Power Management in Mobile and Pervasive Computing Systems

G. Anastasi[•], M. Conti[*], A. Passarella[•]

[•]Univ. of Pisa, Dept. of Information Engineering
Via Diotisalvi 2 - 56122 Pisa, Italy
{g.anastasi, a.passarella}@iet.unipi.it

[*]CNR-IIT Institute
Via G. Moruzzi, 1 - 56124 PISA, Italy
marco.conti@iit.cnr.it

1. Introduction

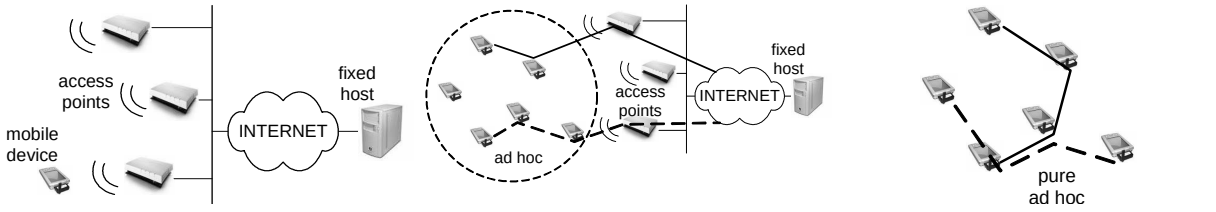
The proliferation of mobile computing and communication devices is producing a revolutionary change in our information society. Laptops, smart-phones and PDAs, equipped with wireless technologies, support users in accomplishing their tasks, accessing information, or communicating with other users anytime, anywhere. Projections show that in few years the number of mobile connections and the number of shipments of mobile terminals will grow yet by another 20-50 percent [WWRF]. With this trend, we can expect the total number of mobile Internet users soon to exceed that of the fixed-line Internet users.

Currently, most of the connections among mobile devices and the Internet occur over fixed infrastructure-based networks, which are extended with a wireless last hop. For example, cellular networks (GSM, GPRS or UMTS) provide a wireless link between mobile phones and a base station, while laptops connect to the Internet via Wi-Fi Access Points (i.e., inside Wi-Fi hotspots). This networking paradigm is throughout referred to as *infrastructure-based* or *single-hop hotspot scenario* (see Figure 1(a)). On the other hand, in the near future the networking environments for mobile devices are expected to include *multi-hop ad hoc networks* (Figure 1(b,c)). In the ad hoc networking paradigm, a group of mobile devices self-organize to create a network, without the need of any pre-deployed infrastructure. Applications running on those devices are able to communicate as in the standard Internet paradigm. For that reason, the most straightforward application of the ad hoc paradigm is using ad hoc technologies to extend Wi-Fi hotspots. For example, in Figure 1(b), mobile devices inside a hotspot build an ad hoc network. Devices that are not in the Access Points' transmission range can nevertheless access the Internet by relying on the ad hoc network. Finally, the ad hoc technology is very likely to fuel the development of new network applications, explicitly designed for pure infrastructure-less environments (Figure 1(c)). The networking paradigms depicted in Figure 1 will be an important

step towards the pervasive computing scenario envisaged by Mark Weiser [W91]. In such a scenario the environment around the users will be saturated with devices which will wirelessly interact among them, and with the users, to automatically adapt the environment to the users' needs (e.g., smart houses, smart offices, …) [IG00,ECPS02].

Many technical problems have to be fixed for this scenario becoming a reality. Among them, one of the most critical is *power management at mobile devices*. To allow users' mobility, devices must be battery-supplied. It is common experience that current mobile devices (laptops, PDAs, etc.) can operate just for few hours before the battery gets exhausted. Even worse, the difference between power requirements of electronic components and battery capacities is expected to increase in the near future [S03]. In a nutshell, power management for mobile devices is a must for the development of mobile and pervasive computing scenarios, and each (hardware or software) component of a mobile device should be designed to be energy efficient. The networking subsystem is one of the critical components from the power management standpoint, as it accounts for a significant fraction of the total power consumption (around 10% for laptops, and up to 50% for small hand-held devices, such as PDAs).

This chapter provides an up-to-date state of the art of power management techniques for mobile and pervasive computing environments, and highlights some open issues. Specifically, Section 2 provides an overview of the main power management approaches that have been proposed for such environments and introduces the research trends that will be analyzed in detail in the rest of the chapter.



(a) single-hop hotspot scenario (b) ad hoc improved hotspot scenario (c) pure multi-hop ad hoc scenario

Figure 1. Mobile and Pervasive Networking scenarios.

2. Research trends for mobile and pervasive computing

The limited resources of mobile devices (e.g., limited CPU and memory) constitute the driving force of mobile and pervasive computing research [FZ94]. The energy available in a portable device is probably the most critical resource. In principle, either increasing the battery capacity, or reducing the power consumption, could alleviate mobile-device energy-related problems. However, as shown in Section 3, projections on progresses in battery technologies show that

only small improvements in the battery capacity are expected in next future [S03, DWBP01]. If the battery capacity cannot be improved significantly, it is vital to manage power utilization efficiently, by identifying ways to use less energy preferably with no impact on the applications.

In Section 4, we provide a general framework for power management in a mobile device. These strategies can operate at different layers of the mobile-system architecture, including hardware, operating system, network protocols and applications. At the operating system level techniques for hard-disk management [HLS96], CPU scheduling [LS97] and screen blanking [LS97] have been proposed. Techniques implemented at the application level include disconnected operations [S93], remote task execution [FPS02], use of agents [PS98], and the exploitation of the application semantic[1] [J00]. Techniques for adapting the application behavior to the changing level of energy also fall in this category [FS99].

Among the components of a mobile device that contribute to power consumption (CPU, video, hard-disk, network interface, etc), the impact of the network interface becomes more and more relevant as the mobile-device size decreases. In a laptop the percentage of power drained by the wireless interface is about 10% of the overall system consumption [KK00, ACL03]. This percentage grows up to 50% when we consider small-size mobile devices like a PDA [KK00, ANF03]. This difference can be justified if we consider that small-size portable computers frequently have no hard disk and limited computational resources. On the other hand, the wireless interface provides almost the same functionalities in a laptop or a desktop PC. It is thus extremely important to design power-efficient network protocols and applications.

In principle there are several approaches to reduce the power consumed by the network interface. In Section 5 we survey the most relevant power management policies for reducing the network-interface power consumption in the infrastructure-based scenario (Figure 1(a)). These techniques are presented in great detail, since the vast majority of today mobile computing environments fall in this category.

Power management schemes may be implemented at different layers of the network architecture [JSAC01]. Some researchers have focused on the impact of the transmission layer errors on the power consumption [RB96, ZR96]. When the bit error rate of the wireless channel is too high a transmitted message will be almost certainly corrupted, and this will cause power wastage. In this case, it is wise to defer the transmission. Other works suggest limiting the number of transmissions, as transmissions needs more power than receptions. The above strategies apply

[1] For example, the application compresses the data before exchanging them, or finds some tradeoffs between performance and power consumption.

well in a cellular environment where the power consumption is asymmetric (the power consumed in the receiving mode is significantly less than in the transmitting mode). However, they are not suitable for a WLAN environment (i.e., Wi-Fi hotspot) where the network interface consumes approximately the same power in transmit, receive, and idle status [SK97, F04]. In this case the only effective approach to power saving consists in switching off the network interface (or, alternately, putting it in a sleep mode) when it is not strictly necessary. In addition, data-transfer phases should be as short as possible in order to reduce the time interval during which the wireless interface must be active (the ideal case is when data transfers are done at the maximum throughput allowed by the wireless link). The effectiveness of this approach was pointed out in several research works [SK97, KK98, ACL03].

Single-hop wireless Internet access represents the first step in the direction of pervasive computing [W91]. However, in a pervasive computing environment, the infrastructure-based wireless communication model is often not adequate as it takes time to set up the infrastructure network, while the costs associated with installing infrastructure can be quite high. In addition, there are situations where the network infrastructure is not available, cannot be installed, or cannot be installed in time. The answer to these problems is represented by the *ad hoc networking paradigm*, in which mobile devices build a multi-hop network, without relying on any pre-deployed infrastructure. Providing the needed connectivity and network services in these situations requires new networking solutions [C04]. In Section 6 we discuss novel power management issues generated by wireless networks developed for pervasive computing environments.

To summarize, in this chapter we survey the solutions proposed to minimize power consumption in mobile devices at different levels of the system architecture. Special emphasis is devoted to techniques aimed at reducing the power consumption in networking activities. We focus primarily on techniques for infrastructure-based wireless networks. However, we also consider power management solutions conceived for infrastructure-less network technologies (ad hoc and sensor networks) that will have an important role for providing mobile and wireless communications in pervasive computing environments.

3. Battery technology

The spectrum of mobile computing devices is quite large. They range from laptops to sensor nodes that have a size of few tens of mm^3 [DWBP01]. Hereafter, we divide mobile devices in three classes: i) high-end devices (such as laptops), ii) medium-size devices (such as PDAs and smart phones), and iii) small-size devices (such as sensors).

Power requirements highly depend on the device type, and on the task the device is designed for. For example, laptops require rechargeable, high-capacity batteries, but dimension and weight are not primary concerns. On the other hand, sensor nodes are typically not reachable after the sensor network is deployed (e.g., they may monitor radioactive or polluted zones, they may be put below buildings to monitor seismic waves, etc.). Therefore, sensor nodes are designed to work unattended until the battery is completely exhausted. Thus, batteries for sensor nodes are typically not rechargeable, and must be very small [PSS01]. To summarize, due to the high variety of devices, the spectrum of battery technologies used in mobile devices is wide.

Today, the most widely used technology is Lithium-Ion Cells [PSS01, DWBP01]. These batteries are available in many different form factors. Moreover, they can be restored at nearly full capacity for many recharge cycles. At the state of the art, lithium batteries are used as an off-the-shelf component, and can be found in devices of almost any class. For example, they are used in:

- **laptops:** capacity around 4000 mAh, volume around 200 cm^3 [LBS];
- **PDAs:** capacity around 900 mAh, volume around 50 cm^3 [IPAQ03];
- **sensors:** capacity around 500 mAh, volume around 10 cm^3 [CMD, PSS01].

The trade-off between capacity and size guarantees a reasonable lifetime to devices such as laptops, PDAs or smart phones. However, for wearable computers and sensor networks, form factors of commercial batteries might not be sufficient. Current research efforts are devoted to miniaturize sensor nodes and wearable devices as much as possible. For example, the SmartDust project of the University of California at Berkeley has developed prototype sensor nodes as small as few tens of mm^3 [DWBP01]. In this scenario, borderline technologies are currently investigated, in order to provide sufficient energy in so small form factors. Not only technologies are investigated aimed at improving the energy density, but also solutions are proposed where devices scavenge energy from the environment around them, without being equipped with traditional batteries. Some examples are highlighted in the following:

- vanadium and molybdenum oxide batteries [HDD98]: this technology allows to produce thin films with capacities comparable to standard lithium cells;
- ultra-capacitors [MT]: these components store less capacity that lithium cells, but can provide high power, due to lower source impedance;
- solar cells [DWBP01]: this technology exploits the well-known photovoltaic effect to produce energy. Therefore, it can be utilized only when some source of lighting is available. Moreover, the surface needed and the rate at which energy is produced may be severe limiting factors [S03];

- vibrations [MMA+01]: ambient vibrations can be converted in electrical energy by means of MEMS (micro-electromechanical systems). Experimental prototypes have shown to produce between 5 and 8 μW with a chip of about 6 mm^2, resulting in performance similar to solar cells (around 1 μW, [DWBP01]);
- more exotic systems have been proposed based, for example, on exploiting heat from micro-rocket engines [TMCP01].

Despite this spectrum of activities, researchers in the mobile networking area agree that battery capacities are today one of the main limiting factors to the growth of this computing paradigm [ECPS02, PSS01, CR00]. More important, in the near future the difference between battery capacities and energetic requirements is likely to become even deeper. [S03] collects the main performance characteristics of several components of a mobile device from 1990 up to 2000, pointing out the improvements achieved in this period by each component, see Figure 2. For easy of presentation, improvements are expressed as multiples of the performance in the year 1990 (please note the log scale on the y axis).

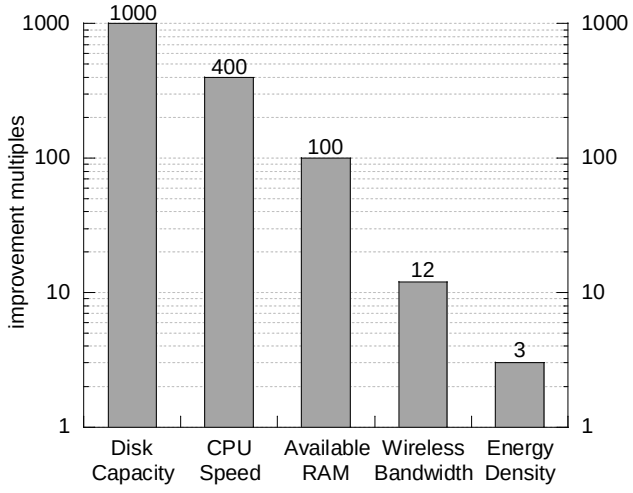


Figure 2. Increase of components' capabilities from 1990 to 2000 [S03].

It is worth noting that battery improvements are far slower than electronic-component improvements. In addition, the increase of components' performance usually produces an increase in energy requirements. For example, the power, *P*, drained by a processor is proportional to the frequency of its clock (i.e., $P \propto C \cdot V^2 \cdot f$, where *C* is the characteristic capacitance of the transistor technology, *V* is the supply voltage, and *f* is the clock frequency[2]).

[2] Hereafter, the symbol ∝ means "proportional to".

To summarize, Figure 2 indicates that a *wise power management is a key enabling factor for the deployment of mobile networks.*

To design effective power management policies, some batteries' properties must be taken into account. The most important are: the *Rate Capacity Effect* and the *Relaxation Effect* [PCD+01, CR00]. The energy that a battery can provide depends not only on its capacity, but also on the profile of the current that it supplies (throughout referred to as I_d). If I_d is constant and greater than a certain threshold (referred to as the *rated current* of the battery), the energetic efficiency can be dramatically low. Specifically, it is well known that the voltage of a battery drops down while it is supplying current. Eventually, a *cut-off voltage* is reached, after which the battery is no longer able to provide energy. It has been shown that, if I_d is constant over time, the more I_d grows beyond the rated current, the less energy is provided before reaching the cut-off voltage [PCD+01, PSS01, DN97] (i.e., the lower is the energetic efficiency). This phenomenon is known as the *rate capacity effect* [PCD+01]. A battery is composed by an anode, a cathode, and an electrolyte. According to the diffusion phenomenon, positive ions flow from the anode to the cathode through the electrolyte, thus balancing the electrons flow on the (outer) circuit supplied by the battery. Specifically, during discharge, positive ions are produced by oxidation at the anode. Ions travel through the electrolyte and arrive at the cathode surface. The cathode contains reactions sites, where positive ions undergo reduction, becoming inactive. At low discharge currents, inactive sites are distributed almost uniformly in the cathode. But, at high discharge currents, inactive sites become concentrated at the outer surface of the cathode, making internal reduction sites inaccessible. In this case, the battery is declared discharged but many reaction sites at the cathode have not been used yet. Furthermore, unavailability of positive ions at the cathode surface contributes to reduce the battery efficiency. In detail, during discharge, ions that undergo reduction at the cathode are replaced by ions travelling through the electrolyte. At low discharge currents, the concentration of ions at the anode-electrolyte interface, and at the cathode-electrolyte interface, is almost uniform. On the other hand, at high discharge currents, the diffusion process in the electrolyte becomes unable to transport enough positive ions at the cathode. As a result, the concentration of positive ions in the electrolyte becomes higher at the anode than at the cathode, and hence the battery voltage decreases. To face this problem, many authors propose to drain current in *pulsed mode*. That way, much higher energetic efficiency can be achieved [PCS02, PCD+01, PSS01]. Specifically, a (constant) current I_d is drained for a time interval t_{load}, then the battery is left idle for a time interval t_{idle}, and so on. During t_{idle} uniform concentration of positive ions at the anode and cathode is restored, and hence the battery recovers the voltage lost during the previous t_{load}. This phenomenon is known as the *relaxation*

effect. By exploiting this effect, if a pulsed scheme is used, the energy provided before reaching the cut-off voltage is higher than in the case of continuous drain. Obviously, the performance improvements depend on the battery capacity, and on the ratio between t_{load} and t_{idle} [PCS02].

To summarize, for the deployment of mobile computing environments, smart power managers are required as i) improvements in battery capacity are significantly lower than the increase of power requirements of electronic components, and ii) the capacity of batteries can be fully exploited only by properly interleaving idle and active phases.

4. Power Management Approaches for Mobile Devices

At a very high level of abstraction, a computer can be seen as a set of hardware and software components servicing user requests. Hardware components need energy to operate. Instead, software components do not consume power by themselves, but they impact on the power consumption as they control the activity of hardware components.

Great efforts have been devoted to design and manage hardware components in a power-efficient way. The most widespread approach consists in focusing on a particular component (e.g., the hard disk, the network interface, ...), and deploying a *dedicated power manager* guaranteeing the component provides the required services, while optimizing the power spent to provide them. For example, laptop screens are usually blanked after a pre-specified inactivity interval, and are resumed when new activity is detected (e.g., a mouse movement or a key press). That is, a power manager is "attached" to the screen, monitoring its activity, and putting it in a low-power operation mode when it is not used for a while. In principle, power managers can be deployed either in hardware or in software. However, due to their inherent flexibility, software managers are usually preferred. It must be finally pointed out that – in general – care should be taken in the additional power consumption of the manager itself.

