http://researchcommons.waikato.ac.nz/

Research Commons at the University of Waikato

Copyright Statement:

The digital copy of this thesis is protected by the Copyright Act 1994 (New Zealand).

The thesis may be consulted by you, provided you comply with the provisions of the Act and the following conditions of use:

- Any use you make of these documents or images must be for research or private study purposes only, and you may not make them available to any other person.
- Authors control the copyright of their thesis. You will recognise the author's right to be identified as the author of the thesis, and due acknowledgement will be made to the author where appropriate.
- You will obtain the author's permission before publishing any material from the thesis.

Semantic In-Network Complex Event Processing for an Energy Efficient Wireless Sensor Network

A thesis

submitted in fulfilment

of the requirements for the degree

of

Doctor of Philosophy

in

Computer Science

at

The University of Waikato

by

Mumraiz Khan Kasi

THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

© 2015

Dedicated to My Parents

Abstract

Wireless Sensor Networks (WSNs) consist of spatially distributed sensor nodes that perform monitoring tasks in a region and the gateway nodes that provide the acquired sensor data to the end user. With advances in the WSN technology, it has now become possible to have different types of sensor nodes within a region to monitor the environment. This provides the flexibility to monitor the environment in a more extensive manner than before.

Sensor nodes are severely constrained devices with very limited battery sources and their resource scarcity remains a challenge. In traditional WSNs, the sensor nodes are used only for capturing data that is analysed later in more powerful gateway nodes. This continuous communication of data between sensor nodes and gateway nodes wastes energy at the sensor nodes, and consequently, the overall network lifetime is greatly reduced. Existing approaches to reduce energy consumption by processing at the sensor node level only work for homogeneous networks.

This thesis presents a sensor node architecture for heterogeneous WSNs, called SEPSen, where data is processed locally at the sensor node level to reduce energy consumption. We use ontology fragments at the sensor nodes to enable data exchange between heterogeneous sensor nodes within the WSN. We employ a rule engine based on a pattern matching algorithm for filtering events at the sensor node level. The event routing towards the gateway nodes is performed using a context-aware routing scheme that takes both the energy consumption and the heterogeneity of the sensor nodes into account.

As a proof of concept, we present a prototypical implementation of the SEPSen design in a simulation environment. By providing semantic support, in-network data processing capabilities and context-aware routing in SEPSen, the sensor nodes (1) communicate with each other despite their different sensor types, (2) filter events at the their own level to conserve the limited sensor node energy resources and (3) share the nodes' knowledge bases for collaboration between the sensor nodes using node-centric context-awareness in changing conditions. The SEPSen prototype has been evaluated based on a test case for water quality management. The results from the experiments show that the energy saved in SEPSen reaches almost 50% by processing events at the sensor node level and the overall network lifetime is increased by at least a factor of two against the shortest-path-first (Min-Hop) routing approach.

Acknowledgements

I am extremely grateful to my chief supervisor Dr. Annika Hinze for trusting, encouraging, and supporting me all throughout my PhD studies. With advices and lively discussions, she has contributed a lot to the development of this work. Her advices have been invaluable and go well beyond the pursuit of an academic title and the fulfilment of a research project. I am also grateful for the precious advices and suggestions of my co-supervisors, Dr. Catherine Legg and Associate Professor Steve Jones, whose experience and support played a crucial role in the progress of the work.

I want to thank all the past and present members of the ISDB group for their friendly, positive, and helpful attitude. Thanks to Michael Rinck and Carole Chang: working with you guys has been a great experience. I would also like to express my heartiest thanks to Adeel Mahmood, Bilal Raza and Hasan Alyamani for being such a good colleagues and friends: I wish you all the best for a future full of achievements. My sincere thanks to all of my friends here in Hamilton, especially to Irfan Habib, Noor Muhammad and Khalid Khan. Their companionship has been a source of great relief and entertainment in this intellectually challenging journey.

I remain indebted to my parents and siblings, who always kept me away from family responsibilities and encouraged me to concentrate on my study. In spite of the distance, they were an incessant source of encouragement. Loving thanks to my wife, Sara, for her constant support, cooperation and taking care of our lovely daughter, Mashal, especially during the final stages of my studies. My gratitude for my parents, siblings, and my wife is beyond the words.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent times, the use of Wireless Sensor Networks (WSNs) in various applications in industry and government has significantly increased [32, 68, 76]. The benefit of a WSN is that it allows users to remotely observe the physical conditions of an environment [62, 91]. WSNs can gather a wealth of information over prolonged periods in potentially inhospitable environments [59].

A WSN is composed of sensor nodes and gateway nodes. Sensor nodes are small, low-cost devices that are distributed to perform monitoring tasks [33, 82]. These sensor nodes are capable of sensing, processing and communication. Generally, sensor nodes sample their sensors and send sample readings (i.e. sensor data) to gateway nodes. Gateway nodes are more powerful than the sensor nodes in terms of processing and communication capabilities. Gateway nodes receive data from the individual sensor nodes, process it and send the results to the end-users [8].

In most applications of WSNs the end-users are interested in *events*. An event can be described as a meaningful change in sensor readings, such as *when the phosphorus concentration in a body of water is less than 10 mg/L* or *when the nitrogen concentration in a body of water is more than 15 mg/L* [1]. These events are typically called *simple events (or lower-level events)* as they require only a single sensor reading to evaluate whether to notify the end-user of this event [37].

With advances in WSN technology, it has now become possible to have different types of sensor nodes within a region to monitor the environment. This provides the flexibility to monitor an environment in a more extensive manner than before. However, as a result, the end-users may be now interested in the occurrence of *complex events (or higher-level events)*. A complex event is a combination of simple events that occur within a specified period [37]. Thus the processing of complex events requires events to be integrated from multiple heterogeneous sensor events [6, 43, 69, 95]. For example, in monitoring lake water quality, the users may be interested in being notified about an event of *nutrient pollution* rather than normal phosphorus or nitrogen values at certain intervals. In such a case, an event of nutrient pollution could be interpreted only by integrating data from the sensors indicating excessive phosphorus compared to the nitrogen values [74].

Traditionally, processing a query from a user for such an event first requires the user to translate it into a lower-level query; nutrient pollution may be translated into the condition where *the ratio of nitrogen to phosphorus is less than 10 (i.e.,* $N/P < 10$*)* [74]. After translation, the lower-level query is sent to the gateway nodes. The gateway nodes then monitor the sensor data obtained from the appropriate sensor nodes. The results obtained from the sensors measuring phosphorus and nitrogen values are merged and processed by the gateway node to identify nutrient pollution. When all the conditions are met the gateway node notifies the end-users of this event [69, 95]. That is, in traditional processing, the sensor nodes communicate all sensor data to the gateway nodes and computation is done at the gateway nodes.

1.1 Motivating Example

This project uses examples from the area of environmental monitoring, such as earthquake monitoring, habitat monitoring, and water/wastewater monitoring. The most important characteristics of these applications are heterogeneous networks in often hard-to-reach areas, and the desired longevity of the network [38]. Additionally,

these applications require events to be filtered in real time to be able to identify and react to critical changes in the environment in a timely manner [36].

The motivational example focuses on a water quality management system. Water quality is a measure of the fitness of water based on several characteristics. Traditionally, samples are collected manually from the water supplies and then water characteristics are analysed in a laboratory. These characteristics are compared to numeric standards and guidelines to decide if the water is suitable for a particular use [11]. This often causes considerable delays in the monitoring process.

Therefore, the example application domain provides a water quality monitoring network through wireless sensor networks, in which various sensors are used to detect pollutants in the water. In addition to the sensors needed to measure water pollutants, the application needs to monitor weather conditions to study the effects of weather patterns on water quality. Gathering information from both these sensor types then allows for well-directed management of water supplies [89]. Typical events a user could be interested in from such a network of sensors include reports of:

- dissolved oxygen (DO) and water pH values during high algal productivity;
- high nutrient conditions based on phosphorus and nitrogen; and
- non-point source pollution (NPS) from turbidity and high precipitation events.

A typical setup of regulations is described by the following pattern: Government authorities receive regulation data and constraints from reputable agencies, such as the Environmental Protection Agency (EPA) and other state agencies. These regulations are then sent to the sensor network in the form of rules, where the sensors gather data and perform analyses on the gathered data. The monitoring operations identify where and when pollutant levels violate the given guidelines. Once a violation is detected, the sensor network sends information about the identified pollutants and references to the polluted water sources to the monitoring agency.

This example is developed throughout the thesis to explain the proposed approach and evaluate its performance.

1.2 Problem Statement

Wireless Sensor Networks (WSNs) are often employed in remote and hostile environments [59, 62, 91]. They consist of sensor nodes, each of which has only limited energy supply [45]. Therefore minimizing energy consumption is of critical importance. Of all the operations of the WSN sensor nodes (i.e. sensing, processing and communication), communication is the dominant energy consumer [35]. Thus, reducing network communication could prolong the network lifetime.

Reduction in communication can be achieved by processing and filtering data at the sensor nodes such that only relevant data will be forwarded within the sensor network [19]. Traditional WSNs support sensor nodes of one type only – they are homogeneous sensor networks. Recent developments have led to the availability of heterogeneous WSNs supporting a variety of sensor node types [64].

Most existing methods for reducing energy consumption (e.g., [57]) consider homogeneous sensor networks only and do not take into account WSN heterogeneity. Approaches that allow for heterogeneous sensor nodes (e.g., [95]) send all sensor data to the centralized gateway nodes, where complex event processing is performed. This includes the sending of irrelevant sensor data to the gateway nodes, leading to a shorter network lifetime.

Complex event processing at sensor nodes is not easily integrated in heterogeneous networks, because sensor nodes have no knowledge of event types beyond their own. Energy consumption is not homogeneous throughout the WSN. Imbalanced energy depletion of sensor nodes may make established communication paths unstable. A number of solutions address this possible imbalance of energy levels by adapting the routing paths between sensor and gateway nodes according to the energy depletion of the affected sensor nodes (e.g., [42]). No work apparently exists that combines

energy awareness and adaptation in routing with complex event processing at the sensor node level.

1.3 Research Objective & Hypothesis

The main objective of this thesis is to:

> *Design an architecture that takes into account the processing capabilities of the sensor nodes and utilizes it for energy conservation in a heterogeneous Wireless Sensor Network (WSN).*

The idea is to embed data processing capabilities within the sensor nodes in order to reduce the amount of data that needs to be transmitted, and dependence upon gateway nodes for processing. This thesis proposes to perform data integration for complex event detection at the sensor node level for maximum energy benefits. Due to the heterogeneity of the network, semantic information may need to be encoded in each node to support complex event processing. Moreover, each node now needs to be aware of the neighbouring sensor nodes (i.e. information of sensor nodes in its own vicinity) to make routing decisions. Deployment of such semantic context-aware in-network processing in heterogeneous WSNs has the potential to reduce the energy consumption for event communication [84]. The central hypothesis is therefore:

> *If the sensor nodes: i) semantically annotate the sensed data, ii) collaborate with the surrounding sensor nodes, and iii) perform filtering and integration of events on the gathered knowledge, then they will be able to detect complex events locally in heterogeneous sensor networks, thereby resulting in reduced overall energy consumption of the WSN.*

Thus, the working hypothesis for the remainder of this thesis is that inclusion of these three factors at the sensor node level is beneficial for reduced energy consumption in a heterogeneous wireless sensor network.

1.4 Research Questions

Sensor nodes have storage and computational limitations. Moreover, sensor nodes rely on their own very limited energy resources [45], which necessitate their efficient use. To achieve the aim of developing an architecture design that performs complex event detection at the sensor nodes for energy benefits, three facets are examined.

> The first question was *"what are the factors that need to be considered for complex event detection in an energy efficient Wireless Sensor Network (WSN)?"*
>
> Secondly, it was necessary to determine *"how could a complex event detection task be performed in a resource-constrained sensor node?"*
>
> Finally, an exploration of *"how much energy benefit could be gained by performing complex event detection tasks at the sensor node level?"* was required.

Each question is discussed in more detail below.

1.4.1 What are the factors that need to be considered for complex event detection in an energy efficient Wireless Sensor Network (WSN)?

This question can be answered by describing existing solutions whose goal is to perform complex event detection in WSNs. The current state of WSNs was reviewed to identify the factors that are needed for an energy efficient WSN (see Chapter 2 for details). The need for complex event detection arises from the constraints of wireless sensor networks, i.e., energy constraints, heterogeneity and unpredictability. This served to identify factors that are important in performing complex event detection in heterogeneous WSNs. These factors are: (i) filtering and integrating the events at the sensor node level for energy benefits, (ii) a need for sensor data to be enriched

with semantic information to make it understandable by heterogeneous sensor nodes, and (iii) adapting or deriving important information from the gathered context for sharing the information within the sensor network.

The research presented in this thesis also examined research solutions provided by various researchers who aimed to solve these problems (see Chapter 3 for details). The review of the existing approaches identified the research gap that we address in this thesis.

1.4.2 How could complex event detection tasks be performed in a resource-constrained sensor node?

To design an architecture that takes into account the above-mentioned factors for complex event detection, an architecture had to be designed that would also fit within the storage and computational constraints of WSNs.

Following the factors identified in the first research question, a new system was developed that combined the strengths of existing approaches while avoiding their weaknesses. The system is described on two levels: at the conceptual level (see Chapter 4 for details) and at an operational level in the form of a selective implementation (see Chapter 5 for details), the SEPSen.

The conceptual design incorporates the factors identified in relation to the first research question. It focuses on support for semantic annotation of heterogeneous sensor data, reasoning at the sensor nodes for filtering and integration of data and context-awareness for sharing the gathered information in an energy efficient manner. These characteristics allow a lower-end sensor node to make decisions locally and efficiently.

The implemented SEPSen architecture also incorporates the identified factors and aims to perform the processing tasks at the sensor node level, the area in which the gap in existing research was identified. The implementation supports semantic annotation through the use of an ontology, filtering and integration of data through a

reasoner that fits within the resource-constrained sensor nodes and data and energy context-awareness at the sensor nodes for sharing the events in the sensor network.

1.4.3 How much energy benefit could be gained by performing complex event detection tasks at the sensor node level?

Any architecture put forward as an answer to the previous question needs to be evaluated to determine whether it actually meets the objective. This thesis examined the processing and communication costs of performing complex event detection. The processing cost was derived from semantic annotation, filtering and routing table lookup, while communication cost was derived from the number of transmissions required while receiving and transmitting events for the detection of complex events.

This thesis provides a theoretical cost-analysis for complex event detection within the network (see Section 4.3 of Chapter 4 for details). The energy costs for complex event detection in heterogeneous environments were analysed. The costs of a centralized approach for complex events were compared with the fully distributed approach of semantic context-aware in-network processing. A tradeoff was identified between lower communication cost and higher processing cost due to semantic annotation, context analysis for routing and processing of complex events.

The overall results were also confirmed in a simulation-based environment (see Chapter 6 for details). The energy saved in SEPSen reaches almost 50% for the sensor nodes and the overall sensor network lifetime was improved by factor of two against the traditional centralized architecture.

1.5 Structure of the Thesis

This thesis is organized into seven chapters. Figure 1.1 shows how the chapters address each of the questions from Section 1.4.

Chapter 2 helps to answer the first research question about the requirements for an energy efficient WSN. An overview of WSNs is also presented to provide

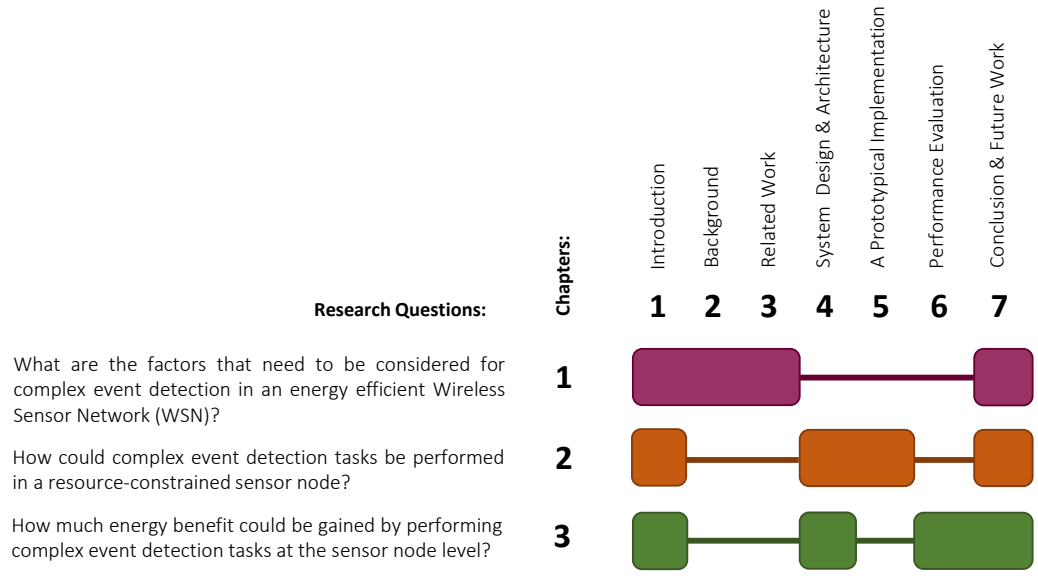


Figure 1.1: Overview of coverage of research questions in the chapters of the thesis.

the background necessary for general understanding of the issues discussed in the remainder of this thesis.

Chapter 3 reviews existing approaches and their limitations to identify the research gap that the second research question aims to answer. It discusses the shortcomings of traditional wireless sensor networks and presents critical issues that must be addressed in order to have an energy-efficient WSN.

Chapter 4 and Chapter 5 complete the answer to the second research question: how can we perform complex event detection tasks at the sensor node level. Chapter 4 proposes a conceptual design of the architecture. It discusses in detail the different components of the architecture and how they interact with each other. Chapter 5 introduces the SEPSen, an implementation of those aspects of the conceptual design that relate to in-network complex event detection in a heterogeneous WSN.

Chapter 6 addresses the third research question: how much energy benefit could be gained by performing complex event detection tasks at the sensor node level. It provides performance evaluation of the proposed architecture introduced in Chapters 4 and 5. The evaluation is done in a simulation environment on PowerTOSSIM [77].

Finally, Chapter 7 summarises the work presented in this thesis including its contributions. It discusses the implications of the findings of the work presented in this thesis and also points out the opportunities for future work.

Chapter 2

Background

This chapter contributes to answering the first research question: *"What are the factors that need to be considered for complex event detection in an energy efficient Wireless Sensor Network (WSN)?"* by reviewing the current state of WSNs and presenting factors that are essential for an energy-efficient WSN.

This chapter is structured as follows: Sections 2.1 and 2.2 provide an overview of the wireless sensor network and its applications respectively. In Section 2.3, we examine the technical challenges for an efficient WSN. In Section 2.4, initial research solutions are identified to address technical challenges described earlier in Section 2.3. The functional requirements for an energy-efficient heterogeneous WSN are presented in Section 2.5. The chapter concludes with a summary in Section 2.6.

2.1 Wireless Sensor Network (WSN)

Recent advances in hardware technology have allowed the integration of sensing, data processing, and wireless communication capabilities into a small, inexpensive, battery-powered device called a *wireless sensor node (or simply sensor node or mote)* [82]. The Mica2 mote [2], shown in Figure 2.1, is a member of the family of UC Berkeley motes that has been used in a number of scientific and commercial applications [82, 91, 53].

Figure 2.1: Mica2 Sensor Node. From "Mica2 Datasheet" by Crossbow Technology Inc., 2004, Crossbow.

Typically, a sensor node consists of sensing, computing, and communication components. Depending on their sensing components, sensor nodes can be used to monitor phenomena such as temperature, light, humidity and other environmental features. The processing module of the sensor node is able to do computation on the sensed values and also on other received values from its neighbours. The communication module in a sensor node is used to send and receive information from neighbouring nodes [9]. Since a single sensor node provides only limited information, a network of these sensors is used to manage large environments [90].

In most settings, tens to thousands of such sensor nodes are distributed throughout a region of interest, where they self-organize into a network through wireless communication and collaborate with each other to accomplish the assigned tasks [8, 82]. Such a network is called a *Wireless Sensor Network*.

Figure 2.2 shows a Wireless Sensor Network (WSN) comprised of several sensor nodes and a gateway node. The sensor nodes continually sense data from the environment and send them to the gateway node. The gateway node receives all the information from these sensor nodes, processes it and sends it to the end-user [30].

2.2 WSN Applications

The integration of sensing, processing and communication components into a small, battery-powered sensor node opens the door to a wide range of real-world appli-

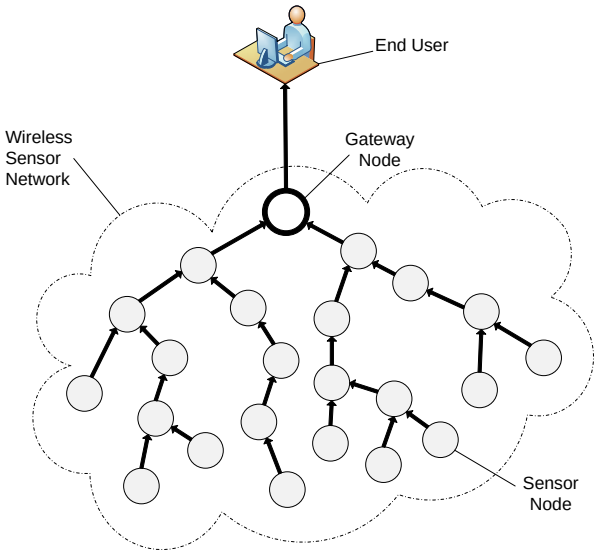


Figure 2.2: Wireless Sensor Network. Adapted from "A survey on sensor networks", by I.F. Akyildiz, Weilian Su, Y. Sankarasubramaniam, and E. Cayirci, 2002, IEEE Communications Magazine, 40(8), p. 103. Copyright 2002 by IEEE.

cations [53]. There is no single set of design requirements that fulfils the entire diversified range of wireless sensor network applications. Therefore, it is critical to explore the application-specific characteristics and requirements before designing a WSN application. This section presents an overview of some of the major application areas and their requirements, which are then summarised in Table 2.1.

2.2.1 Environmental Monitoring

WSN applications for environmental monitoring and control have numerous potential benefits for scientific communities and for society as a whole [16, 10]. Often a large number of sensors are required to gather the data in an environment in order to monitor and control environmental trends and inter-dependencies within various habitats [35]. These applications can involve both indoor and outdoor environments which may consist of huge monitoring areas (i.e. hundreds to thousands of square kilometres) and may also require long periods (i.e. months or years) of monitoring in order to examine long-term and seasonal trends. This may also require large numbers

(i.e tens to thousands) of sensor nodes to be deployed over the desired area in order to collect detailed and meaningful information about the environment [19].

An example of a WSN environmental monitoring application is GlacsWeb [62], which aims to monitor glaciers for long periods (i.e. months or years) in order to understand what is happening under the glaciers and the possible effects on climate change and global warming. Burrel et al. [15] deployed WSN to monitor vineyards in order to prevent freezing damage to crops. Werner et al. [91] used seismic and acoustic sensor nodes to monitor active volcanos in Ecuador. Likewise, SmartCoast [68] was developed to monitor water quality based on the EU Water Framework Directive (WFD).

Such applications often require the system to sense and respond to changes in the environment; therefore, sensor nodes perform data collection by continuously sensing and transmitting the data back to gateway nodes for further analysis. Typically, these applications require low sampling rates, since the most common factors to be monitored, such as temperature, humidity and light, do not change quickly. However, these applications need to operate for long periods in unattended areas. Extended network lifetime is thus the most important requirement of environmental monitoring applications, while data transmission rates can be delayed or reduced in order to improve overall network lifetime.

2.2.2 Surveillance

One of the key advantages of wireless sensor networks is their ability to track and detect patterns in their surroundings, which makes WSNs an integral part of surveillance applications. Military applications are one of the major areas where the fast deployment and self-organizing capabilities of WSNs make them a popular choice to perform battlefield surveillance. In addition to this, WSNs can also be deployed to monitor buildings, airports, shops and homes by promptly sensing and reporting the detected information back to the gateway nodes.

VigilNet [32] is an example of a surveillance application where sensor nodes with magnetic capabilities are deployed over an area of interest to detect magnetic fields generated by the movement of vehicles or other metallic objects. The "A line in the Sand" [12] project by Ohio State University is another similar surveillance application. Like VigilNet, the latter also focuses on the detection and tracking of metallic objects such as armed soldiers and vehicles via a network of nodes deployed over the surveillance area.

Unlike environmental monitoring applications, surveillance applications must immediately report the sensed data back to the gateway nodes in a reliable and timely fashion. For such applications, it is important to protect the data from unauthorized data manipulation. It is also crucial for this category of applications to locate and track selected moving objects within the WSN. Reliable object detection within time bounded latency is an integral part of surveillance applications, which is performed at the cost of high energy consumption. Thus, a major portion of the available energy is consumed by each node communicating its status to neighbouring nodes, whereas the actual transmission of data consumes only a small portion of the available energy.

2.2.3 Health Care

Advances in bio-medical, tele-monitoring and tracking devices have made it possible to use WSNs in a variety of health-care applications. The integration of wireless sensors with health-care applications is highly convenient and beneficial not only for doctors, but also for patients and disabled people. WSNs can be used to monitor patients' movements with the help of tracking devices, while reporting these movements back to the relevant authorities. Elder care is another interesting application, where sensors can send automatic notification to a contact centre in the case of any emergency situations.

The CodeBlue [60] project developed at Harvard University is a good example in the area of health-care applications. CodeBlue continuously monitors patients based

on data gathered from wearable sensors. This involves the monitoring of patients' blood pressure, heart rate, muscular activities and physical movement. The data gathered is continuously transmitted to a Personal Digital Assistant (PDA) device monitored by medical personnel. Other health care applications of WSNs include tracking and monitoring doctors and patients inside hospitals [76], and overseeing drug administration [78].

Similar to surveillance applications, health-care applications also require high accuracy to track the location of patients and medical personnel, as well as returning data to medical personnel in a timely fashion. Moreover, privacy of patients' data is of utmost importance in health care applications. Thus, special consideration of authentication and authorization is undertaken to prevent unauthorized disclosure of information. However, the longevity of the sensor network is not as critical as in environmental applications, since most of the wearable sensor node batteries can be replaced from time to time by patients and/or medical staff.

2.2.4 Other Applications

WSNs can be used in many other areas, including disaster prevention and relief, warehouse inventory tracking, and traffic monitoring [8]. However, these applications' characteristics and requirements are not explained here because the majority of these applications fall into the general category of one of the already mentioned applications (see Table 2.1).

2.3 Constraints of WSN

WSNs are inherently constrained due to the limited resources of the sensor nodes [33]. This section discusses the limitations that complicate the design of WSNs specifically for environmental monitoring applications. It is important to understand the constrained capabilities of sensor nodes in order to address the challenges faced by the WSNs.

Table 2.1: Summary of WSN applications and their requirements

Application	Energy Constraint	Heterogeneity	Adaptability	Real-time operations	Privacy
Environmental Monitoring [62, 15, 68]	High	High	High	Low	Low-Medium
Surveillance [32, 12]	Medium-High	High	High	High	High
Healthcare [60, 76, 94]	Low-Medium	Medium-High	Medium-High	Medium-High	High
Habitat Monitoring [59, 33]	Medium-High	Medium-High	High	High	Medium-High
Disaster Management [92, 61]	Low-Medium	Medium	Medium	High	Low-Medium
Transport and Asset Tracking [28, 80]	Low-Medium	Medium	Medium	Medium-High	Medium-High
Facility Management [18, 17]	Low-Medium	Low-Medium	Low-Medium	High	Medium

2.3.1 Energy Constraints

Energy consumption in sensor networks is a significant problem [45]. Energy constraints in the sensor nodes are due to the power restrictions and limited radio range of the sensor nodes.

The power restrictions of sensor nodes are due to their small physical size. Sensor nodes are typically battery-driven. Moreover, as sensor networks are deployed in remote or hostile environments in most cases, it is difficult to replace or recharge the batteries of the sensor nodes [59, 62, 91].

Sensor nodes consume power for various operations such as running the sensors, processing the information and data communication (see Table 2.2). Communication between sensor nodes consumes most of the power compared to sensing and computation. In fact, each bit transmitted costs as much energy as about processing 800 – 1000 instructions [34].

Moreover, the wireless sensor nodes do not have enough power or communication range to send messages directly to the gateway nodes [63]. In most of the settings, a large number of densely populated sensor nodes continually transmit sensed data back to a set of gateway nodes in a multi-hop manner to extend the coverage of the

Table 2.2: Power model for the Mica2. From "Simulating the power consumption of large-scale sensor network" by Victor Shnayder, Mark Hempstead, Bor-rong Chen, Geoff Werner Allen, and Matt Welsh, 2004, SenSys '04, p. 191. Copyright 2004 by ACM.

Mode	**Current (mA)**
Receive	7.00
Transmit with max power	21.50
CPU (active)	8.00
CPU (idle)	3.20
Sensor board	0.70
LEDs	2.20

network well beyond the the limited radio range of the wireless links connecting the sensor nodes [93, 13]. Although, for a given radio range, multi-hopping increases the network coverage, it decreases the network lifetime. This is due to the increased power consumption at the sensor nodes to relay other sensors' data within the network. When a sensor node runs out of energy, its coverage is lost and the network may fail to carry out the assigned task due to an insufficient number of active nodes [83].

2.3.2 Heterogeneity

The type of sensors that are contained in the sensor network determine whether a network is heterogeneous or homogeneous. If all sensors measure the same phenomenon we call the network homogeneous [64]. If the network is capable of measuring different phenomena we call the network heterogeneous.

The proliferation of WSNs has given rise to monitoring facilities including heterogeneous sensor networks and the ability to extract information from disparate sensor nodes in a meaningful manner to solve real-world problems. On the one hand, this growth has led to monitoring environments in a more extensive and holistic manner [31]. On the other hand, the heterogeneity of the sensor nodes makes finding, extracting and aggregating data at the sensor nodes much harder.

As described in [75], heterogeneity can occur at different levels: system, structure,

syntax and semantics. System heterogeneity is caused by differences in hardware and software components and thus having a set of sensor nodes with different capabilities and functions. For example, a node might have more memory and a more powerful microprocessor than the other nodes in the network. Structural heterogeneity chiefly refers to the fact that different WSNs may employ different storage structures and data models. Syntactic heterogeneity corresponds to differences in data representation and formats, and semantic heterogeneity mainly refers to the fact that the same concept may have different meanings in different WSNs [43].

2.3.3 Unpredictability

Sensor Networks may be very unpredictable in their operations. New nodes can join the network and others may be damaged or run out of power. For example, new sensors could be added to a network which could measure different phenomena from the existing sensors in the network. Other sensors might be displaced due to environmental events, such as flooding in water reservoirs. Moreover, connectivity between the sensor nodes and the routing structures changes dynamically [81]. Thus, the network needs to adapt to changing conditions and requirements in order to remain operable [5, 42].

2.4 Techniques to Mitigate WSN Constraints

This section briefly describes the research techniques used to solve the problems faced by WSNs in terms of energy consumption, heterogeneity and unpredictability. This serves as a criterion for energy efficient heterogeneous WSNs.

2.4.1 In-Network Data Processing

Recent WSN research focuses on the use of in-network data processing to reduce energy consumption by minimizing data volume locally. In-network data processing, shown in Figure 2.3, assumes that only a minimal amount of data is transferred

within the network [19, 57]. Depending upon the application requirements, several processing tasks can be performed in-network such as filtration, aggregation and integration of data, to reduce transmission within the sensor network [79].

For many applications of sensor networks, the end users are not interested in the raw data; instead they are only interested in specific events like *when it is freezing* or *when it is too dark* [57]. In-network filtering reduces transmission of insignificant events by restricting the source to transmitting a message only if there is an event to report.

