Networking Solutions for Sensor Networks

Pedro Brandão[1], João Barros[1]

CS Department, University of Porto & LIACC
Rua do Campo Alegre, 823
4150-180 Porto - Portugal
{pbrandao,barros}@ncc.up.pt

Abstract. Sensor networks are currently the focus of active research in a considerable number of fields. Because of their many applications, ranging from forest surveillance to anti-terrorist protection, from medical monitoring to crops inspection, from traffic sensing to environment control, sensors are gaining a big momentum in our every day life. The deployment of these technologies encompasses some problems related to communications, application development, lifetime of the network, and security. In this article, we present a set of current proposals that address some of the most relevant technical issues.

1 Introduction

Sensor networks have caught the attention of several scientific communities and are in the course of becoming ubiquitous components in our every day life. Sensors are able to detect presence, monitor the environment (temperature, humidity, wind, etc), track the well being of a person (measuring blood pressure, blood components levels, etc), among many other tasks [1] – the scope of applications is limited only by our imagination.

These wireless sensor networks made of tiny, low-cost devices capable of sensing the physical world and communicating over radio links, are significantly different from classical wireless networks like GSM or wireless LANs: (a) the design of a sensor network is strongly driven by its particular application, (b) sensor nodes are highly constrained in terms of power consumption and computational complexity, and (c) since the network is dense and the nodes share a common objective – to gather and convey information – cooperation can be used to enhance the network´s efficiency.

In general terms, a sensor network can be viewed as a collection of transmitters that observe multiple correlated sources of information, encode the picked up data and cooperate to send this information possibly with multiple hops over the wireless medium to a remote fusion center for further processing.

Several research challenges arise from these emerging sensor networks [2]: how to route data in a-priori unknown network or an ever changing one; how to exploit the correlation structure of the data and the sensors' ability to cooperate in order to increase data throughput, save energy, and improve data analysis; how to combine application-driven requirements with sensor resources so as to optimize

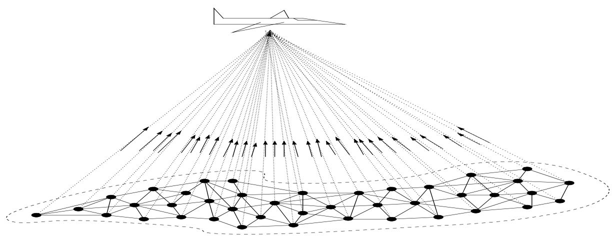

Fig. 1. A sensor network picks up measurements on a physical process evolving in space and transmits this data to a mobile data gathering unit.

the final result; how to secure the identification and the data transmission on a sensor network.

With this article we intend to provide an overview of some of the problems and current proposals for solutions. We will address routing and the definition of middleware for sensor networks. In section 2 we will introduce sensor networks and some of its specificities. Section 3 addresses some of the proposals for routing in sensor networks, whereas section 4 discusses three approaches for designing middleware in sensor networks. We conclude by presenting some thoughts on the current status of sensor networks.

2 What about sensor Networks?

As mentioned in section 1 sensor networks aim to provide different information through small, low price, low power and low computation devices. Ranging from pulse (heartbeat) sensors to video cameras, there are all sorts of *feelings to be sensed*. Many sensors incorporate more than one data acquisition module, which enables them to form networks that can serve more than one purpose ([1]).

As sensors have small resources they usually limit their activity to collecting measurements and transmitting the data to a base station that is a more resourceful node (a PDA, a laptop, etc). In some approaches this base station is mobile (a patrol node) and collects data by entering the transmission region of the sensors (as illustrated in fig. 1). In others, some sensors act as gateways and the information travels several sensor hops until it reaches the base station.

Naturally, the aforementioned mode of operation requires defining routes, and using routing protocols. Some proposals define hierarchies where sensors group themselves in clusters, with a responsible cluster head. In some cases this cluster head is a more powerful node, enabling different schemes. In other cases, this role rotates among nodes to distribute the effort by the data gathering sensors.

On various proposals sensor nodes aggregate and combine data received from other nodes before forwarding it to the next sensor. This occurs when the computation is deemed less energy consuming than simple forwarding and capitalizes from the correlation of data between sensors. However, there are cases (e.g. video cameras) for which aggregation is not an option.

Depending on the objective of the application using the network, sensors may (i) provide continuous information (e.g. room temperature level), (ii) send information only when a threshold is met (temperatures above some value dependent on local weather can indicate a fire), (iii) or act on demand where the base station ask for data (query the current temperature on the pool).

In the next sections we will discuss routing and middleware bearing this diversity in mind.