Hereafter, we survey the power management approaches presented in the literature. To the purpose of this presentation, in Section 4.1 we introduce the breakdowns of the power consumption of hardware components in typical mobile devices.

4.1 Power budget of typical mobile devices

In the past years, several studies have highlighted the power drained by various components of laptops [LS98, US96]. Specifically, [US96] shows that the CPU, the wireless network interface, the hard disk and the display require approximately 21%, 18%, 18% and 36% of the total power, respectively. Similar results are also presented in [LS98]. More recently, several works confirm the large impact on the overall system consumption due to the wireless interface [ACL03, KK98], CPU [CBB+02, PS01] and hard disks [BBD00].

The power budget of devices such as PDAs is shown in [SBGD01]. Usually, such systems do not have hard disks, and use flash memories to store persistent data. The wireless network interface consumes about 50% of the total power, while the display and the CPU consume about 30% and 10%, respectively. The great impact of the network interface was previously highlighted also in [KK98].

Finally, the power budget of typical sensor nodes is highlighted in [DWBP01, RSPS02, ACF+04]. This type of computing devices drastically differ from both laptops and PDAs, and hence also the power breakdown is different. Furthermore, the power consumption may greatly depend on the specific task implemented by the sensor network [DWBP01]. Typically, the wireless network interface and the sensing subsystem are the most power-hungry components of a sensor node. On the contrary, the CPU accounts for a minor percentage of the power consumption.

To summarize, several hardware components of a computing device need power managers. In the past years, many researchers have focused mainly on hard disks and CPUs, as they account for a great portion of the total power consumption in laptops. More recently, increasing interest has been devoted to the networking subsystem, as it is the main source of power consumption for medium and small-size mobile devices (e.g., PDA). In the following of this section we survey power-management approaches that apply to different hardware subsystems. Then, in Sections 5 and 6, power-management approaches tailored to the networking subsystem will be discussed in depth.

4.2 General scheme for HW Components Power Management

[BBD99] presents a very useful scheme to understand how power management works, see Figure 3. To better understand Figure 3, let us define the concept of power-manageable component (PMC). A PMC is a hardware component that provides a well-defined kind of service. For example, hard disks, network interfaces, CPUs can be seen as power-manageable components. A fundamental requirement of a PMC is the availability of different *operating modes*. Each mode is characterized by a specific power consumption and performance level. For example, a hard disk has high power consumption when disks are spinning, and data can be accessed very quickly. Instead, when disk heads are parked, the power consumption is much lower, but the access time becomes very high, since disks must be spun up again before accessing data.

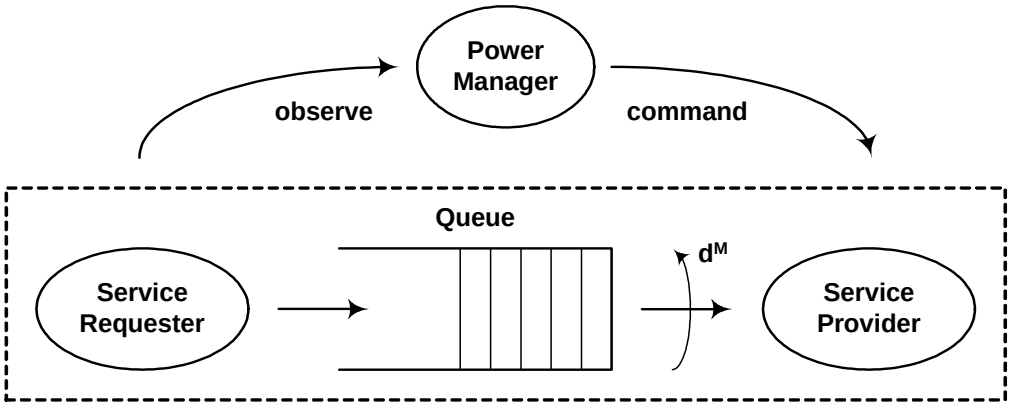


Figure 3. High-level system model [BBD99].

In Figure 3, a power-manageable component is represented as a *Service Provider* (SP). Each item in the *Queue* (Q) is a request for an SP service. A *Service Requester* (SR) --representing, in an aggregate way, the SP users-- puts items in the Queue. The *Power Manager* (PM) *observes* the status of the Service Requester, Queue and Service Provider (that is, the status of the system), and issues *commands* to the Service Provider, dynamically switching it among its operating modes. The algorithm used by the Power Manager to decide the Service Provider operating modes is referred to as *policy*. The goal of a policy is to make the Service Provider handle items in the Queue by: i) spending the minimum possible energy, and ii) still providing an acceptable Quality of Service (QoS) to users. It must be pointed out that transitions between different SP modes usually have a cost. Specifically, each transition requires a time interval during which the Service Provider consumes energy but is not able to service any request. Furthermore, the more the power consumption of the low-power mode decreases, the more the time interval to switch to a high-performance mode increases. For each low-power mode *M*, a *break-even time* (T_{BE}^{M}) can be defined. When the Service Provider is put in low-power mode *M*, it should not switch back to the high-power mode before T_{BE}^{M} seconds. Otherwise, the energetic advantage of using the low-power mode is overwhelmed by the transition cost [BBD00]. Hence, the Service Provider can be modelled as a server with vacations. During vacation periods, the Service Provider operates in low-power modes and it is not able to service requests in the Queue. The vacation lengths depend on the particular mode the Service Provider is operating in, and on the policy implemented by the Power Manager. Specifically, for each power mode *M*, T_{BE}^{M} represents the minimum vacation length. The Power Manager may increase the vacation length, based on the observed system status. Therefore, if d^M denotes the vacation length related to the operating mode *M*, we can express d^M as follows:

$$d^M = T_{BE}^M + f_d(policy) \ . \quad (1)$$

where $f_d(\ )$ denotes the delay introduced by the selected power management policy.

It is worth noting that a typical drawback of any power management policy is a negative impact on the QoS perceived by SP users. With reference to Equation (1), energy saving increases with the increase of d^M. However, d^M also impacts on the delay introduced by power management to service times, thus decreasing the QoS. For example, when a hard disk is spun down, the user must wait for a long time at the first access. Therefore, the Power-Manager design must trade-off between these contrasting requirements.

In the original model [BBD99], the system represented in Figure 3 is completely isolated, i.e., the behavior of SR, SP and PM neither affects, nor is affected, by any other computer component. Though this model has been exploited to analyze several power-saving strategies [BBD00, BD02], it is worth enriching it, to achieve a more complete characterization of power management for mobile devices. Specifically,

- the Power Manager could observe not only the status of the Service Requester, Queue and Service Provider, but also the status of other computer components (i.e., interactions between different computer components could be taken into consideration).
- the Power Manager could manage not only the Service Provider, but also the Service Requester and Queue. That is, not only the SP *behavior* can be controlled based on the requests performed by the Service Requester, but the *workload* generated by the Service Requester could be shaped in order to reduce the power consumption of the Service Provider.

As a final remark, it must be noted that Power Managers considered in this chapter are sometimes referred to as *external* power managers. These power managers are used with Service Providers that allow an external controller to put them in different operating modes. External Power Managers are usually designed separately from Service Providers, and take into account the status of the entire system (at least, SR, Q and SP) to save energy. On the other hand, *internal* power managers switch Service Providers between different operating modes based only on the internal state of the Service Provider itself. They are usually implemented in hardware, and are designed together with the Service Provider (i.e., they are part of the SP itself). Note that operating modes used by internal and external Power Managers differ. For example, hard disks by IBM define an idle mode to be used by external Power Managers; this idle mode is actually a wrapper for three possible idle modes, which can be set only by an internal Power Manager, implemented in the hard-disk circuitry [BBD00]. Using internal Power Managers is one of the

power-saving methodologies that are used in the chip design of Service Providers. For the sake of space, hereafter we omit considering this kind of techniques (the interested reader can refer to [BD02]).

4.2.1 Overview of Power Management Strategies

The core of any power management system is defining the behavior of Power Managers. Research efforts usually follow two complementary approaches. With reference to Figure 3, the first approach focuses on dynamically driving the Service Provider in the various operating modes, based on the knowledge of the system status (i.e., the SP, SR and Q status). This approach is usually referred to as *Dynamic Power Management* (DPM). Solutions of this type range from simple timeout policies, to policies based on dynamically estimating the profile of requests issued by the Server Requester. These solutions do not modify the profile of SR requests (in order to optimize the power consumption of the Service Provider), but they dynamically react to the requests profile by selecting the appropriate mode of the Service Provider. From this standpoint, this approach can be seen as a *reactive approach*. Reactive policies are surveyed in Section 4.3.

A second approach focuses on modifying the workload issued to the Service Provider, i.e., it modifies the profile of SR requests. Policies based on this approach are aware of the impact that the workload shape has on the SP power consumption, and hence modify the workload to optimize the performance. This approach can be seen as a *proactive approach*. For example, some compilers modify the execution order of instructions to reduce transitions between 0s and 1s on the buses' lines. Obviously, it must be assured that the Service Provider provides the same results after processing either the original or the modified workload.

As the reactive and proactive approaches are orthogonal, some solutions exploit a combination of them. Hereafter, we refer to these solutions as *mixed policies*. Mixed policies implement algorithms that dynamically select the Service Provider operating mode, based on the workload shape. If possible, they also modify the workload in such a way that low-power operating modes could be exploited more efficiently than with the original workload. Section 4.4 is devoted proactive and mixed policies.

4.3 Reactive Policies

Reactive policies dynamically drive the SP behavior by deciding on-line the best operating mode for the Service Provider, given the current status of the mobile device. Before analyzing the different types of reactive policies, it is worth spending few words on how different operating modes are deployed.

4.3.1 Dynamic Voltage and Frequency Scaling

Dynamic Voltage Scaling (DVS) and Dynamic Frequency Scaling (DFS) are techniques used to implement low-power operating modes, and hence they are the enabling technologies for dynamic power managers. Moreover, some Power Managers explicitly scale voltage and frequency as a power-saving technique.

The motivation behind DFS is that the power consumption of an electronic component is proportional to $V^2 \cdot f$, where V is the supply voltage and f is the clock frequency. Many researchers highlight that DFS alone may not be sufficient to save energy. Specifically, the time needed to perform a given task is proportional to $1/f$, making the energy spent independent on the operating frequency ($E \propto V^2$). However, [PS02] shows that in some cases just arranging the operating frequency allows saving energy. Specifically, [PS02] focuses on dedicated CPUs executing real-time event handlers periodically. If a handler execution finishes before its deadline, the CPU remains idle till the new execution. Hence the energy spent can be expressed as $E = E_h + E_{idle}$, where E_h is the energy spent to execute the handler and E_{idle} is the energy spent when the CPU is idle. In this case, if the frequency is reduced in such a way that the handler execution finishes immediately before the next execution, the energy spent becomes equal to E_h.

The best performance, in terms of power saving, is achieved when DVS and DFS are used together. When the CMOS technology is used, clock frequency and supply voltage are strictly tied. Specifically, the supply voltage reduces by decreasing the transistors operating frequency [BB95]. Since the energy depends quadratically on V, the energy saving achieved in this way can be quite large [LBD02, PS01].

The most aggressive forms of DFS and DVS are achieved by completely freezing the clock (i.e., $f = 0$), or powering the circuitry down (i.e., $V = 0$). Usually, not the entire SP is frozen or powered down, but only some subsystems. The different operating modes of Service Providers are typically implemented in this way. For example, 802.11 WLAN cards use a sleep mode, where only synchronization-related circuitries are active, while the rest of the card is powered down. It must be pointed out that clock freezing achieves lower energy saving than powering down, because transistors still drain a static current, known as leakage current. On the other hand, transitions times related to clock freezing are typically negligible, while powering up subsystems may take as long as few seconds, depending on the particular Service Provider [BBD99].

4.3.2 Taxonomy and Examples of Reactive Policies

The research on Reactive Policies is a very hot topic. A possible taxonomy of proposed policies is presented in Figure 4 [BBD00]. We can distinguish between predictive and stochastic policies. Predictive policies estimate the (short-term) behavior of the Service Requester, and exploit this information to decide whether it is convenient for the Service Provider to go into low-power states. As shown in Figure 4, predictive policies can be further subdivided in several classes. On the other hand, stochastic policies are not based on estimates of SR future behavior. Instead, SP and SR are modelled by means of Finite State Machines (FSM), and the power-management policy defines the transitions between the states of the SP machine (see below). In the following of this section we describe the main features of each power-management class shown in Figure 4.

4.3.3 Predictive Policies

The fundamental assumption behind these policies is that some correlation exists between the past history of SR requests and the near-future requests. This correlation is exploited to predict the near-future shape of the workload, and hence to decide the operating mode of the SP. Without loss of generality, to understand how policies of each class operate, we can consider a Service Provider having only two operating modes, a high-power high-performance operating mode, and a low-power low-performance operating mode[3].

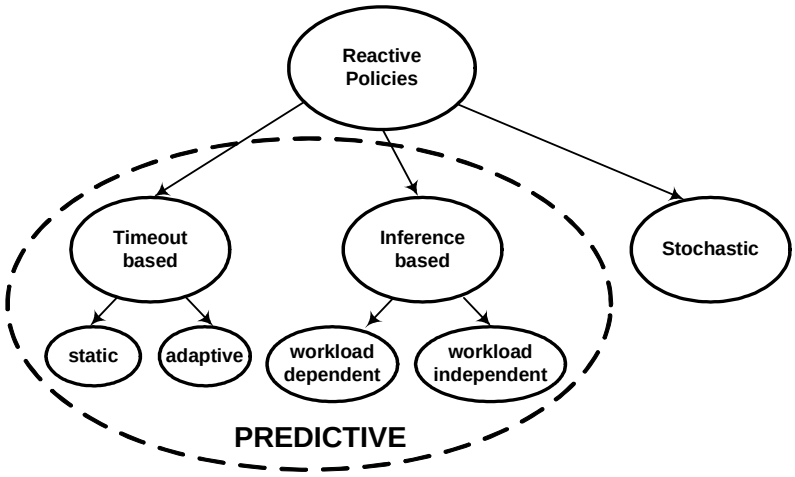


Figure 4. Taxonomy of Reactive Policies.

TIMEOUT-BASED POLICIES. This is the simplest type of reactive policies. Timeouts are typically used to decide when the Service Provider must switch to the low-power operating mode. Specifically, if T_{TO} is the timeout value, the Service Provider is forced in the low-power mode

[3] This is the most common case in the real world, and the discussion can be easily extended to include more general scenarios.

after a T_{TO} time interval since the last served request. An example of such techniques is the policy used by several operating systems to blank the screen.

This approach is effective only if the following equation holds

$$p\left(T > T_{TO} + T_{BE} \middle| T \geq T_{TO}\right) \approx 1 \qquad (2)$$

where T denotes the length of an idle phase, i.e., the time during which the Queue remains empty. As said before, after the transition to a low-power operating mode is decided, the SP should not switch back to the high-power mode before T_{BE} seconds, in order to save energy. Therefore, the probability of having an idle phase longer than $T_{TO} + T_{BE}$, provided that it is longer than T_{TO}, must be close to one; otherwise, this approach may lead to a greater power consumption than letting the Service Provider always in the high-power mode. The main concern of timeout policies is the choice of the T_{TO} value. Typically, by increasing the timeout value T_{TO}, it increases the probability in the left hand-side of equation (2). On the other hand, the Service Provider remains in the high-power mode for T_{TO} seconds after the last request, and this significantly increases power consumption.

Several works use *fixed* timeout values, and propose algorithms for setting the appropriate T_{TO} value. These solutions can be seen as *static timeout-based* policies, as they do not change the timeout value over time. Static-timeout policies have been the first class of reactive policies studied in the literature, and have been mainly applied to hard disks [DKM94, GBS+95, KMMO94]. More recently, [KK00] proposed to use timeout-based policies to manage the wireless network interface (see Section 5 for details). Today, the vast majority of policies implemented in commercial operating systems belong to this class. However, several works have shown that better performances can be achieved if the T_{TO} value is dynamically adjusted, based on the past history of idle phases. These policies are referred to as *adaptive timeout-based* policies. For example, [KLV95] maintains a list of possible timeouts, together with a list of "scores" showing how well each timeout would have performed in the past history. When an idle phase starts, the timeout with the highest score is chosen. Similarly, [HLS96] uses a list of candidate timeouts and a list of corresponding scores. When an idle phase occurs, the timeout value is evaluated as the weighted sum of the candidate timeouts, using the scores as weights. Finally, [DKB95] records the number of transitions to low-power operating modes that have been occurred in the recent past. When this number becomes too high, it means that the T_{TO} value is too short with respect to the inter-arrival times between requests (i.e., new requests

arrive very soon after the SP has been put into low-power operating mode). In this case, the T_{TO} value is increased. On the other hand, the T_{TO} value is decreased when the number of transitions into low-power operating modes becomes too low (i.e., the SP is almost always in the high-power operating mode).

