SEPSen: Semantic Event Processing at the Sensor Nodes for Energy Efficient Wireless Sensor Networks (Short Paper)

Mumraiz Khan Kasi, Annika Hinze, Catherine Legg, Steve Jones
University of Waikato, Hamilton, New Zealand
mk218, hinze, clegg, stevej@waikato.ac.nz

ABSTRACT

Traditionally in WSNs, the sensor nodes are used only for capturing data that is then later analyzed in the more powerful gateway nodes. This requires a continuous communication that wastes energy at the sensor nodes and greatly reduces the overall network lifetime. We propose a semantic-based in-network data processing that reduces energy consumption and improves the scalability of heterogeneous sensor networks. Ontology fragments in each sensor node help identify the data routed through the sensor network. We have adapted a matching algorithm to process a changing knowledge base. Simulation results show that the networks' energy consumption is considerably reduced.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Sensor Networks

Keywords

Semantics, In-network Data Processing

1. INTRODUCTION

Wireless Sensor Networks (WSNs) consist of several interconnected sensor nodes and a gateway node. The sensor nodes continually sense data from the environment and forward it to the gateway node. The gateway node receives the data, processes it and sends (aggregated) information to the end user [4]. For environmental monitoring in hard-to-reach regions (e.g., mountaintops or bush), light-weight nodes are used for applications with high data rates and heterogeneous networks, while also requiring network longevity [6]. Additionally, real-time filtering enables timely identification of and reaction to critical changes in the environment. In most applications of WSNs, the users are interested in higher-level events that require the processing of data from multiple heterogeneous sensor nodes [8, 19]. In these cases, sensor nodes are used for capturing data only. This approach wastes considerable energy at the sensor-node level since all individual lower-level events have to be transmitted, regardless of their relevance. All processing of those events takes place at the higher-level gateway node; sensor nodes have no knowledge of the possible significance of an observed event.

We here present a sensor node architecture where data processing is performed at the sensor node level to obtain maximum energy benefits by processing data locally. We use ontology fragments at the nodes to enable data exchange between heterogeneous sensor nodes within the network. We employ a rule engine based on an adapted RETE algorithm [3] for decision making at the node level. This architecture has the advantage of performing in-network data processing of heterogeneous information since the significance of events can be detected at the sensor node level. This results in more efficient remote monitoring and prolonged network lifetime. The work presented here extends our previous work on a cost model for semantic in-network processing [10] and provides the first realistic test of what has been developed previously on a theoretical basis.

The paper is organized as follows: In Section 2, we present the challenges faced by WSNs. Existing approaches and their limitations are reviewed in Section 3. We introduce our node architecture in Section 4, and report the results of our evaluation in Section 5. Finally, Section 6 summarises our results and presents future research directions.

2. CHALLENGES

Wireless Sensor Networks are inherently constrained as each of the sensor nodes has only limited resources [5]. Here we outline challenges faced by heterogenous sensor networks and present their respective solutions in Section 3.

Energy Constraints. Energy consumption is one of the major problems in remote or hard-to-reach WSNs, where sensor nodes carry limited and generally irreplaceable power sources [13]. In most of these settings, large numbers of small sensor nodes are placed in close proximity to each other and they continually transmit sensed data to a set of gateways without pre-processing at the sensor node level. As radio communication between sensor nodes is energy-expensive [4, 5], this communication potentially wastes energy by transferring uninteresting or repetitive data. In many applications, end users are not interested in the raw data samples but rather in specific, less likely events, such as the passing of a pollution threshold. Moreover, a sensor node running out of energy may threaten the overall sensor network lifetime, as an insufficient number of live nodes may lead to failure of the mission [18].

Heterogeneous Data Sources. Heterogeneity can occur at different levels [15]: system, structure, syntax and semantics. System heterogeneity is caused by differences in hardware and software components, i.e., by a set of sensor nodes with different functionalities and capabilities. Structural heterogeneity chiefly refers to WSNs with different storage structures and data models. Syntactic heterogeneity is linked to differences in data representation and formats; and semantic heterogeneity refers to the fact that the same concept may have different meanings in different WSNs [8]. These

types of heterogeneities are often interrelated. Here we focus on heterogeneity in semantics.

Unpredictable Networks. Sensor Networks may be very unpredictable in their operations. New nodes can join the network and others might be damaged or run out of power. For example, new sensors could be added to the network which could measure different phenomena from the existing sensors in the network. Other sensors might be displaced due to flooding of the water reservoirs. Thus, the network needs to adapt to changing conditions and requirements in order to remain operable [2].

