Information extraction from large-scale WSNs: approaches and research issues: approaches and research issues: part III: towards a hybrid approach

Daniel, T. and Gaura, E.

Sensors & Transducers Journal, Vol. 94, Issue 7, July 2008, pp. 57-82

Sensors & Transducers

ISSN 1726-5479

http://www.sensorsportal.com

Information Extraction from Large-scale WSNs: Approaches and Research Issues
Part III: Towards a Hybrid Approach

Tessa DANIEL, Elena GAURA
Cogent Computing Applied Research Centre, Coventry University, UK CV1 5FB
E-mail: e.gaura@coventry.ac.uk, www.cogentcomputing.org

Received: 29 June 2008 /Accepted: 21 July 2008 /Published: 31 July 2008

Abstract: Regardless of the application domain and deployment scope, the ability to retrieve information is critical to the successful functioning of any wireless sensor network (WSN) system. In general, information extraction procedures can be categorized into three main approaches: agent-based, query-based and macroprogramming. Whilst query-based systems are the most popular, macroprogramming techniques provide a more general-purpose approach to distributed computation. Finally, the agent-based approaches tailor the information extraction mechanism to the type of information needed and the configuration of the network it needs to be extracted from. This suite of three papers (Part I-III) offers an extensive survey of the literature in the area of WSN information extraction, covering in Part I and Part II the three main approaches above. Part III highlights the open research questions and issues faced by deployable WSN system designers and discusses the potential benefits of both in-network processing and complex querying for large scale wireless informational systems. *Copyright © 2008 IFSA.*

Keywords: Information extraction, Mobile agents, Wireless sensor network architectures

1. Introduction

This is the third in a series of three articles surveying information extraction approaches for large scale WSNs. Parts I and II have looked at the three main approaches: agent-based, macroprogramming, and query based. Each of the three approaches to information extraction surveyed in detail in Parts I and II of this suite of papers is suitable for a number of applications. They each have strengths but pose challenges as well. In summary, the agent-based approach provides a high degree of expressiveness and flexibility, as do the macroprogramming approaches, but they are both more difficult to implement into deployable WSN systems. The query-based systems have, as their key feature, ease of use but are limited in the types of queries that can be posed to the network. Macroprogramming tries to address this by providing powerful constructs for capturing higher level behaviour through global programs but introduce a steep learning curve to the user.

In this, the third part of the series, the authors here argue the case for a hybrid approach that retains the simplicity and ease of use of the more traditional query-based approaches while allowing the inclusion of useful logical abstractions provided by macroprogramming approaches to facilitate construction and resolution of more powerful queries. Such an approach would need to incorporate some of the principles of agent-based systems, such as collaboration and decision making in the network, in assisting in query resolution. In this context it can be hypothesized that end-users of WSN applications could be provided with the ability to construct and pose higher level information requests instead of simple queries requiring the collection of raw sensed values or the calculation of simple aggregates over the entire network [1].

Complex query type constructs that allow the expression of phenomenological spatio-temporal characteristics would provide the means for exploiting the very core of the networked sensing concept at large scale. Moreover, providing the end-user with a system that produces responses to these higher level requests for information within the network instead of as a result of post-collection analysis of all data would most certainly demolish one of the largest road-blocks of the WSN technology in its route to adoption: ensuring user acceptability. In the quest for testing the above hypothesis, the paper here looks at some of the open research issues the community faces with respect to information extraction, raises research question relating to the usefulness of in-network processing from an informational viewpoint and concludes the paper suite.

The paper is structured as follows: Section 2 identifies a family of applications which would benefit from a WSN-integrated higher level information extraction and delivery mechanism and surveys the state of the art of deployments in the identified application area. Section 3 describes the requirements for an integrated high-level information extraction mechanism, according to the authors' view, and reviews prior art in complex querying and in-network processing. Section 4 identifies WSN topologies, architectures and protocols suitable for advanced informational systems. Section 5 describes a new, hybrid, approach, termed a Distributed Complex Query Processor, and reports on how well it meets the requirements set in Section 3. Finally, Section 6 concludes the paper.

2. Information Extraction in Monitoring Applications

Military applications were the motivation for much of the initial research into WSNs with the Defense Advanced Research Projects Agency (DARPA) providing funding for a variety of projects from early 1990s to the present day. Today, however, the use of WSNs has expanded to encompass a large number of application domains. A particular group of WSN applications, denoted as monitoring applications, make up a large proportion of the WSN systems being deployed today. Though most monitoring systems are generally characterized by the need for attributes of interest to be observed, sensed and then relayed to a user, most WSN applications of this type show very little genericity in design. Rather, the WSN systems deployed are application-specific, with designs intimately linked to the particular application requirements. This section will identify some of the common types of monitoring application and describe examples of each, highlighting the information gathering model used in each example. An application acting as a motivator for higher level information extraction procedures is also described.

2.1. Habitat and Environmental Monitoring

Habitat and environmental monitoring form a particular set of monitoring applications that have grown over recent years. This is in part because of the great benefit they claim to provide to science and education as well as the fact that funding for improving science education is a priority. There are a number of WSN deployments in this category.

On the environmental side, the ARGO project [2] uses a sensor network to monitor the salinity and temperature of the upper ocean. The aim of the project is to generate a quantitative model of the changing state of the upper ocean while monitoring ocean climate patterns over the short to longer term. The data gathered is expected to act as input to ocean and ocean-atmosphere forecast models for data assimilation and model testing. Nodes are dropped from aircraft and cycle between the surface and a depth of about 2000m every ten days collecting data. When nodes rise to the surface, data collected by nodes are individually transmitted to a satellite.

The Great Duck Island project [3] is an example of a habitat monitoring application where a WSN was deployed to monitor the breeding behaviour of small birds called petrels. Scientists are interested in the usage patterns of nests as well as the environmental changes in and outside of nests during the breeding season. Measurements are taken of humidity, temperature, pressure and light level. Nodes are clustered with each cluster connected to a base station via a long-range antenna. Sampling is done every minute and readings are sent directly to the base station which connects to a database via a satellite link.

Another environmental monitoring application is the GLACSWEB project [4] which uses a sensor network to monitor sub-glacier environments in Briksdalsbreen, Norway. Holes are drilled at different depths in the ice and sensor nodes carrying pressure, temperature and tilt sensors are deployed in them. The nodes communicate with a base station on top of the glacier which measures supra-glacial displacements. The collected data is transmitted via GSM with no information processing occurring within the network itself.

For all of the applications above, raw sensed values are sent via base stations or servers to a *user* and processing occurs at that point.

2.2. Agricultural Monitoring

Increasingly, WSNs are being used to monitor conditions that affect plant growth. These networks are used in precision agriculture which aims to improve crop yields, reduce pollution and monitor the general health of crops. Variables monitored include temperature, soil moisture, humidity and light and are usually referred to as micro-climate variables. The Climate Genie system [5] is one such commercial example and is used to monitor vineyards. Nodes are spread over the vineyard in a wireless mesh configuration and are equipped with sensors for measuring temperature, moisture and light. The data gathered is summarized (aggregates that reflect grape quality and vitality) within the network and sent via Wi-Fi, cellular or satellite to servers for viewing anywhere using a web browser. This system therefore incorporates some level of in-network processing although limited to the calculation of aggregates which are used in decision making by the observer.

Another system was developed to monitor in real-time field conditions including leaf moisture, soil temperature, soil moisture and CO2 [6]. Field monitoring servers (FMSs) similar to web servers were deployed in rice paddy fields, collected data automatically and transmitted the data for permanent storage in publicly accessible databases. Real-time data was made accessible via a web browser. Like the habitat and environmental monitoring applications described in Section 5, raw sensed values are sent to the databases for storage and analysis occurs from that point on.

2.3. Structural Health Monitoring

Structural health monitoring (SHM) refers not only to the state of health of a given structure whether a building or bridge, for example, but also the detection of changes that may affect the structure's health in the future. There are two types of SHM systems: systems that monitor disaster response after some catastrophe has occurred, for instance, an earthquake and dedicated to continuous health monitoring which may check for signs of stress, monitor vibrations, wind, etc.

Disaster response systems are still a young area of WSN research. [7] describe a centralized WSN for structural-response data acquisition called Wisden. Wisden collects structural response data from a multi-hop network and relays and stores it in a base station. [8] also proposed a wireless monitoring system aimed at detecting damage to civil structures after a disaster (such as an earthquake) has occurred. In other research, technology developed by the CodeBlue project [9] is being used in the AID-N project [10] at Johns Hopkins Applied Physics Laboratory in developing systems aimed at disaster response.

For continuous, structural health monitoring applications the focus is on high sample accuracy with minimal distortion, high frequency sampling, time synchronization of readings and efficient data collection as opposed to energy efficiency through reduced power consumption. [11] describes a SHM system deployed on the Golden Gate Bridge. 64 nodes were deployed over a 4200ft long length and measurements taken of ambient structural vibrations. All collected data were relayed to a base station for analysis.

2.4. A Motivating Scenario

A WSN application frequently put forward is that of forest fire detection and monitoring [12]. This application is attractive because forestry is a major industry in many parts of the world, and forest fires are a major cause of loss of wood (in the USA the annual average loss to fire is 17,000 km^2). Early warning of a fire event and the manner in which it spreads are hence necessary.

Theoretically, any practical forest fire detection system is likely to exceed the scale of present WSNs by a considerable margin, with an expectation that hundreds of thousands of nodes would be needed for detailed monitoring and precise fire detection and localization. Considering a network of this scale, it is clear that real-time data searches using conventional centralized query mechanisms are not an option. For each fire detection cycle, hundreds of thousands of readings must be returned to the sink and processed. Following detection of the fire, new queries must be generated and directed to the nodes in the area which would be, ideally, geographically mapped. Finally, those nodes need to return the infrared data required for the map. A more efficient proposition would be for the initial event to be detected within the network, with the subsequent queries for the map data being generated locally, without returning data to the host.