INFERENCE-BASED POLICIES. Two main drawbacks of timeout policies have been highlighted in the literature [BBD00]: i) they waste power waiting for the timeout to expire; and ii) they always pay a performance penalty when a new request arrives and the Service Provider is operating in low-power mode. To overcome these limitations, several works try to estimate the *length* of the next idle phase, and manage the Service Provider accordingly. Once an estimate is available, the Service Provider is forced in the low-power mode if the estimate is greater than T_{BE}. Then, at the expiration of the estimate, it is resumed to manage the next request. [SCB96] proposes two methods for predicting the length of an idle phase, both based on past idle and active phases' lengths. The first one uses a non-linear regression, based on the lengths of the last *k* idle and active phases. The form of the regression equation, as well as *k*, is chosen by means of offline data analysis. The second one is based only on the length of the last active phase, say T_{active}. If T_{active} is below a certain threshold, the next idle phase is assumed to be longer than T_{BE}. This policy is tailored to a particular type of workload, i.e., workloads where short active phases are followed by long idle phases. Furthermore, the value of the threshold is chosen using an offline data analysis. These solutions require an a-priori knowledge of the statistical characteristics of the workload, i.e., they are *workload dependent*. As knowledge about the workload could not be available at design time, [HW97] proposes a *workload-independent* policy. In this work, the length of the next idle phase is estimated by means of a smoothed average algorithm, similar to the one used by TCP to estimate the Round Trip Time. Specifically, if $T_{est}^{(n)}$ is the estimate at time *n*, and T_{idle} is the actual length of the last idle phase, T_{est} is updated as $T_{est}^{(n+1)} \leftarrow \alpha \cdot T_{idle} + (1-\alpha) \cdot T_{est}^{(n)}$, where $\alpha \in [0,1]$ is the smoothing factor. [CBBD99] proposes a workload-independent technique, tailored to hard-disk management and based on adaptive learning trees. Workload dependent and independent policies have been proposed and extensively evaluated for managing wireless network interfaces [ACGP03b, ACGP03c] (see Section 5).

4.3.4 Stochastic policies

Stochastic policies model the Service Provider and the Service Requester as Finite State Machines. The Finite State Machine representing the Service Requester models the SR behavior.

Hence, the meaning of its states and transitions between states depends on the specific Service Requester (see below). On the other hand, the SP states usually represent the different operating modes of the device, and transitions between states are driven by the Power-Manager policy. Specifically, if *P* is the set of the SP states, *R* is the set of the SR states, and *C* is the set of the possible commands used by the Power Manager to drive the Service Provider, then the policy is represented by a function $f_p : (P \times R) \rightarrow C$. These models provide an analytical framework to define optimal power-management policies. Furthermore, trade-offs between power saving and performance penalties are included in the framework by representing performance penalties as constraints of the optimization problem. The optimization problem can be solved offline, and provides the *optimal* power-management policy, i.e., the policy that achieves the greatest power saving under the problem constraints [BBD00]. It must be pointed out that the policy optimality relies on the accuracy of the SR and SP models. Hence, when the policy is used to manage real workloads and Service Providers, its performance can be worse than expected [SBGD01].

A first class of stochastic policies uses discrete-time Markov chains to model both the Service Requester and the Service Provider. In this type of model the time is slotted, and the state of the system is sampled at the beginning of each slot. For example, Figure 5 reproduces a scheme of such a model [BBD00]. The Service Requester is modeled as a two-state Markov chain: state 0 means "no request is issued by SR", while state 1 means "a new request is issued by SR". The example in Figure 5 represents a bursty workload: if a new request is issued during the slot *n*, another request will be issued during the slot *n+1* with probability 0.85. Also the Service Provider is represented by a two-state Markov chain: the Service Provider has a low-power operating mode (off state), and a high-power operating mode (on state). Transitions between states are driven by commands issued by the Power Manager (`s_on` means "switch on" and `s_off` means "switch off"). For example, let us focus on the on state, and let us assume that, at the beginning of slot *n*, the Service Provider receives the `s_off` command. Then, at the beginning of slot *n+1*, the Service Provider will still be in the on state with probability 0.8, while it will be in the off state with probability 0.2. These probabilities model the transition times between the SP operating modes. Finally, the Power-Manager policy is an unknown function that, at the beginning of each slot, gives a command to the Service Provider, based on the current SP and SR state. In [BBD00] the optimization problem obtained from the system of Figure 5 is solved under different performance-penalty constraints. Then, the performance-penalty vs. power-consumption curve is plotted, showing that the stochastic approach provides a very useful tool for PM designers.

Figure 5. Example of Markov models of a Service Requester (a) and a Service Provider (b) [BBD00].

[CBB+02] highlights that the analytical approach used in [BBD00] relies on the assumption that the stochastic processes representing both the Service Requester and the Service Provider are known a-priori, and are stationary (i.e., transition probabilities do not change over time). Unfortunately, these assumptions may be completely wrong, especially with respect to the Service Requester. Therefore, [CBB+02] extends the previous model to the case of unknown stationary and non-stationary environments. In the former case, the parameters of the unknown *stationary* Markov model are estimated on-line, and then refined by monitoring the behavior of the real system. The system steady-state assumption guarantees that estimated parameters tend to the real ones. As the optimal policy depends on these parameters, it should be theoretically recomputed at each time slot. Since this approach is practically unfeasible, [CBB+02] computes offline optimal policies for different values of the system parameters, and then uses an online interpolation technique to approximate the best policy. In the case of non-stationary environments, the technique used to estimate system parameters must be changed. To this end, [CBB+02] uses a window-based system, which provides estimates at run-time, based on the evolution of the system in the recent past. The interpolation technique is still used to compute the best policy.

All above policies are time-slotted: at each time slot the system is observed and the Power Manager sends a command to the Service Provider. This can lead to power wastage, as commands might be computed too often [SBGD01]. Several works [SBGD01, QP00, QP99] overcome this drawback by using time-continuous, event-driven models, where commands are computed only when new events occur in the system. Moreover, [SBGD01] highlights that using Markov chains (both in time-slotted and time-continuous models) means modeling transition times between SP and SR states by means of memory-less distributions (i.e., geometric or exponential). However, these assumptions are rarely met in real systems, resulting in poor performances in terms of power saving. To cope with this problem, in [SBGD01] two analytical approaches are proposed. Both are event-driven, and can be used with any distribution of

transition times. The first one is based on the renewal theory and models the system by a finite state machine. Power-Manager commands are computed only when the system is in a particular state of this finite state machine. Under these hypotheses, the optimal policy can still be derived as the solution of an optimization problem. The second approach is based on time-indexed semi-Markov decision processes (TISMDP), and still models the system by means of a finite state machine. It is more complex than the previous one, but can deal with more general systems, where PM commands are computed in different states of the finite state machine. Even in this case, the optimal policy is the solution of an optimization problem. Finally, it should be noted that both approaches require the a-priori knowledge of the distribution of transition times.

4.4 Workload Modifications

Instead of just driving the Service Provider in different operating modes, the Power Manager could also manage the Service Requester to reduce the SP power consumption. These policies can be roughly divided between *proactive policies*, and *mixed policies*. Proactive policies (see, for example, [PCD+01, KVIY01]) do not exploit possible low-power operating modes of the Service Provider. Instead, they use some knowledge about the impact of different workload shapes on power consumption. Based on this information, Power Managers drive the Service Requester and the Queue. On the other hand, mixed policies [ACGP03b, PS01] manage the Service Requester and the Queue in order to better exploit the use of the low-power operating modes of the controlled Service Provider.

4.4.1 Proactive Policies

Proactive policies include power-aware optimizations of software components. These policies can be classified depending on when code optimization is performed: before or after compilation. *Software synthesis* includes all policies in which the source code is transformed before compilation. Some researchers focus on exploiting computational kernels, i.e., code blocks being executed frequently [BGS94, W96]. Kernels are then specialized having in mind power consumption. An example of this technique is procedure inlining, in which function calls are substituted by the body of the functions to eliminate the overhead related to invocations. A similar approach is based on value profiling and partial evaluation. The main idea is looking for function calls with frequent parameter values, and generating specialized versions of these functions for recurring parameter values [CFE99].

Compilers can also introduce power-aware optimizations. Some authors propose algorithms for instructions selection [TMW94] or registers allocation [MOI+97]. Others works reduce the power drained by bus lines (when switching between successive instructions) by instruction

reordering [STD94]. Finally, techniques that optimize the code to reduce the energy spent by memories have also been investigated [KVIY01].

[LS98] highlights that well-known techniques such as caching and prefetching can be seen as proactive policies. [LKHA94] shows that energy saving can be achieved by properly sizing the cache of hard disks. Furthermore, the benefit of prefetching on energy saving can be highlighted by considering filesystems such as Coda [S93].

Proactive policies have also been proposed to optimize the battery access patterns [PCD+01, CR00]. In these papers the SP is the battery, and the main idea is exploiting the relaxation effect (see Section 3) to extend the battery lifetime. [PCD+01] focuses on the networking subsystem of a general operating system. It is shown that very different battery lifetimes are achieved when the activities (and, hence, the battery accesses) required to process incoming and outgoing packets are scheduled in different ways. [CR00] highlights that lithium polymer cells can be shaped to provide a mobile device with an *array* of independent cells of a reasonable size. This opportunity is exploited by dynamically scheduling the cell that supplies the computer. The scheduler lets each cell relaxing for a sufficient amount of time.

4.4.2 Mixed Policies

These policies are often applied to CPU scheduling. The aim is to arrange dynamically tasks in the ready queue in order to maintain the clock frequency and the supply voltage as low as possible (see Section 4.3.1). This methodology has been investigated for the first time by Mark Weiser et al. [WWDS94], and then several researchers have used it. [PS01, PS02, KL00, G01, MACM00, SC00] use this approach for real-time environments. For example, [PS02] focuses on periodic tasks. When an instance of the task must be executed, a set of estimates is generated indicating the execution times at each available frequency. The clock is then set at the minimum frequency meeting the task deadline. These estimates are based on offline measurements and previous executions of the same task. Finally, execution is frozen periodically, the estimates are updated, and the clock frequency is modified, if necessary. On the other hand, [FRM01, LS01] consider non real-time environments, with the aim of maintaining good interactive performance while reducing the CPU power consumption. For example, [LS01] assigns soft-deadlines to non real-time tasks, and schedules the CPU using these deadlines. Soft-deadlines are assigned based on estimations of tasks execution times derived from the history of previously executed (similar) tasks.

Mixed policies have also been proposed for managing wireless network interfaces. [ACGP02, ACGP03b] focus on Web applications, and propose a power saving network architecture (see Section 5 for details). Among others, the goal is to concentrate data transfers in big bursts, in

order to fully exploit the wireless link bandwidth. When a burst is completely transferred, the network interface is shut down. [ACGP03b] shows that modifying the workload leads to about 10% additional energy with respect to a policy that lets the workload unmodified. Finally, policies have been proposed that drastically modify the networking applications in order to change the workload issued to SP. Examples can be found in [J00, PEBI01].

4.5 Discussion

To summarize, let us draw some qualitative comparison among the presented approaches. Timeout-based policies are very simple, and can be easily implemented. However, they usually exhibit poor performance, even if timeouts are dynamically adjusted. Candidate scenarios for such solutions are systems with long inactivity periods, where approximate timeouts are sufficient to detect idle phases.

Inference-based policies are very flexible, since they can work entirely on-line, without any assumption about the statistical characteristics of the Service Requester [ACGP03c]. Moreover, when awareness of the workload shape is available, they can be customized. For example, [ACGP03b] manages a WLAN card in the presence of Web traffic, and allows a power saving of about 90% with practically negligible performance penalties. In a similar scenario, stochastic policies achieve a power saving of approximately 70%, with higher performance penalties [SBGD01]. A drawback of predictive policies is that, in many cases, they are just heuristics, and hence it is hard to fully understand their behavior.

Finally, the main advantage of the stochastic approach is that it allows developing a general analytical framework. Thus, optimal policies can be derived, and a clear understanding of the Power Manager behavior can be achieved. On the other hand, the stochastic approach usually needs quite strong assumptions about the statistical behavior of the real environment. When these conditions are not met, stochastic policies may perform poorly [SBGD01].

Proactive policies are the only way to save power when the Service Provider is not a Power-Manageable Component, i.e., when it cannot operate in different modes. On the other hand, in presence of Power-Manageable Components, mixed policies join the advantages of both reactive and proactive approaches. For example, in [ACGP03b] a mixed policy tailored to the wireless interface is compared with a reactive policy. Both policies exploit low-power modes of the wireless interface during inactivity periods. Moreover, the mixed policy also modifies the workload of the interface by using caching and prefetching techniques. Experimental results show that the mixed policy consumes about half of the power spent by the reactive one.

5. Power Management in Infrastructure-based Networks

This section introduces some power management policies for wireless interfaces in infrastructure-based wireless networks (Figure 1(a)). According to the classification introduced in Section 4.2.1, all power management policies presented below are mainly *reactive* policies. However, some of them include mechanisms (e.g., caching, prefetching, etc.) that modify the original traffic shape in order to reduce power consumption at the mobile device. Therefore, according to the same classification, these techniques can be regarded as *mixed* policies.

A further classification can be based on the layer of the network architecture where these policies operate. From this standpoint, power management policies used in infrastructure-based networks can be divided in *MAC-layer* and *upper-layer* (e.g., transport or application) policies. In addition, as shown in Figure 6, upper-layer policies may be either *local* or *distributed*. Local policies are implemented locally at the mobile device, while distributed policies are implemented partially at the mobile device, and partially at the Access Point (AP). To the best of our knowledge, MAC-layer strategies are always distributed policies. MAC- and upper-layer policies are not necessarily alternative policies. On the other hand, they are often used jointly.

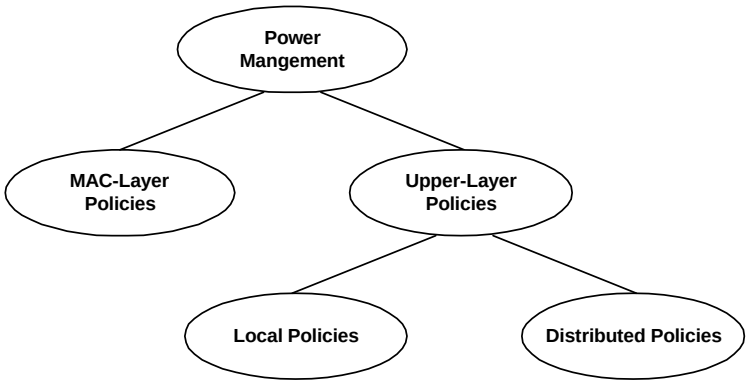


Figure 6. A classification of power-management policies used in infrastructured wireless networks, based on the layer of the network architecture where they operate.

5.1 MAC-Layer Policies

5.1.1 IEEE 802.11 Power Saving Mode (PSM)

Currently, IEEE 802.11 [802.11] (also known as Wi-Fi) is the most mature wireless technology, and Wi-Fi products are largely available on the market. Readers can refer to [C04] for a survey on general aspects of this technology. In this section we focus on power management issues.

IEEE 802.11 wireless cards can operate in two different modes: *active*, and *sleep* (or *doze*) mode. In the active mode, the wireless interface is able to exchange data and can be in one of the following states: *receive*, *transmit* or *idle*. The power consumption in the active mode is very little influenced by the specific operating state and, hence, it is typically assumed as almost

constant (e.g., 750 mW for Enterasys Networks RoamAbout interfaces [KB02]). In the sleep mode, the wireless interface is not able to exchange data but its power consumption is, at least, one order of magnitude lower than in the active mode (e.g., 50 mW for Enterasys Networks RoamAbout interfaces [KB02]).

The IEEE 802.11 standard includes a *Power Saving Mode* (PSM) - operating at the MAC layer - whose objective is to let the wireless interface in the active mode only for the time necessary to exchange data, and to switch it into the sleep mode as soon as there are no more data to exchange. In infrastructure-based 802.11 WLANs (e.g., in Wi-Fi hotspots) power management is achieved by exploiting the central role of the AP. According to the IEEE 802.11 terminology, mobile devices registered with the same AP are said to form a Basic Service Set (BSS). Each mobile device belonging to the BSS has to inform the AP whether or not it is using PSM. The AP buffers data destined for mobile devices that are using PSM. Furthermore, every *Beacon Interval* (usually 100 ms), the AP broadcasts a special frame, called *Beacon*, to mobile devices in the BSS. Mobile devices operating in PSM are synchronized with the AP[4], and wake up periodically to listen to the Beacon. This special frame includes a *Traffic Indication Map* (TIM) that specifies which mobile devices have frames buffered at the AP. Mobile devices indicated in the TIM remains active for the rest of the Beacon Period to receive buffered frames. All the other ones go to sleep until the beginning of the next Beacon Period.

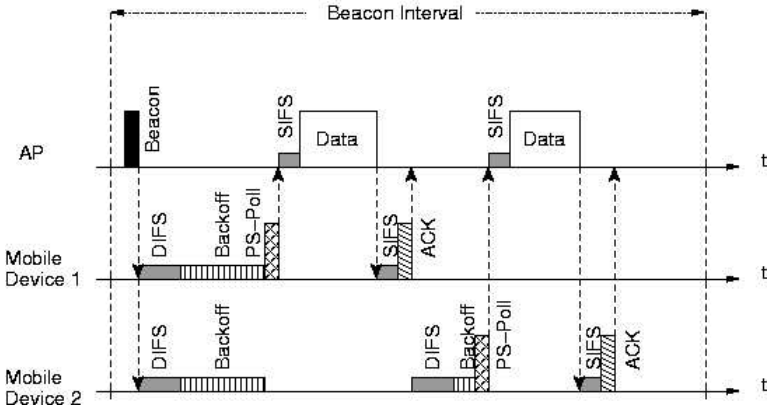


Figure 7. Wi-Fi PSM Operations.