3. RELATED WORK

Very few approaches address performance in *heterogeneous networks at the sensor node level*. In this section, we discuss previous work on in-network data processing, semantic integration and context awareness.

In-Network Data Processing.

Recent WSN research focuses predominantly on the use of in-network data processing to reduce energy consumption by minimizing data volume locally. In-network data processing assumes that only a minimal amount of data is transferred within the network [1, 13]. Data items are filtered and aggregated before they are forwarded to the next node [17]. In this way in-network filtering reduces the transmission of insignificant events by restricting the source communication to relevant events. Directed diffusion [9] is a data filtering and aggregation model that uses an attribute-value-pairs naming scheme for data interests. When a gateway wants to gather data from the sensor nodes in the network, it broadcasts an interest message to all other nodes, describing the type of data it is interested in. When a node senses data matching the interest, it returns the result to the gateway. Aggregation is performed by nodes that are part of different data paths. The aggregation is based on a report gradient, which defines how many reports to send per unit of time. TAG [13] provides a SQL-like language for expressing aggregation queries over streaming sensors. It uses the WHERE and HAVING clauses, respectively, to filter out individual sensor readings and to suppress groups that do not satisfy the predicates. The SELECT clause is used to aggregate one or more attributes. Both TAG and direct diffusion [9, 13] eliminate redundancies and minimize the data transfer through in-network filtering and aggregation.

Semantic Integration.

The problem of integrating data from semantically heterogeneous sources is often addressed by using ontologies [8, 14]. Ontologies provide formal specifications of terms used in a domain by defining relations between terms [15]. They are used to describe the meaning and context of the data elements in the sensor network and enable a semantics-based classification of data throughout the network. Defining semantics for data sources improves the collaboration amongst heterogeneous data sources and facilitates the integration and exchange of sensor data among nodes [14]. It thus allows the simultaneous deployment of differing types of sensor nodes. Wun et al. [19] used content-based filtering over the S-TOPSS middleware to allow semantic event detection to occur both within and across sensor networks. Broker nodes run a semantic engine that uses mapping functions to translate raw sensor data into semantically equivalent events. Similarly, SWASN [8] uses a layered approach to data integration. Sensor data is gathered from heterogeneous sensor networks at the Sensor Network Layer. The data is then processed at the Ontology Layer by attaching semantic information. Further reasoning is undertaken using a rule-based engine at the Semantic-Web Processing Layer. The processed data is then made available to different client applications that require it through an Application Layer.

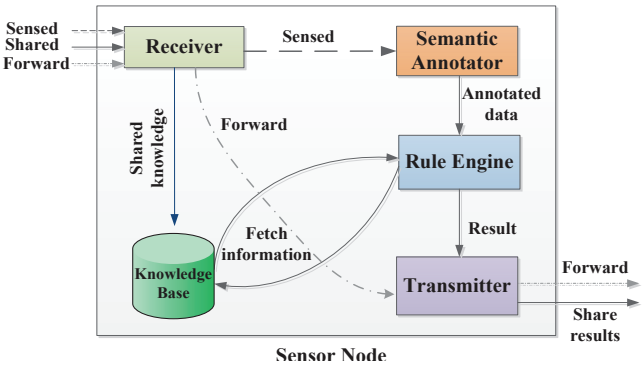

Figure 1: SEPSen architecture

Context-Awareness.

Most context-aware systems are human-centric and context is considered at application-level. However, we consider the context of each sensor node and its environment (i.e., node-centric [7]). Node-centric context awareness requires nodes to adapt to their available context, e.g., by altering the event-reporting frequency of a sensor node, without requiring instructions from gateway or users. The In-network Semantic Sensor Data Model *(ISSDM)* [2] targets the energy consumption by processing the data at the enhanced context-aware sensor nodes. The context input is derived from explicit and implicit context inputs to provide sufficient context information about the sensor network. Explicit context is formed from the actual readings of the sensor, whereas implicit context is formed from secondary information available to the sensor . Overall, the model proves to be useful in performing tasks at the node level and energy consumption can be largely reduced. Similarly, Context-Aware SensorNet(CASN) [7] uses context-awareness in the sensor nodes to adjust its behaviour accordingly.

4. OVERVIEW OF SEPSEN

In this section, we present an overview of SEPSen. The general architecture of a single SEPSen node is shown in Figure 1; its individual components of Receiver, Semantic Annotator, Knowledge Base, Rule Engine and Transmitter are explained subsequently.

