A Framework for Semantic Sensor Network Services

Lily Li and Kerry Taylor

CSIRO ICT Centre
GPO Box 664, Canberra, ACT 2601, Australia
{lily.li, kerry.taylor}@csiro.au

Abstract. We propose that a semantic service-oriented approach is one of the best techniques to cope with challenges in wireless sensor network (WSN) applications. This paper offers a framework for sensor network services that aims to improve query processing. We expect this framework will address current challenges and issues preventing the wider uptake of WSN technology. More specifically, we propose a semantic service-oriented framework with a focus on query processing to allow distributed end-users to request streams of interest easily and efficiently, based on the principle of pushing the query down to the network nodes as much as possible. As such, the lifetime and utility of the sensor network will be maximised, ultimately leading to the success of WSN deployments. The importance of semantics, which aims to support sensor capability modelling and query writing has been highlighted. On the other hand, query rewriting is emphasised followed by examples to illustrate that query rewriting can significantly contribute to the overall power efficiency of WSNs.

1 Introduction

Sensor network applications have been deployed for monitoring space, things and the interactions of things with each other and the encompassing space [1]. Future sensor networks will have sensors with different capabilities wired into a Sensor Web [2] to perform extensive monitoring for timely, comprehensive, continuous and multi-modal observations [3].

Despite existing different domain-specific deployments and the heterogeneity of sensor networks and sensing data, sensor networks share a common feature: they all collect and periodically transmit information from some set of sensors, and they all must carefully manage limited power and radio bandwidth to ensure that essential information is collected and reported in a timely fashion [4]. From a data storage point of view, one may think of a sensor network as a distributed database that is able to conduct query processing. This has led to the design and implementation of query processing techniques, a very popular topic in the literature [4,5,6]. However, query processing in sensor networks works differently to that in a traditional DBMS given that the former is highly resource constrained (e.g. limited energy, memory and computation ability). Furthermore, the volume of the sensing data can be very variable (e.g. depending on sampling rate variations) within a time period and the access to this data may be delayed arbitrarily due to the motion of sensors. These issues create new challenges for query processing and more needs to be done to realise the potential for sensor network applications. Further, as successful sensor network applications rely heavily on effectively using a large range of heterogeneous data resources including streaming data acquired from sensor networks and historical data stored in databases, the development of a technology framework to enable this is highly demanded.

The primary contribution of this paper is an identification of the services that may be offered to the user community of WSNs through a semantic service-oriented framework. We focus in particular on dealing with query processing elegantly through that framework. Employing both declarative semantics and query rewriting techniques are important features of the framework.

It is natural to choose a declarative language [4,7,8] to describe a query, as declarative queries offer both an easy-to-use interface and energy-efficient execution substrate [4]. More importantly, they open up the possibility for optimisation algorithms to transparently handle efficient access strategies and performance improvements enabled by taking account of current state of the sensor network including concurrently or previously executed queries.

The paper is organised as follows. Section 2 is the brief review of related work. Section 3 outlines the design challenges and issues. Section 4 presents our framework that relies on a service-oriented computing approach enriched with semantic modelling. In Section 5, the main features of the framework, semantic query rewriting will be discussed followed by two examples to illustrate the significant impact of query rewriting on the overall power efficiency of WSNs. Finally, we conclude this paper in Section 6.

In this paper, terms such as sensor networks and wireless sensor networks will be used interchangeably unless otherwise specified.

2 Related Work

Sensor networks promise to revolutionise sensing in a wide range of application domains. However, wide acceptance and deployment has not yet been seen mainly because current deployments are closed, single-purpose systems. Tools and standards for easy open access are still unseen. It is well known that programming sensor networks is very hard for end-users who have genuine application requirements for sensor networks. On the other hand, end-users should be shielded from the low-level details and difficulties of sensor networks. While most of the research work in sensor networks to date has been focused on device engineering design and communications and networking questions, interesting work in the near future is most likely to be concentrated on leveraging end-users' workloads in sensor network applications by Web services in a publish-find-bind service-oriented fashion. End-users must be supported for easy access and more control over the sensor network, especially with real time data manipulation.

This section presents recent work that attempts to ease non-trivial programming and reprogramming sensor network tasks.

