2009 IEEE International Symposium on Intelligent Control
Part of 2009 IEEE Multi-conference on Systems and Control
Saint Petersburg, Russia, July 8-10, 2009

Rule-Based In-Network Processing in Wireless Sensor Networks

Ozgur Sanli
Department of Computer Engineering
Middle East Technical University
06531, Ankara, Turkey
Email: ozgur.sanli@ceng.metu.edu.tr

Ibrahim Korpeoglu
Department of Computer Engineering
Bilkent University
06800, Ankara, Turkey
Email: korpe@cs.bilkent.edu.tr

Adnan Yazici
Department of Computer Engineering
Middle East Technical University
06531, Ankara, Turkey
Email: yazici@ceng.metu.edu.tr

***Abstract*— Wireless sensor networks are application-specific networks, and usually a new network design is required for a new application. In event-driven wireless sensor network applications, the sink node of the network is generally concerned with the higher level information describing the events happening in the network, not the raw sensor data of individual sensor nodes. As the communication is a costly operation in wireless sensor networks, it is important to process the raw data triggering the events inside the network instead of bringing the raw data to the sink and processing it there. This helps reducing the total amount of packets transmitted and total energy consumed in the network. In this paper, we propose a new method that distributes the information processing into the sensor network for event-driven applications. We also describe an application scenario, healthcare monitoring application, that can benefit from our approach.**

I. INTRODUCTION

With recent advancements in micro electro-mechanical systems (MEMS), it is possible to employ wireless sensor networks (WSN) that can sense and act on the physical phenomenon that happens around them. A typical wireless sensor network is composed of small sensing devices with short-range radio communication capabilities and limited resources. Sensor nodes generally operate on the power supplied by the irreplaceable batteries, thus, energy efficiency is very important in wireless sensor networks. The energy consumption of sensing and processing operations are far less compared to the energy consumption of communication operations. As described in [7], transmitting 1 Kb of data at a distance of 100 meters costs 3 joules in a noise-free environment. Nevertheless, the same amount of energy is consumed by a general purpose processor with 100 MIPS/W capability when it executes 3 million instructions. This suggests that reducing the network traffic volume must be a major concern for energy efficient sensor networks.

In-network processing is a distributed information processing technique that is used to reduce the network traffic in sensor networks. If in-network processing is used, all or part of the data is processed by the nodes inside the sensor network rather than processing all data at a central node. As a result, refined higher level information is transported inside the network instead of raw sensor data.

When in-network processing is employed, sensor readings can be processed near to the sensed area and the redundant or unnecessary data can be filtered out. In this way, the amount of traffic transported in the network decreases and this improves the energy utilization, prolongs the lifetime of both individual sensor nodes and the entire sensor network. Furthermore, in-network processing of the data is also a more fault-tolerant and scalable approach. If a centralized approach is used for information processing, sink node takes all the responsibility of processing and the raw data first need to be brought to the sink in order to be processed. In such an approach, sink node is a single point of failure. Additionally, adding more sensor nodes will have a negative influence on the sensor network as the total number of packets transmitted in the network and passing through the nodes close to the sink will increase: causing more energy consumption and decreasing the sensor network lifetime.

An important challenge associated with the sensor networks is that different kinds of applications have different sets of requirements, which makes it impossible to design generalized algorithms that can be used in all application scenarios. For example, some applications are deployed to gain insights on an unknown physical phenomenon. Such research oriented applications require all sensor readings to be recorded for offline analysis. On the other hand, expectation from a forest fire detection or health-care monitoring application is the notification of emergency cases. As these examples show, appropriate algorithms for an application should be developed with its own specific requirements.

Sensor network applications can be classified as demand-driven and event-driven applications [9]. In event-driven application model, the processing and reactions take place only after the occurrence of something important for the application, and there is no need for continuous flow of sensor readings or periodic query requests. In order to take advantage of this approach, we need to establish mechanisms that can produce high level information from raw data, decide if the information is of interest to the application and propagate the relevant information to the appropriate nodes in the sensor network.

