

## Chapter 2

# Wireless Sensor Networks: Concepts and Components

In this chapter we describe the main concepts and components of wireless sensor networks (WSN). It gives a basic understanding of the concepts behind wireless sensor networks, which is important as a prerequisite to a discussion of software considerations that is the core subject for the rest of this book.

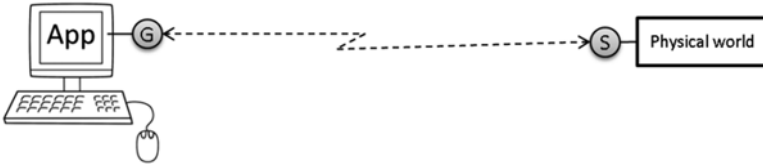
### 2.1 Network Components

The components of a wireless sensor network enable wireless connectivity within the network, connecting an application platform at one end of the network with one or more sensor or actuator devices in any part of the network. As shown in Fig. 2.1, using specific components such as gateways and nodes, a transparent data path is created between the application platform and the physical world.

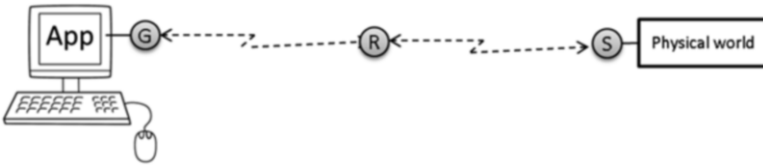
Wireless sensor networks are used to exchange information between an application platform and one or more sensor nodes. This exchange takes place in a wireless fashion. In the example of Fig. 2.1, the data path between the gateway and the node is referred to as a single-hop network link.

To extend the range of a network or circumvent an obstacle, a wireless relay node can be added between a gateway and a leaf node, making a mesh network, as shown in Fig. 2.2.

In this example we represent a multi-hop data path, in which data packets are sent from one node to the next node before reaching their destination (gateway, other node). More elaborate network layouts are discussed later in Sect. 2.4. Now we will describe each network component illustrated in Fig. 2.2 (gateway, relay node, leaf node and sensor/actuator).



**Fig. 2.1** Basic wireless sensor network components



**Fig. 2.2** Adding intermediate node (relay node)

*Gateway:* A gateway is an interface between the application platform and the wireless nodes on the wireless sensor network. All information received from the wireless nodes is aggregated/manipulated (e.g. translation between network packet formats) by the gateway and forwarded to the application. That application may run on a local computer or a networked computer. In the reverse direction, when a command is issued by the application program to a wireless node, the gateway relays the information to the wireless sensor network.

All gateways can perform protocol conversion to enable the wireless network to work with other industry or non-standard network protocols.

*Relay Node:* Each relay node is considered a full-function device (FFD). They are usually called “routers,” and they are used to extend network coverage area, route around obstacles and provide back-up routes in case of network congestion or device failure. In some cases, relay nodes may also be connected via analog and digital interfaces to sensors and actuators, providing the same I/O functionality of a leaf node.

*Leaf Node:* A leaf node is considered as a reduced-function device (RFD). It is sometimes called endpoint. It is designed to provide the physical interface between the wireless sensor network and the sensor or actuator that it is wired to.

Leaf nodes are usually equipped with one or more I/O connections for connecting to and communicating with analog or digital sensors or actuator devices.

*Sensor/Actuator:* This is the device use for interaction with the physical system that you ultimately wish to monitor and/or control. An example is a sensor monitoring the temperature in a room and controlling the air-conditioned equipment.

## 2.2 Hardware Platforms

Sensor, actuation and computation nodes are the fundamental components of distributed systems with wireless sensor nodes. To enable WSN-based applications, nodes (in general) have to provide the following basic functionality:

- Signal conditioning and data acquisition for different sensors
- Storage of data (sample data and configurations)
- Processing capabilities
- Analysis of the processed data for alert generation
- Actuation
- Scheduling and execution of the measurement tasks
- Management of node configuration (e.g. changing the sampling rate and reprogramming of data processing algorithms)
- Reception, transmission and forwarding of data packets
- Scheduling and execution of communication and networking tasks

An FFD node can be an embedded device or a more powerful computer server or workstation, and it may need to provide any of the functionalities described above. A RFD is typically an embedded device and provides only part of the functionalities described above.

Computer-based platforms run mainly on Windows, Linux or other operating systems developed for computer hardware. These platforms are predominantly equipped with standard LAN communication (IEEE 802.11). Because of the high processing ability and high communication bandwidth, these platforms offer the opportunity to use higher-level programming languages (e.g. Java, C++), which make it easier to develop and implement software components. Additionally, they support networking protocols like Internet Protocol (IP), which simplifies the integration into enterprise systems.

But although those platforms are very flexible in terms of configuration and computing power, they are not adequate to deploy in each sensing and actuation location, since they are too expensive, big and requiring external power. Embedded devices are more suitable for those scenarios. Typically, they have limited resources and small sizes, and sometimes they are battery operated (e.g. some wireless devices). Various wireless devices are available today for building WSNs (e.g. MICAz [1], TelosB [2] motes, Wasp mote [3], Econotag [4]), and new ones emerge regularly. This diversity offers the possibility to choose a platform that best fits the needs of specific applications.

Typically, processor, radio and memory capabilities of wireless devices are very constrained, making them cheap. The microcontroller unit (MCU) is most frequently programmed in C. This enables the development of a tight code that fits the limited memory size. Application developers have full access to hardware but at the same time need to take care of resource constraints.

Unlike operating systems for standard computers, such as Windows or Linux, WSN software platforms are highly tailored to the limited node hardware. These are not full-blown operating systems, since they lack a powerful scheduler, memory management and elaborate file system support.

