

# Challenges in Wireless Sensor Networks

Er. Barjinder Singh Kaler<sup>#1</sup>

<sup>1</sup> Lecturer

RIMT-MAEC, Mandigobindgarh

barjinderkaler@rediffmail.com

Er. Manpreet Kaur Kaler<sup>#2</sup>

<sup>2</sup> Lecturer

RIMT-IET, Mandigobindgarh

preet\_kaler@rediffmail.com

**Abstract :** Software development for wireless sensor networks requires novel *programming* paradigms and technologies. This article describes the concept of a new service oriented software architecture for mobile sensor networks. With this architecture, a flexible, scalable programming of applications based on an adaptive middleware is possible. The middleware supports mechanisms for cooperative data mining, self-organization, networking, and energy optimization to build higher-level service structures. The purpose of our research activities is the development of a framework, which radically simplifies the development of software for sensor network applications.

**Keywords:** WSN , Sensors, Distributed Algorithms

## 1. Introduction

Deployment of a sensor network in a target area can be a continuous process, for example to replace nodes with depleted batteries or nodes that have been destroyed due to environmental influences. In general, deployment establishes an association of sensor nodes with objects, creatures, or places in order to augment them with information-processing capabilities. Deployment can be as diverse as establishing one-to-one relationships by attaching sensor nodes to specific items to be monitored [2], covering an area with locomotive sensor nodes [7], or throwing nodes from an aircraft into an area of interest [18]. Due to their large number, nodes have to operate unattended after deployment. Once a sufficient number of nodes has been deployed, the sensor network can be used to fulfill its task. This task can be issued by an external entity connected to the sensor network, such as a user with a PDA, an aircraft flying by, or some device on the Internet. Also conceivable are isolated, self-contained sensor networks which are programmed to fulfill a certain sensing task, whose result controls actuator nodes that are also part of the network. In hybrid architectures, the sensing results control the sensors to trigger more detailed monitoring of a

certain phenomenon, which is then reported to the external task issuer.

The sensing tasks are usually rather high level, such as “report size, speed and direction of vehicles over 40 tons moving through a certain area”. On the other hand, individual sensor nodes typically provide very simple functionality, such as determining movement at a certain place. In order to solve complex high-level sensing tasks, the limited sensor nodes have to coordinate and split the task among themselves, taking into account the individual characteristics of the nodes (e.g., attached sensors, location, residual energy). The readings of the individual sensors then have to be merged in order to obtain a high-level sensing result.

Many current WSN solutions are developed with simplifying assumptions about wireless communication and the environment, even though the realities of wireless communication and environmental sensing are well known. Many of these solutions work very well in simulation. It is either unknown how the solutions work in the real world or they can be shown to work poorly in practice. We note that, in general, there is an excellent understanding of both the theoretical and practical issues related to wireless communication. For example, it is well known how the signal strength drops over distance. Effects of signal reflection, scattering and fading are understood. However, when building an actual WSN, many specific system, application, and cost issues also affect the communication properties of the system. Radio communication in the form of AM or FM broadcast from towers performs quite differently than short range, low power wireless found in self-organizing WSNs. Of course, while the same basic principles apply, the system performance characteristics vary considerably. In other words, the size, power, cost constraints and their tradeoffs are fundamental constraints. In the current state of the art, the tradeoff among these constraints has produced a number of devices currently being used in WSNs. For example, one such device is the Mica mote that uses 2 AA batteries, a 7 MHz microcontroller, an RF Chipcon radio, and costs about \$100. As better batteries, radios and microcontrollers become

available and as costs reduce, new platforms will be developed. These new platforms will continue to have tradeoffs between these parameters. Novel network protocols that account for the key realities in wireless communication are required. New research is needed to:

- > Invent new network protocols that account for the communication realities of real world environments,
- > Test the individual solutions on real platforms in real world settings, and
- > Synthesize novel solutions into a complete system-wide protocol stack for a real application.
- > Measure and assess how the theoretical properties of wireless communication are exhibited in today's and tomorrow's sensing and communication devices,
- > Establish better models of communication realities to feed back into improved simulation tools,

A sensor network normally constitutes a wireless ad-hoc network, meaning that each sensor supports a multi-hop routing algorithm (several nodes may forward data packets to the base station).