A number of deployments have tackled some of the very issues described above but for practical reasons, the deployments have taken very different approaches to those proposed in the scenario described above. The FireWxNet system [13] a wireless sensor system aimed at monitoring weather conditions in woodland fire environments is one such example. The system monitored a variety of weather conditions that influence fire behaviour with an aim to using them to predict fire behaviour. The application, as is the case with most monitoring applications, was driven by a list of requirements acquired through consultation with both fire fighters and fire researchers. The implemented WSN system was deployed in the Bitterroot National Forest in Idaho (USA) and consisted of 3 sensor networks totaling 13 nodes, 5 wireless access points, 2 web cameras and 5 long range links.

The deployment was distinguished by the rugged environment within which the WSN had to function as well as the sparseness of the deployment itself. The authors note that sparse coverage was a deliberate choice aimed at strategically placing nodes to cover as much meaningful terrain with as few nodes as possible. Rather than scale, the focus was on creating an extremely robust design which included not just robust equipment but robust routing protocols as well. Once the system was launched, a number of challenges were encountered particularly given the harsh deployment environment but the system overall was a success. During its operation, over 80,000 data were streamed real-time, with operations applied post-collection to transform that data into usable information.

This real-life deployment presents a marked contrast to the motivating scenario and approach described above: the problems **are** well characterized given the input of domain experts; there is quite detailed knowledge of the application domain and what the corresponding WSN application requirements are. However, on-the-ground challenges usually mean compromises have to be made and as a consequence real-life monitoring systems so far, rarely aim to deploy such large quantities of nodes as put forward in the motivating scenario here.

The FireWxNet example highlights the limitation of the vast majority of monitoring applications today. They are limited in that they are conceptualized, designed and implemented as primarily data collection systems. In-network processing is absent and the systems can be considered automated to the extent that there is little user interaction. In essence, events are defined, queries may be deployed for sensed readings and all data is simply relayed to a centralized location for further analysis. Very few systems look beyond this simple sense-and-send model to incorporate some sort of analysis, in-network, prior to communicating results. Granted, with some systems it is difficult to define beforehand what events or processes may be of interest and these applications by their nature have to be more exploratory at least at the pilot deployment stage, with feedback influencing subsequent iterations. However, with some applications, as in the FireWxNet example, the problems are better characterized and therefore more amenable to some more sophisticated analysis or processing within the network. This could make the application not only more useful but more efficient as well.

Hence, the view put forward here is that in many of monitoring applications (the fire monitoring scenario being only one example) it is possible to define high level information requirements that could incorporate in-network processing techniques in resolving the requests. Besides the efficiency benefits in terms of reduced transmissions, this approach could transform a data collection system to an information generating system, extending the application scope while still fulfilling the basic application requirements.

As an example, an **informational** fire monitoring system would allow the processing of high-level queries aimed at tracking the fire. The queries would need to allow the expression of spatial and temporal characteristics and the querying system would need to support a level of autonomy in terms of some in-network decision-making by the nodes given the impracticality of streaming all data back to the sink. Such a system would perhaps allow a user to zoom in on particular problem spots and query for even more detailed information on-the-fly. Complex queries are proposed as extremely suitable for use in the scenario above as well as other monitoring application scenarios and a degree of autonomy is introduced through the implementation of in-network logical abstractions for query processing and resolution.

3. Requirements for a Higher Level Information Extraction System

With a view of the above, a higher-level information extraction system should to be able to:

- enable the user to construct and disseminate complex queries;
- allow a user to program the sensor network in a similar way to current applicative approaches;
- enable a means for in-network distributed query processing that allows information to be generated within the network;
- be implementable on constrained resource nodes.

3.1. Catering for Complex Queries

Towards the requirements above, some research has addressed specifically the need for complex queries and for the ability to process these queries within the network. [14] for example, define a complex query as a query consisting of one or more subqueries that are combined by conjunctions or disjunctions in an arbitrary manner. In their work on the Active Query Forwarding mechanism (ACQUIRE), they promote the use of these ‘nested queries’ and describe a mechanism that seeks to resolve the query in-network, generating information as a response. The work, however, does not address the implementation of the mechanism and instead presents a mathematical model that is used to analyze the performance of the approach in terms of energy cost.

[15] also identify the need for nested queries (which they too call complex queries) and highlight the problems with evaluating such queries especially in cases where aggregation dependencies exist between the nested queries. They put forward the idea of the query itself supporting abstractions which can then be used in query resolution. In their example the abstractions are geographical regions. Their research resulted in a qualitative study of the requirements of such a system and did not extend to implementation.

Beyond the work reported above, a complex query, in the authors’ view, is a query which:

- consists of one or more subqueries (nested queries) and/or
- contains multiple operations such as aggregates and/or
- contains spatial and/or temporal elements.

The query-based systems currently in use neither provide the facility to construct these complex queries nor give users the ability to process them. Such queries, for instance, queries with dependencies (nested queries) would require more complex in-network interactions than those supported by current query-based systems. One example of a complex query which forms the object of the research here would be:
‘What is the average temperature in those areas in the network where the humidity is greater than 95 and the air pressure is between 900 and 1000 mbar’.

This query exhibits a number of complex elements. First, the query language would have to be able to accommodate the expression of the spatial elements described as ‘areas’ in the example above. Second, the query is a dependent or nested query and can only be answered after some reasoning within and between the defined spatial entities. In effect, parts of the query depend on a previous question being answered and only become relevant if a particular answer is obtained.

With the primary goal of creating a system that exhibits simplicity and usability, using a declarative language already familiar to users of existing applicative query mechanisms as well as traditional database systems is considered a worthwhile approach. Identifying and investigating existing SQL-based query languages towards creating a language capable of expressing the complex query requirements described above is essential.

3.2. Catering for In-Network Complex Query Processing

As will be shown in Section 4.4, a number of in-network processing techniques have already been proposed in the literature and used in existing information extraction systems. These include techniques like aggregation, fusion and filtering which have been shown to improve energy efficiency in WSN systems and have been incorporated extensively in both query-based and agent-based systems. In addition, a powerful feature of many of the macroprogramming systems has been the creation and use of node or network level logical abstractions to facilitate in-network information processing. The literature has shown that so far logical abstractions have not been considered for application in query processing systems. The authors believe that these logical constructs can provide a more powerful means for processing the complex queries identified as being of interest to monitoring applications. The usefulness of abstractions that can be constructed logically within the network and used in conjunction with the in-network processing techniques mentioned above is examined in Sections 5.4 and 5.5. The abstraction proposed in Section 5 can be based on two types of attributes. Static attributes which do not vary over time, for example, the type of reading a node provides and dynamic attributes which do vary over time, for example, the current sensor reading. The key idea here is that the abstractions would be a component of the query itself and constructed prior to the dependent query being posed. These abstractions will drive the manner in which the query is both disseminated and processed within the network.

Consider an example where a user is interested in monitoring the soil acidity and relative humidity in ‘hot patches’ of the vineyard. The query posed would first need to define what the ‘regions of interest’ are. In this case, regions would be logically constructed over areas where the temperature level registers above a given threshold. Once these regions have been defined, the body of the query, aimed at retrieving soil acidity and humidity readings, will be disseminated to the relevant nodes via the region construct and not to any node within broadcast range, for instance. The core concept is that the regions are queried rather than individual nodes. Another key feature is that query-dependent logical abstractions will be used to facilitate query dissemination and processing in the network. The following sections review and identify suitable WSN topologies and adequate architectures and protocols which might enable the implementation of such an information extraction system.

4 WSN Topologies, Routing Protocols and Architectures

4.1. WSN Topologies

The most basic WSN topology is the centralized, sink-based topology, sometimes referred to as a flat or single-tier architecture where nodes in the network are homogeneous, that is, identical in terms of hardware complexity and battery power [16, 17], and last but not least, bandwidth management.

Data collected by the nodes are directed toward the sink or base station (usually the only ‘node’ more powerful both computationally and in terms of energy capabilities) using single or multi-hop communication [19]. Fig. 1 shows a diagram of a typical flat WSN topology. This configuration brings a number of challenges particularly if there are a large number of nodes. These include management of energy consumption, energy optimization, routing, information gathering and general management of the sensor nodes themselves [16, 17]. The University of California at Berkeley’s Redwood forest deployment [20] is one example of a WSN system exhibiting a flat architecture. The 33 node deployment used TASK [21], a self-contained sensor network system based on the TinyDB query processor [22] to monitor the microclimate (temperature, relative humidity and solar radiation) of a redwood tree over a 44 day period.

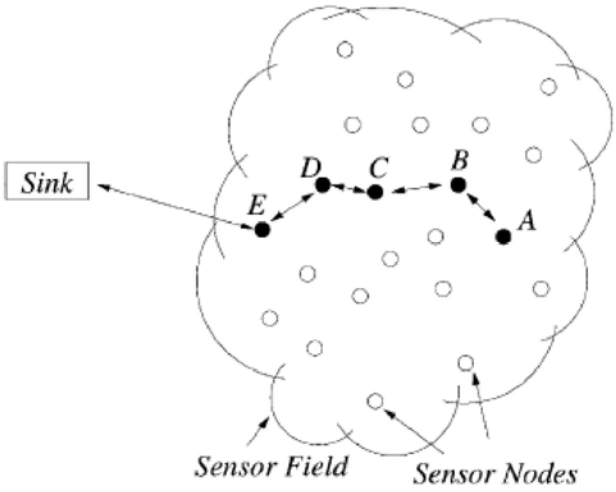

Fig. 1. A typical flat WSN architecture, taken from [18].

Whilst centralized, sink-based topologies are relatively simple to support and implement, they do not fulfill a topical requirement of WSN designs: scalability. Scalability has been described as one of the key design requirements both for conventional communication networks as well as wireless sensor networks. As the number of nodes increases with a flat topology, however, the sink node may become overloaded leading to increased latency as well as severe energy usage [23]. As a result of the apparent problems posed by flat networks, hierarchical heterogeneous architectures were proposed. The simplest example of this type of network consists of two layers: the first contains groups of homogeneous nodes, called clusters, connected to a dedicated micro-server or cluster head [19]. Cluster heads are sparsely distributed and serve as aggregators of data and managers of nodes within their individual clusters. They also serve as communicators both to a gateway or sink node and with other cluster heads in accomplishing application goals. Fig. 2 shows an example of a hierarchical architecture.