## 2.3 Wireless Sensor Operating Software

TinyOS [5] and Contiki [6] are two of the most widespread operating systems. Other operating systems developed for WSNs include MANTIS [7], SOS [8], SensorOS [9] and MagnetOS [10]. In the next subsections, they are briefly described.

### 2.3.1 *TinyOS*

TinyOS [5] is written in nesC [11], an extension to the C language. It supports event-driven component-based programming. The basic concept of component-based programming is to decompose the program into functionally self-contained components. These components interact by exchanging messages through interfaces. The components are event-driven. Events can originate from the environment (e.g. a certain sensor reading exceeds a threshold) or from other components, triggering a specific action. The main advantage of this component-based approach is the reusability of components.

The nesC language extension introduces several additional keywords to describe a TinyOS component and its interfaces. NesC and TinyOS are both open-source projects supported by the research community.

TinyOS is the native operating system of the Tmote [12], but it has been ported to other WSN hardware platforms. TinyOS cannot, however, dynamically load a new executable without a complete image replacement and reboot. Nonetheless, it is the de facto standard tool for WSN programming.

### 2.3.2 *SOS*

SOS [8] improves on TinyOS by providing dynamic memory allocation, loadable modules and a kernel. Application programming is in C, a common and familiar language to many. The kernel takes care of network messaging, dynamic memory allocation and module loading and unloading. For over-the-network reprogramming, TinyOS requires that the entire mote image be shipped and the mote reboot, whereas SOS enables the use of smaller modules that can be dynamically loaded.

### 2.3.3 *Contiki*

Contiki [6] is a memory-efficient open-source operating system for networked embedded devices. Contiki provides standard OS features like proto-threads, timers, random number generator, clock and a file system support. It includes an IPv6 stack with support for TCP and UDP connections, as well as the Rime radio communication stack [13].

Contiki is supported by an event-driven kernel with small footprint. The Contiki kernel consists of a lightweight event scheduler that dispatches events to running processes and periodically calls processes polling handlers. All program execution is triggered either by events dispatched by the kernel or through the polling mechanism. The kernel does not pre-empt an event handler once it has been scheduled. It supports two kinds of events: asynchronous and synchronous.

### 2.3.4 *MANTIS*

MANTIS [7] introduces pre-emptive thread scheduling. A pre-emptive scheduler is particularly important to enforce fairness among threads or tasks. MANTIS also provides multi-granular code injection: at the level of the whole system, a single thread or even thread variables. This provides dynamic and efficient runtime reprogrammability, even after a network has been deployed. MANTIS offers POSIX-style system calls in C and a remote interactive shell.

### 2.3.5 *SensorOS*

SensorOS [9] is a multithreading operating system that supports pre-emptive priority-based scheduling, very fine-granularity timing and message passing inter-process communication. It has been implemented for resource-constrained Tampere University of Technology WSN (TUTWSN) nodes [9]. In TUTWSN node platform with 2 MIPS PIC microcontroller unit, SensorOS kernel uses 6,964 byte code and 115 bytes of data memory. Compared to event-handler kernels, such as TinyOS or ContikiOS, SensorOS enables coexistence of multiple time critical application tasks.

### 2.3.6 *MagnetOS*

MagnetOS [10] is a distributed operating system for ad hoc and sensor networks whose goal is to enable power aware, adaptive and easy-to-develop ad hoc networking applications. These properties are provided through a single system image of a unified

virtual machine to applications over an ad hoc collection of nodes. It automatically and transparently partitions applications into components and dynamically finds a placement of these components on nodes within the ad hoc network to reduce energy consumption and increase system longevity.

### **2.3.7 Nano-RK**

Nano-RK [14] is a real-time operating system (RTOS) with multi-hop networking support for use in wireless sensor networks. Nano-RK supports fixed-priority pre-emptive multitasking for guaranteeing that task deadlines are met, along with support for CPU and network bandwidth reservations. Tasks can specify their resource demands, and the operating system provides timely, guaranteed and controlled access to CPU cycles and network packets in resource-constrained embedded sensor environments.

Nano-RK includes a lightweight wireless networking stack for packet forwarding, routing and TDMA-based network scheduling.

### **2.3.8 ERIKA**

ERIKA Enterprise RTOS [15] is a multiprocessor real-time operating system kernel, implementing a collection of application programming interfaces (APIs) similar to those of OSEK/VDX standard for automotive embedded controllers. ERIKA features a real-time scheduler and resource managers, allowing the full exploitation of the power of new-generation microcontrollers and multicore platforms.

Tasks in ERIKA are scheduled according to fixed and dynamic priorities and share resources using the Immediate Priority Ceiling Protocol. Interrupts always pre-empt the running task to execute urgent operations required by peripherals.

ERIKA Enterprise includes a RT-Druid Eclipse-based development environment, which allows writing, compiling and analysing an application. RT-Druid is composed of a set of plug-ins such as schedulability analysis plug-in, which implements algorithms like scheduling acceptance tests, sensitivity analysis and task offset calculation. It also includes a set of design tools for modelling, analysing and simulating the timing behaviour of embedded real-time systems.

### **2.3.9 RETOS**

RETOS [16] is a resilient, expandable and threaded operating system for wireless sensor networks. RETOS is a multithreaded operating system based on a modular software approach which provides separated modules for kernel and user applications and supports their robust execution on constrained hardware. The RETOS kernel can be dynamically reconfigured, via loadable kernel framework.

In order to run RETOS in constrained devices, its framework provides a software technique called application code checking which consists of static and dynamic checks by accessing memory to check memory outside legal boundary. To achieve this property, the framework inspects all memory instructions.