In this paper we propose an in-network processing method for wireless sensor networks. In our approach, we adopt the rule-based inferencing as the information fusion method. Event-condition-action (ECA) rules in a rule-base express the application logic. For each type of node that has processing capabilities, a new sub-rule-base is created from the original rule-base; i.e, the original central rule-base is decomposed

into multiple sub-rule-bases which are used to construct information processing engines distributed into the sensor network.

Our approach is particularly useful for large scale sensor networks that run event-driven real-time applications. In such applications, transmitting all the raw sensor measurements without considering whether they are related to an event of interest or not causes too much energy consumption. Additionally, for real-time applications that require immediate response, the delay between the occurrence of an event and the respective decision and the accompanying action should be small and bounded. In large networks, this delay may be quite large if processing is only done at a central location.

The rest of the paper is organized as follows: In Section 2, we present the related work. In Section 3 we show the details of our proposed solution and give some examples for clarification. In Section 4, we discuss about an application scenario which benefits from our approach. Finally in Section 5, we conclude the paper.

II. RELATED WORK

Cougar [8], TinyDB [4] and Directed Diffusion [3] are the popular data processing schemes that employ in-network processing in sensor networks. Cougar and TinyDB views the sensor network as a large distributed database and instead of collecting data at the sink, queries are distributed into the network. Query processing systems are demand-driven in nature, so they are not suitable for event-driven applications. In Directed Diffusion, interests, which are attribute-value pairs, are placed in the network by the sink and the nodes send their data to it if the interest is satisfied. Data is aggregated along the way back to the sink. Similar to previous approaches, it is demand-driven and not suitable for applications that require continuous monitoring. Furthermore, attribute-value pairs are not always expressive enough to describe what is interesting. Therefore, unnecessary packets might still be transported.

In [6], a composite event processing framework that works on top of a range of publish/subscribe systems is proposed. Distributed composite event detectors are installed at various locations in the sensor network according to the requirements of the application such as latency, reliability or bandwidth usage. Although this work is similar to our study in the way that composite event expressions are decomposed into sub-expressions as we do in rule-base decomposition process, their proposition is not specifically about the wireless sensor networks and therefore, they do not take the constraints present in the wireless sensor networks into account. Furthermore they do not discuss about how the decomposition is done. On the other hand, we give an algorithmic way of decomposition that suits to the requirements of wireless sensor networks. Additionally, we consider a hierarchical network architecture, and we describe precisely how we can classify the nodes based on what they can process, and therefore where the information processing engines can be placed.

In [5], a rule-based distributed fuzzy logic reasoning engine is described. The reasoning engine employs simple if-then rules for the decision process and it uses fuzzy logic to fuse the individual sensor readings and the neighbor observations to get more accurate results. Although if-then rules which are similar to our ECA rules are used for information processing, the main motivation of this study is to improve reliability of the decisions and it mentions only about processing done at a single node. However, our method is about distributed processing in the entire wireless sensor network, and in our approach sensor nodes cooperate for the purpose of reducing network traffic and energy consumption.

In [1], a proactive and distributed mechanism is proposed to detect the sets of interrelated events, also called contexts. Event notifications are delivered to special nodes which are connected through an overlay network. Similar to our method, these nodes make partial context decisions and forward their decision to the next node in the overlay. However, the approach adopted in this study is to express the logical relations as disjunctions of conjunctions of premises whereas our approach is to express them as conjunctions of disjunctions of premises. Furthermore, in [1] nodes need to keep the address of the sensor nodes that are responsible for processing the next input element. As we use a hierarchical sensor network architecture, sensor nodes in our approach only need to know how they can reach the nearest node in the next hierarchical level. This simplifies the routing strategy.

III. DISTRIBUTED INFORMATION PROCESSING

In this section we give the details of how we distribute the information processing into the sensor network.