The **Receiver** component distinguishes between three types of incoming events: sensed, shared or a forwarded event. The distinction is necessary as different operations are performed on each of the event types. The monitored phenomenon is measured as event data at regular intervals. This *sensed* event is then semantically annotated and compared against the stored rules. Some rules may refer to events that need to be aggregated or integrated from multiple heterogeneous sensor nodes; this event data is *shared* amongst the relevant sensor nodes. Moreover, on arrival, the shared event is added to the node's knowledge base. This process triggers the rule engine for evaluating rules against the shared knowledge base. The *forwarded* event is received from other sensor nodes and is to be passed on in a multi-hop manner to the gateways (routing); no filtering is performed for the forwarded events. The sensor node identifies an incoming event as a *shared* or *forwarded* event by reading the destination field of the incoming event data.

The **Semantic Annotation** of sensed events requires the relevant ontology fragments to be infused into each node in the sensor network. Events are then mapped to the concepts of the infused ontology. For example, when a sensor node monitoring the Wa-

Ontology fragment available at WaterpH sensor node
@prefix ont:<http://www.co-ode.org/ontologies/ont.owl#> @prefix owl:<http://www.w3.org/2002/07/owl#> @prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#> ont:MeasurementSite rdf:type owl:Class; ont:ObservedProperty rdf:type owl:Class; ont:Measurement rdf:type owl:Class; ont:WaterMeasurement rdf:type owl:Class; rdfs:subClassOf ont:Measurement. ont:WaterProperty rdf:type owl:Class; rdfs:subClassOf ont:ObservedProperty. ont:WaterPH rdf:type owl:Class; rdfs:subClassOf ont:WaterProperty. ont:hasProperty rdf:type owl:ObjectProperty; rdfs:domain ont:MeasurementSite; rdfs:range ont:WaterProperty. ont:hasMeasurement rdf:type owl:ObjectProperty; rdfs:domain ont:MeasurementSite; rdfs:range ont:WaterMeasurement. ont:hasValue rdf:type owl:DatatypeProperty; rdfs:domain ont:WaterProperty; rdfs:range xsd:double.

Table 1: Ontology Fragment

terpH in a reservoir senses a value of 5.0, it will annotate the event message with the relevant domain ontology information, such as:

```
ont:WaterpH_1 rdf:type ont:WaterpH,
              ont:hasValue "5.0"^^xsd:double.
```

The annotation is based on the ontology fragment available to this sensor node (see Table 1). The ontology fragments are obtained from the main ontology that models the overall application. For the purpose of our experiments, this process was executed manually.

The **Knowledge Base** of a sensor node contains a *facts base* and a *rules base*. Facts are event data recorded by this sensor and data received from other sensors. Individual facts are stored in triple form as <Subject, Predicate, Object>. This is also the format in which event data is provided by the sensors after the semantic annotation of the sensed values. The facts base is changed during runtime based on (1) new data being recorded by sensors and (2) data obtained as a result of an action triggered by a rule execution. Below is an example of facts stored in the knowledge base for the WaterpH sensor node at a particular time.

```
ont:MeasurementSite_1
    ont:hasMeasurement Ont:WaterMeasurement_1.
ont:WaterMeasurement_1
    ont:hasProperty ont:WaterpH_1.
ont:WaterpH_1
    ont:hasValue "5.0"^^xsd:double.
```

Part of the Knowledge base is also a collection of rules (i.e. the rule base) defined by the application. Rules are updated as the application changes its requirements or new requirements arrive. A rule contains both a set of conditions and the actions to be performed when those conditions are met. A rule fires if all its conditions are true. We used semantic rules in the form of IF-THEN expressions, which are similar to the established Event-Condition-Action rule structure. The IF-statement specifies the event conditions that must be met in order for the rule to apply and make the THEN statement valid. Below is an example rule to detect a pollutant in water reservoirs if the value of Waterp PH is greater than 4.0.

```
[Rule_1:
MeasurementSite(?a) & hasMeasurement(?a, ?b)
& WaterMeasurement(?b) & hasProperty(?b, ?c)
& WaterPH(?c) & hasValue(?c, ?d)
& greaterThan(?d, 4.0) -> PollutedWaterSource(?a)
]
```

The reasoning on the sensed data for data-integration purposes is done using the RETE algorithm [3]. The **Rule Engine** receives the application's requirements in the form of rules. These also specify the actions to be taken when the requirements contained in the rule are met. The implemented RETE algorithm gathers the rules and builds a pattern network of nodes that encodes the condition parts (IF-parts) of the rules. At the bottom of the pattern network are the nodes representing the individual rules. The input consists of changes in the facts as new facts are inserted or deleted from the knowledge base. When the input meets all the conditions of the rules, the corresponding output is activated. The output defines the applicable rules. These rules then trigger the execution of the THEN-parts and the appropriate actions are performed. The actions specified by the rule engine could be: *discard*, *share* or *forward* event. When the incoming fact does not match the rules then the event is discarded producing a filtering of events. When the fact completely matches any of the rules then the event is forwarded to the gateway node. However, if a partial match is found then the event is shared with the relevant sensor node for further processing.