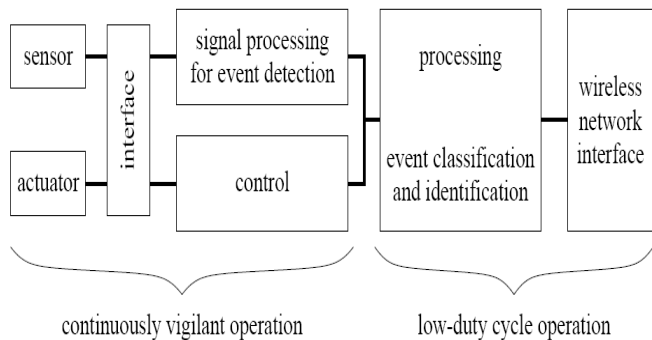


Figure 1. System Diagram for Wireless Sensor Networks (WSN)

### I. IEEE 1451 AND SMART SENSORS

IEEE 1451 is a family of standards that links sensors to users, similar to the way that IEEE 802 (Ethernet) provides connectivity for information systems. Currently, all working groups under the IEEE 1451 umbrella provide standard interfaces for sensors on tethered networks. But the demand for a wireless physical layer is growing. A wireless IEEE

1451 standard should provide seamless connectivity among sensors and users, no matter what distance separates them. And it must do this without requiring the installation of new wires and with reasonable cost and size additions at each sensor node. Wireless sensor networks should satisfy many requirements. Desirable functions for sensor nodes include: ease of installation, self-identification, self-diagnosis, reliability, time awareness for coordination with other nodes, some software functions and DSP, and standard control protocols and network interfaces [IEEE 1451 Expo, 2001].

There are many sensor manufacturers and many networks on the market today. It is too costly for manufacturers to make special transducers for every network on the market. Different components made by different manufacturers should be compatible. Therefore, in 1993 the IEEE and the National Institute of Standards and Technology (NIST) began work on a standard for Smart Sensor Networks. IEEE 1451, the Standard for Smart Sensor Networks was the result. The objective of this standard is to make it easier for different manufacturers to develop smart sensors and to interface those devices to networks.

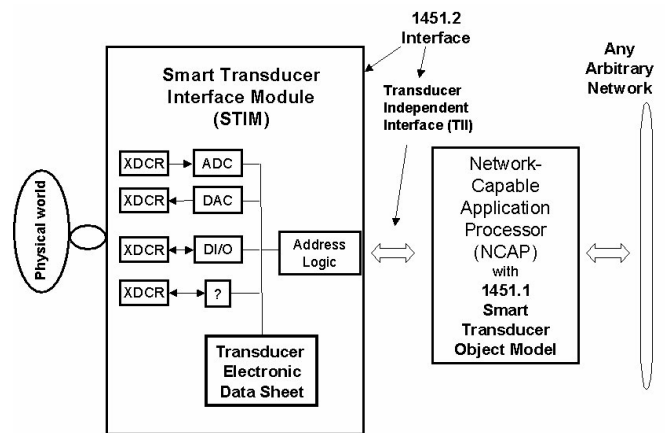


Figure 2. The IEEE 1451 Standard for Smart Sensor Network

### II. REQUIREMENTS OF SENSOR NETWORKS

Sensor networks have several requirements beyond those needed by generic data networks. They must be able to accommodate the Transducer Electronic Data Sheet (TEDS) associated with each sensor. These networks often have more stringent timing and synchronization requirements. Also, the nodes must have lower power consumption and be smaller in size than the products that support PC-to-PC networking.