Different types of queries in sensor networks have been studied in [9,10]. TinyDB [6] is one of the most successful works in sensor network query processing. It offers a SQL-like query interface to raw sensor data. It is advantageous to express queries to a sensor network at a high level logical predicate language as such an abstraction has greatly eased the use. The objective of this paper is to make this abstraction more concise by taking query rewriting techniques into account.

Cougar [8] adds a `query layer` to the protocol stack to accept queries in a declarative language that is then optimised to generate efficient query execution plans with in-network processing. A query plan is constructed by `flow blocks`. The optimiser determines the exact number of flow blocks and interaction between them. However, as pointed out in [8], the creation of the best query plan for an arbitrary query is a hard problem, only little work in query processing has been done in Cougar.

Several other pioneering works have set good examples for the use of sensor services in the presence of Web services and Semantic Web technologies. Existing prototype applications have shown that the heavy burden of programming sensor networks can be alleviated if the underlying infrastructure is flexible enough. Web service technologies are one such enabler that can be used to derive more value from sensor data by making sensor data more accessible to a wider group of people, when combined with services for statistical modelling and machine learning, for example. However, the current state of the art is far from the vision of a highly available, high-performance, easy-to-use sensor web. The following is a brief introduction to these relevant works.

Sensor Web [2], first proposed by NASA in 2001, has received tremendous attention. It is a revolutionary concept toward achieving collaborative, coherent, consistent and consolidated sensor data collection, fusion and distribution. The project GeoSWIFT (*http://sensorweb. geomatics.ucalgary.ca/default.html*) is an exemplar. It offers open geo-spatial sensing services for sensor web developed by sensorWeb@GeoICT [3]. It aims to build a geo-spatial infrastructure to connect distributed sensor networks for the sharing, access, exploitation, and analysis of sensing information. It focuses on a web service approach to answer a request (i.e. HTTP GET request) in an SOA style. However, it does not address efficient query rewriting, and its query language (conjunctive attribute-value style language) relies on a primitive data model that is not amenable to interoperability with other services thus it suffers from common problems of WSNs.

IrisNet [11] (*http://www.intel-iris.net/*) is a general-purpose software infrastructure that supports tasks common to services such as collecting, filtering, and combining sensor feeds, and performing distributed queries (via sensor agent nodes and organising agent nodes). It provides an opportunity to deal with a query over the Internet. It proposes that its sensing service is capable of collecting filtered sensor readings in a database that end-users can query. Although the IrisNet architecture allows filtering code to be uploaded to sensing devices, there is no discussion about sampling control. In order for a sensor web architecture to be aware of energy consumption, an energy efficient design approach should be emphasised. Furthermore, semantics has not be thoroughly explored in IrisNet.

In contrast, SONGS [12,13] advocates semantics by proposing a semantic-service-oriented sensor information system. The system may take advantage of application domain knowledge to optimise its resource utilisation in collecting, storing, and processing data. As a result, the creation of a semantic information hierarchy and the implementation of the semantic transformations are at the core of their work. However, this work focuses on the use of a larger semantically-rich system of sensors, into which sensor devices are embedded, but does not address the requirements for general purpose sensor network services that support such embedding of WSNs.

Similar efforts from the grid computing research community [14] are worth noting too. The focus of the reported work is on sensor resource management [15] to provide middleware support to the connection and share of heterogeneous sensor resources. The work reported at the website (*http://nicta.com.au/research /projects/nicta_open_sensorweb_architecture*) is such an example.

From a standardisation perspective, the efforts from OpenGIS Consortium (*http://www.opengeospatial.org/*) is worth mention. A package of standards called Sensor Web Enablement (SWE) has been developed to define service-oriented interfaces to sensor network services, most strongly influenced by requirements arising for earth observation remote sensing services. The SWE specifications support data and service interoperability at a syntactic level: client tools are specifically designed to parse the standardised data model, and interpretation of the content is left entirely to service providers on one hand and end-users on the other. In order to evaluate these specifications, we deployed SWE services over WSN data collected at an experimental site in Queensland and conducted an evaluation in the context of the OGC testbed programme. SWE service implementations from 52North were used (available from *http://52north.org/*). Referring to the most relevant specification, the Sensor Planing Service (SPS) for example, we found that it suffered from limitations such as an inability to model the relationship between input and output parameters. The current SPS permits neither the phrasing nor the answering of query such as "Provide the parameter A if parameter B is greater than 5 and parameter C is prior to 2 hours from now"; the kind of query we take for granted in modern query languages like SQL. More information about this testbed programme can be found at the website (*http://www.opengeospatial.org/*).