A. Event-Condition-Action Rules

In our approach, application logic is expressed by a set of Event-Condition-Action (ECA) rules. An ECA rule is a formal method to express active capabilities. The rules are in the following format:

```
ON   Event
IF   <Conditions>
THEN <Actions> / <Conclusions>
```

ECA rules represent a set of statements the execution of which depends on the occurrence of a triggering event and satisfaction of a set of conditions. These statements express the actions to be taken and the conclusions to be drawn.

Employing ECA rules instead of embedding the logic into the application code has some advantages [9]. First the rules can be stored outside in a rule-base which improves the modularity, maintainability and extensibility of the applications. Second, ECA rules have a high-level declarative syntax, so they can easily be analyzed and optimized. Finally, ECA rules provide a generic mechanism to express the reactive behavior contrary to the application code that is typically specialized to a particular type of reactive scenario.

B. Hierarchical Information Processing

An ECA rule that is in the form of "`ON <event>; IF <conditions>; THEN <actions>`" can be rewritten as a deductive rule in the form of "`IF (<event> & <conditions>); THEN <actions>`". The triggering

event of a rule and the conditions are the inputs of that rule, whereas the actions or conclusions are the outputs.

A rule can only satisfy once all the inputs are available and they evaluate to *true*. Nevertheless, these inputs become available at different times and in different places in the sensor network (e.g. ordinary sensor nodes, cluster-heads or the sink). If a node cannot process a rule because some part of the input is located at a different node, then these inputs should be collected at some place where it is possible to check the satisfaction of the premises and execute the accompanying statements in the case of a match. The simplest approach would be to collect all the inputs at a central location, and process them once all of them become available. However, it has already been stated that it is desirable to do in-network processing. Therefore, information processing should be distributed into different nodes in the network in an efficient and reliable manner.

All the rules that are required for the application constitute a rule-base. A reasonable way to express the application logic is to devise a central rule-base which does not take where and how the data is processed into consideration. Because, designing multiple rule-bases while taking such issues into account requires much more attention in order not to make mistakes which lead to serious problems common in distributed applications, like consistency problems. Such a design is especially a problem for applications that use large sets of rules.

After the generation of a central rule-base properly expressing the application logic, the central rule-base is decomposed into smaller rule-bases for distributed information processing in the sensor network. The rules in a sub-rule-base is such that all the premises can be evaluated at the place where the information processing engine based on this sub-rule-base will run.

A sub-rule-base is created only for the nodes having processing capabilities. Inputs that are available for these nodes are used to classify the types of nodes in the sensor network. Different types of nodes have different input sets. Two nodes which can process exactly the same inputs are referred to as the same type of nodes, and employ the same rule-base. Although nodes in a homogeneous sensor network are all same kind, it is still possible that they have different input sets because their responsibility in the sensor network might differ. For example, majority of the nodes only process their own data while some of them function as cluster-heads and process their neighbors' data as well.

In a hierarchical sensor network, the input sets of nodes differ at each hierarchical level and as a result, a different rule-base is created for each of these hierarchical levels. For example, if all nodes have the information processing capabilities in a cluster-based sensor network, then there might be three sub-rule-bases employed at ordinary sensor nodes, cluster-heads and the sink, respectively. If ordinary sensor nodes can not process data, only two rule-bases are used: for cluster-heads and the sink.

Algorithm 1 Rule-Base Decomposition

1: RB = original rule-base
2: SRB = new sub-rule-base
3: $k \leftarrow 0$
4: **for all** R in RB **do**
5: $a \leftarrow antecedent(R)$
6: $cnf \leftarrow conjunctive\_normal\_form(a)$
7: $/*\ \ cnf = (\alpha \wedge (\beta_1 \vee \beta'_1) \wedge \cdots \wedge (\beta_n \vee \beta'_n))\ \ */$
8: $\beta \leftarrow beta(cnf)$ // $\{\beta_1, \beta_2, \ldots, \beta_n\}$
9: $\beta' \leftarrow beta'(cnf)$ // $\{\beta'_1, \beta'_2, \ldots, \beta'_n\}$
10: $P \leftarrow powerset(\beta)$
11: **for all** M in P **do**
12: **if** M == β **then**
13: Add "$if\ (\alpha \wedge \beta_1 \wedge \ldots \wedge \beta_n);\ then\ A$" into SRB
14: Remove R from RB
15: **else**
16: $m \leftarrow number\_of\_elements(M)$
17: $a_1 \leftarrow \alpha \wedge (\bigwedge_{i=1}^{m} element(M, i))$
18: Add "$if\ (a_1);\ then\ A_k$" into SRB
19: $M' \leftarrow \{\beta'_i \mid \beta_i \notin M\}$
20: $n \leftarrow number\_of\_elements(M')$
21: $a_2 \leftarrow A_k \wedge \left(\bigwedge_{j=1}^{n} element(M', j)\right)$
22: Add "$if\ (a_2);\ then\ A$" into RB
23: $k \leftarrow k + 1$
24: **end if**
25: **end for**
26: **end for**