3 Routing

This section bases its contents on the survey by Kemal Akkaya and Mohamed Younis [3], which addresses several of the routing protocols used in sensor networks. Here, we will mention their categorization and provide brief descriptions of protocols for each of the proposed classes.

As mentioned in section 2 sensor networks data paths aim to transport data from multiple points (the sensors in the area) to a single point (the base station that handles the information) [1]. The fact that the collected information is closely coupled can be exploited to minimize the transport of redundant data. The main objective of the protocols discussed next is to minimize the overall energy consumption and thus maximize the lifetime of the sensor network, while still guaranteeing that sensor information is forwarded through the network to the base station.

The survey in [3] groups the protocols as follows:

– **Data-centric** – the focus of these protocols is answering data queries from the base station. This precludes the necessity of address schemes for the sensors, enabling the approach of flooding the query to the interesting area and have nodes gather and combine the answer to be provided to the base station. To issue the query, some form of attribute naming must be defined for the query language.
– **Hierarchical** – the protocols of this category aim to reduce the flooding and the delay of data aggregation by defining a hierarchy in the routing path.
– **Location-Based** – these routing protocols use the location information to more directly query the sensors of the area of interest of the base station.
– **Network Flow and QoS Aware** – the protocols characterized in this group use QoS (Quality of Service) parameters for the cost function in the links and/or use flow values to route the data.

These categories are not disjoint in the sense that some of the protocols fall in more than one of the divisions. The authors of [3] provide a table with the classification of the protocols studied and their characteristics.

In the next sub-sections we will briefly describe some examples for each of the categories.

[1] Note that There can be more than one base station, nonetheless the flow paradigm is the same, as usually the data is transferred to the base stations on demand and not multicast.

3.1 Data-Centric

- **Direct Diffusion** – [4] defines attribute value pairs to query the needed information from the sensor network. Based on these pairs it is possible to broadcast our *interest* on the information, which is flooded throughout the network. Messages of this kind enable the inference of a gradient, defined by the data rate, duration and expiration time of the interest message. Sending this interest through specific nodes reinforces the path, thus enabling the definition of the trail. The interests are compared with received data from sensors to devise which nodes to transmit to. In Direct Diffusion nodes perform data aggregation, i.e. they combine the data received with their own collected measurements.
- **Rumour Routing** – in [5] when an event is generated/perceived by the sensors, they send a long lived packet that other sensors can cache. This serves to populate a local table leading back to the node that sensed the event. When a query is issued the sensors can use their local table to direct it to the correct node.
- **COUGAR** – [6] defines a new query layer[2] that enables query based declarative languages using in-network-data. COUGAR requires synchronization. Leader nodes are defined to aggregate data and communicate with the base station. This approach treats the sensor network as a distributed database.

3.2 Hierarchical

- **LEACH** – Low-Energy Adaptive Clustering Hierarchy [7] defines clusters based on the energy of the received signal. It defines cluster heads that are used as routers to communicate with the base station. These cluster heads are elected in a nearly random way, based on the desired cluster head percentage over the total number of nodes. The election algorithm enforces some rotation to distribute the "burden" by all sensors. In LEACH each sensor must be able to directly "talk" to the cluster head and the cluster head must also be at one hop from the base station.
- **PEGASIS** – Power-efficient GAthering in Sensor Information Systems [8] defines paths as chains from each sensor to the base station. It includes the definition of a leader node, which communicates directly with the base station. The data gathered along the chain is aggregated to reduce traffic. Hierarchical PEGASIS [9] defines tree like hierarchy chains that reduce bottleneck problems and delays inherent in PEGASIS. It also defines schemes for simultaneous transmission (using for example CDMA).

3.3 Location-Based

- **GAF** – Geographic Adaptive Fidelity (GAF) presented in [10] tries to preserve energy by turning off some sensors located in the same region. The location information is used to define sections where nodes are grouped. Each

[2] As such COUGAR can be viewed also as a middleware for sensors, which is addressed specifically in section 4.

node on the same area is considered at the same path cost and thus can be put in sleep mode. The authors define three states for each node: discovery (finding neighbours in the same area), active (participating in routing) or sleep.