To our knowledge, no work has been reported in WSNs that considers the logic structure of the problem, sensor capability and environment setting in query rewriting. This is the first time that query rewriting has been addressed in sensor service design.

We believe that the success of applying semantic web services to WSN applications will significantly contribute to the growing deployment of large-scale sensor networks, and leading to a broad scope in sensor network applications.

3 Design Challenges

We discuss issues in the design of semantic sensor network services in this section. As this paper is focused on providing semantic sensor network services to service clients from a broad network community, we avoid discussing typical WSN internal issues such as network connectivity, MAC protocols and routing algorithms, but concentrate on issues related to the service design. In particular, we aim to hide those WSN-specifc issues from the broader network users, and propose embedding query rewriting techniques within the sensor network service to achieve this. We do not discuss broader web service issues here (such as security, scalability and quality of service) as we anticipate that the progress of web service research will address these issues in a more general context.

We expect a sensor network service to be the (technical) custodian of a sensor network: to be responsible for local network management, data management, query processing and response, and information security. We expect the service to accept high level requests for data and for the service to be able to map that request to answers that the sensor network can handle, while enabling the service user to be as ignorant as possible about the sensor network capability, technology, topology, control language and current state. This "ignorance" will enable service clients, such as simple web pages, specialist user-oriented interactive GUIs and composition and integration engines (including workflow engines) to interact with a wide range of such services in a common way. Note that for our purposes we are focusing on sensor services, not actuator services, although we include within our scope the tasking of sensors in order to take measurements – for example the movement or rotation of a sensor in order to take a measurement. In the latter case, the need for movement is only implicit within the request for data and is not a separately identified request for actuation.

3.1 Data Persistence

In common with the design principles for TinyDB [6], we believe that the management and archiving of sensor network data is the responsibility of a unitary sensor network service, rather than some external services. This enables the scope of a service to be represented and understood by its user community irrespective of the temporal nature of information. It also means that quality of service guarantees (such as reliability, response time, cost etc.) can be offered according to user priority, financial incentive, or other features in a uniform way.

When a sensor network service accepts a query it will need to recognise whether it is capable of answering some or all of the query, and if so to determine whether the answer could be partly or fully retrieved from persistent data hosted by the service, presumably previously collected from the sensors controlled by the service's network. Alternatively, the query may be partly or fully answered by the service recognising that its network is already configured to collect the desired data and all it needs is to ensure is that the data is returned to the new requestor (possibly summarised or filtered first). Finally, there may be a remaining part or all of the request that can only be answered by reconfiguring or reprogramming the sensor network.

We envisage this capability to be offered by a query-rewriting algorithm. By describing persistent data and currently collecting data as "views", rewriting techniques such as [16] can be used to split an incoming query into the appropriate components to be handled by each aspect of the service.

3.2 Sensor Network State

Along with management of data persistence arising from sensor network measurements, we would like to see a sensor network service managing internal sensors and network state while offering a simpler stateless service for its clients. This will make service interaction easier for the clients, and enables optimisation within the service to meet multiple client requests. For example, for mobile sensors we would prefer a sensor request to state the spatial coordinates (and if necessary, temporal coordinate) required for a measurement, and have the sensor network service internally plan the optimal movement of the sensor devices to meet multiple requests, resolving conflicts by a priority-based scheduling method if necessary. Furthermore, we have already proposed that the service recognises requests for reuse of data which the sensor network is previously configured to collect: this capability could also be seen as recognition and management of sensor network state.

3.3 Events and Responses

As hinted in the discussion about the need to offer coherent access to persistent data, some requests for data from the service will be answerable with a synchronous response. This might be appropriate for a query for last year's average daily rainfall, for example. Some requests will naturally behave more as a standing query (e.g. a request for next year's average daily rainfall) and may be more appropriately dealt with as an asynchronous response. Further, an event-driven response (e.g. daily rainfall for the next three years, a day at a time) will also be needed. More generally, it seems that any query may require a combination of these approaches and an appropriate interaction design will be required.

3.4 Programming

Many modern sensor network technologies support over-the-air sensor node programming (e.g. the TinyOs Deluge protocol [17]). A sensor network service should be able to accept tasks from wider networks users for deployment onto the network. However, in line with our desire to hide heterogeneity we suggest a declarative, data-oriented interface to the wider user community is more appropriate than a specialist imperative programming language. We are working to develop a translation process from a high level declarative predicate language phrased over a "view", to the executable Snlog language[1] of the DSN architecture [7].