C. Rule-Base Decomposition

The rule-base decomposition process first creates the sub-rule-base for the type of nodes that are classified as being in the lowest hierarchical level. Such nodes' input set does not have any subset which is the input set of some other node. If such a subset existed, then the hierarchical level of these nodes would not be the lowest level, because higher layers have more inputs, which possibly include the inputs from lower layers.

Next rule-base is going to be created for the nodes having the following property: subsets of those nodes' input set might only be the input set of a node at a lower hierarchical level, not at the higher levels. This process is repeated until a sub-rule-base is created for each different type of nodes. The data is processed in the lowest hierarchical level where it can be processed and it is not relayed into upper layers. This bottom-up approach in creating the sub-rule-bases supports the "data should be processed as close to its source as possible" idea.

The inputs of a rule are in the form of conjunctions or disjunctions of the events and conditions. Our algorithm for rule-base decomposition process, Algorithm 1, requires that the antecedent part of the rules are put in the conjunctive normal form which is a conjunction of disjunctive clauses:

$$(c_{11} \vee c_{12} \vee \cdots \vee c_{1k}) \wedge \ldots \wedge (c_{i1} \vee c_{i2} \vee \cdots \vee c_{in})$$

Each c_{ij} is called an input expression and it is a literal

or a comparison that evaluates to *t*rue or *f*alse. For any node in the sensor network, the input expressions of a rule can be classified into two: the input expressions that might be processed by the node, and the expressions that cannot be evaluated by that node. Using this information, we can construct a generalized form of the antecedent of a rule in the following format:

$$\alpha \wedge (\beta_1 \vee \beta_1') \wedge (\beta_2 \vee \beta_2') \wedge \cdots \wedge (\beta_n \vee \beta_n')$$

where α is the conjunction of disjunctive clauses that only contain the boolean expressions that use the input variables from the node's input set; i.e. all the necessary input for the evaluation of α is available at the node. Similarly, $\beta_1, \beta_2, \cdots, \beta_n$ are the disjunctions of the boolean expressions that the node can evaluate, whereas, $\beta_1', \beta_2', \cdots, \beta_n'$ are the disjunctions of the expressions that this node cannot decide on.

Let β be the set $\beta = \{\beta_1, \beta_2, \ldots, \beta_n\}$, β' be the set $\beta' = \{\beta_1', \beta_2', \ldots, \beta_n'\}$ and P be the powerset of β. For a member M of the powerset P, if M and β are not identical, then a rule having the following antecedent will be added into the sub-rule-base:

$$\alpha \wedge \bigwedge_{i=1}^{m} M(i)$$

where m is the cardinality of M and $M(i)$ is the i^{th} element of M. The output part of the rule will be an auxiliary output O_t representing the current matching conditions of the original rule. If β part of a disjunctive clause evaluates to false, it is still possible that overall disjunctive clause holds true as β' part of the clause might result in a true evaluation. For this reason, when an auxiliary output is generated, i.e. it is not possible to decide whether the conditions for the original rule hold true or not, this auxiliary output should be forwarded to the upper layers in the hierarchy so that the inputs available there can be used to further validate the conditions.