The **Transmitter** component is responsible for transmitting the events to the gateway or other relevant sensor nodes. The sensors transmit only those events that are of interest to the application/user. This component is designed to use the context information (e.g., node energy levels) of the sensor network for routing decisions. In the case of partial matches, it shares its knowledge with the sensor nodes that can then perform further processing to complete the event. This routing decision is based on the types of sensors and the available data in the locality. Details of this context-based routing are part of future work.

5. EVALUATION

Simulations of the developed software and application were done using PowerTOSSIM [16]. PowerTOSSIM is based on the TinyOS [12] operating system and the TOSSIM [11] simulation environment. The specified energy model for the simulations was based on Mica2 sensor nodes.

Test-1: Total Energy Consumption.

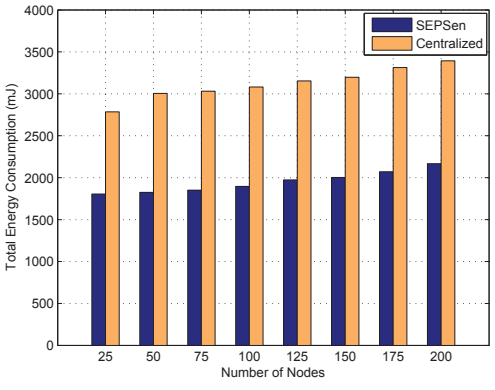

Figure 2: Total energy consumption for different approaches. *Simulations were run for 60 seconds with one message per second for an increasing number of sensor nodes*.

In the test whose results are shown in Figure 2, we compared the performance of our architecture to the centralized one. The experiment was run for varying numbers of nodes to observe the impact of network size on the performance of different architectures. It can be seen from the figure that our architecture performed better than the centralized architecture. The centralized architecture resulted in higher energy consumption because sensor nodes sent events to the gateway node at regular intervals. In our architecture, we used in-network data processing to reduce transmission of unnecessary events at the sensor nodes. This resulted in better performance with energy conservation from almost 13% for 25 nodes to 16% for 200 nodes. We found that as the number of nodes was increased, the total energy consumption of the centralized architecture almost doubled compared to our architecture. This is due to the fact that unnecessary transmissions are also routed by the sensor nodes in the path to the gateway nodes.

Test-2: Radio Energy & Radio Message Counts.

Figure 3 shows the radio energy consumption for each simulated node and the total number of radio messages. The nodes in the centralized architecture sent nearly three times as many messages as in SEPSen, and consumed more radio energy. The gateway nodes consumed higher radio energy as they had to process incoming events from all the child sensor nodes. Furthermore, radio energy is consumed while transmitting and receiving the data particularly when gateway nodes receive a large number of events from the child sensor nodes. In our architecture the number of transmissions was greatly reduced resulting in reduced radio energy consumption not only at the sensor nodes but also at the gateway nodes.

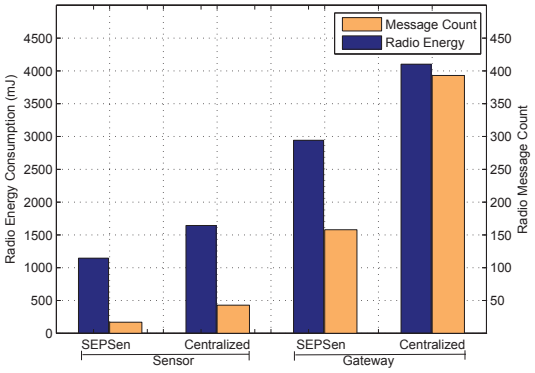

Figure 3: Simulated radio energy consumption vs. radio message counts for different approaches. *Simulations were run for 60 virtual seconds with one message per second over the 10-node topology*.