Data aggregation reduces further propagation by merging data from multiple homogeneous sensor nodes at intermediate nodes and transmitting the aggregated data to the gateways [79, 49]. During aggregation, summarization functions such as minimum, maximum, or average are applied to reduce the volume of data [58]. Aggregating data in this manner helps to remove redundant data and improves the reliability of the information gathered from multiple sensor nodes.

Data integration is performed to detect spatially distributed events by fusing data from multiple heterogeneous sensor nodes. It is similar to filtering of data in that it avoids the transmission of irrelevant sensor data. However, in contrast to simple

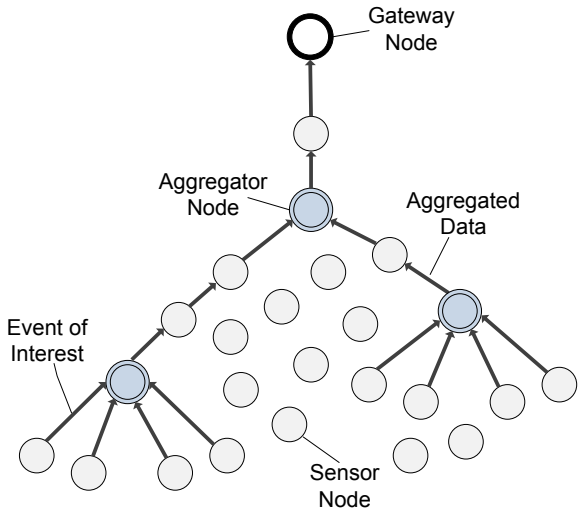


Figure 2.3: In-Network Data Processing in WSNs. Adapted from "Distributed Data Aggregation Scheduling in Wireless Sensor Networks", by Yu, Bo, Jianzhong Li, and Yingshu Li, 2009, INFOCOM'09, p. 2161. Copyright 2009 by IEEE.

filtering, data integration relies on the readings from multiple sensors and their integration to detect higher-level events of interest [52, 65]. Data integration, when performed within the network, reduces overall communication by avoiding the transmission of irrelevant sensor data.

Mostly, aggregation and integration of data is performed at the intermediate sensor nodes [79]; whereas filtering is done at every level, i.e. at intermediate and source node level of the WSN [19]. In this manner, in-network data processing avoids wasting energy on sending large amounts of raw data and significantly reduces energy consumption [49, 57].

2.4.2 Semantic Data Integration

The basic purpose of WSNs is to monitor the environment in which they are placed. They detect events of interest and report them to the user. In addition to simple events, the monitored events may also consist of complex events (or higher-level events) requiring the integration of events from multiple heterogeneous data sources. However, the detection of complex events requires the sensor nodes to be able to understand the received data, perform logical reasoning and integrate such data. Thus, there is a need to attach semantic information to the sensor data [66]. Many researchers [6, 69, 43, 51] have addressed the problem of integrating data from heterogeneous data sources (known as semantic data integration), by using ontologies.

Ontologies provide formal specifications of the terms used in a domain, where relationships between these terms are explicitly defined [67]. They serve to identify the meaning and context of every data element in the sensor network, thereby enabling a semantic-based classification of data throughout the network. Providing semantics to data sources improves the collaboration between heterogeneous data sources and facilitates the integration and exchange of various sensory data among these nodes in WSNs [21]. It thus allows the deployment of different types of sensor nodes in an environment to collect information.

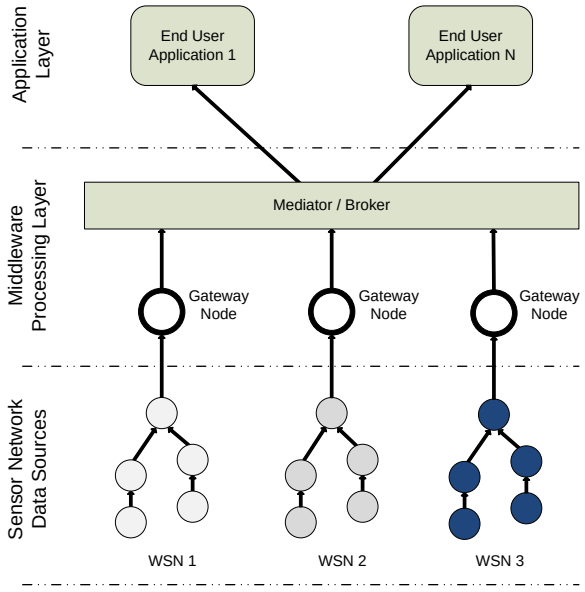


Figure 2.4: Semantic Data Integration from heterogeneous data sources. Adapted from "Semantic sensor information description and processing", by V. Huang and M.K. Javed, 2008, SENSORCOMM '08, p. 458. Copyright 2008 by IEEE.

Various middleware systems, such as SWASN [43] and S-ToPSS [69], offer methods of providing semantics to allow exchange of information between heterogeneous sources. Ontologies have been used to describe and exchange data, thus allowing sensor networks to understand data from different types of sources.

Figure 2.4 shows a general architecture using a layered approach for integrating data from heterogeneous sensor nodes. The sensor nodes may all belong to a single sensor network or may be from different WSNs. As can be seen from Figure 2.4, the bottom level may consist of different WSNs or a single sensor network comprised of different sensor nodes. These nodes send their data to their respective gateways. Gateway nodes perform translation, filtering and aggregation on the gathered data and send results to mediators/brokers. Appropriate merging operations are then performed at a centralized location in the middleware processing layer, and processed information is finally delivered to the application layer for user consumption.

2.4.3 Context-Awareness

Context refers to any information that describes the identity, location, time and activity of an entity. It tells us about the facts surrounding an entity, where the entity may be a person, place or an object [4]. The use of contextual information to determine user needs and provide relevant information and/or services to users makes a system context-aware [5].

Typically, context-aware systems are human-centric and context is applied at the application-level (to adapt or to derive important information from the gathered context). Location-aware information delivery systems and augmented memory systems are examples of human-centric, context-aware systems.

However, in this case, interest is centred on context-awareness including sensor nodes and its application at the sensor node level. This makes it node-centric [56]. Node-centric context-awareness requires nodes to perform different actions, such as altering the event-reporting frequency of a sensor node or the routing path for communication in the sensor network, depending on the available context. It enables a node to adapt its behaviour automatically without requiring instructions from the gateway nodes or the end-user. According to [56], if each sensor node in a sensor network is context-aware then the whole network is context-aware.

Figure 2.5 shows context-awareness in the sensor nodes for energy efficient routing in WSNs. In this case, each sensor is aware of its remaining battery power. They are also aware of other sensor nodes in its vicinity. The sensor nodes adjust their reporting frequency of events based on their available energy and when a system has to send data, it chooses those nodes which have the most energy remaining. In this manner, the energy context-awareness in the sensor nodes prolongs their lifetime and hence that of the sensor network.

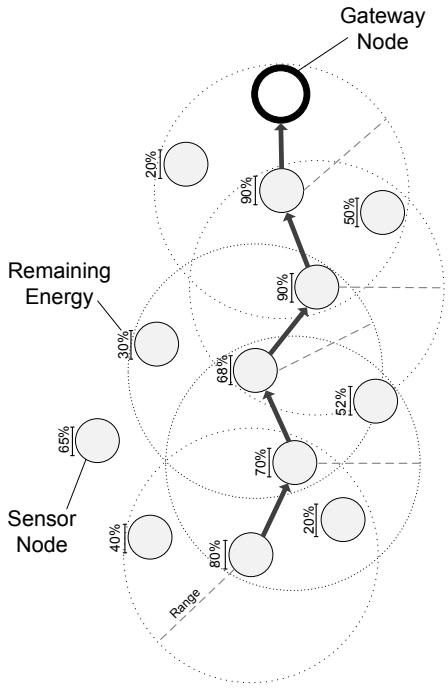


Figure 2.5: Adaptive routing based on context-awareness in WSNs. Adapted from "A Survey on Temperature-Aware Routing Protocols in Wireless Body Sensor Networks", by Christian Henry Wijaya Oey and Sangman Moh, 2013, Sensors, 13, p. 9868. Copyright 2013 by Sensors.

2.5 Functional Requirements

This section provides the factors that are necessary for the detection of complex events within a heterogeneous WSN in an energy efficient manner. The constraints of WSN in environmental applications are presented in Section 2.3. The most relevant constraints of WSNs in environmental applications that need to be addressed are those of energy, heterogeneity and unpredictability. Reduced energy consumption is necessary for long-term operation, while the sensor network should be able to accommodate a wide variety of heterogeneous sensor nodes to monitor various environmental phenomena (e.g., temperature, humidity and light). Moreover, due to remote deployment of the network in harsh or hard-to-reach environments, adaptability is required to manage unpredictable network conditions in any given environment.

Table 2.3: Summary of application requirements, WSN constraints and research solutions

Application Requirement	**WSN Constraint**	**Research Solution**	**Description**
Longer lifetime	Energy constraint	In-network data processing	Many applications of sensor network require network operation for longer periods of time. Sensor nodes have limited energy resources. As radio communication between sensor nodes is energy-expensive, it may not be feasible to transmit large amounts of uninteresting or repetitive data across the network. Various in-network data processing techniques are employed to reduce the transmission and hence the energy consumption.
Event detection	Heterogeneity	Semantic data integration	Unlike other networks, such as simple and ad hoc networks, the goal in WSNs is detection of specific events, not just communication. Integrating data from heterogeneous sources for event detection is a difficult challenge. Semantic technologies such as ontologies can help in collaborative event detection as they allow the data to be shared and processed by the heterogeneous sensor nodes.
Self-organization	Unpredictability	Context-awareness	Given that WSNs are typically placed in remote and hostile environments, the sensor nodes must be able to self-organize. Context-awareness enables the sensor nodes to adapt to changes in the environment and to act accordingly. This ensures the successful operation of the network in unpredictable conditions.

The research solutions mentioned in Section 2.4 present a process of mitigating the constraints of energy, heterogeneity and unpredictability by using in-network data processing, semantic data integration and context-awareness at the sensor nodes in the WSN. We summarize the analysis of application requirements, WSN constraints and research solutions in Table 2.3. The respective solutions to mitigate the constraints of WSNs are thus the functional requirements for the design of an energy-efficient heterogeneous WSN. These requirements are:

(R1) In-network data processing: The processing of data must be done at the sensor node level to limit the communication of unnecessary events and conserve the limited sensor node energy.

(R2) Semantic data integration: The sensor data must be semantically annotated and processed in order to integrate data from heterogeneous sources.

(R3) Context-awareness: The sensor nodes should make their routing decisions based on the available sensor data in the locality and their energy-related context-awareness in order to communicate with other relevant sensor nodes in an energy efficient manner.

Therefore, the hypothesis is that inclusion of these three factors at the sensor node level is beneficial for overall reduced energy consumption in a heterogeneous WSN.

2.6 Summary

This chapter presented an analysis of research in wireless sensor networks that contributes to answering the first research question. The analysis was conducted to derive factors for an approach to achieve the objectives of this thesis, i.e., to *"develop an architecture design that takes into account the processing capabilities of the sensor node and utilize it for energy conservation in WSNs"* (see Section 1.3). The identified factors are in-network data processing, semantic data integration and context-awareness at the sensor node level.

The first part of this chapter provided the background of wireless sensor networks. It discussed a number of potential applications of wireless sensor networks and their requirements. Major applications discussed in this chapter were environmental monitoring, surveillance and health-care applications. Since this thesis focuses on environmental monitoring, the constraints of wireless sensor networks related to environmental monitoring were discussed in detail.

This chapter also examined initial solutions to mitigate network constraints. This provided the functional requirements for the proposed work. Finally, this chapter provided a basis for discussion of the research work in this area.

The next chapter analyses how these recommendations have been realised in existing systems to achieve energy efficiency in a heterogeneous sensor network and its related areas.

Chapter 3

Related Work

This chapter contributes further to the first research question: *"what are the factors that need to be considered for complex event detection in an energy efficient Wireless Sensor Network (WSN)?"* by examining how others have addressed the issues faced by WSNs. It identifies the strengths and weaknesses in existing work and provides a justification for the present study.

This chapter is structured as follows. Section 3.1 explains the focus of this chapter, the scope of the analysis and the criteria used. Sections 3.2, 3.3 and 3.4 then apply these criteria to approaches in their respective areas. Section 3.5 discusses the implications of the research summarized in this chapter for this thesis and for wireless sensor networks in general. The chapter concludes with a summary in Section 3.6.

3.1 Focus

This chapter analyses semantic in-network complex event detection approaches that are related to the objectives of this thesis (see Section 1.3). This section gives more details about the scope of the analysis i.e. which areas are covered, and about the criteria used for the evaluation of related work.

3.1.1 Scope

In-network data processing, semantic data integration and context-awareness have been introduced in the previous chapter. This chapter looks further into specific related work pertaining to the above-mentioned techniques to see how other researchers have covered these areas. Given that we hypothesize (see Section 1.3) that the integration of these three factors is important for an energy-efficient heterogeneous wireless sensor network, related literature is presented wherever possible that includes the integration of these three techniques.

3.1.2 Criteria

The functional requirements for energy-efficient heterogeneous wireless sensor networks described in Section 2.5 are used as criteria for the analysis of work in related areas. These are:

(R1) In-network data processing: Processing of data such as filtering, aggregation or integration can be performed at different levels of the WSN. Existing work is compared on the basis of where the data is processed, i.e. either at the powerful gateway nodes or at the sensor node level.

(R2) Semantic data integration: To process the data from multiple heterogeneous sensor nodes, a mechanism is needed to represent the information in such a way that can be understood by the all the participants. Existing work is examined on the basis of whether it assumes a homogeneous or heterogeneous sensor network i.e. whether it provides a semantic-based classification of data for data processing in the WSN.

(R3) Context-awareness: Sharing the information requires the sensor nodes to make or adapt their decisions based on changing sensor information in the WSN. Existing work is compared based on whether or not it uses context-awareness for sharing of data in an energy efficient manner.

These three requirements represent the choices that must be made in the design of semantic in-network complex event detection in a heterogeneous sensor network. The analysis in this chapter describes what choices were made, implicitly or explicitly, for the analysed systems.

3.2 In-Network Data Processing

Wireless Sensor Networks (WSNs) provide a unique distributed platform that allows users to collect and query events of interest from the sensory data. Different strategies have been proposed for efficient in-network processing of data, such as TinyDB [57, 58], Directed Diffusion [44] and Cougar [96]. In this section, we examine solutions whose purpose is to move the event detection (i.e. users' interest) to the data sources (i.e. sensor nodes) and obtain energy benefits by processing events at the sensor node level.

TinyDB

TinyDB [57, 58] is a distributed query-processing system that allows users to perform event aggregation over streaming sensor data. In practice, the user formulates a query and it is distributed across the sensor network. Along the path of query distribution, a routing tree is formed for sensor nodes to return the results to the user. At each sensor node in the routing tree, the sensor combines its own values with the data received from its children and sends the aggregate to its parent.

An important feature of TinyDB is that it allows the user to express the queries in a simple SQL-like language. In particular, it uses a database view of the sensor network and for this purpose, it maintains a single logical database table (called sensors). The *sensors* table has one row for each node per time instant with one column per attribute that a sensor node can produce. An example of an aggregation query in TinyDB is the following:

```
SELECT AVG(temp)
FROM sensors
WHERE temp > 15
SAMPLE PERIOD 10s
```

This query computes the average temperature of all sensors that sense a temperature value of more than 15° C, and updates the user every 10 seconds.

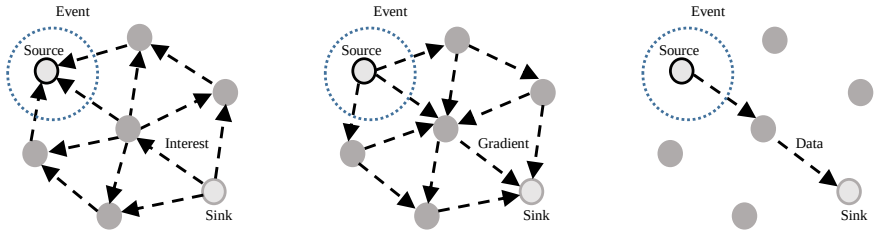
In TinyDB, the dissemination of the query in the network is optimized by maintaining semantic routing trees (SRT). Based on the attribute of the query, the sensor nodes determine whether they have data relevant to the query (i.e. data context). If the attribute of the query applies locally to a sensor node then it produces a result for that particular query. In case it does not apply locally, the sensor node broadcasts it to its children. The parent sensor node then waits for a reply from its children and replies to its own parent when it has heard from all of its children. This process is repeated in the same manner until the replies reach the root of the network. Moreover, if the query does not apply to a sensor node or its child nodes, then those nodes are excluded from reporting the results for a query, thereby resulting in reduced computation and communication costs.

However, TinyDB does not support the integration of data from multiple heterogeneous sensor nodes. It only filters and aggregates data for similar types of sensor data. Thus, user queries based on integrating multiple sensor node data cannot be performed in-network by TinyDB.

Directed Diffusion

Directed Diffusion (DD) [44] is a popular data-filtering and aggregation paradigm in sensor networks (see Figure 3.1). It allows the user to query the sensor network for an event by specifying an interest in attribute-value pairs (called named data). The sink node[1] then broadcasts this interest message to all other nodes in the sensor network. After receiving the interest message, each node re-broadcasts the information to its

[1]The authors use the term *sink node* for the gateway node.



(a) **(b)** **(c)**

Figure 3.1: (a) Interest propagation; (b) Initial gradients setup; and (c) Data delivery along reinforced path in Directed Diffusion. Adapted from "Directed diffusion for wireless sensor networking", by Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin, John Heidemann, and Fabio Silva, 2003, IEEE/ACM Trans. Netw., p. 3. Copyright 2003 by IEEE/ACM.

neighbours and sets up interest gradients for propagating the result back to the sink node. As the gradient set-up phase for a certain interest is complete, only a single path for each source is reinforced and used to route packets towards the sink.

Sensors that hold data matching the interest message send it back to the sink via unicast transmissions. Intermediate nodes that are part of various data paths might also aggregate the data e.g., more accurately pinpointing a pedestrian's location by aggregating reports from multiple sensors. An important feature of directed diffusion is that interest, data propagation and aggregation are determined by localized interactions between the sensor nodes. Moreover, data filtering in DD is based on the suppression of identical data from multiple sources e.g, only an event matched with high confidence levels is sent by the intermediate nodes to the sink node. However, DD does not perform in-network processing of complex events from multiple sources as the task descriptions are based on sampling of a single sensor node for event detection, such as the use of only sampled waveforms for the detection of a pedestrian or an animal in the region.

COUGAR

The Cougar [96] approach is similar to TinyDB in that it performs in-network distributed query processing. In Cougar, the sensor nodes store and process the data by aggregating similar sensor readings. It also eliminates irrelevant readings by sending only the necessary readings to the gateway nodes. Similar to TinyDB, a declarative, user-friendly query language has been developed that allows users to define queries such as *Generate a notification if abnormal temperature is observed by the sensors* or *Return the average temperature on each floor*.

However, unlike TinyDB where each node maintains an SRT, Cougar is based on clustering. In Cougar, the sensor nodes send partially aggregated events only to their selected leader nodes (i.e. cluster heads). When the cluster head collects all the data from its child sensor nodes, it filters and aggregates the sensor readings according to the query plan and sends the results to the gateway nodes. More importantly, aggregation operations such as *average* and *maximum* can be used in the query plan. However, Cougar does not support integration operations which require data to be merged and filtered from heterogeneous sensor nodes.

Evaluation

TinyDB, Directed Diffusion, and Cougar are good at conserving sensor node energy by performing in-network filtering and aggregation of sensor data. However, these approaches are based on homogeneous sensor networks. They do not provide semantic support for different types of sensor data and as a result, do not provide a means to integrate data from heterogeneous sources in order to detect complex events within the sensor network. As in these approaches, cooperation and exchange of data among the sensor nodes is not possible, it hampers the scalability of the sensor networks in cases where data has to be integrated from multiple sensor nodes to answer user queries.

3.3 Semantic Data Integration

In heterogeneous sensor networks, it is a requirement to provide meaning to the sensed data for it to be interpreted and processed by a diverse set of sensor nodes. Moreover, it is important to reason on the gathered events to deduce higher-level events from simple events. This section examines various systems that aim to integrate heterogeneous sensor events in a WSN to detect user-defined complex events.

CPS

Alex Wun et al. [95] propose a Content-based Publish/Subscribe (CPS) technique, that is constructed over the S-ToPSS (Semantic Toronto Publish/Subscribe System) [69], to allow semantic data integration across heterogeneous sensor networks. One of the major roles in a CPS system is that of brokers. The brokers incorporate a *semantic engine* (S-ToPSS) that uses mapping functions to translate raw or aggregated sensor events into semantically equivalent events as shown in Figure 3.2. For example, an event recorded by a sensor node as (light, 250) will be translated to (weather, cloudy) by the broker. These mapping functions are included in ontologies of different domains and can additionally be supplied by the applications. Matching

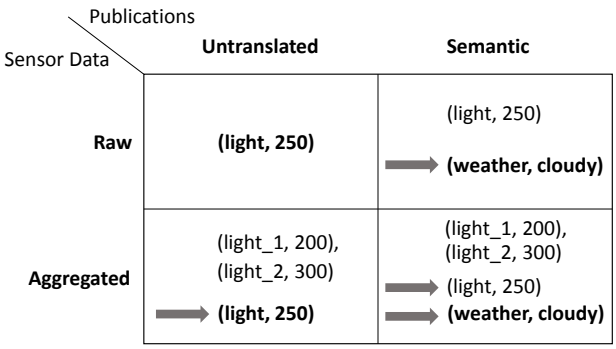


Figure 3.2: Sensor and semantic publications in CPS. From "A system for semantic data fusion in sensor networks", by Alex Wun, Milenko Petrovic, and Hans-Arno Jacobsen, 2007, DEBS '07, p. 77. Copyright 2007 by ACM.

of subscriptions for higher-level or complex events is also performed at the brokers by integrating data from multiple heterogeneous sources.

As CPS allows aggregation of raw sensor data at the sensor node level, a certain level of in-network data processing is achieved. This results in reduced transmission at the sensor nodes by suppressing unnecessary or similar data. However, detection of complex events requiring integration of data from multiple sensor nodes is not performed at the sensor node level. CPS only sends sensor level filtered or aggregated data to the broker overlay network. The brokers then perform semantic matching and integration of events against application interests and publish those application-specific events to the user.

SWASN

In SWASN (Semantic Web Architecture for Sensor Networks) [43], a layered approach to data integration has been proposed. Sensor data is gathered from heterogeneous sensor nodes at the Sensor Network Layer. This data is then processed at the *Ontology Layer* by attaching semantics to it. Further reasoning for the detection of events is done using a rule-based engine at the *Semantic-Web Processing Layer* of the architecture. The processed data is then made available to different client applications that require it through an *Application Layer*. The application Layer basically provides the interaction between the client application and the SWASN middleware through any server over the internet/intranet.

In SWASN, the sensor data is not processed at the sensor node level (i.e. Sensor Network Layer). SWASN therefore does not perform in-network data processing of sensor data as all the sensor data is sent to the gateway nodes. Processing for detection of complex events is then performed at the upper layers in the gateway nodes or the middleware layer (i.e. Semantic-Web Processing Layer). Moreover, SWASN also does not specify the underlying communication model for the sensor nodes in the sensor network. It therefore does not use a context-aware mechanism for managing routes between the sensor nodes and the gateway nodes.

Event Dashboard

In [97], the authors propose an ontology-driven user interface through which users can describe the event constraints on sensor events (e.g. Reporting a high total nitrogen event when Total Nitrogen (TN) is greater than 10). The user-interface also allows users to define conjunctive constraints on the set of sensor data for the detection of complex events, such as the detection of algal bloom events based on the nitrogen and phosphorus sensor events. The event constraints are then translated by a *Semantic Mediator* to be stored in *Global Sensor Network (GSN)* middleware. The GSN middleware receives events from the sensor nodes, matches sensor data to the event constraints and notifies the mediator of it. The mediator then updates the user-interface with the matched event notification.

However, in Event Dashboard, the sensor nodes in the sensor network are decoupled from the other components of a system i.e., the sensor nodes are not aware of user-specified events. Thus, all sensor data, irrespective of its significance, is sent to the middleware, where the processing for user-specific events takes place. Moreover, similar to SWASN [43], the authors do not specify any communication model and the focus of their work is on the processing of user-specific events only.

ACTrESS

In [29], the authors present an architecture (i.e., ACTrESS) that performs automatic context transformation to enable easy event-based integration and development of systems with different semantics. ACTrESS can be added to any existing publish / subscribe middleware. An important aspect of ACTrESS is that it does not rely on a globally accepted base schema. A context repository is maintained which is modifiable at runtime. This allows for transparent context transformation for event producers and consumers; and enables the integration of new event producers and consumers with their respective contexts. However, ACTreSS is designed to be employed in high-end sensor nodes and it also does not provide node-centric context-awareness.

Evaluation

The above systems address the problem of heterogeneity in wireless sensor networks. However, in such systems, sensor nodes are used only for the capturing of data from the environment, while data analysis and interpretation is performed at the more powerful gateways or application levels. This puts a heavy communication load on the sensor nodes for performing their operation in a continuous manner, as all data needs to be forwarded to the gateway nodes regardless of its usefulness.

3.4 Context-aware networks

Wireless Sensor Networks (WSNs) are often deployed in an ad-hoc fashion with no infrastructure support. As described earlier in Section 2.3.1, sensor nodes usually have limited bandwidth and energy resources, which require simple and efficient underlying communication protocols. One of the most fundamental actions that such devices in the sensor networks need to do is to find information about the environment they are operating in. Moreover, to share and use the available context information in the network, sensor nodes first need to discover and locate the required information. This action is called context discovery. With the use of this context information, the sensor nodes can then make intelligent decisions regarding the appropriate usage of their resources. This section presents context-aware solutions that aim for a context-aware sensor network by providing the sensor nodes with a means to adapt to their surroundings.

SPIN

The family of SPIN [50] protocols (SPIN-PP and SPIN-EC) are resource aware and resource adaptive. SPIN-PP protocol aims at diffusing information within the network using information descriptors (i.e. meta data) to reduce the transmission of redundant data in the sensor networks. When a sensor node running SPIN (shown in Figure 3.3) wants to send data, it first advertises (ADV) it to its neighbouring nodes

in the sensor network using the meta-data. If the neighbouring node is interested in the advertised data, it requests (REQ) the source for this kind of data. The source node then responds and sends the data to the sinks (DATA). The sink node, after receiving the DATA sends the advertisement (ADV) for the obtained data further in the network, and the process continues until it reaches the gateway node.

While SPIN-PP uses a negotiation mechanism to reduce energy consumption, sensors running SPIN-EC are resource-aware. They can make informed decisions about the efficient use of their resources. Thus, if a sensor has low residual energy, it controls its participation in the data dissemination process. It does not however take energy-awareness into account in selecting the routes between the sensor nodes. Thus, it partially supports energy-awareness within the sensor network. Moreover, both the protocols allow the sensors to exchange information about their sensed data, thus helping them to obtain data of interest.

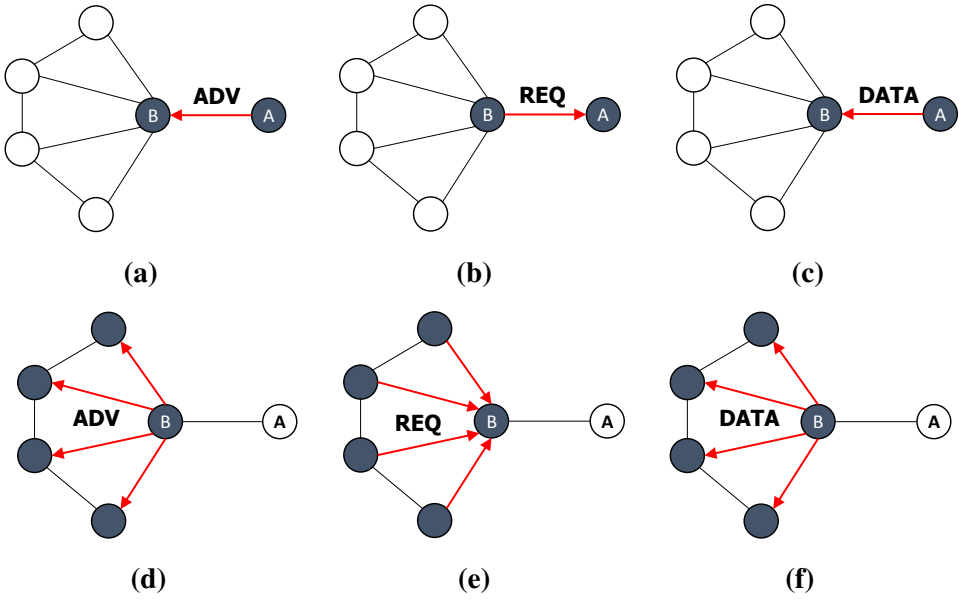


Figure 3.3: SPIN Protocol. (a) Node A advertises its data to node B, (b) Node B sends a request for node A data, (c) Node B receives the requested data, (d) Node B then sends out advertisements to its neighbours, (e-f) Node E and F send requests back to node B and get the data. Adapted from "Negotiation-based protocols for disseminating information in wireless sensor networks", by Joanna Kulik, Wendi Heinzelman, and Hari Balakrishnan, 2002, Wireless Networks, 8, p. 172. Copyright 2002 by Kluwer Academic Publishers.

IS_SDM

The In-network Semantic Sensor Data Model (IS_SDM) [20] targets energy consumption by processing the data at enhanced context-aware sensor nodes. The context input is derived from explicit and implicit context inputs for sufficient context information about the sensor network. Explicit context is formed from the actual readings of the sensor nodes, whereas implicit context is formed from secondary information available to the sensor nodes through their neighbouring nodes in the sensor network. The context input in IS_SDM is based on the defined attribute set for each sensor node. The attribute set of a sensor node (S_i) consists of sensor node readings (R_i), energy required for one sample (E_i), frequency of sampling (F_i), confidence level based on battery voltage (C_i), resolution mask (U_i) and group size (G_i). These attributes enable the sensor nodes to take context into consideration and adapt their operations, such as altering the frequency of sampling when the sensor nodes' energy drops below a certain threshold.

Furthermore, IS_SDM allows filtering of events by imposing a resolution mask on the sensor readings. In this way similar or repeated sensor readings are discarded and only a sudden sharp change in sensor readings is reported. More importantly, IS_SDM allows similar types of sensor nodes to be placed in one group with an additional attribute called group confidence (GC_i). This enables the sensor nodes to report events only when the group confidence is above a certain threshold, such as the reporting of a security problem when noise is detected, with high group confidence, during non-working hours.

However, IS_SDM does not provide a mechanism to integrate events from two or more different groups of sensor nodes. It is thus limited to sensor nodes or groups consisting of similar sensor types, to perform in-network filtering of simple events. Moreover, similar to SPIN-EC [50], IS_SDM uses energy context for altering the sampling frequency of the sensor nodes and does not provide energy-awareness for altering the routes between the sensor nodes and the gateway nodes.

EAR

Energy-aware routing [73] uses energy-awareness in the sensor nodes to alter the paths for communication between the sensor nodes and the gateway nodes. The paths are probabilistically updated and one of several paths is chosen for communication by the sensor nodes in the network. This also includes sub-optimal paths, as using the minimum energy paths all the time may deplete the sensor nodes' energy on a particular path.

