Introduction to Wireless Sensor Networks

Zill-E-Huma Kamal and Mohammad Ali Salahuddin

Abstract Wireless Sensor Networks (WSNs) enjoy great benefits due to their low-cost, small-scale factor, smart sensor nodes. Not only can they be employed in cumbersome and dangerous areas of interest, for monitoring or controlling the region, but they can also be deployed to automate mundane tasks. Early sensory units were expensive and lacked the computational and communicational capabilities of current smart sensor nodes, which can now sense, process, store, and forward data, all being powered by a battery.

Myriad applications exist that leverage WSNs as low-cost solutions for observing the habitat and environment, from military and civilian surveillance and target detection and tracking applications, to precision farming and agriculture, patient monitoring in health care, residential applications like energy management, for safety and efficiency in vehicular networks to outer space explorations.

The diversity of the applications of WSNs imposes varying design, implementation, and performance requirements on the WSNs. Therefore, for a thorough understanding of the different design and implementation techniques, we must understand the inherent characteristics of WSNs and their smart sensor nodes. This intrinsic nature of the application-specific WSNs makes classification and taxonomy delineation difficult and cumbersome.

In this chapter, we will delineate the inherent characteristics of the WSNs and their smart sensor nodes. Then, we will discuss the data delivery models and traffic patterns that instigate the design and development of novel network architecture protocols for WSN and distinguish them from its peers in other infrastructure-less computing paradigms.

We compare WSN with its peers, with respect to the problem space of WSN applications, followed by a brief overview of the challenges in programming WSN motes. Then, we present an overview of TinyOS, an operating system for WSN motes, and conclude with an overview of the challenges and limitations of WSNs.

Z.-E-H. Kamal (✉)
Montreal, Quebec, Canada
e-mail: huma.kamal@gmail.com

M.A. Salahuddin
Université du Québec à Montréal, Montreal, Quebec, Canada
e-mail: mohammad.salahuddin@ieee.org

D. Benhaddou, A. Al-Fuqaha (eds.), *Wireless Sensor and Mobile Ad-Hoc Networks*,
DOI 10.1007/978-1-4939-2468-4_1

1 Introduction to Wireless Sensor Networks

Originally, sensors were electromechanical detectors for measuring physical quantities. Their first use can be traced back to 1933, in the first room thermostats [1]. Early microelectromechanical systems (MEMS) consisted of a multi-chip, where a sensor and its electronics and mechanics were housed on separate chips and packages. This resulted in larger size, more cost, and lower yield of the sensor [2]. Recent advances in MEMS and integrated circuits (IC) have enabled the development of small-scale sensors and the integration of its actuators and electronics into one cost-effective high-performance chip. Over the past decade, these sensors have evolved into smart sensors, which now include an on-board processor, memory, and transceiver, all in a small-scale factor, powered by a battery source. These smart sensors constitute a node in the Wireless Sensor Network (WSN).

The benefit of the small-scale node is twofold, first, the low production cost and second, the easy and low installation cost. Currently, the price of a WSN node ranges in hundreds to thousands of dollars; however, it is envisioned to reduce to a couple of dollars with advances in technology and mass production [3]. The small size and low cost of the node allows an ease in the installation process, where nodes can be randomly air-dropped or precisely placed, based on the application.

These low-cost nodes with processing, storage, and sensing capabilities, coupled with the attractive infrastructure-less networking capabilities of the transceiver, market the WSN as a powerful, low-cost solution for numerous problems in diverse areas of research. They offer collaboration in a distributed environment for sensing and processing information. Vital information is routed through a multi-hop ad hoc network to a WSN sink, a collector of data or to a base station, which acts as a gateway to a fixed infrastructure. Variations of WSNs include nodes with actuators that comprise a Wireless Sensor and Actuator Network (WSAN) [4, 5], mobile nodes in WSNs [3], or nodes capable of handling multimedia content like audio and video streaming or still images in a Wireless Multimedia Sensor Network (WMSN) [6].

Myriad applications exist that leverage WSNs as low-cost solutions for observing the environment, for example, in military and civilian surveillance and target detection and tracking applications (e.g., [7–10]), for monitoring and controlling industrial processes [11], for precision farming and agriculture [12], environment and habitat monitoring [13, 14], patient monitoring in health care [15, 16], residential applications like energy management [17], in vehicular networks for safety [18] and efficiency [19] and even for outer space explorations [20].

In the following sections, we will discuss the various characteristics of WSN nodes and the intrinsic nature of WSN, followed by a discussion of the traffic patterns and network architecture protocols for WSN and their taxonomy. We will proceed with an overview of TinyOS, an operating system for WSN, and delineate a sample program to illustrate the methodology of programming in WSN. We will conclude with a summary of the challenges and limitations of WSN.

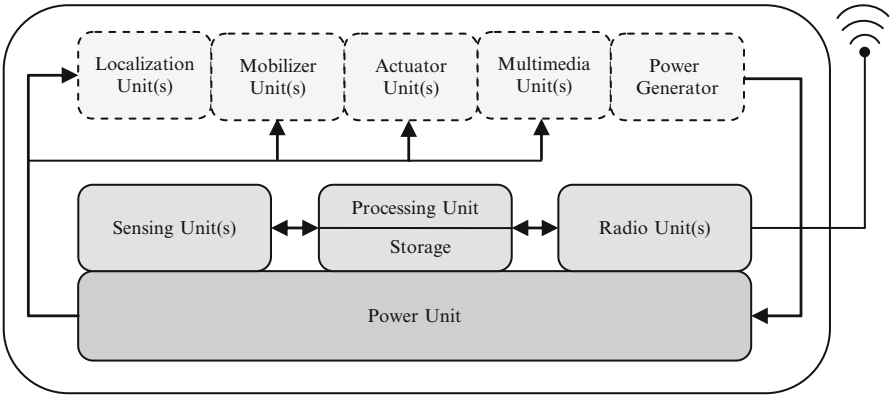

Fig. 1 A typical WSN mote with its fundamental units

1.1 WSN Nodes and Their Characteristics

Technological advances in MEMS and IC instigate the widespread availability of low-cost, small-scale sensors, which evolved into smart, battery powered sensors, with processing and communicating capabilities. These smart sensors constitute the WSN node, referred to as motes, hereafter, in honor of the first WSN nodes, which were University of Berkley's Rene and Mica Motes [21]. Figure 1 illustrates a typical WSN mote with its fundamental units. Minimally, a mote consists of a battery-operated unit with single or multiple sensors, a processing unit with storage and a transceiver. Typically, expansion slots exist or can be attached to expand the mote to include other application-specific units, such as global positioning system (GPS) for localization, or power harvesting units from solar or wind energy, or complementary metal–oxide semiconductor (CMOS) chips for multimedia capabilities.

These motes are placed on programming boards, interfaced with a computer, so that the WSN application and its operating system can be flushed into the memory of the mote. At this time, the motes can also be programmed with a specific identification number and/or group identification number. Various motes can also be programmed without a physical connection to the computer, known as over-the-air (OTA) programming.

WSN motes can vary greatly, with respect to the size, their cost, processing power, communication range, protocols, and operating systems. WSN motes can be as large as a shoebox, e.g. Sensoria Wireless Integrated Network Sensors (WINS) Next Generation (NG) 2.0 [22], or as miniature as a coin, like Moteiv Corporation's Tmote Mini [23], but typical WSN mote dimensions are in the order of a couple of centimeters [24].

WSN motes are typically equipped with multiple sensors for sensitivity to various environmental factors, e.g. mechanical, thermal, biological, chemical, optical,

and magnetic. Often motes have expansion slots that enable them to be equipped with mechanical actuators, wheels for mobility or CMOS chips or microphone for multimedia capabilities. The processors used in these motes can range from ultra-low-power 8 bit processors to more powerful 32 bit processors, similarly, memory space can vary from a couple of kilobytes to the order of megabytes [25].

Motes are equipped with short-range radio frequency (RF) transceivers to enable WSN applications [25] and to ease the query and retrieval of data from the WSN [26]. The use of a short-range radio directly influences the antenna size, since short-range radio waves have higher frequencies and shorter wavelengths they can be received and transmitted by small compact antennas. Through the high frequencies of 800–1,000 MHz, IEEE 802.15.4 or the 2.4 GHz Bluetooth, the low-power radios offer varying bandwidths [25]. The low-transmission range of the motes instigates a multi-hop WSN. Though current WSN motes are equipped with RF radios, communication via infrared, ultrasound, and inductive fields [27] has also been explored [25].

WSN motes can also be equipped with power generation units that harvest power ambient energy sources such as solar [28], mechanical, and thermal [29]. Table 1 presents a comparison of some WSN motes used in academia and industry w.r.t. these characteristics [25, 24, 30].

We will now consider the characteristics of WSNs with respect to the various application and environmental aspects.

1.2 Characteristics of WSNs

WSNs are deployed in a region of interest over a period of time. Since the motes have a short-range radio and a small coverage area, WSNs typically contain a large number of motes. These motes form multi-hop networks and collaborate with each other to maintain connectivity and coverage. Apart from the traditional concerns on collaboration and communication in an ad hoc environment, WSN are also plagued with power and energy management concerns, due to the battery-operated motes. In this section, we will delineate the various design criteria of application-specific WSNs.

Region of Interest The region of interest in WSN applications can be divided into those that are dangerous or isolated versus those that are mundane and cumbersome. In both scenarios, there is little or no infrastructure [3, 26]. For example, volcano monitoring [13], a dangerous task for humans, can be accomplished using low-cost motes that are deployed in the region. These motes are not only expendable but also provide critical data analysis and lifesaving information. However, a WSN used in life rhythm analysis for the elderly [31] can automate the mundane process of gathering vital signs and provides continuous statistics, rather than at fixed intervals.