6. CONCLUSION AND FUTURE WORK

Previous work on efficient use of WSN either ignores the heterogeneity of sensor networks or performs the processing tasks at the powerful gateway nodes, reducing the sensor network lifetime. We developed a sensor node architecture that takes into account the processing capabilities of the sensor nodes and tries to utilize it for energy conservation in WSNs. The sensor nodes annotate the events from the domain ontology and execute application rules on the events. This enables in-network processing for heterogeneous sensor networks. By performing data processing locally, the sensor nodes made decisions quickly and remotely without the need for instructions from gateway nodes. We demonstrate this using results from tests in a water-quality monitoring application. The current SEPSen implementation focuses on efficient filtering. The routing will use energy-related context information. Moreover, currently sensor nodes are unaware of the knowledge their neighbouring sensor nodes have. We plan to develop context-awareness at the sensor node level to share and gather knowledge about the available data in the sensor network. Initially we need to provide the sensor nodes with domain context to enable them to ask for available data from other sensor nodes. Then, once the data is acquired, the next task will be the integration of the sensory events from multiple sources within the sensor network.

7. REFERENCES

[1] Y. Chen, H. V. Leong, M. Xu, J. Cao, K. C. C. Chan, and A. T. S. Chan. In-network data processing for wireless sensor networks. In *Mobile Data Management (MDM'6)*, 2006.

[2] K. W. Chow and Q. Li. Issdm: an in-network semantic sensor data model. In *Proc. of the Symposium on Applied computing*, SAC '07, pages 959–960, 2007.

[3] C. L. Forgy. Expert systems. chapter Rete: a fast algorithm for the many pattern/many object pattern match problem. IEEE press, 1990.

[4] L. Gu, D. Jia, and . Vicaire, Pascal et a. Lightweight detection and classification for wireless sensor networks in realistic environments. In *SenSys'05*, 2005.

[5] T. He, S. Krishnamurthy, J. A. Stankovic, and A. et al. Energy-efficient surveillance system using wireless sensor networks. In *MobiSys '04*, 2004.

[6] A. Hinze, K. Sachs, and A. Buchmann. Event-based applications and enabling technologies. In *Distributed Event-Based Systems (DEBS '09)*, 2009.

[7] Q. Huaifeng and Z. Xingshe. Integrating context aware with sensornet. In *Proc. of International Conference on Semantics, Knowledge and Grid*, SKG '05, page 83, 2005.

[8] V. Huang and M. K. Javed. Semantic sensor information description and processing. In *Sensor Technologies and Applications*, 2008.

[9] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *MobiCom '00*, 2000.

[10] M. K. Kasi and A. Hinze. Cost analysis for complex in-network event processing in heterogeneous wireless sensor networks. In *DEBS '11*, 2011.

[11] P. Levis, N. Lee, M. Welsh, and D. Culler. Tossim: accurate and scalable simulation of entire tinyos applications. In *SenSys '03*, 2003.

[12] P. Levis, S. Madden, J. Polastre, R. Szewczyk, and A. W. et al. TinyOS: an operating system for sensor networks. In *Ambient Intelligence*. Springer Verlag, 2004.

[13] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tag: a tiny aggregation service for ad-hoc sensor networks. *SIGOPS Oper. Syst. Rev.*, 36:131–146, 2002.

[14] L. M. Ni, Y. Zhu, J. Ma, M. Li, Q. Luo, Y. Liu, S. C. Cheung, and Q. Yang. Semantic sensor net: An extensible framework. In *ICCNMC*, pages 1144–1153, 2005.

[15] A. P. Sheth. Changing focus on interoperability in information systems: From system, syntax, structure to semantics. In M. G. et al., editor, *Interoperating Geographic Information Systems*. Kluwer Academic Publishers, 1999.

[16] V. Shnayder, M. Hempstead, B.-r. Chen, G. W. Allen, and M. Welsh. Simulating the power consumption of large-scale sensor network applications. In *SenSys '04*. ACM, 2004.

[17] A. Silberstein, R. Braynard, G. Filpus, G. Puggioni, A. Gelfand, K. Munagala, J.Yang, B.Kamachari, D.Estrin, and S. Wicker. Data driven processing in sensor networks. In *Innovative Data Systems Research (CIDR '07)*, 2007.

[18] X. Wang and J. Li. Precision constraint data aggregation for dynamic cluster-based wireless sensor networks. In *Mobile Ad-hoc and Sensor Networks (MSN '09)*, 2009.

[19] A. Wun, M. Petrovi, and H.-A. Jacobsen. A system for semantic data fusion in sensor networks. In *Distributed Event-Based Systems(DEBS '07)*, 2007.