With the declarative language, it is possible to solve the problems yet unsolved by IrisNet. In particular, we can take account of local knowledge about the current state of the sensor network, such as network topology, residual power, alternative routing protocols, local redundancy, and node-specific sensing capability to push queries into network nodes for efficient execution. Results demonstrating this are reported later in this paper.

3.5 Capability Modelling

Whilst we wish to enable a semblance of homogeneity of sensor networks to an outside user for ease-of-use, inevitably some fundamental differences in sensor networks will remain and must be made visible to the user community to enable their use. We expect that the capability of a network in terms of the observable phenomena and how they might be measured must be declared by the sensor network to its clients. Although this recognition of need is aligned with the approach adopted through the OGC's Sensor Web Enablement suite of standards (*http://www.opengeospatial.org/projects/groups/sensorweb*), we propose that machine executable formal ontologies, such as those based on the W3C's Web Ontology Language (OWL) are the appropriate tools for describing sensor network capability. By employing reasoners[2], we need to rely less on both specialised web clients (in which the knowledge of the capability is embedded in code) and human readability (in which the knowledge relies entirely on a human interpretation) [18].

3.6 Power Management

Power management is a critical issue in a WSN. The typical power management design goal is to meet the required constraints while minimise the energy consumed. Sensor applications have to make trade-offs based on the energy consumption policy (Fig. 1) between the quality of a service (e.g. sensor operations) versus conserving energy, efficiency versus power consumption. It is in the best interest of power management to have each node transmit at the lowest possible power while preserving network connectivity but activating only the necessary number of sensor nodes for a particular task at any particular moment. It is expected that a dedicated power consumption policy should be enforced and met for a specific task.

3.7 QoS

The QoS provisioning, usually described very abstractly, is a crucial issue to provide demanded resources effectively and efficiently. Resource reservation and

[1] Snlog is a dialect of Datalog developed for sensor netowrk programming. Details and examples can be found at the website (*http://db.cs.berkeley.edu/dsn/*).

[2] For example, Racer (*http://www.racer-systems.com/products/tools/index.phtml*) and FaCT++ (*http://owl.man.ac.uk /factplusplus/*).

resource allocation should be enforced aligning with either single query or continuous query and the unique characteristics of WSNs. The QoS specification should also be able to offer differentiated QoS and data quality to different users.

3.8 Security

We cannot propose a shared sensor network service in the absence of addressing the needs for sensor network security. Assuming the sensor network itself offers some local methods for securing information within its scope (for example, [19]), a sensor network service must be able to protect its privacy and integrity in the wider internet context. For a shareable, reusable sensor network we propose Web Service-style role based access control over XML structures [20] coupled with executable privacy policies [21]. Privacy policies would apply to both persistent and real-time data, and to both data access and to scarce resource access (e.g. permission to program the sensor network to collect new measurements). They could be compiled to a run-time policy-enforcement point within the sensor network service.

4 Framework

In response to the challenges and partial solutions we have discussed in Section 3, we propose a framework (Fig 1) in this section, relying on a service-oriented computing approach enriched with semantic modelling. We will address the important features in the following.

1. **Service-oriented computing approach**
   The design of the proposed framework is based on the principle of service-oriented computing as we expect web service technologies would support high performance, scalability, reliability and availability of sensor services. The core of the sensor service is the service proxy, which is composed mainly of the `interaction handler`, `query manager`, and `WSN power manager`. Major components of each layer are shown in Fig. 1.

   The semantic sensor service, which acts as an interface between the WSN and the client, is in charge of the interactions between them (with `interaction handler`). Then it performs query processing (including `query rewriting` and `query planning`, with the support of `persistent store` and `semantic transformation`) and partial results integration - back to the client (by `query planning`). The success of the above functionalities cannot be achieved without concerning about scarce power resource. `WSN power manager` is designed to cope with energy management enforced by an appropriate `energy consumption policy` and corresponding `power management` strategies. Obviously, the `WSN fundamentals` is the foundation on that our work is built on.

