Wireless Sensor Networks: Software Architecture

Muhammad Adeel Javaid
Member Vendor Advisory Council, CompTIA

Abstract

Due to recent advances, Wireless Sensor Networks are moving out of the laboratory and into the field. For a number of reasons, there is likely to be a significant need to be able to remotely update the software in sensor nodes after deployment. The need for remote software updates is driven both by the scale of deployment, and the potential inaccessibility of sensor nodes. This need has been recognized since the early days of sensor networks, and research results from the related areas of mobile networking and distributed systems been applied to this area. However, there still remain important questions to be answered. In this paper we present an analytical view on WSN architecture design issues, its objectives and implementation challenges.

Keywords- Wireless sensor network, Architecture, power unit, WSN design challenges.

I. 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 [17]. 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).

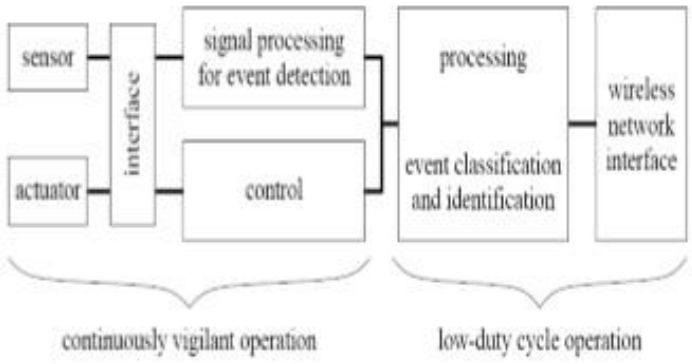

Figure1.System Diagram for Wireless Sensor Networks (WSN)

II. 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.

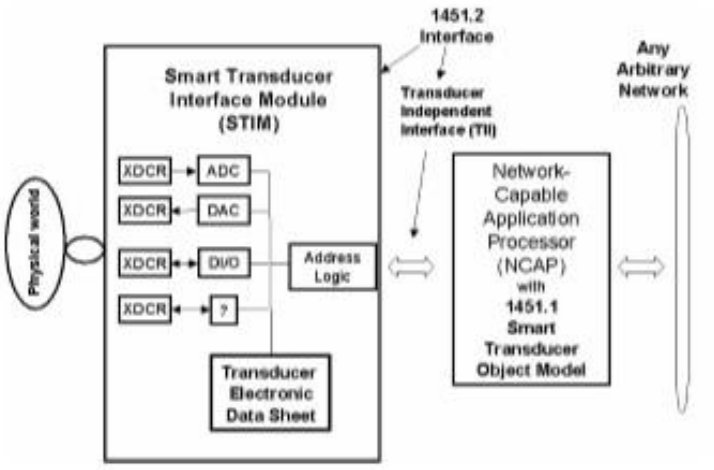

Figure 2. The IEEE 1451 Standard for Smart Sensor Network

III. SOFTWARE ARCHITECTURE COMPONENTS

Figure 3 shows an example of a simple service architecture applicable to a sensor network. In this special case, the client wants to acquire information about the surface conditions in the area of interest. First, the client requests the surrogate proxy via standardized protocols for the surface profile of a part of the observed area. The proxy communicates with the distributed nodes using a proprietary protocol. The nodes located in the target area try to determine the surface profile using cooperative algorithms and send it to the proxy. The proxy translates the information into standardized protocols and sends them back to the client.

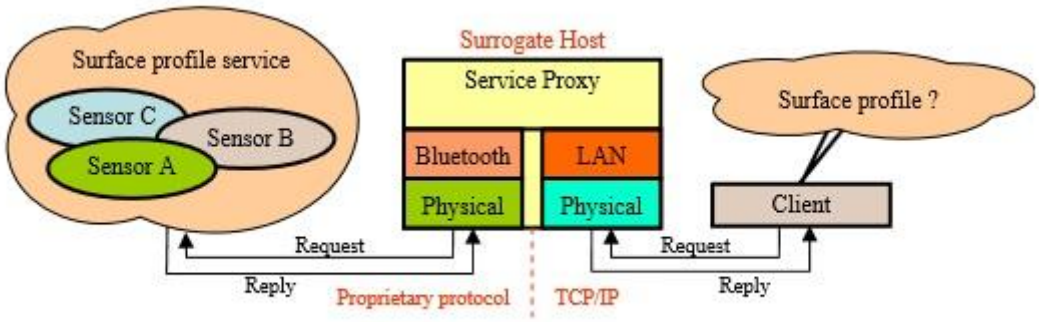

Figure-3: Example of Surrogate Architecture in Sensor Networks

The different requirements and objectives for sensor networks can be achieved only by using a flexible architecture of the node software. Therefore, a node software is divided into three parts according to the main tasks (Figure 4). The Operating System handles the device-specific tasks. This contains bootup, initialization of the hardware, scheduling, and memory management as well as the process management. The OS consists of special tailored parts only needed by the

specific application of the node. The second part is the Sensor Driver. It initalizes the sensor hardware and performs the measurements in the sensor. It encapsulates sensor hardware and provides an optimized Application Programming Interface (API) to the middleware.