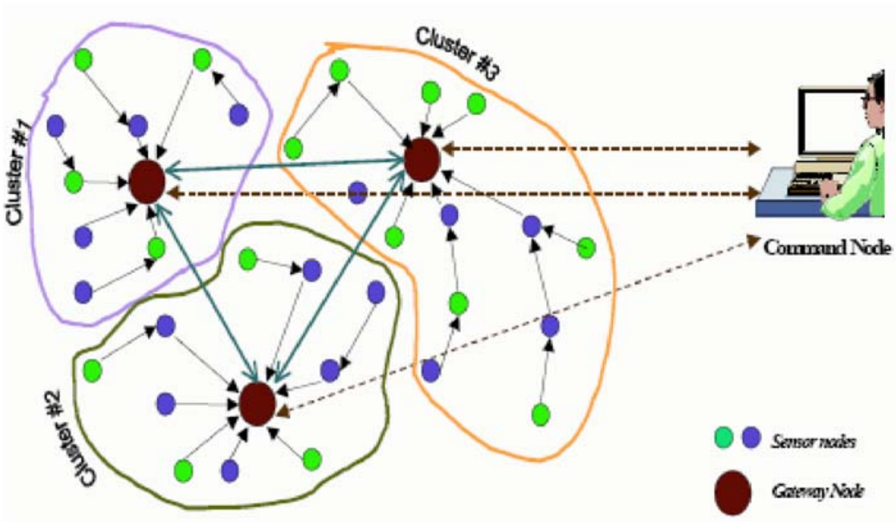

Fig. 2. A hierarchical, heterogeneous WSN architecture, taken from [24].

The introduction of cluster heads with special functionality (usually supported by enhanced hardware) gives the hierarchical structure some advantages over the flat topology particularly with respect to energy consumption.

Note: In this respect, multihop communication has been identified as a more favourable strategy over single hop, given that energy consumption is directly proportional to the square of the distance [25]. Hence, there is potentially more energy conserved with multiple short hops from node to sink, for example, as opposed to a direct long hop from a node to the sink [26]. A problem occurs, however, with the nodes closest to the sink since they are under heavier traffic and more likely to have their energy drained more quickly. High energy consumption can also be a problem if clusters where multihop communication is used are large.

Cluster heads provide additional functions like caching and forwarding of data to the required destination as well as performing data aggregation and fusion in order to decrease the number of transmissions to the sink or gateway. This is of particular importance in data gathering networks. It has been shown that for some applications, using a hierarchical structure can bring about significant improvement in performance in terms of reliability, longevity and flexibility of the network [19]. The Great Duck Island habitat monitoring application [3] is one example of a deployment that used a hierarchical architecture.

Building on the basic two-tier hierarchical model, varying multi-tier architectures have been put forward to capitalize on the apparent advantages. Examples are those proposed in [27] and [28]. Finally, the SENMA architecture [29] proposes a novel two-tier architecture where the upper layer consists of mobile access points that are used for data acquisition from homogeneous sensors within the sensor layer. Sensors communicate with mobile agents who periodically move within radio communication range eliminating the need for multihop communication. This architecture, because of the reduction in multihop communication, showed a significant improvement in energy efficiency.

4.2. Routing Protocols for WSNs

Given the unique characteristics of WSNs the underlying routing protocol used is an important consideration when looking at information extraction. In many cases the information extraction mechanism and how efficiently it works depends to a great extent on the routing method used. In some cases the routing mechanism itself includes techniques for query dissemination and data aggregation as in Cougar [30] and ACQUIRE [14]. Routing is therefore an important aspect of information extraction in WSNs.

Many routing protocols are currently being used in a variety of wired and wireless networks [31]. Although protocols can be classified in different ways, for example, based on the network structure or perhaps on the protocol operation [32] many follow the address-centric (AC) model where routes are found and followed between pairs of addressable nodes. In mobile ad-hoc networks, AC protocols are used widely (MANETS) [33]. Here each source independently sends data along the shortest path to a sink. This path is usually based on the initial route a query took to get to that source node. Although MANETs are similar to WSNs in that they both involve multi-hop communication, their routing requirements are quite different for a number of reasons.

First, communication in a WSN comes from multiple data sources to the sink instead of just between a pair of nodes. Second, redundancy in sensed data is common in WSN since data is being collected by multiple sensors which may be sensing the same phenomena, a fact which data-centric protocols take into account in devising optimal routes. This is not a major consideration in MANETs. Finally, in WSN the major constraint is energy making it essential that data communication rates be made as efficient as possible, much more so than in MANETs.

Given these reasons among many others, the traditional end-to-end protocols used for MANETS are not appropriate for WSNs. Some alternative protocols propose a different model, the data-centric model, where sources send data to a sink, but the content of the data is examined and some processing (whether aggregation or fusion) is executed on that data en-route to the sink. The result in using such protocols is that a better transmission/information ratio is achieved. In the next section some data-centric routing protocols will be examined in more detail.

Although protocol development is usually seen as being beyond the scope of information extraction research, it is important to have an awareness of what protocols are used within the query processing systems in use and their impact on the information extraction mechanism. Data routing, for example, is an important consideration when looking at information extraction as it affects the overall efficiency of the mechanism. For purposes of building advanced informational systems fitting the requirements in Section 3, identifying a suitable cluster-based routing protocol that facilitates the implementation and testing of the logical abstractions is key.

The choice of architecture for a WSN system appears to depend strongly on the application requirements. Pure data collection applications tend to work with hierarchical topologies with cluster heads aggregating and relaying data as it is generated. In such systems information processing requirements are simply sense data collection or at most calculation of aggregates. This is not always the case, however, as some WSN systems like the Redwood Forest deployment [20] which for the most part exhibit a flat topology are also used for data collection. In such systems energy conservation is either not a major concern or techniques for optimizing query processing are incorporated to minimize energy consumption.

Again, cluster-based topologies (single or multihop) appear to be more amenable to logical region-based query processing approaches. Region leaders can be considered somewhat similar to cluster heads in terms of functionality therefore enabling a mapping of the logical construct to the network construct. It is expected that regions, in this type of architecture will be easier to implement and test experimentally.

A review of candidate query processing architectures is given below.

4.3. Query Processing Architectures

Database-Type Architectures: Database-type query processing systems are some of the most popular put forward in the literature. [34] propose a generic database-based query processing architecture for sensor networks (Fig. 3) and describe the components they feel are required at each layer to make the architecture practical.

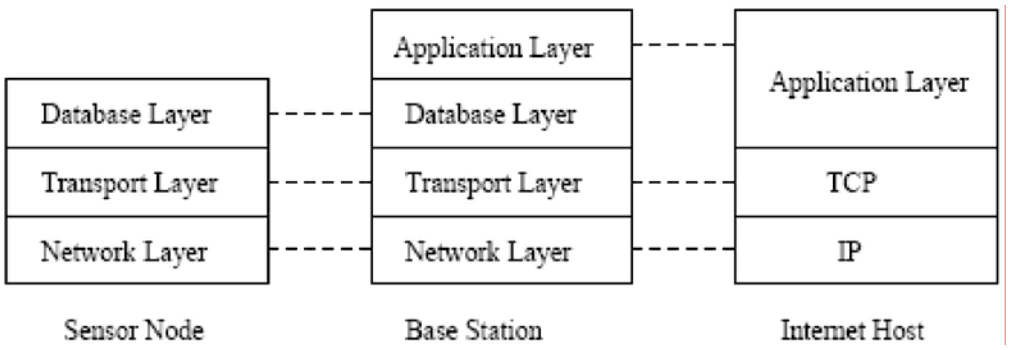

Fig. 3. A database-based query processing architecture taken from [34].

Application Layer: at the base station this layer provides an interface for posing of queries to the sensor network. These queries can be represented either as an SQL-type message or even a SOAP-like web service. The application layer in the base station is designed to communicate with the TCP/IP stack and so query results can be easily accessed by an Internet-based host.

Database Layer: on the base station, the database layer serves the application layer by receiving queries from it and returning results to it. The database layer on the base station contains a number of components including: a Parser - creates the query tree based on the query received from the application layer; Catalog Manager - maintains the relational schema, location and distribution of sensors and monitors nodes' status, for instance, node power levels and node connectivity; a Query Optimizer - generates the optimized query execution plan.

Due to resource limitations, the database layer's function at node level is much more limited. It receives the query tree either from the base station or another node and may need to further optimize the tree based on local catalog information. The database layer returns results in the form of relations.
Transport Layer: this layer provides for efficient end to end communication. At the base station it should be able to communicate with TCP entities on Internet hosts although alternative methods may be needed for nodes since nodes are address-less.

Network Layer: this layer should provide energy-efficient and data-centric routing algorithms. The routing decisions made should be based not only on the current network conditions but also the given query execution plan.

4.3.1. Example Systems

Query processing systems like TinyDB [22] and Cougar [30] are just some of the database-type systems put forward in the literature. They all follow to varying extents the architecture put forward by [34]. The query processing architecture is similar for both, consisting of server-side software running at a base station and responsible for parsing and delivering queries into the network, as well as collecting the results as a stream out of the network.

The architecture of the TinyDB system, both server-side and node-side, is illustrated in Fig. 4. Architecturally the query layer sits between the application and network layers in the case of the server side component (base station) and sits on top of the network layer at node level. Like in Fig. 3 the application layer allows the construction and posing of queries with the database layer, here referred to as the query layer, below parsing the query, constructing query execution plans and delivering the results to the network. The query layer at the server side also collects the results for presentation to the application layer.

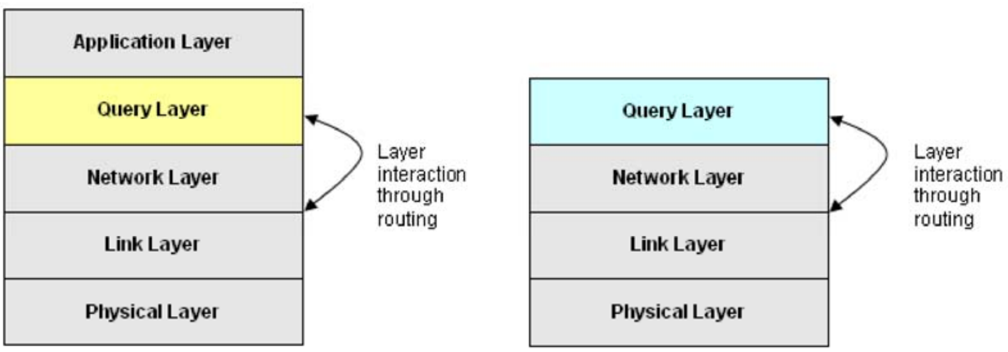

Fig. 4. A high level view of the query processing architecture used in Cougar and TinyDB.