   The extended framework also includes overarching concerns such as `QoS`, `security`, and `service management` that apply to all components in the

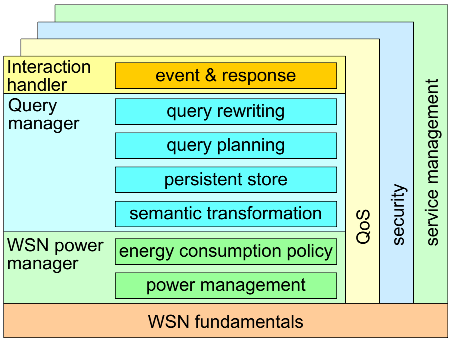

Fig. 1. The Semantic Sensor Network Service Framework

framework, in that `service management` refers to the management of sensor service life-cycle, including service registration, service invocation, and service closure.

2. **Semantic approach**
Sensor Web clients depend on registries and ontology repositories to make use of available sensor services. The repositories provide necessary information about the underlying sensor networks in terms of the phenomena, observation, measurements, sensor capabilities and different models to be used to facilitate the decision making when certain events occur. These repositories can be classified into two categories with the static category providing essential information about sensors (e.g. sensor capabilities) and sensor networks (e.g. sensor platforms), while the dynamic category reflecting the change of the sensor networks, topological change, for example.

   Generally, an ontology provides a medium for capturing and reusing the knowledge and experience gained from prior efforts, it thus leads to a greater level of automation at the semantic level. It is believed that ontologies and related semantic techniques and technologies will allow machines to interpret and understand (human-) agreements and formal policies one day in the future. As OWL is a highly sharable and reusable form of knowledge representation language, (sensor) ontologies will be represented in OWL. These ontologies can encode necessary information for query rewriting. A sensor network service should be able to describe what it can do and how to do it (i.e. sensor capability modelling) in terms of a formal OWL ontology. The recognition of need to provide a powerful "virtual device" (depending on available sensing devices) with differing sensor capabilities for different tasks (e.g. modelling or tasking) has attracted our attention. It will be included in the future work.

Different layers in the framework work together to maintain the integrity of sensor data and answer queries in an efficient way. Like other web services, the semantic sensor network service can be made available to be discovered and composed to support a wide variety of WSN deployments.

5 Query Rewriting

In this section, we will discuss query processing through the proposed framework, in which employing both declarative semantics and query rewriting techniques are important features.

By applying optimisation techniques during query processing, query rewriting can generate a rewritten query with minimised node transmission and therefore improved energy conservation. Below we first introduce declarative semantics, then unification-based propagation optimisation [22,23]. It will be illustrated by working through examples. A discussion which points out that the optimisation technique is particularly advantageous to some kinds of problems in query processing will follow.

5.1 Declarative Semantics and Query Rewriting

It is generally accepted that some sort of knowledge (the knowledge about sensors and the sensor network) can provide necessary information in query processing. The knowledge consists of the sensor hardware and software characteristics. The following code fragment (represents information about the current configuration of a sensor network) is an example in the WSN programming language, Snlog [7].

```
part 1:
--------------------------------------------------------------------------------------------
% initial messages of nodes
toTransmit(@1,0).
toTransmit(@2,0).
toTransmit(@3,0).
toTransmit(@8,0).

% the residual powers of node,
residualPower(@1,509).  %509units.
residualPower(@2,245).
residualPower(@3,455).
residualPower(@8,505).

% nodes' list
mgsList(@1, [0,2,4]). %{0,2,4} node IDs
mgsList(@2, [12,15]).
mgsList(@3, []).
mgsList(@8, []).

% timer setting of nodes
timerx(@1,2,2048).  % 2 minutes
timerx(@2,2,2048).
timerx(@3,2,2048).
timerx(@8,2,2048).
```

Now consider a sensor network program that relies on the configuration to perform a task including message routing details, as follows.

Example 1: "Generate a sequential number and transmit it to one of its neighbours (e.g. @*Next*), update its node list (the list of nodes that this particular node has sent messages to) and cost accordingly."