Modes of Deployment There are two distinct mote deployment strategies, random and precise. In random deployment, motes are randomly distributed like nodes

Table 1 Comparison of some WSN motes

Platform	CPU	Clock (MHz)	RAM/Flash/EEPROM	Radio transceiver	BW (kbps)	Freq. (MHz)	OS
AWAIRS 1	Intel StrongARM SA1100	59–206	1M/4M	Conexant RDSSS9M	100	900	MicroC/OS
μAMPS	Intel StrongARM SA1100	59–206	1M/4M	National LMX3162	1,000	2,400	μOS
Dot	Atmel Atmega 163	8	1K/16K/32K	RFM TR1000	10	916.5	TinyOS
BT Node	Atmel Atmega 128L	8	4K/128K/4K	ZV4002 BT/CC1000	1,000	2,400	TinyOS
Smart-its	PIC 18F252	8	3K/48K/64K	Radiometrix	64	433	Smart-its
Mica2	Atmel Atmega 128L	8	4K/128K/512K	Chipcon CC1000	38.4	900	TinyOS
Mica2Dot	Atmel Atmega 128L	4	4K/128K/512K	Chipcon CC1000	38.4	900	TinyOS
iBadge	Atmel Atmega 128L	8	4K/128K	Ericsson ROK101007 BT	1,000	2,400	Palos
CENS Medusa MK2	Atmel Atmega 128L/Atmel AT91FR4081	4/40	4K/32K/136K/1M	RFM TR1000	10	916	Palos
iMote	Zeevo ZV4002 (ARM)	12–48	64K/512K	Zeevo BT	720	2,400	TinyOS
U3	PIC 18F452	0.031–8	1K/32K/256	CDC-TR-02B	100	315	Pavenet
RFRAIN	Chipcon CC1010 (8051)	3–24	2K/32K	Chipcon CC1010	76.8	0.3–1,000	RFRAIN Libraries
Nymph	Atmel Atmega 128L	4	4K/128K/512K	Chipcon CC1000	38.4	900	Mantis
Telos	TI MSP430F149	8	2K/60K/512K	Chipcon CC2420	250	2,400	TinyOS
MicaZ	Atmel Atmega 128L	8	4K/128K	Chipcon CC2420	250	2,400	TinyOS

Table 1 (continued)

Platform	CPU	Clock (MHz)	RAM/Flash/EEPROM	Radio transceiver	BW (kbps)	Freq. (MHz)	OS
Particle2/29	PIC 18F6720	20	4K/128K/512K	RFM TR1001	125	868.35	Smart-its
eyesIFXv2	TI MSP430F1611	8	10K/48K	Infineon TDA 5250	64	868	TinyOS
iMote2	Intel PXA 271	13–104	256K/32M	Chipcon CC2420	250	2,400	TinyOS
TelosB/Tmote Sky	TI MSP430F1611	8	10K/48K/1M	Chipcon CC2420	250	2,400	TinyOS
Ember RF Module	Atmel Atmega 128L	8	4K/128K	Ember 250	250	2,400	EmberNet
XYZ Sensor node	OKI ML67Q500x (ARM/THUMB)	1.8–57.6	4K/256K/512K	Chipcon CC2420	250	2,400	SOS
Ant	TI MSP430F1232	8	256/8K	Nordic nRF24AP1	1,000	2,400	Ant
ProSpeckz II	Cypress CY8C2764	12	256/16K	Chipcon CC2420	250	2,400	Speckle net
Fleck	Atmel Atmega 128L	8	4K/128K/512K	Nordic nRF903	76.8	902–928	TinyOS
Sun Spot	Atmel AT91FR40162S	75	256K/2M	Chipcon CC2420	250	2,400	Squawk VM (Java)
ECO	nRF24E1 (8051)	16	4K/512/32K	Nordic nRF24E1	1,000	2,400	–
SHIMMER	TI MSP430F1611	4/8	10K/2G	WML-C46A BT/CC2420	250	2,400	TinyOS
IRIS	Atmel Atmega 1281	8	8K/640K/4K	Atmel ATRF230	250	2,400	TinyOS
WaspMote	Atmel Atmega 1281	8	8K/128K/4K	eUnistone 31308/2	38.4	2,400	–
				Zig Bee		2,400/868/900	
				WiFi—802.11 b/g		2,400	
				GSM—SIM900 (SIMCom)		850/900/1800/1,900	
SquidBee	ATmega8/ATmega168	16	1K/8 or 16K/512K	MaxStream	250	2,400	–

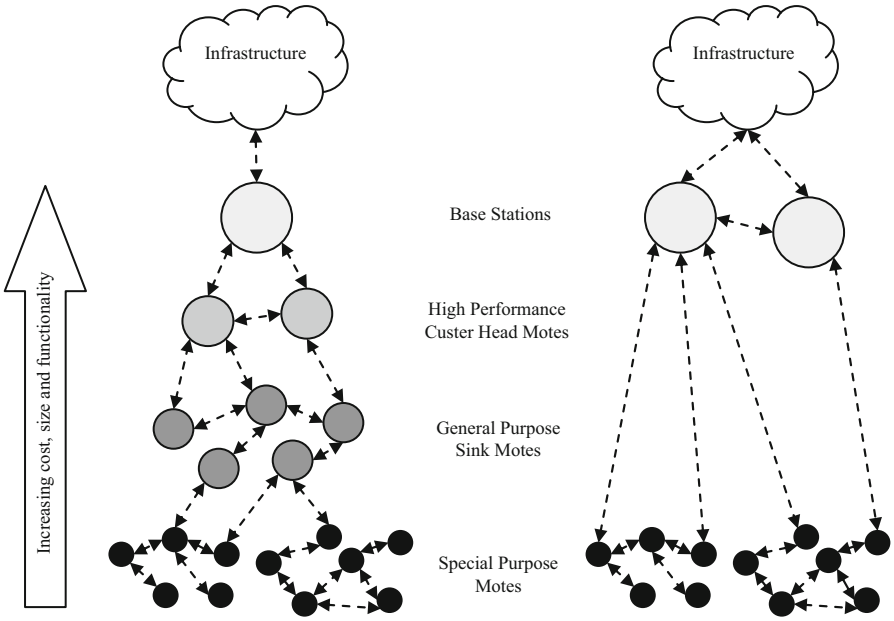

Fig. 2 WSN architecture, hierarchical (*left*) and flat (*right*)

in wireless ad hoc networks. This deployment could be rocket ejected or aircraft sown [32], e.g. in the aerial spread of motes for volcano monitoring [13]. Precise deployment usually consists of manual or pre-planned placement of motes, e.g. habitat monitoring on Skomer Island [33]. The deployment strategies used affect the structure of WSN and can have an impact on coverage of the region and cost of deployment [26].

Organization and Architecture WSN can be organized into two typical structures, flat or hierarchical. Figure 2 illustrates the flat and hierarchical WSN. In a flat network, all motes in the network have the same role and importance, whereas, in a hierarchical organization, motes are clustered or organized into groups with different motes playing different roles, such as general purpose sensing motes or data aggregators or forwarders. As we climb up the hierarchy, the mote's functionality increases with respect to its cost, size, processing power, storage size, etc.

For best results, WSN deployment should consist of a hierarchical organization of heterogeneous motes, integrating a larger number of low-power general purpose motes with smaller number of specialized or high-performance motes [24]. Furthermore, these motes can be equipped with actuators [4] or consist of mobile motes [3].

In WSN architectures, single or multiple base stations preside over the entire WSN. These base stations act as a gateway between the WSN and other fixed infrastructure, e.g. Internet. The base stations are typically, high-performance, human

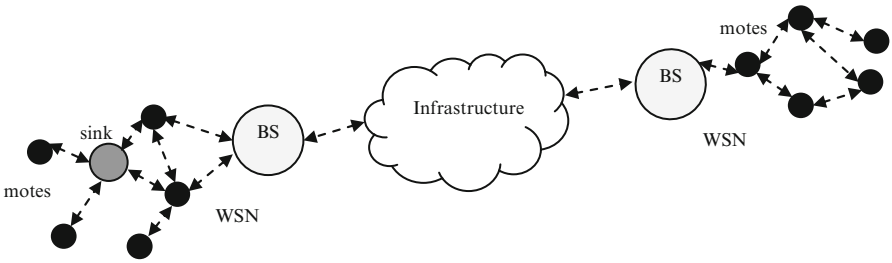

Fig. 3 Global WSN infrastructure

operated devices with rechargeable power source, e.g. laptops interfaced with a mote. Lower down the hierarchy would be specialized motes with slightly better configurations than the low-cost, low-power general purpose motes at the bottom of the hierarchy or pyramid. The special purpose motes offer data aggregation, or fusion, and more complex computational and communicational power, making them ideal sink nodes in the WSN. The general purpose motes at the bottom of the hierarchy typically act as data collectors and forwarders, as illustrated in Fig. 2.

Multiple WSN can be interconnected to form a global WSN, referred to as global sensing infrastructure [34], as illustrated in Fig. 3.

Lifetime of the WSN WSN are deployed with the intention of providing long-term data collection at previously unimaginable scales and resolutions [35]. Typical WSN habitat monitoring applications benefit from long-term data that help decipher data trends and are necessary to detect significant change in habitat [36]. Thus, the lifetime of a WSN is a fundamental characteristic. It is bounded by the finite power source of the battery-operated motes and the nature of the deployed region, and infeasible or cumbersome task of replacing and disposing these batteries. This necessitates power management in hardware and software components of the motes [37, 3] and in WSN protocols [38]. Reinforcement strategies are also being explored in WSN through power harvesting techniques, from solar [28], mechanical, and thermal [29] sources.

Ad Hoc Communication The lack of infrastructure in WSN is an intrinsic property of other communication systems like Bluetooth and mobile ad hoc networks (MANETs) [3]. Bluetooth employs discovery protocols and MANETs use robust and dynamic configuration protocols to establish and maintain the network infrastructure. Though these systems are the closest peers of WSN, there are fundamental differences that distinguish WSN from its peers [3]. First, WSN has larger number of motes with smaller transmission power and radio range, whereas Bluetooth and MANETs have smaller number of nodes with larger transmission power and longer radio ranges. Second, WSN and MANETs suffer from dynamic topology changes due to mote/node mobility and drop-out, but rate of mobility and drop-out is slower in WSN. Lastly, WSN motes have a finite, non-rechargeable power source, whereas Bluetooth and MANET nodes can be recharged. The critical power management in WSN imposes challenging requirements in WSN protocols and inhibits the direct

use of existing Bluetooth and MANET protocols in WSN [3]. Therefore, WSN needs power-aware protocols for a large-scale, autonomous, and heterogeneous network of low-processing power and low-transmission rate motes. These protocols are needed for establishing and maintaining a secure network infrastructure through self-discovery, configuration, and healing, routing and inherent fault-tolerance.

