

A Protocol Architecture for Wireless Sensor Networks

Siddharth Ramesh

School of Computing, University of Utah,
Salt Lake City, UT 84112
sramesh@cs.utah.edu

1 Introduction

Wireless Sensor Networks (WSN) are a new class of networking technology that is increasingly becoming popular today. Huge strides taken in sensing technology, low-power microcontrollers and communication radio have spurred the mass production of relatively inexpensive sensor nodes. Such large scale sensor networks far outweigh use of conventional networks in situations where terrain, climate and other environmental constraints hinder the deployment and setting up of regular networks. Because of the tremendous scale at which such nodes can be deployed, they are extremely robust in terms of individual node failures which makes them all the more favorable in such extreme situations. There has been an explosion in the use of sensor networks for environmental measurement and study. Also, the speed with which a sensor network can be set up renders them very useful in military applications such as monitoring and tracking.

A range of applications have been built using sensor networks, from environmental monitoring [14, 13] to radiation detection [4] to lots of tracking applications [7, 5, 10].

Broadly sensor applications can either be categorized into data gathering or tracking. Data gathering applications use sensor nodes to periodically measure the value of a particular environmental variable and recorded values are collected by a sink node for further processing. Tracking applications, on the other hand, continually monitor the environment for the presence of signals which can uniquely identify an object being tracked. For example, an acoustic signal unique to a car can be used to detect the motion of a car in the region being monitored.

Though such a rapid increase in the number of very application specific sensor networks has definitely been a great boost in terms of protocols and techniques developed, it has created two basic impediments to more growth:

- Lack of interoperability between individual components of different systems
- Lack of a common framework on which new developers can build

Because almost all of these applications are custom built with no standardization in the protocols used for communication, there can be no interoperability between two components developed separately by different research groups. With interoperability, there will be a unification of various efforts in sensor networks.

Also, there is no basic architecture which everyone can use to build on, forcing each developer to build each component in their application from scratch. If there is a common framework of integral components to be used, it would help save greatly in development and testing time since not everything has to be built from scratch everytime.

In this paper we focus on building a standard protocol architecture for communication in sensor networks. Any protocol framework must be resource-aware. Thus the two main requirements of a protocol architecture are:

- (a) It should have a composable framework
- (b) It should be resource aware

Composability, as mentioned earlier, is essential for developers of different applications are able to compose functionalities depending on the needs of the application. So the underlying architecture must be able to support multiple types of application needs.

Resource constraints are one of the major drawbacks on sensor networks. Since sensor nodes run on battery, which cannot be replenished, it is vital that it runs very efficiently, in terms of sensing, computation and communication. Sensing and computation activities, compared to communication are very efficient. It is the communication activities of transmitting and receiving which take up most of the energy. So *resource awareness* should be inbuilt in the protocol architecture for efficient communication.

In this work, we discuss the issues in building a protocol architecture for sensor networks. In particular, we build on the proposal of a having a common Sensornet Protocol in [1]

In the proposed architecture, we make a distinction between data and control packets in sensor networks. We propose that in addition to having a protocol layering like the TCP/IP stack and a common layer like SP, segmenting the communication protocol architecture into a data and control plane is very essential to make best available use of resources in sensor networks. We make this distinction from the fact that there are huge differences in the requirements of control and data messages in sensor network communication. The basic difference is that data packets do not require as much of a reliability guarantee as control packets. Control packets on the other hand require a hard guarantee of reliable delivery.

We also observe that even for control packets sent by the one node of a sensornet, complete reliability in terms of that packet reaching all nodes in the network is not always necessary. For example, the sink node may just want to query the network and make sure that *atleast* a required portion of the network is up. It would be a waste of the network's resources if the query propagates the entire network, all the nodes which are up respond to the sink, and then the sink calculates whether the required number of nodes are up. In this regard, we introduce the concept of *semi-reliability* in sensor networks, wherein a node can choose to send a control message to only a part of the sensor network, and not to all nodes. This will be very useful in network management, where the sink can query parts of the network at a time to check if that part of the network is congested or not.