Suppose we have a top-level clause (i.e. the clause (1)) and the relevant clauses defined as follows. The whole program is composed of parts 1 and 2.

```
part 2:
-----------------------------------------------------------------------------------
message(@Next,Src,Dest,X) :- generatedMgs(@Src,X), updateNode(@Src,X,Result),
                             nextHop(@Src,Dest,Next).                        %...(1)
generatedMgs(@Src,Y) :- toTransmit(@Src,X), Y is X+1,residualPower(@Src,Z), Z>500,
                        timer(@Src,2,TimePeriod).                            %...(2)
timer(@Src,TimerID,TimePeriod) :- timerx(@Src,TimerID,TimePeriod).           %...(3)
updateNode(@Src,X,Result) :- mgsList(@Src,OldList), append(OldList,[X],Result),
                             forward(@Src,Result,Cost).                      %...(4)
forward(@Src,Result,NewCost) :- neighbour(@Src,Neighbours),
                             shortestPath(@Src,Node,Neighbours,Cost),
                             length(Result,Size), timer(@Src,2,TimePeriod),
                             NewCost is (Size*TimePeriod).                   %...(5)
-----------------------------------------------------------------------------------
```

- clause (1): the predicate *message*/4 is the logic consequence if the current node (i.e. *Src*) with the generated message is X and the routing detail (i.e. *nextHop*/3) is determined. That is, the next node (i.e. the *Next*) to send the message to is certain. At the same time, the node list will be updated accordingly.
- clause (2): the predicate *generatedMgs*/2 is defined as a increment of the message determined by the predicate *toTransmit*/2 if the residual power of the node @*Src* is greater than 500 units within the given *TimePeriod*.
- clause (3): the predicate *timer*/3 is defined to convert a *timerx* tuple into a *timer* tuple,
- clause (4): the predicate *updateNode*/3 is defined to update the node list. More specifically, the newly generated message will be appended to an existing list (i.e. the *OldList*) and the result (i.e. *Result*) will be sent away to a particular node defined by the predicate *forward*/3.
- clause (5): the predicate *forward*/3 is defined to update the cost. The new cost equals to the product of the length of the list (i.e. the size of the *Result*) and the time period (i.e. *TimePeriod*).

  For illustration purposes, the predicates *shortestPath*/4, *neighbour*/2 and *nextHop*/3 are defined as built-in predicates in Snlog.

The observation that the larger a program is, the more energy it may consume drive the selection of optimisation techniques in this situation to reduce redundancy. Two major approaches are available to be chosen from. One is "local computation" in which the code will be injected into the nodes as it was and the node itself is responsible for the computation (i.e. no specialisation). The other is "global computation" in which the code will be specialised before it is sent into a node. Our question is: " is there any difference between these two approaches when the power consumption of a node is concerned? ".

Now let us look at the "global computation" scheme. We are interested in using unification-based propagation technique [22,23] for it has potential to make a considerable improvement in terms of the performance because it is capable of reducing most of redundant computation at compile-time so that the computation complexity at nodes can be lessened or lifted greatly. We will not go to the detail in this paper, but highlight the effect of this technique in achieving efficiency instead. A prototype was developed to allow the rewriting to be achieved in an automatic manner. A systematic way about how this optimisation technique is performed will be discussed in another paper.

With the unification-based propagation technique, the top-level clause will be specialised into the following particular code:

```
the specialised code (Note that the variables have been renamed by the system):
--------------------------------------------------------------------------------
message(@_G1247,1,_G1244,1):- neighbour(@1,_G1253), shortestPath(@1,_G1261,_G1253,_G1263),
                             nextHop(@1,_G1244,_G1247).
message(@_G1210,8,_G1207,1):- neighbour(@8,_G1216), shortestPath(@8,_G1224,_G1216,_G1226),
                             nextHop((@8,_G1207,_G1210).
--------------------------------------------------------------------------------
```

This program entirely replaces both part 1 and part 2 given earlier. It is specialised from part 2 to take account of the configuration in part 1 but is much more compact.

Example 2: We now consider a generic program to answer a class of problems. It is: "Find regions with *sensorId* $> \$IntVar$ (e.g. $\$IntVar = 99$) and temperature rise of $X\%$ (e.g. 20%) in the last given time period (e.g. Δt) ".

Assuming a time window is denoted by $[t - \Delta t, t]$, and the memory at each node is sufficient to hold data streams during that period. One of critical steps is to avoid sending back unnecessary messages as much as possible and to detect unwanted messages as early as possible because the message size is a very important factor in power consumption. As such, it is clear only the sensors with sensor$Id > \$IntVar$ will be considered further. These sensors will then check the required temperature readings. Again, only the successful node will be asserted to the *location*/1, which is defined to hold the node information. The total cost of answering this particular query is approximated by the sum of costs at each node to check whether it is the required sensor ID ($Id = 0$ at the base station).