2.3.10 LiteOS

LiteOS [17] is an operating system that provides Unix-like abstractions to wireless sensor networks. LiteOS maps a sensor network into a Unix-like file system. It was designed to support resource-constrained nodes such as MICAz. It uses C programming language natively and allows online debugging.

LiteOS provides a wireless node mounting mechanism through a file system called LiteFS. LiteOS node mounts itself wirelessly to the root filesystem of a nearby base station. Once mounted, a LiteOS node looks like a file directory from the base station. A sensor network, therefore, maps into a higher-level directory composed of node-mapped directories. The shell, called LiteShell, supports Unix commands, such as copy (cp), executed on such directories. LiteOS has a multi-threaded kernel to run applications as threads concurrently. LiteShell provides three commands to control thread behaviour: ps, exec and kill.

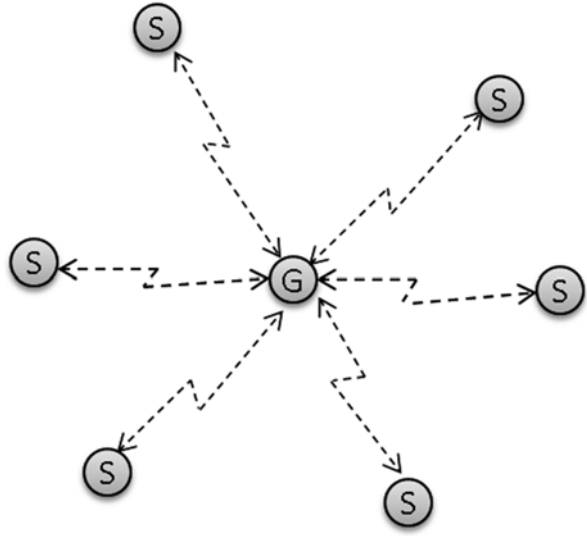
Of these ten operating systems for wireless sensor networks, TinyOS and Contiki are the most familiar to the WSN programmer, offering high-level programming languages and all components and capabilities needed to create a WSN. However, Nano-RK [14] and ERIKA Enterprise RTOS [15] are also widely used in real-time applications.

2.4 Network Topologies

Since a wireless sensor network may consist of tens, hundreds or thousands of devices, network topologies must be considered in its design. The most common network topologies used in wireless sensor networks are star, tree mesh or hybrid networks that combine the other ones. Each of these topologies presents its own set of challenges, advantages and disadvantages, as shown in Table 2.1 and discussed below.

Table 2.1 Network topologies

Topology	Power usage	Communication range	Requires time synchronization
Star	Low	Short	No
Tree	Low	Long	Yes
Mesh	High	Long	No
Hybrid	Low (typically)	Long	(Depends on the configuration)

**Fig. 2.3** Star topology

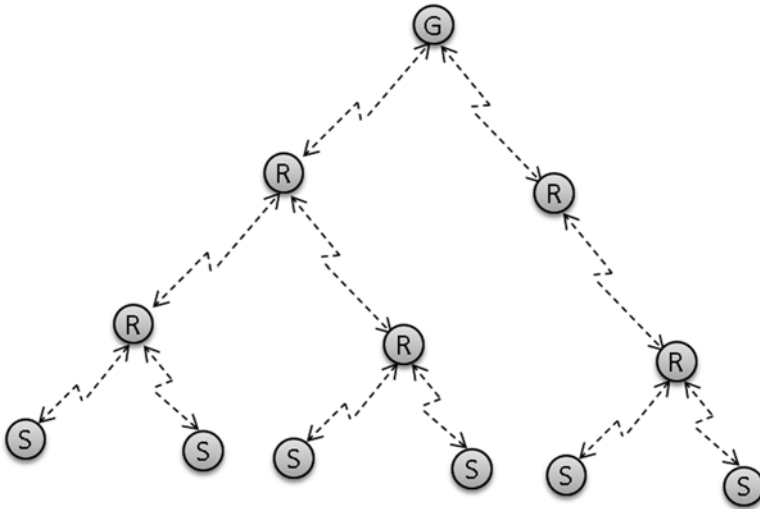
### 2.4.1 *Star Topology*

A star topology, as shown in Fig. 2.3, is a single-hop system in which all wireless sensor nodes communicate bi-directionally with a gateway. This allows easy connection for small networks, but a hierarchical scheme with multiple subnetworks should be considered once a maximum number of nodes is reached. If a node fails in a star configuration, it does not affect other nodes, unless it is the central node that fails. The star topology may have one or more network segments that radiate from the central node. Addition of further nodes is easy and can be done without interrupting network operation.

The gateway can be a computer or a dedicated embedded device, and it acts both to communicate data and commands among nodes and to transfer data to an application or other network, such as the Internet. Nodes do not transmit data or commands to each other, directly. They use the gateway as a coordination point where all data flow is concentrated.

Among wireless sensor networking topologies, the star topology is the lowest in overall power consumption but is limited by the transmission distance of the radio in each node back to the gateway. Typically, this distance ranges from 10 to 100 m, but this depends on the radio frequency used by the node (radio technology). Generically, higher ranges correspond to higher power consumption, which is an important factor when considering the ranges for autonomous, battery-backed devices.