The query layer houses a number of components some of which have already been described in the architecture put forward by [34] and are illustrated in Fig. 5.

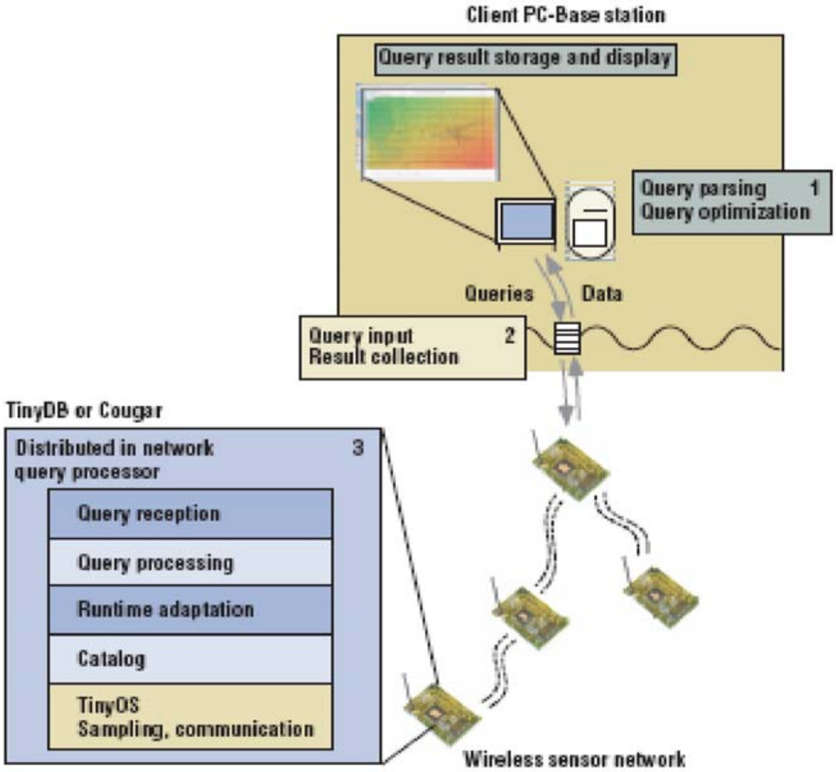

Fig. 5. A detailed view of the query processing architecture of TinyDB and Cougar taken from [35].

The query layer is the backbone of the query processing system and provides a number of critical functions:

- the processing of the query itself. The query layer uses sophisticated techniques like catalogue management, query optimization to abstract the user from the physical details of the network in processing queries.
- performing in-network processing to reduce communication given the need to preserve resources like energy and bandwidth. To do this the query layer generates different query plans with different trade offs for requirements such as accuracy, energy consumption and latency.
- interacts with the routing layer to facilitate in-network processing. With traditional routing, the network layer on a node will automatically forward packets to the next hop towards the destination with the upper layers, completely unaware that data packets are moving through the node. In implementing in-network aggregation, for example, the query layer needs to be able to communicate with the network layer when it wants to intercept packets destined for the sink or leader node. In the processing system described above, filters are used to first access the packets then modify and delete if necessary before passing on to the next hop onto the destination.

Although they can probably be implemented on both hierarchical and flat network topologies, these database type approaches have so far been implemented on flat networks.

Middleware Architectures: Another approach to query processing in WSNs is that of query-based middleware systems. Two of the more popular are DSWare and SINA described in Parts I and II of this papers suite. Although also inspired by database systems in that they use SQL-based languages for construction of queries, these systems are considered middleware approaches as they also provide services that go beyond database query processing systems. Architecturally, these middleware approaches are similar to the database approaches. In both cases, the middleware layer sits between the application and network layers and in addition to providing facilities for query processing, supplies the services necessary for ensuring adequate WSN functionality. In the case of DSWare the application layer allows the construction of SQL-based event type queries which are then registered, parsed and execution plans generated. The underlying DSWare middleware layer is then responsible for registering and executing these events. The DSWare layer also interacts with the routing layer for improving network functioning, for example, improving power awareness.

SINA also comprises a middleware layer that sits above the network layer and provides an application programming interface to the middleware layer via SQTL scripts. The middleware layer runs on all nodes in the network and like the database approach it selects the most suitable distribution of the query based on the current network status and the type of query being executed. In addition to query processing functions, SINA also provides services for accessing sensor hardware and communication and supporting changing network topology (for example, sink mobility).

Both DSWare and SINA promote processing of aggregation-based queries and are suited to cluster-based topologies.

4.4. In-Network Processing Techniques for WSNs

Experiments have shown that the major part of power consumption costs is due to communication rather than computation [36, 37, 38]. [39] describe **in-network processing** as the pushing of operations, particularly selections and aggregation of data, into the network to reduce communication. [30] describe it as the moving of computation from outside to inside the network in an effort to reduce energy consumption and improve network lifetime. The main goal in effect is to reduce communication in exchange for some form of computation within the network. In-network processing techniques are therefore accepted as being essential to improving energy efficiency in gathering information within the WSN and critical to any system that has improved network lifetime and energy efficiency as goals. The two most common in-network processing techniques used for reducing communication in WSN systems are packet merging and partial aggregation [35]. In addition, a number of other widely used techniques exist. These are discussed below.

Filtering: Filters are constructs used within the network (on nodes) to assist in processing [40]. A filter registers what kind of data it handles through matching and is triggered each time that type of data enters the node. Once it has been activated, the filter can manipulate the data by, for example, determining whether it is forwarded on or even generating a new message. Filtering is sometimes used in conjunction with data aggregation. [40] proposed using a filter to detect concurrent detections of four-legged animals from different sensors. It could then record what the desired interval was and ensure that only one response per interval, suppressing responses from other sensors. [41] take advantage of event properties of monitoring queries, and carry out data filtering during in-network processing.

Packet Merging: Packet merging, described by [35] combines a number of smaller records into a few larger ones in a bid to reduce the number of packets needing to be transmitted and in turn reducing the cost of communication as the cost of transmitting several smaller packets is greater than transmitting just one large packet. The Cougar query processing system is an example of a system that uses packet merging for reducing communication in processing some types of aggregate queries.

Partial Aggregation: Partial aggregation refers to the computation of intermediate results as readings are received by the node. Communication is reduced as this combined partial result is transmitted on instead of all individual records. Aggregation can be achieved in various ways based on the type of correlation that exists in the WSN. This may be spatial correlation due to physical proximity of nodes and therefore similarity in the readings, temporal if there is little variation in the sensed attribute based on the sampling frequency and finally, correlation in the data itself due to overlap in sensor coverage [42].

Tree-based Aggregation: In tree-based aggregation one node is designated the root node. A broadcast message is sent out with data on the ID of the node sending the message and its depth in the tree. In the case of the root the depth would be zero. When nodes receive this message they assign themselves a level by adding one to the value in the message received and assign the ID in the message as their parent. Broadcasting of the message continues until all nodes within range have received it and assigned themselves a level and parent. TinyDB is one query processing system which uses a tree-based aggregation service called TinyAGgregation (TAG) [43, 35].

There are a number of advantages to the TAG approach. First, a reduction in the number of communications needed to calculate aggregates when compared to aggregation that occurs at one centralized location. Second, as data moves up the tree back to the base station nodes usually are required to transmit a maximum of one message. This is in direct contrast with centralized aggregation methods where the number of transmissions increases dramatically as data moves towards the root node. This can of course lead to a decrease in the lifetime of the network as battery power is quickly drained. Third, it allows aggregation even in networks where connectivity is intermittent or disconnections occur since disconnected nodes can reconnect by listening to other nodes' partial state records as they flow up the tree. This is possible since the partial state record includes information on the query that was issued.

TAG, therefore, presents a simple interface, flexible naming and generic operators for constructing aggregate queries and is not application specific as with other approaches using aggregation like Directed Diffusion [44] and Greedy Aggregation [45]. Further, it separates the logic of aggregation from routing details so that the focus is squarely on the application and leaves routing decisions to the system. This is in contrast to Directed Diffusion which puts aggregation mechanisms within the routing layer. The TAG approach allows a stream of aggregate values that change as sensor readings and the underlying network layout change and does this in an energy-efficient and bandwidth-efficient way.

TAG, however, does have limitations as it does not allow joins and cannot respond to events that occur within the network. The authors have indicated that future work will focus on developing an efficient way of aggregating the results of those event-based queries across nodes before transmitting that information to interested nodes.

Cluster-based Aggregation: In some cases nodes are in close physical proximity to each other and queries may need to retrieve information based on this spatial correlation. For example, a query may want to determine the average humidity over a particular area. Typically, clusters are formed consisting of nodes in close proximity based on some metric, for example, signal strength. Clusterheads are elected and act as the data aggregator and router for cluster members. SINA and Cougar, are examples of query processing systems that incorporate cluster-based aggregation in processing aggregate type queries.

Discussion: Given the importance of considering in-network processing when developing efficient WSN applications, this section examined some of the more popular techniques used. Data aggregation is most widely used, with the type of aggregation incorporated dependent on the network topology and to some extent the type of queries being issued. Systems that incorporate in-network processing and are therefore information-generating systems (as opposed to simple data gathering) have tended to be built on cluster-based topologies. In heterogeneous configurations, cluster heads can be configured to be more computationally powerful and are conducive to supporting data aggregation. There have, however, been, information-generation systems based on flat topologies which allow tree-based aggregation. Where scalability is an issue, and where in-network processing is computationally intensive, reported systems have had the tendency to lean towards a cluster-based configuration, this being more energy efficient and practical in creating a system with an extended lifetime.