Frames buffered at the AP can be downloaded as shown in Figure 7. The mobile device sends a special frame, *PS-Poll*, to the AP using the standard DCF (Distributed Control Function) procedure. Upon receiving a PS-Poll, the AP sends the first DATA frame to the mobile device and waits for the related ACK frame. If there are still data to be sent, the AP sets the *More Data*

[4] The Beacon frame contains parameters for clock synchronization.

bit in the DATA frame. To download the next DATA frame, the mobile device must send another PS-Poll. When the mobile device realizes that there are no more data to fetch, it puts the wireless interface in the sleep mode. As soon as the mobile device has a data frame to send it switches back to the active mode and performs the standard DCF procedure.

PSM achieves the goal of saving energy especially when the network activity is low. In [KB02, ACGP04a] it is shown that using PSM in a standard TCP/IP architecture allows saving up to 90% of the energy spent by the wireless interface. However, this result refers to a scenario where there is a single mobile device in the BSS. Since PSM relies upon the standard DCF procedure, mobile devices served by the same AP have to contend with each other to access the wireless medium and download buffered messages. Therefore, the PSM ability to save energy depends on the number of mobile devices in the BSS (e.g., in a Wi-Fi hotspot). In [ACGP04a] it is shown that with 10 mobile devices in the BSS the power saving achieved by PSM reduces to 60-70%.

An additional major limitation of PSM is that it implements a static power-management policy, i.e., the Beacon Period remains constant, irrespectively of the network activity. This static behavior may result in significant performance degradations in some specific scenarios [KB02, ANF03], and, in some cases, may even increase the overall power consumption [ANF03]. Specifically, the effects described below have been observed. In all cases it is assumed that the AP is serving a single mobile device. Things are expected to be even worse when several mobile devices operate under the same AP.

- **Increase in the Round Trip Time (RTT) of a TCP connection.** When using PSM the wireless interface is put in the sleep mode whenever the mobile device is not sending or receiving data. If the mobile device has a data to send (e.g., a TCP SYN, or an ACK segment, or a TCP segment containing a Web request, etc.) it immediately wakes up the wireless interface. However, the wireless interface is switched back to the sleep mode as soon as data have been transmitted. When the response data arrive from the server, they must be buffered at the AP until the next Beacon occurs. This delay increases the RTT of the connection. Specifically, the first RTT is increased by a value that depends on the time instant at which the TCP SYN segment was sent by the mobile device[5]. Subsequent initial RTTs are *rounded up* to the nearest 100 ms (even if the actual RTT is 5 ms the observed RTT is in the order of 100 ms) [KB02]. This can be explained as follows. Since TCP segments addressed to the mobile device are transmitted at the beginning of the next Beacon Period,

[5] For instance, if the actual RTT is 120 ms, then the observed RTT is 150 ms if the TCP SYN was sent 50 ms after the Beacon arrival, while it is 210 ms if the TCP SYN was sent 90 ms after the Beacon arrival

the mobile device always responds with TCP ACKs immediately after the Beacon, and the TCP connection becomes synchronized with the PSM Beacon Period.

- **Increase in the execution time of interactive request/response applications.** Applications involving many successive request-response transactions, like file operations on a Network File System (NFS), may be negatively affected by the RTT increase described above. NFS is based on Remote Procedure Calls (RPCs) and issues requests in sequence, i.e., the next RPC request is issued after receiving the response for the previous one. According to the above remarks, RPCs experience a delay of approximately 100 ms[6]. If we consider the simple ls command to list files in a directory (NFS makes two RPCs for each file), the delay experienced in PSM may be up to 16-32 times greater than that experienced without power management [ANF03]. Similar considerations apply to other file and directory operations. Therefore, applications and programs containing a large number of NFS operations may experience a significantly increase in their execution times [ANF03].

  The above results show that the PSM sleep interval is too large especially when the connection RTT is small. In this case PSM may degrade significantly the performance of interactive request/response applications [KB02]. In addition, it may cause two paradoxical phenomena that are discussed below.

- **Increase in the power consumption of interactive request/response applications.** PSM may even increase the power consumption of interactive request/response applications. It may be worthwhile to recall here that the power consumed by the wireless interface is only a fraction of the total power consumed by the system. PSM actually reduces the power consumed by the wireless interface but, as shown above, it may increase considerably the execution times. Since the total power consumed by the system is the integral of power over time, it may happen that the power consumed by the system in the additional execution time is greater than the power saved by PSM [ANF03]. As mentioned above, in a palmtop computer the power consumption of the wireless interface is approximately 50% of the total consumption, while it is about 10% in a laptop computer [KK00, ACL03, ANF03]. In the latter case, reducing the power consumed by the wireless interface while increasing, at the same time, the execution time of the application may result in an increase of the overall system consumption. Therefore, power management should take into consideration the overall power characteristics of the entire system.

[6] It is assumed that the delay between the NFS server and the AP is small with respect to the Beacon period.

- **Performance inversion.** PSM may also cause a performance inversion so that TCP applications achieve higher throughput with a slower wireless link. Since the TCP connection becomes synchronized with the PSM Beacon, at the beginning of a Beacon period the amount of data buffered at the AP is equal to the TCP window size (assuming that there is sufficient bandwidth between the server and the AP). If the AP finishes transmitting the buffered data before new data arrive from the server, the mobile device enters the sleep mode until the next Beacon, and the throughput of the TCP connection is limited to one window per Beacon period. When the TCP window size increases beyond a certain threshold, new data arrives at the AP when the AP itself is still transmitting and, hence, the wireless interface remains continuously active. The threshold value is given by the product of the wireless link bandwidth and the actual RTT between the server and the mobile device. Therefore, a lower wireless bandwidth saturates sooner and prevents the wireless interface from entering the sleep mode. This performance-inversion effect shows that when moving, for example, from IEEE 802.11b to IEEE 802.11g products, the performance of TCP applications might degrade [KB02].

From the above remarks it clearly emerges that the 100-ms sleep interval of PSM can be *too large* to provide good performance, especially to interactive applications that are sensitive to the RTT of the connection. In principle, the problem could be overcome by considering a shorter sleep interval. On the other hand, many network applications exhibit a bursty behavior characterized by data-transfer phases interleaved by idle periods without any network activity. For example, a Web user typically reads information for several seconds after downloading them. Although there is no network activity during idle periods, PSM wakes up the wireless interface every 100 ms resulting in a large amount of energy wastage. From this standpoint, the sleep interval of PSM is *too short* to minimize power consumption during long idle periods.

5.1.2 Adaptive PSM

The problem with IEEE 802.11 PSM is its static nature that does not allow it to adapt to the network conditions and the application requirements. Ideally, power management should maintain the wireless interface active when there are packets to transfer, and disable it during idle periods. Cisco Aironnet 350 cards partially implement this policy. They appear to disable PSM (thus remaining continuously active) when more than one packet destined for the mobile device is waiting at the AP. PSM is re-enabled after approximately 800 ms without receiving a packet [ANF03]. Throughout, this protocol will be referred to as adaptive PSM. Other adaptive power-management policies are presented below.

5.1.3 Bounded-Slowdown (BSD)

Starting from the evidence that the 100-msec PSM sleep interval is too large especially for interactive request/response applications, and too short for minimizing power consumption during long inactive periods, Krashinsky et al. [KB02] propose the Bounded-Slowdown (BSD) protocol that dynamically adapts its behaviour to the network conditions. The target of this protocol is to minimize the power consumption while guaranteeing that the RTT does not increase by more than a given percentage. To achieve this goal the wireless interface remains active for a short period after sending a request, and adaptively listens to a subset of beacons when the network is idle. Staying in the active state reduces communication delays but increases power consumption. On the other hand, listening to fewer beacons saves energy but may increase communication delays.

Formally, let R be the RTT of the TCP connection between the mobile device and the remote server when the wireless interface is continuously active (i.e., without power management). The goal of the BSD protocol is to minimize power consumption while limiting the observed RTT to $(1+p)\cdot R$, where $p>0$ is a specified parameter. The protocol works as follows. After sending any packet (e.g., a request, or a TCP ACK) the mobile device maintains the wireless interface in the active mode for a time interval T_{awake}. Then, it enters a loop during which, at each step, performs the following actions: (i) sleeps for a period T_{sleep}; (ii) wakes up and polls the AP for receiving newly arrived packets (if any). The loop is exited as soon as the mobile device has a new packet to send. When this occurs, the algorithm is restarted (on the other hand, packets received from the AP have no effect on the algorithm). At each iteration, the sleep interval T_{sleep} is obtained as the time elapsed since the (last) packet was sent by the mobile device, multiplied by p. Since T_{sleep} is the maximum additional delay introduced by the BSD protocol, the observed RTT is, at most, equal to $R+T_{sleep}=(1+p)\cdot R$.

To implement the BSD protocol in a realistic environment, e.g., in IEEE 802.11 cards, T_{awake} and T_{sleep} need to be synchronized with the Beacon period T_{bp} (T_{bp} =100 ms in IEEE 802.11 PSM). Specifically, T_{sleep} must always be *rounded down* to a multiple of T_{bp}. The first time the wireless interface is put in the sleep mode T_{sleep} is equal to T_{bp} and, hence, the RTT might increase by, at most, T_{bp}. Therefore, T_{awake} must be equal to T_{bp}/p. Figure 8 shows the evolution of PSM and BSD when p=0.5. In this example $T_{awake}=T_{bp}/p=100/0.5=200$ ms [KB02].

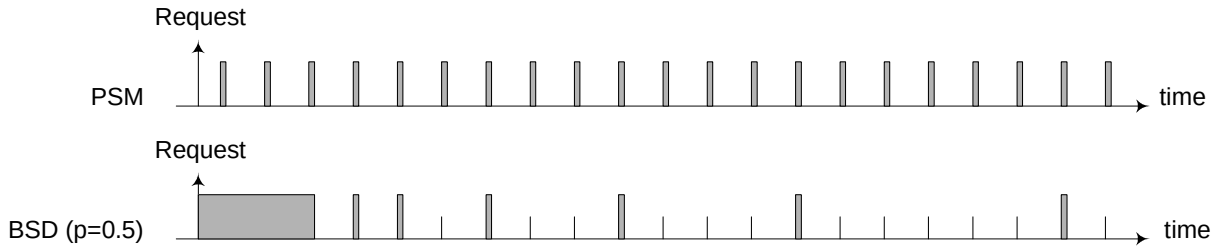


Figure 8. Evolution of PSM and BSD with p=0.5 [KB02].

In [KB02] a simulation analysis of BSD is performed based on Web traffic traces. A single mobile device communicating with the AP is considered. It is shown that, compared to PSM, BSD reduces average page retrieval times by 5-64%, and (at the same time) the power consumption by 1-14%. However, the benefits of using BSD instead of PSM are more evident when the RTT of the TCP connection is smaller than the Beacon period (i.e., 100 ms).

The BSD protocol has some important drawbacks. Since the wireless interface may be disabled even for very long times, this policy is nice for a mobile device operating as a client in request/response applications (e.g., Web, e-mail, file transfer, etc.). It is not appropriate when the mobile device acts as a server, or in a peer-to-peer scenario. In these cases, long disconnections may prevent the mobile device from receiving packets (e.g., a service request) from other devices. Furthermore, the proposed protocol has no knowledge about the application behavior. This limits the potentialities of the protocol. Finally, to be used in Wi-Fi cards the BSD protocol requires the mobile device to be able to skip a dynamically varying number of Beacons. This may not be easy to implement in current devices.

5.2 Upper-layer Policies

Both adaptive PSM and BSD are modified versions of the IEEE 802.11 PSM trying to adapt to the network activity. Therefore, these proposals cannot be used in Wi-Fi hotspots where mobile devices are equipped with fully-compliant IEEE 802.11 wireless cards. In addition, they operate at the MAC layer and, thus, have no knowledge about the application behavior. Upper-layer solutions discussed hereafter overcome these problems as they can be used, in principle, with any wireless interface. In addition, they can exploit application-specific information.

In the next section we will introduce several upper-layer policies: some of them are local policies, while some others are distributed policies.

5.2.1 Application-driven Power Management

The simplest form of upper-layer power management is an application-specific policy that disables the wireless interface (i.e., put it in the doze mode) whenever there are no more data to be transferred. If we look at the typical behavior of a (non real-time) network application such as

Web browsing -- shown in Figure 9 -- we can observe that *active times*, during which data transfers occur, alternate with *inactive times* characterized by no network activity. Furthermore, data transfers inside the same active time are separated by short idle periods called *active off times*. During both inactive times and active off times the wireless interface is idle. However, active off times are generated by the interaction between the mobile device and the remote server, while inactive periods are related to the user activity (they are also called user think times). Inactive times are usually much longer than active off times, and are typically greater than 1 sec [BC98]. Therefore, simply disabling the wireless interface during inactive phases can save a large amount of energy.

Based on this evidence, two simple application-layer policies are investigated in [SK97]. They are local policies and are specifically tailored to e-mail and Web-browsing applications (currently, the most popular applications for handheld devices), respectively. For e-mail applications, the wireless interface is usually in the doze mode. It wakes up periodically to check for possible messages, and remains active only for the amount of time necessary to send/receive e-mail messages to/from the mail server. For Web-browsing applications, the wireless interface is disabled after a certain period without any network activity and re-enabled when a new request is generated by the application.

Simulation results shows that the power consumption can be reduced to the minimum power necessary to send/receive e-mail messages, while for web browsing the power consumption can be reduced by approximately 75% with a negligible impact on the Quality of Service (QoS) perceived by the user [SK97, ACGP04b]. However, the proposed solutions require some application modifications. Furthermore, a software module that coordinates the activity of different network applications is required: the wireless interface can be disabled only when all the applications are not transferring data, and must be enabled as soon as any application has data to transmit. Again, suspending the wireless interface for long time periods may be suitable only when the mobile device acts as a client, while it is not appropriate when the mobile device that has to respond to external requests (e.g., when the mobile device acts as a server or in a peer-to-peer environment).

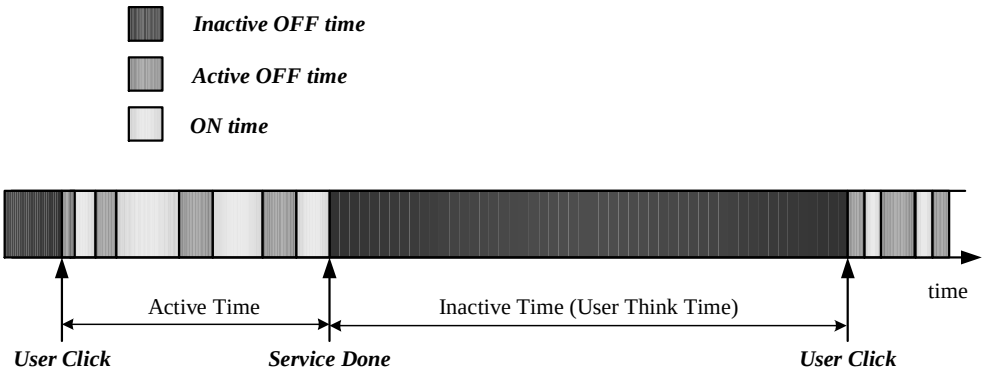


Figure 9. Typical behaviour of a network application.

5.2.2 Self-Tuning Power Management (STPM)

Anand et al. [ANF03] propose a Self-Tuning Power Management (STPM) protocol that extends the idea of application-driven power management to any network applications. The proposed solution not only takes into account the application behavior, but also adapts to (i) the network activity; (ii) the characteristics of the wireless interface; and (iii) the system on which the protocol is running. The goal is to have a flexible power management that optimizes the power consumption, based on the energy characteristics of both the wireless interface and the entire system, while guaranteeing good performance to delay-sensitive applications. To achieve this goal, STPM switches the wireless interface between the different operation modes that are supported (active, doze, PSM), depending on several conditions. Specifically, the protocol is built upon the following design principles:

- **Power management should know the application behavior**. This is important to take appropriate decisions since different applications have different behaviors.
- **Power management should be predictive**. If it is possible to know in advance that a large number of successive data transfers will occur back-to-back it might be convenient remaining continuously active to improve performance. The wireless interface must be disabled (or put doze mode) only at the end of the overall data transfer.
- **Power management should differentiate network data transfers.** In particular, it should discriminate between *foreground* data transfers that are delay-sensitive, and *background* data transfers that are not.
- **Power management should reach a performance/energy tradeoff.** This tradeoff should depend on the context where the user's activity is carried on. If the battery of the mobile device is fully charged, a better performance could be provided at the cost of greater power consumption. If the battery is nearly exhausted performance should be sacrificed to save power.

- **Power management should adapt to the operating conditions.** The goal of power management should be to minimize the power consumed by the entire system, not only the power consumed by the wireless interface.

Figure 10 shows a possible implementation of the STPM protocol as a kernel module. Applications provide the STPM module with hints about their behavior. Specifically, before each data transfer, the application specifies whether the forthcoming data transfer is a foreground or background activity and, optionally, the expected amount of data to be transferred. At the end, the application informs the STPM that the data transfer has been completed. To implement a predictive power management, STMP monitors the number of data transfers that are closely correlated in time so that they can be assumed to form a cluster of transfers, or *run*. STPM uses an empirical distribution of run lengths to calculate the number of expected data transfers in the current run, given the number of data transfers already observed. Based on this information, it decides whether it is convenient to remain continuously active or switching to PSM (or to doze mode).