## 2. Challenges in real time:

WSN deal with real world environments. In many cases, sensor data must be delivered within time constraints so that appropriate observations can be made or actions taken. Very few results exist to date regarding meeting real-time requirements in WSN. Most protocols either ignore real-time or simply attempt to process as fast as possible and hope that this speed is sufficient to meet deadlines. Some initial results exist for real-time routing. For example, the RAP protocol [1] proposes a new policy called velocity monotonic scheduling. Here a packet has a deadline and a distance to travel. Using these parameters a packet's average velocity requirement is computed and at each hop packets are scheduled for transmission based on the highest velocity requirement of any packets at this node. While this protocol addresses realtime, no guarantees are given. Another routing protocol that addresses real-time are called SPEED [2]. This protocol uses feedback control to guarantee that each node maintains an average delay for packets transiting a node. Given this delay and the distance to travel (in hops), it can be determined if a packet meets its deadline (in steady state). However, transient behavior, message losses, congestion, noise and other problems cause these guarantees to be limited. To date, the limited results that have appeared for WSN regarding real-time issues has been in routing. Many other functions must also meet real-time constraints including: data fusion, data transmission, target and event detection and classification, query processing, and security. New results are needed to guarantee soft realtime requirements and that deal with the realities of WSN such as lost messages, noise and congestion. Using feedback control to address both steady state and transient behavior seems to hold promise. Dealing with real-time usually identifies the need for differentiated services, e.g., routing solutions need to support different classes of traffic; guarantees for the important traffic and less support for unimportant traffic. It is important not only to develop real-time protocols for WSN, but associated analysis techniques must also be developed (see the section below on Analysis).

## 3. Challenges in power managements:

Low-cost deployment is one acclaimed advantage of sensor networks. Limited processor bandwidth and small memory are two arguable constraints in sensor networks, which will disappear with the development of fabrication techniques. However, the energy constraint is unlikely to be solved soon due to slow progress in developing battery capacity. Moreover, the untended nature of sensor nodes and hazardous sensing environments preclude battery replacement as a feasible solution. On the other hand, the

surveillance nature of many sensor network applications requires a long lifetime; therefore, it is a very important research issue to provide a form of energy-efficient surveillance service for a geographic area. Much of the current research focuses on how to provide full or partial sensing coverage in the context of energy conservation. In such an approach, nodes are put into a dormant state as long as their neighbors can provide sensing coverage for them. These solutions regard the sensing coverage to a certain geographic area as binary, either it provides coverage or not. However, we argue that, in most scenarios such as battlefields, there are certain geographic sections such as the general command center that are much more security-sensitive than others. Based on the fact that individual sensor nodes are not reliable and subject to failure and single sensing readings can be easily distorted by background noise and cause false alarms, it is simply not sufficient to rely on a single sensor to safeguard a critical area. In this case, it is desired to provide higher degree of coverage in which multiple sensors monitor the same location at the same time in order to obtain high confidence in detection. On the other hand, it is overkill and energy consuming to support the same high degree of coverage for some non-critical area.

Middleware sits between the operating system and the application. On traditional desktop computers and portable computing devices, operating systems are well established, both in terms of functionality and systems. For sensor nodes, however, the identification and implementation of appropriate operating system primitives is still a research issue [6]. In many current projects, applications are executing on the bare hardware without a separate operating system component. Hence, at this early stage of WSN technology it is not clear on which basis future middleware for WSN can typically be built.

## 4. Scope and Functionality:

The main purpose of middleware for sensor networks is to support the development, maintenance, deployment, and execution of sensing-based applications. This includes mechanisms for formulating complex high-level sensing tasks, communicating this task to the WSN, coordination of sensor nodes to split the task and distribute it to the individual sensor nodes, data fusion for merging the sensor readings of the individual sensor nodes into a high-level result, and reporting the result back to the task issuer. Moreover, appropriate abstractions and mechanisms for dealing with the heterogeneity of sensor nodes should be provided. All mechanisms provided by a middleware system should respect the design principles sketched above and the special characteristics of WSN, which mostly boils down to

energy efficiency, robustness, and scalability. The scope of middleware for WSN is not restricted to the sensor network alone, but also covers devices and networks connected to the WSN. Classical mechanisms and infrastructures are typically not well suited for interaction with WSN. One reason for this are the limited resources of a WSN, which may make it necessary to execute resource intensive functions or store large amounts of data in external components. This may result in a close interaction of processes executing in the WSN and a traditional network. One example of such “external” functionality are so-called virtual counterparts, components residing in the Internet which augment real-world objects with information-processing capabilities [9]. Thus, middleware for sensor networks should provide a holistic view on both WSN and traditional networks, which is a challenge for architectural design and implementation. Another unique property of middleware for WSN is imposed by the design principle *application knowledge in nodes*. Traditional middleware is designed to accommodate a wide variety of applications without necessarily needing application knowledge. Middleware for WSN, however, has to provide mechanisms for injecting application knowledge into the infrastructure and the WSN. *Data-centric communication* mandates a communication paradigm which more closely resembles content-based messaging systems than traditional RPC-style communication. Moreover, event-based communication matches the characteristics of WSN much better than traditional request-reply schemes. In general, communication and application specific data processing is much more integrated in WSN middleware than in traditional systems. The design principle *adaptive fidelity algorithms* requires the infrastructure to provide appropriate mechanisms for selecting parameters or whole algorithms which solve a certain problem with the best quality under given resource constraints.