Clustered networks have the advantage of conveniently allowing aggregation at the clusterhead and are most effective in networks that are static and where the cluster structure stays unchanged for a considerable period. With dynamic clusters, however, problems can occur with energy expended in continuously updating nodes in order to keep the clusters consistent with the underlying network topology. Tree-based aggregation is simple and useful but can lead to problems. Given a node failure in the network, for example, a packet lost at a given level of the tree can lead to all data being aggregated up that node's sub-tree being lost as well. This can lead to incomplete data making its way up the tree particularly if nodes only have one route to the sink and connectivity gaps occur.

Having set the requirements in Section 3, and outlined the topologies, architectures, and protocols in this section, the next section describes a new, hybrid, approach that combines query-based and macroprogramming concepts and evaluates it in light of the requirements previously discussed.

5. A Distributed Complex Query Processor

The database-type architecture forms the basis for the complex query processing architecture put forward by the authors here. The two part architecture consists of server-side software which is accessed by the user and used for constructing and posing declarative-type queries and node-side software which is in effect the distributed complex query processor (DCQP). A key component of the architecture will be the Region and Query Management Layer which will be responsible for not only query processing but will have additional tasks related to the creation, maintenance and update of regions within the network.

A number of factors influenced this selection. First, a key feature of the distributed complex query processor (DCQP) proposed is ease-of-use and familiarity for users. An SQL-based, database-like approach is therefore preferred. Although attractive, a middleware approach was not selected as the aim with the DCQP is to create a system that is separated as far as possible from network level configuration issues. The idea is that the system will sit on top of a functioning network. Second, the need for in-network processing for resolution of complex queries dictates that a query layer that allows in-network processing of queries is essential. The approaches above have shown that distributing the query layer to all nodes and providing functions for query processing and optimization are effective. The DCQP query layer therefore, will allow the formulation of query execution plans, dynamic optimization of these plans based on the attributes of interest and node state as maintained by the regions. Third, the DCQP architecture will incorporate a query layer that is decoupled as far as possible from the underlying layers. The idea here is to make the system as portable as possible by not strictly linking it to any particular routing protocol, for example, but instead identifying features that would make particular protocols more suitable than others.

A number of candidate architectures were investigated as a starting point for a distributed query processing system that incorporates best practices for in-network processing and resolution of complex queries while at the same time allowing the incorporation of novel techniques and strategies for the types of queries proposed. The next subsections describe in more detail the design choices and assumptions made and give reasons for these selections.

5.1. The Network Model

Key to the research work is the creation of logical regions within the network. These regions constitute a critical component in making a decision on where a query should be disseminated, how the query should be processed within and between regions and where the complex query will be ultimately resolved. The regions will have one ‘leader node’ each which acts as a manager, data aggregator and communicator to a gateway as well as other region leaders when needed. This role is remarkably similar to that of clusterheads in network-level clusters [19]. Additionally, like some clusters, regions should, and are likely to be in the approach proposed here, dynamic. Consequently, it makes sense that these logical region leaders map to physical clusterheads. These requirements, therefore, directly influence the choice of network model for the proposed work.

First, the network model proposed has a hierarchical topology. Sensors are randomly deployed and the transmission range is identical for all devices. For simplicity, an ideal MAC layer is assumed and node death considerations are not taken into account at this stage. Also assumed is that a route has been established between nodes in the network and the gateway node. (Gateway node here is defined as the node from which a query is disseminated and through which results are acquired. Gateways are not fixed; any node could potentially become a gateway at some point in time.) To facilitate region-based routing, a cluster-based routing protocol that allows dynamic cluster formation and supports inter-node communication and communication with the external world, is selected. Two possible choices identified so far are the Hybrid Energy-Efficient Distributed Clustering protocol (HEED) presented by [46] and which focuses on scalable data aggregation and increased network lifetime; and a dynamic clustering algorithm (DCRR) presented by [47] in which cluster heads are dynamically selected in the region where an event occurs according to their residual energy. Fig. 6 gives a pictorial description of the network model.

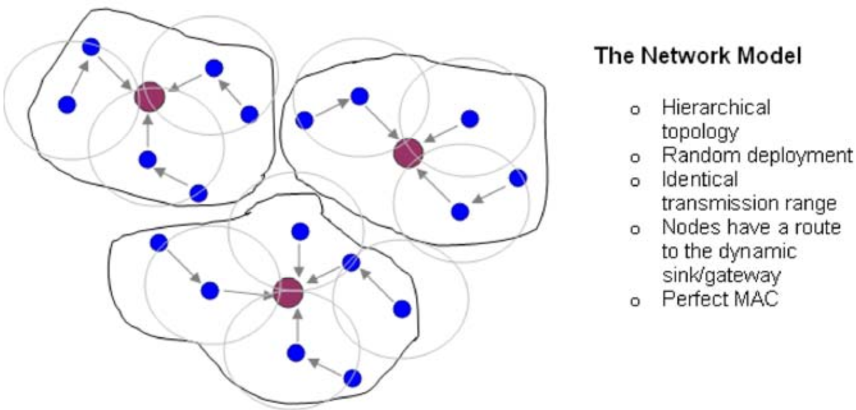

Fig. 6. The proposed network model.

5.2. A Distributed Complex Query Processing Architecture

The architecture proposed for the DCQP consists of server-side software which is accessed by the user and used for constructing, posing and parsing of declarative-type queries and node-side software which is in effect the distributed complex query processor (DCQP) implementing query processing functions in-network.

Server-side Software: This component will host an application layer which will allow the construction of queries with an underlying Region Query Management Layer which will be responsible for parsing of these queries and dissemination to the network. Additionally, the server-side software will allow the receipt of query results for both presentation purposes and for persistent storage. Fig. 7 shows the server-side architecture.

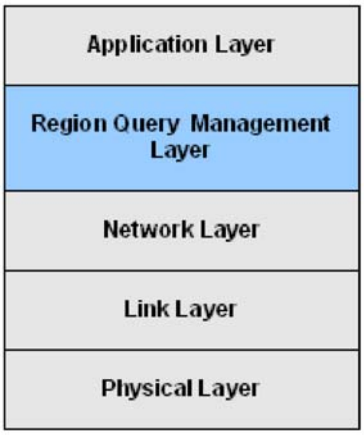

Fig. 7. DCQP Server-side architecture.

A key output of the work described here will be the creation of a system that will allow the user to either specify new logical regions as a component of a query being constructed or make use of regions already in existence within the network. This will involve the extension of a SQL-based query language to incorporate the region construct and the development of an associated parser for the language. The region abstraction will be a part of the query language used, and is essential to query dissemination as well as the creation of query execution plans.

Node-side Software: This component will also host a region query management layer which will from an architectural standpoint sit above the network layer. The node-side component is really a distributed query processor hosted by all nodes in the network.

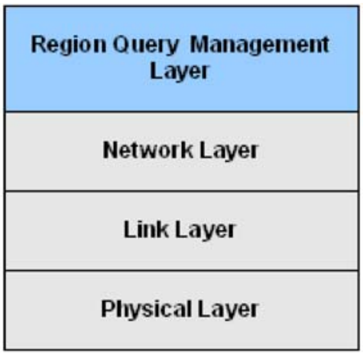

Fig. 8. DCQP Node-side architecture.

The region query management layer will deal with query reception, dissemination, processing and delivery of results. In addition, this layer will need to manage the logical regions which are implemented as part of the complex query. This functionality will be implemented as a region management component of the query management layer itself.

Region management will include the creation, maintenance and updating of region information on each node within the network. This may involve, for example, a region leader upon receipt of a query, informing its members of a new attribute of interest; or perhaps the termination of a previously posed query along with its associated regions; or the handing-off of a region leader's duties to a backup leader when the node's energy level falls below a certain threshold. Once a query is received, execution plans are created dynamically based on the required region formation or existing regions which need to be accessed. Region management is therefore an integral part of query plan formulation and query optimization. The query plan is then executed. Once results have been acquired these are communicated to the server-side component.

As in the Cougar system, the query management layer may have to interact with the routing mechanism in the network layer in order to enable in-network processing which is a key feature of the region-based query processing approach proposed. This interaction, however, is anticipated to be minimal since a cluster-based routing protocol which maps the logical regions to physical clusters is proposed as a way of minimizing intrusion into the network layer by the query management layer. Fig. 9 shows diagrammatically, the proposed complex query processing system architecture.

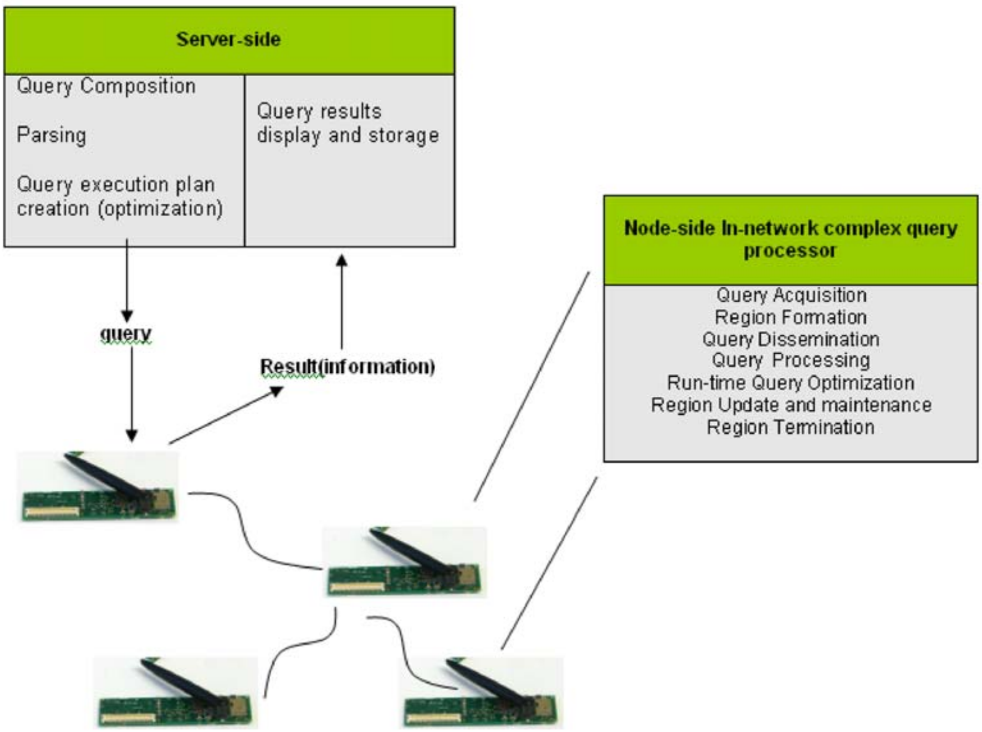

Fig. 9. A detail view of the functions provided by the complex query processing system.

