developed to support K-HAS and shows how current ontologies do not sufficiently cover all of the concepts involved with a *scientific observation*. Chapter 6 shows how we use rules to determine how valuable a piece of sensed data is and to explain how K-HAS is able to use these rules, and human feedback, in order to inform future classifications. Chapter 7 highlights the technical limitations of implementing K-HAS today and shows our simulations of it running as it was designed. Chapter 8 then concludes this thesis and summarises our contributions and findings, as well as highlighting work that could be undertaken to take this project further.

Chapter 2

Background

Using knowledge in a WSN is related to existing research into sensor networks that utilise context-awareness in order to improve their efficiency or adapt their sampling rate.

This chapter is split into the following sections. Section 2.1 outlines the issues surrounding WSN design and deployment. Section 2.2 details relevant existing routing protocols for sensor networks. Section 2.3 highlights popular sensor middleware in use today. Section 2.4 shows some examples of existing WSNs that are related to our motivating scenario. Section 2.5 introduces research into local and global knowledge and Section 2.6 shows some related work into WSNs that utilise knowledge or context to prioritise data and/or improve efficiency.

2.1 Wireless Sensor Network Issues

WSNs have been used in a number of domains, for a range of different purposes, from habitat monitoring [**?**] to military purposes [**?**] and healthcare [**?**]. While these applications are vastly different, the technology behind each is very similar. Each requires the use of nodes with sensors attached and each node requires a power source and storage devices.

According to [**?**], there are eight factors that affect the design of sensor networks, but we focus on a subset that are the most relevant to our research

problem. The following points must be considered when designing a WSN:

2.1.1 Fault Tolerance

WSNs typically contain a large number of nodes and each can fail for various reasons, from a lack of power, filling its storage capacity, to factors of the environment causing the hardware to fail. While the sensor nodes that are used in WSNs typically consist of the same platform, the variation between each deployment means that the device itself must be adapted to its environment, [**?**] used a custom protective casing for their nodes so that they were able to survive being in the open while ensuring that the transmission range was not affected.

2.1.2 Hardware Constraints

A sensor node typically consists of: a platform that contains the memory and processing power, a sensor (or sensors) and a transceiver that uses a wireless standard, such as Wi-Fi or Zigbee. Cost and size are the most common barriers to entry when designing a WSN. [**?**] mentions that the expectation of a sensor node is a matchbox-sized form factor. While the research is over ten years old, the original focus on sensor node was *smart dust* [**?**], small, inexpensive, disposable nodes that can transmit until their power reserve is depleted, [**?**] mentions that it is a requirement for the nodes to cost less than USD10. In [**?**], it is noted that, a decade on from the first WSN papers, smart dust has not been realised and the focus has instead been on larger, more powerful nodes that have reduced in cost and grown in power.

The Gartner Hype Cycle for 2013 [**?**] shows that smart dust is still in early innovation stages and may not be fully commercialised for another ten years. To counter this, research has been focussed on using software solutions to maximise the battery life in these more powerful, more expensive nodes, accompanies with the use of renewable energy sources.

2.1.3 Energy Constraints

The majority of sensor nodes do not have access to a constant power supply and must run on a battery that is, generally, a similar size to the node itself; or smaller. This means that the nodes must be as efficient as possible, knowing when to transmit data and when to sleep. The lifetime of a sensor network is extremely dependent on the battery life of each node and, unlike other mobile devices, they cannot typically be recharged [**?**]. Much work has been done on power efficient routing protocols, as well as the control of which attached devices are active [**?**, **?**, **?**].

The limited resources on the nodes mean that the sensing devices, and transceivers, attached must consume as little power as possible. Some routing protocols implement turning off wireless radios and scheduling a wakeup across the network [**?**], but the cost of turning off a device can waste just as much energy as leaving it on and sampling at a lower rate; if not more [**?**].

The use of energy in a node is dependent on how active the sensor(s) are, how much it transmits and receives, the transmission medium used as well as the environment it is in.

2.1.4 Transmission Medium

Common transmission media, such as Wi-Fi, are viable solutions in WSNs when a high data rate is required and power is readily available. However, research has shown that Wi-Fi is extremely power-hungry and [**?**] shows that Wi-Fi consumes almost 9 times more energy, while transmitting, than other standards, such as Zigbee. Bluetooth is a more power efficient standard that is becoming increasingly popular for sensing devices that are [art of the *Quantified Self* movement [**?**], with wearable devices that report measurements, such as heart rate, steps taken and calories burned. With the advent of the new low-power Bluetooh 4.0 , also known as Bluetooth Low Energy (BLE), this standard is supposed to allow months of continuous use on a coin-cell battery [**?**]. However, the range is limited to 100m and, using the same frequency, as Wi-Fi (2.4GHz) means that it is as susceptible to path loss and reduced transfer rates. [**?**] shows that a 2.4GHz Wi-Fi antenna is

capable of transmitting up to 350m, while a considerably lower frequency of 41MHz was able to achieve links of 10km. The use of 2.4GHz frequencies in wet conditions have been shown to reduce the performance by up to 28New low-power, low-frequency standards have emerged in recent years and allow for a considerably longer range and increased battery life, at the cost of transmission speeds. Digmish is an example of this and, while it can achieve 250kb/s using 2.4GHz, it has much slower speeds of 125kbps when using the 900MHz spectrum. However, it does offer a range of, up to, 64Km [**?**].

2.1.5 Environment

The environment that a node is deployed in can have a great impact on almost all aspects of a WSN, such as range and battery life. Harsh environments that are not easily accessible make it difficult to place nodes and protected environments may limit where nodes can be placed. In [**?**], nodes were deployed within glaciers and had to survive extreme temperatures, lasting without human intervention, for months at a time. [**?**] attached collars to Zebras that had to withstand high speed movement, dust and high temperatures. The deployment of any node requires extensive research as to the environment that it will be deployed in and adjustments must be made to ensure it is able to survive for extended periods without continued maintenance. Section 2.1.4 also shows that environment does not simply affect the hardware, but humidity can reduce the transmission range significantly, as well as moisture collecting on wireless antenna can reduce the range for days at a time.

2.2 Routing Protocols

Routing protocols specify how nodes in a WSN are organised, as well as how they transmit data throughout the network. In [**?**], the more popular routing protocols are surveyed and split into three of the main identified categories: data-centric, hierarchical and location-based. We use the aforementioned categories, as well as flat, to highlight some of the key protocols that are

relevant to our work. The protocols have the task of ensuring that a network is performing at its best, providing the best lifetime and ensuring reliable and consistent delivery of data. This must reduce *flooding*, where nodes send every message to every link, aside from themselves, effectively flooding the network with unnecessary messages, and find a way to deliver data to the endpoint using the most efficient path possible.

2.2.1 Flat

Initially, this was the most common structure for a WSN, dozens of nodes spread out over a geographical area, with one or more neighbours, sending observations to a single endpoint.

MCFA

The Minimum Cost Forwarding Algorithm (MCFA) is an flat routing protocol that works by assigning costs to each node, based on how many hops they are from the base station [**?**].

Each node has a path-estimate of the cost of transmission from itself to the base station. The base station sends out a broadcast message and it is received by all nodes in range. The message contains a cost from the base station (initially zero) while every node has their cost set to infinity. If the cost in the message, plus the link it was received, is less than the current cost. If yes, the estimate is update on both the node and the message; the message is then passed on to other nodes in range.

This approach allows for dynamic reconfiguration of the network, as well as a reduced overhead due to not having to maintain a global routing table on each node. The assumption with MCFA, however, is that the direction of routing is always towards a fixed endpoint.

2.2.2 Data-centric

Data-centric routing protocols are not like traditional WSNs where nodes are given addresses; they use a method that involves the advertising, or querying,

of the data that has been sensed and those with the relevant data can respond to the request.

SPIN

Sensor Protocols for Information via Negotation is one of the first data-centric protocols and attempts to address the issue of flooding the network whenever new data is sensed by addressing the data through metadata [**?**]. SPIN works on three messages passed between nodes:

1. ADV - A message sent by a node when it has sensed new data, advertising what it has recorded.
2. REQ - Sent by nodes that received an ADV to request the data.
3. DATA - Message containing the sensed data.

When a node has sensed data, it sends an ADV message to all nodes within range. If any of those nodes are interested in the data, then they respond with a REQ message, at which point the DATA message is sent to nodes that responded.