Self-configuration and Organization Various types of self-discovery algorithms exist for use in WSN, some are adaptations from Bluetooth discovery mechanism and others developed specially for WSN. Algorithms delineated in [39–41] and their like consist of a suite of protocols responsible for discovering motes and organizing the wireless infrastructure, despite topology changes and mote failures.

Connectivity and Coverage Establishing and maintaining connectivity and coverage is an essential step in the deployment of a WSN. It is important to determine the initial number of motes required to maintain connectivity and coverage in the region of interest, in face of mote failure and drop-out. It is generally assumed that mote communication and sensing range are uniform and unidirectional, usually depicted as circular regions (cf. Sect. 2.1.1, Fig. 4).

Furthermore, it is imperative for the configuration algorithms to maintain connectivity in the face of dynamic topology changes. Berman et al. [42] propose dominating k-connectivity, such that in the face of up to k mote failures, connectivity to a WSN sink is still maintained. Abbasi et al. [43] present a recovery technique for preserving connectivity in the face of mote failure. Others ([44–47], etc.) have also studied the matter of connectivity and coverage in WSN in great detail.

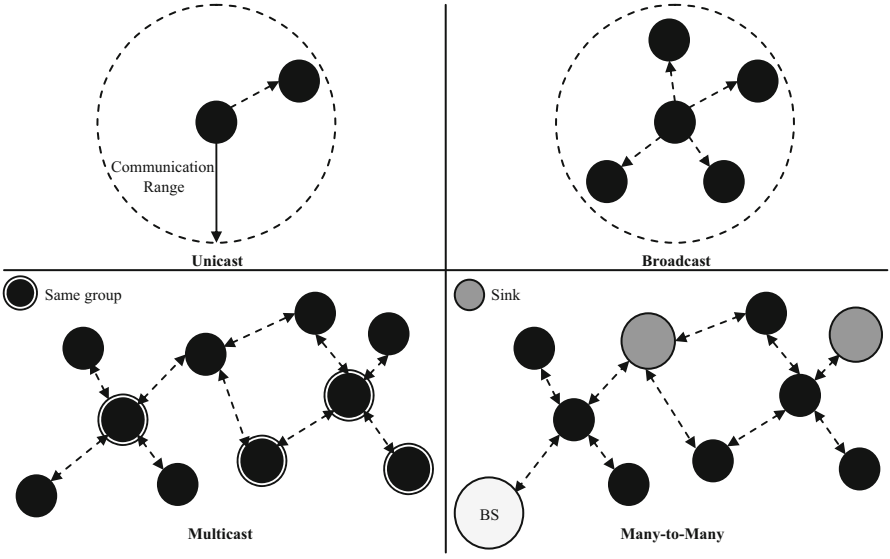

Fig. 4 WSN infrastructure-based data delivery models, unicast, broadcast, multicast, and many-to-many

2 Communication Patterns and Protocols in WSN

After the WSN configuration and organization algorithms have set up the WSN infrastructure for routing and communication. WSN motes can collaboratively perform the application-specific distributed tasks across the WSN. Over the lifetime of the WSN, human operated base stations, which are essentially gateways to fixed infrastructure, e.g. Internet backbone, are used to query and retrieve data from the WSN.

2.1 Communication Patterns

There are two different approaches in sampling and retrieving data from the WSN, namely, the push and pull approach. In a push approach, WSN motes are programmed to autonomously sample the environment at fixed intervals and *push* their data into the network. They push data towards data collectors like the base station or sink. On the other hand, in the pull approach, motes wait for an explicit command from the base station or sink to start sampling [48]. Both approaches have their own advantages and drawbacks. Obviously, push approach suffers from continuous sampling, bottleneck at sink or base station but enjoys low latency query response [48], whereas in the pull approach, there is an increase in the number of messages required for querying and there is a longer delay in query response, but conserves power by explicit sampling rather than continuous sampling [48].

WSN applications typically utilized the push approach to conserve energy, when radio communication was more expensive than sensing on a mote [48]; however, advances in radio technologies have significant improvements in the power consumption, resulting in ultra-low-power radios, e.g. ZigBee radio [48]. Bose and Helal [48] propose a hybrid push–pull approach to sampling and retrieving data from a WSN to leverage the benefits of both techniques. The push, pull, or hybrid approach yields three distinct network traffic patterns in WSN. The push approach causes a many-to-one communication pattern, where motes are pushing data towards the sink or base station. In the pull approach, before motes-to-sink communication occurs, the sink or base station sends the explicit command to start sensing, which is one-to-many communication. When motes are collaboratively communicating to self-configure, localize, or for data fusion at multiple sinks, many-to-many communication occurs [49].

Apart from these traffic patterns in WSN, Tilak et al. [50] decompose communication paradigms in WSN into two categories, that is, application-based and infrastructure-based communication. Application-based communication refers to getting the sensed data from nodes to application user, whereas infrastructure-based protocols are those that are used to provide the underlying primitives to achieve the application-based communication.

We will discuss data delivery models from the application and infrastructure-based communication perspectives, followed by a discussion of the network architecture in WSN.

2.1.1 Data Delivery Models

From the aspect of infrastructure-based communication protocols the flow of data packets is based on four distinct data delivery models: unicast, broadcast, multicast [50], or many-to-many. Unicast is one-to-one communication between two motes, versus, the one-to-many (i.e., all motes in the communication range of a mote) communication of broadcast. The motes within the communication range of a mote are often referred to as one-hop neighbor or simply neighbors. In multi-hop communication, data is routed between motes, using unicast communication.

Multicast pertains to communication of motes in a group. Motes are categorized into groups by the application and only process those packets that contain a group identification number that matches the motes group identification number. These WSN groups can form a connected or disconnected sub-network. In a disconnected group, multi-hop communication would be used to enable multicast data delivery models. If there is a geographic overlay on the multicast WSN and packets are transmitted to motes based on their specific geographical location [51], then this specialized form of multicast data delivery is known as Geocasting. Multicast data delivery model can also be specialized to behave in a many-to-one data delivery manner.

Finally, many-to-many data delivery models result in the presence of multiple sinks or gateways accessing the WSN for data and information [49]. These data delivery models in WSN are illustrated in Fig. 4.

From the perspective of the application-based communication protocols, the data delivery models consist of continuous, event-driven, query-driven, or a hybrid approach of these [50, 52], as illustrated in Fig. 5. In continuous, also known as periodic, data delivery models, data is transmitted from the motes to the sink or gateways at periodic intervals. In event-driven models, the motes are sensitive to one or more physical factors of the environment, when the sensed value (sensor readings) meets or exceeds a predetermined threshold, an event is triggered. The triggered motes propagate their sensor readings back to the sink(s) or gateways. In query-driven data delivery models, user initiates a query and network motes which meet the query criteria transmit their sensor readings. In the hybrid approach, one or more of the data delivery models are used together. For example, in a volcano monitoring application, the motes would be primarily event-driven. However, occasionally the user may want to poll or query the WSN for current seismic or temperature readings.

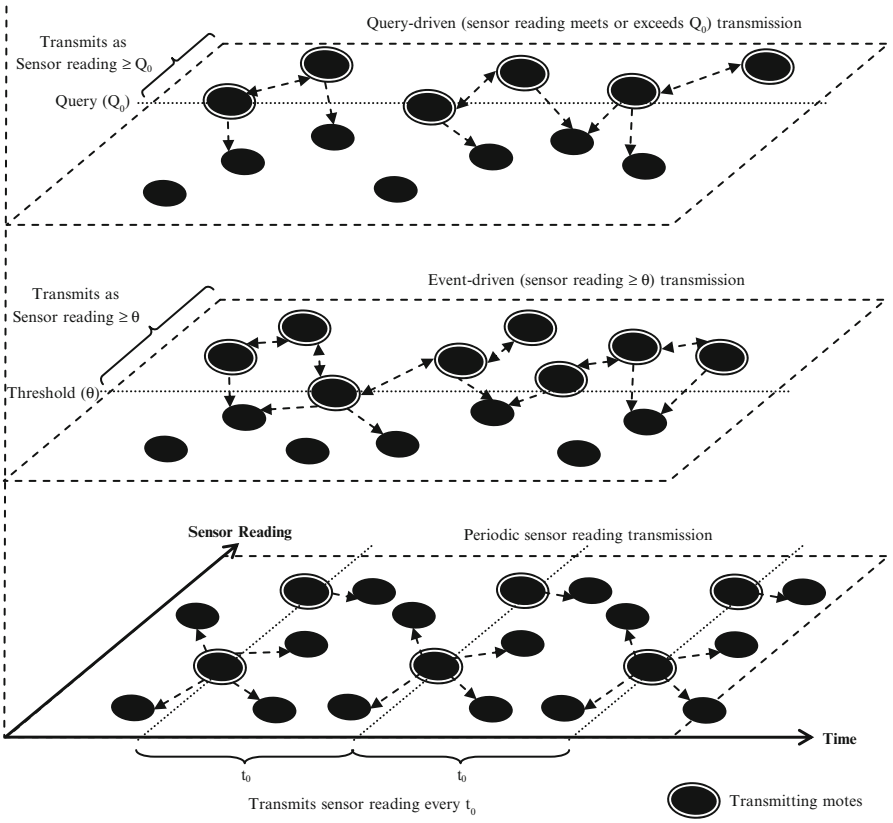

Fig. 5 WSN application-based data delivery models in WSN, periodic, event-driven, and query-driven

2.2 Network Architecture of WSNs

The inherent characteristics of the motes in a WSN, such as limited processing, memory and communication capabilities, impose design and performance criteria on the routing and communication protocols in WSN. Communication protocols are used to dictate, manage, and control all aspects of communication, from the lowest layer of accessing the physical medium to higher layers responsible for end-to-end transmission of packets and routing of data, to the top most application layer for data and packet formation. Figure 6 depicts an overview of the network architecture in WSN.