If we define the set M' as $M' = \{\beta_i' \mid \beta_i \notin M\}$ with cardinality n, then a new rule with the original output O and the following antecedent is added into the central rule-base:

$$O_t \wedge \bigwedge_{i=1}^{n} M'(i)$$

where $M'(i)$ is the i^{th} element of M'.

If M and β are identical, the formula for adding a new rule into the newly generated rule-base does not differ from the previous case with the exception that the rule will have the original output O as its output part, not an auxiliary output. Another difference from the previous case is that the original rule will be removed from the central rule-base instead of the addition of a new rule.

The above process should be repeated for each member of the powerset P so that all possible combinations of condition matching are enumerated. As a result of this process, a central rule is decomposed into multiple sub-rules placed in two rule-bases: newly created sub-rule-base and the modified central rule-base.

The above steps are for the decomposition of a single rule. In order to generate the complete sub-rule-base, the operations taken for just one rule should be repeated for every rule in the original rule-base. The complete distribution of information processing into the sensor network requires the creation of sub-rule-bases for every different type of node residing in different hierarchical levels.

1) Rule Decomposition Example: The following example is given to illustrate the rule-base decomposition process. Let's consider the following rule:

$$if\ (\alpha \wedge (\beta_1 \vee \beta_1') \wedge (\beta_2 \vee \beta_2'));\ then\ O$$

where α represents the conjunction of disjunctive clauses that contain only the input expressions that can be processed by the node; similarly β_1 and β_2 represent the disjunctions of input expressions that can be processed by the node, and β_1' and β_2' represent the disjunctions of input expressions that cannot be evaluated by that node. For such a rule, the powerset P is equal to $\{\{\beta_1, \beta_2\}, \{\beta_1\}, \{\beta_2\}, \{\}\}$. In such a case, if we follow the steps described above, the following rules would be added into the new sub-rule-base:

$$\begin{aligned} if\ (\alpha \wedge \beta_1 \wedge \beta_2);\ then\ O \\ if\ (\alpha \wedge \beta_1);\ then\ O_1 \\ if\ (\alpha \wedge \beta_2);\ then\ O_2 \\ if\ (\alpha);\ then\ O_3 \end{aligned}$$

If all of α, β_1 and β_2 hold true, then the event detection engine reaches a conclusion and the associated actions are taken. On the other hand, if one or both of the β_1 or β_2 cannot be evaluated as true, then the available information is fused and the result is sent to the next node in the hierarchy. The original rule in the central rule-base is removed and the following rules are added into it:

$$\begin{aligned} if\ (O_1 \wedge \beta_2');\ then\ O \\ if\ (O_2 \wedge \beta_1');\ then\ O \\ if\ (O_3 \wedge \beta_1' \wedge \beta_2');\ then\ O \end{aligned}$$

IV. APPLICATION SCENARIO

Our approach is useful for event-driven sensor network applications where the meaning of events depend on the conditions in different contexts. Furthermore, the application should not require the raw sensor data to be recorded, like it is the case for research oriented applications where the aim is to extract knowledge about the inner workings of an unexplored real-world phenomena. On the other hand, there are many application scenarios where the only interest is in the high-level knowledge of whether some event happens or not. For example early detection of forest fires, healthcare monitoring, etc.

Healthcare monitoring is an appropriate application scenario for demonstrating the benefits of our proposed in-network processing scheme. In a healthcare monitoring application, physiological signals of a person also called vital

signs, such as pulse rate, blood pressure, blood oxygen saturation, respiration rate, body temperature, etc., together with the environmental conditions are assessed so that an emergency case or an abnormal situation can be detected and handled immediately. Vital signs may have a different meaning in different conditions. For example, pulse rate of a person while he is exercising might be twice the value of it while he is resting. Besides, normal values of vital signs differ according to the person's age or sex.

Healthcare applications require continuous evaluation of sensor values since an emergency case might happen at any time and any place. What's more, time is very critical in such applications and immediate reaction is required in the case of an emergency. Finally, it is very important to relay only the relevant information to the medical center as the person there responsible for monitoring and managing alerts might be overwhelmed by the number of those alerts and miss some important ones.