SPIN eliminates the need for a global view of the network topology, as nodes only need to know their single hop neighbours. However, SPIN does not guarantee equal diffusion of data throughout the network as a node may be interested in the data sensed at the other edge of the network, with only nodes that are not interested in between. This would mean that those nodes would not request the data or pass it on.

SPIN-IT

An extension to SPIN, SPIN-IT uses a slightly different approach to receiving data and is developed solely for the transfer of images [**?**].

Nodes use the existing message structure of SPIN, but REQ messages are used as queries, sent to all nodes in transmission range. The receiving nodes keep these requests and generate a new REQ message, thus allowing nodes to store temporal paths. When a REQ reaches a node that has the

desired data, it responds with a ROUTE-REPLY message. This message is used because images are large and resource-constrained WSNs would have a much shorter lifetime if a lot of unnecessary transmissions were made. The ROUTE-REPLY is used in case multiple nodes, in range of the requesting node, have the data it has requested and it can then choose the optimal route. As each node keeps a history of REQ messages, these can be used to trace the requested data back through the network, to the originating node, without the overhead of maintaining a global routing table.

COUGAR

A slightly different data-centric approach is the proposed COUGAR protocol, viewing the network as a distributed database. While similar to SPIN due to the fact that it does not forward data as soon as it is sensed, COUGAR uses a query language that abstracts the underlying network structure and uses that query to generate a plan that utilises in-network processing to provide an answer [**?**].

Within the network, a *leader* is selected and this node is used to aggregate the data from nodes that were able to fulfil all, or some, of the query. At risk of failure, each query should result in a *leader* being dynamically selected and it must have sufficient resources to be able to satisfy the request. This protocol was only proposed, and much of the technical detail has yet to be completed, but the concept of treating the network as a distributed database is a novel idea and this is one of the first protocols to suggest the use of a query language that could be used by people without specific domain knowledge.

2.2.3 Hierarchical

Hierarchical networks are WSNs that contain nodes of different classes; nodes at the end of the network are typically clustered into groups and served by a gateway node. This gateway could be in charge of aggregating the data, processing the data, or simply forwarding it to an endpoint. Clusters of nodes allow the network to be spread out over a wider geographical area and gateway nodes can use a different transmission method to provide long

distance links to the base station. Gateway nodes serving a cluster of nodes means that the network can scale easily as well, simply by adding a new cluster to the network.

TEEN

The Threshold sensitive Energy Efficient sensor Network (TEEN) protocol is designed for reactive sensor networks, networks that require instant reactions to changes sensed in their environment. TEEN recognises that transmission is the most power hungry action for a node so each node is coded with a hard and soft threshold. The hard threshold is a value that makes nodes transmit the reading to their cluster head. Similarly, the soft threshold is a small change in the value of the sensed attribute that causes further transmissions.

During the initialisation of the network, the base station sends information about the thresholds and sensing attributes to all cluster heads in the network; the cluster heads then forward this on to all nodes in their cluster. When a node senses data over the hard threshold, it transmits to the cluster and only transmits again when new sensed values are greater than the hard threshold and the difference between the current sensed value and the previous is greater than the soft threshold [**?**].

Clusters are assigned for a period of time and then new clusters are selected by the base station, at which point new attributes and thresholds are broadcast to all nodes. This kind of protocol allows the network to be dynamic after deployment and allows user input based on the data that has been sensed in the previous cluster times.

For example, a network could be tasked with sensing humidity in a rainforest but the thresholds have been set such that nodes are transmitting readings that are not of interest. A user can change these thresholds and they will be pushed out to the nodes at the time that the next clusters are chosen, without any need to visit the node or configure them individually.

2.2.4 Location-based

Instead of using the physical addresses of nodes, or the data they store, location-based protocols are based on the region that nodes are deployed in. **FLESH OUT**

Span

Span is a protocol where nodes are selected as *coordinators* based on their positions. A node can decide to be a coordinator based on the amount of energy it has and the number of neighbouring nodes it would benefit if they were able to use it as a bridge [**?**].

An example of this would be node B placed between node A and C. C and A are unable to communicate directly so, when node B wakes up, it decides whether it should become a coordinator. It knows that it has sufficient energy levels and it can provide connectivity for a previously disconnected area of the network, so it chooses to become a coordinator, staying awake and routing sensed data to other coordinators, which form the backbone of the network.

Results showed that using Span, in a system that transmits using 802.11, provides an network lifetime increase of more than a factor of 2 over networks that just use the 802.11 protocol.

GEAR

The Geographic Energy-Aware Routing protocol (GEAR) is similar to SPAN in that it makes routing choices based on both energy-awareness and location. Each node maintains an *estimated cost* and a *learning cost* of forwarding a packet through its neighbours. The estimated cost is calculated using the distance to the packet destination and the energy remaining on the node whereas the learning cost is the estimated cost that takes holes in the network into consideration [**?**].

GEAR is designed to perform in two phases: forwarding a packet towards a region and disseminating a packet within a region. When sending a packet towards a destination, GEAR either sends a packet on to the node in range

that is closest to the destination or, if such a node does not exist, then a hole is identified. If a hole is identified then the node that minimises a cost is selected.

To disseminate a packet within a geographic area, uses algorithms based on the density of the network. Recursive geographic forwarding is typically used but this can result in an endless loop if the density of the network means that the region is unable to contact the destination. In that case, restrictive flooding is used.

Provide a concluding section for routing protocols here?

2.3 Sensor Middleware

Acting as a bridge between the hardware and the user, sensor middleware is software that abstracts the underlying network from the user and provides a means of accessing sensed data and administrating how the network performs. These middlewares must not be specific to a single network and provide support for as many different sensor nodes as possible. In [**?**], a middleware is said to provide standardised services to many applications and perform operations that make effective use of limited system resource.

In [**?**], a middleware should include four major components: programming abstraction, system services, runtime support and Quality of Service (QoS) mechanisms. In this section, we will discuss the challenges surrounding middlewares for WSNs and highlight some existing middleware that are particularly relevant to our research problem and/or motivating scenario.

2.3.1 Issues

WSNs present a range of new challenges to existing middleware, due to their resource constraints, deployment environments and more. However, there has been research into the key issues that must be addressed in order for middleware to be considered suitable. While there have been a number of surveys into these challenges [**?**, **?**, **?**], we will detail those that we believe to be most relevant to our work.

Resource Constraints

It is rare that nodes in a WSN would have a constant power source, unlimited memory and a casing that can survive a harsh environment without decaying. In order to ensure that the lifetime of nodes is mazimised, middleware needs to offer a power scheduling system that makes efficient use of the hardware on the node, typically turning off the radio at particular times.

Ideally, a middleware will be able to coordinate nodes through wireless communication, making efficient use of transmissions and dynamically modifying sleep schedules based on the power remaining.

Heterogeneity

Not every node in a network will have the same capabilities, manufacturer or hardware. WSN middleware needs to provide a standard interface to add data, regardless of the format it is recorded in. Some middlewares have been built for a specific set of hardware, however this homogeneity can provide increase the performance and efficiency of the network by only supporting one device.

Real-world Integration

WSNs are often tasked with recording phenomena that are time-crucial, so a sensor middleware should provide a real-time interface to the data that it has sensed. Ideally this data would be available outside of the network, though the use of an API.

Quality of Service

This issue is perhaps the most complex as QoS could apply to almost all aspects of the networks, such as efficiently using bandwidth, 100% uptime for nodes, guaranteed packet delivery or access to data stores. Some of these requirements are managed by the implementation of the routiong protocol, the middleware should be able to monitor deployed nodes and report on their current status, as well as identify failures.

2.3.2 Existing Middlewares

In this section, we identify existing middlewares, explain how they address the issues highlighted in Section 2.3.1 and highlight how they relate to our research. While there are a lot of existing middlewares, our research did not show any that directly utilised the environment to make informed classifications. We did, however, find some applications that utilise context and rules to administrate the network.

FACTS

One such example is the FACTS middleware, an approach that uses a fact repository to coordinate nodes. Rules can then be implemented to process sensed data and fired when certain conditions are met [**?**]. More traditional sensor middlewares control the network and manage sensed data, this rule based approach allows for more flexibility, where rules can control the transmissions and process the data upon receipt.

Figure 2.1 shows the FACTS architecture, with the middleware holding the rulesets and a dstributed fact repository. Data within the network is stored as facts, providing a standard data format throughout the network and hardware abstraction. When new facts are received, ususally because of new sensing data, the rule engine checks to determine whether any rules should be fired. The ruleset definition language (RDL) is introduced here and each ruleset contains a group of relevant rules. Each rule is given a priority so that, if more than one rule is triggered by a fact, then the higher priority rules are fired first.