The sensor nodes in EAR are addressed based on their type, such as temperature or humidity sensors. However, sensor events are communicated based only on type, such as *"Temperature of Room 5"* rather than on user-specified events such as reporting sensor events *"When the temperature is below 10 in Room 5"*. It therefore does not perform in-network processing of data for filtering and integration purposes. Moreover, in EAR, information about multiple routes (from source to destination) has to be maintained by the sensor nodes to enable changing between different routes to manage sensor nodes' energy resources fairly. This proves to be useful in increasing the lifetime of sensor networks, but it puts a heavy burden on limited sensor storage.

Pub-2-Sub

Pub-2-Sub [85] is a content-based routing mechanism for P2P networks. It has the virtualization and indexing component for maintaining the overlay network structure and determining the subscription and publication paths for given queries, respectively. A unique virtual address is assigned to each of the sensor node. The routing for subscription and publication is then performed based on the virtual addresses of the nodes. Pub-2-Sub+ [86] is an extension to Pub-2-Sub, which is built on a logic overlay without requiring the location information. However, energy-awareness in deciding the routing paths is not considered in PUB-2-SUB+, which can cause high routing overhead at the relay sensor nodes, especially in regions to which many

events and queries are directed such as the sensor nodes nearer to the gateway node. The sensor nodes in this region may deplete their energy quicker and thus the network lifetime is severally affected.

Evaluation

Integrating data within a heterogeneous sensor network requires the sensor nodes to be aware of the available sensor data in the locality (i.e. data context). It also requires the sensor nodes to share the available information by keeping in view the available energy resources (i.e. energy context) in the sensor network. The context-aware networks mentioned above perform different operations by considering either the data context or the energy context only, such as SPIN-PP and SPIN-EC [50]. Moreover, in SPIN-EC [50] and IS_SDM [20], energy context-awareness is used to trigger power saving functionalities in the sensor nodes such as increasing or decreasing event reporting rates based on sensor node energy levels. However, they do not consider the available energy of the different routes as in EAR [73]. EAR [73] uses both data and energy contexts in considerating selection of routes but does not perform any data processing at the sensor node level for event detection.

3.5 Discussion

This section discusses the implications of the approaches analysed in this chapter. The relevant approaches in the surveyed research are viewed along with insights into the effectiveness of the approaches where such insights are available.

Table 3.1 provides a comparison of different approaches to the criteria for the proposed work. It can be seen that the problem with existing methods for in-network data processing is that it considers homogeneous sensor networks and does not perform integration of heterogeneous sensor data at the sensor node level. This limits the potential scale of sensor networks as the sensory data cannot be shared between various sensor nodes and thus only the processing of simple events can be

Table 3.1: Comparison of related work to the criteria.
(+) supported, (−) not supported, (±) partially supported

Approach / Criteria	**R1:** In-network Data Processing	**R2:** Semantic Data Integration	**R3:** Context-Awareness	
			Data-context	Energy-context
TinyDB	+	−	+	−
DD	+	−	+	−
COUGAR	+	−	+	−
CPS	±	+	+	−
SWASN	−	+	−	−
Event Dashboard	−	+	−	−
ACTrESS	−	+	−	−
SPIN	+	−	+	±
IS_SDM	+	−	+	±
EAR	−	−	+	+
Pub-2-Sub+	±	−	+	−

performed at the sensor node level. Moreover, the solutions provided for semantic data integration perform the operations at centralized locations which are normally designated, more powerful gateway nodes. In both cases, the scalability of the overall network is hampered and a large amount of energy is wasted on sending/receiving irrelevant data.

In addition, context-awareness has been used only to conserve energy at the sensor nodes by selecting optimal routes to the destination nodes; whereas others provide a method to request and acquire sensor data in the sensor network by enabling data context-awareness at the sensor nodes. Most importantly, the context-aware networks that take both the data and energy context into account do not combine energy awareness and adaptation in routing with complex event processing at the sensor node level.

3.6 Summary

This chapter focused on recent research work related to wireless sensor networks. It contributed further to answering the first research question (see Section 1.4.1) by examining how others have addressed the issue of complex event detection for an

energy efficient Wireless Sensor Network (WSN). The related work was divided into three main categories: in-network data processing, semantic data integration and context-aware sensor networks.

It showed that most systems which aim to support complex event detection (e.g., [95]) perform the processing at powerful gateway nodes, whereas solutions supporting in-network data processing (e.g., [57, 58]) lack semantic capabilities and hence can be only used for homogeneous sensor networks. Systems that provide context-awareness (e.g., [50]) are again isolated solutions that do not handle processing of events at the sensor node level.

The review of studies into the effectiveness of these approaches showed that no single approach achieves the goal of this thesis and no single approach meets all recommendations made in the previous chapter. The material presented in this chapter forms a basis for the approach of this study, which aims at overcoming some of the limitations of prior research.

The following chapter introduces a new approach for semantic in-network complex event detection that addresses the gaps in the surveyed approaches with regards to the recommendations and builds on the demonstrated strengths of existing approaches.

Chapter 4

System Design & Architecture

This chapter introduces the conceptual design of the SEPSen architecture. The design is based on the results of the previous two chapters, on the recommendations for efficient wireless sensor networks derived from the constraints and strengths of existing wireless sensor networks. This chapter explains how these recommendations are fulfilled, keeping in mind the resource-constrained nature of WSNs, and thus finding an answer to the second research question raised in Section 1.4.3: *"how can complex event detection tasks be performed in a resource-constrained sensor node?"* This chapter also presents an energy-based cost model for processing and communication of events in heterogeneous WSNs. This contributes to answering the third research question, i.e., *"how much energy benefit could be gained by performing complex event detection tasks at the sensor node level?"*

The chapter is structured as follows. Section 4.1 gives a high-level description of the focus employed in the design of the proposed SEPSen architecture. Section 4.2 introduces the conceptual architecture and Section 4.3 provides a cost-analysis of the proposed architecture during various operations. The limitations of the system design are discussed in Section 4.4 and the chapter concludes with a summary in Section 4.5.

Early versions of parts of Sections 4.2 and 4.3 have been previously published in [48] and [47], respectively.

4.1 Design Requirements

The objective of this thesis, as introduced in Section 1.3 is to develop a sensor node architecture that performs processing tasks at its own level. Using terms and concepts common to wireless sensor networks, the central hypothesis of this thesis was stated in Section 1.3 as follows:

> *If the sensor nodes: i) semantically annotate the sensed data; ii) collaborate with the surrounding sensor nodes; and iii) perform filtering and integration of events on the gathered knowledge, then they will be able to detect complex events locally in heterogeneous sensor networks, thereby resulting in reduced energy consumption.*

This thesis puts forward a novel solution that uses semantic annotation and integration of data at the sensor node level. Moreover, since knowledge sharing is a key aspect of data integration in an unpredictable and dynamic sensor network, this architecture employs context-awareness at the sensor node for data integration purposes.

The functional requirements for semantic in-network complex event detection in a heterogeneous sensor network have already been specified in Section 2.5. The first step in achieving the goals of this thesis is to look at general sensor node architecture and its characteristics.

Figure 4.1 shows the general architecture of a sensor node. It consists of: (i) a sensing unit including sensors for data sampling; (ii) a processing unit including a micro-controller to execute applications and manage resources, and provide storage for local data processing; (iii) a communication unit consisting of transceiver for transmitting and receiving the data and coordinating with other sensor nodes; and finally (iv) a power unit that supplies power to all the components [9].

Mica2 motes, made commercially available by Crossbow, are the de facto standard for sensor nodes in industry, government and research [41]. A Mica2 mote has

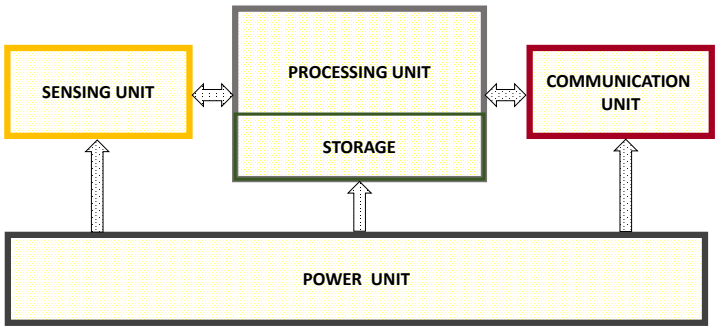


Figure 4.1: General architecture of a Sensor Node. Adapted from "Energy conservation in wireless sensor networks: A survey", by Giuseppe Anastasi, Marco Conti, Mario Di Francesco, and Andrea Passarella, 2009, Ad Hoc Networks, 7, p. 538. Copyright 2009 by Elsevier.

128 KiB of program flash memory, 4 KiB of RAM, and 512 KiB of measurement serial flash memory for data logging [2]. Thus, if processing has to be done at the sensor nodes, the design has to cater for the sensor node's limited storage capacities.

Keeping in mind the issues discussed above regarding sensor nodes' architecture and available storage, functional requirements from Section 2.5 and the goal of integrating heterogeneous data at the sensor node level, the design requirements are:

(D1) a mechanism to support semantic annotation of sensed data for filtering within the sensor node and for enabling other sensor nodes to use it for integrating data within the heterogeneous sensor network.

(D2) a pattern-matching rule engine for processing complex events. As the sensor nodes regularly generate events, the sensor events (i.e. facts) stored in the knowledge base for pattern-matching should be within the limits of sensor nodes' storage capacity.

(D3) a communication model that allows the sensor nodes to share data and energy-related context within the sensor network, so that information can be shared in an energy-efficient manner.

These requirements are necessary to support semantic in-network data integration in a heterogeneous sensor network. Most importantly, the design of these components should fit within the limited sensor node storage. The next section discusses the details of the design and how these requirements are fulfilled in the design of the new architecture.

4.2 Overview of SEPSen

This section introduces the conceptual design of an energy-efficient sensor network that builds on the results of the previous two chapters. Figure 4.2 shows the general architecture of a single SEPSen node[1]. The basic components of the proposed architecture are: Receiver, Semantic Annotator, Knowledge Base, Rule Engine and Transmitter. This relates to the general architecture of the sensor node in the sense that the communication unit is split into receiver and transmitter components. The receiver component deals with the sensed and shared sensor data, while the transmitter deals with the transmission of data to the gateways or other relevant sensor nodes in a context-aware fashion. The processing unit is split into semantic annotator and rule engine components, where it performs the annotation, filtering and integration of the sensor data. The available storage in the processing unit is used for storing the facts and rules in the knowledge base of the sensor nodes. The following sections discuss the individual components of the system architecture in detail.

4.2.1 Receiver

The *Receiver* component distinguishes between three types of incoming events: sensed, shared or forwarded. The distinction between these types of events is necessary as different operations are performed on each of the event types.

[1] The power unit is omitted in the figure as it does not add any SEPSen specific functionality to the overall system.

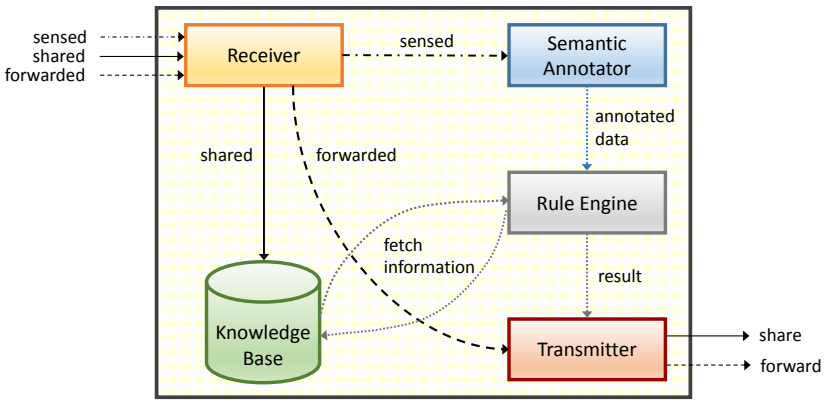


Figure 4.2: SEPSen Node Architecture

The monitored phenomenon is measured as event data at regular intervals. This sensed event is then annotated and compared against the rules in the rule engine. For rules that refer to events that need to be integrated from multiple heterogeneous sensor nodes, the data is shared among the relevant sensor nodes. The node identifies shared events by looking into the destination-ID field of the packets. If the destination-ID belongs to the node itself then it is an event that needs further processing. Moreover, on arrival, the shared event is added to the knowledge base and it then triggers the rule engine for evaluating rules against the shared knowledge base. If the forwarded event is received from other sensor nodes and is to be passed on in a multi-hopping manner to the gateways (routing), no filtering is performed for the forwarded events. This component is also responsible for maintaining routes along with the transmitter component (see details in Section 4.2.5).

4.2.2 Semantic Annotator

There is a need to add an explicit description to measurements: for example, a temperature provided by heterogeneous sensors can be measured in Celsius or Fahrenheit. The *Semantic Annotator* component converts the sensor readings into semantically equivalent descriptions. For this purpose, semantic technologies (i.e. RDF) are integrated with sensor measurements and domain ontologies to convert sensor measurements into semantic measurements.

Resource Description Format (RDF) is based on triples. A triple is like a sentence and consists of subject, predicate and object. For example, in *"Sensor_1 measures 25 degrees Celsius"*, *"Sensor_1"* is the subject, *"measures"* is predicate and *"25 degree Celsius"* the object. We can describe a large number of triples in this manner such as: *the sensor is at location X*, *location X belong to Site A* and *Site A is managed by party P1*.

The semantic annotation of sensed events requires the relevant ontology fragments to be infused into each node in the sensor network. Events are then mapped to the concepts of the infused ontology. For example, if a sensor node monitoring water pH in a lake senses a value of 5.0, it will then annotate it with the relevant domain ontology information, such as:

```
wqo:WaterpHSensor_1 is_a wqo:WaterpHSensor,
            wqo:hasValue "5.0"^^xsd:double.
```

The annotation is based on the ontology fragment available to this sensor node (see Figure 4.3). The ontology fragments are obtained from the main ontology that models the overall application (see details in Section 5.3). The ontology fragmentation can be performed by following the method discussed in [71]. Another option is using Protégé: while running the specified rules on Protégé, it imports the required classes and instances based on the relationships between the concepts of the ontology. These selected classes of the ontology can been used to deploy ontology fragments (i.e., relevant ontology concepts) into the sensor nodes. In our prototype, the ontology fragments are currently manually infused in the sensor nodes. However, the gateway nodes would be responsible for automatically injecting ontology fragments in the sensor nodes. Alternatively, some high-end nodes may also be able to retrieve or pull ontology fragments on the fly and process them for the sensor nodes.

4.2.3 Knowledge Base

The *Knowledge Base* of a sensor contains a facts base and a rules base. Facts are data recorded by the sensor or data received from other sensors. The facts base

Figure 4.3: Ontology fragment for Water pH sensor node

```
@prefix wqo:<http://www.co-ode.org/ontologies/wqo.owl#>
@prefix epa:<http://tw2.tw.rpi.edu/zhengj3/owl/epa.owl#>
@prefix ssn:<http://purl.oclc.org/NET/ssnx/ssn#Sensor#>
@prefix owl:<http://www.w3.org/2002/07/owl#>
@prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>

epa:MeasurementSite rdf:type owl:Class;

epa:WaterMeasurement rdf:type owl:Class;

epa:WaterProperty rdf:type owl:Class;

ssn:Sensor rdf:type owl:Class;

wqo:WaterpH rdf:type owl:Class;
        rdfs:subClassOf epa:WaterProperty.

wqo:WaterpHSensor rdf:type owl:Class;
        rdfs:subClassOf ssn:Sensor.

epa:hasMeasurement rdf:type owl:ObjectProperty;
        rdfs:domain epa:MeasurementSite;
        rdfs:range epa:WaterMeasurement.

epa:hasProperty rdf:type owl:ObjectProperty;
        rdfs:domain epa:WaterMeasurement;
        rdfs:range epa:WaterProperty.

wqo:observedBy rdf:type owl:ObjectProperty;
        rdfs:domain epa:WaterProperty.
        rdfs:range ssn:Sensor.

wqo:hasValue rdf:type owl:DatatypeProperty;
        rdfs:domain ssn:Sensor;
        rdfs:range xsd:double.
```

changes throughout the execution of a program as new data is recorded by sensors or new data is obtained as result of an action triggered by a rule execution. Individual facts are stored in triple form as <Subject, Predicate, Object>. This is the format as provided by the sensors after the semantic annotation of the sensed values. Below is an example of facts stored in the knowledge base for the WaterpH sensor node at a particular time.

```
epa:WaterMeasurement_1
        epa:hasProperty wqo:WaterpH_1.

wqo:WaterpH_1
        wqo:observedBy wqo:WaterpHSensor_1.

wqo:WaterpHSensor_1
        wqo:hasValue "5.0"^^xsd:double.
```

Part of the knowledge base is also a collection of rules, i.e. the rule base, defined by the application. Rules are updated as the application changes its requirements or new requirements arrive. A rule contains both a set of conditions and the actions to be performed when those conditions are met. A rule fires if all its conditions are true. In this project, semantic rules are used in the form of IF-THEN expressions, which is similar to Event-Condition-Action (ECA) rule structure. The IF-statement specifies the conditions to which the rule should apply in order to make the THEN statement valid. Below is an example of a rule that is used to detect a pollutant in water reservoirs if the value of the water pH is less than 7.0.

```
epa:WaterMeasurement(?p) & wqo:hasProperty(?p, ?q)
& wqo:WaterpH(?q) & wqo:observedBy(?q, ?r)
& wqo:WaterpHSensor(?r) & epa:hasValue(?r, ?s)
& lessThan(?s, 7.0) -> epa:NutrientPollution(?q)
```

The knowledge base consumes a major portion of sensors' limited storage resources and it is desirable to minimize its size. This is achieved by only storing the information that is related to a particular sensor node. In this way, irrelevant information is discarded and only the required portion of information is stored on each sensor node. Details of this are provided in the next section.

Duplicate Facts Deletion (DFD)

In many applications, such as environmental monitoring, the data typically changes slowly, perhaps only daily or even over weeks. In contrast, the sensors will produce fresh readings several times per second. Although the impact of such changes is mitigated by the sensor filtering the unwanted data at its layer, it still forwards the sensed events after they have passed the filtering thresholds. These events are forwarded if they are required for complex event detection. In such a case, the reading needs to be stored by the receiving nodes.

The conventional insert-based fact management and rule processing method would result in serious performance degradation in resource-limited sensor devices. The

strategy in this project is to keep the sensor data up-to-date by regulating the number of facts. With such mechanisms, the sensor can maintain adequate but not excessive levels of information.

Thus, to achieve the desired level of performance, facts that have already been sent by a particular sensor node and placed in the memory (i.e. knowledge base) are updated and the previous similar facts are deleted from the memory. The impact of discarding historical values is discussed in Section 4.4.

4.2.4 Rule Engine

Most of the systems discussed earlier in Section 3.3, employ a rule engine for filtering and integrating sensor data in heterogeneous sensor networks. Similarly, external standalone rule engines such as Jess[2] and Drools[3] can also be used to process sensor data for filtering and integration purposes. However, due to the sensor nodes' limited storage and computational capabilities, these engines cannot be employed directly at the sensor nodes but rather are used at the powerful gateway nodes. Thus, a rule engine is needed that can be directly used at the sensor nodes for processing sensor data.

Therefore, a reasoner is used at the sensor nodes as the *Rule Engine* component. The rule engine performs reasoning on the sensed data for data filtering and integration purposes, using the adapted Rete algorithm [27].

For the rule engine, the application submits its requirements in the form of rules. It also specifies the action to be taken when the requirements specified in the rule are met. The rule engine, shown in Figure 4.4, gathers the rules and builds a pattern network of nodes which encodes the condition parts (IF-parts) of the rules. At the bottom of the pattern network are the nodes representing the individual rules (i.e. THEN-parts of the rule). The input consists of changes in the *facts* as new facts are inserted or deleted from the knowledge base.

[2] http://www.jessrules.com/
[3] http://www.jboss.org/drools/

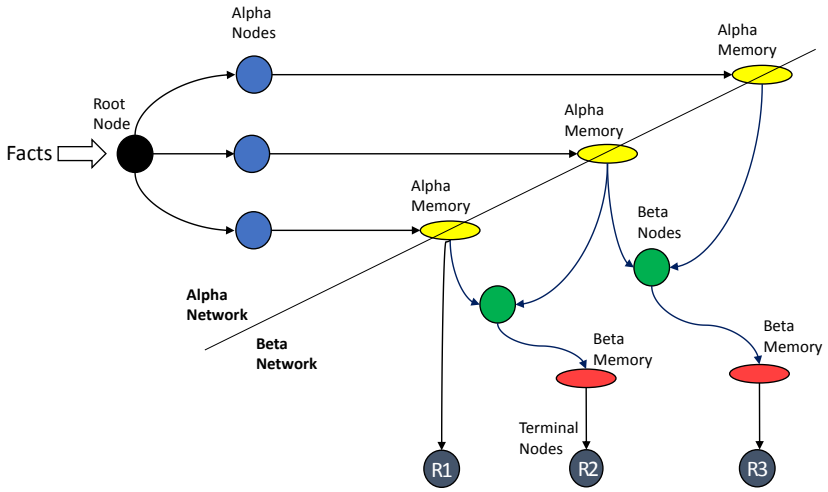


Figure 4.4: Rule engine pattern-matching process. Adapted from Wikimedia Commons, by Charles Young, 2006, Retrieved from http://commons.wikimedia.org/wiki/File:Rete.JPG

The pattern matching starts with the *root node*. For each fact, there is a selection that follows the root node. Each selection is based on the pattern of conditions that are looked for and *alpha nodes* are created which conform to the corresponding selected pattern. The Rete algorithm makes the pattern matching efficient by having memories to remember the matched facts for pattern nodes. Thus with each alpha node, there is an associated *alpha memory*. If there is a rule that requires more than one condition to be checked then a *beta node* is created. Beta nodes are the joiner nodes, which join two alpha node inputs. The beta node has a *beta memory* which is split into left and right memories. Partial facts are stored in each memory after matching and when both the join conditions are successfully evaluated, it passes the joined facts downwards to the *terminal nodes*.

When an input traverses all the way down the network to the terminal nodes, it has passed all the tests on the left hand side of the rules and activation is performed, which produces the corresponding output. The outputs, after matching the rules against the facts, are the applicable rules. The applicable rules then trigger the execution of the THEN-parts and the respective actions are performed.

The different actions specified by the rule engine could be: *discard*, *share* or *forward* an event. In case the incoming fact does not match the rules then the event is discarded, resulting in the filtering of events. When the fact completely matches any of the rules then the event is forwarded to the gateway node. If a partial match is found then the event is shared with the relevant sensor node for further processing.

4.2.5 Transmitter

The *Transmitter* component is responsible for sending events to the gateway nodes or sharing the information with other relevant sensor nodes. The decision regarding with whom to share the information is based on the types of sensors and available data in the locality. Moreover, the communication paths (i.e. routes) are chosen based on energy-awareness. Thus, context and context-awareness are used in the following way:

- Data-context for sharing the information with the relevant sensor nodes, and
- Energy-context for selecting the energy efficient path to the destination.

As data is processed locally at the sensor nodes, information sharing is of utmost importance. A data-related context helps the sensor nodes in finding and sharing their information with the other relevant sensor nodes (i.e. in cases of partial matches). It is based on traditional publish/subscribe (pub/sub) communication patterns [26], where different sources subscribe to information of interest, and sensor nodes that can provide such information become the publishers of the information. An important aspect of pub/sub is in the way events flow from senders to receivers: receivers are not directly targeted by publishers, but rather they are indirectly addressed according to the content of the events.

Most of the earlier pub/sub systems use a centralized architecture, where a centralized broker/mediator regulates the subscriptions and publications [43, 95]. However, as this study aims to process data at the sensor nodes, a distributed solution is provided where every sensor node can be a publisher and subscriber at the same time.

Another important aspect of the pub/sub systems is event routing. Events that match subscribers' interests have to be sent in a manner that does not significantly reduce the network's lifespan. For this reason, this project employs an energy-related context in the sensor nodes. The sensor nodes are aware of their energy levels and the energy requirements of different routes to the subscribers. This is achieved by calculating path costs to each of the subscribers and selecting the most energy-efficient path amongst them for communication. Thus, when the publishers publish an event, they use those routes that have sufficient energy left.

4.3 Cost Analysis of SEPSen

Cost models for in-network data processing have been proposed considering hardware parameters [45, 98] and for routing in homogeneous WSNs [84, 87]. No cost model appears to have been proposed for processing and routing in heterogeneous WSNs. The present energy-based cost model shows the cost of in-network data processing and presents circumstances under which it is advantageous (in terms of energy benefits) for heterogeneous sensor networks. It is important to note that the cost analysis does not cover the data storage and latency cost caused due to in-network data processing. However, the evaluation (see Chapter 6) of the implemented architecture provides analysis and simulation results of all these aspects (i.e. sensing, processing, communication, storage and latency) of in-network data processing in heterogeneous WSNs.

4.3.1 Comparative Cost Model

Costs are measured as energy consumed at sensor nodes for sensing, processing and communicating events to the gateway node. The total energy consumption of a WSN depends on the set of events E available to the network and the set of filters F for simple and complex events:

$$C_{wsn}(E,F) = C_{sense}(E) + C_{proc}(E,F) + C_{comm}(E,F) \tag{4.1}$$

$$= \sum_{e \in E} C_{sense}(e) + \sum_{\substack{e \in E \\ f \in F}} C_{proc}(e, f) + \sum_{e \in E_F} C_{comm}(e)$$

where E_F refers to the set of events being communicated within the network (i.e. the events $e \in E$ that are matched by filters $f \in F$). Note that $\varnothing \subseteq E_F \subseteq E$ if F contains only simple event filters ($E_F = E_F^{sim}$). For complex events, E_F may be larger or smaller than E when using in-network processing ($E_F = E_F^{sim} + E_F^{com}$). For centralized processing it always holds that $E_F = E_F^{sim}$.

To compare the costs of in-network processing $C_{wsn}^{I}(E, F)$ to centralized processing $C_{wsn}^{C}(E, F)$, each of the three cost components need to be evaluated. Naturally, the costs of sensing events are equal $C_{sense}^{I}(E) = C_{sense}^{C}(E)$ (equal node sets and sensors).

Processing cost

In the centralized setup, the processing cost at each sensor node $n_i \in N$ (set of all nodes N, $i \in \mathbb{N}$) is twofold: simple event filtering at the source node $n(e) \in N$ of an event e (F^{sim} being the set of simple event filters; and routing table looking-up to identify the direction of communication towards the gateway node GW:

$$C_{proc}^{c}(E, F) = \sum_{\substack{e \in E \\ f \in F^{sim}}} C_{filter}(e, f) + \sum_{\substack{n_i \in N \\ e \in E_F^{sim}}} C_{rlup}(e, n_i) x \chi(n_i, n(e), GW) \quad (4.2)$$

$$\text{with } \chi(n_i, n(e), GW) = \begin{cases} 1 & n_i \text{ part of route from } n(e) \text{ to GW} \\ 0 & \textit{otherwise} \end{cases}$$

Note that any calculations at the gateway node do not need to be factored in for energy consumption.

In complex in-network processing, a source node $n(e)$ consumes energy by filtering the sensed event data for simple events and processing the matching events for semantic annotation. Source nodes and other nodes en-route to the gateway

additionally perform filtering of complex events, semantic annotation of complex events and routing. Complex event detection may require several sensor nodes to recognize an event; nodes that identify complex events are called *rendezvous nodes*. There may be several rendezvous nodes identifying parts of complex events ($r(e)$ refers to the number of nodes for event e). The overall processing costs are:

$$\begin{aligned} C^{I}_{proc}(E,F) = & \sum_{\substack{e\in E \\ f\in F^{sim}}} C_{filter}(e,f) + \sum_{e\in E^{sim}_{F}} C_{semantic}(e) \\ & + \sum_{\substack{e\in E^{com}_{F} \\ f\in F^{com}}} C_{filter}(e,f) xr(e) + \sum_{e\in E^{com}_{F}} C_{semantic}(e) \\ & + \sum_{\substack{n_i\in N \\ e\in E_F}} C_{rlup}(e,n_i) * \bar{\chi}(n_i, n(e), GW) \end{aligned} \tag{4.3}$$

$$\text{with } \bar{\chi}(n_i, n(e), GW) = \begin{cases} 1 & n_i \text{ part of the context-dependent} \\ & \text{route from } n(e) \text{ to GW} \\ 0 & \textit{otherwise} \end{cases}$$

By comparing the above equations (4.2) and (4.3), we can see that:

$$C^{C}_{proc}(E,F) \leq C^{I}_{proc}(E,F) \tag{4.4}$$

Communication cost

In the centralized setup, all sensor nodes forward their matching simple events $e \in E^{sim}_{F}$ to the gateway node according to their sampling frequency $f(e)$ traversing along a multi-hop network. Each hop within the network is assumed to incur a base energy cost of C^{hop}, which may vary for different WSNs. Complex events are detected at the gateway node. Their contributing simple events need to be communicated to the gateway. For a complex event e^{c}, the set of all contributing

simple events is denoted by $E_F^{sim}(e^c)$.

$$C_{comm}^{C}(E,F) = \sum_{\substack{n_i \in N \\ e \in E_F^{sim}}} f(e) * \chi(n_i, n(e), GW) * C^{hop} + \sum_{\substack{n_i \in N \\ e^c \in E_F^{com} \\ e \in E_F^{sim}(e^c)}} (f(e,n_i) \times \chi(n_i, n(e), GW)) * C^{hop} \tag{4.5}$$

Simple event detection in in-network processing is identical. For complex event detection, let R be the set of all rendezvous nodes and $S(n_r, e^c)$ the set of all sensor nodes that collaborate to form a complex event e^c at node n_r. Overall communication costs are:

$$\begin{aligned} C_{comm}^{I}(E,F) &= C_{comm}^{I}(E_F^{sim}) + C_{comm}^{I}(E_F^{com}) \\ &= \sum_{\substack{n_i \in N \\ e \in E_F^{sim}}} f(e) * \bar{\chi}(n_i, n(e), GW) * C^{hop} \\ &+ \sum_{\substack{n_r \in R \\ n_i \in N \\ e^c \in E_F^{com}}} \Big(f(e^c, n_r) \times \bar{\chi}(n_i, n_r, GW) \\ &+ \sum_{\substack{n_j \in S(n_r, e^c) \\ e \in E_F^{sim}(e^c)}} (f(e,n_j) \times \bar{\chi}(n_j, n(e), n_r)) \Big) * C^{hop} \end{aligned} \tag{4.6}$$

For both centralized and in-network communication, the cost-reduction factor caused by overlaps between simple events and events contributing to complex events is omitted.

4.3.2 Tradeoff Analysis

This section analyses the conditions for in-network processing to be advantageous for energy consumption, i.e., $C_{wsn}^{I}(E,F) \leq C_{wsn}^{C}(E,F)$. For this to be true, the

following condition is required while using the above equations:

$$\sum_{e \in E_F} C_{semantic}(e) + \sum_{\substack{e \in E_F^{com} \\ f \in F^{com}}} C_{filter}(e, f) * r(e) + \sum_{\substack{n_i \in \Delta N \\ e \in E_F}} C_{rlup}(e, n_i) \leq C_{comm}^{C}(E, F) - C_{comm}^{I}(E, F) \tag{4.7}$$

with ΔN being the set of nodes that may additionally have to be traversed using context-aware routing. Thus the condition states that the costs for semantic annotation, complex filtering and routing-table lookup in additional nodes have to be less than or equal to the improvements in communication cost.

Communication costs will decrease if the frequency of simple events contributing to complex events is higher than that of the complex events. This has been argued in [84, 87]. However, this depends on both the set of filter definitions F and the WSN layout.