There have been very few efforts in the past on protocol layering in sensor networks. The research proposal on the introduction on Sensornet Protocol (SP) [1] as a common protocol layer for sensor networks was the first consolidated effort in this direction. SP is intended to be a "narrow waist" common layer for all sensor networks, just as the Internet Protocol (IP) is common across all computers connected to the Internet. SP

is defined to be a *best-effort single hop broadcast protocol*. We build our protocol architecture on top of this SP-layering. There has, however, been no previous effort in trying to define a separate control plane in sensor networks for separating the treatment of control and data packets.

In the next section we present the architecture proposed by [1] and explain the positioning and functionality of SP. In section 3, we will introduce the concept of Control Plane for Sensor network and merge it with the architecture. In Section 4, we detail the aspects of semi-reliability in sensor networks and how it can be achieved under various requirements. Section 5 summarizes this work and presents future directions in research.

2 A Sensor Network Architecture with SP

SP, according to [1], is a *best-effort single hop broadcast protocol* which is situated between the network and the link layer in the protocol stack. The placement of SP between the network and link layers is quite obvious. As any sensor network application is closely tied to the network layer routing protocols, the common layer of SP cannot be at the network layer. And similarly there are numerous link layer protocols and each Sensor-net application developer might prefer to be able to choose a protocol which will be best suited for the application. And the functionalities offered by the various link layers are widely varying in nature.

SP is basically an interface between the network and the link layer abstracting the functionalities of the link layer and offering them as services to the network layer. [1] talks about how SP should be able to provide the network layer functionalities such as generation of link level acknowledgements, performance of initial and collision-avoidance backoff, timestamping, error control, retransmission and power management. SP has to abstract over the numerous link layer protocols and provide a common interface to the network layer.

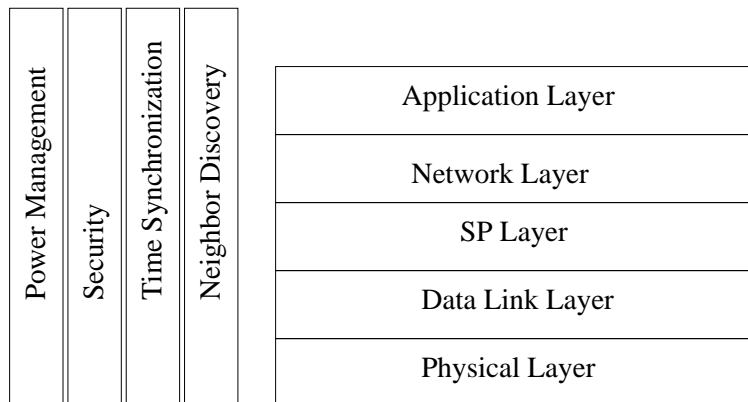


Figure 1: Sensor Network Architecture

Figure 1 shows the network architecture as proposed by [1]. As is seen, the SP layer is an interface between numerous link and network protocols. There are also many Verticals like *Security*, *Power Management*, *Neighbor Discovery* and *Time Synchronization*. These verticals provide cross-layer functionalities as a set of interfaces, which can be used by all of the network layers.

For example, though neighbor discovery is a responsibility of the link layer, the neighbor information will be used by all layers. So, this *Neighbor Discover* vertical would ideally abstract from the link or SP layer and present interfaces for querying. Similarly, *Time Synchronization* between neighbors could be needed at the application level if the application is a tracking application which requires highly synchronized data for Signal Processing. At the same time, time synchronization could also be needed at the link level if some form of TDMA is used. *Power Management* is a very essential cross layer functionality needed for resource optimization in a sensor node. The network layer might use it to find the power remaining, before it makes a decision on whether to route a packet or not. At the same time, a link layer might use it before deciding to retransmit a frame. *Security* is another vertical which could consist of various methods to encrypt data and might be invoked either at the application, network or link layer.

3 Control Plane in Sensor Networks

The layering of the protocols and functionalities is very essential for the standardization and development of a common framework for all future work in sensor networks. And the introduction of a common SP-like layer is needed for interoperability between individual components developed at different places. But since resource constraints are one of the major drawbacks of sensor nodes, any architecture should be resource-aware inherently. This will be possible only if there is a distinction made in the architecture for different types of communication requirements, especially with respect to reliability.