**Fig. 2.4** Tree topology

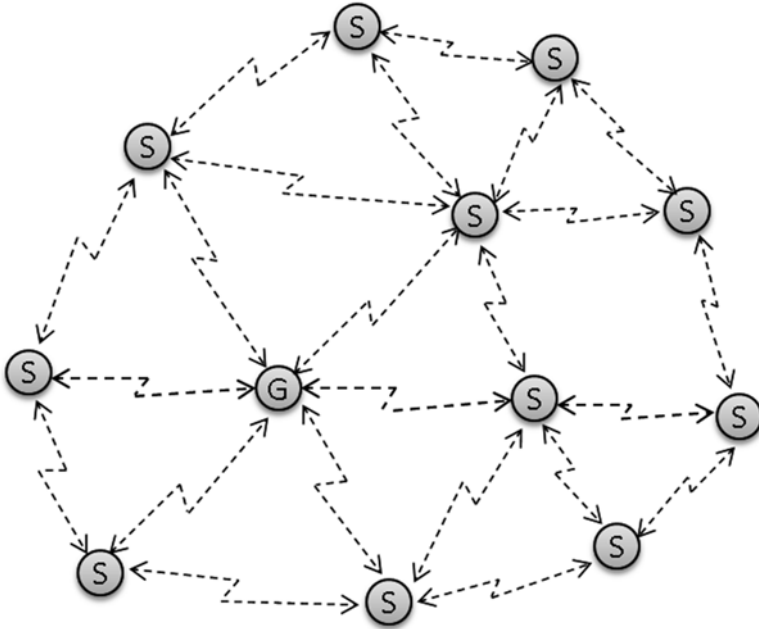
### 2.4.2 Tree Topology

In the tree topology, there are leaf nodes, relay nodes and parent nodes. This topology consists of a central node called the root node, and it is the main communications router and interface with the application or other parts of a larger network. One level down from the root node in the hierarchy is a relay node that may have several children nodes connected to it and so on. An example of a tree network is shown in Fig. 2.4.

This network configuration is designed to optimize power consumption and to extend the network communication range. However, data transmission and operations are scheduled, which requires time synchronization between all nodes.

### 2.4.3 Mesh Topology

This kind of network topology connects each node to all the nodes within its radio range (Fig. 2.5). It has the potential for reduced power consumption when compared to other topologies using long distances, since each node can establish a radio link with the closest neighbours. However, it is required that nodes are always waked up, because they can be part of the path between any other node and the gateway.



**Fig. 2.5** Mesh topology

A mesh network can be highly fault tolerant, because each sensor node has multiple paths back to the gateway or to other nodes. The multi-hop system allows for a much longer range than a star topology.

#### **2.4.4 Hybrid Topology**

Some network topologies used for wireless sensor network applications use a hybrid combination of the previous topologies, to create larger networks consisting of hundreds, even thousands of nodes.

A hybrid network consists of a combination of star and mesh topologies. This combination results on a star-mesh network that seeks to take advantage of the low power and simplicity of the star topology, as well as the extended range and self-healing nature of a mesh network topology. In this case, nodes serve to sense, extend the range of the network and provide fault tolerance.

Since nodes can communicate with multiple other nodes, if a node fails or if a radio link goes down (e.g. due to interferences or lack of battery), the network will reconfigure itself around the remaining nodes.

## 2.5 Data Models

The data model characterizes and describes the interaction between the sensors and the application. Unlike the topology, which is a function of the network protocol, the data model is a function of the application. The most appropriated data model for an application is determined according to the application's requirements. There are different models for monitoring applications, where the data flows primarily from the sensor node to the gateway, and for control applications, where the data also flows very frequently from the gateway to sensor nodes. The most usual data models for monitoring are:

*Periodic Sampling:* For applications where certain conditions or processes need to be monitored constantly, such as the temperature in a conditioned space or pressure in a process pipeline, sensor data is acquired from a number of sensor nodes and forwarded to the gateway on a periodic basis. The sampling period mainly depends on how fast the condition or process varies and what intrinsic characteristics need to be captured.

In many cases, the dynamics of the process to be monitored can vary over the time. Therefore, sensor nodes must be able to adapt their sampling rates to the changing dynamics of the process. This adaptation should be done automatically or through commands issued by the application.

*Event Driven:* There are many cases that require monitoring crucial variables immediately following a specific event or condition. Common examples include fire alarms, pressure alarms or door and window sensors. To support event-driven operations with adequate power efficiency and speed of response, the sensor node must be designed such that its power consumption is minimal in the absence of any triggering event, and the wake-up time is relatively short when the specific event or condition occurs. Many applications require a combination of event driven and periodic sampling.

*Store and Forward:* In many applications, data can be captured and stored or even processed by a sensor node before it is transmitted to the gateway. Instead of immediately transmitting every sample as it is acquired, aggregating and processing data by sensor nodes can improve overall network performance in both power consumption and bandwidth efficiency.

## 2.6 Routing Techniques

In this section, we describe the advantages and disadvantages associated with the different routing techniques developed specifically for wireless sensor networks.

Once the data models are described, it is important to understand how data is forwarded to the gateway. A wireless sensor network relies on its network layer's routing algorithm to discover routes and deliver data packets from sources

to destinations. The routing layer protocol is also responsible for maintaining and repairing routes when radio links (or hops) along established routes are broken, due to relocation or failure of nodes, server RF interference or congestion. It is the routing algorithm that enables a wireless sensor network to self-organize and self-heal.

Routing in sensor networks is very challenging due to several characteristics that distinguish them from contemporary communication and wireless ad hoc networks. First of all, it is not possible to build global addressing and routing algorithms exactly as for classical IP-based protocols for the deployment of sheer numbers of energy and processing capacity constrained sensor nodes. Second, in contrary to typical communication networks, almost all applications of sensor networks require the flow of sensed data from multiple regions (sources) to a particular sink or gateway. Third, generated data traffic has significant redundancy in it, since multiple sensors may generate the same data within the vicinity of a phenomenon. Such redundancy needs to be exploited by the data and goal-oriented routing protocols to improve energy and bandwidth utilization. Fourth, sensor nodes are tightly constrained in terms of transmission power, on-board energy, processing capacity and storage and thus require careful resource management.