4.4 Limitations of the System Design

Three functional requirements are mentioned in Section 2.5 (i.e. in-network data processing, semantic data integration and context-awareness) for an energy-efficient WSN. For in-network data processing, the focus is on support for detection of complex events within the WSN, throughout the research presented in this thesis. A complex event is described as a *combination of simple events that occur within a specified period* in Chapter 1. The authors in [87] classified the detection of events into the following four main categories:

- Local instantaneous event detection,
- Local history-sensitive event detection,
- Distributed instantaneous event detection, and
- Distributed history-sensitive event detection.

For this thesis, analysis of related work (see Chapter 3) mainly focused on distributed event detection in any aspect (i.e. instantaneous or history-sensitive) of the event detection described above for an unbiased review of those approaches. However, the design developed in this thesis addresses and focuses on instantaneous event detection, both local and distributed, for the detection of simple and complex events respectively. This is due the fact that history-sensitive event detection requires the sensor nodes to store historical sensor data. Example of such events are *when the temperature has been above* 20° C *for the last five minutes* or *when the temperature has been above* 20° C *for the last five minutes and dissolved oxygen has decreased by 0.5 mg/L in the same region* [87]. Since, in our design, the facts are being regularly updated (i.e. sensor events) as and when they arrive in the knowledge base of the sensor nodes, detection of history-sensitive events cannot be performed. The decision to limit the knowledge base is dictated by the sensor nodes' limited storage and the advantage is that only minimal amounts of information (facts) are stored at the sensor nodes. However, the consequence of such an approach is that the detection of events requiring historical context can not be performed at the sensor nodes.

4.5 Summary

This chapter contributes to answering the second research question identified in Section 1.4.2, i.e., *"How can complex event detection tasks be performed in a resource constrained sensor node?"* by proposing the conceptual design of such a system. It builds on the recommendations discussed in Section 2.5, keeping in view the design requirements mentioned in Section 4.1. The design fulfils the recommendations by providing support for semantic annotation, in-network event detection and context-awareness for communicating events for the detection of complex events within a heterogeneous WSN. The design caters for the limited storage capacity of the sensor nodes by using ontology fragments for semantic annotation, a limited facts base for pattern-matching and efficient context-awareness at the sensor nodes in the sensor network.

This chapter also analyses the energy costs of complex event processing in heterogeneous WSNs. The costs of a centralized approach for complex events (simple events are processed in the distributed WSN) were compared with the fully distributed approach of semantic context-aware in-network processing. This contributes to answering the third research question, i.e., *"how much energy benefit could be gained by performing complex event detection tasks at the sensor node level?"*

A trade-off between lower communication cost and higher processing cost due to semantic annotation, context analysis for routing and processing of complex events was identified. Any approach for context-based routing and semantic processing of complex events need to be under the identified threshold for increased network lifetime.

In the proposed architecture, by processing event data locally, the sensor nodes will make decisions quickly and remotely without the need for instructions from gateway nodes. This makes the proposed architecture particularly appropriate for remote environments and situations in which a stable WSN cannot be guaranteed. The following chapter introduces the implementation of all components (i.e., Receiver, Semantic Annotator, Knowledge Base, Rule Engine and Transmitter) of the conceptual design.

Chapter 5

Prototypical Implementation of SEPSen

The previous chapter described the conceptual design of a novel architecture for semantic data integration in a heterogeneous sensor network. This chapter describes SEPSen, a prototypical implementation of this architecture that focuses on in-network data processing in a heterogeneous sensor network. Together Chapters 4 and 5 address the second research question, i.e., *"how can complex event detection tasks be performed in a resource-constrained sensor node?"*

This chapter is structured as follows: Section 5.1 links SEPSen to the components introduced in the previous chapter. Section 5.2 briefly describes the implementation environment. Section 5.3 explains the ontology design for the implemented SEPSen architecture. Sections 5.4, 5.5 and 5.6 discuss the annotation, event detection and communication processes of the implementation. Section 5.7 discusses the limitations of the implemented prototype and the chapter closes with a summary in Section 5.8.

Early versions of parts of Sections 5.6 have been previously published in [46].

5.1 Focus of Implementation

This section explains how SEPSen fulfills the requirements discussed in the previous chapter. The conceptual design of SEPSen consists of five components i.e. Receiver, Semantic Annotator, Knowledge base, Rule engine and Transmitter. Figure 5.1

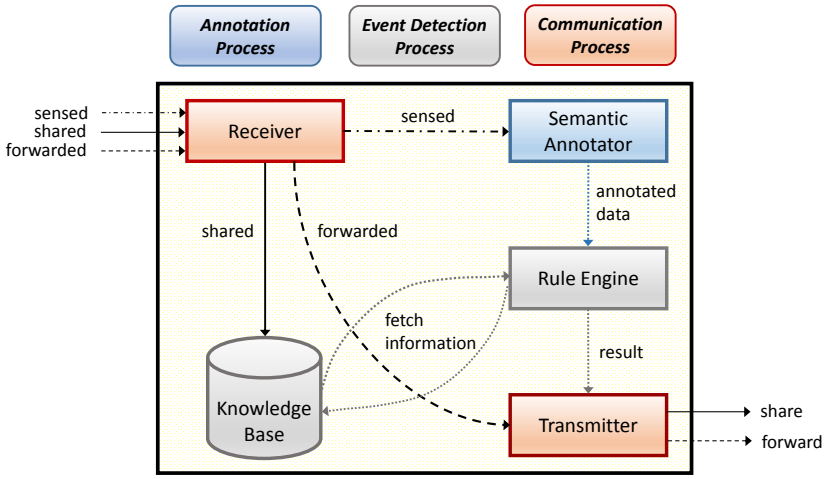


Figure 5.1: SEPSen node architecture: visualization of the processes involved

repeats the visualization of these components according to the processes in which they participate: annotation, event detection, and communication.

The annotator performs the annotation of the sensed data based on the provided ontology fragments. Once the data has been semantically annotated, it is sent to the filtering process. The annotated data and the rules from the Knowledge Base component are gathered by the Rule Engine component and filtering is performed to detect events of interest. Finally, the communication process ensures the transmission and reception of the filtered events through the Transmitter and Receiver components of SEPSen. Details of the implementation according to the processes involved is provided later in this chapter.

5.2 Implementation Environment

We used Protégé 3.4.4[1] for designing the ontology and the OntoViz[2] plugin for visualization. The ontology was exported in N3 (i.e. Notation3) format[3] to be used in the sensor nodes for semantic annotation purposes. The complete core ontology can be viewed in Appendix A.

[1] http://protege.stanford.edu/
[2] http://protegewiki.stanford.edu/wiki/OntoViz
[3] http://www.w3.org/TeamSubmission/n3/

The application is implemented in TinyOS [55]. TinyOS is an event-driven, open-source operating system for wireless sensor networks. The application is simulated in PowerTOSSIM [77], which is an extension of TOSSIM simulator [54].

5.3 Ontology as Knowledge Base

There are numerous available ontologies that can be used to model the sensor data. Since building ontologies from scratch is a laborious process, a search for an existing ontology that fully or nearly matched the knowledge-representation requirements of the project was done to avoid unnecessary work. For this project, TWC-SWQP [88, 89] ontology was reused and extended as the basis for the core ontology for this thesis, called the Water Quality Ontology (WQO), as shown in Figure 5.2.

TWC-SWQP provides a water core ontology and regulation ontology. The water core ontology includes terms for relevant water pollution concepts such as MeasurementSite, ObservedProperty and PollutedWaterSource. The regulation ontology models water quality regulations for various federal and state organizations such as

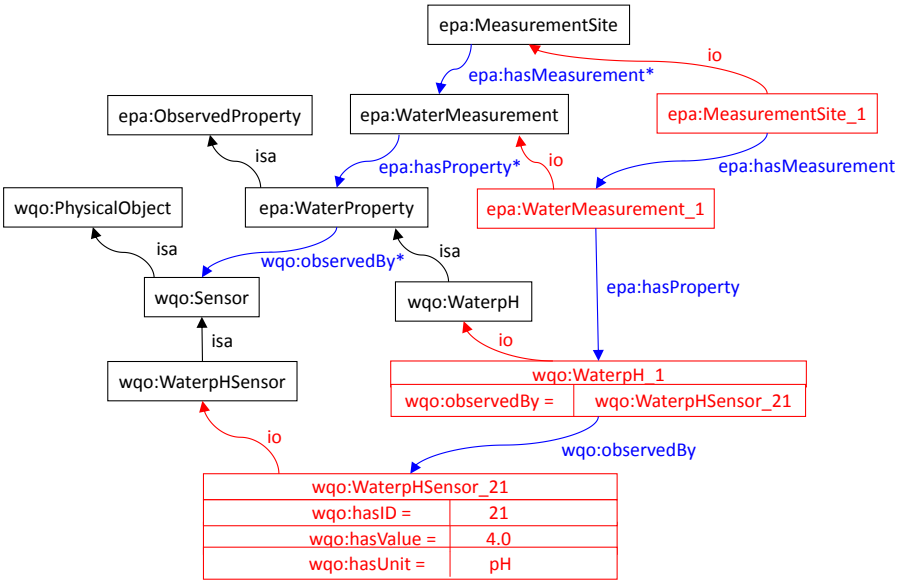


Figure 5.2: Fragment of WQO core ontology: black elements denote classes and subclasses, blue elements refer to ObjectProperties i.e. relationships between classes, and pink elements refer to instances of classes

the New Zealand Environmental Protection Agency (EPA). It defines the pollutants and their maximum containment levels, such as *"any measurement has value 0.01 mg/L is the limit for Arsenic"*. Together these ontologies are used to identify water sources that are polluted due to violations of regulations.

This ontology was therefore extended to support regulatory standards of interest in the present study. The TWC-SWQP water core ontology was not completely suitable for the purposes of this project. Its shortcomings are twofold: (i) TWC-SWQP does not model sensors, and (ii) TWC-SWQP uses OWL2 [39] classification inference to support regulation features. The present study introduced sensor modelling, and customised rules to support regulation features.

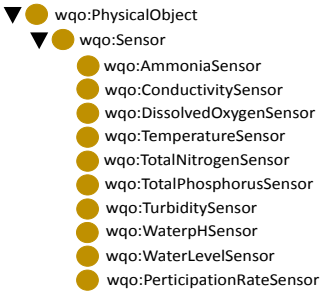


Figure 5.3: PhysicalObject Class

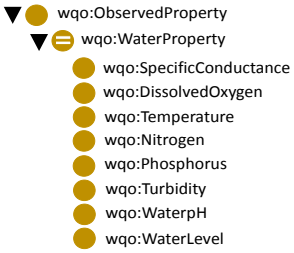


Figure 5.4: ObservedProperty Class

To incorporate the modelling of sensors, the class PhysicalObject was introduced (see overview of classes and subclasses in Figure 5.3). To represent different types of sensors, the subclass Sensor was added to this class. The sensors class was further categorised into subclasses for each sensor type, e.g. TemperatureSensor, TurbiditySensor and WaterpHSensor. An object property ObservedBy was added to indicate that sensors observe the WaterProperty, which in turn is an ObservedProperty. Different water properties that sensors observe under the class WaterProperty were also added (see overview of classes and subclasses in Figure 5.4). These include Water Temperature, Turbidity, and WaterpH. With the introduction of these water properties, links were possible with the specific sensors that make those observations, such as WaterpH is observed by WaterpHSensor for instance.

TWC-SWQP infers water pollution events using OWL2 inference. Based on this study's design pattern, this could not be done at the ontology level, due to the involvement of the expressive OWL property restrictions in TWC-SWQP regulation ontology which cannot be implemented on resource constrained devices such as sensor nodes [7]. The custom rule is a suitable solution for expressing this rationale. For example, the WQO rule *"If WaterpH is less than 7.0, then water source is polluted"* may be expressed through the following rule:

```
epa:MeasurementSite(?p) & wqo:hasMeasurement(?p, ?q)
& epa:WaterMeasurement(?q) & wqo:hasProperty(?q, ?r)
& wqo:WaterpH(?r) & wqo:observedBy(?r, ?s)
& wqo:WaterpHSensor(?s) & epa:hasValue(?s, ?t)
& lessThan(?t, 7.0) -> epa:PollutedWaterSource(?p)
```

This rule checks each sensor observing WaterpH, to filter the sensor observations that are less than 7.0. If the observations satisfying all these conditions are met, then that water source is polluted. In the implementation for this thesis, fragments of this ontology along with the custom rules are provided to the relevant sensor nodes for event detection (more details in Section 5.5).

5.4 Annotation Process

The annotation process is implemented in the semantic annotator component of the SEPSen node. The semantic annotation of sensor data in SEPSen is based on the Resource Description Framework (RDF) data model. Information in RDF is represented in triplets of subject-predicate-object statements [14]. Since the object of one statement can be the subject of another statement, a set of statements forms a graph, with subjects/objects as nodes and predicates as edges.

Figure 5.5 shows the corresponding graph of nodes (subjects/predicates) and edges (predicates), and the RDF statement for the English language statement *"WaterTemperature is observed by a TemperatureSensor"*. All the nodes are labelled. Edges are shown directed and labelled.

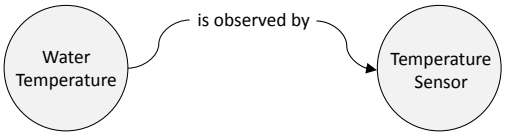


Figure 5.5: RDF Graph

OWL ontologies provide types for subjects/objects and predicates for the RDF statements. The subject of an RDF triplet is an instance of one or more OWL classes. The predicate of an RDF triplet is an instance of an OWL property. The object of a triplet is either an instance of one or more OWL classes (where the predicate is an object property) or a literal (where the predicate is a datatype property). Classes and properties form inheritance hierarchies, since a class or property can be extended to other classes or properties.

The semantic annotation is performed in two phases, explained in the following sections.

5.4.1 Parsing Ontology into a Data Structure

Semantic annotation begins by parsing infused ontology into a data structure that can be maintained in a sensor node. Tables 5.1, 5.2 and 5.3 show the data structure for parsing the ontology into an RDF graph. Each class of the ontology is stored as a graph node with its name as the class name. The node stores information about which other classes it is linked with through the use of an edge. Each node also has related instances that are populated as the sensors collect sample readings within the environment. The edges represent the features of ObjectProperty and DataProperty for the defined classes and their properties. The process links different classes and subclasses as specified by the provided ontology.

5.4.2 Converting Sample Readings into RDF Statements

The second phase in semantic annotation converts the sample readings of the sensor into RDF statements, thus creating instances of the ontology classes. However, the

Table 5.1: Fields of the Nodes

Variable	Description	Defined for ontology concept
name	node's name	class name
instances	null / link to instances	instances of this class
edges	null / link to edges	links the other classes
next node	null / link to next nodes	

Table 5.2: Fields of the Edges

Variable	Description	Defined for ontology concept
name	edge's name	property name
parent	link to parent class	domain of the property
child	link to child class	range of the property
property type	property description	object property / data property
next node	null / link to next egde	

Table 5.3: Fields of the Instances

Variable	Description	Defined for ontology concept
number	instance number	auto-generated instance number
value	instance value	value of this instance
next_instance	null / link to next instance	

sensors need to have their characteristics defined. This is done manually by the user. The sensor descriptions contain the metadata defining the sensor characteristics. Sensor characteristics metadata are described for each sensor following a predefined schema. Table 5.4 shows a description of the WaterpH sensor in SEPSen.

Table 5.4: Sensor Description in SEPSen

Characteristics	Value
hasID	25
isa	WaterpHSensor
hasUnit	pH

Following the characteristics defined for the sensors, when a sensor senses the environment, the raw sensor reading obtained is converted into RDF triplets (fields as described in Table 5.5). The subject of each triplet corresponds to the instances of parent classes, whereas the object of the triplet corresponds to the instances of the child classes. The predicate in the triplet links the subjects to the predicates through property instances defined in the edges.

Table 5.5: Fields of Triples in SEPSen

Variable	Description
id	triple unique id
subject	link to instances of parent of the edges
predicate	link to edges name
object	link to instances of child of the edges

Figure 5.6 shows an example of RDF encoding of observations made by the WaterpH sensor during a given interval (note that epa and wqo correspond to the prefixes for the epa and Water Quality ontology).

No.	Subject	Predicate	Object
1	wqo:WaterpHSensor_21	is_a	wqo:WaterpHSensor
2	wqo:WaterpHSensor_21	wqo:hasID	"21"
3	wqo:WaterpHSensor_21	wqo:hasUnit	"pH"
4	wqo:WaterpHSensor_21	wqo:hasValue	"4.0"
○	○	○	○
○	○	○	○
n	wqo:WaterpH_1	wqo:observedBy	wqo:WaterpHSensor_21

Figure 5.6: Triples for water pH sensor data

These triplets of the instances of various classes are represented in N3 format. The reason for using N3 format is its compactness compared to other formats. This saves bandwidth during transmission between sensor nodes in the sensor network.

5.5 Event Detection Process

The event detection process is performed by the *Rule Engine* component of SEPSen based on the facts and rules gathered from the Knowledge Base component. The fact triplets obtained through the annotation process along with the user-specified rules are part of the Knowledge Base (KB). In SEPSen, the rules are specified in a SWRL-like language [40]. The syntax for specifying the rule in SEPSen is:

$antecedent \rightarrow consequent$

The antecedent is the conjunction of conditions ($c_1 ... c_n$) and the consequent consists of a single condition that is the conclusion of the rule. For example, a rule asserting that the composition of WaterpH sensor values that exceed a certain threshold imply the nutrient pollution condition would be written as:

```
wqo:WaterProperty(?p) & wqo:observedBy(?p, ?q)
& wqo:WaterpHSensor(?q) & epa:hasValue(?q, ?r)
& lessThan(?r, 7.0) -> wqo:NutrientPollution(?p)
```

The variables in the rule conditions are assigned by following the standard convention of prefixing them with a question mark (e.g., ?p as shown in the rule above) [40]. This assists in associating various conditions of the antecedent together for rule processing. The implemented rule engine gathers these rules and forms a rule network (i.e. Rete network). It evaluates the rules and triggers specific actions when the conditions are met. Actions are not explicitly specified as discard, share or send, as complete matches are always sent to the gateway without further processing (see Section 5.5.2 for details). Partial matches are always shared amongst the sensor nodes (see Section 5.5.2 for details) and non-matches are always discarded by the sensor nodes.

The prototype of SEPSen in this thesis implements the Rete algorithm for the pattern matcher component. The original Rete algorithm is well explained in [27]. The implemented Rete algorithm can be understood by dividing it into two major steps: (i) constructing the Rete network and (ii) the matching process. The construction step is generally executed only once (except when the rules are updated). The matching process is executed repeatedly as and when new sensor data is received or updated.

5.5.1 Constructing a Rete Network

The construction of a Rete network is based on the patterns of the specified rules. Following the patterns of the rule, three types of nodes are constructed for a Rete network i.e. alpha nodes, beta nodes and rule nodes. Alpha nodes are one-input

nodes that are created for each atomic condition occurring in the left-hand side of each rule. Beta nodes are two-input nodes that correspond to the conjunction of conditions in the left-hand side of each rule. Rule nodes are the terminal nodes having one input and correspond to the right-hand side of each rule.

A node in Rete is a data structure with fields as described in Tables 5.6, 5.7 and 5.8 for alpha, beta and rule nodes respectively. All the fields of alpha, beta and rule nodes are filled with values in the construction of the Rete network, with the exception of the memory field. The memory field is empty after the construction phase and is filled with values in the matching process (see Section 5.5.2 for details) of the Rete algorithm.

Table 5.6: Fields of the alpha nodes

Variable	Description
formula	atomic formula in LHS of the rule
memory	null / link to matching triples
children	null / link to beta nodes

Table 5.7: Fields of the beta nodes

Variable	Description
id	beta node id
memory	null / link to matching triples
child	null / link to other beta or rule nodes

Table 5.8: Fields of the rule nodes

Variable	Description
formula	atomic formula in RHS of the rule
memory	null / link to matching triples

To illustrate the construction phase of the rule engine, thermal pollution and oxygen depletion are used as examples. Temperature varies naturally on a seasonal basis; however, many activities (such as the use of water as a coolant by industrial power plants) have the ability to change the temperature of bodies of water. This causes an increase in the water temperature, thereby resulting in thermal pollution. The elevated temperature mostly affects dissolved oxygen levels. With an increase

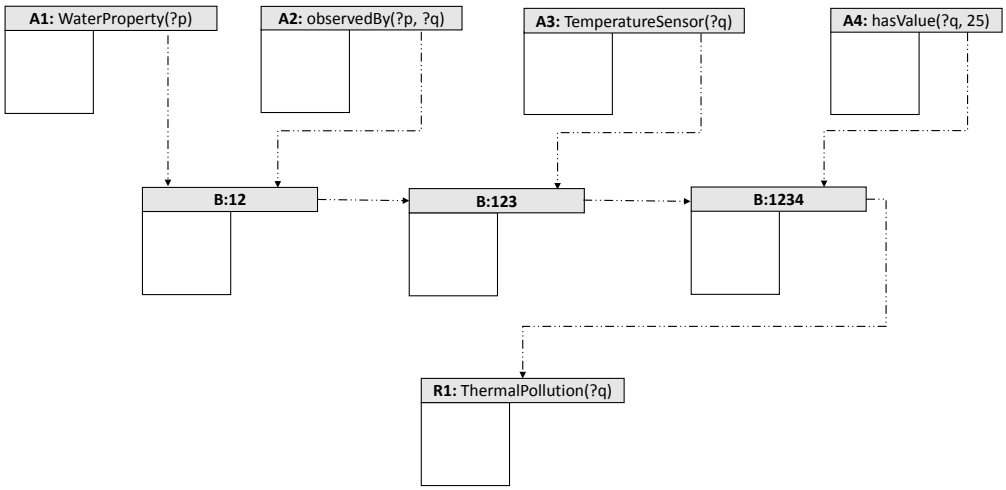


Figure 5.7: Rete network construction for thermal pollution

in temperature, dissolved oxygen decreases and this greatly affects the ecological balance [23].

To detect events of thermal pollution and oxygen depletion, the rules in the sensor nodes measuring temperature and dissolved oxygen of water bodies are used. For example, a rule for detection of thermal pollution in a temperature sensor node can be expressed as[4]:

```
wqo:WaterProperty(?p) & wqo:observedBy(?p, ?q)
& wqo:TemperatureSensor(?q) & epa:hasValue(?q, 25)
-> wqo:ThermalPollution(?q)
```

Figure 5.7 shows the partial Rete network constructed for the thermal pollution rule described above. Each condition pattern gets its own alpha node. The beta nodes are linked together with child links and perform joins on the conditions of the alpha nodes. The formula field of the alpha node links to the corresponding condition, while the rule node has a link to the resultant action specified in the RHS of the rule.

The portion of the network in Figure 5.7 represented by the nodes A1, A2, A3 and A4 corresponds to the simple conditions in the structure of the rule body. Similarly, the portion of the network represented by the nodes B12, B123, and B1234

[4] While temperature above a certain threshold, such as a temperature of above 25° C, is used to detect thermal pollution [23], a fixed value is used here for brevity and compactness.

corresponds to the conjunction of these conditions. Finally, node T1 corresponds to the right hand side of the rule.

Now, we consider a rule for detecting decreased levels of dissolved oxygen (DO) in bodies of water due to elevated water temperature. This oxygen depletion rule depends on the thermal pollution event generated by the temperature sensor node and is detected in conjunction with decreased DO levels by the sensor node measuring the DO of the water. It can be expressed in a rule as[5]:

```
wqo:WaterProperty(?p) & wqo:observedBy(?p, ?q)
& wqo:DissolvedOxygenSensor(?q) & epa:hasValue(?q, 7)
& wqo:ThermalPollution(?r) -> wqo:OxygenDepletion(?q)
```

Figure 5.8 shows the partial Rete network constructed for the oxygen depletion rule described above. Note that the portion of the network in Figure 5.8 represented by the alpha nodes also consists of the thermal pollution condition. The result of all the conditions for reduced DO values are then merged with this condition at the beta node B:12345, which outputs the results to the rule node T1.

5.5.2 The Matching Process

After the construction of the Rete network, triplets (i.e. facts) from the knowledge base (KB) are inserted into the Rete network to perform matching of the facts against the specified rules. The triplet is first sent to all the alpha nodes where the match operation filters only those triplets that match the pattern associated with the alpha nodes. Selected triplets from the alpha nodes are then forwarded down to the beta nodes if they satisfy the conjunction condition of the beta nodes. This process is repeated until it reaches the rule node. When the fact reaches the rule node, it has passed the matching test for a rule body and represents an instantiation of the rule head and its associated action.

The examples for the thermal pollution and oxygen depletion rules presented in

[5] While dissolved oxygen (DO) below a certain threshold, such as DO below 7 mg/L, is used to detect oxygen depletion [23]; a fixed value is provided here for brevity and compactness.

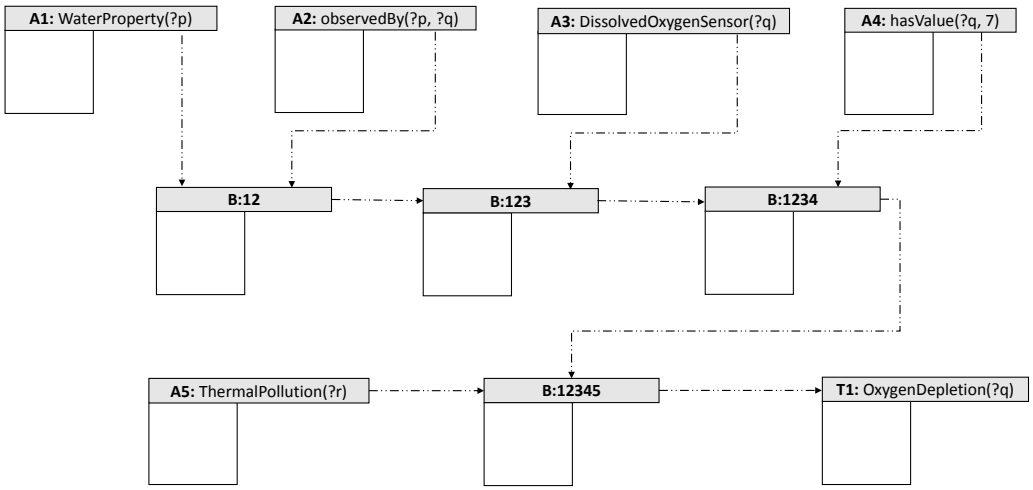


Figure 5.8: Rete network construction for oxygen depletion

pages 72 and 73 respectively are revisited. The thermal pollution rule can be stated in an IF-THEN expression as[6]:

```
Rule ThermalPollution :
IF
  "p" IS A wqo:WaterProperty
  AND "p" wqo:observedBy "q"
     WHERE "q" IS A wqo:TemperatureSensor
    AND "q" HAS VALUE "25" FOR epa:hasValue
THEN
  "q" IS A wqo:ThermalPollution
```

Suppose that the temperature sensor node has the following 10 facts stored in its KB.

```
Asserted Facts:
F1: wqo:P1 is_a wqo:WaterProperty
F2: wqo:P2 is_a wqo:WaterProperty
F2: wqo:P1 wqo:observedBy wqo:T1
F3: wqo:P1 wqo:observedBy wqo:T2
F4: wqo:P2 wqo:observedBy wqo:S1
F5: wqo:T1 is_a wqo:TemperatureSensor
F6: wqo:T2 is_a wqo:TemperatureSensor
F7: wqo:S1 is_a wqo:PhosphorusSensor
```

[6] http://protegewiki.stanford.edu/wiki/Axiomé

```
F8: wqo:T1 epa:hasValue "25"
F9: wqo:T2 epa:hasValue "18"
F10: wqo:S1 epa:hasValue "25"
```

An illustration of the matching process for the thermal pollution example is provided in Figure 5.9. The facts that match the alpha node conditions are placed in the respective alpha node memories. It can be seen from Figure 5.9 that the fact $F7$ does not match the condition *"where q is_a wqo:TemperatureSensor"*, thus, it is not stored in the memory of the alpha node checking for this condition. These facts are then forwarded to the beta nodes where they are checked for the conjunction conditions, and facts that satisfy its condition are then stored in the beta memory. After the successful matching of all the conditions, a fact is passed to the rule node which then activates the action part of the rule. In this scenario, the temperature sensor node shares this event with the dissolved oxygen (DO) sensor node (see Section 5.6 for communication details).

Recall the oxygen depletion scenario from Section 5.5.1 on page 73. The rule states that an increase in the water temperature and a decrease in the dissolved oxygen levels in the bodies of water constitute an oxygen depletion event. The

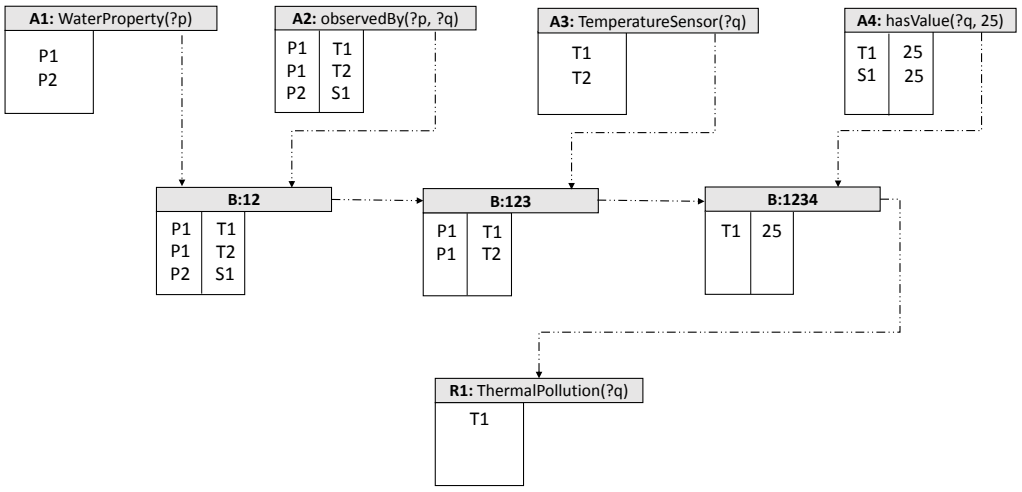


Figure 5.9: Rete matching process for thermal pollution

specified rule for this event can be rephrased in an IF-THEN expression as[7]:

```
Rule OxygenDepletion:
IF
  "p" IS A wqo:WaterProperty
  AND "p" wqo:observedBy "q"
     WHERE "q" IS A wqo:DissolvedOxygenSensor
    AND "q" HAS VALUE "7" FOR epa:hasValue
AND IF
  "r" IS A wqo:ThermalPollution
THEN
  "q" IS A wqo:OxygenDepletion
```

Again, suppose that the DO sensor node has the following 13 facts stored in its KB. Note that the DO sensor node has now also received the thermal pollution event (represented as a fact) from the temperature sensor nodes.

```
Asserted Facts:
F1: wqo:P1 is_a wqo:WaterProperty
F2: wqo:P2 is_a wqo:WaterProperty
F2: wqo:P1 wqo:observedBy wqo:D1
F3: wqo:P1 wqo:observedBy wqo:D2
F4: wqo:P2 wqo:observedBy wqo:H1
F5: wqo:D1 is_a wqo:DissolvedOxygenSensor
F6: wqo:D2 is_a wqo:DissolvedOxygenSensor
F7: wqo:H1 is_a wqo:WaterpHSensor
F8: wqo:D1 epa:hasValue "7"
F9: wqo:D2 epa:hasValue "10"
F10: wqo:H1 epa:hasValue "7"
F11: wqo:T1 is_a wqo:TemperatureSensor
F12: wqo:T1 epa:hasValue "25"
F13: wqo:T1 is_a wqo:ThermalPollution
```

Figure 5.10 shows the matching process for the oxygen depletion event. The alpha memories discard facts $F7, F9, F11$ and $F12$ as they do not match the alpha node conditions. The remaining facts are further pruned at the beta nodes and only when all the conditions are matched is a fact passed to the rule node. The rule node then notifies the gateway node of this event (see Section 5.6 for communication details).