Communication in sensor networks can be broadly classified into two classes - data and control messages. Data messages, which will clearly be in majority in any application, are those messages which involve transfer of sensor data in one way or the other. Control messages broadly encompass all communication which do not involve transfer of user/sensor data. They include messages used for configuration, topology detection, neighbor discovery, route discovery, time synchronization, reprogramming, reliability and network management such as congestion and flow estimation and recovery.

The main difference between data and control messages is the fact that control messages are usually much more precious than data messages. The loss of, say a control message by the sink to re-program nodes in a particular region is costlier than the loss of a sensor value while it is being transmitted from a node to the sink. In general, more care should be taken in the reliable delivery of control messages. With regard to reliability, we should also clearly define what are the various reliability semantics essential. But whatever be the range of semantics, it must be made clear that these need to be present only for control messages. Inclusion of reliability for all data transfer is very much against the energy conservation principles for sensor networks. However there might be instances when data communication might also need reliability. However we trivialize this case by considering those data messages as a form of control traffic.

Also, control messages will have a higher priority than normal data messages, to make sure that in the event of a congested input queue in a node, it gives higher priority to control messages and does not drop them. This is to ensure that control messages such as those used to congestion and flow control do not themselves become victims of congestion.

Figure 2 shows the architecture of sensor networks after including the control / data plane abstraction. The data plane is the same as the architecture of 1. But the control plane has a transport layer which takes care of reliable data transmission of control messages. The transport layer, just as in TCP, would provide *reliable*

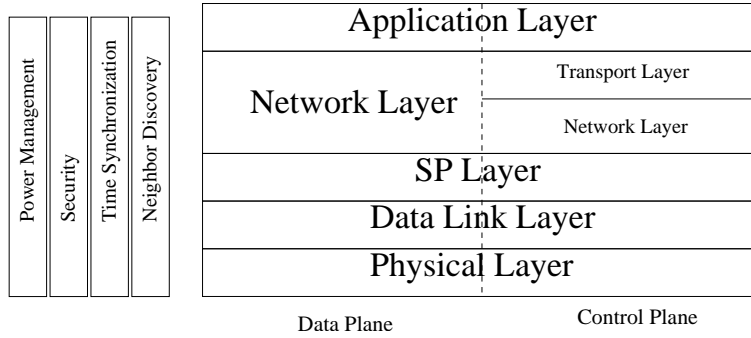


Figure 2: Data and Control Plane Architecture

transmission of messages, congestion control, and flow control. But just using TCP in sensor networks is not a good option as TCP was built and optimized for wired networks and there are lots of factors in wireless networks which are not addressed by TCP. There have been efforts in the past to make TCP viable over wireless adhoc networks like in [3, 15]. [19, 11] also suggest changes in TCP for making it suited for wireless adhoc networks. But even adhoc networks differ from sensor networks quite a bit in that resource awareness and optimization is much more essential in sensor networks than in mobile adhoc networks.

3.1 Congestion Control

A major problem with sensor networks is that they communicate through radio, and air is a broadcast medium by default. Almost all of the link protocols for wireless transmissions are also broadcast inherently. Because of this, congestion is a very realistic concern in sensor networks. The fact that the cost of retransmission of a lost frame is very high for the energy-constrained sensor node exacerbates the problems of congestion. Thus there must be effective means to detect congestion and control it once detected.

Congestion control in TCP is very different from that in sensor networks. In TCP, both congestion and reliability are coupled with the receipt of an ACK from the receiver. TCP assumes the non-receipt of an ACK as a congestion problem and it slows down its transmission rate along with retransmitting the packet for reliability. This is because dropping of packets due to errors in wired networks is very less. However in wireless networks, this is a huge problem as error rates are usually high in wireless medium. Thus, congestion in sensor networks' transport layer has to be treated more of a network problem.