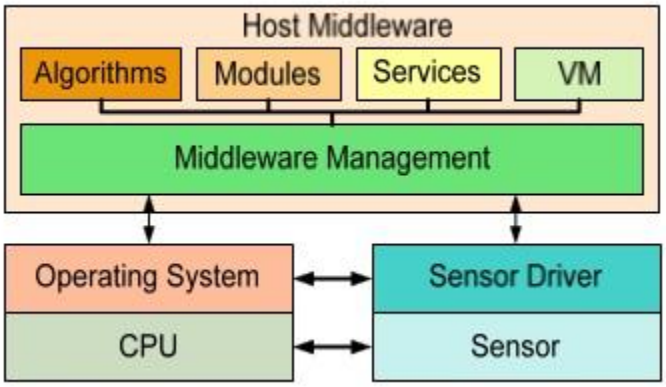

Figure 4: Structure of a Node Application

The Host Middleware is the superior software layer. Its main task is to organize the co- operation of the distributed nodes in the network. The Middleware Management handles four optional components, which can be implemented and exchanged according to the node's task. Modules are additional components that increase the functionality of the middleware. Typical modules are routing modules or security modules. Algorithms describe the behavior of modules. For example, the behavior of a security module can vary if the encryption algorithm changes. The services component contains the required software to perform local and cooperative services. This component usually cooperates with other nodes to fulfill its task. Virtual Machines (VM) enable an execution of platform independent programs.

The software components in a node can be linked together statically or dynamically. Statical linking facilitates an optimization of interfaces between several components within a node. This optimization is called software scaling. It performs in faster and smaller programs. The dynamic link process is used for components exchanged during runtime, e.g. algorithms downloaded from other nodes. This procedure results in system-wide interfaces with significant overhead.

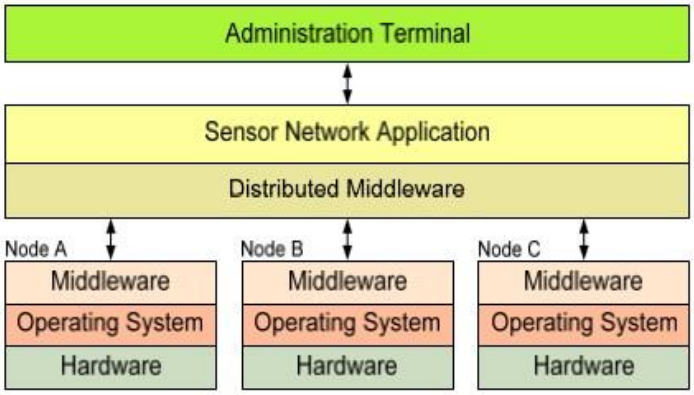

Figure 5: Structure of a Sensor Network

Figure 5 shows the logical view on a sensor network application. The nodes can be contacted only through services of the middleware layers. They do not perform any individual tasks. The Distributed Middleware coordinates the cooperation of the services within the network. It is logically located in the network layer but it exists physically in the nodes. All layers together in conjuction with their configuration compose the sensor network application. The Administration-Terminal is an external entity to configure the network and evaluate the results. It can be connected to the network at any location.

IV. CHARACTERISTICS OF A MIDDLEWARE FOR SENSOR NETWORKS

The term middleware refers to the software layer between operating system and sensor application (As shown in Figure 4 and 5 above) on the one hand and the distributed application which interacts over the network on the other hand. The 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 [14]. A middleware for sensor networks has to be:

- scalable
- generic
- adaptive
- reflective

The resource constraints (memory, processing speed, bandwidth) of available node hardware requires the optimization of every node application. The optimization is performed at compile time. Thereby, the application is reduced to all essential components and datatypes as well as interfaces are customized (scalable middleware).

The components of the middleware require a generic interface in order to minimize the 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 the 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 the first parameter. However, a node that has only one interface, does not need this parameter. Consequently, the 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 the network traffic. Besides hardware-oriented optimization, an application specific data type optimization exists.

The mobility of nodes and changes in the infrastructure requires 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 behaviour. Thereby, two essential mechanisms are distinguished – the inspection and the adaptation of the own behaviour [1][4]. Inspection covers ways to analyze the behavior, e.g., with debugging or logging. The adaptation allows modifying the internal layers to change the behaviour presented to the application. In contrast to an adaptive middleware, a reflective middleware does not exchange components but changes the behaviour of some components. An example of reflective behavior is the modification of the routing strategy depending on mobility. The interface between the software layers remains constant.

V. 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 enables the solution of complex tasks.

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, the node uses 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 [9]. 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 protcol and vice versa. It acts as the service provider to the IP based network. Service architectures for sensor networks are part of the sensor application and operate in contrast to the event-driven node application on the client-server principle.

VI. CONCLUSION

Based on the requirements of sensor networks, this article describes aspects of software engineering. The main objective is the simplification of development of service applications in sensor networks. A key issue is to separate the software from the underlying hardware. The presented service-oriented software concept facilitates the programming on high abstraction layers. Our current research activities concentrate on the realization of the proposed architecture embedded in a framework. It simplifies the development of sensor-, node-, and sensor network applications. Besides that, it provides functionalities to configure and manage the whole network, whereby the scalability and portability of applications increases.

VII. 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).