### 5. Programming Abstractions:

A key to the growth of WSN is raising the level of abstraction for programmers. Currently, programmers deal with too many low levels details regarding sensing and node to node communication. For example, they typically deal with sensing data, fusing data and moving data. They deal with particular node to node communication and details. If we raise the level of abstraction to consider aggregate behavior, application functionality and direct support for scaling issues then productivity increases. Current research in programming abstractions for WSN can be categorized into 7 areas: environmental, middleware APIs, database centric, event based, virtual machines, scripts and component-based. As an example, consider an environmental based abstraction called EnviroTrack [3]. Here the programmer deals with

entities found in an application. If the application tracks people and vehicles, then the programmer can define people and vehicle entities and utilize library routines that support low level sensing functions that can detect and classify objects of these types. They can also easily specify the application level processing associated with each type of entity. This allows programmers to deal with application level functionality rather than low level details. Since WSN deal primarily with collecting, analyzing and acting on data, a database view of such systems is popular. In this view, a programmer deals with queries written in an SQL-like format. However, real-world data issues such as probabilistic data, various levels of confidence in data and missing or late data sometimes make the SQL paradigm insufficient. It is likely that no one programming abstraction for WSN will exist. Rather, a number of solutions will emerge, each better for certain domains. Results in this area are critical in order to expand the

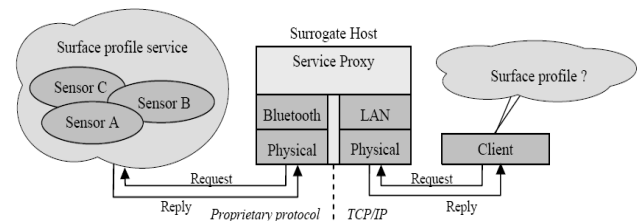


Fig.3 Example of a Surrogate Architecture in Sensor Networks

For the development of application software for sensor networks, the use of a component based framework is desirable. The components of the framework provide the functionality of single sensors, sensor nodes, and the whole sensor network. According to these components, applications are classified into *sensor applications*, *node applications* and *network applications*. A *sensor application* contains the readout of a sensor as well as the local storage of data. It has full access to the hardware and is able to access the operating system directly. The sensor application provides essential basic functions of the local sensor node, which may be used by the node application. The *node application* contains all application specific tasks and functions of the middleware to build up and maintain the network e.g., routing, looking for nodes, discovering services, and self localization. The *sensor network application* describes the main tasks and required services of the entire network without assigning any tasks or services to individual nodes. It represents an interface to the administrator to evaluate the network results. The purpose of our research activities is the development of a framework, which radically simplifies the development of software for sensor-, sensor node-, and sensor network applications. It provides support for distribution,

configuration, scalability and portability. The framework should contain a middleware that allows machine-intimate programming of embedded systems. Thereby, the programming effort is reduced.

## 6. System Considerations:

**A. Current Software Architectures of Small Distributed Devices with Wireless Network Connection** For embedded systems, solutions already exist, which support service architectures and context awareness. In [10] the distributed middleware infrastructure GAIA is presented. It features coordination of software units and heterogeneous networks. The network appears to the environment as a single enclosed device. Because of the use of CORBA, XML, SQL and JAVA, GAIA is not an efficient choice for sensor networks because of its resource requirements. Another approach is Tiny-OS that already provides an advanced framework for sensor networks [3]. It is optimized in terms of memory usage and energy efficiency. Tiny-OS [11] provides mechanisms (events and components) to statically define linking between layers. The predefinition of needed instances at compile time prevents from dynamical memory allocation at runtime. Tiny-OS supports the execution of multiple threads and provides a variety of additional extensions like the virtual machine Maté [2] and the database TinyDB for cooperative data acquisition. Services are currently not provided. Programs for Maté are precompiled and, because of the minimal instruction set, very short. They can be carried in a single Tiny-OS data packet of a maximum length of 24 bytes. Thus, Maté allows a dynamical adaptation of the node application at runtime. The open and platform independent architecture OSGI is suggested in [9] for services in embedded systems. However, OSGI is not appropriate for sensor networks due to its very high resource demands. Therefore, it is mandatory to develop a service architecture that features minimal resource consumption. As a general rule, services have, at least partly, to be executed locally to start communication to the service provider. The execution can take place in native code or in an adapted virtual machine.