This is essentially different from typical network models due to the need for power management across all layers of the model. Furthermore, due to the constrained nature of motes and the application-specific nature of WSN, there is no clear demarcation in the network architecture layers, as in traditional network

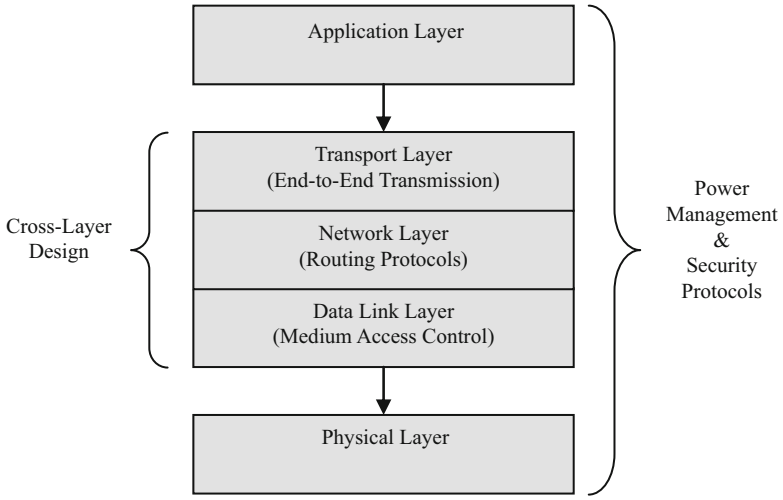

Fig. 6 Network architecture in WSN

architecture. Often, fundamental WSN communication protocols for querying, routing, and data delivery are integrated under data dissemination and aggregation techniques in WSN [53]. These limitations instigate the use of a cross-layer approach in developing network protocols for WSNs.

In the following sections, we will discuss the various concepts of the transport, network, and data link layers with respect to WSNs.

2.2.1 Transport Layer Protocols

Transport layer protocols are responsible for end-to-end transmission of data packets. These protocols include provisions for reliable or unreliable data delivery and for centralized or distributed congestion control. It is imperative that these protocols be designed carefully, since they can quickly overwhelm the constrained WSN. As WSN applications mature, it is important to design and implement efficient protocols for the network architecture to ensure reliability in a data-driven network. Primitives are built into transport layer protocols for reliable data delivery that prompt retransmission of packet(s) in the event of packet drop. This can be achieved by using packet sequence numbers, a series of acknowledgements to confirm delivery of packets, or time-out or round-trip intervals to access packet loss. These primitives allow senders and receivers to account for packets received and packets missed. Such primitives are crucial in WSN applications, which are essentially data-driven networks.

Furthermore, transport layer protocols are also responsible for congestion controlled. This is achieved by installing primitives that control transmission rate of motes, to ensure that the rate at which data packets are being transmitted does not

Table 2 Classification of some transport layer protocols for WSN

WSN transport layer protocols	Reliable	Unreliable	Congestion control		
			Distributed	Centralized	None
Flush	×		×		
STCP	×		×		
Hop	×		×		
RCRT	×			×	
Wisden	×				×
Tenet	×				×
RMST	×				×
WRCP		×	×		
IFRC		×	×		
Fusion		×	×		
CODA		×	×		
QCRA		×		×	
ESRT		×		×	
Surge (TinyOS)		×			×
CTP		×			×
RBC		×			×
CentRoute		×			×
Koala		×			×
XLP	×		×		
CRRT	×			×	
CTCP	×		×		
ERTP	×				×
GARUDA	×				×
DTSN	×				×
PHTCCP		×	×		

overwhelm the underlying network or create a bottleneck at the base station. Table 2 classifies some of the transport layer protocols developed for WSN with respect to these primitives [54–56].

2.2.2 Routing in Network Layer

As in Wireless Computing, routing protocols can be decomposed into reactive, proactive, cooperative [57], or hybrid protocols. Proactive routing protocols are also known as table-driven protocols since each node maintains a route table for all destinations at all times. Reactive protocols are on-demand routing protocols, which only discover routes to a destination when it is required, outdated, or invalid. The obvious trade-offs include data delivery latency, routing protocol traffic overhead, and storage requirements. Data delivery latency is low in networks using proactive routing protocols, since path information is readily available. However, there is

overhead incurred by the motes in the WSN, to discover and maintain paths to all destinations, even those that may never be used in the WSN. Routing protocol traffic overhead is minimized in reactive networks, when there is light network traffic and little changes to the network topology. Moreover, storage requirements of proactive routing protocols increase proportionally to the size of the network. In cooperative routing protocols, data is sent to a central entity that can further process the data and disseminate it accordingly [57].

Routing protocols can also be distinguished based on the communication entities involved in the routing protocol. In this perspective, there is node-centric and data-centric routing. In node-centric routing, data is routed between nodes, based on the node addresses. However, in data-centric routing, attribute based addressing is used, where nodes query for an attribute and only those nodes respond that can satisfy the query. For example, a mote can query for temperature greater than 72 °C, and those motes that have temperature readings more than 72 °C will respond.

Routing protocols are also designed for specific WSN topologies, classified as flat and hierarchical topologies, as illustrated in Fig. 2. On top of these WSN topologies, routing can also be geographical in nature, where packets are routed to motes closest to destination.

WSN routing protocols can also be based on different application defined criteria or operation-based attributes [57]. These operation-based attributes can include multipath, negotiation-based, query-based, QoS parameters (reliability, data integrity, energy efficiency), coherent [57] based routing techniques to increase reliability, security, etc., attributes.

Figure 7 depicts the various characteristics of WSN routing protocols, which can be integrated to design an efficient application-specific routing protocol for a WSN.

Multipath features in routing protocols can have a twofold benefit for WSN. Multipath routing can reduce frequency of updating routes, balance traffic load, and increase rate of data delivery [58]. This consequentially increases the lifetime and reliability of WSN. Earlier researchers proposed to use suboptimal routing paths with low energy consumption [59] or select a routing path based on the residual energy of the motes on the path [60]. The authors in [61] study the trade-off between traffic overhead and reliability using multipath routing in WSN. More recently, [58] uses multipath routing to increase lifetime of WSN and reduce collisions for transmission. Furthermore, researchers are utilizing ant colonization optimization, a swarm intelligence optimization technique, due to their proven success [62] in routing protocols for WSN. In [63], ant colonization is used to discover optimal routes to increase network lifetime and reliability, whereas [62] uses multipath routing to reduce congestion at the mote and link level. Authors in [64] survey swarm intelligence based routing protocols in WSN.

Routing protocols that are built on the pull approach, where queries can be propagated, are known as query-based routing protocols. Routing algorithms such as directed diffusion and rumor routing [57] are specifically implemented for query-driven data delivery models, where motes that meet the query requirements set up interest gradients or paths along which the data is routed to the sink.

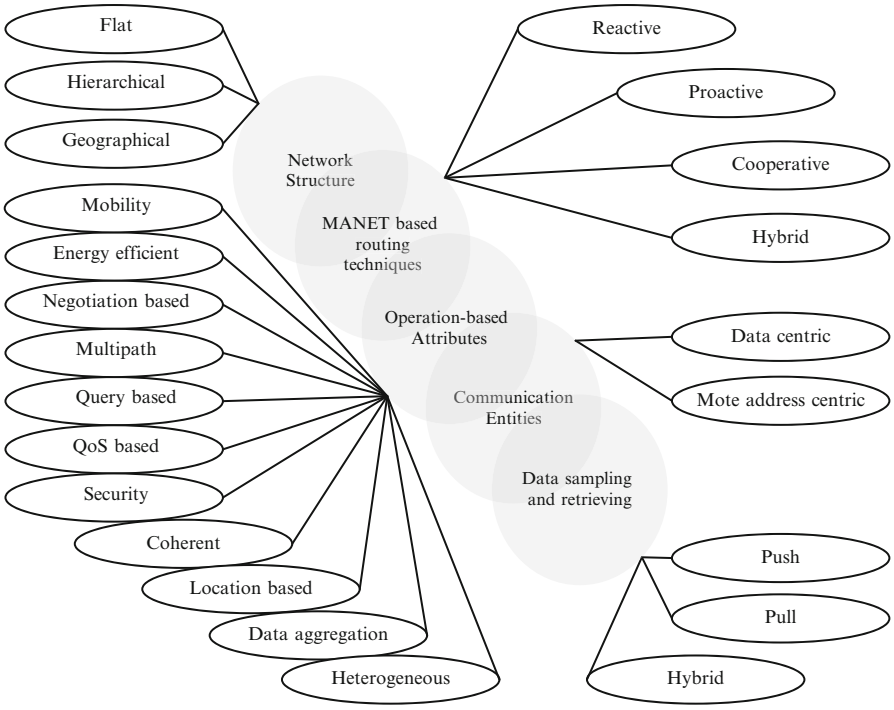

Fig. 7 WSN routing protocol characteristics

Negotiation-based routing algorithms suppress redundant data and use negotiation messages to ensure non-redundant data before transmission, e.g. Sensor Protocols for Information via Negotiation (SPIN) [65]. QoS-based routing ([66–69], etc.) ensures that routing in WSN meets application defined QoS metrics, of delay, energy, bandwidth, reliability, fault-tolerance, data integrity, etc. Data aggregation is also an important aspect of WSN and their applications, and is often accounted for when designing and developing routing protocols, to ensure efficient performance.

Not only is it important to secure the data and the motes in the WSN, but also the network traffic patterns. Since malicious motes can redirect traffic, inject false data or drop packets and cause havoc in the WSN, typical uses of asymmetric key cryptography or complicated symmetric key cryptography are infeasible for resource constraint WSN [70]. While there are techniques for implementing security on the data link layer in WSN, researchers are designing novel techniques for securing routing protocols [71–73], to increase reliability of WSN.