3.4 QoS Aware

- **Maximum lifetime energy routing and data gathering** – These approaches try to maximize the lifetime of the sensor network. [11] uses the remaining sensor energy and the required transmission energy at each link as input to the cost function for the link, then uses Belman-Ford shortest path to define the path. [12] uses data-gathering heuristics. It defines sensor network lifetime as the number of periodic readings from sensors until the first one dies. It elaborates on an algorithm to produce schedules that define data aggregation tree based paths for each of the periods. The algorithm defines a schedule that maximizes the lifetime of the network.
- **QoS Routing** – SPEED [13] uses geographic information and *"packets' speed"* to infer the delay in end-to-end communication. Packet's ACKs are used to estimate the delay between neighbouring nodes. [14] uses the sensor's energy, the transmission energy needed, error rate and other QoS parameters to define the cost function. It uses class-based queuing to allow for different types of traffic.

4 Middleware

There is a current trend on trying to define a middle layer between the application that uses the sensor network and the sensor network itself. This layer would manage the network, getting the information from it and defining its operation using as input the application's requirements and the reliability attached to those requirements.

The rationale behind the approach is that the usage and functioning of the sensor network is highly mandated by the application. The required number of nodes, the geographic coverage, threshold reports, continuous monitoring, mobility, and so on, are defined by the goals of the application utilizing the network. Consequently, the natural solution would be to customize network management (what nodes are needed, can data be aggregated, what is the degree of reliability needed, etc) for each application. The middleware approach tries to provide a common layer that allows this customization. The application can define its necessities (through the middleware's application program interface, API) and the middleware will control the sensors to optimize data deliverance. The key contribution of this approach is a common interface for applications regardless of its objective — customization is taken care of by the middleware. In the next subsections we will briefly address related proposals.

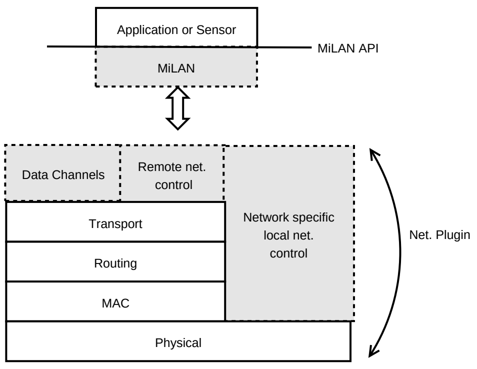

Fig. 2. MiLAN network plug-in architecture example based on [15]. Gray boxes correspond to developed components of MiLAN.

4.1 MiLAN

Middleware Linking Applications and Networks (MiLAN) [15] tries to address the issues raised, by tackling what they devise as the features of sensor applications: distribution, dynamicity in the availability of sensors, constraint application QoS demands, resource limitation (bandwidth and energy) and cooperative applications. This last issue is related to different applications using the same network to achieve different objectives; as such they must cooperate or at least not *"step in each other toes"*. MiLAN tries to cope with different application requests using their QoS requirements as input. It also takes into account the network information (energy and bandwidth) and the system's information on the relevance/precedence of the different applications.

The authors propose a middleware that also dwells on the network stack, so as to take into account and control the network properties- The main objective is to vary the network parameters over time and maintain the QoS needs for the applications.

MiLAN uses as input: the variables that the application requires, the required QoS for each variable and the level of QoS that each sensor or group of sensors can provide for each variable. This information is defined using *"State-based Variable Requirements"* (for the QoS of each variable) and *"Sensor QoS"* graphs (defining which sensors are used to satisfy the QoS). These graphs enable the definition of sets of sensors to fulfil the requirements of the applications.

The MiLAN plugin in the network stack is responsible for using the information of the network to decide what sensors are to be used and for what purpose. It uses a service discovery protocol to query the network on the attributes of the nodes. It should be possible to influence the states of the sensors as well as the routing protocol used to get the most of the network and define a set of sensors on the network capable of evaluating the needed information.

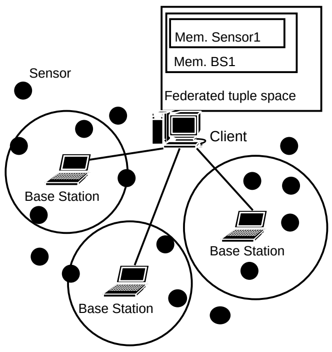

Fig. 3. TinyLime architecture example based on [16]. Denoting managers (clients), base stations and sensors. The federated tuple space in the client is also portrayed as a shared memory.

The final set of available sensors is defined by the intersection of the set given by the network plugin and the one from the upper layer given the QoS requirements of the application. This set is used to choose the sensors such as to maximize the time that the information can be given. Nonetheless, MiLAN is able to incorporate rules to sacrifice quality in order to extend the lifetime.