[7] http://protegewiki.stanford.edu/wiki/Axiomé

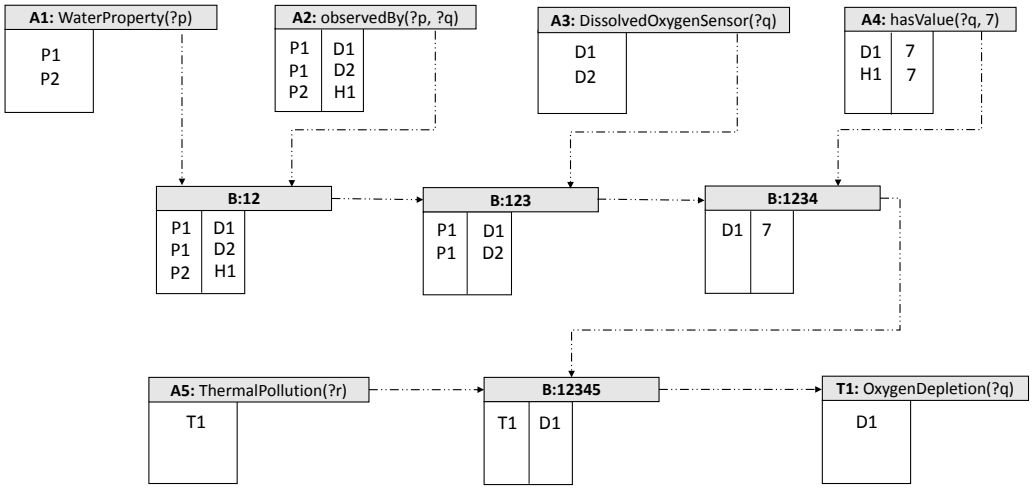


Figure 5.10: Rete matching process for oxygenation

5.6 Communication Process

Communication of partial and complete results is done by the Transmitter and Receiver component of SEPSen. The communication process is performed using Publish/Subscribe (pub/sub) messaging systems [26]. Pub/Sub systems are well-known examples of data-centric communication. The communication model of pub/sub consists of three phases. Initially, the nodes in the network advertise their characteristics or the kind of data that is produced by the sensor nodes (e.g., temperature and humidity). The advertised messages are then sent throughout the network node in a multi-hop fashion. Next, the interested nodes in the network select the desired phenomena to be monitored. These subscription messages are then sent down to the relevant sensor nodes from which this kind of data is requested. After receiving the subscription messages, sensor nodes publish matched events to the subscriber nodes in the network. The generic publish/subscribe system thus can be modelled in terms of these operations:

- Advertise operation: adv(e)
- Subscribe operation: sub(e)
- Publish operation: pub(d,e)

Where 'e' represents an *event topic* that a particular sensor can produce and 'd' represents the actual data against the event topic subscription for the published events. Similarly, different behaviours can be considered with respect to communication models i.e. unicast (u) and broadcast (b). These are:

- u: sensor node sending its data directly to a destination sensor node, and
- b: sensor node sending its data to all the sensor nodes in the network.

The three publish/subscribe operations (i.e. advertisement,subscription, and publication) as used in this scenario are discussed in the following sections.

5.6.1 Advertisement

The advertisement operation is performed by every sensor node in the WSN to announce the availability of a certain kind of data that this sensor node can produce. In the present scenario, the sensor nodes' advertisement is based on the sensor description (see Section 5.4.2) referring to event types i.e. this sensor node can provide events of type *e*. The advertised operation is performed before the actual data is produced by the advertising sensor nodes to let other sensor nodes subscribe to this sensor and hence its data. Thus, the advertise operation is mapped in the following way:

adv_b: As the sensor nodes are unaware of any network topology at this stage, the sensor nodes broadcast the advertisement to all the other sensor nodes in the WSN.

When subscribers initially connect to the WSN, they will not be aware of any prior advertisements. The sensor nodes therefore perform this operation once at the start of network setup and also whenever an advertisement message is received by other sensors.

Figure 5.11 shows the advertisement process for sensor nodes A to F. Sensor nodes receiving the advertisements will forward this *adv* message in the network until it reaches the gateway node. The sensor nodes *en route* will also store information from sensor nodes in their routing tables during the advertisement process.

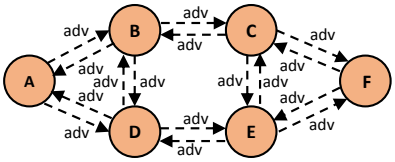


Figure 5.11: Example advertisement process

Figure 5.12 shows the advertisement message frame format. The message type identifies the type of operation as advertisement, subscription or publication. Source ID is the address of the sensor node from which this message is received. Originator ID is the address of the originator of this message (note that initially source and originator ID's will be the same). Sequence No. is the packet unique ID that is used to suppress redundant messages sent by a particular sensor node. Path Energy is the energy of the path selected for packet transmission. Hop count is the number of hops this packet has travelled and meta-data contains the sensor description of the originator sensor node.

Message Type
Source ID
Orig. ID
Seq. Number
Path Energy
Hop Count
Meta data

Figure 5.12: Advertisement message frame format

Sensor nodes *en route* will update Source ID, path energy and hop count fields when propagating the adv message in the network. Sensor nodes receiving the advertisements will check whether they are interested in this sensor event. This sets up the subscription process.

5.6.2 Subscription

Sensor nodes interested in a particular event will subscribe to this kind of event data produced by other sensor nodes. The subscribe operation is performed in the

following way:

sub_u: Since the sensor nodes develop a routing tree in the network during the advertisement process, the subscription message is sent by subscriber sensor nodes in a unicast manner directly towards the producer sensor nodes.

Figure 5.13 shows the subscription process for sensor node C. After receiving the advertisements, the nodes will perform a match to check if they are interested in the available data. This match is based on the rule patterns in the rule engine. An interested sensor node then sends a subscription message to the source node to notify it of the interest. The source node stores the subscription information in its subscription table and notifies the interested nodes of events when they occur.

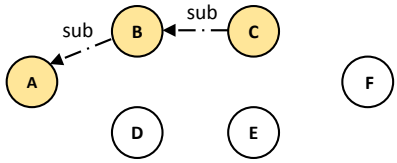


Figure 5.13: Example subscription process

Figure 5.14 shows the subscription message frame format. The message format is similar to the advertisement message format except that the message type field will contain subscription as a message type identifier. Source ID contains the subscriber node's address. Destination ID is the address of the advertiser sensor node, events of which are being subscribed to. The next hop is the address of the neighbouring node through which the subscription message is sent to the advertiser.

Message Type
Source ID
Dest. ID
Next Hop
Seq. Number
Path Energy
Hop Count
Meta data

Figure 5.14: Subscription message frame format

5.6.3 Publication

The publish operation is used by a sensor to publish data, d, to all the subscribers with meta-data, e. This operation is done in the following way:

pub_u: the producer sensor nodes send the event data directly in a unicast manner toward the subscribing sensor nodes in the WSN.

Figure 5.15 shows the publication process from sensor node A to C. The source node notifies the interested destination node by sending a publication of the matched events.

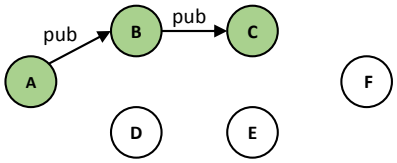


Figure 5.15: Example publication process

Figure 5.16 shows the publication message frame format. The message format is similar to the subscription format except that here the actual data are sent. Here the message type field contains the publication identifier, specifying the publication of an event. Source ID is the address of the publisher. Destination ID contains the subscribers' addresses. The next hop is the chosen neighbour's address for forwarding the publication to the subscriber.

Message Type
Source ID
Dest. ID
Next Hop
Seq. Number
Path Energy
Hop Count
Data

Figure 5.16: Publication message frame format

5.6.4 Energy-aware Routing

Sensor nodes are also provided with energy context-awareness for event routing. This enables the sensor nodes to take the energy levels of different routes into account when selecting energy-efficient paths for communication. The process of path calculation and path recalculation for altering routes efficiently is described in the following sections.

Path Calculation

Energy-aware path selection is performed by the destination sensor nodes during the advertisement process. This process is performed in the following way:

1. At the originator node (N_o), initially the path energy (pe) field is set to zero. i.e.

$$pe(N_o) = 0$$

2. Before forwarding a message to other nodes, the originator node adds its residual energy ($re(N_o)$) to the path energy. Thus, the path energy becomes:

$$pe(N_o) = pe(N_o) + re(N_o)$$

3. On receiving a message from the originator, the receiving nodes (N_r) *en route* to the gateway node then add their residual energies to the path energy received from the neighbouring nodes (N_n). The nodes (N_r) calculate the mean energy of this path and set the path energy for this node to reach the originator node through a neighbouring node as:

$$pe(N_r, N_o, N_n) = \frac{1}{2} \cdot (pe(N_n) + re(N_r))$$

   Note that nodes (N_r) that are 1 hop away from the originator (i.e., direct neighbors of the originator node) will calculate the path energy for the originator

node (N_o) as:

$$pe(N_r, N_o, N_o) = \frac{1}{2} \cdot (pe(N_o) + re(N_r))$$

4. Each node *en route* then selects the best neighbour (in terms of more residual energy along the path) for communicating with the originator node. This is done by calculating the best route (br) to the originator through one of its neighbours. The selection of the best route through a particular neighbour is based on the highest path energy along the route to the originator. In case of a node having multiple neighbours (N_{n_i}, N_{n_j}) having routes with the *same* path energy to the originator, i.e. $pe(N_r, N_o, N_{n_i}) = pe(N_r, N_o, N_{n_j})$, the node selects the best route through a neighbour with the least number of hop counts (hc). [Note that in the equation below $pe(N_r, N_o, N_{n_i})$ is denoted v_i and $pe(N_r, N_o, N_{n_j})$ as v_j in the condition part].

$$br(N_r, N_o, N_n) = \begin{cases} \max\left(pe(N_r, N_o, N_{n_i}), pe(N_r, N_o, N_{n_j})\right) & \text{if } v_i \neq v_j \\ \min\left(hc(N_r, N_o, N_{n_i}), hc(N_r, N_o, N_{n_j})\right) & \text{if } v_i = v_j \end{cases}$$

However, the 1-hop neighbours of the originator will always have the best route value for the originator node as:

$$br(N_r, N_o, N_o) = pe(N_r, N_o, N_o)$$

5. In case of multiple routes to the originator (see step 4 of path calculation), the node also stores the second best route value i.e. alternate route (ar) value for the originator. This serves as a threshold for path recalculation to alter the paths for balanced energy depletion and uniform network usage.

$$T(N_o) = \begin{cases} 0 & \text{if no alternate route is available} \\ ar(N_r, N_o, N_n) & \text{otherwise} \end{cases}$$

6. Each of the nodes *en route* then forwards this message to its neighbours (except the one it received the message from) until it reaches the gateway node. The message is forwarded only if the path energy received through the other node is more than the existing path energy for that particular originator. Nodes that are 1 hop away from the originator discard the advertisement messages of originators from other nodes.

Once the path has been selected, the destination and originator sensor node will use it for subscriptions and publications. However, this might deplete one segment of the network's node energy, therefore path recalculation will be used (see description later in Section 5.6.4).

A small example is provided to illustrate initial path calculation. The small example network is shown in Figure 5.17. Each step corresponds to one network sketch.

(a) The first image shows the topology of a small example network with six nodes. Each sensor node's residual energy is indicated next to the individual nodes. Node A is assumed to be the source and node F is the destination.

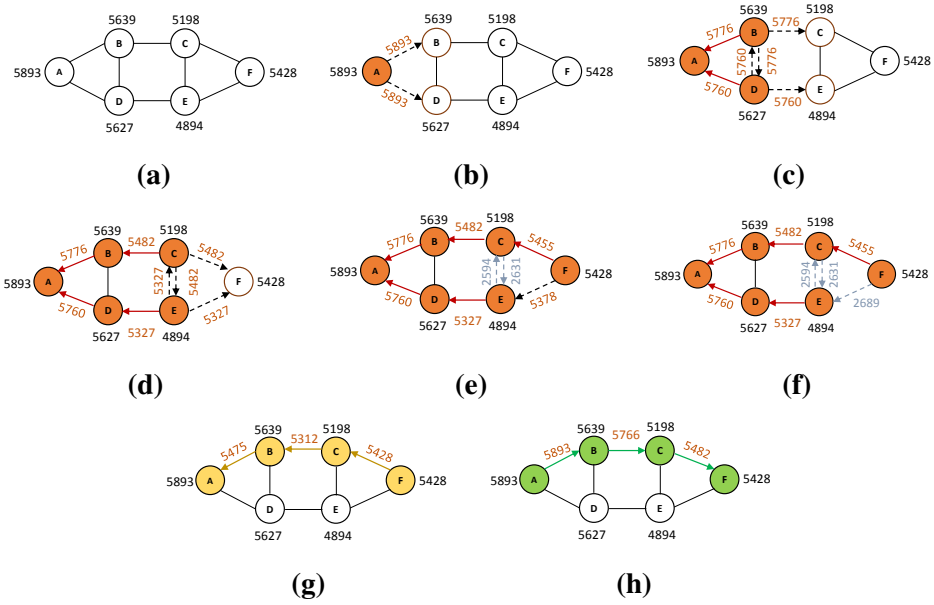


Figure 5.17: Path calculation example

(b) Node A broadcasts the advertisement message to all its neighbours. The path energy is set to the originator node's residual energy.

(c) Nodes receiving the advertisement message will further propagate it to their neighbours with updated path energy. Nodes within one hop of the originator will store the originator node's information in their neighbour tables.

(d) Nodes receiving the advertisement will only re-broadcast the message if the originator is not in their neighbour table or the calculated path energy is higher than the existing path energy for that originator.

(e) Nodes receiving multiple advertisement messages for the same originator will update the alternate route value. This value is used as a threshold to alter the route for balanced energy depletion.

(f) Node F receives two advertisement messages about node A from nodes C and E. It calculates the best and alternate routes (see further details in Section 5.6.4) for node A in its forwarding table.

(g) Node F sends a subscription message using the neighbour with the best route value. The path energy is updated for every sent packet. Node A, receiving the subscription, will place Node F information in its subscription table.

(h) Node A sends the publication to Node F using the path information from its forwarding table for Node F.

Path Recalculation

Once the path has been selected, the destination and originator sensor nodes will use it for subscriptions and publications. However, this might deplete one segment of the network's node energy. To ensure graceful degradation and balanced energy depletion, path recalculation is used. The path recalculation is based on the concept that the sensor nodes know that there is an alternate route available that can be used

(step 6 of the original routing process introduces above) for packet transmission. This process is performed in the following way:

1. Each node will adjust the threshold value (T) (i.e. alternate route value) for the originator as obtained in Step 5 of path selection. That is:

$$T'(N_o) = T(N_o) \cdot \alpha$$

   Since the sensor nodes do not keep the complete information of alternate routes in their routing tables (not feasible due to limited sensor storage), a weighting factor α is added to normalize the threshold value to the initial calculations of the alternate route. This is necessary as the nodes have no idea of any change in the path energy that may have occurred on alternate routes once they select the best route for a particular originator and use it for communicating with the originator node. The value of α is predefined for all the alternate routes, where $0 \leq \alpha \leq 1$.

2. After normalizing the threshold value, the node monitors path energy received from the events published by the originator to this node. Whenever the path energy (pe) for a particular originator decreases below that of the normalized threshold (T'):

$$pe(N_r, N_o, N_n) < T'(N_o),$$

   the receiver node N_r modifies the best route value and threshold value for that originator as:

$$br(N_r, N_o, N_n) = T'(N_o) \quad \& \quad T(N_o) = pe(N_r, N_o, N_n),$$

   and sends a request for path recalculation.

3. Path recalculation is performed by the receiver node (N_r) to notify the originator to use an alternate path, as the best route is no longer feasible for publishing events. Thus, the receiver node sends a path recalculation message through all the nodes in the path to the originator node. All the nodes involved in this recalculation process will follow the same path selection process discussed earlier. That is, now the receiver node will broadcast a request through its neighbours by following steps 1 and 2 of the path selection process. The neighbours *en route* will send this request further by forwarding it to their neighbours until it reaches the originator. All the nodes *en route* will perform steps 3, 4, 5 and 6 of the path selection process. The originator node (N_o) then selects the best route and publishes events using the updated path.

4. The receiver node (N_r) then puts this newly obtained best route value in its best route value for that originator. It then continues to repeat the path recalculation steps to recalculate the routes whenever necessary.

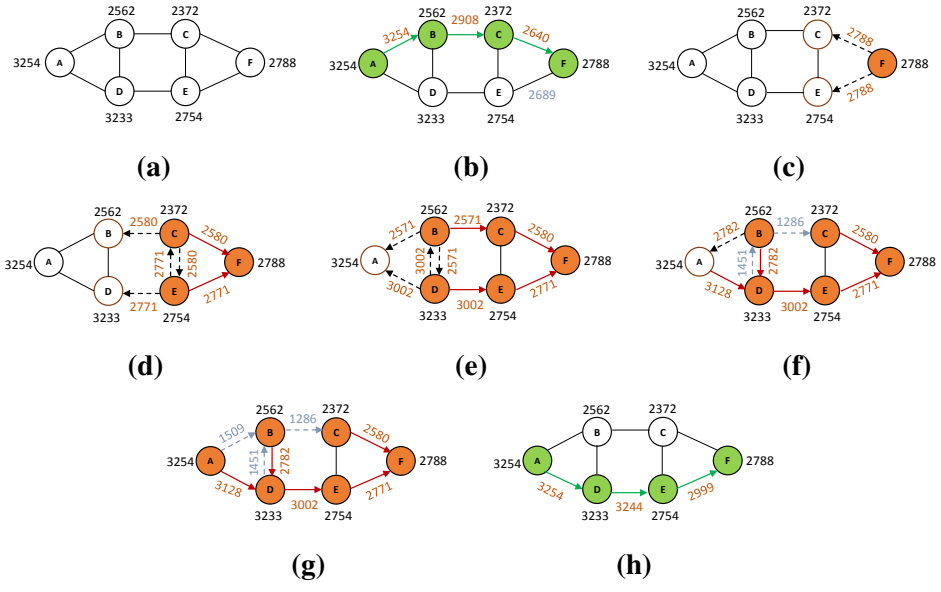


Figure 5.18: Path recalculation example

The previous example is revisited and the recalculation steps are shown in Figure 5.18. Again, each of the following steps refers to one network sketch.

(a) The first figure shows the latest residual energies of the individual nodes. Again, node A is the source and F is the destination.

(b) Node A sends the publication to Node F using the path information from its forwarding table for Node F. Node F monitors the path energy from the neighbour it receives the publication from.

(c) The path energy for Node A through neighbour Node C decreases below that of the adjusted threshold for Node A (recall Figure 5.17f), thus Node F sends a request for path recalculation.

(d) The request for path recalculation is performed in the same manner as the original path calculation was performed.

(e) Node A receives two path energy values for Node F through Nodes B and D. It calculates the best and alternate route for Node F in its forwarding table.

(f) Node B receives a better path energy value through Node D. It sets its new best route through neighbour D and an alternate route through Node C to Node F. Node B also forwards the updated best route value to C and A. Node C discards the message as it is one hop away from the originator of the message i.e. Node F.

(g) Node A updates the alternate route value for Node F in its forwarding table.

(h) Node A applies the original path selection process and uses the best neighbour for sending the publication to node F. Node F then updates the best route value for Node A in its forwarding table.

5.7 Limitations of the Implemented Architecture

The semantic technologies used for annotation and transmission of events in SEPSen are based on the standards suggested by the Semantic Web community. Representations produced by the Semantic Web community, such as Notation 3 (N3), Turtle, and N-Triples, have good semantic expressive power and are easy to be interpreted. This allows for easy adaptation as new networks can be added without requiring any transformation of the existing knowledge representation.

We acknowledge that different binary data formats, such as ASN.1, might be beneficial in terms of storage and processing [70]. However, they cannot be transformed into any knowledge representation in a straightforward manner. Another alternative would have been to compile down the ontology into an executable form and distribute it in the sensor network. Since byte code is much smaller compared to the compiled binary code, updates in the systems can be easily distributed. This way, however, interpreted execution would be slower and some resources would be always used by the virtual machine.

The prototypical implementation of SEPSen, however, has several limitations, such as:

- Triplets in semantic webs generally use URIs to name things and their relationships. However, due to space restrictions in each node, the present implementation does not use full URIs but only prefixes in triplets. At the gateway node, post-processing can be used to include the URIs for further communication towards the end-user.

- Different variants of web ontology languages (OWL) provide different levels of expressivity. A subset of OWL-Lite is implemented for the reasoner and therefore it cannot fully utilize the expressivity of other more expressive OWL variants such as OWL-DL and OWL-full.

- The SWRL rule language supports a range of built-in operations which greatly

expand its expressive power (e.g., subtract, lessThanOrEqual), whereas the current implementation only supports basic equality and inequality comparison operators (i.e. greaterThan and lessThan).

- Since the focus of this thesis is to perform in-network data processing of complex events in a heterogeneous WSN, automatic infusion of ontology fragments into the sensor nodes was not performed. Rather, the ontology fragments were manually infused into the sensor nodes and used for semantic annotation and event detection purposes.

However, these limitations of the implemented prototype do not limit the overall functionality and viability of the concept introduced in this thesis.

5.8 Summary

This chapter contributes further to answering the second research question identified in Section 1.4.2, i.e., *how can complex event detection be performed in a resource-constrained sensor node?* by introducing and implementing the conceptual design described in the previous chapter.

This study focuses on the event detection of water pollutants in a river. Thus an ontology, called Water Quality Ontology (WQO), was developed that relates to water quality monitoring. The developed ontology extends an existing ontology called TWC-SWQP by adding concepts for water quality parameters observed by the sensor nodes.

The prototype implementation (SEPSen) divides the overall complex event detection into three processes, namely annotation, event detection and communication. In the annotation process, the sensor data is semantically annotated to be provided for event detection and communication of events. The semantic annotation of the sensor data is based on the ontology fragments of WQO. The annotation of sensor data allows the sensor nodes to collaborate with each other and perform detection of events within the sensor network.

Event detection using SEPSen is based on the rule engine that builds a pattern network of the rules (water regulations) gathered from the knowledge base and matches the facts (sensor data) against it. Facts that match the rules are then shared or forwarded to the relevant sensor nodes or the gateway nodes respectively. This communication is performed using the data and energy context-awareness of the sensor nodes in the network.

This prototype implementation, however, uses limited SWRL constraints for defining rules. Moreover, a subset of OWL-Lite features are used to reason over the provided facts, and the triplets use only prefixes for names and their relationships. However, these limitations do not hamper the functionality and viability of the conceptual design for the purposes of the objective mentioned in Section 1.3. The next chapter evaluates the performance of the implementation. The evaluation is based on the simulation results in various scenarios.

Chapter 6

Performance Evaluation

The previous two chapters described the conceptual design and a prototypical implementation of the SEPSen architecture. The goal of this architecture, following the objective of the thesis, is to perform complex event detection at the sensor node level for energy benefits. To determine whether the architecture reaches this goal, it needs to be evaluated.

The third research question in Section 1.4.3 asks about energy benefits obtained by processing complex events at the sensor node level. This chapter contributes to answering this question by evaluating the architecture based on how much energy is consumed by different architectures, i.e., SEPSen versus a centralized architecture.

This chapter is structured as follows. Section 6.1 describes the simulation environment used for the evaluation of the architecture. Section 6.2 provides a comparison of total energy consumption by different architectures. Section 6.3 evaluates the overall network lifetime based on the energy distribution at the sensor nodes. Section 6.4 provides insight into the memory usage by various components of the sensor node. Section 6.5 presents the implications, in terms of time consumption, for processing events at the sensor node level. The chapter concludes with a summary in Section 6.6.

Early parts of Sections 6.2 and 6.3 have been published previously in [47] and [46], respectively.

6.1 Simulation Environment

Simulations of the SEPSen prototype were done using PowerTOSSIM [77]. PowerTOSSIM is based on the TinyOS [55] operating system and the TOSSIM [54] simulation environment. The specified energy model for the simulations was based on Mica2 sensor nodes.

The filters for simple events in the SEPSen architecture are based on water quality guidelines mentioned in [11] and [24]. These guidelines provide a "trigger" value for various water quality parameters (such as pH, DO and temperature) and thus these triggers act as a filtering condition for the generation of simple events. The filters for complex events are based on strong positive or negative correlations amongst various water quality parameters mentioned in [1, 11, 23, 72]. Examples of such events are oxygen depletion due to high temperature values and the ratio of total nitrogen to total phosphorus for nutrient limitation conditions. In addition, the filters for complex events are also based on multiple water quality parameters to determine the primary and secondary symptoms of existing water conditions. An example of this is the assessment of high chlorophyll *a* as a primary symptom and depleted dissolved oxygen as a secondary symptom for determining eutrophic conditions in a body of water [11].

Four sets of tests were performed, the results of which are reported in the following sections. The first two tests (i.e. Test 1 and Test 2), compared the new architecture against a centralized approach (shown in Figure 6.1) and analyses the energy consumption at the sensor nodes in each architecture and its effects on the overall network lifetime. In the centralized approach, sensor nodes forward the raw sensor data to the gateway nodes and all processing for simple and complex events is performed at the gateway nodes. In contrast, in the new architecture (SEPSen), sensor nodes process the sensor data at their own level. SEPSen is characterised as operating within two modes i.e. simple and context-aware. In the simple mode of SEPSen, sensor nodes annotate the sensed data and apply filters for detection

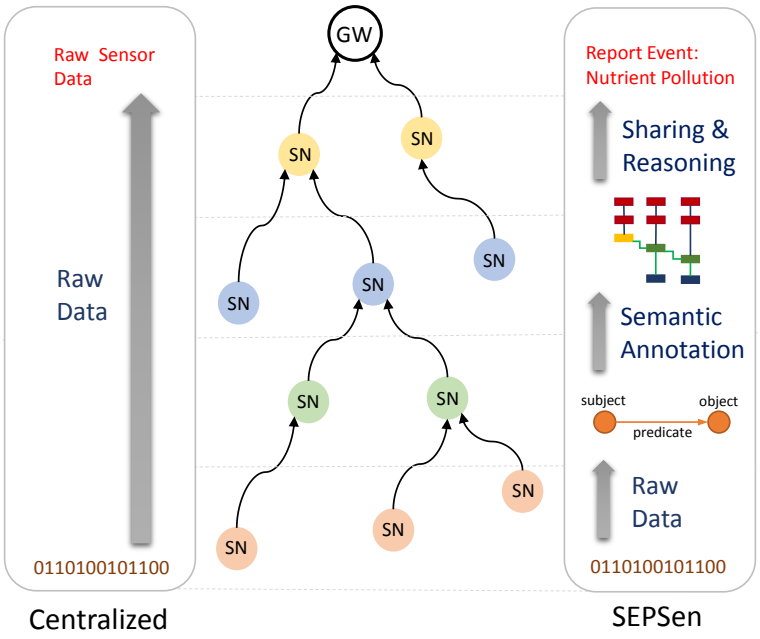


Figure 6.1: Processing for centralized approach vs SEPSen

of simple events. The events of interest are then sent to the gateway nodes. This mode does not, however, perform the detection of complex events and does not use context-aware routing as sensor nodes only share events with a single destination i.e. gateway nodes. The gateway nodes then perform the detection of complex events at their level. In the context-aware mode of SEPSen, detection of simple and complex events, with the help of context-aware routing, is performed at the sensor nodes and only results are provided to the gateway nodes (explained earlier in Chapters 4 and 5).

The other two tests (i.e. Test 3 and Test 4) analysed only the memory usage and processing time of the SEPSen architecture. This gave a measure of the limitations of the implemented SEPSen prototype.

6.2 Test 1: Total Energy Consumption

The purpose of this test was to evaluate the total energy consumption for both the centralized architecture and the SEPSen architecture in simple and context-aware modes. These experiments show the total amount of energy consumed due to sensing,

processing and communication by the sensor nodes in both the architectures and also provide an extension to it by showing the amount of energy consumed in the individual tasks of sensing, processing and communication by the sensor nodes in these architectures. It also provides threshold analysis for the SEPSen architecture by measuring the energy consumption for varying filtering ratios. These results were then compared with the cost-analysis for in-network complex event processing presented in Section 4.3, to provide proof of the assumptions made in the cost analysis.

Table 6.1: Parameters for energy consumption experiment

Setting	Value
No. of Nodes	25-200
Energy Model	Mica2 [77]
Sample Period	1024 ms
Simulation Time	60 virtual seconds

Figure 6.2 shows the total energy consumption for varying numbers of sensor nodes in various architectures according to the experiment settings shown in Table 6.1. It can be seen from the Figure 6.2 that the centralized architecture consumed more energy for all network sizes than either the simple or the context-aware modes of the SEPSen architecture. This is because the sensor nodes send events to the gateway node at regular intervals in the centralized architecture. The SEPSen architecture in simple mode reduces transmission of unnecessary events by filtering the events at the sensor node level. In addition, in the context-aware mode of the SEPSen architecture, the sensor nodes were also able to process complex events, resulting in better performance than others for all network sizes. In fact, energy conservation of about 46% against centralized and 15% against the SEPSen simple mode was observed in the context-aware mode of the SEPSen architecture.

An extension of Figure 6.2 is presented, analysing the energy consumed on different tasks (such as sensing, processing and communication) by the sensor nodes in both the architectures. Figure 6.3 presents the amount of energy consumed by

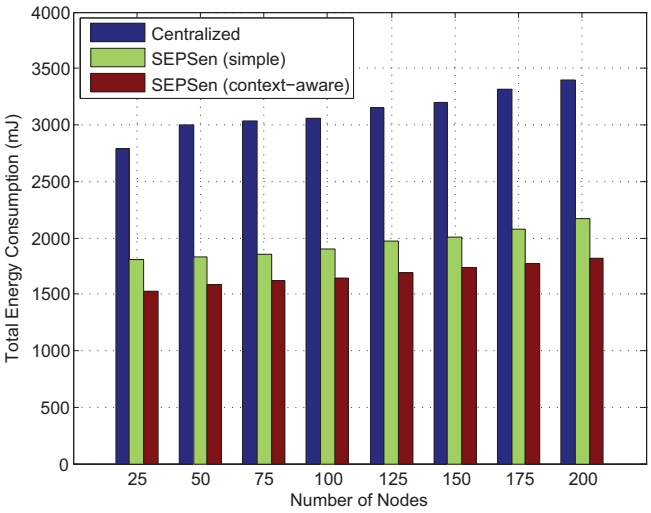


Figure 6.2: Total energy consumption for different approaches. *Simulations were run for 60 virtual seconds with one message per second for an increasing number of sensor nodes.*

the sensor nodes for the sensing task. In both the architectures, the sensor nodes sample their sensors at regular intervals for an equal amount of time, thus the energy consumed in sampling events in both the architectures is the same[1]. This reiterates the claim made on page 56 of Section 4.3.1 that the same amount of energy is spent on sensing events for all architectures and therefore sensing does not play a major role in the total energy consumption of the sensor nodes.

Figure 6.3 also presents the amount of energy consumed by the sensor nodes in processing the events. It is evident that the amount of energy spent on processing events is higher in the SEPSen architecture than the centralized architecture. As events are being processed at the sensor node level, the sensor nodes have to perform computations to detect events of interest before they are forwarded to the gateway nodes. This results in higher processing energy consumption compared to the centralized architecture. The sensor nodes in the centralized architecture only perform analog-to-digital (ADC) conversion of the sampled events and routing table lookup

[1] Each sensor node is equipped with a single sensor to observe environmental phenomena such as nitrogen, phosphorus and temperature.