**B. Characteristics of a Middleware for Sensor Networks:** The term *middleware* refers to the software layer between operating system and sensor application on the one hand and the distributed application which interacts over the network on the other hand. Primary objective of the middleware layer is to hide the complexity of the network environment by isolating the application from protocol handling, memory management, network functionality and parallelism [13]. A middleware for sensor networks has to be:

- scalable

- generic
- adaptive
- reflective

Resource constraints (memory, processing speed, bandwidth) of available node hardware require an optimization of every node application. Optimization is performed at compile time. Thereby, the application is reduced to all essential components and data types and interfaces are customized (*scalable middleware*). The components of the middleware require a generic interface in order to minimize customization effort for other applications or nodes. The use of identical middleware components in different applications leads to a higher number of complex interfaces. Reducing this overhead is the objective of a *generic middleware*. It is important to customize interfaces to the application in contrast to customize the application to common interfaces. As example, a middleware function SetBaudrate (int transmitter, long baudrate) identifies the network interface with its first parameter. However, a node that has only one interface, does not need this parameter. Consequently, knowledge of this information at compile time can be used for optimization. Another possibility is to change the semantics of data types. A potential use case is the definition of accuracy of addresses that results in a change of data type's width. The width of a data type has vital influence on network traffic. Besides hardware-oriented optimization, an application specific data type optimization exists. The mobility of nodes and changes of infrastructure require adaptations of the middleware at runtime depending on the sensor network application. The middleware must be able to dynamically exchange and run components (*adaptive middleware*). Reflection covers the ability of a system to understand and influence itself. A reflective system is able to present its own behavior. Thereby, two essential mechanisms are distinguished – the inspection and the adaptation of the own behavior [1][4]. Inspection covers ways to analyze behavior e.g., with debugging or logging. Adaptation allows the modification of internal layers to change the behavior presented to the application. In contrast to an adaptive middleware, a *reflective middleware* does not exchange components but changes their behavior. An example of reflective behavior is the modification of the routing strategy depending on mobility. The interface between the software layers remains constant.

### C. Services in Sensor Networks:

Besides the native network functions, such as routing and packet forwarding, future service architectures are required

enabling location and utilization of services. A service is a program which can be accessed about standardized functions over a network. Services allow a cascading without previous knowledge of each other, and thus enable the solution of complex tasks.

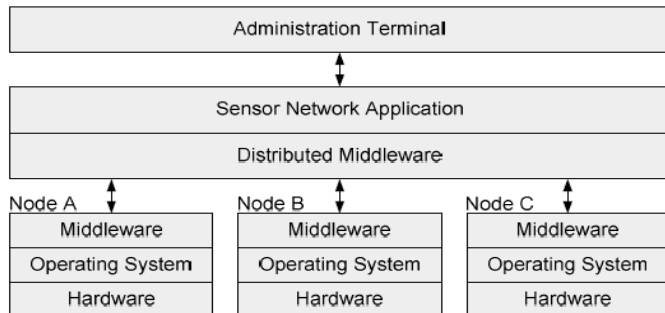


Figure 4. System Diagram for Middleware Sensor Networks (MSN) for (WSN).

A typical service used during the initialization of a node is the localization of a data sink for sensor data. Gateways or neighboring nodes can provide this service. To find this service, nodes use a service discovery protocol. JINI is an emerging technology for desktop applications, but for sensor networks unsuitable due to resource requirements. Sun Microsystems suggests the surrogate host architecture for embedded systems [7], [8]. This is primarily suitable for systems that are controlled by an IP based network. The client can access non-standardized services in a sensor network by inquiring a proxy server. The surrogate host translates the standardized protocol to the proprietary protocol and vice versa. It acts as service provider to the IP based network. Service architectures for sensor networks are part of the sensor application and, in contrast to the eventdriven node application, are based on client server principle. is generally needed to achieve satisfactory transmission reliability in most wireless sensing applications.