With regard to congestion control, [8] talks about some of the earlier mechanisms such as *Hop-by-Hop Flow Control* [12, 18], *Rate limiting* and *Prioritized MAC* [2] for solving common congestion problems. It also presents a new algorithm *Fusion* which combines the effectiveness of the above three methods.

Hop-by-hop flow control works by setting a congestion bit in the header of every outgoing packet depending on the congestion at its input queue. Mitigating congestion is based on reading the congestion bit of neighbor's transmission and not transmitting to them if their congestion bit is set. *Rate limiting* is based on each sensor node listening to its parent's communication to find out the number of unique nodes routing through its parent. It then bases its rate of transmission to its parent based on that number. *Prioritized MAC* advocates making the amount of backoff in the CSMA MAC protocol a function of its congestion state, so

that a congested node waits for a lesser amount of time before it transmits. Any of these methods can be used for congestion control in sensor networks, though hop-by-hop flow control is the most generic in the sense that it assumes neither a fixed topology nor a particular link layer. Also, hop-by-hop flow control can be used to find out congestion in parts of the networks as we will discuss in Section 4.

[17] proposes a congestion control mechanism for mobile adhoc networks, which could be extended to sensor networks also. In this, a sender sends a control packet to one part of the network. All the intermediary routing nodes fill in congestion information and it reaches the receiver (which is usually the leaf) node. The receiver collates the information of all the nodes in this congestion information packet. It then sends the sender the maximum delay at any node along the path. The sender can then adjust its rate of transmission depending on the information in this packet.

3.2 Flow Control

Flow Control in sensor networks is not that much of a problem because of the scale of data which is being transmitted in it. In sensor networks the scale of data is usually very less compared to a wired network, since all the nodes have to transmit is the data they sense at regular intervals and control information occasionally. Flow control also usually manifests as a congestion problem in the area surrounding the receiver, and it backpropagates up to the sender, which can then reduce its reduce its transmission rate.

3.3 Reliability

Reliability encapsulates two requirements

- Reliable delivery of the messages
- Error free delivery

As mentioned earlier, the error rates in wireless networks are particularly high, and the problem in sensor networks is worse because of the energy limitations of the sensor nodes. Hence the separation of congestion control from the receipt of ACKs. But both reliable and error-free delivery of messages depend on the receipt of ACKs from the receiver. There are two major issues in retransmission for reliability based on ACKs - that of deciding the timeout for retransmission and that of deciding where to retransmit from.

Deciding timeout is crucial in networks as a small timeout will lead to flooding the network with duplicate packets and a large timeout leads to delayed loss recovery and under-utilization of the network. In protocols like AODV [16] where a route is found before transmission of data begins, this is not much of a problem. But most of the network protocols do not belong to this category. So intuitively, a timeout on the higher side can be chosen initially and can be varied depending on the time to receive ACKs, like how TCP does. But in general, in sensor networks, control packet transmissions will be for short periods. Also, burstiness of traffic affects this calculation badly.

Where to retransmit from might have an obvious answer in normal networks, but in sensor networks end-to-end retransmission is not the most energy-efficient. [6] suggests a method *Distributed TCP Caching (DTC)* which actually pertains to using TCP/IP over wireless networks. DTC uses segment caching and

local retransmissions with the cooperation of the link layer. A solution borrowed from this using caching of the segments at intermediary nodes can improve the performance manifold. Whenever a sensor node gets a control packet to be forwarded, it first caches the segment and then forwards it. In case it does not receive an ACK within a local timeout, it can resend the segment from within its cache, instead of letting the sender timeout and retransmit the packet the whole way again. But this comes at an extra overhead of maintaining states in its buffer. So there has to be a judicious decision of what and how many segments to cache.

There are two ways in which reliability could be enforced - one using end-to-end ACKs and the other using link-level ACKs. In general, especially if the link layer is ACK based, a link-level ACK scheme will be more energy efficient. Any of the intermediate nodes could send back a NAK if they are unable to deliver the segment. This could be because of many reasons - the TTL of the segment could have reached zero, the packet could have got corrupted or a node would have been unable to send it to its proper neighbor because the neighbor might be dead. In this scheme of reliability, unless the sender receives a specific NAK from any one of the intermediate nodes, it will have to assume that the packet reached the receiver successfully. The advantages of this method are

- (a) It eases the calculation of a time-out by the sender. Since any node will know (from topology creation by the link layer) the time of transmission to its neighbors, the timeout value is more accurate. End-to-end timeouts are not needed.
- (b) Since transmission link by link is assured to be reliable, the problems that arise in DTC (many nodes in a chain retransmitting a cached segment at the same time) do not arise here.