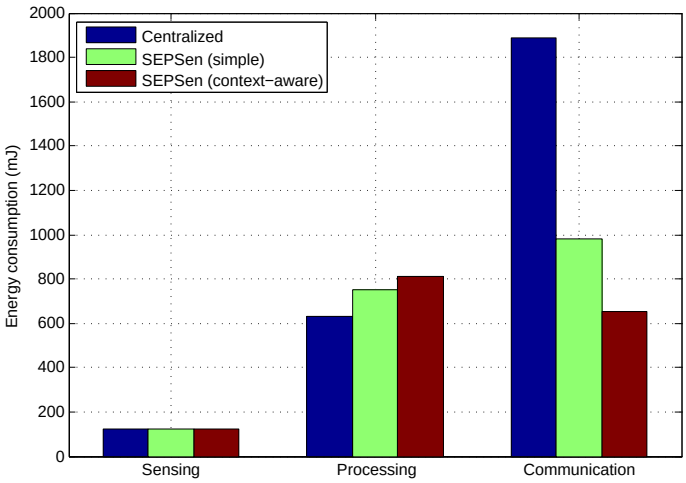


Figure 6.3: Energy consumption on different tasks at the sensor nodes for different approaches. *Simulations were run for 60 virtual seconds with one message per second over the 25-node topology.*

for forwarding sample data to the gateway nodes. However, the energy spent on processing tasks in the centralized architecture is not very much lower compared to the SEPSen architecture, because the sensor nodes often have to process received data through other sensor nodes to relay it in the sensor network until it reaches the gateway nodes. This frequent communication increases the energy expended on processing at the sensor nodes in the centralized architecture. It is also clear that SEPSen in context-aware mode spent more energy on computation than did SEPSen in simple mode. Again, this is attributed to the extra processing for detection of complex events compared to simple events, and the routing table lookup for communicating the events to the relevant sensor nodes in the sensor network. This also validates the assumption in the cost-analysis of the processing cost shown in Equation (4.4) in Section 4.3.1.

The next step is examination of the energy consumption at the sensor nodes for the communication task. Figure 6.3 shows the radio energy consumption for each simulated node in both the architectures. The sensor node in the centralized architecture consumed nearly twice as much radio energy as the SEPSen architecture

in both modes of operation. In both the SEPSen architecture modes, the number of transmissions was greatly reduced, resulting in reduced radio energy consumption at the sensor nodes. Since, in the context-aware mode of SEPSen, complex events are reported by the sensor nodes, this further reduces the communication of events and the energy spent on communication tasks. The cost analysis demonstrated a threshold (see Equation (4.7) at Page 59), indicating that for in-network processing to be advantageous for energy consumption, the costs for processing in additional nodes have to be less than or equal to the improvements in communication cost. The results shown in Figure 6.3 clearly show the improvement in communication costs and thus an overall reduced energy consumption at the sensor nodes in the SEPSen architecture.

In addition, Figure 6.4 evaluates the cost-benefit for in-network processing of events. For this experiment, the sensor nodes in the SEPSen are modelled to filter (by limiting the communication) a certain percentage of sampled and received events. As it can be seen from the figure, no filtering at the sensor nodes in SEPSen would incur higher energy consumption than the centralized architecture. This is because, the sensor nodes in SEPSen, are functioning with a higher computation cost and since communication is not restricted by filtering the events at the sensor nodes thus it incurs high energy cost at the sensor nodes. The benefit of processing events in-network appears when each of the sensor node filters around 7% of incoming events. This provides as a threshold for the improvement in the communication cost against the higher processing cost and resulted in better energy conservation at the sensor nodes.

Experiments on energy consumption at the sensor nodes, as shown in Figure 6.2, Figure 6.3 and Figure 6.4, clearly indicate that distributing the event detection tasks to the sensor nodes, as is done in SEPSen's complex mode, helps in conserving the sensor nodes. The reduced energy consumption can be attributed to the reduced radio energy consumption by the sensor nodes in the SEPSen architecture. Moreover,

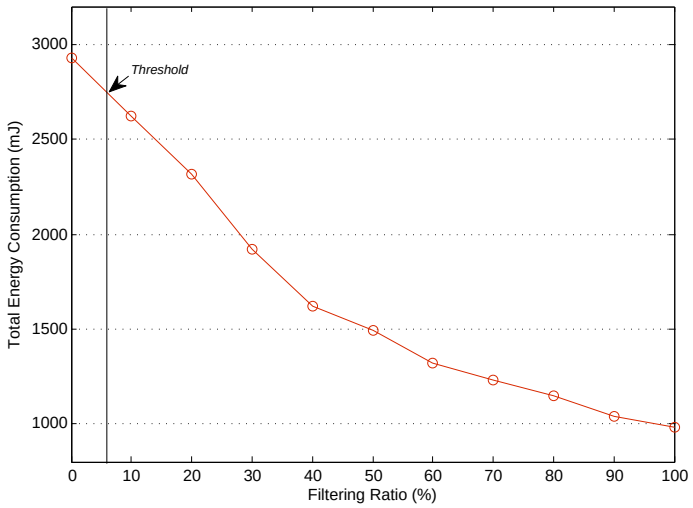


Figure 6.4: Energy consumption for different filtering ratios in SEPSen.

as the increase in energy consumption in processing the events is less than the energy consumed on communication, energy benefits are obtained.

6.3 Test 2: Overall Network Lifetime

The purpose of this test was to evaluate the effects of routing between the sensor nodes and gateway node on the sensor network lifetime. Network lifetimes are compared for the centralized and SEPSen architectures. In addition to the frequency of event generation, another important factor that affects network lifetime is the path selection for routing sensor events to the gateway nodes. Since the sensor nodes rely on the other nodes in the network to propagate their data to the gateway nodes, path selection plays an important role in the distribution of communication load at the sensor node level in the sensor network.

This experiment simulated a network of sensor nodes uniformly distributed in a 100m$\times$100m area. The number of sensors in the network was fixed to 100 nodes with one gateway node. The transmission range for all the sensors in the network was set to 10 meters. Each sensor node had an initial energy of 10 Joules and the gateway was located at (50, 50). The results shown in Figure 6.5 compare the network lifetime

Table 6.2: Parameters for network lifetime experiment

Setting	Value
No. of Nodes	100
Energy Model	Mica2 [77]
Sample Period	1024 ms
Initial Energy	10 Joules
Network Topology	100m$\times$100m area
Gateway Location	(50,50)
Transmission Range	10m

of the SEPSen context-aware routing described in Section 5.6 to that of shortest-path-first multi-hop (i.e. MIN-HOP) routing [3] provided in TinyOS. As described in [25], the network lifetime can be described by the time the first sensor node fails (in terms of remaining energy) or when a certain percentage of sensor nodes fail. The following experiment compared the network lifetime of various architectures by finding both when the first sensor node failed and also when a certain percentage i.e. 5% and 25% of the sensor nodes failed.

Figure 6.5 shows the network lifetime for the centralized and the SEPSen architecture in simple and context-aware modes. In the centralized architecture and the simple mode of SEPSen (i.e. SEPSen(simple)), the MIN-HOP routing protocol was used. For the SEPSen architecture in context-aware mode (i.e. SEPSen(context-aware)), the context-aware routing mentioned in Section 5.6 was applied. It is important to note that the sensor nodes in all the architectures initially use a broadcast mechanism to find routes to other sensor nodes and gateway nodes. The sensor nodes then use this information for routing (in a unicast manner) the sensor events to the gateway nodes. In the MIN-HOP routing protocol once the best route (i.e. the shortest path) is found, the sensor nodes then use it for communication towards the gateway nodes for the lifetime of the sensor nodes. Thus, no routes updates are performed. In contrast, in the context-aware SEPSen, paths are updated based on information on the alternate routes to other sensor nodes and the gateway nodes (see path re-calculation in Section 5.6.4). This path re-calculation is based on a threshold

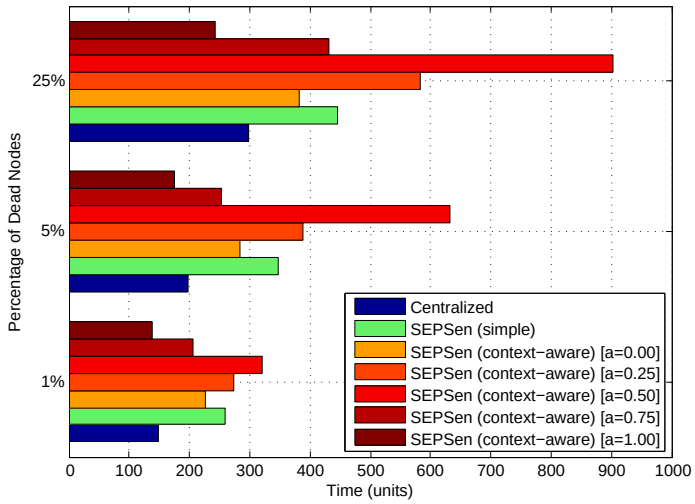


Figure 6.5: Network lifetime for different approaches. *Simulations were run until 1%, 5% and 25% of sensor nodes fail over the 100-node topology.*

value which is normalized according to the value of α. As explained earlier (see Page 85) the value of α can be adjusted between 0 and 1.

As can be seen from the results shown in Figure 6.5, the value of α has a significant effect on the network lifetime. Firstly, the two extreme cases of path re-calculation and its effects on the sensor nodes energy and hence the network lifetime are explained. The context-aware routing performs worse than the centralized and SEPSen(simple) when α is set to 1. This is due to the fact that initially all the sensor nodes have the same residual energies and since the threshold value for the alternate route is almost same as the best route value, frequent path re-calculation requests are sent by the sensor nodes. Similar results were obtained when path re-calculation was performed based on α=0.75. This overshadows any advantage that is obtained through filtering events at the sensor nodes.

Another case of poor performance of context-aware routing is when α is set to 0. In this case, the sensor nodes will never update the paths and will use the initially obtained best route for routing the sensed data to the sensor nodes and the gateway nodes. As the initial route (i.e. best route) is selected based on the maximum available energy along the path, the sensor data may have to traverse more hops to reach the

destination. Thus, it performs worse than SEPSen(simple), which uses MIN-HOP routing. The better performance in this case against the centralized architecture (also using MIN-HOP routing) is only because of event filtering at the sensor nodes.

The network lifetime is considerably extended by using context-aware routing when the value of α is between 0.25 and 0.75. When α is set to 0.25, it performs better than the MIN-HOP routing used in the centralized and SEPSen(simple) architecture. This is because of the route updates based on the path re-calculation mechanism. However, since the path updates take place after a considerable delay, it only marginally improves the network lifetime by altering the routes for balanced energy depletion. The network lifetime is greatly increased when α is set to 0.50. In fact, the network lifetime is extended by a factor of 2 for first sensor node failure and a factor of 3 for 25% of sensor node failure through context-aware routing. This provides a balanced approach, avoiding both too-frequent updating of the routes and significant delays in updating. Experiments on the distribution of sensor nodes' residual energies (shown in Figure 6.6) further highlighted this advantage.

Figure 6.6 shows the distribution of the sensor nodes' remaining energy for all the approaches. This is a qualitative metric, rather than quantitative one. This metric measures how evenly the energy dissipation is distributed. The distribution of energy consumption at the sensor nodes considerably affects the lifetime of the overall sensor network (see Figure 6.5). In this test, since MIN-HOP routing was used in both the centralized and SEPSen(simple) architecture, similar results were obtained for network energy distribution (see Figure 6.6a). The differences in those architectures have been highlighted earlier in Figure 6.5: in SEPSen (simple), events are filtered, thus less energy is consumed at the sensor nodes and the network lifetime is extended.

However, the distribution of energy is still variable for the sensor nodes and the sensor nodes that are selected to route the events to the gateway nodes incur high energy consumption. SEPSen context-aware routing, shown in Figure 6.6b, with no

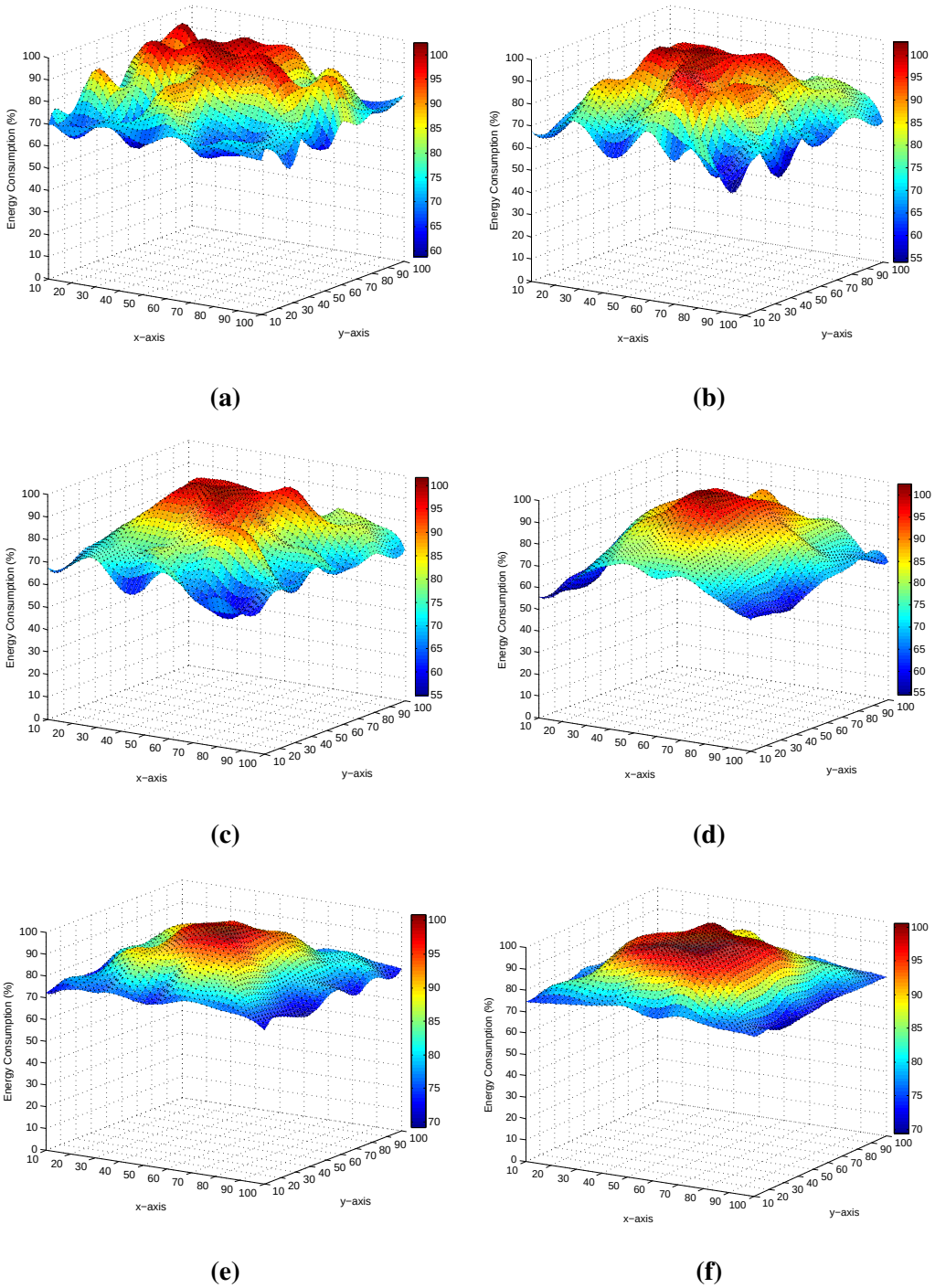
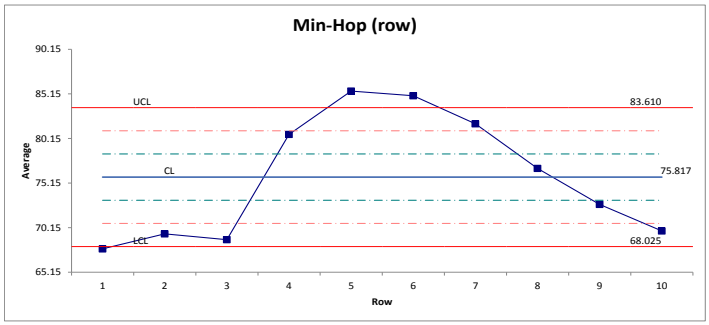
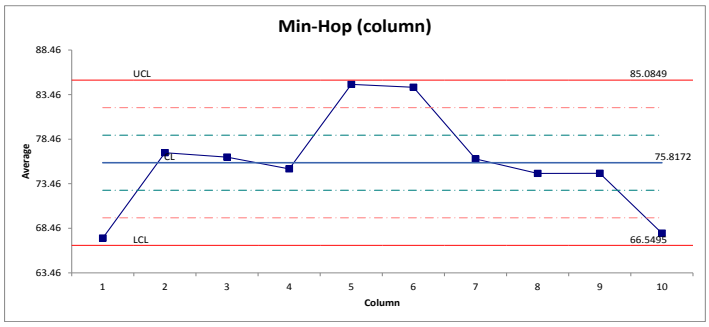
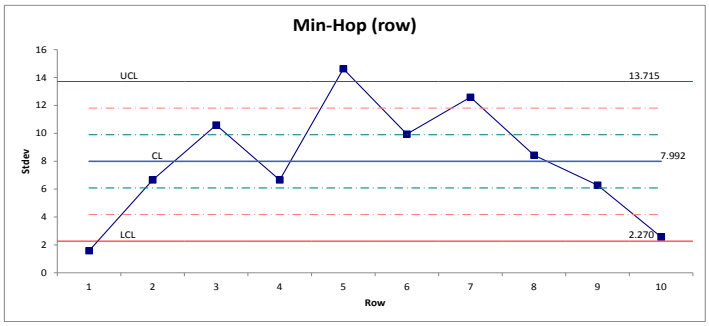


Figure 6.6: Distribution of sensor node energy consumption for (a) Centralized and SEPSen (simple) using the MIN-HOP routing algorithm and (b to f) SEPSen (context-aware) using the context-aware routing algorithm for $\alpha = 0.00$, $\alpha = 0.25$, $\alpha = 0.50$, $\alpha = 0.75$, and $\alpha = 1.00$ respectively.







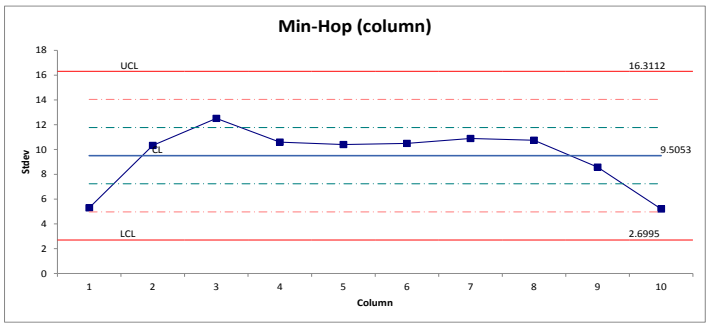
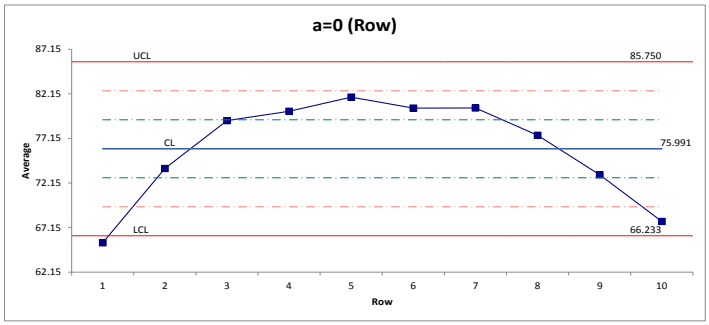
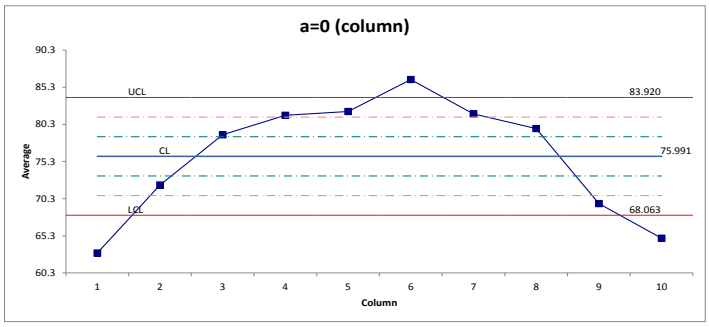
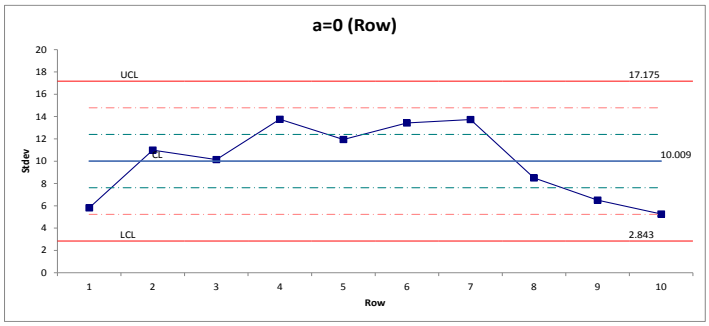


Figure 6.7: 2D analysis for min-hop algorithm







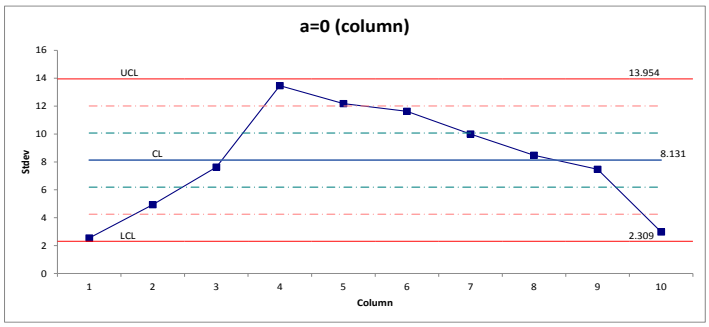
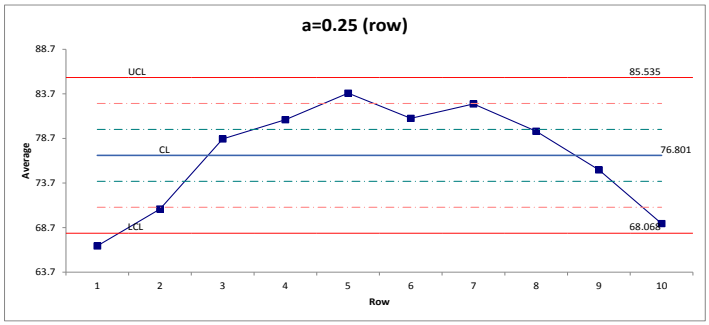
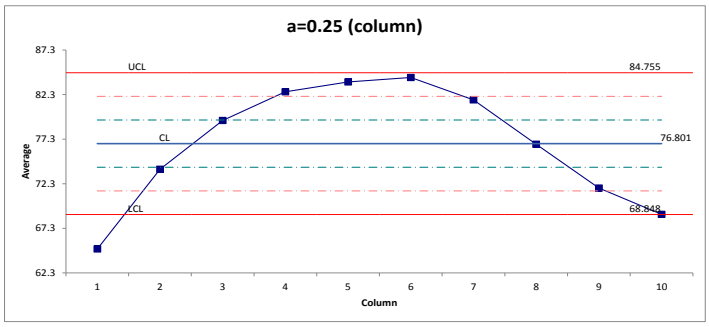
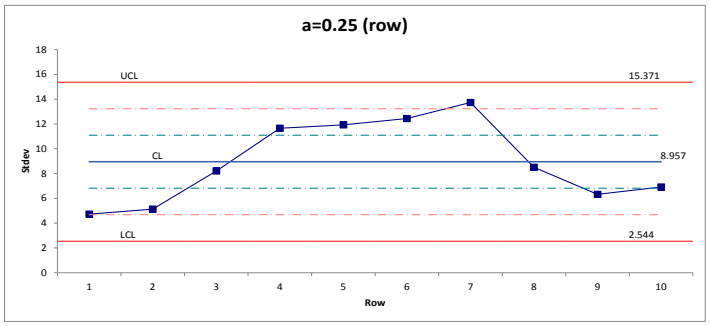


Figure 6.8: 2D analysis for context-aware algorithm ($\alpha = 0$)







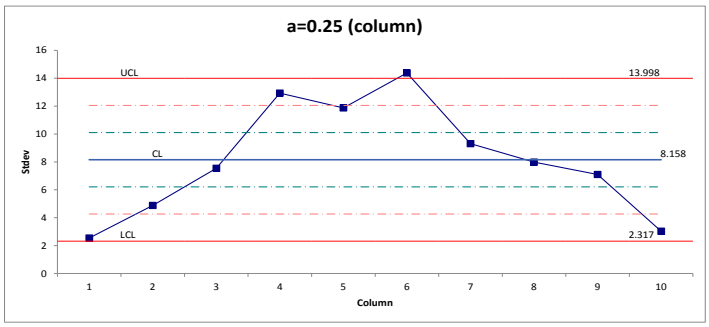
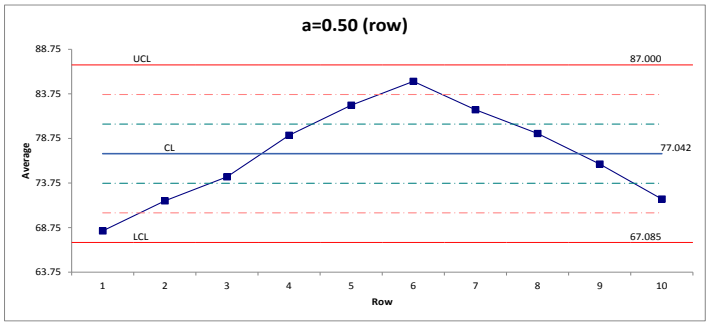
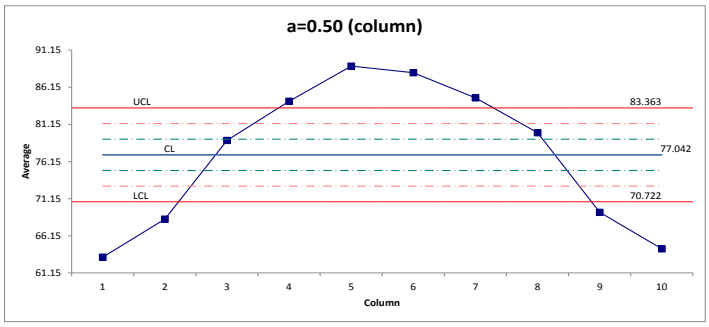
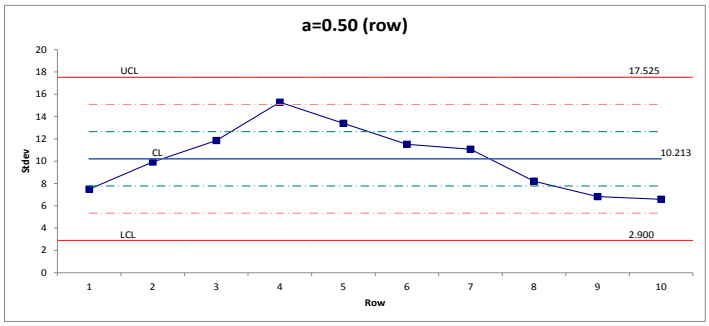


Figure 6.9: 2D analysis for context-aware algorithm (α = 0.25)







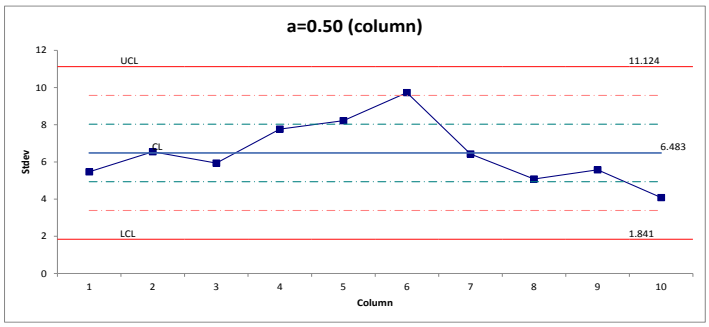
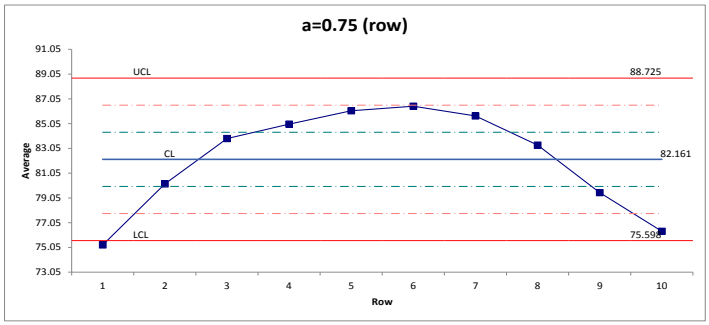
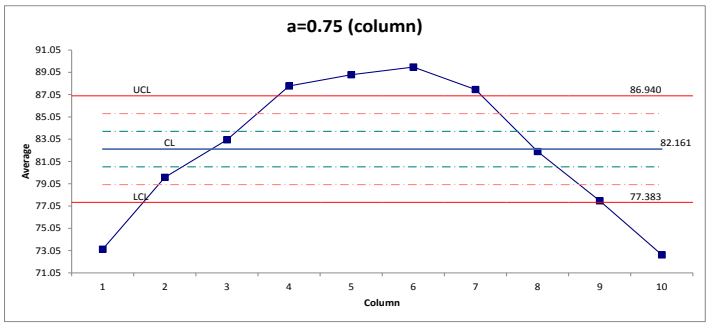
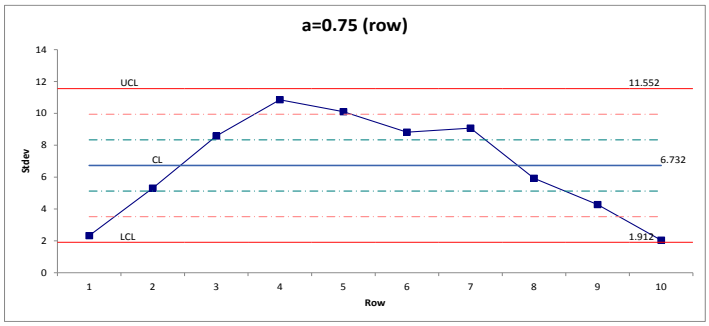


Figure 6.10: 2D analysis for context-aware algorithm (α = 0.5)







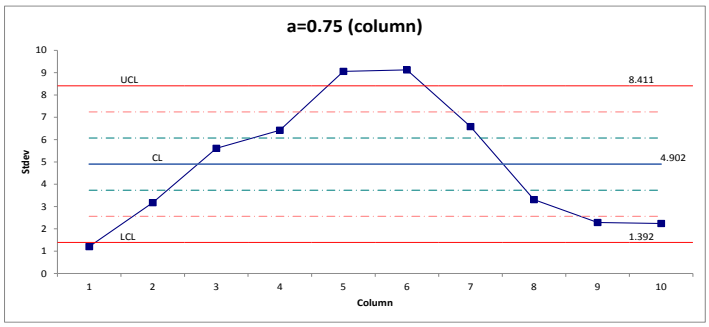
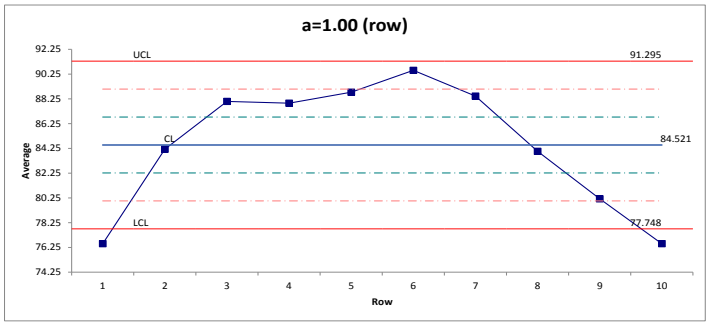
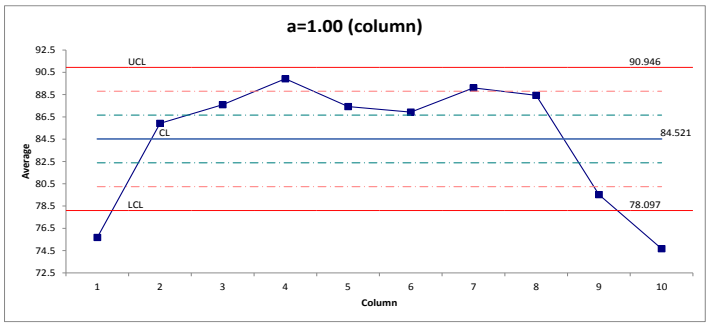
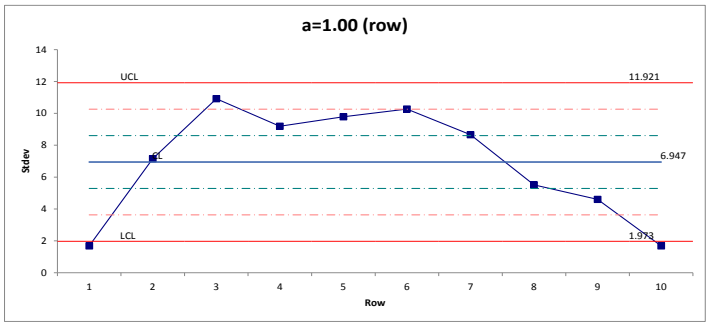


Figure 6.11: 2D analysis for context-aware algorithm (α = 0.75)







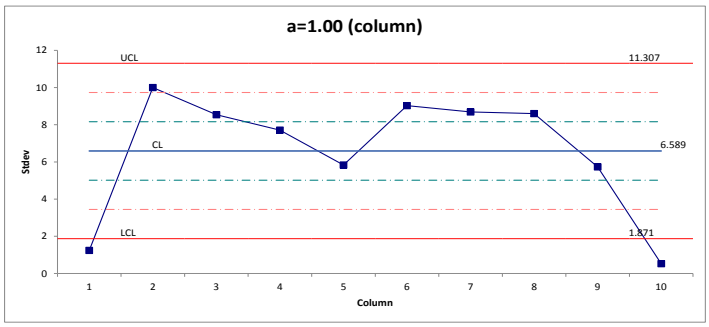


Figure 6.12: 2D analysis for context-aware algorithm (α = 1)

route updates (i.e. when α is set to 0), provides similar results as shown in Figure 6.6a but at the cost of possibly higher hop counts to the destination. Also, the energy distribution shown in Figure 6.6c shows a low energy balancing performance but relatively better performance than the results shown in Figure 6.6b. Moreover, it can be seen in Figure 6.6d, that the context-aware routing balances the routing load very well. This benefits the overall network lifetime, as shown earlier in Figure 6.5. However, Figures 6.6e to 6.6f show that while frequent path re-calculations improves the distribution of energy depletion, it incurs high overhead at the sensor nodes and thus reduces the network lifetime. This has been highlighted earlier in Figure 6.5.