2. *Coexistence with Other Networks/Interferers* : As more and more wireless systems are fielded, coexistence becomes a bigger issue. It's not legal to build a system that overrides or interferes with other licensed systems. In short, it should have minimal impact on the performance of other systems. generally speaking, these requirements favor the use of spread-spectrum technology, although conventional

## 7. Algorithms

WSNs are composed of a large number of sensor nodes, therefore, an algorithm for a WSN is implicitly a distributed algorithm. A distributed algorithm is an algorithm designed to run on computer hardware constructed from interconnected processors. Distributed algorithms are used in many varied application areas of distributed computing, such as telecommunications, scientific computing, distributed information processing, and real-time process control. Standard problems solved by distributed algorithms include leader election, consensus, distributed search, spanning tree generation, mutual exclusion, and resource allocation.

Distributed algorithms are typically executed concurrently, with separate parts of the algorithm being run simultaneously on independent processors, and having limited information about what the other parts of the algorithm are doing. One of the major challenges in developing and implementing distributed algorithms is successfully coordinating the behavior of the independent parts of the algorithm in the face of processor failures and unreliable communications links. The choice of an appropriate distributed algorithm to solve a given problem depends on both the characteristics of the problem, and characteristics of the system the algorithm will run on such as the type and probability of processor or link failures, the kind of inter-process communication that can be performed, and the level of timing synchronization between separate processes

In WSNs the scarcest resource is energy, and one of the most energy-expensive operations are data transmission and idle listening. For this reason, algorithmic research in WSN mostly focuses on the study and design of energy aware algorithms for saving energy by reducing the amount of data being transmitted - using techniques like data aggregation -, changing the transmission power of the sensor nodes or turning nodes off while preserving connectivity and coverage - applying Topology control algorithms -.

Another characteristic to take into account is that due to the constrained radio transmission range and the polynomial growth in the energy-cost of radio transmission with respect to the transmission distance, it is very unlikely that every node will reach the base station, so data transmission is usually multi-hop (from node to node, towards the base stations).

The algorithmic approach to WSN differentiates itself from the protocol approach by the fact that the mathematical models used are more abstract, more general, but sometimes less realistic than the models used for protocol design.

## 8. Applications

The applications for WSNs are varied, typically involving some kind of monitoring, tracking, or controlling. Specific applications include habitat monitoring, object tracking, nuclear reactor control, fire detection, and traffic monitoring. In a typical application, a WSN is scattered in a region where it is meant to collect data through its sensor nodes.

**1. Area monitoring:** Area monitoring is a common application of WSNs. In area monitoring, the WSN is deployed over a region where some phenomenon is to be monitored. For example, a large quantity of sensor nodes could be deployed over a battlefield to detect enemy intrusion instead of using landmines. When the sensors detect the event being monitored (heat, pressure, sound, light, electromagnetic field, vibration, etc), the event needs to be reported to one of the base stations, which can take appropriate action (e.g., send a message on the internet or to a satellite). Depending on the exact application, different objective functions will require different data-propagation strategies, depending on things such as need for real-time response, redundancy of the data (which can be tackled via data aggregation and information fusion<sup>[5]</sup> techniques), need for security, etc.

**2. Environmental monitoring :**A number of WSNs have been deployed for environmental monitoring. Many of these have been short lived, often due to the prototype nature of the projects. Examples of longer-lived deployments are monitoring the state of permafrost in the Swiss Alps: The PermaSense Project, PermaSense Online Data Viewer and glacier monitoring.

**3. Water/Wastewater Monitoring:** There are many opportunities for using wireless sensor networks within the water/wastewater industries. Facilities not wired for power or data transmission can be monitored using industrial wireless I/O devices and sensors powered using solar panels or battery packs. As part of the American Recovery and Reinvestment Act (ARRA), funding is available for some water and wastewater projects in most states.