Due to such differences, many new algorithms have been proposed for the problem of routing data in sensor networks. These routing mechanisms have considered the characteristics of sensor nodes along with the application and architecture requirements. Almost all of these routing protocols can be classified as data centric, hierarchical or location based, there being also some distinct ones based on network flow or QoS awareness. Data-centric protocols are query based and depend on the naming of desired data, which helps in eliminating many redundant transmissions. Hierarchical protocols aim at clustering the nodes so that cluster heads can do some aggregation and reduction of data in order to save energy. Location-based protocols utilize the position information to relay the data to the desired regions rather than the whole network. The last category includes routing approaches that are based on general network-flow modelling and protocols that strive for meeting some QoS requirements along with the routing function.

In this section, we will explore the routing mechanisms for sensor networks developed in recent years. Our aim is to help better understanding of the current data and goal-oriented routing protocols for wireless sensor networks and point out open issues that can be subject to further research.

There are some previous works surveying the characteristics, applications and communication protocols in WSNs [18–25]. While most existing surveys address several design issues and techniques for WSNs, describing the physical constraints on sensor nodes, applications and architectural attributes, this section is devoted to the study of data and goal-oriented routing in sensor networks, describing and categorizing the different approaches for data delivery. Different protocols and algorithms proposed in recent years are reviewed. We begin with data-centric protocols, which are centred on the data itself. Secondly, we analyse hierarchical routing, which consists on establishing a hierarchical route towards the data collection points. Thirdly, we survey protocols that use position information to relay the data to the desired regions (location-based protocols), and lastly we consider QoS-aware protocols,

**Table 2.2** Routing algorithms and classes

Routing	Data centric	SPIN	SPIN-PP
			SPIN-EC
			SPIN-BC
			SPIN-RL
		Direct Diffusion	
		Energy aware	
		Reliable Energy Aware Routing (REAR)	
		Rumor	
		MCFA	
		Link quality estimation based	
		Gradient based	
		Information driven	
		Acquire	
	Hierarchical	LEACH	
		EWC	
		PEGASIS	
		TEEN/APTEEN	
		Energy-aware cluster based	
		Self-organized	
		Minimum energy communication network	
		Small minimum energy communication network	
	Location based	Geographic Adaptive Fidelity	
		Energy Aware Greedy Routing (EAGR)	
		Geographic and Energy Aware	
	QoS aware	SPEED	
		MMSPEED	
		Sequential Assignment	
		Real-Time Power-Aware	
		DCEERP	
		Energy Efficient with Delay Guaranties (EEDG)	

which take into account energy consumption and data quality. The network may have to satisfy certain QoS metrics (delay, energy, bandwidth) when delivering data to the base station, metrics which are used in the QoS-aware protocols. Table 2.2 lists the routing algorithms and classes that we discuss in this chapter.

In order to select the most suitable routing mechanism for a sensor application, we have to classify the routing protocols according to the network and operational characteristics and an objective model that describes the routing goal.

Table 2.3 categorizes the routing protocols from Table 2.2 taking into consideration, besides the main classification used in that table, also the number of base stations, mobility characteristics, whether transmission power is considered fixed or variable, whether the approach uses data aggregation or not and whether it is query based or not. Finally, we list the main goal for each alternative algorithm.

**Table 2.3** Properties of routing algorithms

Protocols	Classification	Number of base stations	Mobility	Transmission power	Data aggregation	Query based	Goal
SPIN	Data centric	1	Possible	Fixed	Yes	Yes	Lifetime, exchange metadata to reduce number of messages
Direct Diffusion	Data centric	1 or more	Limited	Fixed	Yes	Yes	Establish efficient n-way communication paths for fault tolerance
Energy aware	Data centric	1 or more	Limited	Adjustable	Yes	Yes	Lifetime
REAR	Data centric	1 or more	Limited	Adjustable	Yes	Yes	Lifetime and data delivery
Rumor	Data centric	1	Very limited	Fixed	Yes	Yes	Reduce queries in network
MCFA	Data centric	1	No	Fixed	No	No	Data delivery
LQEBR	Data centric	1	No	Fixed	No	No	Data delivery with minimal retransmissions
Gradient	Data centric	1	Limited	Fixed	Yes	Yes	Data delivery through minimal number of hops
Information driven	Data centric	1	Limited	Fixed	Yes	Yes	Optimization of Direct Diffusion to save more energy
Acquire	Data centric	1 or more	Limited	Fixed	Yes	Yes	Optimization of queries in network
LEACH	Hierarchical	1	Fixed BS	Fixed	Yes	No	Formation distributed cluster to extend Lifetime
EWC	Hierarchical	1	Fixed BS	Fixed	Yes	No	Forming distributed cluster to extend lifetime and data delivery guaranties