Table 3 classifies some WSN routing protocols based on the routing protocol characteristics illustrated in Fig. 7 [52, 57, 74–76, 71, 77–79].

Table 3 Classification of some WSN routing protocols

WSN routing protocol	Network structure	MANET technique	Operation-based attribute	Communication entities	Data sampling and retrieving
SPIN	Flat	Proactive	Data aggregation, negotiation-based, query-based, multipath	Data-centric	Pull
Directed diffusion	Flat	Reactive	Data aggregation, negotiation-based, query-based, multipath, localization, coherent	Data-centric	Pull
Rumor routing	Flat	Reactive	Query-based	Data-centric	Hybrid
TEEN and APTEEN	Hierarchical	Reactive	Data aggregation, localization, energy efficient	Data-centric	Push
LEACH	Hierarchical	Proactive	Data aggregation, localization	Mote address-centric	Push
ACQUIRE	Flat	Reactive	Data aggregation, query-based	Data-centric	–
COUGAR	Flat	Reactive	Data aggregation, query-based	Data-centric	Hybrid
PEGASIS	Hierarchical	Proactive	Data aggregation, localization, energy efficient	Mote address-centric	Push
GAF	Geographical	Proactive	Energy efficient	Mote address-centric	–
SEAD	Geographical	Proactive	Mobility, energy efficient	Mote address-centric	–
SPEED (SNFG)	Geographical	Proactive	QoS, energy efficient	Mote address-centric	–
CHR	Geographical	Proactive	Data aggregation, heterogeneous	Mote address-centric	–
APR	Geographical	Proactive	Security	Mote address-centric	–

2.2.3 Medium Access Control (MAC) Protocols at Data Link Layer

The MAC protocols at the data link layer manage how motes access the shared wireless medium and the backoff approach they employ in the event of a collision. In typical Wireless Networks, the radio channel is decomposed into multiple channels using techniques like time division multiple access (TDMA), frequency division multiple access (FDMA), or code division multiple access (CDMA) to allow collision-free transmission of multiple message, simultaneously. Traditional WSN motes used a simple single channel radio to conserve energy, which is required to maintain the complex multiple channel radios [80]. However, recent advances in research have given rise to multi-channel radios for WSN motes.

WSN MAC protocols can be broadly classified into schedule based or contention-based medium access protocols. Schedule or reservation based MAC protocols devise an assignment that motes can follow for accessing the physical medium to avoid collision. Contention-based MAC protocols do not require an assignment, they define a backoff algorithm for motes to follow, when there is a collision in accessing the physical medium. Schedule based protocols can be further decomposed into synchronous, asynchronous, and hybrid protocols.

Since the MAC protocol directly controls the radio on a mote, an important technique used by motes is called duty cycling, where active motes periodically turn their radio off and go to sleep [81]. Synchronous schedule based MAC protocols require time synchronization and topology information to ensure that neighboring motes are active at the same time to communicate. However, asynchronous protocols do not require such synchronization or topology information and instigate communication between motes in different active cycles.

Contention-based MAC protocols eliminate the overhead of synchronization or network topology from schedule based MAC protocols. Motes can transmit packets immediately and retransmit in the event of a collision. A refined approach is to sense the physical medium for transmission, and when the medium is idle then access the medium and transmit the packet. This is the general concept in carrier sense multiple access (CSMA), which is one of the canonical contention-based MAC protocols.

Ye et al. [82] have identified four significant aspects of wasteful energy consumption, namely, collision, overhearing, control packet overhead, and idle listening at the data link layer. When two or more motes transmit at the same time, their respective packets are corrupted and require retransmissions increasing energy consumption and latency. Energy consumed in overhearing is energy spent listening for packets destined for other motes. Sending and receiving control packets without useful data is also wasteful with respect to energy consumed. Listening to the channel even when there are no radio transmissions is considered idle listening, which is an energy consumption overhead [81].

Contention-based protocols include techniques like CSMA and offer multiple benefits like low implementation complexity, flexibility in face of mobility, and varying traffic patterns [80]. However, schedule based MAC protocols like TDMA have intrinsic energy conserving features since there are no collisions, overhearing,

Table 4 Classification of some WSN MAC protocols

WSN MAC protocol	Organization	Channel
SMACS	Frames	FDMA
T-MAC	Slots	Single
TRAMA	Frames	Single
BuzzBuzz	Random	Single
DW-MAC	Slots	Single
X-MAC	Random	Single
B-MAC	Random	Single
WiseMAC	Random	Single
PW-MAC	Hybrid	Single
S-MAC	Slots	Single
PicoRadio	Random	CDMA

and idle listening. Due to the energy conservation necessary in resource constraint WSN, it is important for MAC protocols for WSN to incorporate primitives for conserving energy by reducing collisions, overhearing, reduced control packet overhead, and idle listening, without the need for expensive time synchronization.

These MAC protocols require different degrees of collaboration and cooperation amongst the motes to achieve medium control access. Motes in the network could work completely independently in a random manner like those in contention-based MAC protocols, to slightly organized slotted MAC protocols and then to the very organized synchronous schedule based protocols like TDMA [80].

Various MAC protocols have been designed and implemented including synchronous duty cycling MAC protocols such as S-MAC [82], T-MAC, TRAMA, SCP, and DW-MAC. and asynchronous duty cycling MAC protocols such as X-MAC [83], B-MAC, WiseMAC, PW-MAC [81], and ASCEMAC [84]. An exhaustive MAC protocols survey is presented in [85]. MAC protocols are also being designed to include important features like security [86], power management [87], and QoS [88]. Table 4 delineates some of these protocols and their classification [80, 89].

3 WSN Applications and Problem Space

Numerous intrinsic WSN characteristics distinguish it from its peers in other infrastructure-less communication systems [3] and distributed computing environments [90], where their nodes are more powerful (w.r.t. processor and radio) and reliable than WSN motes. Figure 8 [90] illustrates the difference in the problem space of WSN with other computing paradigms, with respect to functionality of the network nodes, size of the network, and spatial-temporal awareness of the nodes in the network.

WSN are distinguishable from real-time embedded systems primarily due to the lack of spatial awareness in embedded system nodes. Real-time embedded

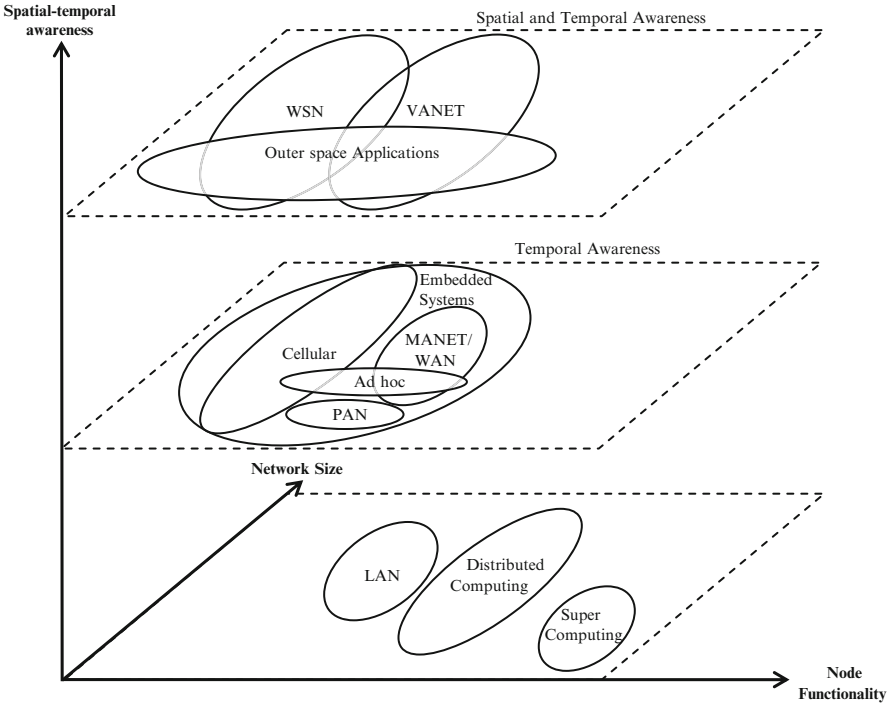

Fig. 8 Comparison of problem space of WSN with other computing paradigms

systems can include various other computing paradigms, especially, infrastructure-less mobile environments like Bluetooth personal area networks (PAN), MANET, cellular networks, and ad hoc networks.

WSN are significantly different from traditional distributed and supercomputing paradigms. Distributed computing includes paradigms such as peer-to-peer (P2P) networks, grid and high-performance computing. Apart from the lack of spatial and temporal awareness of the network nodes, another fundamental difference lies in the functionality of the network nodes. Typically, distributed and supercomputing paradigms consist of high-performance powerful nodes, as illustrated in Fig. 8.

WSN applications are sensitive to spatial and temporal parameters and can be classified into spatially and temporally related paradigms as follows. Firstly, in the spatial domain, WSN applications can be broadly classified into local and global WSN applications based on the size and coverage of the region of interest of the application. Similarly, WSN applications can also be temporally categorized into continuous, event-driven, query-driven, or hybrid applications (cf. Sect. 2.1.1, Fig. 5). This imposes a constraint on when the motes transmit data, either motes transmit data at periodic intervals, or transmit only the data that exceeds application defined triggers or threshold levels, or only those motes transmit data that meets

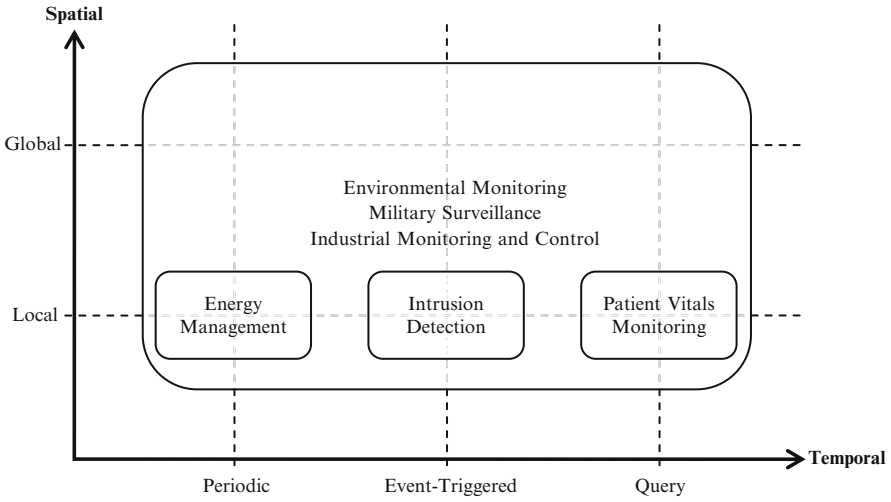

Fig. 9 Classification of WSN applications

a query criterion or a hybrid approach of these techniques. Figure 9 [49] broadly classifies some WSN applications into these domains.