In the following subsections, we give the details of the system architecture of a healthcare monitoring application and the information processing in this sensor network.

A. System Architecture

The architecture of the healthcare monitoring application consists of a body area sensor network, a home network and a medical center network. Body area sensor network consists of wearable medical sensors that sense the physiological signals of a person and sensor nodes that detect the posture and movement or other relevant physical activities or characteristics of the person. Medical sensors detect pulse rate, pulse rhythm, blood pressure, respiration rate, body temperature, blood oxygen saturation, and similar vital signs. Additionally, physical motion sensors like the accelerometer and the gyroscope is used to detect the current physical condition of the person. An accelerometer is used to measure the forward or upward acceleration so that it is possible to determine if the person is running, walking, falling down or stationary. A gyroscope measures the orientation, so it might help to detect the orientation of the body of the person, such as sitting, lying or standing. Wearable sensor nodes have limited processing capabilities and they only check if the sensed value is above or below a threshold value.

Sensor nodes communicate with a PDA or a special device that is used to collect and process sensor nodes' readings, communicate with the medical center network, and react to emergency cases. Apart from the wearable sensors, temperature and light sensors placed in the home give the extra information which helps in the clarification of the state of the person being monitored. Sensor nodes communicate with the special device/PDA using 802.15.4 or a similar low power and low data-rate protocol.

Medical center's network contains a central server that stores and processes the information coming from individuals. Medical history of the people is also stored in this center for additional analysis. The special device/PDA in the home network communicates with this network using GSM or UMTS (3G) networks. In addition to the information processing systems, there are also experts who are responsible for monitoring and managing the emergency cases.

B. Healthcare Monitoring Rules

Rules that are used in monitoring the healthcare should be developed by domain experts. Although we are not domain experts, the following rules are given for illustration purpose. These rules are used to show how we distribute the processing so that not all data is collected at a central place.

```
ON (Blood_Oxygen_Saturation < 80)
IF ((Blood_Pressure < 100/70) & (Respiration_Rate > 20))
THEN Send_Alert("Shock")

ON (Pulse_Rate > 100)
IF ((Blood_Pressure < 100/70) &
    ((Blood_Oxygen_Saturation < 80) |
     (Respiration_Rate < 15)))
THEN Send_Alert("Hearth Attack")

ON (Speed >= 6 km/h)
THEN Running

ON (Pulse_Rate > 100)
IF (!Running & (Respiration_Rate > 20))
THEN Send_Alert("Abnormal situation")

ON (Blood_Pressure > 120/80)
THEN Increment(High_Blood_Pressure_Count)

ON (Blood_Pressure > 120/80)
IF ((High_Blood_Pressure_Count > 5) &
    (High_Blood_Pressure_in_Family = true))
THEN Warn("See Doctor")
```

The above rules contain parts that can be processed at different places in the network and if we follow our decomposition algorithm we come up with several rule-bases for ordinary sensor nodes, a rule-base for the PDA/special device and a final one for central server at the medical center. Sensor nodes make comparison of their measurements with the appropriate threshold values in order to detect events that might be interesting for the application. The following is a collection of rule-bases for different types of sensor nodes:

```
ON (Blood_Oxygen_Saturation < 80)
THEN low_oxygen_saturation

ON (Blood_Pressure < 100/70)
THEN low_blood_pressure
ON (Blood_Pressure > 120/80)
THEN high_blood_pressure

ON (Respiration_Rate > 20))
THEN high_respiration_rate
ON (Respiration_Rate < 15)
THEN low_respiration_rate

ON (Pulse_Rate > 100)
THEN fast_pulse_rate

ON (Speed >= 6 km/h)
THEN Running
```