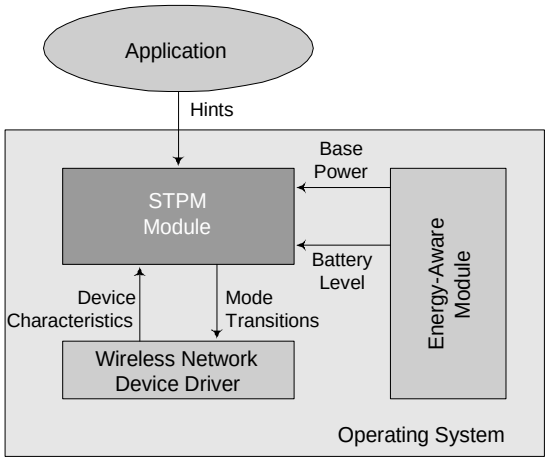


Figure 10. STPM Software Architecture [ANF03].

Applications also specify the maximum additional delay (due to power management) they are willing to tolerate for incoming packets. Information about the power consumption of the mobile device (base power), as well as hints for deciding the right performance/energy tradeoff, can be provided either by the user or the operating system itself (as shown in Figure 10).

Finally, to take appropriate decisions, STMP needs to know the energy characteristics of the wireless interface. Wireless cards differ in the types and number of power saving modes they support, energy costs to transit from one mode to another, and transitions times. In the implementation proposed in [ANF03], STPM uses a code component that includes the energy characteristics of the specific wireless card used by the mobile device. This code component is

loaded by STPM in the same way that the operating system loads the device driver for a given component.

Based on the above information, STPM decides when it is convenient to switch the wireless interface from one operational mode to another (e.g., from PSM to active, or vice versa). Assuming for simplicity that the wireless card only supports two operational modes, i.e., PSM and active, STPM calculates the total costs (in terms of energy and times) of performing the data transfer in each mode. It compares the results obtained and decides to switch to the active mode if it predicts that performing the data transfer staying continuously active will both save energy and improve performance. If STPM estimates that a transition to the active mode will either reduce power consumption or improve performance but will not reach both objectives, the decision is based on information (provided by either the user or the operating system) about the relative tradeoff between performance and energy conservation. Similarly, STPM decides to switch back to PSM when no data transfer is in progress and it predicts that the expected benefits of reduced power consumption during the remaining idle period will exceed the performance and energy costs of starting the next data transfer in PSM (see [ANF03] for details).

In [ANF03] it is shown that, compared to PSM, STPM reduces the power consumption of a palmtop computer operating on a distributed file system by 21% and, simultaneously, reduces the filesystem interactive delay by 80%. In addition, the proposed solution is flexible as it is able to adapt to different systems (e.g., laptops and palmtops) and wireless interfaces with different energy characteristics. However, in the presence of Web-browsing applications (i.e., the most popular application, along with e-mail, for handheld computers) it is not clear which is the performance of STPM. A major SMTP drawback is that application clients need to be modified to provide the STPM module with specific hints about the application behavior. Although these modifications usually consists in adding few lines of code, this aspect prevents using legacy network applications with STPM.

5.2.3 Communication-based Power Management

According to the classification provided at the beginning of Section 5, both application-driven power management and STPM are *local* policies because the power management algorithm is confined inside the mobile device. On the other hand, MAC-layer policies described in Section 5.1 implement a distributed power management that relies on the AP to buffer incoming packets during mobile-device disconnection periods. When using a local approach, the mobile device remains disconnected from the rest of the system when the wireless interface is disabled. This may prevent the mobile device from receiving external packets. Hence, the wireless interface can be disabled only when the mobile device does not expect to receive any packet from the network

(e.g., during Think Times in Web browsing applications). However, even if all data transfers are controlled by the mobile device itself, by looking at Figure 9, the wireless interface can be disabled only during Inactive Times (or User Think Times), while it must be continuously active (or in PSM) during Active Times (when the mobile device can receive packets from the server). This may result in power wastage especially when the server is busy and idle times between consecutive data transfers (i.e., Active Off Times in Figure 9) are not short. Policies presented hereafter overcome this problem by taking a distributed approach similar to IEEE 802.11 PSM. In addition, they are implemented at the transport or application layer and can, thus, exploit hints from the applications.

Kravets et al. [KK98, KK00] propose a communication-based power management protocol that operates at the transport layer but allows applications to control the power management parameters. The protocol is based on the Indirect-TCP model [BB97], i.e., the TCP connection between the mobile device and the remote server is split into two TCP connections: one from the mobile device to the AP, and another from the AP to the remote server. Furthermore, the protocol is designed as a master-slave protocol where the role of master is played by the mobile device. The mobile device decides when to suspend/resume the wireless interface, and tells the AP when data transmissions can occur. Data arriving at the AP, while the mobile device is disconnected, are buffered and delivered later, when the mobile device reconnects.

Ideally, the wireless interface should be disabled as soon as the application enters a phase of inactivity, i.e., during user thinks times or between active off times (see Figure 9). However, unless the application sends an explicit notification, the power management protocol is not able to know when the inactivity phase exactly starts. Therefore an *inactivity timer* is used that is reset after each transmission or reception. If the inactivity timer expires, the mobile device informs the AP that it is going to sleep and disables the wireless interface. The wireless interface will be re-enabled either when the application generates new data to send, or upon expiration of a *wakeup timer*. Then, the mobile device sends a WAKE_UP message to the AP, along with any new data, and waits for receiving buffered packets (if any).

A key role in the above protocol is played by two timing parameters, the *timeout period* (for the inactivity timer) and the *sleeping duration* (for the wake_up timer). As highlighted in Section 4, if the timeout period is too short, the wireless interface may go to sleep too early, causing an increase in the application response time. If it is too large, the wireless interface remains active for long periods of time, thus wasting energy unnecessarily. Similarly, if the sleeping duration is too small, the wireless interface is almost always active, thus consuming more than necessary. If it is too long, the additional delay introduced by the power management protocol may degrade

the QoS perceived by the user. Clearly, the appropriate sleeping duration depends on the application. Therefore, the power management protocol should set the sleeping duration by exploiting hints provided by the application. Furthermore, if more than one application is running on the mobile device, the power management protocol must consider hints from all applications and set the timing parameters according to the more stringent requirements. Finally, the power management protocol should be adaptive by varying the timeout period and sleep duration according to the application activity. Compared to the solution based on fixed timeouts, adaptive power management improves power saving and reduces additional delays introduced by power management [KK00].

5.2.4 Application-dependent Adaptive Power Management

Adaptation is the key idea also in [ACGP02, ACGP03a, ACGP03b]. These authors take an approach similar to the one in [KK00]. Specifically, they propose a policy based on the Indirect-TCP model [BB97] that (i) operates just below the application; (ii) exploits the characteristics of the application itself; and (iii) buffers at the AP packets addressed to the mobile device during disconnection periods (distributed policy). The novelty of this approach it its ability to adapt to the application behavior without requiring any modification of the application itself. Specifically, this approach can be implemented either with an *application-dependent* way or in an *application-independent* way.

The *application-dependent* approach is proposed in [ACGP02, ACGP03b] by developing a power management algorithm which is application specific (e.g., tailored to Web-browsing, e-mail or some other application). The AP is aware of some (off-line) statistics about the traffic profile generated by the application (e.g., in Web-browsing applications, expected number of embedded files in a Web page, expected size of html and embedded files, etc.). In addition, the AP also continuously monitors the network activity in order to obtain run-time estimates of the throughput available along the connection with the remote server.

To better explain this policy in the following we will make reference to Web-browsing applications. Upon receiving a Web-page request from the mobile device, the AP derives an estimate of the main-file transfer time (i.e., the time needed to download the main html file from the remote Web server), and decides whether (or not) it is convenient for the mobile device to disconnect in the meanwhile. This decision takes into account: (i) energy and delay costs associated with suspending and resuming the wireless interface at the mobile device; and (ii) maximum additional delay the application is willing to tolerate. If disconnection is convenient,

the AP suggests this decision to the mobile device along with the time interval it should sleep. The mobile device suspends the wireless interface for the notified sleeping time[7]. While the mobile device is disconnected, the AP stores the main html file and, automatically, prefetches embedded files, if any. When the mobile device reconnects, the AP delivers all the available data (the main file and the embedded files, if available). If more data need to be downloaded, the AP also derives an estimate for the residual transfer time. If disconnection is convenient, the AP suggests the mobile device to disconnect.

From the above description it appears that the AP dynamically generates updated estimates for the residual transfer time, and based on these estimates, suggests the mobile device either to disconnect or to remain active. Furthermore, the AP acts as an agent for the mobile device, i.e., it performs some actions on behalf of the mobile device to improve power saving and/or reduce the delay. Specifically, it includes mechanisms for file caching and/or prefetching. In practice, the AP can be seen as a Web proxy with power management capabilities.

Results obtained by considering Web-browsing applications in a real environment (i.e., by using a real Web server accessed by the client through a real Internet connection) have shown that the application-dependent power management protocol is able to achieve power saving not only during inactivity periods (i.e., during User Think Times) but also during the data transfer phase, i.e., during the download of Web pages. Furthermore, by continuously monitoring the network connection between the AP and the Web server, it is able to dynamically adapt to network and server conditions. In [ACGP03b] it is shown that the application-dependent protocol reduces significantly the overall time during which the mobile device should be connected with respect to the application-driven protocol (this protocol switches off the wireless interface only during User Think Times). In addition, the performances of the application-dependent protocol are not far from an optimal (unfeasible) algorithm that switches on the wireless interface exactly when data becomes available at the AP, and disable it immediately after the data transfer.

5.2.5 Application-independent Adaptive Power Management

The major drawback of the application-dependent power management protocol is that it must be tailored to a specific application. Therefore, a new version of the power management module is required for each network application. Furthermore, when there are more applications running on the mobile device, power management modules related to the various applications need to coordinate themselves in such way that the wireless interface is actually disabled only when *all*

[7] The mobile device may reconnect before the sleeping interval has expired if a new request is generated by the client-side application.

power management modules have decided to suspend it, and is re-enabled as soon *any* of them decides to resume it.

To overcome these difficulties an *application-independent* power management protocol is proposed in [ACGP03a]. This protocol autonomously becomes aware of the application's requirements and communication pattern, and exploits this knowledge to achieve power saving. Specifically, the protocol continuously monitors the application trying to learn its behavior and adapt to it. The protocol can be used with any network application, since no hints are required from the application. Furthermore, legacy applications can be used unmodified.

To take appropriate decision about when to suspend and resume the wireless interface, the application-independent power management protocol relies on an algorithm that provides estimates of both packet inter-arrival times (i.e., time intervals between consecutive packets) and inactivity periods' duration. The Variable-Share Update algorithm [HW95], customized to the specific problem, is used to this purpose.

Basically, the Variable-Share Update algorithm works as follows (see Figure 11). Let I be the range of possible values for a variable y to be estimated. To predict the y value, the Variable-Share Update algorithm relies upon a set of *experts*. Each expert x_i provides a (fixed) value within the range I, i.e., a value that y can potentially assume. The number of experts to be used, as well as their distribution among the range I, are input data for the algorithm. Each expert x_i is associated with a weight w_i, a real number that measures the reliability of the expert (i.e., how accurately the expert has estimated y in the past). At a given time instant, an estimate of y is achieved as the weighted sum of all experts, using the current w_i value as the weight (i.e., reliability) for the expert x_i. Once the actual value of the variable y becomes known, it is compared with the estimate provided by the experts to evaluate their reliability and update the weight associated with each of them. The update policy requires two additional parameters, α and η, and a loss function L. This function provides a measure of the deviation of each expert from the actual value of the variable, and its values must lie in [0,1]. According to [HLS96], α=0.08 and η=4 are used in the application-independent protocol. Finally, $L(x_i, y) = |x_i - y| \Big/ \max_i |x_i - y|$ is used as error function (see [HW95] for details).

Parameters: $\eta > 0$, $0 \le \alpha \le 1$, n (number of experts)

Variables: x_i (experts), w_i (weights), y (actual variable's value), $\hat{y}$ (estimated variable's value)

Initialization:
$$w_i = 1/n \quad \forall i = 1,...,n$$

Prediction:
$$\hat{y} = \sum_{i=1}^{n} w_i x_i \Big/ \sum_{i=1}^{n} w_i$$

Loss Update:
$$w_i' \overset{\Delta}{=} w_i e^{-\eta L(y,x_i)}$$

Variable share:
$$\begin{cases} pool = \sum_i \left[1-(1-\alpha)^{L(y,x_i)}\right] w_i' \\ w_i = (1-\alpha)^{L(y,x_i)} w_i' + \dfrac{1}{n-1} \\ \qquad \left\{ pool - \left[1-(1-\alpha)^{L(y,x_i)}\right] w_i' \right\} \end{cases}$$

Figure 11. Variable-Share Update algorithm

The Variable-Share Update algorithm has to provide estimates for both packet inter-arrival times and inactivity periods. These variables span different range of values since inter-arrival times are typically smaller than inactivity periods. Two different sets of experts, with non-overlapping intervals for expert values, are thus used. The first interval ranges in [0, P_{max}], where P_{max} is the maximum expected inter-arrival time. Idle periods greater than P_{max} are assumed to belong to inactivity periods. The second interval ranges in (P_{max}, T_{max}], where T_{max} is the largest estimated inactivity period. This means that the mobile device will poll the AP for any incoming packets, at most, every $T_{max.}$ Therefore, T_{max} is also the maximum additional delay introduced by the power management protocol to the first packet of a data transfer.

In [ACGP04b] the application-independent protocol is evaluated in a realistic prototype system (real Internet environment) considering different applications running in the mobile device. In particular, three scenarios are considered. In the first and second scenario there is a single active network application: Web and e-mail, respectively. In the third scenario, Web and e-mail are simultaneously active. The experimental results show that the application-independent protocol is able to save a considerable amount of energy irrespectively of the specific application running in the mobile device. In the mixed-traffic (Web + e-mail) the protocol reduces the overall time during which the wireless interface should be on of about 72% with respect to the case without power management. In addition, this reduction is obtained without a significant impact on the QoS perceived by the user. In the mixed-traffic scenario the average additional delay introduced by the power management protocol is, on average, 0.41 sec for downloading a Web page and 1.9 sec for checking, downloading and uploading e-mail messages. In the other single-application scenarios analyzed the performance of the application-independent protocol are even better.

6. Power Management and Pervasive Computing

Nineties have started the mobile computing age. Mobile users can use their cellular phone to check e-mail and browse the Internet. Travellers with portable computers can access Internet services from airports, railway stations, and other public locations. In the same period, the hardware and software progresses provided the basic elements (PDAs, wearable computers, wireless networks, devices for sensing and remote control, etc.) for implementing a new computing paradigm, *pervasive* or *ubiquitous computing*, that will further extend the users possibility to access information or communicating with other users at anytime, anywhere, and from any device [W91].

The nature of ubiquitous devices[8] makes wireless communication the easiest solution for their interconnection. Wireless Local Area Network (WLAN) constitutes the basis for these environments. However, the heterogeneity of devices to be interconnected (ranging from small sensors and actuators to multimedia PDAs) results in a large spectrum of requirements that cannot all be covered by WLANs. From one extreme, we have very simple devices for which small size, low cost, and low power consumption are the main constraints, while a high data rate is not necessary. On the other side, we have devices that need to exchange multimedia information for which a high data rate is a must. Also coverage requirements may vary significantly (from few meters to an entire building). This generates a strong push for new classes of networks, referred to as *Body Area Network* (*BAN*) and *Personal Area Network* (PAN), to cover all the pervasive-computing communication requirements [C03].

A BAN constitutes the solution for connecting devices distributed on the human body (e.g., head-mounted displays, microphones, earphones, processors, and mass storage) that form a wearable computer [C03]. A BAN is a network with a transmission range of a human body, i.e., 1-2 meters. Short-range, low-power wireless communications provide an efficient solution to BAN implementation.

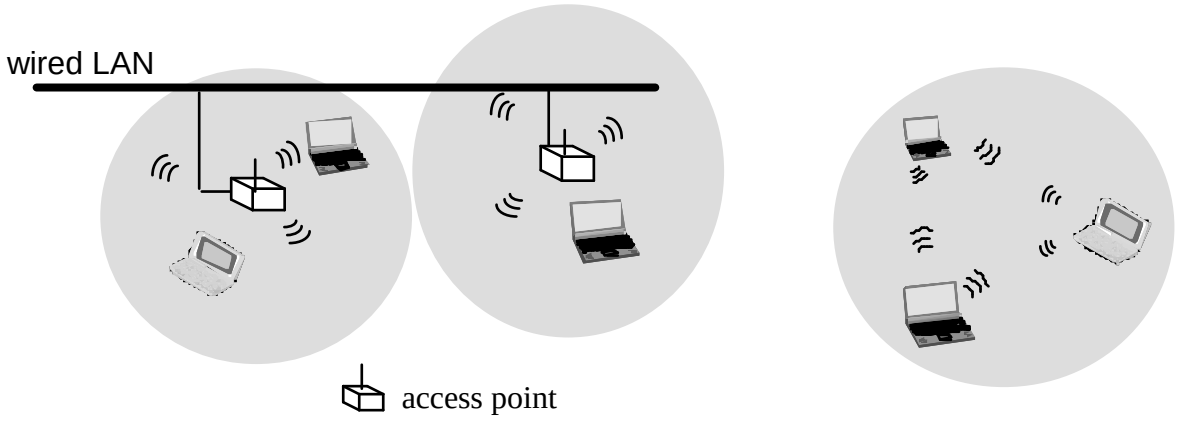
While a BAN is devoted to the interconnection of one-person wearable devices, a PAN enables the interconnection of the BANs of persons close to each other, and the interconnection of a BAN with the environment around it [C03]. A PAN communicating range is typically up to ten meters and, of course, wireless communications constitute the basis for PANs.