While FACTS itself does not utilise any local knowledge, the repository is used as a source for all previously sensed data and would prove as an excellent source of knowledge to assist with the classification of future readings. Also, the ability to add new rulesets, without technical knowledge of the hardware of each node, means that users of the network have the ability to add knowledge in the form of rules as they learn it.

ITA Sensor Fabric

The ITA Sensor Fabric is collaboration project between IBM, the US Army and the UK MoD. Sensor Fabric, or Fabric, is a two-way messaging bus and set of middleware services connecting network assets to each other and users [**?**].

The core difference between the Fabric middleware and others is that not every node is sensing all of the time, sensor nodes are tasked when there is a requirement and they stop as soon as that task has been fulfilled. Similar to sinks in a traditional WSN, Fabric utilises Fabric nodes, which run the following three pieces of software:

1. Message Broker - Provides the communication infrastructure.
2. Fabric Registry - Holds information about the current deployment, such as all nodes deployed, all assets, routing information and tasks. Deployed in the form of a database.
3. Fabric Manager - The main service on the node to track the status of connected sensors, establish communication channels, provide a container for processing, plug-ins and to extends the capabilities of the Fabric.

Fabric runs on a Publish/Subscribe model, a sensing requirement is sent to a messaging broker as a subscription and this is distributed through all Fabric nodes and, thus, all sensor nodes. Sensor nodes then publish their data and the relevant data is sent to all applications that have subscribed to the data.

The plugin structure of Fabric makes it stand out from existing middlewares, allowing its functionality to be extended through web interfaces.

Because Fabric has been developed for military purposes that cross countries, policy enforcement has been implemented to restrict access to the granularity of sensed data but these access levels do not simply apply to a military context. Using our motivating scenario, researchers and professors should see animal images whereas the Sabah Wildlife Department should see images of hunters and people in the forest.

GSN

The Global Sensor Networks (GSN) is a middleware that has been developed to manage heterogeneous sensor networks and be suitable for those without any technical knowledge [**?**].

GSN provides hardware abstraction through the use of *virtual sensors*, a data stream that abstracts implementation details from the actual sensed data. A virtual sensor can be comprised of many streams and it can even consist of many virtual sensors.

Virtual sensors are described using XML, with tags that consist of meta-data for the sensor, the structure of the incoming data stream, SQL queries for processing the incoming data and querying times. What makes GSN stand out is that virtual sensors do not have to be sensors deployed within your network, or even sensors at all, some examples of GSN show virtual sensors being added that read in data from the weather websites. This also means that the underlying structure of the network is irrelevant to GSN, as well as the physical locations of the nodes. Unlike some middlewares that have an expectation of how data will be routed, GSN is decoupled from the routing protocol, allowing them to act independently.

GSN is completely open source and, while it does not provide the plug-in architecture that is available in Fabric, the Java code can be modified to suit a specific deployment.

Figure 2.2 outlines the architecture of GSN, showing that the virtual sensors are stored on a central node and their inputs are managed and stored. GSN also comes bundled with a web interface to show all active sensors and their most recent recordings, as well as the implementation of web services to access the data outside of the interface.

Data from virtual sensors pass through the virtual sensor manager to the storage layer. Once the data has been stored, the query manager is invoked and queries are loaded from the repository and executed by the manager. The results of the queries are then handled by the notification manager and also made available to the web interface. Notifications can be extended to support many different forms of communication, such as SMS, email or web

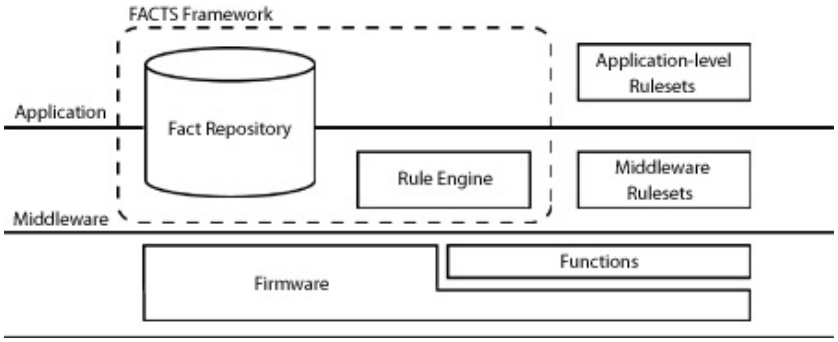

Figure 2.1: FACTS Architecture

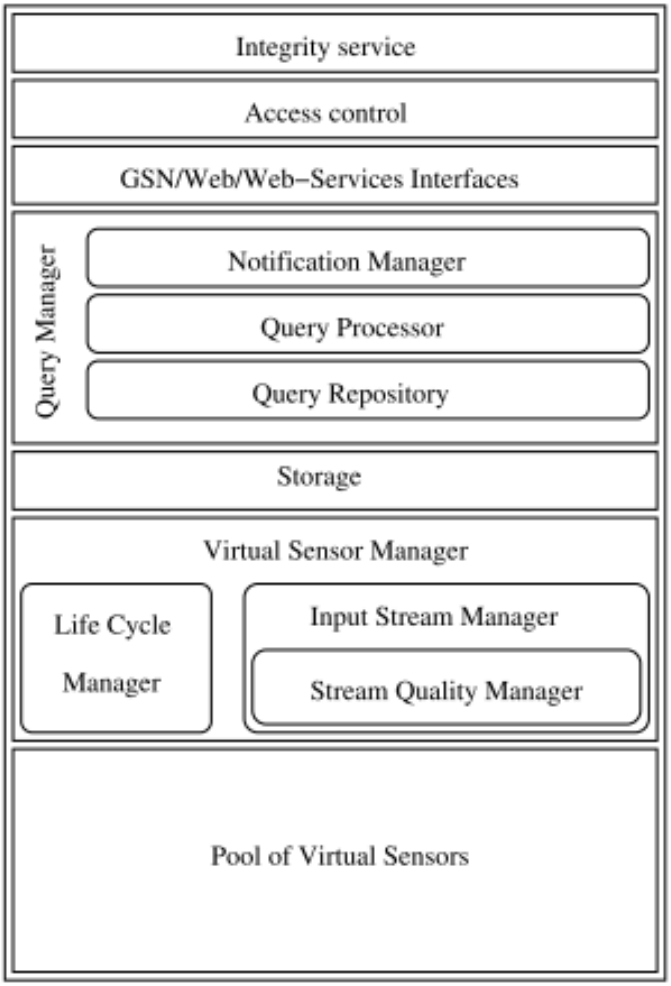

Figure 2.2: GSN Architecture

services.

Virtual sensors do not natively support all hardware, although new virtual sensors can be described using XML, and there may be a need to implement an entirely new virtual sensor. In this case, technical knowledge is required, and new sensors can be implemented through use of the Java programming language. This provides more control over the use of XML and allows users to specify how sensed data is stored in a database, use external libraries to receive proprietary data, specify processing workflows before the data is stored or implement new notification methods for the users of the network.

To show the simplicity of a basic virtual sensor, [**?**] describes a temperature sensor that we reproduce here in Listing **??**. The file is human-readable and has a shallower learning curve than programming languages, with tags that explain what data they contain. In this example, the output structure shows that only a temperature reading is received and that the data should be stored permanently. The stream source specifies the content of the stream and the query details the standard query that should be used to extract data from GSN.

```
<life-cycle pool-size="10" />
<output-structure>
        <field name="TEMPERATURE" type="integer"/>
</output-structure>
<storage permanent-storage="true" size="10s" />
<input-stream name="dummy" rate="100" >
<stream-source alias="src1" sampling-rate="1" storage-size="1h">
<address wrapper="remote"> <predicate key="type" val="temperature" />
<query>select avg(temperature) from WRAPPER</query>
</stream-source>
<query>select * from src1</query>
</input-stream>
\caption{Virtual Temperature Sensor}
\label{bg:lst:gsn}
```

The modularity and flexibility of GSN makes it different to existing mid-

dlewares as it has not been designed for any specific hardware and modules of the middleware can be replaced, such as the database.

2.4 Biodiversity Sensor Networks

In this section we will cover existing WSNs that are related to our motivating scenario or, more specifically, biodiversity focussed WSNs that have been deployed to monitor wildlife and/or the environment. WSNs for habitat, and wildlife, monitoring are especially important because these are areas that often need to be untouched by humans. Areas with high human disturbance can influence the abundance of species and some habitats, i.e. underground burrows, may be impossible to monitor without destruction.