One of the main disadvantages of this approach is that this will method might prove to be costlier if the link layer does not implicitly support link-level acknowledgements.

4 Semi-Reliability in Sensor Networks

The introduction of a control plane in Sensor Networks is to make sure that ensuring reliability in the presence of different reliability requirements by different types of messages do not lead to a wastage of resources. But many times needs arise in sensor networks which require the sink to send control messages to only a part of the network or to a particular number of nodes in the network. For example, the sink might want to reprogram nodes in one geographic location so that it starts to sense data at a different sampling rate; the sink might want to check if a required number of nodes are alive in a particular region; or the sink might even wish to measure the congestion in one part of the network. All these require semi-reliability in delivery of messages. Semi-Reliability refers to a part of the network reliably receiving control messages.

Below we describe many forms of semi-reliability and how they can be achieved. Here we assume a tree-like topology in the network, which is usually readily achieved with the sink being at the root of the tree. So each node will know one of its parent (one of its neighbor node closer to sink than it) and all its children (all its neighbors which are one hop farther away from the sink than it). We do not assume though that the sink knows the entire topology of the network as a) it trivializes the problem and b) the topology will keep changing as nodes die and the sink will have to update its information regularly.

- 1) **Depth-search based semi-reliability:** Suppose the sink node of a Sensor Network wants to send out

a control message and make sure that atleast a certain minimum number of sensor nodes receive it. It is a waste to send to more than the required number also, since communication is very expensive for a sensor node. In this case, each parent node (starting with the sink) sends the control message to one of its children. This way the query propogates down the length of the network like a depth-first search. When the message reaches the leaf node, or if the required number of nodes have got the message, ACKs are sent back parent by parent back to the sink. The ACKs are aggregated on their way back to the sink further reducing the number of messages transferred.

- 2) **Incremental Increase in TTL:** A similar requirement as in the previous case can also be satisfied by the sink node sending out these control packets with increasing TTL. So, with $TTL = 1$, it will reach all its neighbors. If it is not sufficient, it will send the message with $TTL = 2$, and so on till the required number of nodes have responded. This would be like a bread-first search of the tree rooted at the sink. In order not to waste power, the nodes in the level i can cache the segments so that if the sink sends the same message with a $TTL = i+1$, then the sink need not actually transmit the message again, but only a small indicator packet to signal level i to further transit the cached packet.
- 3) **Cluster based semi-reliability:** Most often when there are clusters formed in the network, the sink node might want to contact all nodes in a particular cluster for sending some control messages. If the clusters are static clusters, i.e., formed at the time of network creation and are static through out the life span of the network, then the sink will know the members of the clusters and how to address them. In this case, all the sink has to do is to send the control segment to the cluster leader which in turn will broadcast it to the cluster members. However, if the clusters are formed dynamically, as in the case of tracking applications, then the sink has no way of knowing which nodes belong to a particular cluster, where that cluster is located, who its leader is etc. Even if the sink comes to know of these from previous messages from the cluster leader, as these clusters are dynamic in nature it is not necessary that when the sink tries to contact the cluster leader again, that the cluster still exist.

However, the sink might want to send a particular piece of control information to all the nodes in a particular region. Then, this might be similar to Geographic Forwarding [9] with the exception that the destination is a group of node in a specific region instead of a specific node. The sink could address the region with the centre and radius of the circle which most overlaps the region. More complex location specifiers can also be used. The nodes could greedily forward it, until any cluster head in that particular region gets it. Cluster heads could then broadcast the packets to members of its cluster, communicate and pass on the packets to other cluster heads in the same region and collect ACKs from its cluster members and pass them on back to the sink.