Figures 6.7 – 6.12 show the quantitative analysis of the six 3D energy plots shown in Figure 6.6. The plots are analysed based on the energy consumption at the sensor nodes along the x-axis and y-axis of the underlying grid. Figure 6.7 shows the average energy consumption along the x-axis and the y-axis for the Min-Hop routing algorithm. It can be seen that the average energy consumption for Min-Hop routing algorithm is 75.817. In addition, it also shows the average variance in the energy consumption of the sensor nodes at different locations. It can be seen from the above stated figures that the average variation in the energy consumption for Min-Hop is relatively high at 7.992 and 9.5053 for x-axis and y-axis of the underlying grid, respectively.

Our context-aware routing shows similar behaviour (see Figure 6.8) when alpha is set to 0 as the variance of energy consumption amongst the sensor nodes along the x-axis and y-axis is found to be at 10.009 and 8.131 respectively; but with a slightly higher average energy consumption of 75.991. The context-aware routing shows improvement in reducing the variance of energy consumption at the sensor nodes when the alpha is set to values above 0.25. A gradual improvement is observed when alpha is set to 0.25 (see Figure 6.9), where the variance for sensor nodes energy consumption at x-axis and y-axis is 8.957 and 8.158, respectively. Similar results were observed when alpha value is set to 0.5, as the variance along the axis and y-axis

was found to be 10.213 and 6.483, respectively (see Figure 6.10). A considerable improvement is observed when alpha is between 0.75 and 1, when the variance is as low as 4.902 (see Figure 6.11) and 6.589 (see Figure 6.12). However, in such cases, the sensor nodes consumed more energy due to frequent updates and this reflects in the average energy consumption of 82.161 (see Figure 6.11) and 84.521 (see Figure 6.12) for alpha values of 0.75 and 1 respectively.

Experiments on overall network lifetime (shown in the Figure 6.5) and distribution of remaining sensor nodes energies (shown in Figures 6.6b to 6.6f) clearly show that if path re-calculation is set properly, the context-aware routing approach can distribute the network energy uniformly and this can result in an extended lifetime for the network.

6.4 Test 3: Memory Usage

Sensor nodes have limited storage capacity. In fact, Mica2 sensor nodes have only few kilobytes (i.e. 4 KiB) of working memory (i.e. data size) available to them. Thus, the memory consumption for an application must be within the limits of available data size.

In SEPSen, memory is consumed in storing the: i) ontology fragment concept that is used for annotation of sensor data; ii) rules provided by the user for filtering the sensed data and building pattern networks of matching rules against the sensor data; iii) facts generated and shared by the sensor nodes; and iv) routing table entries in the sensor nodes for communication purposes. In the memory usage test, the memory usage of a single SEPSen node with different configuration parameters is analysed as shown in Table 6.3.

Figure 6.13 shows the memory usage for a SEPSen sensor node with different numbers of rules in it. The number of rules was increased from 5 to 25 for the ontology fragment consisting of 10 classes, 7 properties and 50 triplets. The network size was set to 50 nodes. It can be seen that the application can be run on a sensor

Table 6.3: Parameters for memory usage experiments

Figure No.	No. of Rules	No. of Facts	Network Size	Ontology Size
6.13	—	50	50	10/7
6.14	5	—	50	10/7
6.15	5	50	—	10/7
6.16	5	50	50	—

node with varying numbers of conditions for up to 10 rules. However, as the number of rules increases beyond 10, in most cases, all of the memory resource is consumed, thus blocking other processes from being performed.

In the experiment whose results are shown in Figure 6.14, the memory usage was analysed for varying numbers of facts (i.e. triplets) with the network configuration as shown in Table 6.3. It was observed that the application can be run on a sensor node

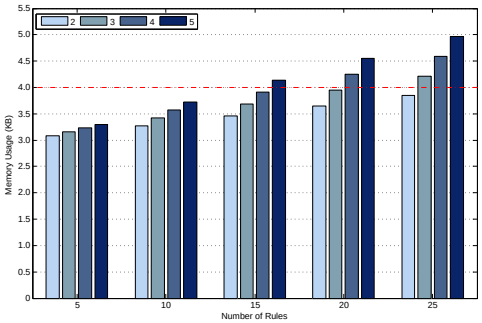


Figure 6.13: Memory usage for varying number of rules with conjunction of 2, 3, 4 and 5 conditions.

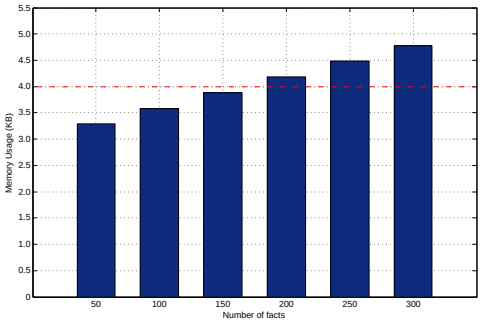


Figure 6.14: Memory usage for varying number of facts.

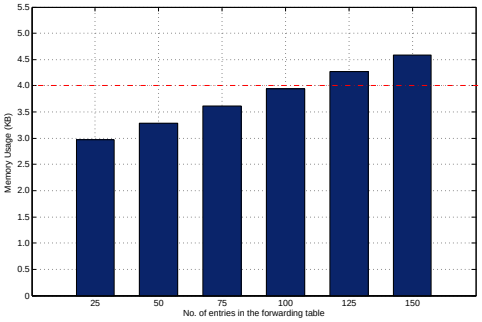


Figure 6.15: Memory usage for different network sizes.

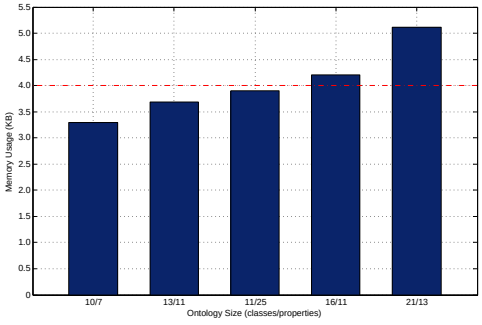


Figure 6.16: Memory usage for different ontology sizes.

with varying numbers of facts until it reaches 150 facts per sensor node. Similarly, the experiments on network sizes shown in Figure 6.15 provide the limitations of sensor nodes for the number of entries stored in their routing tables. The nodes can store routing entries for a network size of up 100. Any number of facts or routing entries beyond these limits blocks the sensors' operations.

Figure 6.16 shows memory usage for the different ontology sizes with the network configuration as shown in Table 6.3. It can be seen that the application can be run on a sensor node with varying numbers of classes and properties, up to 13 and 25 respectively. However, as the number of classes and properties increases beyond 13 and 25 respectively, all the memory resource is consumed, thus blocking other processes from being performed.

The experiments on memory usage indicate that the rules and ontology fragments consume major chunks of sensor nodes' data memory. The rules require memory for pattern network construction and matching of events to the rules: higher numbers of rules cannot be supported by the sensor nodes. However, this is not a major limiting factor for monitoring water quality parameters as the event filters mentioned in [11] and [24] for the sensor nodes are well within the limits of the sensor nodes' storage requirements. The ontology fragments also consumed a major chunk of data memory, due to the fact that sensor nodes require space for annotating and processing the sensor data. This might be a limiting factor as some domain ontologies might not provide enough flexibility in terms of fragmentation that could be kept in the sensor nodes. However, for the work described in this thesis, the ontology fragments fit well within the limits of sensor node storage.

The experiments also showed the limits on the number of facts the sensor node can store and the network size that it can support. The facts in SEPSen are updated as and when they arrive and old entries on the same sensor nodes are discarded. As already noted, only a limited number of rules are supported by the sensor nodes, thus only the facts related to those rule conditions are kept. This helps in keeping the

number of facts within the limits of the sensor nodes. For support of large networks, most real world deployments (such as SmartCoast [68] and GlacsWeb project [62]) also employ a limited number of sensor nodes to monitor the environment. Thus, this limit on the network size is also reasonable for real-world deployments of the sensor network.

6.5 Test 4: Latency

The latency tests aimed to analyse the time the new architecture took for various activities such as ontology parsing, knowledge base loading and rule parsing for pattern network construction. Since some applications have strict latency requirements and are highly time critical, such as surveillance systems, it is necessary to know the time required to process data locally at the sensor nodes. The experiment used three different test scenarios with parameters set as shown in Table 6.4.

The results of the tests, shown in Figure 6.17 indicate that ontology parsing consumes a minimal time compared to knowledge base loading and pattern network construction for reasoning at the sensor nodes. It was observed that the reasoning process is the most time intensive as it has to process the rules for creating a pattern network to check the rules against the loaded knowledge base facts. It must be also noted that all these activities are done once at start-up and later execution for reasoning does not require ontology parsing or rule processing. However, any update of the rules or the ontology fragments will require the same time limits for these activities and thus, for later executions, they can induce latency in overall processing events at the sensor nodes.

Table 6.4: Parameters for processing time tests

Test No.	No. of Rules	No. of Facts	Ontology Size
1	3	25	9/5
2	5	50	9/5
3	7	50	10/7

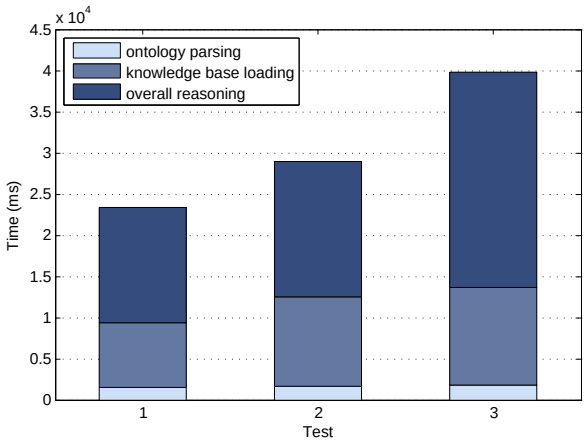


Figure 6.17: Ontology parsing, Knowledge Base loading and Overall reasoning time.

6.6 Summary

This chapter presented a performance analysis of the SEPSen architecture as compared to traditional centralized architecture. The analysis was conducted to observe the energy benefits of processing events at the sensor node level and also to highlight the processing capabilities of the SEPSen architecture. It answers the third research question of *"how much energy benefit could be gained by performing complex event detection tasks at the sensor node level?"*

SEPSen works on a flat network architecture where it forms a multi-hop route from each node to neighbouring nodes and the gateway. Each node performs periodic estimations of the routing cost to its neighbours and attempts to select a neighbour node that will maximize the probability of its messages reaching the gateway by selecting energy-efficient paths. Each node samples its sensor and generates a message once events are detected. Therefore, relay nodes will relay both the messages of their neighbours as well as their own messages.

This topology has been used in several real-world deployments, such as [33, 91, 62]. The simulation results shown in this thesis were performed for varying number of sensor nodes (network sizes) and hop counts. We selected realistic parameter settings according to various real-world deployments with network sizes ranging

from few sensor nodes (e.g., [91]) to few hundred sensor nodes (e.g., [33]). We acknowledge that the radio range in most of the real-world applications (e.g.,[62, 15]) would be higher than the simulation radio parameters, the increase in the radio range would result in higher transmission cost at the sensor nodes (see [32, 12]). However, this would be same for both the centralized and the SEPSen architecture and therefore does not invalidate the results.

Moreover, we performed our simulation tests on PowerTOSSIM. PowerTOSSIM is demonstrated to accurately estimate the energy consumption of the simulated sensor node, with an average error of just 4.7% compared to the actual sensor node. Some of this difference between simulated energy and measured energy can be attributed to voltage fluctuations, noise and rounding error in the experimental setup [77]. We believe it therefore provides a good estimate of the overall energy consumption for the actual deployment of the sensor network in an environment.

Four tests were performed. The first test established the energy benefits of our architecture in terms of total energy consumption. It also measured the energy consumption of sensing, processing and communication tasks at the sensor nodes. The results confirmed that by distributing the processing tasks to the sensor nodes, as is done in the SEPSen architecture, energy consumption can be significantly reduced at the sensor nodes in a WSN. The threshold for effectiveness of SEPSen in terms of energy benefits was also investigated under a controlled communication environment. It was observed that the energy benefits in SEPSen can be achieved only if they filter and limit the communication of events by 7% against the centralized architecture.

The second test measured the overall network lifetime for both centralized and SEPSen architectures in terms of number of dead sensor nodes in the network. A significant improvement in the overall network lifetime was observed in the SEPSen architecture due to energy-aware routing, as compared to traditional shortest-path-first multi-hop routing. However, in certain cases, the energy-aware routing in SEPSen does not perform well. This is due to the high overhead produced by

frequent path re-calculations at the sensor nodes.

The third test assessed the storage limitations of sensor nodes in the SEPSen architecture. Analysis was conducted on storage requirements for storing rules, facts, ontology concepts and routing table entries. The experiments performed in this test presented different configuration setups that can be used in the SEPSen architecture within the limited storage capacity of the sensor nodes. For the example application of this thesis (i.e. water quality management), the presented storage consumption is reasonable and well within the limits of the sensor nodes.

The fourth test calculated the amount of time required for various processing tasks (i.e. ontology parsing, knowledge base loading and overall reasoning) in the SEPSen architecture. The results show that overall reasoning produces considerable latency as compared to ontology parsing and knowledge base loading. However, for most of the environmental applications (such as water or air quality monitoring) this is not an issue since data transmission rates can be delayed or reduced in order to improve overall network lifetime. This might be a problem for applications (such as surveillance) that require operations to be performed within strict latency bounds.

Overall, the findings from the experiments demonstrate that SEPSen achieves its objective (see Section 1.3) of utilizing sensor nodes' processing capabilities for energy conservation in a heterogeneous WSN. The sensor nodes stored and processed heterogeneous events within its limited storage capacity and reasonable latency bounds for the example application area respectively. Moreover, the sensor nodes filtered the events which resulted in the reduced energy consumption at the sensor nodes. This reduction in the communication of unnecessary events along with the context-aware routing prolonged the overall networks' lifetime.

The next chapter concludes the research. It presents a summary of the thesis. The contributions and limitations of the research work are discussed and future research directions are suggested.

Chapter 7

Conclusions and Future Work

This thesis proposed a new method of processing heterogeneous sensor data within WSNs. Its objective was to minimise the energy consumption of the severally constrained sensor nodes. It explored the following central hypothesis (Section 1.3):

> *If the sensor nodes: i) semantically annotate the sensed data, ii) collaborate with the surrounding sensor nodes, and iii) perform filtering and integration of events on the gathered knowledge, then they will be able to detect complex events locally in heterogeneous sensor networks, thereby resulting in reduced overall energy consumption of the WSN.*

The research reported in this thesis confirms this hypothesis and shows that the combination of the three factors of semantic annotation, context-awareness and filtering of events at the sensor nodes reduces the energy consumption in heterogeneous WSNs.

This chapter is structured as follows. Section 7.1 gives a summary of this thesis and describes the steps that were undertaken to address the research objective and to answer the research questions derived from the objective. Section 7.2 outlines the contributions of this thesis. Section 7.3 discusses the limitations of the work described in the thesis. Section 7.4 gives directions for future work and the chapter concludes with a summary of the overall outcome of this research in Section 7.5.

7.1 Summary

This thesis analysed research on the applications and constraints of wireless sensor networks to derive requirements for the design of energy-efficient event processing in a heterogeneous WSN (Chapter 2). The operation of WSNs in environmental monitoring, surveillance and health care applications was surveyed. Since a motivating force for this thesis was water quality monitoring, the demands for successful operation of a WSN for environmental applications were presented. The constraints of WSNs with respect to those application demands were highlighted, along with a description of the existing techniques for mitigating those constraints in WSNs. Three major requirements were defined that are crucial to energy-efficient event processing in a heterogeneous sensor network: semantic annotation, filtering and context-aware sharing of events at the sensor nodes.

Analysis of approaches addressing these three requirements in related areas revealed the strengths of such approaches that could be built upon, as well as shortcomings that needed to be avoided (Chapter 3). Three main research areas were covered in the analysis i.e. in-network data processing, semantic data integration and context-aware sensor networks. A gap was identified in the combination of these aspects, as none of the approaches met all of the requirements for annotation, filtering and context-aware communication of events at the sensor node level.

A new conceptual design was proposed for energy-efficient event processing in a heterogeneous wireless sensor network that takes all these requirements into account (Chapter 4). It bridges the gap identified in existing approaches and incorporates all the aspects (i.e. semantic annotation, filtering and context-aware communication) needed for event detection at the sensor node level. The design for the sensor node architecture (i.e. SEPSen node) consists of five components: receiver, semantic annotator, knowledge base, rule engine and transmitter. The receiver component is designed to collect sensed or incoming data from the sensor nodes. The sensed data is then provided to the semantic annotator component for

annotation of the raw sensed data. The annotation is based on the ontology fragments available to the sensor nodes. The annotated data from the sensor nodes (either sensed or received through other nodes) along with the specified rules, are placed into the knowledge base component of the SEPSen node. The rule engine gathers the annotated data (facts) from the knowledge base and performs matching of these facts against the specified rules. Whenever the facts match the criteria of the specified rule, they are shared with other sensor nodes or forwarded to the gateway nodes by the transmitter component of the SEPSen node. The design fulfils the recommendations by providing support for semantic annotation, in-network event detection and context-awareness for communicating at the sensor nodes within a heterogeneous WSN.

The SEPSen architecture was implemented to achieve energy-efficient event processing in a WSN (Chapter 5). All three requirements of semantic annotation, filtering and context-aware sharing were dealt with in the implementation. An existing ontology (TWC-SWQP [88, 89]) was extended to be used in the sensor nodes for annotation and processing of the sensor data. The prototype implementation (i.e. SEPSen) divides the overall energy-efficient event detection into three processes: annotation, event detection and communication. During the annotation process, the sensed data is converted into RDF triplets (i.e. facts). These facts and rules from the knowledge base component are then processed for event detection by a pattern-matcher which filters the events at the sensor node level. The pattern-matching is performed by first constructing a network of patterns based on the specified rules. Facts are then matched to the patterns and when a match is found, it is shared with other sensor nodes or forwarded to the gateway nodes. The sharing of events is performed by using context-aware communication. The context-aware communication in SEPSen is based on both the data and the energy context. It enables sharing of the sensor nodes' knowledge bases for collaboration in an energy-efficient manner. The SEPSen implementation provides efficient filtering at the sensor node level and effective routing using data and energy-related context information.

The implementation of the SEPSen architecture was evaluated by performing various experiments in a simulation environment (Chapter 6). The experiments evaluated SEPSen's effectiveness in terms of energy consumption at the sensor node level and also its effects on the overall network lifetime. The results from the experiments clearly indicate that distributing the event detection task to the sensor nodes helps in conserving the sensor nodes' energy. The network lifetime was also considerably extended through context-aware routing when optimized properly. The evaluation also tested adverse cases for cost-benefit threshold analysis of the SEPSen, and discussed restrictions of the SEPSen system in terms of memory usage and latency introduced due to processing events at the sensor nodes.

7.2 Contributions

This section summarizes the contributions made by this thesis to the research area of in-network event processing in heterogeneous sensor networks.

Requirements for energy efficient event processing in heterogeneous WSN. The literature review of the research into the applications of the wireless sensor network (WSN) and its constraints contributed a thorough understanding and clear definition of the problems faced in environmental applications of WSNs. This is important because approaches to related problems in the past have neglected this perspective. The review yielded three requirements (i.e. semantic annotation, filtering at the sensor nodes, and context-aware communication) for the design of energy-efficient heterogeneous WSN. These requirements clarify the choices to be made when designing a WSN and are helpful in conserving sensor nodes' energy.

Synthesis of work in related domains. The analysis of existing approaches enables future research in the area of event processing in heterogeneous WSN to build on the strengths and avoid the shortcomings of existing work. The

approaches selected for analysis are either designed for homogeneous WSNs or perform processing tasks at the powerful gateway nodes. No existing approach combines energy and adaptation in routing with processing at the sensor nodes. Most approaches thus suffer from a narrow focus on the technical issues.

Conceptual design for an energy efficient event processing in heterogeneous WSN. The conceptual design of an architecture that supports event processing at the sensor node level for energy efficiency in sensor networks applied the understanding of the problem and the analysis of the related work in a new solution. The design integrated techniques such as semantic annotation, rule-based filtration and context-awareness at the sensor nodes to enable processing of sensor events at their own level. A version of this has been presented earlier in [46] and [47].

Prototype implementation of the design. The implementation of the conceptual design, SEPSen, further clarified the design by providing a realisation of the selected aspects of annotation, event processing and communication. To annotate the sensed data, an existing ontology was extended and used in the sensor nodes. This enabled the sensor nodes to annotate the sensed readings to be used for event detection in heterogeneous sensor networks. For event processing, a well-known pattern matching algorithm, Rete, was adapted to filter and match the interests of subscribers to that of the publishers. The context-aware communication was introduced to efficiently disseminate the events between the sensor nodes. This enabled the sensor network to uniformly distribute the energy consumption among the sensor nodes. A version of this has been presented earlier in [46].

Theoretical cost analysis of the architecture. The cost model for complex event processing in a heterogeneous sensor network described in this thesis shows how to evaluate a WSN designed for in-network processing of heterogeneous

events. It provides a threshold of circumstances under which the processing of events at the sensor level is beneficial for energy conservation in the sensor networks. This cost model can be used in evaluation of other approaches targeting in-network processing in heterogeneous WSNs. A version of this has been presented earlier in [48].

Simulation-based analysis of the architecture. Experiments on the simulation-based environment of the architecture evaluated the performance and effectiveness of the design and implementation of the SEPSen. The experiments showed that the SEPSen approach performs significantly better in terms of energy benefits in WSNs than the traditional approaches. They also showed the limits of the architecture in the form of memory and latency bounds. The findings from the analysis can be used as a guidance and reference for the future experimental work.

7.3 Limitations

This section discusses the choices made in the direction of research presented in this thesis that led to a number of limitations.

Design Choices. The design choices were mainly determined by the sensor nodes' limited storage capacity. The design focused on support for detection of events in an instantaneous manner. This is due to the fact that history-sensitive event detection requires sensor nodes to provide extra storage for the detection of historical events. This design therefore limited the application user to monitoring events without taking into account the temporal aspects of the monitored events.

Implementation Choices. The implementation of SEPSen includes support for semantic annotation, rule processing and context-aware communication. Of these factors, semantic annotation was found to be effective, but open questions

remain regarding the automatic infusion of ontology fragments into the sensor nodes. Rule processing was found to be most effective in conserving the sensor nodes' energy by limiting the communication of irrelevant events and detecting both simple and complex events at the sensor node level in a heterogeneous sensor network, but more expressiveness in rules and event processing is desirable. Sharing the knowledge using context-aware routing was found to be effective in prolonging the network lifetime, however, in certain situations the mechanism for fair distribution of energy depletion incurs high energy overheads and thus it decreases the network lifetime.

Evaluation Choices. Implementation of the SEPSen prototype was done on an extension of the TOSSIM simulator, called PowerTOSSIM. It is designed to emulate the behaviour of applications based on the TinyOS operating system. Thus sensor nodes based on the MICA-family are the only sensor node platforms that it can simulate. Moreover, as the architecture is evaluated in a simulation-based environment in perfect conditions, many real-world deployment issues such as degradation in performance of the sensor nodes and anomalous sensor data are not considered.

7.4 Future Work

There are a number of promising points that could be pursued to address the limitations and extend this work further. This section describes the directions in which this research can be extended.

Support for history-sensitive event detection. An important aspect of this work that needs further investigation is the expansion of the SEPSen architecture to accommodate history-sensitive event detection both locally and in a distributed manner. While the lightweight approach demonstrates the general feasibility of event detection in an instantaneous manner, many applications require

support for history-sensitive event detection. The challenge in supporting history-sensitive events is the allocation of additional memory for data samples involved in the recognition of events. Such an extension would therefore require a support for managing limited memory resources of the sensor nodes.

Automatic Ontology Infusion and Rule update. The current implementation of the architecture assumes that the ontology fragments have been infused into each sensor node that can be used for annotating sensor data. Future work could focus on the automatic infusion of ontology fragments into the sensor nodes for classifying events. Also, since SEPSen does not support automatic rule changing during run-time, future work could examine the implementation of a small scale rule adaptation at run-time for the sensor devices. The associated challenges with remote automatic updates are the management of increased latency and energy consumption at the sensor nodes.

Improvement in reasoning capabilities. While the prototype implementation of SEPSen provides the capability to detect events at the sensor nodes, there is an opportunity to provide support for more comparison operators in order to increase the expressive power of the reasoner. This may not provide any significant contribution to the design and implementation of the architecture, but such an extension is likely to have an impact on the performance and effectiveness of the architecture.

Improvement in energy-aware routing. The energy-aware routing in SEPSen provides the capability to alter routes based on the energy awareness, at the sensor nodes, of different routes. However, the performance evaluation showed that in certain cases, it performed worse than the traditional shortest-path-first multi-hop (MIN-HOP) routing. This performance degradation is mainly attributed to the path re-calculation process. Future work could aim to incorporate more advanced means of path re-calculation in order to assess the

energy state of the sensor nodes and make necessary amendments in the path re-calculation process.

Other application domains. The experiments suggest that this approach is on the right track, however, the results are specific to one domain i.e. water quality management. Further work could be conducted in other domains (such as monitoring air pollution or rain-forests) to determine the actual capability of the system in different scenarios and conditions. Such an extension would require domain specific ontologies and rules for filtering events at the sensor node level in a heterogeneous WSN.

7.5 Final Words

The key contribution of this research in the area of WSNs is to detect heterogeneous sensor events at the sensor node level. An architecture was introduced to provide support for semantic annotation, filtering and context-aware communication at the sensor node level. The sensed data is annotated, filtered and communicated in a context-aware manner to detect events of interest at the sensor nodes. This results in energy benefits at the sensor nodes and prolonged network lifetime.

This work is an initial step towards the realization of processing event at the sensor nodes for energy benefits. A number of steps for extending this work have been suggested.