4.2 TinyLIME

[16] is based on a middleware model for Mobile Ad-hoc NETworks (MANETs) named LIME (Linda in a Mobile Environment). LIME [17] itself is based on LINDA. LINDA [18] uses a shared memory model to represent data. It defines tuples (typed fields) to hold the data structures. The coordination between processes is based on reading and writing on these tuples on the shared memory. It defines basic operations for: adding tuples (**out**), removing (**in**) and reading (**rd**). The operations have blocking and non blocking variants. The removal and reading can use patterns (defined also as tuples) to query for the data.

LIME breaks the tuple space on several spaces in order to decouple the shared memory from space and time. LIME is intended for mobile environment and as such the nodes are not available all the time. The *break* allows nodes to have each one a tuple space that is synchronized when the nodes are accessible. As such, the shared memory is being reshaped according to nodes' connectivity, creating a federated tuple space. The operations are extended to include a location parameter, which enables to indicate which tuple-space to query/write. LIME also adds reactions that enable an effect when a tuple matching a pattern is found in the tuple space.

In TinyLIME, the authors consider that it is not feasible to know all sensor locations, that multi-hop communications to the base station enforce a great burden on the sensors forwarding the data and that it is not feasible to assume that all sensors can communicate with the base station. As such they define a model where the sensors only communicate with a one hop mobile base station that recalls sensor information. Managers of the data are clients of these mobile base stations. They define the model as a mixture of the flexibility of MANETs (as it uses its routing capacities between the managers and base stations) and the sensor network capabilities (sensing the environment). In this model sensors are different from other nodes, as they are only visible when there is a base station on their range (see fig. 3). In that case, their tuple-space is visible to base station, being part of it. However remove or modify operations are not possible in sensor data as they are considered read-only. TinyLIME introduces conditions to the reactions operations and a freshness parameter to these reactions. The last addition enables having a frequency for refresh to the queried reaction.

As can be seen TinyLime is more dedicated to data retrieval issues, providing a middleware that hides the details of retrieving data and using the simple operations of LINDA.

4.3 Energy Efficient Resource Allocation

[19] dictates as its primary goals to define an architecture to provide (a) a common standardized system to applications with diverse objectives, (b) an environment capable of supporting and coordinating multiple applications and (c) a means to use the sensor network resources efficiently and adaptively. As in MiLAN (see section 4.1) it intends to optimize the computation, communication and sensing (CCS) energy spendings. For such, it enables nodes to sleep (thus saving energy) and it defines guarding nodes that are responsible for detecting a target phenomenon and selectively wake up the sleeping nodes. The coordination of these active nodes to sense the phenomenon, aggregate data and route the decision to the base station is a key point.

As in MiLAN it also aims to use the application knowledge, but it tries to reach a balance between the specificity of the application and the generality needed by the middleware to cope with very different applications. It also tries to balance the different needs of each application to reach the greater common benefit possible. For this it suggests the use of adaptive fidelity algorithms and inter-application coordination.

It uses a cluster based approach where each cluster is considered a basic unit of the middleware, functioning as distributed software. The clusters are considered dynamic as the phenomenon to be sensed is. The nodes' resources also impose dynamicity on the cluster, with nodes leaving and joining the cluster to optimize energy. As such, on-the-fly self-configuring distributed clustering mechanisms are addressed. Periodic information regarding nodes CCS capabilities and energy should flow to the cluster head through efficient mechanisms.

Aggregation and comparison of data is also considered mandatory, and for this purpose distributed and energy-aware protocols should be used. Inter-cluster coordination is also of importance with cluster heads seen as another source of information from other clusters.

The proposal defines a three-phase heuristic to cope with the problems of distributing sensor tasks (see [20] for further details).

5 Conclusion

In this article, we introduced some of the problems in sensor networks. We presented some of the current proposals that address routing and middleware.

The routing protocols described try to minimize energy consumption while maximizing information delivery. Data-centric ones focus on broadcasting queries and defining methods for the sensors to forward the query to the appropriate nodes. Some approaches rely more deeply on defining a hierarchy for the data paths. As such they define clusters or hierarchy levels and rotate the leader nodes so to distribute the load. Others rely on the location information to selectively turn on/off sensors based on their proximity. We also described protocols that are more dedicated to QoS measures. They use network delay (and other QoS parameters) as part of the cost function for traversing links. None of these approaches can be categorized in one group, for example COUGAR is more prone to a data-centric approach, but also defines some hierarchy levels. One could say that data-centric and location based approaches are better suited for monitoring event based phenomenon whereas QoS aware protocols are more appropriate for real time monitoring.