PEGASIS	Hierarchical	1	Fixed BS	Fixed	No	No	Lifetime and bandwidth optimization
TEEN/APTEEN	Hierarchical	1	Fixed BS	Fixed	Yes	No	Real time and lifetime
Energy-aware cluster based	Hierarchical	1	No	Adjustable	No	No	Real time and lifetime
Self-organized	Hierarchical	1	Possible	Fixed	No	No	Fault tolerance
Minimum energy communication network	Hierarchical	1	No	Adjustable	No	No	Lifetime and self-reconfiguration
Geographic Adaptive Fidelity	Location	1 or more	Limited	Fixed	No	No	Increase the network lifetime with the number of node increase
EAGR	Location	1 or more	Limited	Fixed	No	No	Optimization of Geographic Adaptive Fidelity algorithm
Geographic and Energy Aware	Location	1	Limited	Fixed	No	No	Reduce interest msgs in the whole network
SPEED	QoS	1 or more	No	Fixed	No	Yes	Real time and lifetime
MMSPEED	QoS	1 or more	No	Fixed	No	Yes	Real time and lifetime (improve SPEED)
Sequential Assignment	QoS	1	No	Fixed	Yes	Yes	Real time
Real-Time Power-Aware	QoS	1 or more	No	Adjustable	Yes	No	Real time and lifetime
DCEERP	QoS	1	No	Fixed	Yes	No	Real time and lifetime
EEDG	QoS	1 or more	No	Fixed	Yes	No	Deadlines guaranties and lifetime

In most applications, the number of gateways can be one or more than one. The increased number of gateways provides more robust data gathering and may also decrease the network delay. If only one gateway is present, the destination node for all messages is the same, while in the case of multiple gateways, the routing algorithms can take this into consideration to achieve desired goals. However, many techniques do not try to optimize the system when more than one gateway is available.

Another important factor is the transmission power, which can be either dynamically adjustable or fixed. When the transmission power is fixed, each sensor node transmits each message using the same energy level. In the other case, every node can calculate what energy level should be used to transmit a message to a neighbouring node. This energy level may be inversely proportional to the cost assigned to the neighbouring node.

The task of routing in many protocols is to deliver the queries coming from the gateway to the sensor which have the requested data and return the requested data to the gateway.

Some protocols are based on hierarchy. A hierarchy level is assigned to each node, and a node only forwards those messages that are originated from a lower-level node. Optionally, a node aggregates incoming data and forward this aggregated data to upper-layer nodes. The base station can be found on the top of the hierarchy. The hierarchy construction can be dynamic or static. In the dynamic case, the role of the aggregator is rotated, and all nodes that have selected an aggregator will forward all data to their aggregator. The aim of forming hierarchies is to prolong the network lifetime.

Some sensor applications only require the successful delivery of messages between a source and a destination. However, there are applications that need more guaranties. These are the real-time requirements of the message delivery, maximization of network lifetime and fault tolerance.

The main objective of the real-time protocols is to completely control the network delay. The average-case performance of these protocols can be evaluated by measuring the message delivery ratio with time constraints. The lifetime is another important goal. Protocols try to balance energy consumption equally among nodes, considering their residual energy levels. However, the metric used to determine network lifetime is also application dependent. Most protocols assume that all nodes are equally important, and they use the time until the first node dies as a metric, but the average energy consumption of the nodes may also be used as a metric.

Generically, these goals or qualities are achieved by introducing a set of heuristics into the routing algorithms: typically, lifetime can be controlled in routing algorithms by taking into account the level of power still available in the batteries during routing decisions, routing through the least power-consuming routes, controlling the transmission ratio or managing the duty cycle carefully; minimizing delays is controlled typically by delay-based scheduling of node transmission queues; fault tolerance is achieved in the reviewed algorithms by keeping multiple alternative routing paths and using an available one on-demand. While a minimum number of hops may minimize qualities such as delays, when nodes fail, the algorithms are



**Table 2.4** Routing goals

Protocols	Lifetime, min energy	Min. delay or *deadlines	Fault tolerance, reconfiguration	Opt. bandwidth, min. nr. of msgs	Min. nr. hops
SPIN	×			×	
Direct Diffusion			×		
Energy aware	×				
REAR	×		×		
Rumor				×	
MCFA					×
LQEB				×	
Gradient					×
Information driven	×				
Acquire				×	
LEACH	×				
EWC	×				
PEGASIS	×			×	
TEEN/APTEEN	×	×			
Energy-aware cluster based	×	×			
Self-organized			×		
Minimum energy communication network	×		×		
Geographic Adaptive Fidelity	×				
EAGR	×			×	
Geographic and Energy Aware				×	
SPEED	×	×			
MMSPEED	×	×		×	
Sequential Assignment		×			
Real-Time Power-Aware	×	×			
DCEERP	×	×			
EEDG	×	×			

\* Means that the protocol was designed to meet deadlines and not to achieve minimum delay for all transmissions

also able to choose alternative routes; optimum bandwidth and minimum number of messages are typically achieved by using an optimal transmission packet size.

Table 2.4 categorizes the routing protocols from Table 2.2 taking into consideration the main goals that each algorithm goes after. We describe these goals after the table.

Most of the algorithms in Table 2.4 take into consideration the battery lifetime as the main goal (e.g. SPIN, energy-aware routing, LEACH, minimum energy communication network, Geographic Adaptive Fidelity). A few of them consider

**Table 2.5** Routing typical application

Protocols	Project	Application type
Direct Diffusion	Sensor Web [26]	Environment monitoring
Gradient	Vital Sign [27]	Health
Acquire	Flood Detection [28]	Environment monitoring
LEACH	Artificial Retina [29]	Health
TEEN/APTEEN	Aware Home [30]	Industrial
Self-organized	SOMoM [31]	Military
Geographic Adaptive Fidelity	Great Duck [32]	Military
Geographic and Energy Aware	Aware Home [33]	Habitat monitoring
Sequential Assignment	Vital Sign [28]	Health

delays (e.g. Sequential Assignment), fault tolerance (Direct Diffusion, self-organized protocol) and bandwidth optimization (Rumor, Acquire). Some protocols focus on more than one goal (e.g. TEEN/APTEEN, energy-aware cluster based, SPEED, Real-Time Power-Aware). The principal goal for these last protocols is to control the network delay but simultaneously try to minimize the dissipated energy in transmission, in order to extend the network lifetime as much as possible under the delay constraints.