WSN application operations cannot be discretely categorized into the different temporal domains, since these networks are fundamentally data-driven networks. This implies events could be triggered in an otherwise periodic sampling WSN application like environmental monitoring. Furthermore, periodic sampling WSN applications like patient vitals health monitoring could quickly become query-based, if the WSN for patient vitals monitoring is polled for specific vital reading. Therefore, WSN applications are usually designed to operate in multiple temporal domains. Figure 9 illustrates these characteristics of WSN applications and delineates some WSN application examples that accurately fit the domains.

Figure 10 [49] delineates the various design characteristics that form the intrinsic nature and behavior of WSN applications and their motes. This directly influences the design for the components of the network architecture of the WSN. The figure also summarizes the various WSN characteristics we have discussed in this chapter.

4 Programming WSNs

WSNs can only be fully utilized, if WSN programmers have adequate software platforms [49] for efficient and reliable programming of large number of motes. However, due to the lack of programming, debugging, and development environment platforms, WSN programmers are forced to implement application-logic intertwined with low-level WSN implementation issues [49, 90]. This makes the

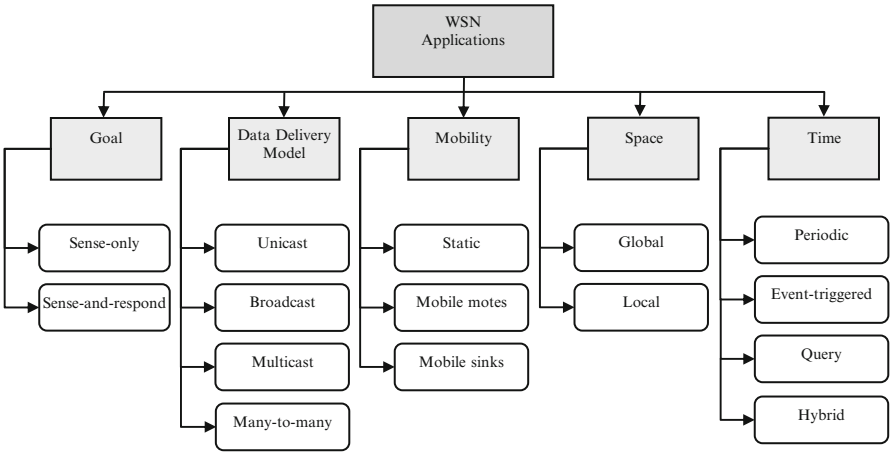

Fig. 10 WSN application design criteria

application code complex, harder to debug and non-scalable for complex application development due to the interdependencies between performance and function [90]. Furthermore, this sort of programming falls out of the expertise of WSN application domain experts [49]. Mottola and Picco [49] discuss various other design criteria and components of WSN programming languages and their affect on the design and development of WSN applications.

Myriad WSN motes exist that differ in functionality as delineated in Table 1. Manufacturers often ship an operating system with the motes to ease in the handling of hardware and software components. Most WSN operating systems (OS) are either event-driven or multithreaded [91]. Multithreaded OS are similar to traditional OS, therefore easier for programmers to manage, however, infeasible for resource constrained WSN. Therefore, lightweight multithreaded OS have been designed and implemented for WSN motes. Event-driven OS do not tax the resources of the WSN motes but pose a learning curve for programmers of WSN applications. Farooq and Kunz [91] survey and classify operating systems such as TinyOS, Contiki, and MANTIS, in WSN.

In the following section, we will discuss TinyOS, an operating system for WSN, and consider a component-based approach to programming applications for WSNs using TinyOS.

4.1 TinyOS: An Operating System for WSN Motes

Operating systems (OS) are programs that coordinate all other programs, such as process scheduling and memory access. in a computer. The typical size of operating systems for embedded devices ranges in the order of megabytes; however, TinyOS

is a much smaller operating system for WSN motes, approximately 400 bytes [92]. It is implemented in nesC [93], a variant of C. TinyOS is different from traditional OS, since it provides a framework and a set of component for programming motes to *build* application-specific operating system for a WSN application. Therefore, there is an operating system for each application. Typical TinyOS applications can range from approximately 15 Kbytes to more complex applications in the range of 64 Kbytes, of these the core operating system is about 400 bytes [92]. The networking architecture is encapsulated in Active Message interfaces, which are also implemented in nesC and offer fundamental communication primitives to TinyOS applications [49, 92]. Surge is the transport layer protocol implemented in TinyOS, which is unreliable and offers no primitives for congestion control [54]. Berkley MAC (B-MAC) is the asynchronous schedule based MAC protocol implemented in TinyOS [94], with interfaces available to change or adapt the MAC protocol.

TinyOS is a component-based programming model, where components are software abstractions of hardware components [92]. Therefore, a TinyOS application, which is really an application-specific operating system, is built on top of these reusable components [92], wiring them together to provide or use services. Components use interfaces to encapsulate a set of services [92]. For example, turning an LED on is a service that can be modeled by using the LED interface. Components use commands and events for inter-component communication. Commands to a component are requests to start a service, whereas events are signaled to mark the completion of the service at the component [92].

Programming in nesC consists of a configuration and module file. The configuration file connects components together, whereas the module file contains the implementation, by *providing* or *using* interfaces. Figures 11 and 12 illustrate a "hello world" TinyOS application, called Blink [95], where the red LEDS on a mote are toggled every 1,000 ms.

The `Blink.nc` configuration file shows how the components are wired together. A TinyOS application starts by executing the `init` command of the `StdControl` interface of the `Main` component. `Blink` configuration file wires the `Main` component's `StdControl` to `SingleTimer` and `Blink` component `StdControl`. This binds the implementation of `Main.StdControl` to `SingleTimer.StdControl` and `BlinkM.StdControl`. The `BlinkM.StdControl` interface is implemented in the module file, `BlinkM.nc`. Similarly, the `Timer` and `Leds` interfaces used in `Blink` actually invoke the

```
configuration Blink {}

implementation {
  components Main, BlinkM, SingleTimer, LedsC;
  Main.StdControl -> SingleTimer.StdControl;
  Main.StdControl -> BlinkM.StdControl;
  BlinkM.Timer -> SingleTimer.Timer;
  BlinkM.Leds -> LedsC;
}
```

Fig. 11 Blink.nc configuration file

```
module BlinkM {
  provides {
    interface StdControl;
  }
  uses {
    interface Timer;
    interface Leds;
  }
}
implementation {

  command result_t StdControl.init() {
    call Leds.init();
    return SUCCESS;
  }

  command result_t StdControl.start() {
    return call Timer.start(TIMER_REPEAT, 1000);
  }

  command result_t StdControl.stop() {
    return call Timer.stop();
  }

  event result_t Timer.fired()
  {
    call Leds.redToggle();
    return SUCCESS;
  }
}
```

Fig. 12 BlinkM.nc module file

`Timer` interface provided by component `SingleTimer`, and `Leds` interface of component `LedsC`, respectively. Note, `SingleTimer` and `Leds` provide software abstraction of the hardware components for the clock and LEDS.

The configuration file delineates how control of execution is passed amongst components. The module file (`BlinkM.nc`) provides the application-specific implementation by providing and using interfaces. `BlinkM.StdControl.init()` initializes the application and `BlinkM.StdControl.start()` starts the component. When `Blink` starts, it sets the timer to repeat every 1,000 ms. In the event-driven operating system, since, there is no other code to execute, `Blink` waits until event `Timer.fired()` is triggered. When the event is triggered, `Blink` toggles the red `LEDS`.

The application is compiled for the target platform, any TinyOS compatible device, e.g. micaz motes, and flushed on the mote hardware using a programming board, which interfaces a mote with the development environment on a computer or laptop.

5 Summary

WSNs typically consist of large number of heterogeneous motes organized in a hierarchical manner, to monitor and/or control a region of interest, based on ambient environment factors. Generally, the number of motes deployed in the region increases and the resources decrease as you move down the hierarchy. The mobile, infrastructure-less WSN are able to truly achieve untethered communications in dangerous or mundane regions for autonomous data collection, analysis, and response via actuators. However, the greatest resource limitation of WSN is the power source, which must be optimally used or alternatively harnessed to ensure network lifetime.

There is a lack of standards for traditional network concepts, like architecture, topology, routing, and security, since WSN are application-specific in nature. Due to the different types of motes and WSNs, there are interoperability issues amongst the heterogeneous hardware units [96]. Further limitations of WSN are attributed to the lack of software for programming and debugging motes and WSN [49, 90].

Energy conservation and power management in the radio and network communications are the underlying design criteria for protocols and primitives in WSN. However, Bose and Helal [48] show that advances in technology have allowed low-power radios, like the Atmel Zlink RCB [48] for effective use in WSN and now attention needs to be on the sensor sampling technique, which consumes significantly more energy than network and application costs. Anastasi et al. [94] present a survey of energy conservation techniques in WSN.

Furthermore, apart from energy conservation, security challenges in WSN are a make it or break it criteria in the widespread use of WSN in all envisioned domains. Boyle and Newe [97] compare some security protocols in use today in WSN, like SPIN [98], consisting of SNEP (Secure Network Encryption Protocol) and μTESLA (micro version of Timed Efficient Stream Loss-tolerant Authentication), LEAP [99], and TinySec [100], which aim at building security features into network protocols.