– The top-level clause is defined as follows:

```
go(@Next,Id) :- zone(@Src,Id,TH:TM:TS,TimePeriod,Lb,Ub,Percent),NextHop(@Src,Dest,Next).
```

– the predicate *zone*/7 is denoted by *zone*(*HostId*, *Id*, *EndTime*, *TimeDiff*, *LowerBound*, *UpperBound*, *Percent*). It is a relation which contains a tuple for each node. It is defined as follows:

```
zone(@Src,Id,TH:TM:TS,T,Lb,Ub,Percentage):-
               sensorId(@Src,Id),less(Lb,Ub),
               check_range(Ub,Id,Lb),
               reading(@Src,Id,TH:TM:TS,TempVal_1),
               integer(T),
               T1 is TH*3600+TM*60+TS-T,
               T2H is T1 // 3600,
               T2 is T1 mod 3600,
               T2M is  T2 //60,
               T2S is T2 mod 60,
               reading(@Src,Id,T2H:T2M:T2S,TempVal_2),
               TempVal_1 >= f_mult(TempVal_2,Percentage), % a built-in function in Snlog
               assert(location(Src)).
```

– the predicate *reading*/4 is a relation which contains tuples of sensor readings at each Src in the form of $reading(@Src, Id, TimeStamp, TemperatureVal)$. $TimeStamp$ is expressed in the form of H:M:S.
– the predicate $nextHop(@Src, Dest, Next)$ is a built-in.

Given the readings at all nodes (these readings have been omitted here due to space limitation), in this example, the top-level clause is specialised by unification-based program to: go(@_G1022,12):-nextHop(@12,_G1025,_G1022).

This is because among available nodes, only node 12 meets the requirement. Clearly, the specialised top-level clause is very concise (no irrelevant information, only need to deal with the built-in *nextHop*/3). The full advantages of the query rewriting technique will be discussed next.

5.2 Discussion

Generally speaking, the specialised top-level clauses have the following advantages over the original ones:

- the length of the specialised code is greatly shorter than that of the original one.
  Intuitively, this means fewer rules/facts will be fired for the same problem. The reason behind is that much of the computation has been performed thanks to the optimisation technique. Those predicates whose definitions are available at compile-time, such as the residual power of nodes, should provide sufficient information to instantiate relevant variables and the whole process is propagated throughout the code when the bindings of variables are computed. In the end, some computation and abstraction which can be performed at compile-time, have been completed.
- the specialised code is more efficient than the original one in the following aspects.
  - The clause has been simplified.
    By removing the superfluous call to `true`, the original top-level clause have been replaced by the new clauses which are very concise. The above specialised codes provide interesting insights into it.
  - The message will be sent only to the relevant nodes rather than to all nodes.
    Note that in reality, a WSN can have many sensors. As only the specific nodes (e.g. That is, the nodes 1 and 8 in example 1 and the node 12 in example 2) will be informed, there is no need for irrelevant nodes to wake up. More power can be saved because of less computation and transmission. In this way, we also achieved smart distribution of the code throughout the network as a result of specialisation.
  - The storage on nodes has been minimised considerably.
    It is obviously that only the relevant nodes need to consider the storage issue, not all nodes.
  - The execution performance has been improved considerably.
    Since the code has been specialised with much redundancy being removed, the node will only consider the run-time.

From the above discussion it is evident that the query rewriting technique has potential to enhance the efficiency of the WSN.

6 Conclusion

In this paper, we proposed a service-oriented framework to address some critical design issues and challenges of sensor network services. The sensor network services may allow end-users not only to take advantage of QoS, security and scalability which are promised by Web services, but query processing through the framework as well. We focus in particular on dealing with query processing through that framework.

We highlighted two important features of the framework. As energy efficiency is a main concern of the sensor network deployment, energy consumption at every phrase of the sensor network should be considered carefully. We have taken reducing redundancy as a starting point to demonstrate that applying query rewriting techniques at compile-time can improve the performance significantly for some problems as long as it is desirable to take advantage of the logic structure of the problem, input and static variables known a priori. Early progress from query rewriting has shown an efficient sensor network program can be obtained.

We plan to investigate query processing under the proposed framework in depth. Many issues need to be solved to develop a query optimiser with the presence of unpredictable WSN characteristics. We believe that the optimisation technique discussed in this paper will be constructive for sensor networks to deal with the tight energy and bandwidth limitation. We are also interested in sensor capability modelling.