One of the most well known WSNs to monitor habitat is the network deployed on Great Duck Island (GDI), an island off the coast of Maine, USA. A network consisting of 32 nodes was deployed to monitor a bird, known as the Leachs Storm Petrel [**?**]. This network used a clustering approach for groups of nodes to send data to a gateway, which would then route it back to the base station. The base station, located a few kilometres away on the island, has internet access and uploads the data to allow users to browse and process the data.

A multihop approach was used here as they found that, for sufficient coverage, single hop connectivity would not cover all of the island. Acrylic enclosures were developed to ensure the nodes were weatherproofed for the conditions of the island, while maintaining the functionality of each sensor and not impeding transmission range. While the nodes, their casing and their sensors have been designed specifically for the deployment on GDI, the success of the network, running for 123 days in the early stages of WSN research [**?**], shows that this approach can be used elsewhere with similar effects; allowing hard to monitor and/or inaccessible areas to be continuously monitored.

On a smaller scale, INternet-Sensor InteGration for HabitaT monitoring (INSIGHT) is a single-hop WSN that allows remote access for data and reconfiguring of nodes [**?**]. While this network does use commercial hardware,

their findings do show that their nodes could survive for 160 days on a single battery, supporting their claim that a single hop network allows for a longer network lifetime.

The key feature of this network is the ability for humans to remotely set reporting thresholds for sensor nodes. This means a user can prolong the lifetime of nodes by limiting the threshold they report on, as well as the fact that these thresholds are a way for users to add knowledge, albeit primitive, into a network.

While there is research on cameras used to monitor animals [**?**, **?**], these networks are generally cameras deployed with their memory cards manually retrieved and processed. In recent years, however, the use of wireless technologies and image-based WSNs has increased, [**?**] uses wireless cameras to monitor the movement of animals between roads. Using commercial hardware and controlled sleep scheduling, this solution employs the use of nodes to detect movement and wake up more power-hungry camera nodes. While the nodes are wireless, the distance of the network from civilisation means that the data does still need to be collected manually and uploaded to a computer.

Due to the advent of smartphones and tablets, as well as the improvements in 3G technology, projects taking advantage of more modern technologies have grown in popularity. Using 3G enabled cameras, [**?**] have deployed a number of devices in locations all over the world, such as: Kenya, Indonesia and the USA. The images captured are transmitted to a server and a website allows the general public to not only see the images in near real-time, but to classify the images as well. This crowdsourcing of collective knowledge lets people, that may not have domain knowledge, vote on an image and those votes are used to make classification easier.

Over the past fifteen years, WSNs have grown from a concept to a real solution for monitoring the habitats, movements and eating habits of wildlife all over the world. Whether it is using GPS collars to monitor the movement of cattle [**?**], monitoring animal habitats on a remote island or using cameras to capture the animals themselves, the popularity of these networks has grown considerably and advances in technology have allowed these networks

to be deployed in places that humans cannot.

2.5 Local and Global Knowledge

The environment of a sensor network is rich and varied and we believe that patterns in the data sensed can be used to inform the network on decisions surrounding the transmission and processing of newly sensed data. As our research began, we simply called this knowledge but, as our work continued, it became apparent that it could be split further.

While we believe that we are the first to use the concept of local and global knowledge within the wireless sensor network domain, the terms have been around for many years. In 1999, a book that referred to local knowledge as *indigenous knowledge* defined local knowledge as systematic information that remains in the informal sector, usually unwritten and preserved in oral traditions rather than text [**?**].

Over the past twenty years, local knowledge has been used in various contexts, from researching lending and the credit market [**?**] to extracting local knowledge from natives to improve farming techniques [**?**]. This research, as well as work that will be covered later, showed us that there are two kinds of knowledge: global and local.

It was from agriculture research that we were able to refine our definition of local knowledge, [**?**] defines local knowledge as knowledge that farmers have derived locally through experience and experimentation. They also say that indigenous knowledge is different in that it is culturally specific. From this definition, as well as our work with our motivating scenario, we were able to generalise the definition and expand upon it.

We now define local knowledge as *knowledge of an area, held by a domain expert, that has been gained through experience or experimentation.* This then means that global knowledge is *knowledge of an area that can be accessed by anyone, without the need to visit the area directly.* The weather of a region is global knowledge because it can be found through a variety of media, whereas the level of a rain for a field within that region would be local knowledge, as it would require experimentation.

Using these definitions, we believe that encoding local and/or global knowledge onto sensors will inform routing decisions to make better use of the bandwidth in resource-constrained WSNs by sending data that it believes to be important first, rather than just chronologically. Patterns in the data, and knowledge of the environment surrounding a node, will allow a node to infer what the data may be classified as, automate the classification process, learn from previously sensed data and utilise global knowledge of ongoing projects within the network to determine what data is thought to be of a higher priority.

2.6 Relevant Existing Networks

In this section, we discuss existing networks that have done work relating to our research question and/or our motivating scenario. While most of the existing networks relevant to Danau Girang have already been covered in Section 2.4, there are some WSNs that are not directly related to biodiversity but have been deployed in harsh conditions or involve interdisciplinary collaboration.

2.6.1 Context-Awareness

While standard WSNs have been prevalent for many decades, new research on the Internet of Things [**?**] has brought about interest in context-aware sensing, in some cases this is for small wearable devices to track fitness but the applications are much broader. Here we will look at WSNs that use context to make informed routing decisions, save power, or prioritise the transmission of data.

Health

Health monitoring is one of the more obvious choices for context awareness as classifying readings can often help determine the health of someone, rather than their self-reports. AlarmNet is a WSN that uses context to provide long-term health monitoring for people in assisted-living environments.

AlarmNet employs context-awareness to learn about the activity levels of the patient and uses that knowledge to determine when changes in the readings may mean that the patient is at risk. Once the AlarmNet system has completed its learning period, deviations, from what it has recorded as the normal activities for the patient, are sent to nurses and doctors, with the idea that this information can assist with a diagnosis.

As some nodes in the system will be battery powered, AlarmNet also employs a subsystem, called the Context-Aware Power Management System (CAPM), to control the power consumption of a device based on the recorded activities of the patient. The system uses policies, based on the context, to save power for all nodes in the system. For example, the system could put mains-powered nodes into a low-power state and disable all nodes outside of the bedroom when it detects that the patient has gone to sleep.

This use of context has not only shown that it can assist with the lifetime of a network, but it can also provide valuable insight into sensed data to provide data enriched with semantics.

Wearable Devices

As wearable devices, such as the Fitbit and the Pebble smartwatch, are increasing in popularity, context-awareness is a useful tool to differentiate between the many activities that a person can undertake. The MObile PErsonal Trainer system (MOPET) is a wearable fitness devices that uses context to record data on jogging and fitness exercises [**?**].

While the project is five years old now, it is one of the earliest proof-of-concept devices created and shows how simple rules applied to sensor readings can be used to apply context. For example, the device consists of a GPS sensor and, when active, it is constantly recording positions. Using pairs of points, it is able to calculate the speed and, if the speed is consistent with jogging, then it records a running exercise.

Fitness is not the only purpose for wearable computing, the eWatch is an early design for a *smartwatch* that uses a microphone and light sensor to record the locations that the user visits, storing previous recordings in flash

memory and matching them with current recordings [**?**].

UNETS

*UNETS is tiered and context aware

2.6.2 Harsh Environments

There are a number of situations where we may want to record data in an environment that is not safe for humans, such as a volcano [**?**], or that may be difficult for electronics to survive. In these cases, special considerations must be made during the design and deployment of the WSN, in order to ensure maximum network lifetime with reliable readings.

GLACSWEB

Glacsweb is a sensor network to monitor the rate that which glaciers are melting. Deployed in Norway, specially designed sensors have been drilled into glaciers to monitor pressure, temperature, orientation and strain. Due to the high pressure and exposure to moisture, a polyester casing was used so that, once bonded, the node inside would not be recoverable.

Reventador

Volcano Reventador is in northern Ecuador and a WSN has been deployed on there to monitor eruptions. Similar to Glacsweb, weatherproof enclosures were used to prevent ash and moisture from breaking the sensors and long-range external antenna mounted to a pole was used to achieve communications over large distances when wireless communications have proven to be difficult [**?**].

Rainforest

There have been many studies on how the rainforest affects the range and quality of wireless links [**?**, **?**, **?**]. In [**?**], the humidity was shown to reduce 802.11 range by up to 78% and [**?**] explains that periods of rainfall reduce

the link quality up to 100% in some cases, resulting in the loss of a hop and this could prevent data from reaching the endpoint.