To cope with these new communication requirements, IEEE 802 Committees are currently developing standard specifications that constitute the basis for wireless communications in

[8] The devices are often located in inaccessible places or are spread on the human body.

pervasive computing environments. Specifically, in addition to the IEEE 802.11 standards for WLANs [802.11], the IEEE 802.15 working groups [802.15] are developing the specifications for short-range wireless communications among computers, peripherals, and (wearable) consumer electronic devices [C04]. These standards provide suitable solutions for both wireless PANs and BANs. A common feature of these network technologies is the ability to self-configure: a network exists as soon as two or more devices communicate on the same physical channel. No pre-existing infrastructure is required.

In mobile computing environments, most of the connections among wireless devices are achieved via fixed infrastructure WLAN networks (see Figure 1(a) and Figure 12(a)). The drawbacks of infrastructure-based networks are the costs and time associated with purchasing and installing the infrastructure. These costs and delays may not be acceptable for dynamic environments where people and/or vehicles need to be temporarily interconnected in areas without a pre-existing communication infrastructure (e.g., inter-vehicular and disaster networks), or where the infrastructure cost is not justified (e.g., in-building networks, specific residential communities networks, etc.). In these cases, infrastructure-less or ad hoc networks can provide a more efficient solution. An ad hoc network is a peer-to-peer network formed by a set of stations within the range of each other that dynamically configure themselves to set up a temporary network (see Figure 12(b)). Ad hoc networks are created, for example, when a group of people come together, and use wireless communications for some computer-based collaborative activities; this is also referred to as *spontaneous networking* [FAW01]. In the ad hoc configuration, no fixed controller (AP) is required, but a controller is dynamically elected among all the stations participating to the communication.



(a) infrastructure-based (b) ad hoc

Figure 12. WLAN configurations.

It is worth remembering that while 802.11 networks are generally implemented as infrastructure-based networks, the 802.11 standards also enable the construction of peer-to-peer WLANs. In this case, the IEEE 802.11 stations communicate directly without requiring the intervention of a centralized AP. Ad hoc networks (hereafter referred to as single-hop ad hoc networks) interconnect devices that are within the same transmission range. This limitation can be overcome by exploiting the multi-hop ad hoc (MANET) technology. In a MANET, the users' mobile devices are the network, and they must cooperatively provide the functionality usually provided by the network infrastructure (e.g. routers, switches, servers). In a MANET, no infrastructure is required to enable information exchange among users' mobile devices. Nearby terminals can communicate directly by exploiting, for example, wireless LAN technologies. Devices that are not directly connected communicate by forwarding their traffic via a sequence of intermediate devices.

Ad hoc networks inherit the traditional problems of wireless infrastructure-based networks discussed in the previous section. To these problems, the (multi-hop) ad hoc nature and the lack of fixed infrastructure add a number of characteristics, complexities, and design constraints that are specific of ad hoc networking, such as [CMC99]: multi-hop routing, dynamically changing network topologies, etc. Among these, power management is a key issue. The lack of fixed network elements, directly connected with a power supply, increases the processing and communication load of each mobile device, and this heavily impacts on the limited power of mobile devices. This becomes even a bigger issue in MANETs where a mobile device must spend its energy to act as a router forwarding the other mobile devices' packets [CCL03].

Hereafter, we will first discuss the power-management problem in single-hop ad hoc networks, and then we will address power-management issues in multi-hop ad hoc networks.

6.1 Power Management in Single-hop Ad Hoc Networks

Currently, the widespread use of IEEE 802.11 cards makes this technology the most interesting off-the-shelf enabler for ad hoc networks [ZD04]. For this reason, hereafter, we will concentrate on power management policies applied to 802.11 technology. A more general discussion on power management for ad hoc networks can be found in [WC02, F04].

Power-management policies for IEEE 802.11 networks operate at the MAC layer with the aim to maximize the mobile device battery lifetime without affecting the behavior of the high-level protocols. Two types of policies, operating with different time scales, have been designed/investigated for 802.11 networks. The first class operates on small time scales (say milliseconds) and tries to optimize the behavior of the CSMA/CA protocol used by 802.11 by avoiding power wastage during packet transmissions. The latter class extends to infrastructure-

less 802.11 networks the policy developed for infrastructure-based networks (see Section 5.1.1): mobile devices operating in PSM spend most of the time in the sleep mode (also referred to as *doze* mode), and periodically wake up to receive/transmit data.

6.1.1 Power Management during Transmission

The power-management policies belonging to this class are designed to avoid transmitting when the channel is congested, and hence there is a high collision probability [BCD0, BCG02]. Power saving is obtained by minimizing the energy required to successfully transmit a packet. This is achieved in two steps:

i) The development of an analytical model to characterize the system-parameter values (mainly the backoff parameters) that correspond to the minimum power consumption;

ii) The enhancement of 802.11 backoff mechanism to tune (at run time) the backoff window size to minimize the energy required for a successful packet transmission given the current status of the network, i.e., number of active mobile devices, length of the messages transmitted on the channel, etc.

As far as point i), in [BCG02] was developed an analytical model for the energy drained by the wireless interface of a tagged mobile device to successfully transmit a packet. Specifically, the model provides a characterization of the power consumption of the tagged device by studying the system behavior in the interval (referred to as *virtual transmission time*) between two consecutive successful transmissions of the tagged device itself, which corresponds to a system renewal interval. As shown in Figure 13, the virtual transmission time can be partitioned into: idle periods, collisions and successful transmissions. Collisions and successful transmissions can be further partitioned into two subsets depending on the tagged device involvement. This distinction is useful because the tagged-device power consumption is different in tagged collisions (i.e., collisions where the tagged device is involved) and non-tagged collisions.

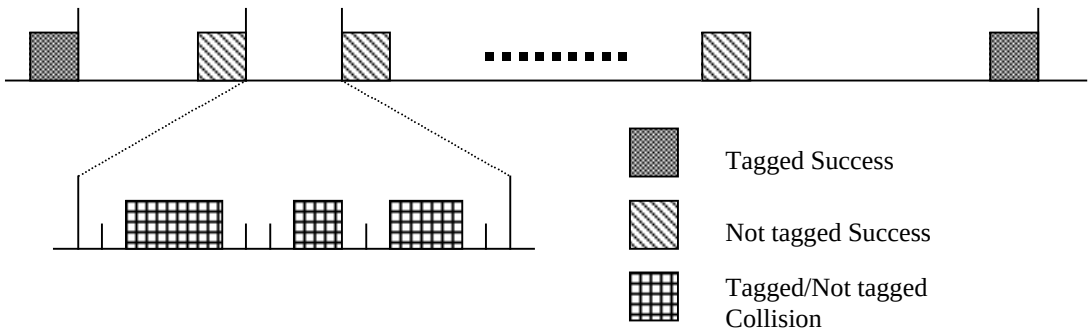


Figure 13. Channel structure during a virtual transmission time.

It can be shown that the system regenerates with respect to the starting point of a virtual transmission time. Hence, the system power efficiency, from the tagged-device wireless-interface standpoint, is the ratio between the average energy it uses to successfully transmit a message divided by the average energy it consumes in a virtual transmission time:

$$\rho_{energy} = \frac{PTX \cdot l}{E\left[Energy_{virtual\_transmission\_time}\right]},$$

where l is the average time required to the tagged device to transmit a message, PTX denotes the power consumption (per time unit) of the network interface when it is in the transmit state ($PTX \cdot l$ is the average energy required to the tagged device to successfully transmit the payload of a message), and $E[Energy_{virtual\_transmission\_time}]$ is the overall energy consumed by the tagged device during a virtual transmission time, and includes all the protocol overheads (idle slots, collisions, etc.).

The optimal energy consumption for the tagged device corresponds to the maximum value of ρ_{energy}, say ρ_{energy}^{MAX}. As $E\left[Energy_{virtual\_transmission\_time}\right]$ depends on the current status of the network, ρ_{energy}^{MAX} can be expressed as $\rho_{energy}^{MAX} = f(network\ status)$. The function *f()*, derived in [BCG02], shows that, to achieve the minimum energy consumption, the parameters of the backoff algorithm must be dynamically updated. In [BCD01] it is presented and evaluated a simple and effective extension of the 802.11 backoff algorithm to dynamically tune its parameters thus allowing the optimal energy consumption. This is achieved by exploiting a distributed mechanism, named *Asymptotically Optimal Backoff* (*AOB*), which dynamically adapts the backoff window size to the current load [BCG04]. AOB guarantees that an IEEE 802.11 WLAN asymptotically (i.e. for a large number of active mobile devices) achieves its optimal behavior. The AOB mechanism adapts the backoff window to the network contention level by using two simple load estimates that can be obtained with no additional costs or overheads [BCG04].

In [BCG02] it has been shown that, for 802.11b technology, the tuning of the network interface for achieving the minimal energy consumption almost coincides with the optimal channel utilization. This behavior is associated with the energy consumption model of WLANs interface in which transmit, receive, and idle states are almost equivalent from a power consumption standpoint. This implies that, for 802.11b networks, energy savings can be achieved by optimizing at the same time also the other parameters that characterize the QoS perceived by the users, such as bandwidth, and MAC delay.

6.1.2 Power Saving Mode (PSM) in IEEE 802.11 Ad Hoc Networks

In the IEEE 802.11 standards' terminology, ad hoc networks are named *Independent Basic Service Set* (IBSS). An IBSS enables two or more IEEE 802.11 mobile devices to communicate directly without requiring the intervention of a centralized AP. Synchronization (to a common clock) of mobile devices belonging to an IBSS is sufficient to receive or transmit data correctly [ACG04]. In an IBSS, no centralized controller exists and Beacon frames, which contain timing information, are managed through a distributed process. The mobile device that initializes the IBSS defines the Beacon Interval. At the beginning of the Beacon Interval, all mobile devices wake up and randomly contend to transmit the synchronization Beacon. Specifically, all mobile devices in the IBSS schedule the transmission of a Beacon frame at a random time just after the target time identified by the Beacon interval. After the first successful transmission of the Beacon, all the other transmissions are cancelled. Mobile devices synchronize themselves with the first Beacon they receive. A more detailed description of IBSS setting up and management can be found in [ACG04].

For power management, a fixed-length ATIM (Ad hoc Traffic Indication Map) window is associated to the IBSS. All mobile devices must stay awake for an ATIM window after each beacon. During the ATIM window, each mobile device sends an *Ad hoc Traffic Indication Map* (ATIM) to every other mobile device for which it has pending traffic. Each mobile device that receives an ATIM message responds with an ATIM acknowledgment.[9] At the end of the ATIM window, mobile devices that have neither sent nor received ATIM announcements go back to sleep. All other mobile devices remain awake throughout the remainder of the Beacon Interval, in order to send and receive the announced traffic.

Figure 14 shows the operations in an IBSS with two mobile devices, A and B. Two consecutive Beacon Intervals are considered in which the Beacon is issued by mobile devices A and B, respectively. In addition, in the second interval mobile device A has a message to be delivered to mobile device B. In this case, an ATIM frame and an ATIM ACK are exchanged during the ATIM window interval. At the end of this interval, mobile devices A and B remain awake to complete the data exchange.

[9] These transmissions occur following the basic 802.11 access mechanisms (e.g., DCF). To avoid contention with data traffic, only beacons, frames' announcements, and acknowledgments are transmitted during the ATIM window.

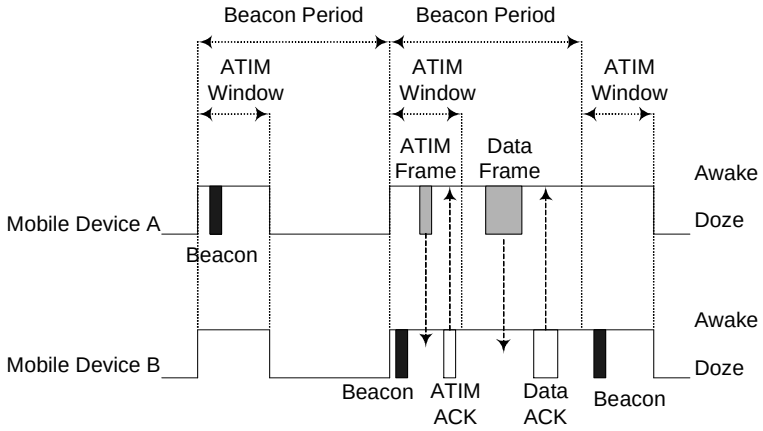


Figure 14. A data exchange between stations operating in PS mode in an 802.11 ad hoc network.

The efficiency of IBSS power management mechanisms highly depends on the values selected for the beacon intervals, the offered load and the length of the ATIM window. Results presented in the literature point out performance problems. Specifically, simulation results presented in [WESW98] indicate that power saving is obtained only at quite moderate loads; as offered load increases the saving declines substantially. The offered load also influences the choice of the parameter (e.g., the ATIM interval), hence the authors recommend adopting an adaptive ATIM window.

6.2 Power Management in Multiple-hop Ad Hoc Networks

Multi-hop ad hoc networking introduces a new metric for measuring the power saving: the network lifetime [E02]. In an infrastructure-based wireless network, power management policies are aimed at minimizing the mobile-device power consumption. This metric is not appropriate for ad hoc networks where mobile devices must also cooperate to network operations to guarantee the network connectivity. In a MANET, a selfish mobile device that remains most of the time in the doze mode, without contributing to routing and forwarding, will maximize its battery lifetime but compromise the lifetime of the network. On the other hand, policies aimed at maximizing the network lifetime must have a network-wide view of the problem. The idea is that, when a region is dense in terms of mobile devices, only a small number of them need to be turned on in order to forward the traffic. To achieve this, a set of mobile devices is identified which must guarantee network connectivity (to participate in packet forwarding), while the remaining mobile devices can spend most of the time in the doze mode to minimize power consumption. Mobile devices participating in packet forwarding may naturally exhaust their energy sooner, thus compromising the network connectivity. Therefore, periodically, the set of active mobile devices is recomputed by selecting alternative paths in a way that maximizes the overall network lifetime. Identifying the network's dominating sets is a typical goal of a global strategy. A dominating set is a subset of mobile devices such that each mobile device is in the

set, or it has a neighbor in that set. Dominating sets, if connected, constitute the routing/forwarding backbone in the multi-hop ad hoc network. As the computation of the minimal dominating set is computationally unfeasible, in the literature several distributed algorithms exist to approximate suitable dominating sets, see for example [DB97, WL99, WDGS02, CJBM02, XHE01, XHE00].

SPAN [CJBM02] is a distributed algorithm to construct dominating sets using local decisions to sleep, or to join the routing backbone. Mobile devices participating in the backbone are named *coordinators*. Coordinators are always in an active mode, while non-coordinator mobile devices are normally in the doze mode, and wake up to exchange traffic with the coordinators. Periodically, the coordinators' set is recomputed. The effectiveness of SPAN depends on the power consumption in the idle and sleep state: SPAN performance improves with the increase of the idle-to-sleep power-consumption ratio [CJBM02]. SPAN integrates with the 802.11 PSM, thus guaranteeing that non-coordinator mobile devices can receive packets that are buffered by the coordinators while they are sleeping.

The physical position of mobile devices (obtained for example via GPS) is used in the GAF algorithm to construct the routing/forwarding backbone. A grid structure is superposed on the network, and each mobile device is associated with a square in the grid using its physical position. Inside the square only one mobile device is in the non-sleeping mode [XHE01]. AFECA [XHE00] is an asynchronous distributed algorithm for constructing a routing backbone. Mobile devices alternate between active and sleep modes. In principle, a mobile device remains in the sleep mode for a time proportional to the number of its neighbors, thus guaranteeing, in average, a constant number of active mobile devices.

Controlling the power of the transmitting mobile device is the other main direction for power management in ad hoc networks. In wireless systems, the existence (or lack) of a link between two mobile devices mainly depends (given the acceptable bit error rate) on the transmission power and the transmission rate. By increasing the transmission power the number of feasible links is increased, but at the same time this increases the power consumption and the interference [E02]. Recently, several studies focused on controlling network topology by assigning per-device transmit powers that guarantee network connectivity, and minimize the transmit power [NKSK02, WLBW01, RRH00, F04, R04]. The algorithmic aspects of topology control problems are discussed in [LLM+02].

Transmission power is highly correlated with power consumption. It determines both the amount of energy drained from the battery for each transmission, and the number of feasible links. These two effects have an opposite impact on the power consumption. By increasing the transmission

power we increase the per-packet transmission cost (negative effect), but we decrease the number of hops to reach the destination (positive effect) because more and longer links become available. Finding the balance is not simple. On one hand, we have to consider the fact that signal strength at a distance r from the sender has non-linear decay, specifically $S(r) = S \cdot r^{-\alpha}$ $(\alpha \in [2,4])$, where S is the amplitude of the transmitted signal [E02]. This implies that, from the transmission standpoint, to cover the sender-to-receiver distance a multi-hop path generally requires less power. On the other hand, on a multi-hop path the delay (due to the multiple hops), as well as the processing energy (to receive and locally process a packet) increase.