As recent routing protocols have gained from swarm intelligence, Kulkarni et al. [101] discuss the use of other computational intelligence paradigms like neural networks, fuzzy logic, evolutionary algorithms, reinforcement learning, and artificial immune system in designing and implementing effective and efficient protocols and primitives for WSN operations like routing, deployment, localization, security, scheduling, data aggregation, and QoS management.

To summarize, in this chapter we considered the characteristics and features of WSNs and their motes. Furthermore, we reviewed the fundamental concepts in WSNs pertaining to architecture, topology, data delivery models, transport, routing, and data link layer protocols. We also presented a classification of some of the protocols for transport, routing, and data link layers. We concluded with a discussion of TinyOS, an operating system for WSN motes, and exemplified programming WSN motes in nesC.

References

1. History of Innovation (1995–2012) [Online]. Available: http://www.ti.com/corp/docs/company/history/sensortimelinelowbandwidth.shtml. Accessed 6 Jan 2012
2. Weinberg H (2012) How they work: MEMS (micro electro-mechanical systems) technology. Sensorland [Online]. Available: http://www.sensorland.com/HowPage023.html. Accessed 6 Jan 2012
3. Akyildiz IF, Su W, Sankarasubramainiam Y, Cayirci E (2002) A survey on sensor networks. IEEE Commun Mag 40(8):102–114
4. Akyildiz IF, Kasimoglu IH (2004) Wireless sensor and actor networks: research challenges. Ad Hoc Netw 2(4):351–367
5. Xia F (2008) QoS challenges and opportunities in wireless sensor/actuator networks. Sensors 8(2):1099–1110
6. Akyildiz IF, Melodia T, Chowdury K (2007) A survey on wireless multimedia sensor networks. Comput Netw 51(4):921–960
7. Lamont L, Toulgoat M, Deziel M, Patterson G (2011) Tiered wireless sensor network architecture for military surveillance applications. In: International conference on sensor technologies and applications (SENSORCOMM), Nice, Saint Laurent du Var
8. Antepli MA, Gurbuz SZ, Uysal-Biyikoglu E (2010) Ferromagnetic target detection and localization with a wireless sensor network. In: Military communications conference (MILCOM), San Jose
9. Teng J, Snoussi H, Richard C (2010) Decentralized variational filtering for target tracking in binary sensor networks. IEEE Trans Mob Comput 9(10):1465–1477
10. Medagliani P, Ferrari G, Gay V, Leguay J (2013) Cross-layer design and analysis of WSN-based mobile target detection systems. Elsevier Ad Hoc Netw 11(2):712–732
11. Krishnamurty L, Adler R, Buonadonna P, Chhabra J, Flanigan M, Kushalnagar N, Nachman L, Yarvis M (2005) Design and deployment of industrial sensor networks: experiences from a semiconductor plant and the North Sea. In: 3rd international conference on embedded networked sensor systems (SenSys 05), San Diego
12. Sikka P, Corke P, Valencia P, Crossman C, Swain D, Bishop-Hurley G (2006) Wireless adhoc sensor and actuator networks on the farm. In: International conference on information processing in sensor networks, Nashville
13. Song W-Z, Huang R, Xu M, Ma A, Shirazi B, LaHusen R (2009) Air-dropped sensor network for real-time high-fidelity volcano monitoring. In: International conference on mobile system, applications and services (MobiSys 09), Krakow
14. Ceriotti M, Mottola L, Picco GP, Murphy AL, Corra M, Pozzi M, Zonta D, Zanon P (2009) Monitoring heritage buildings with wireless sensor networks: the Torre Aquila deployment. In: International conference on information processing in sensor networks (IPSN), San Francisco
15. Ko J, Lim JH, Muvaloiu-E R, Terzis A, Masson GM, Gao T, Destler W, Selavo L, Dutton RP (2010) MEDiSN: medical emergency detection in sensor networks. ACM Trans Embed Comput Syst (TECS) 10(1):1–29
16. Chen Y, Shen W, Huo H, Xu Y (2010) A smart gateway for health care system using wireless sensor network. In: International conference on sensor technologies and applications (SENSORCOMM 10), Venice
17. Ota N, Ahrens S, Redfern A, Wright P, Yang X (2006) An application-driven architecture for residential energy management with wireless sensor networks. In: IEEE international conference on mobile adhoc and sensor systems, Vancouver
18. Song H, Zhu S, Cao G (2008) SVATS: a sensor-network-based vehicle anti-theft system. In: IEEE conference on computer communications, Phoenix
19. Tubaishat M, Zhuang P, Qi Q, Shang Y (2009) Wireless sensor networks in intelligent transportation systems. Wirel Commun Mob Comput 9(3):287–302, Special Issue: Distributed Systems of Sensors and Actuators

20. Vladimirova T, Bridges CP, Paul JR, Malik SA, Sweeting MN (2010) Space-based wireless sensor networks: design issues. In: IEEE aerospace conference, Big Sky
21. Culler D, Hill J, Horton M, Pister K, Szewczyk R, Woo A (2002) MICA: the commercialization of microsensor motes. Sensors Magazine, 1 April 2002 [Online]. Available: http://www.sensorsmag.com/networking-communications/mica-the-commercialization-microsensor-motes-1070. Accessed 1 Feb 2012
22. Merrill W, Kaiser W (2004) Wireless integrated network sensors (WINS) next generation. Airforce Research Laboratory, Rome
23. Wireless Applications – Moteiv launches wireless mote with interface to mobile devices. Sensors, 11 May 2007 [Online]. Available: http://www.sensorsmag.com/wireless-applications/news/moteiv-launches-wireless-mote-with-interface-mobile-devices-2523. Accessed 7 Jan 2012
24. Hill J, Horton M, Kling R, Krishnamurthy L (2004) The platforms enabling wireless sensor networks. Commun ACM 47(6):41–46
25. Healy M, Newe T, Lewis E (2008) Wireless sensor node hardware: a review. In: IEEE sensors, Lecce
26. Yick J, Mukherjee B, Ghosal D (2008) Wireless sensor network survey. Comput Netw (Elsevier) 52(12):2292–2330
27. Sun Z, Akyildiz IF (2010) Magnetic induction communications for wireless underground sensor networks. IEEE Trans Antennas Propag 58(7):2426–2435
28. Bhuvaneswari P, Balakumar R, Vaidehi V, Balamuralidhar P (2009) Solar energy harvesting for wireless sensor networks. In: International conference on computational intelligence, communication systems and networks (CICSYN), Indore
29. Seah WK, Eu ZA, Tan H-P (2009) Wireless sensor networks powered by ambient energy harvesting (WSN-HEAP) – survey and challenges. In: International conference on wireless communication, vehicular technology, information theory and aerospace & electronic systems technology (Wireless VITAE), Aalborg
30. Bokareva T. Mini hardware survey [Online]. Available: http://www.cse.unsw.edu.au/~sensar/hardware/hardware_survey.html. Accessed 7 Jan 2012
31. Uchiyama T, Uehara Y, Mori M, Saito H, Tobe Y (2006) A system for analyzing life rhythm using wireless sensors on mules. In: International conference on mobile data management (MDM), Nara
32. Romer K, Mattern F (2004) The design space of wireless sensor networks. IEEE Wirel Commun 11(6):54–61
33. Naumowicz T, Freeman R, Kirk H, Dean B, Calsyn M, Liers A, Braendle A, Guilford T, Schiller J (2010) Wireless sensor network for habitat monitoring on Skomer island. In: IEEE conference on local computer networks (LCN), Denver
34. Fok C-L, Roman G-C, Lu C (2007) Towards a flexible global sensing infrastructure. ACM SIGBED Rev 4(3):1–6, Special Issue on the Workshop on Wireless Sensor Network Architecture
35. Polastre J, Szewczyk R, Mainwaring A, Culler D, Anderson J (2004) Analysis of wireless sensor networks for habitat monitoring. In: Wireless sensor networks. Kluwer Academic, Norwell, pp 399–423
36. Schoonmaker P, Luscombe W (2005) Habitat monitoring: an approach for reporting status and trends for state comprehensive wildlife conservation strategies. 22 March 2005 [Online]. Available: http://www.defenders.org/resources/publications/programs_and_policy/biodiversity_partners/habitat_monitoring.pdf. Accessed 8 Jan 2012
37. Agarwal R, Martinez-Catala R, Harte S, Segard C, O'Flynn B (2008) Modeling power in multi-functionality sensor network applications. In: International conference on sensor technologies and applications (SENSORCOMM), Cap Esterel
38. Dargie W (2012) Dynamic power management in wireless sensor networks: state-of-the-art. IEEE J Sensors 12(5):1518–1528
39. Sohrabi K, Gao J, Ailawadhi V, Pottie GJ (2000) Protocols for self-organization of a wireless sensor network. IEEE Pers Commun 7(5):16–27