5 Conclusions and Future Work

In this work, we argued the necessity of a standad protocol architecture for sensor networks. Building on the research proposal to include a common IP-like Sensornet Protocol (SP) in the architecture, we focussed on the reliability aspects in the architecture. The differences in reliability requirements of data and control messages were addressed. The introduction of a control plane to separate reliability concerns of data and conrtrol messages was a unique step in this direction and is the main contribution of this work. Further, another aspect of reliability namely Semi-Reliability was studied in the context of sensor networks and ways proposed on how to achieve them. There is lots of scope for future work since the work on standardizing

architecture for sensor networks is relatively new. First of all, the proposals of this work have to be given more shape and made concrete. Then they would have to be simulated and empirically tested to analyze the applicability and robustness of the solutions.

References

- [1] Nets-noss: Creating an architecture for wireless sensor networks.
- [2] Imad Aad and Claude Castelluccia. Differentiation mechanisms for iee 802.11. In *INFOCOM*, pages 209–218, 2001.
- [3] Hari Balakrishnan, Srinivasan Seshan, Elan Amir, and Randy H. Katz. Improving tcp/ip performance over wireless networks. In *MobiCom '95: Proceedings of the 1st annual international conference on Mobile computing and networking*, pages 2–11. ACM Press, 1995.
- [4] Sean M. Brennan, Angela M. Mielke, David C. Torney, and Arthur B. Maccabe. Radiation detection with distributed sensor networks. *IEEE Computer*, 37(8):57–59, 2004.
- [5] W. Chen, J. Hou, and L. Sha. Dynamic clustering for acoustic target tracking in wireless sensor networks, 2003.
- [6] Adam Dunkels, Juan Alonso, Thiemo Voigt, Hartmut Ritter, and Jochen H. Schiller. Connecting wireless sensornets with tcp/ip networks. In *WWIC*, pages 143–152, 2004.
- [7] Tian He, Sudha Krishnamurthy, John A. Stankovic, Tarek Abdelzaher, Liqian Luo, Radu Stoleru, Ting Yan, Lin Gu, Jonathan Hui, and Bruce Krogh. Energy-efficient surveillance system using wireless sensor networks. In *MobiSYS '04: Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pages 270–283. ACM Press, 2004.
- [8] Bret Hull, Kyle Jamieson, and Hari Balakrishnan. Mitigating congestion in wireless sensor networks. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 134–147. ACM Press, 2004.
- [9] B. Karp. Geographic routing for wireless networks, 2000.
- [10] D. Li, K. Wong, Y. Hu, and A. Sayeed. Detection, classification, and tracking of targets, 2002.
- [11] Victor H. Li, Zhi-Qiang Liu, and Steven H. Low. Enhancing tcp performance over wireless networks.
- [12] Chenyang Lu, Brian M. Blum, Tarek F. Abdelzaher, John A. Stankovic, and Tian He. Rap: A real-time communication architecture for large-scale wireless sensor networks. Technical report, 2002.
- [13] Alan Mainwaring, David Culler, Joseph Polastre, Robert Szewczyk, and John Anderson. Wireless sensor networks for habitat monitoring. In *WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 88–97. ACM Press, 2002.
- [14] Kirk Martinez, Jane K. Hart, and Royan Ong. Environmental sensor networks. *IEEE Computer*, 37(8):50–56, 2004.

- [15] Maulin Patel, Nisarg Tanna, Pratik Patel, and Raja Banerjee. Tcp over wireless networks: Issues, challenges and survey of solutions.
- [16] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (aodv) routing, 2003.
- [17] Karthikeyan Sundaresan, Vaidyanathan Anantharaman, Hung-Yun Hsieh, and Raghupathy Sivakumar. Atp: a reliable transport protocol for ad-hoc networks. In *MobiHoc '03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 64–75. ACM Press, 2003.
- [18] Chieh-Yih Wan, Shane B. Eisenman, and Andrew T. Campbell. Coda: congestion detection and avoidance in sensor networks. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 266–279. ACM Press, 2003.
- [19] S. West and N. Vaidya. Tcp enhancements for heterogeneous networks, 1997.