Table 2.5 categorizes the routing protocols from Table 2.2 taking into consideration their application type. Some typical application areas, such as habitat monitoring, environment monitoring and health-care and industrial environment, are shown, and the routing protocols are categorized in the table by typical application type.

In this section, we have summarized recent research results on data routing in sensor networks and classified the approaches into four main categories, namely, data centric, hierarchical, location based and QoS aware, and we have discussed the effect of node placement strategies on the operation and performance of WSNs.

Protocols which name the data and query the nodes based on some attributes of the data are categorized as data centric. Many of the researchers follow this paradigm in order to avoid the overhead of forming clusters, the use of specialized nodes. On the other hand, we present the cluster-based routing protocols, which group sensor nodes to efficiently relay the sensed data to the gateway. The cluster heads are sometimes chosen as specialized nodes that are less energy constrained. A cluster-head performs aggregation of data and sends it to the gateway on behalf of the nodes within its cluster.

Performance-controlled planned networks, where placement and routing must be intertwined and everything from delays to throughput to energy requirements is well-defined and relevant, is an interesting subject of current and future research. Real-time, deadline guaranties and their relationship with routing, mac-layer, duty cycles and other protocol stack issues are interesting issues that would benefit from further research.

Another interesting issue for data and goal-oriented routing protocols is the consideration of node mobility. Most of the current protocols assume that the sensor nodes and the sink are stationary. However, there might be situations such as medical environments where the sink and possibly the sensors need to be mobile. In such cases, the frequent update of the position of nodes and the propagation of that information through the network may excessively drain the energy of nodes. There already exist some works addressing the mobility, such as [34–38], but new data and goal-oriented routing algorithms are needed in order to handle the overhead of mobility and topology changes in such energy constrained environment.

Other possible research consists on the integration of sensor networks with wired networks in industry. Nowadays, most of sensor networks in industry are made with wired cable. To expand these existing sensor networks, we can use wireless sensors, but it must be integrated into the existing wired network seamlessly.

## References

1. Crossbow (2007) MICAz datasheet. Open Automation, Inc [Online]. Available: [http://www.openautomation.net/uploads/productos/micaz\\_datasheet.pdf](http://www.openautomation.net/uploads/productos/micaz_datasheet.pdf). Accessed 31 July 2014
2. Crossbow (2004) TelosB datasheet. Moteiv Corporation, pp 1–28 [Online]. Available: <http://www4.ncsu.edu/~kkolla/CSC714/datasheet.pdf>. Accessed 31 July 2014
3. Libelium (2010) Waspote datasheet. Libelium Comunicaciones Distribuidas S.L. [Online]. Available: [http://web.univ-pau.fr/~cpham/ENSEIGNEMENT/PAU-UPPA/RESA-M2/DOC/waspote-datasheet\\_eng.pdf](http://web.univ-pau.fr/~cpham/ENSEIGNEMENT/PAU-UPPA/RESA-M2/DOC/waspote-datasheet_eng.pdf). Accessed 31 July 2014
4. Semiconductor F (2011) Econotag datasheet. [Online]. Available: <https://github.com/malvira/libmc1322x/wiki>. Accessed 31 July 2014
5. Levis P, Madden S, Polastre J, Szewczyk R, Woo A, Gay D, Hill J, Welsh M, Brewer E, Culler D (2005) Tinyos: an operating system for sensor networks. In: Ambient intelligence, vol II. Springer, Berlin/New York, pp 115–148
6. Dunkels A, Gronvall B, Voigt T (2004) Contiki – a lightweight and flexible operating system for tiny networked sensors. In: Proceedings of the 29th annual IEEE international conference on local computer networks. IEEE Computer Society, Los Alamitos, pp 455–462
7. Bhatti S, Carlson J, Dai H, Deng J, Rose J, Sheth A, Shucker B, Gruenwald C, Torgerson A, Han R (2005) MANTIS OS: an embedded multithreaded operating system for wireless micro sensor platforms. *Mobile Netw Appl* 10(4):563–579
8. Han CC, Kumar R, Shea R, Kohler E, Srivastava M (2005) SOS: a dynamic operating system for sensor networks. In: Proceedings of the third international conference on mobile systems applications and services (Mobisys). ACM, New York, pp 163–176
9. Kuorilehto M, Alho T, Hannikainen M, Hamalainen TD (2007) SensorOS: a new operating system for time critical WSN applications. In: Embedded computer systems: architectures, modeling, and simulation. Springer, Berlin/New York, pp 431–442
10. Barr R, Bicket J, Dantas D, Du B, Kim TW, Zhou B, Siler EG (2002) On the need for system-level support for ad hoc and sensor networks. *ACM SIGOPS Operat Syst* 36(2):1–5
11. Gay D, Levis P, Von Behren R, Welsh M, Brewer E, Culler D (2003) The nesC language: a holistic approach to networked embedded systems. *ACM SIGPLAN Not* 38(5):1–11
12. Polastre J, Szewczyk R, Culler D (2005) Telos: enabling ultra-low power wireless research. In: IPSN 2005 fourth international symposium on information processing in sensor networks 2005, vol 00, no C. IEEE, Piscataway, pp 364–369