2.7 Summary

In this chapter we have explored the components that make a WSN, as well as some existing deployments that are relevant to our motivating scenario. Sensor middlewares have come a long way in the past decade and increased capabilities for sensor nodes have allowed for more intense processing to be carried out before any data is seen by a human. Context is now used on sensor nodes to infer the activity it is undertaking or even to determine when it should wake to sample.

Local knowledge is not a new concept and it has been used for many years to extract information from indigenous people and in industry to adapt their processes to a local area, such as farming. However, we believe that our work is the first to apply local knowledge in the context of WSNs, but context-aware sensor networks support our hypothesis that knowledge of its surroundings, or of the data it has sensed, can improve network performance.

The two primary components of a WSN that could be injected with local knowledge is the middleware and the routing protocol, each providing different benefits. Existing work has shown middleware that uses context-awareness can use that to make global changes to the network, such as power management, whereas routing protocols affect the data that is sampled and sent to the endpoint(s) of the network, such as adaptive thresholds.

One issue in proving this hypothesis is the deployment of a network that utilises local knowledge. The deployment environment of our motivating scenario is not only interdisciplinary, it is in a region that is humid, dynamic and dense. Existing research has shown that the deployment of a WSN in these considerations means that range will be greatly reduced and changes in humidity can prevent communication altogether, as well as moisture affecting the hardware itself. To deploy a network, hardware must be adapted and the right medium must be chosen in order to maximise link quality and minimise dropped connections.

In Chapter 2, we show how we used this research, as well as our own findings, to choose suitable hardware and experiments, that were informed by previous rainforest range tests in the literature, supported our choice for transmission medium.

Chapter 3

Technical

In this chapter, we explain our motivating scenario in more detail, explore the sensor hardware that we researched and outline the results of experiments we undertook in the Malaysian rainforest. As highlighted in Section 1.2, we have been working with Cardiff University School of Bioscience to design and deploy a WSN that utilises local knowledge, using an area of rainforest in Malaysia owned by the Sabah Wildlife Department, called Danau Girang.

The structure of this chapter is as follows. Section **??** explains what we aimed to deploy in Danau Girang and what our considerations were. Secion **??** introduces sensor hardware that is in use today and details the choices we made. Section **??** details the transmission medium choices we tried and also shows the results of experiments performed in both the UK and Malaysia. Finally, Section **??** summarises our findings and explains the choices we made for the sensor nodes we used in DG.

3.1 Danau Girang

Based in Sabah, Malaysia, Danau Girang is a field centre located in Lot 6 of the Lower Kinabatangan Wildlife Sanctuary (LKWS), surrounded by secondary rainforest that had been logged up until the 1970s. Experiencing typical wet and dry seasons, the LKWS can receive more than 500mm of rainfall during the rainy season, dropping to lows of around 150mm during

the dry seasons [**?**], and up to 100% humidity all year round.

Danau Girang is uniquely situated in a rainforest corridor that joins two areas of rainforest together, with the corridor surrounded by palm oil fields on each side. Because of this, animals use the corridor to move between the rainforest regions and some use it to enter the palm oil plantation for new feeding grounds. This gives Danau Girang insight into the movement patterns of these animals in the corridor as well as in the rainforest itself, with a wide variety of species that are not commonly seen in other tropical regions of the world. Due to the remote nature of the centre, power is provided by a set of diesel generators which, typically, provide power from 10 am to 1 pm and 5pm to 11pm daily. Wireless Internet access is provided by satellite with speeds comparable to that of 56k, although the upload speeds are considerably faster than downloads.

As outlined in Section 1.2, the corridor monitoring programme is a scheme that has been in place for more than five years to use wildlife cameras to track the movement of animals through the corridor and to capture species that are rare or unique to South-East Asia, such as the Bornean Clouded Leopard or Sun Bear. Currently, the use of Reconyx Hyperfire HC500 cameras are being used [**?**]. These are standalone cameras that store a pre-defined set of images to an SD card on each trigger, which is triggered by an infrared (IR) motion sensor when the beam is broken.

Images must be collected manually every two weeks from the cameras and the batteries are changed at that point as well, although a typical charge should last three months. The cameras are equipped with watertight casing but, due to the humidity and the opening of the cameras every 2 weeks, silica gel is used to prevent moisture inside the camera. We believe that humidity also reduces the battery life, as the charge drops from three months to around three weeks within the first few months of usage. However, the lack of constant power availability could also negatively impact the charge cycles of the batteries when at the field centre, reducing their capacity. Each unit is secured to a tree and more dangerous sites, such as known elephant paths, have protective cases as well.

In 2010, twenty cameras were deployed for six month periods and then

relocated based on the needs of the projects at that time. As of 2013, there are now ninety with a view to expand and dozens of projects within the field centre use the images gathered from these cameras. Initially, it was the job of visiting research students to collect the images but, since the number of deployed cameras has grown, full-time staff have been taken on to maintain them.

Our belief is that we can use the LKWS, and the locations of the existing cameras, to deploy a WSN that uses local knowledge gained from the researchers at DG to automate the collection of images, improve the battery life by not exposing the internals of the camera to the elements so often and, most important, prioritise the flow of data through the network by in-network processing.

Annual visits, lasting three weeks, have been made to DG to test out hardware, software and wireless choices, in an effort to optimise the network. These visits have also been used to extract local knowledge from the area and researchers, by semi-structured interviews and watching them work.

3.2 Hardware

Before any visits were made to DG, meetings with staff members of the field centre were held in order to gain a better understanding of the environment and the project. This is where we were alerted to the humidity of the region and the fact that the failure rate of the Reconyx cameras has been as high as thirty per cent.

Reconyx cameras have no external interface support and the only way to access the images is through the removable SD card, because of this there is no way of attaching external sensor hardware to the existing cameras. In-network processing is an important requirement for our WSN and this did limit our choices to nodes that are computationally capable than more common sensors, such as the IMote 2 [**?**].

In this section, we detail our research into suitable sensor hardware that met the following requirements:

1. Able to perform processing of images and metadata
2. Common interface availability (Serial, USB)
3. Wireless enabled
4. Battery-powered
5. Expandable memory

This section also details the modifications we made to the devices in order to ensure they would survive in a humid environment.

3.2.1 Pandaboard

Texas Instruments supported the development of a reference Single Board Computer (SBC) that had specs similar to that of a modern smartphone and was capable of running desktop Linux, known as the Pandaboard [**?**]. A dual core 1GhZ ARM processor with 1GB of RAM, support for external storage, expansion ports, USB, Wi-Fi and Bluetooth in a board the size of two credit cards can be a powerful addition to a data heavy WSN, especially one that deals with images.

There is no mention of Pandaboards in the literature being used in WSNs but, the low power of the system and advanced capabilities, make it suitable for processing and transmitting data simultaneously.

3.2.2 IGEP v2

The IGEP v2 is another ARM based SBC that uses a 1GhZ single core processor with 512MB RAM and similar connectivity features to the Pandaboard, but around half the size. This does result in a reduced power draw and the device is still capable of running desktop Linux.

Due to the smaller size, and easier commercial availability, the IGEP has been used as a sensor node to record, process and send readings from multiple devices, such as air temperature, GPS and oxygen saturation as part of environmental monitoring [**?**]. In their research, they found that the IGEP

achieved 9.1 hours of uptime using a 4000mAH battery, a capacity used in many modern smartphones.

3.2.3 Waspmote

The Waspmote is a general purpose sensing board that is designed to allow plug-and-play connectivity for multiple sensor modules. The node can be programmed with C++, using the prepackaged SDK. The processing power is not comparable to the more powerful SBCs, but the 600mAh battery is reported to be three months and the size is much smaller [**?**].

One of the benefits of the Waspmote is that they are commercially available with an actively maintained programming environment.

3.3 Transmission Medium

In this section, we explain the experiments we carried out to test the performance of different transmission media in the UK as well as the Malaysian rainforest. While range is the most important feature, a data rate that can handle hundreds of large readings in a day is a requirement. Our motivating scenario is focussed on the transmission of sets of 3 images for every trigger; where a sensor can trigger hundreds of times in a day.

3.3.1 Wi-Fi

Wi-Fi was already available on our initial test platforms and the high data rate made it suitable for sending a large volume of images in a short period. We knew that current cameras deployed in DG were up to 1km apart and we did not expect to cover that range completely, but were did anticipate that coverage with intermediate nodes.