Middleware solutions try to provide a common framework to all applications while trying to customize the sensor network to fit the applications needs. MiLAN and Energy Efficient Resource Allocation are similar in that they provide an API for input of QoS needs by the applications. The first expects to know which sensors are intended to yield the measured parameters and which quality they provide for that result. The second strives for a simple and general API with some focus on cluster definition in order to optimize CCS energy consumption. TinyLime provides a shared memory model approach where the given API enables the configuration and retrieval of sensor information. The focus is on information gathering based on a single approach regardless of the underlying network mobility and state.

The common view seems to be that there is no *"one size fits all"* solution – sensor networks are in essence application-specific – but middleware appears as a reasonable option.

There are two set of issues, which remain on our agenda: (1) security issues, which are either related to the vulnerabilities of the infrastructured world and the ad-hoc networking paradigm or specific to sensor networks; (2) data gathering, in its own is yet another source of ongoing research.

References

1. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirc, E.: Wireless sensor networks: a survey. In: Computer Networks 38, Elsevier (2002) 393–422
2. Akyildiz, I.F., Kasimoglu, I.H.: Wireless sensor and actor networks: research challenges. In: Ad Hoc Networks 2, Elsevier (2004) 351–367
3. Akkaya, K., Younis, M.: A survey on routing protocols for wireless sensor networks. In: Ad Hoc Networks 3, Elsevier (2003) 325–349

4. Intanagonwiwat, C., Govindan, R., Estrin, D.: Directed diffusion: a scalable and robust communication paradigm for sensor networks. In: Proceedings of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'00). (2000)
5. Braginsky, D., Estrin, D.: Rumor routing algorithm for sensor networks. In: Proceedings of the First Workshop on Sensor Networks and Applications (WSNA). (2002)
6. Yao, Y., Gehrke, J.E.: The cougar approach to in-network query processing in sensor networks. In: Sigmod Record 31(3), Sigmod (2002)
7. Heinzelman, W., Chandrakasan, A., Balakrishnan, H.: Energy-efficient communication protocol for wireless sensor networks. In: Proceeding of the Hawaii International Conference System Sciences. (2000)
8. Lindsey, S., Raghavendra, C.: Pegasis: power efficient gathering in sensor information systems. In: Proceedings of the IEEE Aerospace Conference. (2002)
9. Lindsey, S., Raghavendra, C., Sivalingam, K.: Data gathering in sensor networks using the energy*delay metric. In: Proceedings of the IPDPS Workshop on Issues in Wireless Networks and Mobile Computing. (2001)
10. Xu, Y., Heidemann, J., Estrin, D.: Geography-informed energy conservation for ad hoc routing. In: Proceedings of the 7th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'01). (2001)
11. Chang, J.H., Tassiulas, L.: Maximum lifetime routing in wireless sensor networks. In: Proceedings of the Advanced Telecommunications and Information Distribution Research Program (ATIRP 2000). (2000)
12. Kalpakis, K., Dasgupta, K., Namjoshi, P.: Maximum lifetime data gathering and aggregation in wireless sensor networks. In: Proceedings of IEEE International Conference on Networking. (2002)
13. et al, T.H.: Speed: a stateless protocol for real-time communication in sensor networks. In: Proceedings of International Conference on Distributed Computing Systems. (2003)
14. Akkaya, K., Younis, M.: An energy-aware qos routing protocol for wireless sensor networks. In: Proceedings of the IEEE Workshop on Mobile and Wireless Networks. (2003)
15. Heinzelman, W.B., Murphy, A.L., Carvalho, H.S., Perillo, M.A.: Middleware to support sensor network applications. In: IEEE Network, IEEE (2004)
16. Curino, C., Giani, M., Giorgetta, M., Giusti, A., Murphy, A.L., Picco, G.P.: Tinylime: Bridging mobile and sensor networks through middleware. Third IEEE International Conference on Pervasive Computing and Communications (PerCom'05) (2005) 61–72
17. Murphy, A.L., Picco, G.P., Roman, G.C.: Lime: A middleware for physical and logical mobility. In: Proc. of the 21st Int. Conf. on Distributed Computing Systems (ICDCS). (2001) 524–533
18. Gelernter, D.: Generative communication in linda. ACM Computing Surveys 7(1) (1985) 80–112
19. Yu, Y., Krishnamachari, B., Prasanna, V.: Issues in designing middleware for wireless sensor networks. In: IEEE Network 18(1), IEEE (2004) 15–21
20. Yu, Y., Prasanna, V.: Energy-balanced task allocation for collaborative processing in wireless sensor networks. In: Algorithmic Solutions for Wireless, Mobile, Ad Hoc and Sensor Networks 10(1-2), Springer (2005) 115–131