Appendix A

Water Quality Ontology

This appendix list the Water Quality Ontology (WQO) that is referred to in the SEPSen design and implementation. The ontology defines vocabulary to specify water pollutants and sensor observations related to the water quality monitoring. All OWL and RDF is presented in Notation 3 format[1].

```
@prefix body: <http://sweet.jpl.nasa.gov/2.1/realmHydroBody.owl#> .
@prefix chem: <http://sweet.jpl.nasa.gov/2.1/matr.owl#> .
@prefix comp: <http://sweet.jpl.nasa.gov/2.1/matrCompound.owl#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix elem: <http://sweet.jpl.nasa.gov/2.1/matrElement.owl#> .
@prefix epa: <http://tw2.tw.rpi.edu/zhengj3/owl/epa.owl#> .
@prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#> .
@prefix wqo: <http://www.co-ode.org/ontologies/wqo.owl#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix pmlp: <http://inferenceweb.stanford.edu/2006/06/pml-
   provenance.owl#> .
@prefix protege: <http://protege.stanford.edu/plugins/owl/protege#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix repr: <http://sweet.jpl.nasa.gov/2.1/repr.owl#> .
@prefix ssn: <http://purl.oclc.org/NET/ssnx/ssn#> .
@prefix time: <http://www.w3.org/2006/time#> .
```

[1] http://www.w3.org/TeamSubmission/n3/

```
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix xsp: <http://www.owl-ontologies.com/2005/08/07/xsp.owl#> .

<http://www.co-ode.org/ontologies/wqo.owl#> a owl:Ontology ;
   rdfs:comment "Proposed upper ontology for WQO"^^xsd:string ;

body:BodyOfWater a owl:Class ;
   rdfs:subClassOf wqo:SpaceTimeRegion .

geo:Site a owl:Class ;
   rdfs:comment "Potential uses of a particular site"^^xsd:string ;
   rdfs:subClassOf wqo:SpaceTimeRegion .

epa:WaterMeasurementSite a owl:Class ;
   rdfs:comment "A site where water quality was measured."
       ^^xsd:string ;
   rdfs:subClassOf [ a owl:Restriction ;
          owl:allValuesFrom epa:WaterMeasurement ;
          owl:onProperty epa:hasMeasurement ],
      body:BodyOfWater,
      geo:Site .

epa:hasMeasurement a owl:ObjectProperty ;
   rdfs:comment "Links an object to a measurement taken of it."
       ^^xsd:string ;
   rdfs:domain epa:WaterMeasurementSite ;
   rdfs:range epa:WaterMeasurement .

epa:WaterMeasurement a owl:Class ;
   rdfs:comment "A measurement about a water sample."^^xsd:string ;
   rdfs:subClassOf repr:Measurement .

wqo:Water_Contaminant a owl:Class ;
   rdfs:comment "A situation in which hazardous materials pollute a
       source of water."^^xsd:string ;
   rdfs:subClassOf wqo:AbstractObject .

wqo:WaterProperty a owl:Class ;
   rdfs:comment "An observable property in the body of water."
```

```
        ^^xsd:string ;
    rdfs:subClassOf wqo:ObservedProperty ;
    owl:equivalentClass [ a owl:Restriction ;
          owl:onProperty epa:hasMeasurement ;
          owl:someValuesFrom epa:WaterMeasurement ] .

epa:PollutedWaterSource a owl:Class ;
    rdfs:comment "A facility that has violated regulatory requirements
         on pollution"^^xsd:string ;
    rdfs:subClassOf epa:PollutedThing,
       geo:Site ;
    owl:equivalentClass [ a owl:Class ;
          owl:intersectionOf ( [ a owl:Restriction ;
                    owl:onProperty wqo:hasProperty ;
                    owl:someValuesFrom wqo:WaterProperty ]
                       body:BodyOfWater ) ] .

epa:ExceededThreshold a owl:Class ;
    rdfs:subClassOf repr:Measurement .

epa:Threshold a owl:Class ;
    rdfs:subClassOf epa:HealthEffect .

epa:hasSite a owl:ObjectProperty ;
    rdfs:comment "Links a site to potential uses for it."^^xsd:string
        ;
    rdfs:range epa:WaterMeasurementSite .

epa:hasSymptom a owl:ObjectProperty ;
    rdfs:domain epa:HealthEffect ;
    rdfs:range epa:Symptom .

epa:hasUnit a owl:DatatypeProperty ;
    rdfs:domain ssn:Sensor .

epa:hasValue a owl:DatatypeProperty,
       owl:FunctionalProperty ;
    rdfs:domain ssn:Sensor .

xsd:nonNegativeInteger a rdfs:Datatype .
```

```
time:Interval a owl:Class ;
   rdfs:subClassOf repr:Measurement .

wqo:ObservedBy a owl:ObjectProperty ;
   rdfs:comment "Links Sensor to a Property that the sensor can
      observe."^^xsd:string ;
   rdfs:domain wqo:WaterProperty ;
   rdfs:range ssn:Sensor .

wqo:ObservedProperty a owl:Class ;
   rdfs:comment "ObservedProperty is a process in which a Sensor has
      been used to estimate or calculate a value of a property."
      ^^xsd:string ;
   rdfs:subClassOf [ a owl:Restriction ;
         owl:onProperty wqo:ObservedBy ;
         owl:someValuesFrom wqo:PhysicalObject ],
      owl:Thing .

wqo:hasProperty a owl:ObjectProperty ;
   rdfs:comment "Links a measurement to the characteristic measured."
      ^^xsd:string ;
   rdfs:domain epa:WaterMeasurement ;
   rdfs:range wqo:WaterProperty .

epa:PollutedThing a owl:Class ;
   rdfs:comment "The set of all things that are polluted."
      ^^xsd:string ;
   rdfs:subClassOf wqo:SpaceTimeRegion .

epa:Symptom a owl:Class ;
   rdfs:subClassOf epa:HealthEffect .

owl:Thing a owl:Class .

wqo:PhysicalObject a owl:Class .

wqo:AbstractObject a owl:Class .

wqo:SpaceTimeRegion a owl:Class .
```

```
repr:Measurement a owl:Class ;
   rdfs:comment "Measurement class is imported from SWEET 2.1"
      ^^xsd:string ;
   rdfs:subClassOf wqo:AbstractObject .

epa:HealthEffect a owl:Class ;
   rdfs:subClassOf wqo:AbstractObject .

ssn:Sensor a owl:Class ;
   rdfs:comment "Sensors are the physical devices that observe some
      Property."^^xsd:string ;
   rdfs:subClassOf wqo:PhysicalObject .

wqo:Acidification a owl:Class ;
   rdfs:subClassOf wqo:Water_Contaminant .

wqo:AlkalinitySensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .

wqo:Ammonia a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:AmmoniaSensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .

wqo:Nitrate a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:NitrateSensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .

wqo:BODSensor a owl:Class ;
   rdfs:comment "Biochemical Oxygen Demand Sensor"^^xsd:string ;
   rdfs:subClassOf ssn:Sensor .

wqo:BiochemicalOxygenDemand a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .
```

```
wqo:Chlorine a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:ChlorineSensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .

wqo:ChlorophyllSensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .

wqo:Chlorophyll_a a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:ConductivitySensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .

wqo:Depleted_DO a owl:Class ;
   rdfs:subClassOf wqo:Water_Contaminant .

wqo:DissolvedOxygen a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:DissolvedOxygenSensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .

wqo:Eutrophic_Condition a owl:Class ;
   rdfs:comment "moderate-to-highly productive waters are called
      eutrophic"^^xsd:string ;
   rdfs:subClassOf wqo:Water_Contaminant .

wqo:Fluoride a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:FluorideSensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .

wqo:Nitrogen a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:Nutrient_Condition a owl:Class ;
```

```
    rdfs:subClassOf wqo:Water_Contaminant .

wqo:Oligotrophic_Condition a owl:Class ;
    rdfs:comment "waterbodies with low productivity are called
        oligotrophic"^^xsd:string ;
    rdfs:subClassOf wqo:Water_Contaminant .

wqo:Pathogens a owl:Class ;
    rdfs:subClassOf wqo:Water_Contaminant .

wqo:Phosphorus a owl:Class ;
    rdfs:subClassOf wqo:WaterProperty .

wqo:Poor_optical a owl:Class ;
    rdfs:comment "Species composition/ abundance"^^xsd:string ;
    rdfs:subClassOf wqo:Water_Contaminant .

wqo:PrecipitationRate a owl:Class ;
    rdfs:subClassOf wqo:WaterProperty .

wqo:PrecipitationSensor a owl:Class ;
    rdfs:subClassOf ssn:Sensor .

wqo:RHSensor a owl:Class ;
    rdfs:comment "Relative Humidity Sensor"^^xsd:string ;
    rdfs:subClassOf ssn:Sensor .

wqo:RelativeHumidity a owl:Class ;
    rdfs:subClassOf wqo:WaterProperty .

wqo:Salinity a owl:Class ;
    rdfs:subClassOf wqo:Water_Contaminant .

wqo:SpecificConductance a owl:Class ;
    rdfs:subClassOf wqo:WaterProperty .

wqo:TDSSensor a owl:Class ;
    rdfs:comment "Total dissolved solids (TDS) Sensor"^^xsd:string ;
    rdfs:subClassOf ssn:Sensor .
```

```
wqo:TSSSensor a owl:Class ;
   rdfs:comment "Total suspended solids (TSS) Sensor"^^xsd:string ;
   rdfs:subClassOf ssn:Sensor .

wqo:TemperatureSensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .

wqo:Thermal_Pollution a owl:Class ;
   rdfs:subClassOf wqo:Water_Contaminant .

wqo:TotalAlkalinity a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:TotalDissolvedSolids a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:TotalNitrogenSensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .

wqo:TotalPhosphorusSensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .

wqo:TotalSuspendedSolids a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:ToxicContaminants a owl:Class ;
   rdfs:subClassOf wqo:Water_Contaminant .

wqo:Turbidity a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:TurbiditySensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .

wqo:WaterLevel a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:WaterLevelSensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .
```

```
wqo:WaterTemperature a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:WaterpH a owl:Class ;
   rdfs:subClassOf wqo:WaterProperty .

wqo:WaterpHSensor a owl:Class ;
   rdfs:subClassOf ssn:Sensor .

wqo:hasID a owl:DatatypeProperty ;
   rdfs:domain ssn:Sensor ;
   rdfs:range xsd:int .

wqo:hasLatitude a owl:DatatypeProperty ;
   rdfs:domain geo:Site .

wqo:hasLocation a owl:DatatypeProperty ;
   rdfs:domain ssn:Sensor ;
   rdfs:range xsd:string .

wqo:hasLongitude a owl:DatatypeProperty ;
   rdfs:domain geo:Site .
```

Listing A.1: Water Quality Ontology in N3 format

Bibliography

[1] The concept of limiting nutrients. http://lakewatch.ifas.ufl.edu/circpdffolder/nutrpt2.pdf. Accessed: 2013-09-20.

[2] Mica2 mote datasheet. http://www.eol.ucar.edu/isf/facilities/isa/internal/CrossBow/DataSheets/mica2.pdf. Accessed: 2014-01-10.

[3] Multihop routing. http://www.tinyos.net/tinyos-1.x/doc/multihop/multihop_routing.html. Accessed: 2013-09-20.

[4] D Abowd, Anind K Dey, Robert Orr, and Jason Brotherton. Context-awareness in wearable and ubiquitous computing. volume 3, pages 200–211. Springer, 1998.

[5] Gregory D. Abowd, Anind K. Dey, Peter J. Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a better understanding of context and context-awareness. In *Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing*, HUC '99, pages 304–307, London, UK, 1999. Springer-Verlag.

[6] Marcos K. Aguilera, Robert E. Strom, Daniel C. Sturman, Mark Astley, and Tushar D. Chandra. Matching events in a content-based subscription system. In *Proc. of Symposium on Principles of Distributed Computing*, PODC '99, pages 53–61, New York, NY, USA, 1999. ACM.

[7] Lule Ahmedi, Edmond Jajaga, and Figene Ahmedi. An ontology framework for water quality management. In Corcho et al. [22], pages 35–50.

[8] I.F. Akyildiz, Weilian Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):102–114, 2002.

[9] Giuseppe Anastasi, Marco Conti, Mario Di Francesco, and Andrea Passarella. Energy conservation in wireless sensor networks: A survey. *Ad Hoc Networks*, 7(3):537–568, 2009.

[10] T. Antoine-Santoni, J.F. Santucci, E. De Gentili, and Bernadette Costa. Using wireless sensor network for wildfire detection. a discrete event approach of environmental monitoring tool. In *Environment Identities and Mediterranean Area, 2006. ISEIMA '06. First international Symposium on*, pages 115–120, 2006.

[11] ANZECC & ARMCANZ. Australian and New Zealand guidelines for fresh and marine water quality, National Water Quality Management Strategy Paper no 4, 2000.

[12] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y. Choi, T. Herman, S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, and M. Miyashita. A line in the sand: a wireless sensor network for target detection, classification, and tracking. *Comput. Netw.*, 46(5):605–634, December 2004.

[13] G. Asada, M. Dong, T. S. Lin, F. Newberg, G. Pottie, W.J. Kaiser, and H.O. Marcy. Wireless integrated network sensors: Low power systems on a chip. In *Solid-State Circuits Conference, 1998. ESSCIRC '98. Proceedings of the 24th European*, pages 9–16, 1998.

[14] Valerie Bonstrom, Annika Hinze, and Heinz Schweppe. Storing RDF as a graph.

In *Web Congress, 2003. Proceedings. First Latin American*, pages 27–36. IEEE, 2003.

[15] Jenna Burrell, Tim Brooke, and Richard Beckwith. Vineyard computing: Sensor networks in agricultural production. *IEEE Pervasive Computing*, 3(1):38–45, January 2004.

[16] Rachel Cardell-Oliver, Mark Kranz, Keith Smettem, and Kevin Mayer. A reactive soil moisture sensor network: Design and field evaluation. *International journal of distributed sensor networks*, 1(2):149–162, 2005.

[17] M. Ceriotti, M. Corra, L. D'Orazio, R. Doriguzzi, D. Facchin, S.T. Guna, G.P. Jesi, R. Lo Cigno, L. Mottola, A.L. Murphy, M. Pescalli, G.P. Picco, D. Pregnolato, and C. Torghele. Is there light at the ends of the tunnel? wireless sensor networks for adaptive lighting in road tunnels. In *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pages 187–198, April 2011.

[18] M. Ceriotti, L. Mottola, G.P. Picco, A.L. Murphy, S. Guna, M. Corra, M. Pozzi, D. Zonta, and P. Zanon. Monitoring heritage buildings with wireless sensor networks: The torre aquila deployment. In *Information Processing in Sensor Networks, 2009. IPSN 2009. International Conference on*, pages 277–288, April 2009.

[19] Yingwen Chen, Hong Va Leong, Ming Xu, Jiannong Cao, Keith C. C. Chan, and Alvin T. S. Chan. In-network data processing for wireless sensor networks. In *Proceedings of the 7th International Conference on Mobile Data Management*, MDM '06, pages 26–, Washington, DC, USA, 2006. IEEE Computer Society.

[20] Kin Wah Chow and Qing Li. Issdm: an in-network semantic sensor data model. In *Proc. of the Symposium on Applied computing*, SAC '07, pages 959–960, New York, NY, USA, 2007. ACM.

[21] Michael Compton, Holger Neuhaus, Kerry Taylor, and Khoi-Nguyen Tran. Reasoning about sensors and compositions. In Kerry Taylor and David De Roure, editors, *SSN*, volume 522 of *CEUR Workshop Proceedings*, pages 33–48. CEUR-WS.org, 2009.

[22] Óscar Corcho, Cory A. Henson, and Payam M. Barnaghi, editors. *Proceedings of the 6th International Workshop on Semantic Sensor Networks co-located with the 12th International Semantic Web Conference (ISWC 2013), Sydney, Australia, October 22nd, 2013*, volume 1063 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2013.

[23] Young D, Afoa E, Meijer K, Wagenhoff A, and Utech C. Temperature as a contaminant in streams in the auckland region, stormwater issues and management options. Auckland Council technical report TR2013/044, Prepared by Morphum Environmental Ltd for Auckland Council. Auckland Council.

[24] R J Davies-Colley. Trigger values for New Zealand rivers. NIWA Client Report MfE002/22, Prepared for Ministry for the Environment.

[25] Isabel Dietrich and Falko Dressler. On the lifetime of wireless sensor networks. *ACM Trans. Sen. Netw.*, 5(1):5:1–5:39, February 2009.

[26] Patrick Th. Eugster, Pascal A. Felber, Rachid Guerraoui, and Anne-Marie Kermarrec. The many faces of publish/subscribe. *ACM Comput. Surv.*, 35(2):114–131, June 2003.

[27] Charles L Forgy. Rete: A fast algorithm for the many pattern/many object pattern match problem. volume 19, pages 17–37. Elsevier, 1982.

[28] M. Franceschinis, L. Gioanola, M. Messere, R. Tomasi, M.A. Spirito, and P. Civera. Wireless sensor networks for intelligent transportation systems. In *Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th*, pages 1–5, April 2009.

[29] Tobias Freudenreich, Stefan Appel, Sebastian Frischbier, and Alejandro Buchmann. Actress - automatic context transformation in event-based software systems. In *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, DEBS Í2, pages 179–190, New York, NY, USA, July 2012. ACM.

[30] Lin Gu, Dong Jia, Pascal Vicaire, Ting Yan, Liqian Luo, Ajay Tirumala, Qing Cao, Tian He, John A. Stankovic, Tarek Abdelzaher, and Bruce H. Krogh. Lightweight detection and classification for wireless sensor networks in realistic environments. In *Proceedings of the 3rd international conference on Embedded networked sensor systems*, SenSys '05, pages 205–217, New York, NY, USA, 2005. ACM.

[31] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Gener. Comput. Syst.*, 29(7):1645–1660, September 2013.

[32] Tian He, Sudha Krishnamurthy, Liqian Luo, Ting Yan, Lin Gu, Radu Stoleru, Gang Zhou, Qing Cao, Pascal Vicaire, John A. Stankovic, Tarek F. Abdelzaher, Jonathan Hui, and Bruce Krogh. Vigilnet: An integrated sensor network system for energy-efficient surveillance. *ACM Trans. Sen. Netw.*, 2(1):1–38, February 2006.

[33] Tian He, Sudha Krishnamurthy, John A. Stankovic, Tarek Abdelzaher, Liqian Luo, Radu Stoleru, Ting Yan, Lin Gu, Jonathan Hui, and Bruce Krogh. Energy-efficient surveillance system using wireless sensor networks. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, MobiSys '04, pages 270–283, New York, NY, USA, 2004. ACM.

[34] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and

Kristofer Pister. System architecture directions for networked sensors. In *Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS IX, pages 93–104, New York, NY, USA, 2000. ACM.

[35] Jason Lester Hill. *System architecture for wireless sensor networks*. PhD thesis, 2003. AAI3105239.

[36] A. Hinze and A.P. Buchmann. *Principles and Applications of Distributed Event-based Systems*. IGI Global research collection. Information Science Reference, 2010.

[37] Annika Hinze. Efficient filtering of composite events. In Anne James, Muhammad Younas, and Brian Lings, editors, *New Horizons in Information Management*, volume 2712 of *Lecture Notes in Computer Science*, pages 207–225. Springer Berlin Heidelberg, 2003.

[38] Annika Hinze, Kai Sachs, and Alejandro Buchmann. Event-based applications and enabling technologies. In *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems*, DEBS '09, pages 1:1–1:15, New York, NY, USA, 2009. ACM.

[39] Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter F. Patel-Schneider, and Sebastian Rudolph, editors. *OWL 2 Web Ontology Language: Primer*. W3C Recommendation, 27 October 2009. Available at http://www.w3.org/TR/owl2-primer/.

[40] Ian Horrocks, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. SWRL: A semantic web rule language combining OWL and RuleML. W3c member submission, World Wide Web Consortium, 2004.

[41] M. Horton, D. Culler, K. Pister, Jason Hill, R. Szewczyk, and A. Woo. MICA, the commercialization of microsensor motes. *Sensors*, 19(4):40–48, April 2002.

[42] Qin Huaifeng and Zhou Xingshe. Integrating context aware with sensornet. In *Proceedings of the First International Conference on Semantics, Knowledge and Grid*, SKG '05, pages 83–, Washington, DC, USA, 2005. IEEE Computer Society.

[43] V. Huang and M.K. Javed. Semantic sensor information description and processing. In *Sensor Technologies and Applications, 2008. SENSORCOMM '08. Second International Conference on Sensor Technologies and Applications*, pages 456–461, 2008.

[44] Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin, John Heidemann, and Fabio Silva. Directed diffusion for wireless sensor networking. *IEEE/ACM Trans. Netw.*, 11(1):2–16, February 2003.

[45] Deokwoo Jung, Thiago Teixeira, and Andreas Savvides. Sensor node lifetime analysis: Models and tools. *ACM Trans. Sen. Netw.*, 5(1):3:1–3:33, February 2009.

[46] Mumraiz Khan Kasi, Annika Hinze, Steve Jones, and Catherine Legg. Energy-efficient context-aware routing in heterogeneous WSN. In *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems*, DEBS'14, pages 166–176, New York, NY, USA, 2014. ACM.

[47] Mumraiz Khan Kasi, Annika Hinze, Catherine Legg, and Steve Jones. SEPSen: Semantic event processing at the sensor nodes for energy efficient wireless sensor networks. In *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, DEBS '12, pages 119–122, New York, NY, USA, 2012. ACM.

[48] Mumraiz Khan Kasi and Annika Marie Hinze. Cost analysis for complex in-network event processing in heterogeneous wireless sensor networks. In *Proceedings of the 5th ACM international conference on Distributed event-based system*, DEBS '11, pages 385–386, New York, NY, USA, 2011. ACM.

[49] Bhaskar Krishnamachari, Deborah Estrin, and Stephen B. Wicker. The impact of data aggregation in wireless sensor networks. In *Proc. of International Conference on Distributed Computing Systems*, ICDCSW '02, pages 575–578, Washington, DC, USA, 2002. IEEE Computer Society.

[50] Joanna Kulik, Wendi Heinzelman, and Hari Balakrishnan. Negotiation-based protocols for disseminating information in wireless sensor networks. *Wirel. Netw.*, 8(2/3):169–185, March 2002.

[51] Iker Larizgoitia, Leire Muguira, and Juan Ignacio Vázquez. Architecture for WSN nodes integration in context aware systems using semantic messages. In *ADHOCNETS*, pages 731–746, 2009.

[52] Maurizio Lenzerini. Data integration: A theoretical perspective. In *Proceedings of the Twenty-first ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '02, pages 233–246, New York, NY, USA, 2002. ACM.

[53] Martin Leopold. *Sensor Network Motes: Portability & Performance*. PhD thesis, University of Copenhagen, Faculty of Science, Department of Computer Science, 2007.

[54] Philip Levis, Nelson Lee, Matt Welsh, and David Culler. TOSSIM: accurate and scalable simulation of entire TinyOS applications. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, SenSys '03, pages 126–137, New York, NY, USA, 2003. ACM.

[55] Philip Levis, Sam Madden, Joseph Polastre, Robert Szewczyk, Alec Woo, David Gay, Jason Hill, Matt Welsh, Eric Brewer, and David Culler. TinyOS: An operating system for sensor networks. In *in Ambient Intelligence*. Springer Verlag, 2004.

[56] Zhigang Li, Xingshe Zhou, Huaifeng Qing, and Shining Li. Model and implementation of context-aware sensor networks. In *Information Science and Engineering, 2008. ISISE '08. International Symposium on*, volume 2, pages 16–19, 2008.

[57] Samuel Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. TAG: a tiny aggregation service for ad-hoc sensor networks. *SIGOPS Oper. Syst. Rev.*, 36:131–146, December 2002.

[58] Samuel R. Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. TinyDB: an acquisitional query processing system for sensor networks. *ACM Trans. Database Syst.*, 30(1):122–173, March 2005.

[59] Alan Mainwaring, David Culler, Joseph Polastre, Robert Szewczyk, and John Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, WSNA '02, pages 88–97, New York, NY, USA, 2002. ACM.

[60] David Malan, Thaddeus Fulford-Jones, Matt Welsh, and Steve Moulton. Codeblue: An ad hoc sensor network infrastructure for emergency medical care. In *In International Workshop on Wearable and Implantable Body Sensor Networks*, 2004.

[61] Mateusz Malinowski, Matthew Moskwa, Mark Feldmeier, Mathew Laibowitz, and Joseph A. Paradiso. CargoNet: A Low-cost Micropower Sensor Node Exploiting Quasi-passive Wakeup for Adaptive Asychronous Monitoring of Exceptional Events. In *Proceedings of the 5th International Conference on*

Embedded Networked Sensor Systems, SenSys '07, pages 145–159, New York, NY, USA, 2007. ACM.

[62] K. Martinez, R. Ong, and J. Hart. Glacsweb: a sensor network for hostile environments. In *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, pages 81–87, Oct 2004.

[63] Kheireddine Mekkaoui and Abdellatif Rahmoun. Short-hops vs. long-hops - energy efficiency analysis in wireless sensor networks. In Abdelmalek Amine, Otmane Aït Mohamed, Boualem Benatallah, and Zakaria Elberrichi, editors, *CIIA*, volume 825 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011.

[64] V. Mhatre and C. Rosenberg. Homogeneous vs heterogeneous clustered sensor networks: a comparative study. In *Communications, 2004 IEEE International Conference on*, volume 6, pages 3646–3651 Vol.6, 2004.

[65] Eduardo F. Nakamura, Antonio A. F. Loureiro, and Alejandro C. Frery. Information fusion for wireless sensor networks: Methods, models, and classifications. *ACM Comput. Surv.*, 39(3), September 2007.

[66] Lionel M. Ni, Yanmin Zhu, Jian Ma, Minglu Li, Qiong Luo, Yunhao Liu, S. C. Cheung, and Qiang Yang. Semantic sensor net: An extensible framework. In *ICCNMC*, pages 1144–1153, 2005.

[67] Natalya F. Noy and Deborah L. Mcguinness. Ontology development 101: A guide to creating your first ontology. Technical report, 2001.

[68] B. O'Flyrm, R. Martinez, J. Cleary, C. Slater, F. Regan, D. Diamond, and H. Murphy. Smartcoast: A wireless sensor network for water quality monitoring. In *Local Computer Networks, 2007. LCN 2007. 32nd IEEE Conference on*, pages 815–816, 2007.

[69] Milenko Petrovic, Ioana Burcea, and Hans-Arno Jacobsen. S-topss: Semantic toronto publish/subscribe system. In *Proceedings of the 29th International Conference on Very Large Data Bases - Volume 29*, VLDB '03, pages 1101–1104. VLDB Endowment, 2003.

[70] D. Pfisterer, M. Wegner, C. Werner, and S. Fischer. Energy-optimized Data Serialization For Heterogeneous WSNs Using Middleware Synthesis. In *Proceedings of the Sixth Annual Mediterranean Ad-Hoc Networking WorkShop*, pages 180–187, 2007.

[71] Tomasz Rybicki and Jarosław Domaszewicz. Sensor-Actuator Networks with TBox Snippets. *Advances in Grid and Pervasive Computing*, pages 317–327, 2009.

[72] Sujata Sen, Mrinal Kanti Paul, and Madhab Borah. Study of some physico-chemical parameters of pond and river water with reference to correlation study. *Int. J. ChemTech Res*, 3(4):1802–1807, 2011.

[73] R.C. Shah and J.M. Rabaey. Energy aware routing for low energy ad hoc sensor networks. In *Wireless Communications and Networking Conference, 2002. WCNC2002. 2002 IEEE*, volume 1, pages 350–355 vol.1, Mar 2002.

[74] B.H. Shaw, Christine Mechenich, Lowell L. Klessig, and University of Wisconsin-Extension. *Understanding Lake Data*. Publication (University of Wisconsin–Extension). University of Wisconsin–Extension, Cooperative Extension, 1993.

[75] A. P. Sheth. Changing focus on interoperability in information systems: From system, syntax, structure to semantics. In *Interoperating Geographic Information Systems*, pages 5–30, Norwell, MA, 1999. Kluwer Academic Publishers.

[76] Victor Shnayder, Bor-rong Chen, Konrad Lorincz, Thaddeus R. F. Fulford Jones, and Matt Welsh. Sensor networks for medical care. In *Proceedings*

of the 3rd international conference on Embedded networked sensor systems, SenSys '05, pages 314–314, New York, NY, USA, 2005. ACM.

[77] Victor Shnayder, Mark Hempstead, Bor-rong Chen, Geoff Werner Allen, and Matt Welsh. Simulating the power consumption of large-scale sensor network applications. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, SenSys '04, pages 188–200, New York, NY, USA, 2004. ACM.

[78] Barbara Sibbald. Use computerized systems to cut adverse drug events: report. *Canadian Medical Association Journal*, 164(13):1878–1878, 2001.

[79] A. Silberstein, R. Braynard, G. Filpus, G. Puggioni, A. Gelfand, K. Munagala, J.Yang, B.Kamachari, D.Estrin, and S. Wicker. Data driven processing in sensor networks. In *Proc. of Biennial Conference on Innovative Data Systems Research*, CIDR '07, pages 10–21, Asilomar, California, USA, 2007.

[80] Hui Song, Sencun Zhu, and Guohong Cao. SVATS: A Sensor-Network-Based Vehicle Anti-Theft System. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 13–18, April 2008.

[81] J.A. Stankovic, T.F. Abdelzaher, Chenyang Lu, Lui Sha, and J.C. Hou. Real-time communication and coordination in embedded sensor networks. *Proceedings of the IEEE*, 91(7):1002–1022, 2003.

[82] Robert Szewczyk, Alan Mainwaring, Joseph Polastre, John Anderson, and David Culler. An analysis of a large scale habitat monitoring application. In *Proc. of International Conference on Embedded Networked Sensor Systems*, SenSys '04, pages 214–226, New York, NY, USA, 2004. ACM.

[83] Xueyan Tang and Jianliang Xu. Optimizing lifetime for continuous data aggregation with precision guarantees in wireless sensor networks. *Networking, IEEE/ACM Transactions on*, 16(4):904–917, 2008.

[84] Kirsten Terfloth, Katharina Hahn, and Agnès Voisard. On the cost of shifting event processing within wireless environments. In Mani Chandy, Opher Etzion, and Rainer von Ammon, editors, *Event Processing*, number 07191 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2007. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

[85] Duc A. Tran and Cuong Pham. Pub-2-sub: A content-based publish/subscribe framework for cooperative p2p networks. In *Proceedings of the 8th International IFIP-TC 6 Networking Conference*, NETWORKING '09, pages 770–781, Berlin, Heidelberg, 2009. Springer-Verlag.

[86] Duc A. Tran and Cuong Pham. A content-guided publish/subscribe mechanism for sensor networks without location information. *Comput. Commun.*, 33(13):1515–1523, August 2010.

[87] Agnes Voisard and Holger Ziekow. Designing sensor-based event processing infrastructures. *Proceedings of the 43rd Hawaii International Conference on System Sciences*, pages 1–10, 2010.

[88] Ping Wang. Semanteco. http://escience.rpi.edu/ontology/semanteco/2/0/pollution.owl. Accessed: 2012-08-20.

[89] Ping Wang, Jin Guang Zheng, Linyun Fu, Evan W. Patton, Timothy Lebo, Li Ding, Qing Liu, Joanne S. Luciano, and Deborah L. McGuinness. A semantic portal for next generation monitoring systems. In *Proceedings of the 10th international conference on The semantic web - Volume Part II*, ISWC'11, pages 253–268, Berlin, Heidelberg, 2011. Springer-Verlag.

[90] Xiaoyan Wang and Jie Li. Precision constraint data aggregation for dynamic cluster-based wireless sensor networks. In *Proc. of International Conference on Mobile Ad-hoc and Sensor Networks*, MSN '09, pages 172–179, Washington, DC, USA, 2009. IEEE Computer Society.

[91] Geoffrey Werner-Allen, Konrad Lorincz, Matt Welsh, Omar Marcillo, Jeff Johnson, Mario Ruiz, and Jonathan Lees. Deploying a wireless sensor network on an active volcano. *IEEE Internet Computing*, 10(2):18–25, March 2006.

[92] J. Wilson, V. Bhargava, A. Redfern, and P. Wright. A wireless sensor network and incident command interface for urban firefighting. In *Proceedings of the 2007 Fourth Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services (MobiQuitous)*, MOBIQUITOUS '07, pages 1–7, Washington, DC, USA, 2007. IEEE Computer Society.

[93] Alec Woo and David E. Culler. A transmission control scheme for media access in sensor networks. In *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, MobiCom '01, pages 221–235, New York, NY, USA, 2001. ACM.

[94] A. Wood, J.A. Stankovic, G. Virone, L. Selavo, Zhimin He, Qiuhua Cao, Thao Doan, Yafeng Wu, Lei Fang, and R. Stoleru. Context-aware wireless sensor networks for assisted living and residential monitoring. *Network, IEEE*, 22(4):26–33, July 2008.

[95] Alex Wun, Milenko Petrovic, and Hans-Arno Jacobsen. A system for semantic data fusion in sensor networks. In *Proceedings of the 2007 Inaugural International Conference on Distributed Event-based Systems*, DEBS '07, pages 75–79, New York, NY, USA, 2007. ACM.

[96] Yong Yao and Johannes Gehrke. The cougar approach to in-network query processing in sensor networks. *ACM Sigmod Record*, 31(3):9–18, 2002.

[97] Jonathan Yu and Kerry Taylor. Event dashboard: Capturing user-defined semantics events for event detection over real-time sensor data. In Corcho et al. [22], pages 19–34.

[98] Lingyun Yuan, Xingchao Wang, Jianhou Gan, and Yanfang Zhao. A data gathering algorithm based on mobile agent and emergent event-driven in cluster-based wsn. *JNW*, 5(10):1160–1168, 2010.