Acknowledgment

The authors would like to thank David Ratcliffe for his help on the improvement of the unification-based propagation prototype.

References

1. Culler, D.E., Estrin, D., Srivastava, M.B.: Guest editors' introduction: Overview of sensor networks. IEEE Computer 37, 41–49 (2004)
2. Delin, K., Jackson, S.: The sensor web: a new instrument concept. In: Proceedings of the SPIE International of Optical Engineering, vol. 4284, pp. 1–9 (2001)
3. Liang, S.H.L., Croitoru, A., Tao, C.V.: A distributed geospatial infrastructure for sensor web. Computers & Geosciences 31, 221–231 (2005)
4. Gehrke, J., Madden, S.: Query processing in sensor networks. Pervasive Computer 3, 46–55 (2004)
5. Bonnet, P., Gehrke, J., Seshadri, P.: Towards sensor database systems. In: Proceedings of the 2nd International Conference on Mobile Data Management (January 2001)
6. Madden, S., Franklin, M.J., Hellerstein, J.M., Hong, W.: TinyDB: An acquisitional query processing system for sensor networks. Transactions on Database Systems (TODS) 30, 122–173 (2005)

7. Chu, D.C., Popa, L., Tavakoli, A., Hellerstein, J.M., Levis, P., Shenker, S., Stoica, I.: The design and implementation of a declarative sensor network system. In: The 5th ACM Conference on Embedded Networked Sensor Systems (SenSys 2007), Sydney, Australia, November 2007, pp. 175–188 (2007)
8. Yao, Y., Gehrke, J.: The Cougar approach to in-network query processing in sensor networks. ACM SIGMOD Record 31, 9–18 (2002)
9. Park, K., Elmasri, R.: Query classification and storage evaluation in wireless sensor networks. In: ICDE Workshops (2006)
10. Sadagopan, N., Krishnamachari, B., Helmy, A.: Active query forwarding in sensor networks. Ad-Hoc Networks 3, 91–113 (2005)
11. Karp, P.B.G.B., Ke, Y., Nath, S., Seshan, S.: Irisnet: An architecture for a worldwide sensor web. IEEE Pervasive Computing 2 (2003)
12. Liu, J., Zhao, F.: Towards semantic services for sensor-rich information systems. In: Proceedings the 2nd IEEE/CreateNet International Workshop on Broadband Advanced Sensor Networks (Basenets 2005), Boston, MA (October 2005)
13. Whitehouse, K., Liu, J., Zhao, F.: Semantic streams: a framework for composable inference over sensor data. In: Römer, K., Karl, H., Mattern, F. (eds.) EWSN 2006. LNCS, vol. 3868, pp. 5–20. Springer, Heidelberg (2006)
14. Coulson, G., Kuo, D., Brooke, J.: Sensor networks + grid computing = a new challenge for the grid? IEEE Distributed Systems Online 7 (2006)
15. Lim, H.B., Teo, Y.M., Mukherjee, P., Lam, V.T., Wong, W.F., See, S.: Sensor grid: Integration ofwireless sensor networks and the grid. In: LCN, pp. 91–99 (2005)
16. Compton, M.: A framework for finding equivalent rewritings. Technical report, CSIRO ICT Centre, Australia (2008) (submitted to ICDE 2009)
17. Hui, J.W., Culler, D.: The dynamic behavior of a data dissemination protocol for network programming at scale. In: SenSys 2004: Proceedings of the 2nd international conference on Embedded networked sensor systems, pp. 81–94. ACM Press, New York (2004)
18. Taylor, K., Ayyagari, A.: Research topics in semantic sensor networks: Preface to the proceedings of the semantic sensor networks workshop. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273. Springer, Heidelberg (2006)
19. Caniot, G., Lamb, P.: Key establishment in sensor networks. Technical report, CSIRO ICT Centre Conference, Sydney (2007)
20. Lamb, P.: Arc-based XML access control for general DTDs (in preparation, 2008)
21. He, D.D., Compton, M., Taylor, K., Yang, J.: Analysing access control in collaborative environments with description logic. Technical report, CSIRO ICT Centre (2008)
22. Lloyd, J.W.: Foundations of Logic Programming, 2nd edn. Springer, Heidelberg (1987)
23. Leuschel, M.: Logic program specialisation. In: Partial Evaluation, pp. 155–188 (1998)