**4. Landfill Ground Well Level Monitoring and Pump Counter:** *Wireless sensor networks can be used to measure and monitor the water levels within all ground wells in the landfill site and monitor leach ate accumulation and removal. A wireless device and submersible pressure transmitter monitors the leach ate level. The sensor information is wirelessly transmitted to a central data logging system to store the level data, perform calculations,*

*or notify personnel when a service vehicle is needed at a specific well.*

It is typical for leach ate removal pumps to be installed with a totalizing counter mounted at the top of the well to monitor the pump cycles and to calculate the total volume of leach ate removed from the well. For most current installations, this counter is read manually. Instead of manually collecting the pump count data, wireless devices can send data from the pumps back to a central control location to save time and eliminate errors. The control system uses this count information to determine when the pump is in operation, to calculate leachate extraction volume, and to schedule maintenance on the pump.

**5. Flare Stack Monitoring:** *Landfill managers need to accurately monitor methane gas production, removal, venting, and burning. Knowledge of both methane flow and temperature at the flare stack can define when methane is released into the environment instead of combusted. To accurately determine methane production levels and flow, a pressure transducer can detect both pressure and vacuum present within the methane production system.*

Thermocouples connected to wireless I/O devices create the wireless sensor network that detects the heat of an active flame, verifying that methane is burning. Logically, if the meter is indicating a methane flow and the temperature at the flare stack is high, then the methane is burning correctly. If the meter indicates methane flow and the temperature is low, methane is releasing into the environment.

**6. Water Tower Level Monitoring :***Water towers are used to add water and create water pressure to small communities or neighborhoods during peak use times to ensure water pressure is available to all users. Maintaining the water levels in these towers is important and requires constant monitoring and control. A wireless sensor network that includes submersible pressure sensors and float switches monitors the water levels in the tower and wirelessly transmits this data back to a control location. When tower water levels fall, pumps to move more water from the reservoir to the tower are turned on.*

**7. Agriculture:** Using wireless sensor networks within the agricultural industry is increasingly common. Gravity fed water systems can be monitored using pressure transmitters

to monitor water tank levels, pumps can be controlled using wireless I/O devices, and water use can be measured and wirelessly transmitted back to a central control center for billing. Irrigation automation enables more efficient water use and reduces waste.

**8. Windrow Composting:** Composting is the aerobic decomposition of biodegradable organic matter to produce compost, a nutrient-rich mulch of organic soil produced using food, wood, manure, and/or other organic material. One of the primary methods of composting involves using windrows.

To ensure efficient and effective composting, the temperatures of the windrows must be measured and logged constantly. With accurate temperature measurements, facility managers can determine the optimum time to turn the windrows for quicker compost production. Manually collecting data is time consuming, cannot be done continually, and may expose the person collecting the data to harmful pathogens. Automatically collecting the data and wirelessly transmitting the data back to a centralized location allows composting temperatures to be continually recorded and logged, improving efficiency, reducing the time needed to complete a composting cycle, and minimizing human exposure and potential risk.

An industrial wireless I/O device mounted on a stake with two thermocouples, each at different depths, can automatically monitor the temperature at two depths within a compost windrow or stack. Temperature sensor readings are wirelessly transmitted back to the gateway or host system for data collection, analysis, and logging. Because the temperatures are measured and recorded continuously, the composting rows can be turned as soon as the temperature reaches the ideal point. Continuously monitoring the temperature may also provide an early warning to potential fire hazards by notifying personnel when temperatures exceed recommended ranges.

**9. Greenhouse Monitoring:** *Wireless sensor networks are also used to control the temperature and humidity levels inside commercial greenhouses. When the temperature and humidity drops below specific levels, the greenhouse manager must be notified via e-mail or cell phone text message, or host systems can trigger misting systems, open vents, turn on fans, or control a wide variety of system responses. Because some wireless sensor networks are easy to install, they are also easy to move as the needs of the application change.*<sup>[7]</sup>

**10. Vehicle Detection:** Wireless sensor networks can use a range of sensors to detect the presence of vehicles ranging from motorcycles to train cars.