The trade-off between minimum transmission power, and number of hops, further complicates the design of routing algorithms. A large part of recent work on power efficiency in ad hoc networks is concentrated on routing [RM99, SL00, RRH00], where the transmitting power level is an additional variable in the routing protocol design [F01].

Power-efficient routing has been addressed from two different perspectives: (i) energy is an expensive, but not a limited resource (battery can be recharged/replaced), or (ii) the energy is finite. The former case applies to (mobile) ad hoc networks in general, while the latter appears to be a suitable model for sensor networks. In case (i), power consumption must be minimized; typically, this translates in the following target: *minimize the total power consumed per packet to forward it from source to destination*. Minimizing per-packet energy does not maximize network lifetime, as residual energy of the mobile devices is not taken into consideration. On the other hand, in case (ii), the energy is a hard constraint [E02], and the maximum lifetime is the target.

Minimum-energy routings minimize the energy consumed to forward a packet from the source to the destination [LH01, RM99, GCNB03]. Similarly to proactive routing algorithms, [LH01, RM99] try to find minimum energy routes for all mobile devices; on the other hand, PARO [GCNB03] behaves as a reactive algorithm by minimizing the energy consumption of ongoing flows. In PARO, mobile devices intermediate to the source-destination pair elect themselves to forward packets, thus reducing the aggregate transmission power consumed by network devices.

On-line maximum-lifetime routing is a complex problem [LAR01]. In [CT00], for a static network with known and constant flows, the maximum lifetime routing is modeled as a linear programming problem. The solution of this model provides the upper bound on the network lifetime that is used to analyze the effectiveness of the algorithms. For a single power level, an optimal algorithm is presented; while, for the general case, the authors present an algorithm that

selects routes and adjusts the corresponding power levels achieving a close to the optimal lifetime.

A balance between minimum-energy and maximum lifetime is the target of the CMMBCR policy [T01]. CMMBCR applies a conditional strategy that uses the minimum energy route, if the mobile devices residual energy is greater than a given threshold. Otherwise, a route that maximizes the minimum residual energy is selected.

6.2.1 Power Management in Sensor Networks

Sensor networks constitute basic elements of emerging pervasive environments. Sensor nodes are small devices with computing, communication and sensing capabilities that can be used for various purposes. Typical sensing tasks could be temperature, light, sound, etc. Different kinds of "infrastructures" can be deployed to deliver the collected information from the sensing field to the place this information is elaborated. In recent years, advances in simple, low power, and efficient wireless communication equipments made wireless sensor networks the most interesting way to deliver the sensed data to the mobile device(s) in charge to collect/elaborate them. However, due to the technologic constraints, single-hop wireless communications are not efficient from the power consumption standpoint, and multi-hop communications are typically used to deliver the collected information. Specifically, a wireless sensor network is typically organized as shown in Figure 15.

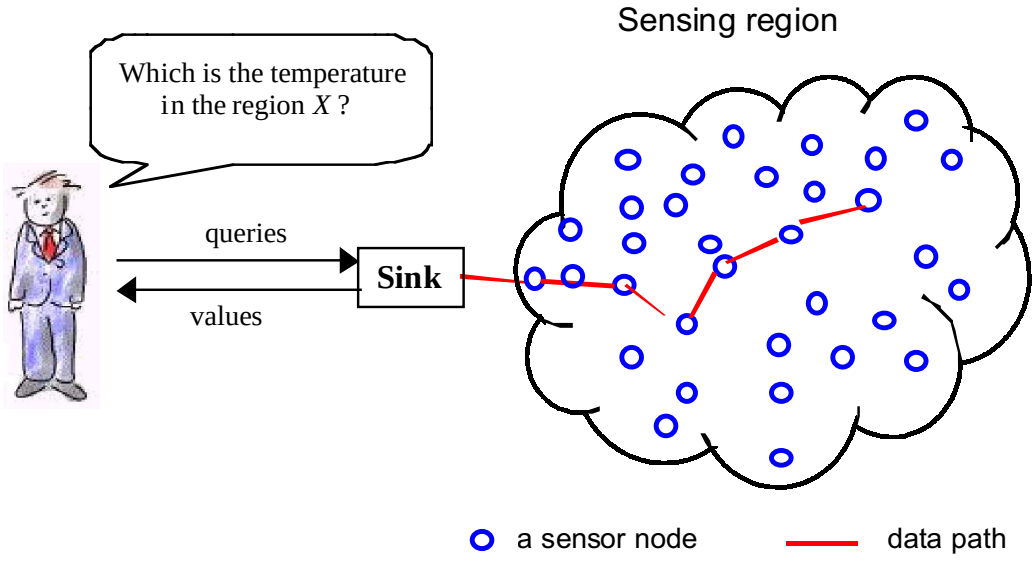


Figure 15. A wireless Sensor Network.

The sensor nodes are densely (and randomly) deployed inside the area in which a phenomenon is being monitored. Each sensor node delivers the collected data to one (or more) neighbor node,

one hop away. By following a multi-hop communication paradigm the data are routed to the sink and through this to the user. Therefore, multi-hop ad hoc techniques constitute the basis also for wireless sensor networks. However, the special constraints imposed by the unique characteristics of sensing devices, and by the application requirements, make solutions designed for multi-hop wireless networks generally not suitable for sensor networks [ASSC02]. First of all, power management is a "pervasive" issue in the overall design of a sensor network. Sensor networks utilize on-board batteries with limited energy that cannot be replenished in most application scenarios [S03]. This strongly affects the lifetime of a sensor network, and makes power awareness a critical factor in the design of sensor network protocols. Furthermore, sensor networks produce a shift in the networking paradigm from a node-centric to a data-centric view. This introduces a new dimension to power management. The aim of a sensor network is to collect information about events occurring in the sensor field. As sensor nodes are densely deployed several sensor nodes collect the same information and hence the importance of (the information collected by) each node is reduced with respect to MANET nodes. Indeed, while in MANETs the correct delivery of all data is important, the delivery of the information collected by each sensor node may tolerate a lower reliability. Moreover, while in node-centric networks the information is not manipulated inside the network (at most packets can be discarded if resources are not available for forwarding them), in sensor networks intermediate devices -- along the path from the sender (a sensor device) to the receiver (the sink) -- can exploit the correlation in the observed data to reduce (via compression and aggregation) the amount of information flowing into the network, and thus increasing the power efficiency. This is based on the observation that, in sensor devices, from the power consumption standpoint, processing is a much cheaper operation than transmitting [DWBP01, S03].

In a MANET only protocols operating at layer 1-3 of the OSI reference model participate to the operations inside the network (higher layers are affected only in the sender and receiver devices); on the other hand, in a sensor network also higher layers[10] contribute to power-efficient data delivery, making power management a "cross-layer" issue [CMTG04]. Tiny DB [MFHH03] represents the extreme case, where almost all the networking functions are implemented in the application layer.

The interested readers can find an excellent and comprehensive coverage of sensor networks in a recent survey [ASSC02]. Below, we will briefly discuss issues related to power management.

[10] Mainly the presentation and the application layers where compression and aggregation of data are applied to reduce redundancy in delivered data.

Specifically, we will survey power management policies at different layers of the OSI reference model.

For Physical and Data Link layers the power efficiency questions are similar to those addressed in wireless networks: how to transmit in a power efficient way bits and frames, respectively, to devices one-hop away. These problems include identifying suitable modulation schemes, medium access control policies [VL03, YHE02], efficient FEC schemes, etc., see [ASSC02, KW03]. Of course the solutions of these problems are strongly affected by the sensor-device resources' constraints. The proposed solutions are generally independent from the applications, however, recently some authors [VA03] proposed to apply data-centric policies also at the MAC layer. The basic idea is to exploit spatial correlation among neighboring nodes to reduce the number of transmissions at the MAC layer.

Power efficiency at network layer has several aspects in common with MANETs. Specifically, it deals with identifying paths for delivery the information (from the source to the receiver) that maximize the network lifetime. This involves the solutions of different problems such as: the control of the network topology, and the selection of the paths to be used for data delivery given a network topology. Similarly to MANETs, topology control algorithms either exploit power control techniques to construct energy efficient sub-graphs of the sensor networks (see Section 6.2), or re-organize the network in energy efficient clusters (see below).

Topology management techniques, like GAF and SPAN (see Section 6.2), can be applied to sensors networks for minimizing the number of devices that must be active to guarantee a connected topology. Furthermore, in a sensors network, additional savings can be obtained by observing that devices spend most of their time waiting for events without forwarding traffic, and hence, the network interface of active devices can remain in a sleep state for most of the time. This feature is exploited by STEM (Sparse Topology and Energy Management) [STGM02] to implement an energy efficient topology management scheme for sensor networks. STEM, when necessary, efficiently wakes up devices from a deep sleep state with a minimum latency. In addition, this scheme can be integrated with topology management techniques, like GAF and SPAN thus achieving compounded energy savings.

Clustering was introduced in 80's to provide distributed control in mobile radio networks [BEF84]. Inside the cluster one device is in charge of coordinating the cluster activities (*clusterhead*). Beyond the clusterhead, inside the cluster, we have: *ordinary nodes* that have direct access only to this one clusterhead, and *gateways*, i.e., nodes that can hear two or more clusterheads [BEF84]. As all nodes in the cluster can hear the clusterhead, all inter-cluster communications occur in (at most) two hops, while intra-cluster communications occurs through

the gateway nodes. Ordinary nodes send the packets to their clusterhead that either distributes the packets inside the cluster, or (if the destination is outside the cluster) forwards them to a gateway node to be delivered to the other clusters. Only gateways and clusterheads participate in the propagation of routing control/update messages. In dense networks this significantly reduces the routing overhead, thus solving scalability problems for routing algorithms in large ad hoc networks. Several clustering algorithms have been proposed for wireless sensor networks, e.g., see [BC03] and the references herein. On of most popular is the Low Energy Adaptive Clustering Hierarchy (LEACH). LEACH includes distributed cluster formation, local processing to reduce global communication, and randomized rotation of the cluster-heads to evenly distribute the energy load among the sensors in the network [HCB00].

The routing and forwarding of information inside a sensors network are data-centric functions through which data is disseminated to/from the sensors node. For example the query "tell me the temperature in the region *X*" needs to be disseminated to sensor nodes of a region *X* (see Figure 15). At the same time, data coming from the region X have to be delivered to the user(s) issuing the query. Simple techniques such as flooding and gossiping can be used to disseminate the data inside the sensor network [ASSC02]. However, these techniques waste energy resources sending redundant information throughout the network. Several application-aware algorithms have been devised to efficiently disseminate information in a wireless sensor network. These algorithms are based on the publishing/subscribe paradigm. Nodes publish the available data that are then delivered only to nodes requesting them. One of the most popular approaches is *Directed Diffusion* [IGE03], which delivers data only to nodes that have expressed interest to them. Similarly, SPIN sends data to sensor nodes only if they are interested [HKB01]. In addition, SPIN has access to the current energy level of the node and adapts the protocol behavior on the energy remaining in the node.

Dissemination algorithms achieve additional energy savings through in-network data processing based on data aggregation (or data fusion). Data aggregation is useful when sensed data come from sensors that have overlapping sensing regions, or belong to the query region *X*. By combining data values (and descriptions) at intermediate nodes, the amount of information to be delivered is minimized thus contributing to energy savings. For example, if a user queries "tell me the highest temperature in the region *X*" not all temperature samples need to be delivered, but an intermediate node needs to forward only the data related to the highest sample among those it received.

The sensor networks' data-centric nature combined with the strong resources' limitation make the Transport Control Protocol (TCP) protocol not suitable for the sensor network domain.

Indeed, inside a sensors network, different reliability levels and/or different congestion control approaches may be required depending on the nature of the data to be delivered. The transport layer functionalities must be therefore designed in a power-aware fashion, to achieve the requested service level, and minimize the energy consumption, at the same time. This implies the use of different policies for the forward path (from sensor nodes towards the sink) and the reverse path (from the sink towards sensor nodes).

In the forward path an event-reliability principle needs to be applied, i.e., not all data needs to be correctly delivered but, by exploiting spatial and temporal correlations, the transport protocol must guarantee the correct delivery of a number of samples sufficient for correctly observing (at the user side) the monitored event. Typically, sensor networks operate under light loads, but suddenly become active in response to a detected event and this may generate congestion conditions. In [WEC03] an event-driven congestion control policy is designed to manage the congestion in the nodes-to-sink path by controlling the number of messages that notify a single event. Indeed, the transport protocol should guarantee that, when an event is detected, the user correctly receives enough information. In [SAA03] the concept of event-driven transport protocol introduced in [WEC03] is enhanced to guarantee reliable event detection with minimum energy expenditure.

The reverse path typically requires a very high reliability as data delivered towards the sensors contain critical information (e.g., queries and commands or programming instructions) delivered by the sink to control the activities of the sensor nodes. In this case more robust, and hence power-greedy policies must be applied (see e.g., [WCK02]).

Sensor nodes in the sensing region X are typically set up to achieve in a cooperative way a pre-defined objective (e.g., monitoring the temperature in region X). This is achieved by distributing tasks to be performed on the sensor nodes. Therefore a sensor network is similar to a distributed system on which, at the same time, multiple applications are running. Each application is composed by several tasks that run on different (sensor) nodes. Starting from this view of a sensor network, in [BS03] the authors propose middleware-layer algorithms to manage, in a power-efficient way, a set of applications that may differ for the energy requirements and users' rewards. Specifically, the authors propose an admission control policy that, when an application starts, decides (given its energy costs and users' rewards) to accept/reject it to maximize the users' rewards. A policing mechanism is adopted, at runtime, to control that applications conform to the resource usage they declared at the admission stage.

7. Conclusions and Open Issues

In this chapter we have presented the main research trends in the field of power management for mobile and pervasive computing systems. Since mobile devices are supplied by batteries, we have firstly presented techniques to improve the battery capacity. We have highlighted that just relying on such improvements, without designing power management strategies, will greatly limit the use of mobile devices. Therefore, the remaining part of the chapter concentrates on power management policies. We have presented a general framework for mobile device power management. Within this framework, several approaches have been presented which can be applied to any component of a mobile device. Since networking activities account for a significant portion of the total power consumption, we have then focused on power management policies designed for the networking subsystem.

Most of the current mobile computing environments fall into the infrastructure-based or single-hop hotspot scenario (e.g., 2.5/3G cellular networks and Wi-Fi hotspots) in which a fixed network is extended by means of a wireless last-hop to provide connectivity to mobile devices. Thus, we have surveyed the main power management policies proposed for such a scenario. Specifically, we have focused on power management solutions for non real-time applications. To the best of our knowledge, less attention has been devoted to power management in the presence of real-time applications [CV02, C03, MCD+03, SR03], and this is still an important open issue.

Finally, we have presented power management techniques for mobile devices building single and multi-hop ad hoc networks. This networking scenario is fundamental, since it is the basis of pervasive computing environments.

In our opinion, the design of cross-layer power-management policies is a very promising direction, which should be further explored. Cross-layering can reduce the traffic on ad hoc networks by sharing information that is of interest to different layers [CMTG03]. Moreover, information collected at a particular layer (e.g., a route failure) can be exploited by different layers to tune the protocol behavior (e.g., the transport protocol can avoid sending data until a new route is found). Such cross-layer optimizations are not explicitly designed for power management, but reduce the power consumption as a side effect. On the other hand, cross-layer power-management policies could be designed, that exploit information residing at different layers in the protocol stack to manage communications on the wireless links. We have highlighted that the most appropriate way to save power is putting the wireless interface of mobile devices in low-power operating modes whenever possible. In ad hoc networks, the decisions on when to switch the wireless interface in the different operating modes is typically

based on MAC [VL03, YHE02] or routing-level information [CJBM02, HCB00]. In a cross-layer approach also information about the application behavior could be used to schedule the time intervals during which nodes are available for communications. Moreover, such an approach could be exploited also in infrastructure-based environments. So far, little attention has been devoted to such policies [VA03, ANF03].

We also believe that experimental validations of power management solutions are strongly required. Usually, performance evaluations rely just on simulations. However, simulation models tend to drastically approximate the complex physical behavior of wireless links. Therefore, experimental measurements typically show several features of network protocols that are completely unknown from simulations [ABCG04, LNT02].

Acknowledgements

This work was partially funded by the Information Society Technologies program of the European Commission, Future and Emerging Technologies under the IST-2001-38113 *MobileMAN* project, and partially by the Italian Ministry for Education and Scientific Research in the framework of the projects *Internet: Efficiency, Integration and Security* and *FIRB-VICOM*

References

[802.11] Web site of the IEEE 802.11 WLAN: http://grouper.ieee.org/grups/802/11/main.html

[802.15] Web site of the IEEE 802.15 WPA.N: http://www.ieee802.org/15/pub/main.html

[ABCG04] G. Anastasi, E. Borgia, M. Conti, and E. Gregori, "Wi-Fi in Ad Hoc Mode: A Measurement Study" Proceedings of the *IEEE Conference on Pervasive Computing and Communications (PerCom 2004)*, March 2004.

[ACF+04] G. Anastasi, M. Conti, A. Falchi, E. Gregori, A. Passarella, "Performance Measurements of Mote Sensor Networks", Proceedings of the *ACM/IEEE Symposium on Modeling, Analysis and Simulation of Wireless and Mobile System (MSWIM 2004)*, Venice (Italy), October 4-6, 2004.

[ACG04] G. Anastasi, M. Conti, E. Gregori, "IEEE 802.11 Ad Hoc Networks: Protocols, Performance and Open Issues", Chapter 3 of *Ad hoc networking*, (S. Basagni, M. Conti, S. Giordano, I. Stojmenovic, Editors), IEEE Press and John Wiley and Sons, Inc., New York, 2004.