In the following section, the DCQP approach is evaluated via simulation.

5.3. Acoustic Monitoring Using Region-based Querying

Habitat monitoring is already a rich area of research in the area of WSNs particularly because of the benefits to science and education. Within this group are applications based on acoustic sensing, which are concerned with event detection and classification as well as monitoring and localization. Bioacoustics research is a specific example and acoustic sensing in that context can be used to help scientists acquire acoustic data which can then be used to distinguish between animals, species and census counts.

The processing required for such applications usually involves complex signal processing operations and therefore present a number of challenges and requirements that are not evident in traditional monitoring systems. The challenges include the heavy computation needed due to very high data rates and the need for development of specific algorithms to facilitate on-line processing of very large amounts of data.

One such acoustic monitoring application, VoxNet focused on creating a system that allowed the detection of marmot alarm calls. The work was informed by the requirements of scientists and their desire to be able to detect these marmots in the field and then localize their positions. These calls, therefore, were used to help determine the location of the animal at the time the call was detected, relative to known burrow locations. Although for some systems simple recording and offline analysis of the data fulfills application requirements, in the case of this bioacoustic monitoring system it was important that the system produce timely results. The acoustic event detection and localization application consisting of eight nodes was deployed over an area of about 9800 sq meters (2.4 acres). A gateway node was then positioned about 200 m away from the nearest node. A map of the deployment is displayed in Fig. 10.

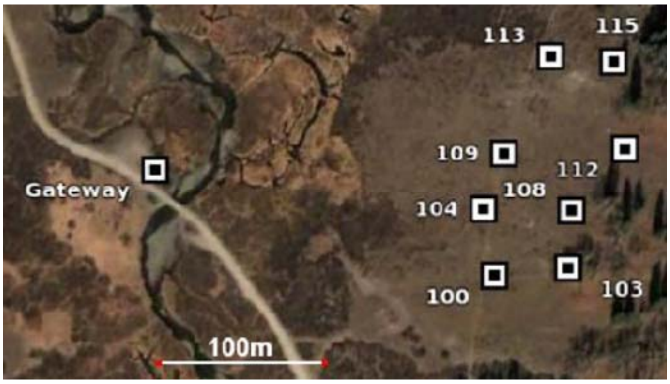

Fig. 10. Map of VoxNet deployment area.

Localization of Acoustic Signals – the VoxNet approach: Once an acoustic signal is detected by a node it is timestamped and an attempt made to determine the location of the sound. The localization algorithm used, the Approximated Maximum Likelihood algorithm (AML) [48] functions as follows: a stream of audio is processed through a 'fast path' detector to identify possible alarm calls; the algorithm then estimates bearing to the caller from multiple points followed by fusion of those estimates to produce an estimate of the sound location.

The algorithm was expressed as a WaveScript program which is a logical dataflow graph of stream operators connected by streams. Once the program runs on these input streams results are streamed back to data storage components, in this case the gateway or sink node. This execution of the WaveScript program constitutes the extent of in-network processing with the result being an estimate of the direction of arrival (DOA) of the acoustic signal. Timestamped, DOA values are then sent on by nodes detecting acoustic events to the sink. In some cases, depending on time availability, AML may not be executed on the node to produce DOA estimates and instead streams of raw detections are sent instead. The size of a DOA packet is 800 bytes as compared to 32 KB for a raw detection.

In the application a minimum of 3 acoustic signals or DOA estimates are needed to localize a sound. At the sink, these DOA estimates along with the timestamps are used to determine, first, whether the signals are indeed from the same acoustic event and if they are, they can be combined to determine the location at which the sound occurred.

For purposes of this work, metrics from the VoxNet deployment representing the number of data packets transmitted in localizing one event, the number of hops over which these were sent as well as the total distance traveled were used. A comparison was then made between the real-life deployment results and the results of the simulations incorporating in-network, region-based processing.

A Region-based Approach to Acoustic Signal Localization: The authors maintain that an approach that incorporated in-network processing using the concept of dynamic regions would be just as effective, if not more, than sending all raw data or partially processed data, that is, DOA estimates, back to the gateway or sink for analysis. In this set of experiments, given the limitations of the simulator used, estimation of query processing times and generation of random events were not possible. Instead, metrics not dependent on time, like packet size, hop count and number of packets transmitted were used to compare the efficiency and feasibility of the approach as compared to that taken in the VoxNet application in the field.

Simulation Scenario and Setup: For all simulations, a number of controls were maintained:
- The deployment of nodes in the simulator mirrored the actual real-life deployment topology. Nine nodes including the sink were positioned in the simulation window at the same relative locations as they were positioned in the actual deployment.
- All nodes were initialized with an equal and consistent amount of energy.
- Radio broadcast range was set at a 200 unit radius (a unit corresponding to a meter). This was the broadcast range of nodes in the VoxNet deployment.
- Each node's location was unique within the two-dimensional plane.
- A list of acoustic events, the ID of nodes which should detect these events and the intervals at which these events are to occur was defined prior to the start of each simulation. Lists contained between 3 and 5 events.

5.3.1. Region Based Query Resolution

A continuously running, event detection task (referencing the event list) is implemented on all nodes. At each time interval (a system clock tick), each node checks the event list. If an event for that node exists and the current time matches the scheduled event an acoustic event is triggered.

The detecting node logs the time of the event and checks if it has any related inquiries in its 'inquiry cache'. An inquiry indicates that another node has previously detected a possibly correlated event and has requested and is possible awaiting a response. If a node with an inquiry message in its inquiry cache detects a *related* event (that is, it is within the valid time slot) it sends a response to the node that sent the inquiry. If the node detecting the event does not have any inquiries, it sets itself as a region leader and sends an *inquiry* message to its one hop neighbours. This inquiry message contains the ID of the node that has detected the event and the event's timestamp. A node receiving this message registers the inquiry if it has not had an event that matches that inquiry. A matching or related event is one that falls within a time slot which was determined practically through using the simulator. If the node detecting the event does have an inquiry, it checks to see if its event has occurred within the *valid time slot* (a time range is used to indicate whether two or more signals are correlated and therefore considered as relating to the same acoustic event; this is referred to as the *valid time slot*) and a 'DATA' message is transmitted to the node who sent the inquiry message (the region leader). Any event occurring out of the valid time slot will not be sent to the region leader in response to its inquiry message.

The region leader upon receipt of a DATA message, records that message and if it has already received the minimum (3) required or more sends the result to the sink via a RESULT message. In the simulation, if the region leader has already received 3 messages it still waits for a period of time before sending a result on. The reason is that the greater the number of messages the better the accuracy of the measurement (this is also the model used in the VoxNet system.) The time a region leader waits, again, was determined through use of the simulator although in the real life deployment this value was determined experimentally and is referred to as a fuzz factor.

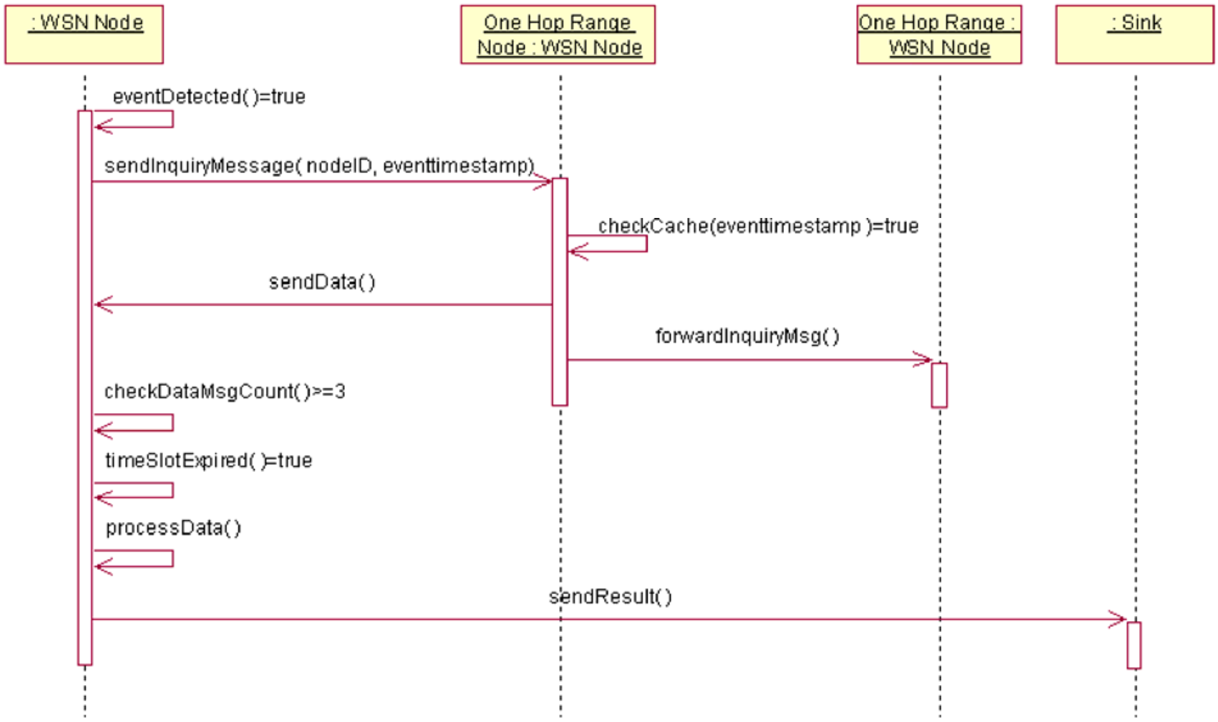

Fig. 11. Region-based approach to processing of acoustic data.