Research, outlined in Section 2.1.4, showed us that the rainforest could reduce the range by up to 78% and the ideal maximum range of 2.4GhZ Wi-Fi is 100m [**?**].

We tested Wi-Fi range using two IGEP boards, powered by 4 D Cell batteries and running a lightweight Linux operating system. The IGEP nodes

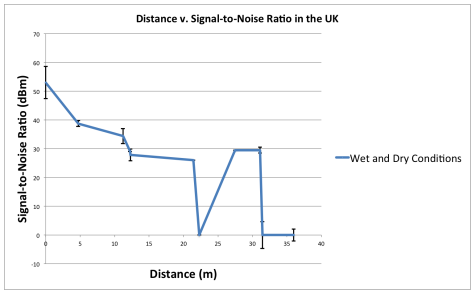

Figure 3.1: Signal-to-Noise Ratio for Wi-Fi in UK Woodland

we used did not have any additional hardware and the nodes were tested without the use of an external antenna. A Java application was written to periodically scan for available networks and store those results in a text file. One IGEP board was set as the base station and attached to a tree, at the same height it would be if it was attached to a camera, and another was walked to specified points around the base station at defined locations. These locations were chosen to include as many distances as possible and as many different forms of obstacle between the searching node and the base station, such as: line of sight (LOS), medium vegetation or thick trees.

This experiment was run in a wooded area in the UK and in the rainforest at DG. The specified maximum range of 802.11g is 120m. When considering attenuation and obstacles we were expecting the signal to be reduced by up to 50% in the UK. However, we found that we received a maximum range of 30m, with LOS. Figure 3.1 shows the results we experienced, while testing in Cardiff, some of the drops in signal can be attributed to dense foliage and readings that were not LOS, but a maximum range of 31m, with an SNR of 29.5 dBm, is less than we expected, as the UK does not experience high humidity often.

The graph does show a drop at 22m, this was due to the dense foliage that restricted the LOS between the base station and the receiving node, with five runs of this test we observed the same results. The primary aim of this experiment was to prove the viability of Wi-Fi and to to ensure our application functioned as intended, which it did. Further experiments could have been run to remove the anomaly but the results of the experiments in

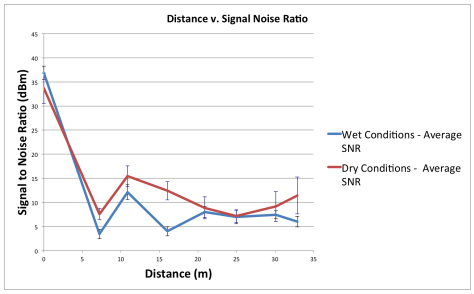

Figure 3.2: Signal-to-Noise Ratio for Wi-Fi in Malaysian Rainforest

Danau Girang were the focus.

Despite the poor range from the tests in the UK, it was consistent with other studies reporting signal degradation of up to 78% in areas with moderate foliage. We visited Danau Girang in 2011 to gather the requirements of the network and ensure the hardware is able to survive the humidity. Range experiments were run in the rainforest to see if a more humid environment impacts range any further, Figure 3.2 shows this.

Comparing figures 3.1 and 3.2 shows that the maximum distance to receive a signal is approximately the same in Malaysia as it is in the UK. There are more signal drops but this seems to be due to denser foliage blocking the line of sight. However, it does suggest that the humid environment of the rainforest does not have a significant impact on the received signal. It is clear that the denser rainforest does impact the signal-to-noise ratio in a much shorter distance from the base station but a link is still made, allowing for a successful transmission of data.

Due to the poor results of these experiments we researched alternative methods to increase the range without impacting the environment the network is to be deployed in. We considered using intermediate nodes, not attached to cameras, to account for the lack of range but, because some cameras can be up to 1km apart, we would need more than 30 nodes to create a connection between two locations.

We also researched wireless technologies that are more common in sensor networks. This does mean that the data rate is not as high as Wi-Fi and error correction in packet streams is not always as robust, but it is more

suited to sensor networks, using less power and providing longer range.

Finally, we considered using the researchers or animals at Danau Girang, as 'data mules', creating temporary links between nodes while they are in the forest. However, the trip to Danau Girang yielded the information that researchers generally do not cover those distances in the forest and data delivery would be sporadic.

Although the range of Wi-Fi is poor, for our requirements, in both Malaysia and the UK, it did show that the results we experience in the UK are very similar to the results in Malaysia. This means that tests run in the UK should be indicative of what we can expect in Danau Girang.

3.3.2 Digimesh

Due to the poor range results of Wi-Fi, we created a second prototype of the network, using Digimesh. Digimesh is a proprietary wireless protocol, based on the 802.15.4 standard and designed for devices with limited power. Using the same frequency as Wi-Fi, Digimesh has been reported to provide 7km of range, with a data rate of 250kbps.

In our prototype implementation, we are using Waspmote sensor boards [**?**], a general purpose node that is capable of transmitting through various communication mediums. Our Waspmotes are provided with Digimesh modules and a 2GB SD card to store sensed data.

When testing the range of the Waspmotes, we followed a similar method to that which is outlined in Section **??**. One board is in a fixed location and running a C++ application to poll for nodes in the network, once a node is found it sends a message to the node every 10 seconds. The second board is set to scan the network and receive packets as soon as a base node is found, this node is then moved to different locations.

The receiving node prints out variables related to the received packet, such as: RSSI, source MAC address and packet ID. However, not all packets are received so the RSSI can display 0 if there are errors reading or if packet collision occurs. We found this to affect the results and have just used the two nodes to identify the maximum distance they can be apart, while maintaining

a stable connection.

Initial experiments were run in a moderately vegetated area in the UK which yielded 497m of range. Limitations with buildings preventing us from testing any further but the signal strength still proved to be strong.

The initial results for the range tests proved positive and Digimesh does seem to be a viable solution to account for the lack of range when using Wi-Fi. As the frequency is the same as 802.11g, thus licensing it for worldwide use, we expected similar results in Danau Girang..

Experiments were run in 2 areas of the rainforest around Danau Girang and the results yielded were not the same as we experienced in the UK, and thick vegetation proved to have a significant impact on the range, reducing it by almost 50%.

In more open areas of the rainforest, we achieved 199m on average, more dense regions of the forest reduced this to 102m on average. While these results are not as high as we achieved in the UK, they are still suitable to use Digimesh in the deployment of a WSN.

3.3.3 Adapting for Harsh Environments

All of the hardware that we used for experiments had their components exposed and would have become compromised if moisture came in contact with them. To protect them from this, we used waterproof cases with a protective foam inside, known as Pelican cases, to keep the nodes watertight, but still allowing airflow, shown in Figure 3.3. External antenna can be fed through the lip of the case, using thin cable, ensuring that the range of the transmissions is not affected by the case.

3.3.4 Commercial Solution

Using commercial hardware, either bespoke for sensing or more general-purpose, our experiments showed that the range experienced in practice is much less than what is expected. Due to the lack of available interfaces available on the Reconyx cameras in use at Danau Girang, any sensor node would have to be connected to an external camera. Because of these factors,

Figure 3.3: Pelican Waterproof Case

we began to look into commercial solutions that combined wildlife cameras with wireless transmission capabilities. -

3.4 Software

3.4.1 GSN

Covering the sensor middleware in Chapter 2, our implementation in Danau Girang required a simple, extensible middleware that could be used by those without technical knowledge. Open source was also a feature that we looked for as it meant that we could easily edit the functionality of existing sensor support, as well as add support for sensors at a much lower level.

Our initial plan was to deploy the middleware on every node, so we wanted to use something that did not need to be compiled for an individual device. The Global Sensor Networks (GSN), covered in Section 2.3, middleware was chosen because it was implemented in the cross-platform Java programming language and it allowed for the description of sensors in XML, a language that can be used by those without any domain knowledge. Its extensive support for sensors and administrative web interface allows users to see the latest sensed data and view the location of each sensor on an interactive map. The open source nature allowed us to remove features to deploy a more basic version to be deployed on sensor nodes and implement new notification methods when sensed data is received, such as Twitter and SMS.

3.4.2 Darwin Core

To pass sensed data through the network, we first had to choose a standard format that would allow us to encode the sensed data, as well as enrich it with inferences made through processing. Darwin Core (DwC) is a body of standards with predefined terms that allows for the sharing of biodiversity occurrences through the means of XML and CSV data files [**?**].

The Global Biodiversity Information Facility (GBIF) indexes more than 300 million Darwin Core records published by organisations all over the web, allowing datasets that were previously siloed from the public to be accessed by both human and machine. The main of Darwin Core it to provide a common language for sharing biodiversity data that reuses standards from other domains [**?**].