40. Saglam O, Dalkili ME (2009) A self organizing multihop clustering protocol for wireless sensor networks. In: International conference on mobile ad-hoc and sensor networks (MSN), Fujian
41. Kacimi R, Dhaou R, Beylot A (2008) Energy-aware self-organization algorithms for wireless sensor networks. In: IEEE global telecommunications conference (GLOBECOM), New Orleans
42. Berman K, Annexstein F, Ranganathan A (2006) Dominating connectivity and reliability of heterogeneous sensor networks. In: IEEE international conference on pervasive computing and communications workshop (PerCom), Pisa
43. Abbasi A, Younis M, Baroudi U (2010) Restoring connectivity in wireless sensor-actor networks with minimal topology changes. In: International conference on communications (ICC), Cape Town
44. Ghosh A, Das SK (2008) Coverage and connectivity issues in wireless sensor networks: a survey. Pervasive Mob Comput 4(3):303–334
45. Wang P, Sun Z, Vuran M, Al-Rodhaan MA, Al-Dhelaan A, Akyildiz IF (2011) On network connectivity of wireless sensor networks for sandstorm monitoring. Comput Netw 55(5):1150–1157
46. Lu M, Wu J, Cardei M, Li M (2009) Energy-efficient connected coverage of discrete targets in wireless sensor networks. Int J Ad Hoc Ubiquitous Comput 4(3–4):137–147
47. Wang X, Xing G, Zhang Y, Lu C, Pless R, Gill C (2003) Integrated coverage and connectivity configuration in wireless sensor networks. In: International conference on embedded networked sensor systems (SenSys), Los Angeles
48. Bose R, Helal A (2010) Sensor-aware adaptive push–pull query processing in wireless sensor networks. In: International conference on intelligent environments (IE), Kuala Lumpur
49. Mottola L, Picco GP (2011) Programming wireless sensor networks: fundamental concepts and state of the art. ACM Comput Surv (CSUR) 43(3):1–51
50. Tilak S, Abu-Ghazaleh NB, Heinzelman W (2002) A taxonomy of wireless micro-sensor network models. ACM SIGMOBILE Mob Comput Commun Rev 6(2):28–36
51. Villalba L, Orozco A, Cabrera A, Abbas C (2009) Routing protocols in wireless sensor networks. Sensors 9(11):8399–8421
52. Akkaya K, Younis M (2005) A survey on routing protocols for wireless sensor networks. Elsevier Ad Hoc Netw 3(3):325–349
53. Niculescu D (2005) Communication paradigms for sensor networks. Commun Mag 43(3):116–122
54. Paek J, Govindan R (2007) RCRT: rate-controlled reliable transport for wireless sensor networks. In: Proceedings of the 5th international conference on embedded networked sensor systems (SenSys'07), Sydney
55. Rathnayaka AJD, Potdar VM (2013) Wireless sensor network transport protocol: a critical review. J Netw Comput Appl 36(1):134–146
56. Vuran M, Akyildiz I (2010) XLP: a cross-layer protocol for efficient communication in wireless sensor networks. IEEE Trans Mob Comput 9(11):1578–1591
57. Al-Karaki JN, Kamal AE (2004) Routing techniques in wireless sensor networks: a survey. IEEE Wirel Commun 11(6):6–28
58. Wang Z, Bulut E, Szymanski BK (2009) Energy efficient collision aware multipath routing for wireless sensor networks. In: IEEE international conference on communications, Dresden
59. Shah RC, Rabaey JM (2002) Energy aware routing for low energy ad hoc sensor networks. In: IEEE wireless communications and networking conference, Orlando
60. Chang JH, Tassiulas L (2004) Maximum lifetime routing in wireless sensor networks. IEEE/ACM Trans Netw 12(4):609–619
61. Dulman S, Nieberg T, Wu J, Havinga P (2003) Trade-off between traffic overhead and reliability in multipath routing for wireless sensor networks. In: IEEE wireless communications and networking, New Orleans
62. Saleem K, Fisal N, Hafizah S, Kamilah S, Rashid R (2009) A self-optimized multipath routing protocol for wireless sensor networks. Int J Recent Trends Eng (IJRTE) 2(1):93–97

63. Okdem S, Karaboga D (2009) Routing in wireless sensor networks using an ant colony optimization (ACO) router chip. Sensors 9(2):909–921
64. Saleem M, Di Caro GA, Farooq M (2011) Swarm intelligence based routing protocol for wireless sensor networks: survey and future directions. Inform Sci 181(20):4597–4624
65. Kulik J, Heinzelman W, Balakrishnan H (2002) Negotiation-based protocols for disseminating information in wireless sensor networks. Wirel Netw 8(2/3):169–185
66. Huang X, Fang Y (2008) Multiconstrained QoS multipath routing in wireless sensor networks. Wirel Netw 14(4):465–478
67. Kusy B, Lee HJ, Wicke M, Milosavljevic N, Guibas L (2009) Predictive QoS routing to mobile sinks in wireless sensor networks. In: International conference on information processing in sensor networks (IPSN), San Francisco
68. Ben-Othman J, Yahya B (2010) Energy efficient and QoS based routing protocol for wireless sensor networks. Elsevier J Parallel Distrib Comput 70(8):849–857
69. Chen Y, Nasser N, El Salti T, Zhang H (2010) A multipath QoS routing protocol in wireless sensor networks. Int J Sensor Netw 7(4):207–216
70. Zhou Y, Fang Y, Zhang Y (2008) Securing wireless sensor networks: a survey. IEEE Commun Surv Tutorials 10(3):6–28
71. Sheu J-P, Jiang J-R, Tu C (2008) Anonymous path routing in wireless sensor networks. In: IEEE international conference on communications (ICC), Beijing
72. Zhang Z, Jiang C, Deng J (2010) A secure anonymous path routing protocol for wireless sensor networks. In: IEEE international conference on wireless communications, networking and information security (WCNIS), Beijing
73. Zahariadis T, Leligou H, Voliotis S, Maniatis S, Trakadas P, Karkazis P (2009) Energy-aware secure routing for large wireless sensor networks. World Sci Eng Acad Soc Trans Commun 8(9):981–991
74. Singh S, Singh M, Singh D (2010) Routing protocols in wireless sensor networks—a survey. Int J Comput Sci Eng Surv (IJCSES) 1(2):63–83
75. Koliousis A, Sventek J (2007) Proactive vs reactive routing for wireless sensor networks. Department of Computing Science, University of Glasgow, Glasgow
76. Du X, Lin F (2005) Improving routing in sensor networks with heterogeneous sensor nodes. In: Vehicular technology conference, Stockholm
77. Kim H, Abdelzaher T, Kwon W (2003) Minimum-energy asynchronous dissemination to mobile sinks in wireless sensor networks. In: 1st international conference on embedded networked sensor systems (SenSys'03), Los Angeles
78. Yao Y, Gehrke J (2002) The cougar approach to in-network query processing in sensor networks. ACM SIGMOD Rec 31(3):9–18
79. Lindsey S, Raghavendra C (2002) PEGASIS: power-efficient gathering in sensor information systems. In: IEEE aerospace conference, Big Sky
80. Langendoen K, Halkes G (2005) Energy efficient medium access control. In: Embedded systems handbook, CRC, pp 34.1–34.29
81. Tang L, Yanjun S, Gurewitz O, Johnson DB (2011) PW-MAC: an energy-efficient predictive-wakeup MAC protocol for wireless sensor networks. In: IEEE INFOCOM, Shanghai
82. Ye W, Heidemann J, Estrin D (2002) An energy-efficient MAC protocol for wireless sensor networks. In: INFOCOM 2002, 21st annual joint conference of IEEE computer and communications societies, New York
83. Buettner M, Yee GV, Anderson E, Han R (2006) X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks. In: Proceedings of 4th international conference on embedded networked sensor systems (SenSys), Boulder
84. Ren Q, Liang Q (2005) An energy-efficient MAC protocol for wireless sensor networks. In: IEEE Globecom, St. Louis
85. Bachir A, Dohler M, Watteyne T, Leung K (2010) MAC essentials for wireless sensor networks. IEEE Commun Surv Tutorials 12(2):222–248
86. Rao P, Rani R, Sankar S, Pradeep K, Kumar N, Kumar V (2010) Secure MAC for wireless sensor networks through RBFNN. Int J Eng Sci Technol 2(8):3510–3514

87. Hu Q, Tang Z (2011) ATPM: an energy efficient MAC protocol with adaptive transmit power scheme for wireless sensor networks. J Multimedia 6(2):122–128
88. Yigitel M, Incel O, Ersoy C (2011) QoS-aware MAC protocols for wireless sensor networks: a survey. Elsevier Comput Netw 8(1):1982–2004
89. Langendoen K (2011) The MAC alphabet soup served in wireless sensor networks – taxonomy. Delft University of Technology, 16 March 2011 [Online]. Available: http://www.st.ewi.tudelft.nl/~koen/MACsoup/taxonomy.php. Accessed 12 July 2012
90. Sugihara R, Gupta RK (2008) Programming models for sensor networks: a survey. ACM Trans Sensor Netw (TOSN) 4(2):1–29
91. Farooq M, Kunz T (2011) Operating systems for wireless sensor networks: a survey. Sensors 11(6):5900–5930
92. Levis P, Madden S, Polastre J, Szewczyk R, Whitehouse K, Woo A, Gay D, Hill J, Welsh M, Brewer E, Culler D (2005) TinyOS: an operating system for sensor networks. In: Ambient intelligence. Springer, New York, pp 115–148
93. Gay D, Levis P, von Behren R, Welsh M, Brewer E, Culler D (2003) The nesC language: a holistic approach to networked embedded systems. In: Proceedings of programming language design and implementation (PLDI), San Diego
94. Anastasi G, Conti M, Di Francesco M, Passarella A (2009) Energy conservation in wireless sensor networks: a survey. Elsevier Ad Hoc Netw 7(3):537–568
95. TinyOS Tutorial Lesson 1: getting started with TinyOS and nesC. TinyOS, 9 September 2003 [Online]. Available: http://www.tinyos.net/tinyos-1.x/doc/tutorial/lesson1.html. Accessed 8 Jan 2012
96. Thusu R (2010) Wireless sensor use is expanding in industrial applications. Sensors, 1 June 2010 [Online]. Available: http://www.sensorsmag.com/networking-communications/wireless-sensor/wireless-sensor-use-is-expanding-industrial-applications-7212. Accessed 8 Jan 2012
97. Boyle D, Newe T (2007) Security protocols for use with wireless sensor networks: a survey of security architectures. In: International conference on wireless and mobile communications (ICWMC), Guadeloupe, French Caribbean
98. Perrig A, Szewczyk R, Tygar JD, Wen V, Culler D (2002) SPINS: security protocols for sensor networks. Wirel Netw 8(5):521–534
99. Zhu S, Setia S, Jajodia S (2003) LEAP: efficient security mechanisms for large-scale distributed sensor networks. In: ACM conference on computer and communications security, Washington, DC
100. Karlof C, Sastry N, Wagner D (2004) TinySec: a link layer security architecture for wireless sensor networks. In: International conference on embedded networked sensor systems, Baltimore
101. Kulkarni RV, Forster A, Venayagamoorthy GK (2011) Computational intelligence in wireless sensor networks: a survey. IEEE Commun Surv Tutorials 13(1):68–96