## 9. Research Challenges

The severe constraints and demanding deployment environments of wireless sensor networks make computer security for these systems more challenging than for conventional networks. However, several properties of sensor networks may help address the challenge of building secure networks. First, we have the opportunity to architect security solutions into these systems from the outset, since they are still in their early design and research stages. Second, many applications are likely to involve the deployment of sensor networks under a single administrative domain, simplifying the threat model. Third, it may be possible to exploit redundancy, scale, and the physical characteristics of the environment in the solutions. If we build sensor networks so they continue operating even if some fraction of their sensors is compromised, we have an opportunity to use redundant sensors to resist further attack. Ultimately, the unique aspects of sensor networks may allow novel defenses not available in conventional networks. Many other problems also need further research. One is how to secure wireless communication links against eavesdropping, tampering, traffic analysis, and denial of service. Others involve resource constraints. Ongoing directions include asymmetric protocols where most of the computational burden falls on the base station and on public-key cryptosystems efficient on low-end devices. Finally, finding ways to tolerate the lack of physical security, perhaps through redundancy or knowledge about the physical environment, will remain a continuing overall challenge. We are optimistic that much progress will be made on all of them.

## REFERENCES

- [1] Perrig, A., Szewczyk, R., Wen, V., Culler, D., and Tygar, J. SPINS: Security protocols for sensor networks. *J. Wireless Nets.* 8, 5 (Sept. 2002), 521–534.

- [2] Przydatek, B., Song, D., and Perrig, A. SIA: Secure information aggregation in sensor networks. In *Proceedings of the 1st ACM International Conference on Embedded Networked Sensor Systems (SenSys 2003)* (Los Angeles, Nov. 5–7). ACM Press, New York, 2003, 255–265.
- [3] Wood, A., Stankovic, J., and Son, S. JAM: A mapping service for jammed regions in sensor networks. In *Proceedings of the IEEE Real-Time Systems Symposium* (Cancun, Mexico, Dec. 3–5, 2003).
- [4] Wood, A. and Stankovic, J. Denial of service in sensor networks. *IEEE Comput.* (Oct. 2002), 54–62.
- [5] E. Altman, T. Basar, T. Jimenez, and N. Shimkin, “Competitive routing in networks with polynomial costs,” *IEEE Trans. Automat. Control*, vol. 47, no. 1, pp. 92-96, 2002.
- [6] R. Bronson and G. Naadimuthu, *Operations Research*, 2 ed., Schaum’s Outlines, McGraw Hill, New York, 1997.
- [7] N. Bulusu, J. Heidemann, D. Estrin, and T. Tran, “Self-configuring localization systems: design and experimental evaluation,” pp. 1-31, *ACM TECS special Issue on Networked Embedded Computing*, Aug. 2002.
- [8] J. Cao and F. Zhang, “Optimal configuration in hierarchical network routing,” *Proc. Canadian Conf. Elect. and Comp. Eng.*, pp. 249-254, Canada 1999.
- [9] T.-S. Chen, C.-Y. Chang, and J.-P. Sheu, “Efficient path-based multicast in wormhole-routed mesh networks,” *J. Sys. Architecture*, vol. 46, pp. 919-930, 2000.
- [10] J. Choi, C. Conrad, C. Malakowsky, J. Talent, C.S. Yuan, and R.W. Gracy, “Flavones from *Scutellaria baicalensis* Georgi attenuate apoptosis and protein oxidation in neuronal cell lines,” *Biochemica et Biophysica Acta*, 1571: 201-210 (2002).
- [11] C.W. de Silva, *Control Sensors and Actuators*, Prentice-Hall, New Jersey, 1989.
- [12] J. Duato, “A necessary and sufficient condition for deadlock-free routing in cut-through and store-and-forward networks,” *IEEE Trans Parallel and Distrib. Systems*, vol. 7, no. 8, pp. 841-854, Aug. 1996.
- [13] R. Frank, *Understanding Smart Sensors*, 2nd Ed., Artech House, Norwood, MA, 2000.
- [14] M.R. Garey, and D.S. Johnson, *Computers and Intractability: a Guide to the Theory of NP-completeness*. Freeman, San Francisco, CA, 1979.
- [15] F. Giulietti, L. Pollini, and M. Innocenti, “Autonomous formation flight,” *IEEE Control Systems Mag.*, pp. 34-44, Dec. 2000
- [16] Hu, Y.-C., Perrig, A., and Johnson, D. Packet leashes: A defense against wormhole attacks in wireless ad hoc networks. In *Proceedings of IEEE Infocom 2003* (San Francisco, Apr. 1–3, 2003).
- [17] Karlof, C. and Wagner, D. Secure routing in wireless sensor networks: Attacks and countermeasures. In *Proceedings of the 1st IEEE International Workshop on Sensor Network Protocols and Applications* (Anchorage, AK, May 11, 2003).