DwC follows a star record structure, where a record can contain many occurrences, which is the recording of a species in nature or in a dataset. In an occurrence, there is an *event*, a recording of a species in space and time, enriched with other terms such as *identification* and *location*.

The core file, *meta.xml*, contains all of

3.4.3 Triton

3.5 Summary

In this chapter, we have shown the current hardware choices available for more computationally capable sensors and detailed our experimental results on how rainforest environments impact wireless transmissions. Although technologies, such as Wi-Fi, provide a high data rate, their range is limited and not suitable for sparse sensor networks; especially in a humid environment. Newer technologies, designed for long-range communication in sensor networks, are becoming increasingly more viable and, while they do have lower data rates and less robust protocols, they are more suitable for a resource constrained WSN that requires minimal power draw when transmitting. Using general-purpose hardware also means that there is no protection against water, humidity and animal or human intervention. We have adapted

existing waterproof cases to suit our needs and two week deployments have shown that they are able to prevent any moisture from entering the case.

Chapter 4

Architecture

In this chapter, we propose a network architecture that uses local and global knowledge to make informed routing decisions and to classify sensed data within the network. Our approach, K-HAS, uses a three tiered approach with each subsequent tier providing increased knowledge processing capabilities.

We believe that sensors capable of processing knowledge will provide a more efficient network and be able to prioritise data delivery that it believes to be important, rather than chronologically. We also believe that human input is a valuable learning process for such a network and feedback, from humans, on data that has been classified can be used to inform future classifications. To prove this, we have developed an architecture that uses different levels of knowledge processing throughout the network, the Knowledge-based Hierarchical Architecture for Sensing (K-HAS).

The rest of this chapter is structured as follows. Section 1 outlines the main aims of K-HAS and what it is capable of that typical sensor networks are not. Section 2 introduces an example, from our motivating scenario, that will be used to better explain each tier of the architecture. Section 3 explains the data collection tier. Section 4 explains the data processing tier. Section 5 explains the data aggregation tier. Section 6 concludes the chapter and summarises the key features of K-HAS.

4.1 K-HAS

K-HAS has been designed as an architecture for WSNs that is able to handle changes in the data sensed, as well as the structure of the network. By pushing knowledge bases out to the edge of the network, all nodes in the network have some awareness of the data they are sensing, as well as how important it is, based on the current projects that the network is involved in. This is achieved by using rules with different levels of granularity based on the knowledge processing capabilities of that tier.

4.2 Scenario

In order to explain K-HAS more coherently, we will use an example from our motivating scenario that will show how sensed data is enriched, classified and prioritised as it moves through the network. In this example, a network of wireless cameras nodes are deployed in the Malaysian rainforest, tasked with sensing the movments of animals through a specific corridor of the rainforest.

From previous research, we know that animals that are not of interest often move through that region of the forest at all times of the day, such as macaques and wild boar. However, researchers at Danau Girang also hypothesise that rare species, such as the clouded leopard, move through the corridor when certain conditions are met. These conditions are: a temperature between thirty and thirty five degrees celsius, a night with a full moon and a time between one a.m. and four a.m.

The network is using the K-HAS architecture and has been tasked with prioritising the transmission of clouded leopards, but they also want to receive all pictures; regardless of the content.

4.2.1 Data Collection

The data collection (DC) tier is a very similar to standard nodes in a typical WSN, using hardware that has similar capabilities. These DC nodes are deployed at the edge of the network and tasked with sensing their environment,

pre-processing the sensed data and using each other to relay data to the next tier.

DC nodes are capable of performing processing on data, such as the time it was recorded and its size, but their limited knowledge processing capabilities allow them to have an increased battery life and reduced size, making them suitable for a variety of deployments.

Knowledge Base

Reduced knowledge processing capabilities and low memory restrict the knowledge that these nodes can hold and they are limited to a static knowledge base that is encoded at the time of deployment. DC nodes only perform simple operations on the properties and content of the data that they sense, such as the time it was recorded, the location and its size. For more complex data, such as images and video, DC nodes do not possess the computational power required to process them and instead use the metadata associated. Unlike modern rule engines, these static rules do not use forward chaining and the outcome of one rule does not cause the rules to be fired again. Listing **??** shows an example of some of the rules in the knowledge base.

Listing 4.1: Example DC Node Rules

```
if(reading.dateCreated.month == JUNE AND reading.
   timeCreated.between(17:00, 19:00)
        data.write( Potential Otter sighting )

if(reading.temp == 37 AND reading.timeCreated.between
   (01:00, 05:00)
        data.write( Potential Leopard sighting )
        data.write( PRIORITY=HIGH )
```

When the data is recorded by the DC node, the knowledge base is fired and inferences are made about the contents of the data. Each DC node has a different knowledge base encoded based on the local knowledge of the area that it is deployed in. For example, a node deployed on the bank of a river

would have a different knowledge base to a node deployed in the fields of a plantation.

Once a trigger has been processed, the data is packaged and then sent on to the Data Processing (DP) node.

4.2.2 Data Processing

DP nodes act as cluster heads of the network, serving a subset of all deployed DC nodes. When data is sensed, it is forwarded through all DC nodes to the DP node that is tasked with serving the originating DC node. These nodes have more knowledge-processing capabilities than a DC node and do not typically do any direct sensing.

Due to the greater capabilities, DP nodes have a much shorter battery life and a network typically consists of fewer DP nodes. This also allows DP nodes to run a complete rule engine and process complex data. When a DP node receives data, it processes everything associated, this includes metadata, the data itself and the inferences made by the DC node. If the DC node infers that the data is of a higher priority, then this data is processed first.

In our current implementation, DP nodes use two different radios, a Zigbee radio to allow long range communication from DC nodes and a Wi-Fi radio that provides short range communication that allows for higher data rates.

Knowledge Base

In our motivating scenario the network is image-based, this means that the DP node would perform image processing, as well as processing the image metadata. The increased knowledge processing capabilities allow DP nodes to run rules dynamically, learning from the sensed data and providing classifications that change based on changes in the environment. For example, if a DP node has not seen an elephant before, and it is not aware of the object in the image, then it will await a human classification. The node will then record the time period that it receives elephant pictures, i.e. June to July,

and become more alert the following year. Similarly, the node will know not to look for pictures of nocturnal animals during the day. This local knowledge allows processing power to be saved and, thus, time; this ensures that the processing of sensed data is optimised as much as possible in order to reduce the time it spends in the network.

The rule engine used in our current implementation is Drools, a Java based rule engine that allows for rules to be defined in *.drl* files and these can be loaded dynamically into a knowledge base. This flexibility allows to be changed on the fly without the need to restart the device, or even require human access, as all of this can be achieved through network communication.

Upon receiving sensed data from a DC node, the rule base is fired on the metadata of each file received. If the rules determine that the data is of interest or, in the best case scenario, provides a classification, then the data is packaged and sent on to the Data Aggregation (DA) node.

4.2.3 Data Aggregation

Placed at the edge of the network, these are nodes with high knowledge-processing capabilities and would be accessible by users of the network. When DA nodes receive sensed data, it is unpacked and stored in a folder that represents the node that it originated from.

Any information added by the DP node is parsed and classifications are extracted. If a classification is found, it is stored and the DA node checks for any active projects that contain the classification. If a match is found then all users involved in the project are informed via their preferred method of communication. Using the motivating scenario as an example, the people involved with projects could be researchers and professors and they may be looking for images of leopards, requesting to be informed via Twitter.

All sensed data received, regardless of whether it has been classified, is accessible through a web interface hosted by each DA node. The interface shows all of the sensed data from each deployment, along with the associated classification. More importantly, it allows users to classify the data using a voting system. Users have roles which give them different privileges within

the system. Normal users are able to vote and the majority vote is seen to be the current classification.

However, privileged users are able to confirm a classification and prevent any further votes. Once a classification has been confirmed, it is then sent back to the DP node it originated from. If the classification made by a user is different to the one inferred by the node, then it updates its knowledge base and acknowledges receipt.

Knowledge Base

DA nodes do not typically experience the resource constraints that DC and DP nodes must compensate for. Because of this, they hold a global knowledge that contains a history of all observations made by all nodes, as well as the location and deployment times of all nodes in the network.

While DC do not store any of the observations they capture, and DP nodes only store part of the observation that can be used in future classifications, DA nodes store the complete observation made by every DC node, as well as any extra data that is added by users upon receiving the sensed data.