Simple events from sensor nodes are gathered and fused in the PDA/special device and the rules used for this purpose are as follows:

```
ON low_oxygen_saturation
IF (low_blood_pressure & high_respiration_rate)
THEN Send_Alert("Shock")

ON (fast_pulse_rate)
IF (low_blood_pressure &
    (low_oxygen_saturation | (low_respiration_rate))
```

```
THEN Send_Alert("Hearth Attack")

ON (high_pulse_rate)
IF (!Running & high_respiration_rate)
THEN Send_Alert("Abnormal situation")

ON (high_blood_pressure)
THEN Increment(High_Blood_Pressure_Count)

ON (high_blood_pressure)
IF (High_Blood_Pressure_Count > 5)
THEN High_Blood_Pressure_Alarm
```

We assume that the information about whether there is high blood pressure problem in a family member is stored in a database in medical center network. The rule-base for the central server at the medical center is as follows:

```
ON (High_Blood_Pressure_Alarm)
IF (High_Blood_Pressure_in_Family = true)
THEN Warn("See Doctor")
```

The above rule-bases make sure that the processing is distributed in the network and the data is transported only if there is an interest in it. Sensor readings are transported if they satisfy a filtering rule. Similarly, PDA eliminates false positives to be sent to the central server. For example, heart rate goes up while exercising and respiration rate decreases while sleeping, and these should be considered normal. Actually, for a typical person, we expect a large value for the ratio of these false positives to the real problems.

V. CONCLUSIONS AND FUTURE WORKS

In this paper we proposed a new method to distribute information processing into different nodes in a wireless sensor network for event-driven applications. We presented how we decompose a central rule-base expressing the application logic into multiple sub-rule-bases which are employed in the appropriate places inside the network. Our motivation was to process the data inside the network as much as we can, since communication operations are responsible for most of the energy consumption in the sensor nodes. We also described an application scenario which might benefit from our approach.

In this study we only considered logical relations between events and conditions. But temporal relations between these events and conditions are also important and need to be considered. In our future work we will study how we can incorporate temporal aspects of information processing into our approach. As another future work, we will work on how we can handle the uncertainty and imprecision of sensor readings.

ACKNOWLEDGMENTS

This work is supported in part by a research grant from TUBITAK EEEAG 106E012.

REFERENCES

[1] S. Ahn, D. Kim, "Proactive Context-Aware Sensor Networks", *Lecture Notes in Computer Science*, Vol. 3868/2006, pp. 38-53, 2006.

[2] I. F. Akyildiz, S. Weilian, Y. Sankarasubramaniam, E. Cayirci, "A Survey on Sensor Networks", *IEEE Communications Magazine*, Vol. 40, No. 8, pp. 102-114, Aug 2002.

[3] C. Intanagonwiwat, R. Govindan, D. Estrin, "Directed Diffusion: a scalable and robust communication paradigm", *Proceedings of the 6th. Annual International Conference on Mobile Computing and Networking*, pp. 56-67, Boston, 2000.

[4] S. R. Madden, M. J. Franklin, J. M. Hellerstein, W. Hong, "TinyDB: an Acquisitional Query Processing System for Sensor Networks", *ACM Trans. Database Syst.*, Vol. 30, No. 1, pp. 122-173, 2005.

[5] M. M. Perianu, P. Havinga, "D-FLER: Distributed Fuzzy Logic Engine for Rule-Based Wireless Sensor Networks", *Ubiquitous Computing Systems*, Vol. 4836/2007, pp. 86-101, 2007.

[6] P. R. Pietzuch, B. Shand, J. Bacon, "Composite Event Detection as a Generic Middleware Extension", *IEEE Network Magazine, Special Issue on Middleware Technologies for Future Communication Networks*, Jan/Feb 2004.

[7] G. J. Pottie, W. J. Kaiser, "Wireless Integrated Network Sensors", *Communications of the ACM*, Vol. 43, No. 5, pp. 51-58, 2000.

[8] Y. Yao, J. Gehrke, "The Cougar Approach to In-Network Query Processing in Sensor Networks", *SIGMOD*, 2002.

[9] M. Zoumboulakis, G. Roussos, A. Poulovassilis, "Active Rules for Sensor Databases", *Proceedings of the First Workshop on Data Management for Sensor Networks (DMSN'04)*, August 2004.