The simulation was allowed to run until either a result was sent to the sink or if the time for detection of a particular event had elapsed. For example, an insufficient number of detections (less than 3) were detected within the valid time slot. The results of this simulation in comparison to the in-the-field results are analyzed in Sections 5.4 and 5.5.

5.4. Efficiency of Region-based Querying

The results obtained indicate that regions **can** be used to support in-network query processing. In analyzing how much more efficient, if at all, the region-based approach is as compared to the in-network aggregation and centralized approaches, energy efficiency is evaluated using one parameter, that is, the number of messages generated in returning a query response to the sink.

The results of running a number of 50-node simulations indicate that on average region-based query processing resulted in an almost 86% decrease in the number of transmissions over the centralized approach and a 68% decrease on average over the approach using in-network aggregation.

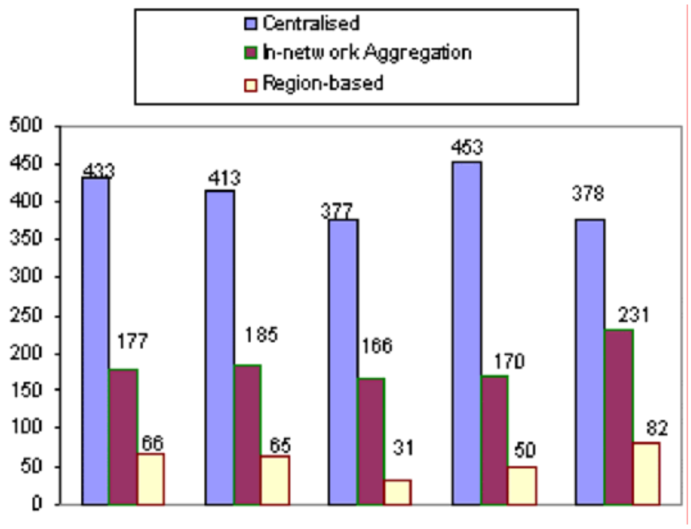

Fig. 12. Comparison of number of messages required for query resolution in a 50-node network.

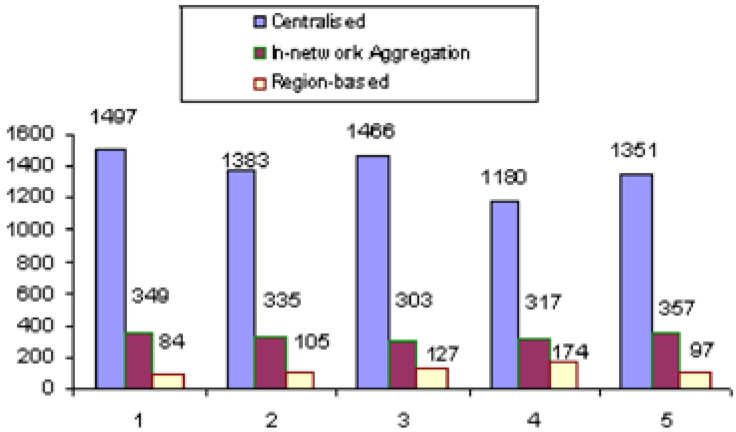

Fig. 13. Comparison of number of messages required for query resolution in a 100-node network.

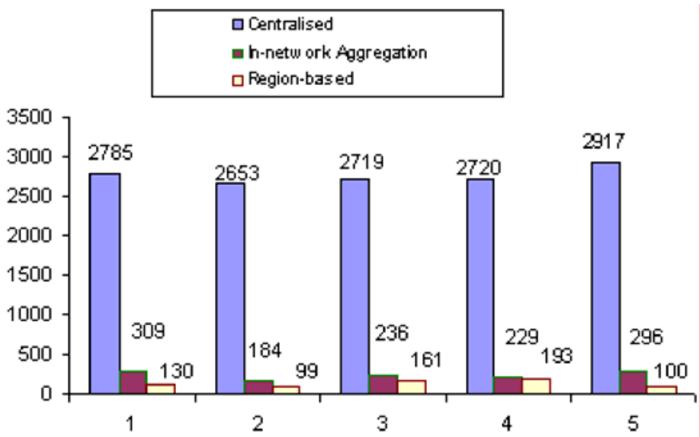

Fig. 14. Comparison of number of messages required for query resolution in a 150-node network.

For a 100-node simulation the results were equally impressive with over 91% reduction over the centralized approach and an almost 65% decrease when compared to the simulation using in-network aggregation. For a 150-node simulation the results for the region-based approach were more marked when compared to the centralized approach with over 95% reduction in the number of messages while there was an over 45% decrease when compared to the simulation using in-network aggregation.

The results although showed that the efficiency of the region-based querying approach increased as the size of the network increased when compared to the centralized approach, going from 86% to 91% to 95% for a 50, 100 and 150 node network respectively. In comparison to the in-network aggregation approach, however, this was not the case. Although communication was less, the relative percentages showed an overall decrease, going from 68% to 65% to 45% for a 50, 100 and 150 node network respectively. It would have been interesting to run these algorithms on even larger networks in determining whether this trend would continue, however, the limitation of the simulator used made this impossible at this time. Based on these results, however, although feasible, the scalability of the region-based approach is an issue to be considered and examined more closely in future work.

5.5. Effectiveness of Region-based Querying

Again, the results obtained indicate that regions **can** be used to support in-network query processing and are a viable approach in the context of a real-life deployment scenario. In analyzing the effectiveness and feasibility of the region-based processing approach compared to that used in the VoxNet application a number of measures were taken. First, the number of data packets was measured and along with the packet size was used to calculate the total data transmission required for query resolution. This was done in multiple runs for cluster/region sizes of 3 and 4 nodes in the case of the VoxNet and region-based approaches respectively, and an average taken. In addition, the average total data transported was calculated for the VoxNet system in scenarios where all raw data was sent to the sink and also in cases where in-network processing for DOA estimation was carried out.

The results, displayed in Fig. 15, clearly indicate that in terms of data transmission savings the region-based approach exhibits an advantage over the query processing approach used in the VoxNet system. This was evident even when some in-network processing was carried out. For a 3 and 4 node region there was approximately a 28% and 38%, reduction respectively in the amount of data transmitted over the VoxNet approach with in-network processing. The decrease is even more marked with figures of 61% and 67% for the VoxNet processing approach with no in-network analysis.

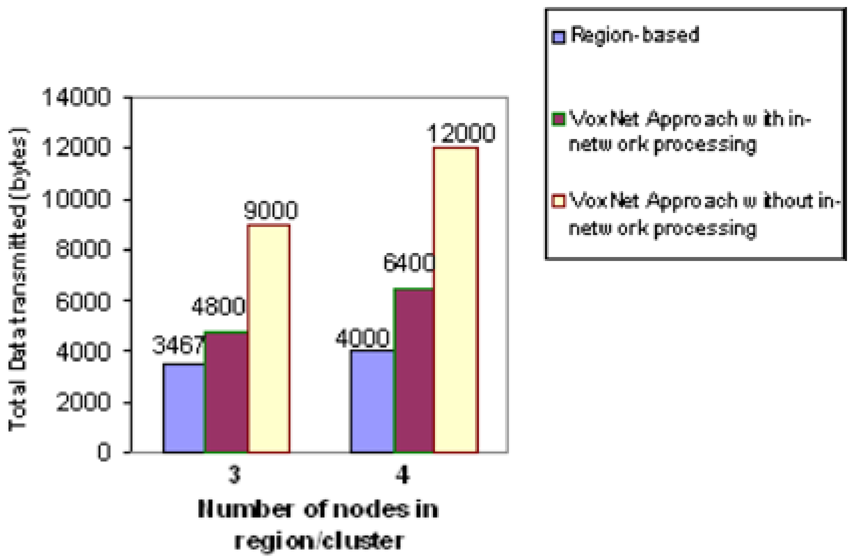

Fig. 15. Total amount of data transmitted in query resolution.

The simulation again confirmed the feasibility of the approach although a number of issues need to be investigated in future work. One, as identified before, is the scalability of the approach. In this simulation the network was quite small (9 nodes) and the regions created as a consequence also limited in size. An interesting exercise for the future would be to investigate the approach using larger regions and also to test the region query processing algorithm in the field.

6. Conclusions

An extensive literature review has shown the absence of complex query mechanisms in the query processing systems in use today. The review did highlight, however, applications in which these queries could be used along with in-network processing to both extend and improve WSNs' deployment value.

Both the usefulness of complex queries and the feasibility of using the proposed logical abstractions (called regions) to facilitate query processing in WSN systems have been positively assessed. The preliminary experiments have produced promising results but also highlighted a number of areas that present the community with open research questions: What are the means for efficient query-based region setup? How can dynamic regions be implemented which can change while query processing is occurring? (This becomes extremely important in cases where regions are established over an attribute that is changing with time); What is the network size/scale at which the cost of region set-up is justified? (Combining it with cluster formation to reduce energy consumption is perhaps one approach); For simplicity, the work here considered a scenario where a node could only be a member of a maximum of one region. Ideally, nodes should be able to be members of multiple regions. A node's single region membership can be an issue in facilitating the processing of multiple queries simultaneously or processing complex queries containing subqueries with additional attributes of interest. Moreover, here, once a response was sent to the sink the querying process terminated. In future work, the idea of the sink issuing further queries needs to be investigated as well as inter-region communication. Finally, what are the hardware support requirements to enable a system such as the one here be reliably deployed?

This is clearly a fruitful domain and although much has been achieved, it is the authors' belief that there is much still to be done to make WSN information extraction acceptable and accessible to a wider audience.

References

[1]. T. Daniel, R. M. Newman, E. Gaura, and Mount S, Complex query processing in wireless sensor networks, In *Proc. 2nd ACM International Workshop on Performance Monitoring, Measurement, and Evaluation of Heterogeneous Wireless and Wired Networks, PM2HW2N'07*, 2007, pp. 53–60.

[2]. Argo. Argo: part of the integrated global observation strategy, 2007, www-argo.ucsd.edu

[3]. A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, J. Anderson, Wireless sensor networks for habitat monitoring, In *Proceedings of the ACM International Workshop on Wireless Sensor Networks and Applications*, Atlanta, GA, United States, 2002, pp. 88–97.