As well as this, DA nodes provide administrative operations on the network, such as the recording of node locations, time of deployment and viewing all active nodes. This allows the DA node to monitor active nodes and alert users if a node has not sent any data in a while.

4.3 Technological Components

In this section, we describe the technologies used in every layer of K-HAS and how they integrate in order to use local knowledge based on their respective knowledge processing capabilities. The majority of components, both hardware and software, used in K-HAS are used so they are applicable for any WSN, but some choices have been made to remain in line with our motivating scenario and, thus, are more specifically suited for the capture of scientific observations.

4.3.1 Data Standard

To pass sensed data through the network, we first had to choose a standard format that would allow us to encode the sensed data, as well as enrich it with inferences made through processing. Darwin Core (DwC) is a body of standards with predefined terms that allows for the sharing of biodiversity occurrences through the means of XML and CSV data files [**?**].

The Global Biodiversity Information Facility (GBIF) indexes more than 300 million Darwin Core records published by organisations all over the web, allowing datasets that were previously siloed from the public to be accessed by both human and machine. The main of Darwin Core it to provide a common language for sharing biodiversity data that reuses standards from other domains [**?**].

DwC follows a star record structure, where a record can contain many occurrences, which is the recording of a species in nature or in a dataset. In an occurrence, there is an *event*, a recording of a species in space and time, enriched with other terms such as *identification* and *location*. The core files in a Darwin Core archive are:

1. Meta.xml
2. EML.xml
3. Data files

Ecological Metadata Language (EML) is a metadata used by ecologists and the language is used to describe projects and those involved. This file acts as a form of certificate and descriptor as to what the data is related to and who owns it. The XML file, shown in Listing 4.2, outlines a sample project and users involved in the project.

Listing 4.2: Darwin Core: EML.xml

```
<?xml version='1.0' encoding='utf-8'?>
<eml:eml xmlns:eml="eml://ecoinformatics.org/eml-2.1.1"
    xmlns:md="eml://ecoinformatics.org/methods-2.1.1"
    xmlns:proj="eml://ecoinformatics.org/project-2.1.1"
```

```
    xmlns:d="eml://ecoinformatics.org/dataset-2.1.1"
    xmlns:res="eml://ecoinformatics.org/resource-2.1.1"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-
        instance"
    xmlns:dc="http://purl.org/dc/terms/"
    packageId="e71fda1c-dcb9-4eae-81a9-183114978e44/eml
        -1.xml" system="GBIF-IPT" scope="system">
    <dataset>
        <alternateIdentifier>e71fda1c-dcb9-4eae-81a9
            -183114978e44</alternateIdentifier>
        <title>Images from Danau Girang during the PTY
            Project 2011-12</title>
        <creator>
            <individualName>
                <givenName>Christopher</givenName>
                <surName>Gwilliams</surName>
            </individualName>
            <organizationName>Cardiff University</
                organizationName>
            <positionName>PhD</positionName>
            <address>
                <city>Cardiff</city>
```

The core file, *meta.xml*, shown in Listing 4.3 lists the files that contains the actual sensed data, as well as the terms used to describe it. Examples include: date, time, location, type of data, filename and species contained.

Listing 4.3: Darwin Core: meta.xml

```
<?xml version="1.0" encoding='utf-8'?>
<archive xmlns="http://rs.tdwg.org/dwc/text/" metadata=
    "eml.xml">
        <core encoding="UTF-8" linesTerminatedBy="\n"
            fieldsTerminatedBy="," fieldsEnclosedBy=''
            ignoreHeaderLines="1" rowType="http://rs.
```

```
    tdwg.org/dwc/terms/Occurrence">
        <files>
                <location>set.csv</location>
        </files>
        <id index="0" />
        <field index="0" term="http://rs.tdwg.
            org/dwc/terms/occurrenceID" />
        <field index="1" term="http://rs.tdwg.
            org/dwc/terms/basisOfRecord" />
        <field index="2" term="http://rs.tdwg.
            org/dwc/terms/recordedBy" />
        <field index="3" term="http://rs.tdwg.
            org/dwc/terms/associatedMedia" />
        <field index="4" term="http://rs.tdwg.
            org/dwc/terms/eventDate" />
        <field index="5" term="http://rs.tdwg.
            org/dwc/terms/eventTime" />
        <field index="6" term="http://rs.tdwg.
            org/dwc/terms/locationID" />
        <field index="7" term="http://rs.tdwg.
            org/dwc/terms/scientificName" />
        <field index="8" term="http://rs.tdwg.
            org/dwc/terms/identifiedBy" />
        <field index="9" term="http://rs.tdwg.
            org/dwc/terms/dateIdentified" />
</core>
<extension encoding="UTF-8" linesTerminatedBy="
    \n" fieldsTerminatedBy="," fieldsEnclosedBy=
    '' ignoreHeaderLines="1" rowType="http://rs.
    gbif.org/terms/1.0/Image">
        <files>
                <location>images.csv</location>
        </files>
```

```
            <coreid index="0" />
            <field index="1" term="http://purl.org/
                dc/terms/identifier" />
        </extension>
</archive>
```

Data files contain the actual sensed data, based on how it is supported, and these files are linked in *meta.xml*. For example, temperature readings or direct human sightings would be stored in a CSV file and linked, however, images or video would require the metadata to be store in a CSV file and a filepath would be referenced in the XML. The structure of the CSV file contains a header line that matches the terms in the meta file and each line would be an observation. The terms are linked to the Darwin Core glossary so the archive can be validated and processed by a DwC archive reader.

All of these files are then archived and sent as a ZIP folder throughout the network. If the sensed data is media based, then the media is included as well. DwC archive processing libraries are included on both DP and DA nodes.

Darwin Core is suited to K-HAS because it fits our motivating scenario and the archive can be easily created by a DC node, as it does not require any heavy processing and all of the files are common formats.

4.3.2 Middleware

The knowledge-processing capabilities of DA and DP nodes are the same and this is part of what makes K-HAS different from most other WSNs; both types of node run the sensor middleware, but each for different purposes. DA nodes use the middleware for administrating the network, receiving and archiving sensed data and allowing users to provide classifications. DP nodes use if for the receiving, sending and controlling the flow of processing of sensed data before it is passed on.

Existing suitable middlewares have been detailed in Section 2.3.2 and our requirements for K-HAS were partially determined by the expertise of the

users in our motivating scenario. Below is a list of our three core requirements:

Portability Heterogeneous WSNs utilise nodes with different architectures and capabilities, if middleware is to be used on the nodes it must be able to run on these varied devices.

Usability Users of K-HAS should not be expected to have knowledge of computer science or the underlying architecture, this network should be usable by almost anyone. The same must be said for the middleware as well.

Extensibility A closed-source middleware can be used, but it must then support all sensor nodes and data types, as well as receive regular updates. Open-source, or extensible, middleware can be used to add support for newer nodes.

GSN is a Java-based open-source middleware. New generic sensors can be added through XML files, while more complex sensors can be added through custom Java classes. GSN is covered in more detail in Section **??**. Because GSN can run on any architecture that supports the Java Virtual Machine (JVM) then it meets our portability requirements and the web interface to provide administrative functionality makes it usable by those without any domain knowledge. Finally, the ability to add new sensors through XML means that it can be extended by almost any user of the network and with very little guidance.

4.4 Implementation

In this section, we discuss the implementation made for our motivating scenario, a deployment where the hardware choices were using the 'cutting edge' technology that was readily available and software packages that can be reused in any scenario.

Experimentation in the Malaysian rainforest revealed that our ideal implementation of K-HAS was not feasible for a prototype that could be used

by those without any domain knowledge. This was mainly due to the effects of the humidity and vegetation on the range of the DC nodes, but also because of the difficulty creating a DC node with a camera that could be remotely controlled.

The key point of K-HAS' development was to show that local knowledge can improve the efficiency of the network, the quality of the data received and automate the processing of sensed data. In order to implement a network that showed this, we had to modify the architecture of K-HAS to create a network that was designed specifically for its use in Malaysia, as well as using commercial hardware that could be used by anyone.

While this implementation does not exactly match the architecture of K-HAS, the principles are the same and the implementation is based on the best technology that is currently available. We call this modification to K-HAS, the Local-knowledge Ontology-based Remote-sensing Informatics System (LORIS).

4.4.1 Hardware Used

Data Collection

Data Processing

Data Aggregation

4.4.2 Software Used

Data Collection

Data Processing

Data Aggregation

4.4.3 Walkthrough