13. Dunkels A (2007) Rime-a lightweight layered communication stack for sensor networks. eprints.sics.se. [Online]. Available: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.64.360&rep=rep1&type=pdf>. Accessed 31 July 2014
14. Eswaran A, Rowe A, Rajkumar R (2005) Nano-RK: an energy-aware resource-centric RTOS for sensor networks. In: Proceedings of the 26th IEEE international real-time systems symposium RTSS05, vol 0. IEEE Computer Society, Miami, pp 256–265
15. ERIKA Enterprise (Online). Available: <http://erika.tuxfamily.org/drupal/>. Accessed 23 Aug 2013
16. Cha HCH, Choi SCS, Jung IJ, Kim HKH, Shin HSH, Yoo JYJ, Yoon CYC (2007) RETOS: resilient, expandable, and threaded operating system for wireless sensor networks. In: Proceedings of the 2007 6th international symposium on information processing in sensor networks. ACM, New York
17. Cao QCQ, Abdelzaher T, Stankovic J, He THT (2008) The LiteOS operating system: towards Unix-like abstractions for wireless sensor networks. In: Proceedings of the international conference information processing sensor networks (IPSN 2008), St Louis
18. Abu-Ghazaleh NB, Heinzelman W, Tilak S (2002) A taxonomy of wireless micro-sensor network models. ACM SIGMOBILE Mob Comput Commun Rev 6(2):28–36
19. Garg N, Aswal K, Dobhal DC (2012) A review of routing protocols in mobile Ad Hoc. Int J Inf Technol Knowl Manag 5(1):177–180
20. Govindaswamy V, Blackstone WL, Balasekara G (2011) Survey of recent position based routing mobile Ad-hoc network protocols. In: Proceedings of the 2011 Uksim 13th international conference on computer modelling and simulation. IEEE, Piscataway, pp 467–471
21. Lee KC, Lee U, Gerla M (2009) Survey of routing protocols in vehicular ad hoc networks. In: Advances in vehicular ad-hoc networks: developments and challenges, IGI Global, October 2009
22. Lee K, Lee U, Gerla M (2009) Survey of routing protocols in vehicular Ad Hoc networks. In: Advances in vehicular Ad-Hoc networks: developments and challenges. Information Science Reference, Hershey, p 22
23. Mauve M, Widmer A, Hartenstein H (2001) A survey on position-based routing in mobile ad hoc networks. Network, IEEE, vol 15, no 6, pp 30–39
24. Akkaya K, Younis M (2005) A survey on routing protocols for wireless sensor networks. Ad Hoc Netw 3(3):325–349
25. Akyildiz IF, Su WSW, Sankarasubramaniam Y, Cayirci E (2002) A survey on sensor networks. IEEE Commun Mag 40(8):102–114
26. Delin KA, Jackson SP (2000) Sensor web for in situ exploration of gaseous biosignatures. In: 2000 IEEE aerospace conference proceedings (Cat. No.00TH8484), vol 7. IEEE, Piscataway
27. Baldus H, Klabunde K, Müsch G (2004) Reliable set-up of medical body-sensor networks. In: Wireless sensor networks LNCS 2920. Springer, Heidelberg, pp 353–363
28. Bonnet P, Gehrke J, Seshadri P (2000) Querying the physical world. IEEE Pers Commun 7(5):10–15
29. Schwiebert L, Weinmann J (2001) Research challenges in wireless networks of biomedical sensors. In: Proceedings of the 7th annual international conference on mobile computing and networking – MobiCom’01. ACM, New York, pp 151–165
30. Kidd C, Orr R, Abowd G, Atkeson C, Essa I, MacIntyre B, Mynatt E, Starner T, Newstetter W (1999) The aware home: a living laboratory for ubiquitous computing research. In: Streitz N, Siegel J, Hartkopf V, Konomi S (eds) Cooperative buildings. Integrating information, organizations, and architecture, vol 1670. Springer, Berlin/Heidelberg, pp 191–198
31. Liu J (2006) On a self-organizing multipath routing protocol in mobile wireless networks. J Netw Syst Manag 14(1):103–126
32. Ding R, Yang L (2010) A reactive geographic routing protocol for wireless sensor networks. In: Proceedings of the 2010 sixth international conference on intelligent sensors, sensor networks and information processing. IEEE, Piscataway, pp 31–36
33. Mainwaring A, Polastre J, Szewczyk R, Culler D, Anderson J (2002) Wireless sensor networks for habitat monitoring. In: WSNA’02: proceedings of the 1st ACM international workshop on wireless sensor networks and applications. ACM, New York, pp 88–97

34. Alsalih W, Akl S, Hassanein H (2007) Placement of multiple mobile base stations in wireless sensor networks. In: Proceedings of the IEEE international symposium on signal processing and information technology. IEEE, Piscataway, pp 229–233
35. Azad AP, Chockalingam A (2006) Mobile base stations placement and energy aware routing in wireless sensor networks. In: Proceedings of the IEEE wireless communications and networking conference, vol 1. IEEE, Piscataway, pp 264–269
36. Ismail Z, Hassan R (2011) Effects of packet size on AODV routing protocol implementation in homogeneous and heterogeneous MANET. In: Proceedings of the 2011 third international conference on computational intelligence, modelling and simulation. IEEE, Piscataway, pp 351–356
37. Malany AB, Dhulipala VRS, Chandrasekaran RM (2009) Throughput and delay comparison of MANET routing protocols. *Int J Open Probl Comput Math* 2(3):8
38. Westphal CWC (2006) Opportunistic routing in dynamic Ad Hoc networks: the OPRAH protocol. In: Proceedings of the 2006 IEEE international conference on mobile Ad-hoc and sensor systems IEEE. IEEE, Piscataway

Wireless Sensors in Heterogeneous Networked  
Systems

Configuration and Operation Middleware

Cecilio, J.; Furtado, P.

2014, XVIII, 143 p. 66 illus., 15 illus. in color.,

ISBN: 978-3-319-09280-5