[4]. K. Martinez, R. Ong, J. Hart, Glacsweb: A sensor network for hostile environments, In *Proceedings of 1st Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*, Santa Clara, CA, United States, 2004, pp. 81–87.

[5]. Grape Networks, Grape networks' climate genie, 2007, http://findarticles.com/p/articles/mi_m0EIN/ is_2007_Nov_12/ai_n21094407

[6]. M. Hirafuji, T. Fukatsu, T. Haoming, T. Watanabe, S. Ninomiya, A wireless sensor network with field-monitoring servers and metbroker in paddy fields, In *Proceedings of World Rice Research Conference*, Tokyo and Tsukuba, Japan, 2004.

[7]. Ning Xu, Sumit Rangwala, Krishna Kant Chintalapudi, Deepak Ganesan, Alan Broad, Ramesh Govindan, and Deborah Estrin, A wireless sensor network for structural monitoring, *In Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, Baltimore, MD, United States, 2004, pp. 13–24.
[8]. Jerome Peter Lynch, Kincho H. Law, Anne S. Kiremidjian, Thomas Kenny, and Ed Carryer, A wireless modular monitoring system for civil structures. In *Proceedings of SPIE - The International Society for Optical Engineering*, Los Angeles, CA, United States, 2002, pp. 1–6.
[9]. Codeblue: Wireless sensor networks for medical care, 2007, http://www.eecs.harvard.edu/~mdw/proj/codeblue
[10]. AID-N. Aid-n Project 2007, http://www.aidn.org.au
[11]. Kim Sukun, Pakzad Shamim, Culler David, Demmel James, Fenves Gregory, Glaser Steven, and Turon Martin, Health monitoring of civil infrastructures using wireless sensor networks, In *Proceedings of Information Processing in Sensor Networks, IPSN 6th International Symposium,* April 2007, pp. 254-263.
[12]. B. Son, Yong-sork Her, and Jung-Gyu Kim, A design and implementation of forest-fires surveillance system based on wireless sensor networks for South Korea mountains, *International Journal of Computer Science and Network Security*, 6, 9B, 2006, pp. 124–130.
[13]. Carl Hartung, Richard Han, Carl Seielstad, and Saxon Holbrook, Firewxnet: A multi-tiered portable wireless system for monitoring weather conditions in wildland fire environments, Vol. 2006 of *4th International Conference on Mobile Systems, Applications and Services (MobiSys 2006),* Uppsala, Sweden, 2006, pp. 28–41.
[14]. Narayanan Sadagopan, Bhaskar Krishnamachari, and Ahmed Helmy, Active query forwarding in sensor networks. *Ad Hoc Networks*, 3, 1, 2005, pp. 91–113.
[15]. Atish Datta Chowdhury and Shivashankar Balu, Consensus: A system study of monitoring applications for wireless sensor networks, *Proceedings - Conference on Local Computer Networks, LCN, Tampa, FL, United States*, IEEE Computer Society, Los Alamitos, CA United States, 2004, pp. 587–588.
[16]. I. Carreras, I. Chlamtac, H. Woesner, and H. Zhang, Nomadic sensor networks, In *Proceedings of the 2nd European Workshop on Wireless Sensor Networks, EWSN 2005*, Istanbul, Turkey, 2005, pp. 166–175.
[17]. H. Karl and A. Willig, A short survey of wireless sensor networks, *Technical TKN-03-018*, Telecommunication Network Group, Technische Universitat, 2003.
[18]. I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, Wireless sensor networks: a survey, *Computer Networks*, 38, 4, 2002, pp. 393–422.
[19]. Ali Iranli, Morteza Maleki, and Massoud Pedram. Energy efficient strategies for deployment of a two-level wireless sensor network, In *Proceedings of the International Symposium on Low Power Electronics and Design* , San Diego, CA, United States, 2005, pp. 233–238.
[20]. Gilman Tolle, Joseph Polastre, Robert Szewczyk, David Culler, Neil Turner, Kevin Tu, Stephen Burgess, Todd Dawson, Phil Buonadonna, David Gay, Wei Hong, A Macroscope in the Redwoods, *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*, ACM Press, New York, NY, USA, 2005, pp. 51-63.
[21]. Phil Buonadonna, David Gay, Joseph M. Hellerstein, Wei Hong, and Samuel Madden, Task: Sensor network in a box, In *Proceedings of the 2nd European Workshop on Wireless Sensor Networks, EWSN 2005* , Istanbul, Turkey, 2005, pp. 133–144.
[22]. S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, Tinydb: an acquisitional query processing system for sensor networks. *ACM Transactions on Database Systems*, 30, 1, 2005, pp. 122–173.
[23]. K. Akkaya and M. Younis, A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks*, 3, 3, 2005, pp. 325–349.
[24]. M. Younis, M. Youssef, and K. Arisha, Energy-aware routing in cluster-based sensor networks, In *Proceedings of the 10th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems* , 2002, p. 129.
[25]. Nauman Israr and Irfan Awan, Multihop clustering algorithm for load balancing in wireless sensor networks, *International Journal of Simulation: Systems, Science and Technology*, 8, 3, 2007, pp. 13–25.
[26]. C. Chen, J. Ma, Ke Yu, Designing energy-efficient wireless sensor networks with mobile sinks, 2006, http://www.sensorplanet.org/wsw2006/1_Chen_WSW06_final.pdf
[27]. P. Gupta and P. R. Kumar, The capacity of wireless networks, *Information Theory, IEEE Transactions on*, 46, 2, 2000, pp. 388–404.
[28]. Rahul C. Shah, Sumit Roy, Sushant Jain, and Waylon Brunette, Data mules: Modeling and analysis of a three-tier architecture for sparse sensor networks, *Ad Hoc Networks*, 1, 2-3, 2003, pp. 215–233.

[29].Gokhan Mergen, Qing Zhao, and Lang Tong, Sensor networks with mobile access: Energy and capacity considerations, *IEEE Transactions on Communications*, 54, 11, 2006, pp. 2033–2044.

[30].Yong Yao and Johannes Gehrke, The cougar approach to in-network query processing in sensor networks, *SIGMOD Rec.*, 31, 3, 2002, pp. 9–18.

[31].S. Ramanathan and M. Steenstrup, A survey of routing techniques for mobile communications networks, *Journal of Special Topics in Mobile Networks and Applications (MONET)*, 1, 2, 1996, pp. 89–104.

[32].J. N. Al-Karaki and A. E. Kamal, Routing techniques in wireless sensor networks: a survey, *IEEE Wireless Communications*, 11, 6, 2004, pp. 6–28.

[33].B. Krishnamachari, D. Estrin, and S. Wicker, Modelling data-centric routing in wireless sensor networks, In Proceedings of *IEEE InfoCom*, 2002.

[34].R. Eskicioglu, S. Ahmed, S. Hussain, A query processing architecture for sensor networks, In *Proceedings of WICON Workshop on Information Fusion and Dissemination in Wireless Sensor Networks (SensorFusion)*, Budapest, Hungary, 2005.

[35].J. Gehrke, S. Madden, Query processing in sensor networks, *IEEE Pervasive Computing*, 3, 1, 2004, pp. 46–55.

[36].W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, Energy-efficient communication protocol for wireless microsensor networks, *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, Maui, HI, USA, Vol. 2, pp. 10, 2000.

[37].S. Madden, R. Szewczyk, M. J. Franklin, and D. Culler, Supporting aggregate queries over ad-hoc wireless sensor networks, In *Proceedings of 4th IEEE Workshop on Mobile Computing Systems and Applications* , Callicoon, NY, USA, 2002, pp. 49–58.

[38].G. J. Pottie and W. J. Kaiser, Wireless integrated network sensors, *Communications of the ACM*, 43, 5, 2000, pp. 51–8.

[39].S. Madden, M. J. Franklin, J. M. Hellerstein, W. Hong, The design of an acquisitional query processor for sensor networks, In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, San Diego, CA, United States, 2003, pp. 491–502.

[40].John Heidemann, Fabio Silva, Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin, and Deepak Ganesan, Building efficient wireless sensor networks with low-level naming, *Operating Systems Review (ACM)*, Banff, Alta., Canada, Vol. 35, 2002, pp. 146–159.

[41].Yang Xiaoyan, Lim Hock Beng, M. Tamer, zsu, and Tan Kian Lee, In-network execution of monitoring queries in sensor networks, In *Proceedings of SIGMOD Conference, 2007*, pp. 521-532.

[42].K. Ramamohanarao, L. Kulik, S. Selvadurai, B. Scholz, U. Roehm, E. Tanin, A. Viglas, A. Zomaya, and C. Leckie, A survey on data processing issues in wireless sensor networks for enterprise information infrastructure, May 2006, http://www.eii.edu.au/files/EIIAnnualRpt2006.pdf

[43].S. Madden, M. J. Franklin, J. M. Hellerstein, and Hong Wei, Tag: a tiny aggregation service for ad hoc sensor networks, In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI'02)*, Boston, MA, USA, 2002, pp. 131–46.

[44].C. Intanagonwiwat, R. Govindan, and D. Estrin, Directed diffusion: a scalable and robust communication paradigm for sensor networks, MobiCom 2000, In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, Boston, MA, USA, 2000, pp. 56–67.

[45].Chalermek Intanagonwiwat, Deborah Estrin, Ramesh Govindan, and John Heidemann, Impact of network density on data aggregation in wireless sensor networks, In *Proceedings of International Conference on Distributed Computing Systems*, Vienna, Austria, 2002, pp. 457–458.

[46].O. Younis and S. Fahmy, Heed: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks, *IEEE Transactions on Mobile Computing*, 3, 4, 2004, pp. 366–379.

[47].Guo Bin, Li Zhe, and Meng Yan, A dynamic-clustering reactive routing algorithm for wireless sensor networks, In *Proceedings of 1st International Conference on Communications and Networking in China, ChinaCom '06* , Beijing, China, 2007, pp. 4149783.

[48].A. M. Ali, Yao Kung, T. C. Collier, C. E. Taylor, D. T. Blumstein, and L. Girod, An empirical study of collaborative acoustic source localization, In *Proceedings of the 6th International Symposium on Information Processing in Sensor Networks* , Cambridge, MA, USA, 2007, pp. 41–50.