[ACGP02] G.Anastasi, M.Conti, E.Gregori and A.Passarella, "A Power Saving Architecture for Web Access from Mobile devices", Proceedings of the *IFIP Networking Conference (Networking'02)*, Pisa (I), May 2002, Lecture Notes in Computer Science, LNCS 2345, pp. 240-251.

[ACGP03a] G. Anastasi, M. Conti, E. Gregori, A. Passarella, "Balancing Energy Saving and QoS in the Mobile Internet: an Application-Independent Approach", Proceedings of the *Hawaii International Conference on System Science (HICSS 2003)*, Minitrack on Quality of Service Issues in Mobile and Wireless Networks, Hawaii, January 6-9,

2003.

[ACGP03b] G.Anastasi, M.Conti, E.Gregori and A.Passarella, “Performance comparison of Power Saving Strategies for Mobile Web Access”, *Performance Evaluation*, Vol. 53, N. 3-4, August 2003, pp. 273-294.

[ACGP03c] G.Anastasi, M.Conti, E.Gregori and A.Passarella, “A Performance Study of Power Saving Policies for Wi-Fi Hotspots”, *Computer Networks*, Vol. 45, No. 2, June 2004.

[ACGP04a] G.Anastasi, M.Conti, E.Gregori and A.Passarella, “Saving Energy in Wi-Fi Hotspots through 802.11 PSM: an Analytical Model”, Proceedings of the *Workshop on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt 2004)*, University of Cambridge (UK), March 24-26, 2004.

[ACGP04b] G.Anastasi, M.Conti, E.Gregori and A.Passarella, “Experimental Analysis of an Application-independent Energy Management Scheme for Wi-Fi Hotspots”, Proc; of the *IEEE Symposium on Computers and Communications (ISCC 2004)*, Alexandria (Egypt), June 29 – July 1, 2004.

[ACL03] G. Anastasi, M. Conti, W. Lapenna, “A Power Saving Network Architecture for Accessing the Internet from Mobile devices: Design, Implementation and Measurements”, *The Computer Journal*, Vol. 46, N. 1, January 2003, pp. 3-15.

[ANF03] M. Anand, E. Nightingale, J. Flinn, “Self-Tuning Wireless Network Power Management, Proceedings of the ACM Mobicom 2003, S. Diego (CA), September 14-19, 2003.

[ASSC02] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, “Wireless sensor networks: a survey”, Computer Networks, Vol. 38 (2002), pp 393-422.

[BB95] T.D. Burd and R.W. Brodersen, “Energy Efficient CMOS Microprocessor Design”, in *Proc. of the 28th IEEE Annual Hawaii International Conference on System Sciences* (*HICSS’95*), Jan. 95.

[BB97] A.Bakre, B.R.Badrinath, “Implementation and Performance Evaluation of Indirect TCP”, *IEEE Transactions on Computers*, Vol.46, No.3, March 1997.

[BBD00] L. Benini, A. Bogliolo and G. De Micheli, “A Survey of Design Techniques for System-Level Dynamic Power Management”, *IEEE Transactions on VLSI Systems*, Vol. 8, N. 3, June 2000.

[BBD99] L. Benini, A. Bogliolo and G. De Micheli, “Dynamic Power Management of Electronic Systems”, *System Level Synthesis*, 1999.

[BC03] S. Bandyopadhyay, E. J. Coyle An Energy Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks”, Proc. of IEEE INFOCOM 2003.

[BC98] P.Barford e M.Crovella, “Generating Representative Web Workloads for Network and Server Performance Evaluation”, *Proceedings of ACM SIGMETRICS ´98*, Madison, WI, pp. 151-160, June 1998.

[BCD01] L. Bononi, M. Conti, L. Donatiello, “A Distributed Mechanism for Power Saving in IEEE 802.11 Wireless LANs”, *ACM/Kluwer Mobile Networks and Applications Journal*, Vol 6. N.3 (2001) pp. 211-222.

[BCG02] R. Bruno, M. Conti, E. Gregori, “Optimization of Efficiency and Energy Consumption in p-persistent CSMA-based Wireless LANs”, *IEEE Transactions on Mobile Computing*, Vol 1. N.1, 2002, pp. 10-31.

[BCG04] L. Bononi, M. Conti, E. Gregori, “Runtime Optimization of IEEE 802.11 Wireless

LANs Performance", IEEE *Transactions on Parallel and Distributed Systems*, January 2004.

[BD02] L. Benini and G. De Micheli, "Energy-efficient system-level design", in *Power-Aware Design Methdologies* (J. Rabaey and M. Pedram, Editors), Kluwer, pp. 473-516, 2002.

[BEF84] D. J. Baker, A. Ephremides, and J. A. Flynn, "The Design and Simulation of a Mobile Radio Network with Distributed Control", *IEEE Journal on Selected Areas in Communications*, Volume 2, Number 1, January 1984.

[BGS94] D. Bacon, S. Graham, and O. Sharp, "Compiler transformation for high-performance computing" *ACM Computing Survey*, Vol. 26 (1994), N. 4, pp. 345–420.

[BS03] A. Boulis and M.B. Srivastava, "Node-Level Energy Management for Sensor Networks in the Presence of Multiple Applications", Proc. of the IEEE International Conference on Pervasive Computing and Communications (PerCom 2003).

[C03] M. Conti, "Body, Personal, and Local Wireless Ad Hoc Networks", Chapter 1 of *Handbook of Ad Hoc Networks* (M. Ilyas Editor), CRC Press, New York, 2003.

[C03] S. Chandra, "Wireless Network Interface Energy Consumption Implications for Popular Streaming Formats" , Multimedia Systems, Volume 9, N. 2 (August 2003).

[C04] M. Conti, "Wireless Communications and Pervasive Technologies", Chapter 4 of *Smart Environments: Technologies, Protocols and Applications* (Diane Cook and Sajal K. Das, Editors), John Wiley, 2004.

[CBB+02] E.-Y. Chung, L. Benini, A. Bogliolo, Y.-H. Lu and G. De Micheli, "Dynamic Power Management for Nonstationary Service Requests", *IEEE Transactions on Computers*, Vol. 51, No. 11, pp. 1345-1361, Nov. 2002.

[CBBD99] E.-Y. Chung, L. Benini, A. Bogliolo, and G. De Micheli, "Dynamic Power Management Using Adaptive Learning Tree", *Proc. of the IEEE Int.'l Conference on Computer Aided Design*, pp. 274-279, Nov. 1999.

[CCL03] I. Chlamtac, M.Conti, J. Liu, "Mobile Ad hoc Networking: Imperatives and Challenges", *Ad Hoc Network* Journal, Vol.1 N.1 January-February-March, 2003.

[CFE99] B. Calder, P. Feller, and A. Eustace, "Value profiling and optimization", *J. Instruction-Level Parallelism*, vol. 1, 1999.

[CJBM02] Benjie Chen, Kyle Jamieson, Hari Balakrishnan, Robert Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks", ACM/Kluwer *Wireless Networks*, Vol. 8, N. 5, September 2002, pp. 481–494.

[CMC99] M. S. Corson, J.P. Maker, J.H. Cernicione, "Internet-based Mobile Ad Hoc Networking", *IEEE Internet Computing*, July-August 1999, pp. 63-70.

[CMD] Crossbow MICA2DOT Datasheet, available at http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/6020-0043-03_A_MICA2DOT.pdf

[CMTG04] M. Conti, G. Maselli, G. Turi, S. Giordano, "Cross-Layering in Mobile Ad Hoc Network Design", IEEE Computer, Vol. 37, N. 2, February 2004, pp. 48-51.

[CR00] C.F. Chiasserini and R.R. Rao, "Energy Efficient Battery Management", Proc. of *IEEE INFOCOM*, 2000.

[CT00] Jae-Hwan Chang, Leandros Tassiulas, "Energy conserving routing in wireless ad-hoc

networks", Proc. of IEEE INFOCOM, March 2000, pp. 22–31

[CV02] S. Chandra, A. Vahadat, "Application-specific Network Management for Energy-aware Streaming of Popular Multimedia Formats, 2002 *USENIX Annual Technical Conference,* June 2002.

[DB97] B. Das, V. Bharghavan, "Routing in ad-hoc networks using minimum connected dominating sets", Proceedings of the IEEE International Conference on Communications (ICC'97), June 1997.

[DKB95] F. Douglis, P. Krishnan and B. Bershad, "Adaptive Disk Spin-down Policies for Mobile devices", in *Proc. of the 2nd Usenix Symp. on Mobile and Location-Independent Computing*, pp. 121-137, Apr. 1995.

[DKM94] F. Douglis, P. Krishnan and B. Marsh, "Thwarting the power-hungry disk", in *Proc. of the Usenix Technical Conference*, pp. 292-306, 1994.

[DN97] M. Doyle and J.S. Newman, "Analysis of capacity-rate data for lithium batteries using simplified models of the discharge process", *J. Applied Electrochem.*, vol. 27, no. 7, pp. 846-856, July 1997.

[DWBP01] L. Doherty, B.A. Warneke, B.E. Boser, K.S.J. Pister, "Energy and Performance Considerations for Smart Dust," *International Journal of Parallel Distributed Systems and Networks*, Volume 4, Number 3, 2001, pp. 121-133.

[E02] A. Ephremides, "Energy concerns in wireless networks", IEEE Wireless Communications, Volume 9, Number 4, August 2002. pp. 48- 59

[ECPS02] D. Estrin, D. Culler, K. Pister and G. Sukhatme, "Connecting the Physical World with Pervasive Networks", *IEEE Pervasive Computing*, Vol. 1, No. 1, pp. 59-69, 2002.

[F01] L.M. Feeney, "An Energy-consumption Model for Performance Analysis of Routing Protocols for Mobile Ad Hoc Networks," *ACM/Kluwer Mobile Networks and Applications (MONET)*, Volume 6, Number 3, June, 2001.

[F04] L. Feeney, "Energy efficient communication in ad hoc wireless networks", in *Ad Hoc Networking* (S. Basagni, M. Conti, S. Giordano and I. Stojmenovic Editors), IEEE Press and John Wiley and Sons, Inc., New York, 2004.

[FAW01] L. Feeney, B. Ahlgren, A. Westerlund, "Spontaneous Networking: An Application-oriented approach to Ad Hoc Networking", *IEEE Communications Magazine*, June, 2001.

[FPS02] J.Flinn, S.Y. Park, M.Satyanarayanan, "Balancing Performance, Energy, and Quality in Pervasive Computing", Proceedings of the *IEEE International Conference on Distributed Computing Systems* (ICDCS'02), Wien (Austria), July 2002.

[FRM01] K. Flautner, S. Reinhardt, T. Mudge, "Automatic Performance-setting for Dynamic Voltage Scaling, Proc. of the *ACM International Conference on Mobile Computing and Networking* (Mobicom 2001), Rome (I), July 2001.

[FS99] J. Flinn, M. Satyanarayanan, "Energy-aware Adaptation for Mobile Applications", Proc. of the *ACM Symposium on Operating System Principles*, December 1999.

[FZ94] G. H. Forman, J. Zahorjan, "The Challenges of Mobile Computing", *IEEE Computer*, Vol. 27, No. 4., pp. 38-47, 1994

[G01] F. Gruian, "Hard Real-time Scheduling for Low Energy Using Stochastic Data and DVS Processors", Proc. of the *International Symposium on Low Power Electronics and Design* (ISPLED 2001), Huntington Beach (CA), August 2001.

[GBS+95] R. Golding, P. Bosh, C. Staelin, T. Sullivan and J. Wilkes, “Idleness is not sloth”, in *Proc. of the Usenix Technical Conference*, pp. 201-212, Jan. 1995.

[GCNB03] J. Gomez, A. Campbell, M. Naghshineh and C. Bisdikian, “PARO: Supporting Dynamic Power Controlled Routing in Wireless Ad Hoc Networks”, ACM/Kluwer *Wireless Networks*, 2003.

[HCB00] W. Heinzelman, A. Chandrakasan and H. Balakrishnan, “Energy-Efficient Communication Protocol for Wireless Microsensor Networks," Proceedings of the 33rd *Hawaii International Conference on System Sciences* (HICSS'00), January 2000.

[HDD98] J.H. Harreld, W. Dong and B. Dunn, “Ambient pressure synthesis or aerogel-like vanadium oxide and molybdenum oxide”, *Materials Research Bullettin*, Vol. 33 No. 4, 1998.

[HKB01] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, “Adaptive Protocols for Information Dissemination in Wireless Sensor Networks”, Proc. 5th Ann. *Intl. Conf. on Mobile Computing and Networking*, pages 174–185, Seattle, WA, August 2001.

[HLS96] D.P. Helmbold, D.E. Long, B. Sherrod "A Dynamic Disk Spin-down Technique for Mobile Computing", Proceedings of the Second Annual *ACM International Conference on Mobile Computing and Networking*, NY, pp. 130 - 142, November 1996.

[HW95] M.Herbster and M.K.Warmuth, “Tracking the Best Expert”, in the *Proceedings of the Twelfth International Conference on Machine Learning*, (Tahoe City, CA), pp. 286-294, Morgan Kaufmann, 1995.

[HW97] C.-H. Hwang and A. Wu, “A Predictive System Shutdown Method for Energy Saving of Event-driven Computation”, in *Proc. of Int.’l Conference on Computer Aided Design*, pp. 28-32, Nov. 1997.

[IG00] T. Imielinski and S. Goel, “DataSpace: Querying and Monitoring Deeply Networked Collections in Physical Space”, IEEE Personal Communications, pp. 4-9, Oct. 2000.

[IGE03] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. Directed Diffusion for Wireless Sensor Networking, *IEEE Transaction on Networking*, February 2003.

[IPAQ03] Hp iPAQ Pocket PC h2200 Series Datasheet, available at http://www.hp.com/hpinfo/newsroom/press_kits/2003/ipaq/ds_h2200.pdf.

[J00] A. Joshi, “On Proxy Agents, Mobility, and Web Access”, *ACM/Baltzer Mobile Networks and Applications*, Vol. 5 (2000), pp. 233-241.

[JSAC01] J C. Jones, K. Sivalingam, P. Agarwal, J.C. Chen, “A Survey of Energy Efficient Network Protocols for Wireless and Mobile Networks”, *ACM/Kluwer Wireless Networks* (WINET), vol. 7, No. 4, 343-358, 2001.

[KB02] R. Krashinsky and H. Balakrishnan, “Minimizing Energy for Wireless Web Access with Bounded Slowdown”, Proceedings of the *ACM International Conference on Mobile Computing and Networking* (Mobicom 2002), 2002.

[KK00] R. Kravets and P. Krishnan, “Application-driven Power Management for Mobile Communication”, *ACM/Baltzer Wireless Networks* (WINET), Vol. 6 (2000), pp.263-277.

[KK98] R.Kravets and P.Krishnan, “Power Management Techniques for Mobile Communication”, Proceedings of *the Fourth Annual ACM/IEEE International*

Conference on Mobile Computing and Networking (Mobicom'98).

[KL00] C. M. Krishna, Y. H. Lee, "Voltage-clock-scaling Techniques for Low Power in Hard Rela-time Systems", Proc. of the *IEEE Real-time Technology and Applications Symposium*, Washington D.C., May 2000, pp. 156-165.

[KLV95] P. Krishnan, P. Long and J. Vitter, "Adaptive Disk Spindown via Optimal Rent-to-buy in Probabilistic Environments", *Int.'l Conf. Machine Learning*, pp. 322-330, July 1995.

[KMMO94] A. Karlin, M.S. Manasse, L.A. McGeoch and S. Owicki, "Competitive randomized algorithms for non-uniform problems", in *Algorithmica*, Vol. 11, No. 6, pp. 542-571, June 1994.

[KVIY01] M. Kandemir, M. Vijaykrishnan, M. Irwin and W. Ye, "Influence of compiler optimizations on system power" *IEEE Transactions on VLSI Systems*, Vol. 9, No. 6, pp. 801-804, Dec. 2001.

[KW03] Holger Karl, Andreas Willig, "A short survey of wireless sensor networks", TKN Technical Report TKN-03-018.

[LAR01] Qun Li, Javed Aslam, Daniela Rus, "Online power-aware routing in wireless ad-hoc networks", Proceedings of *ACM International Conference on Mobile Computing and Networking* (MOBICOM 2001), July 16-21, 2001, Rome, Italy.

[LBD02] Y.-H. Lu, L. Benini and G. De Micheli, "Dynamic Frequency Scaling With Buffer Insertion for Mixed Workloads", *IEEE Transactions on CAD-ICAS*, Vol. 21, No. 11, Nov. 2002.

[LBS] Laptop Battery Store site, at http://shop.store.yahoo.com/batteryjuice/.

[LH01] L. Li and J. Y. Halpern, "Minimum-energy mobile wireless networks revisited", Proceedings of the *IEEE International Conference on Communications* (ICC01), pp. 278–283, 2001.

[LKHA94] K. Li, R. Kumpf, P. Horton, T. Anderson, "A Quantitative Analysis of Disk Drive Power Management in Portable Computers", *Proc. of the 1994 Winter Usenix Conference*, pp. 279-291, Jan 1994.

[LLM+02] E. L. Lloyd R. Liu M. V. Marathe R. Ramanathan, S. S. Ravi, "Algorithmic Aspects of Topology control problems for ad hoc networks", Proceedings of the ACM International Symposium on Mobile Ad hoc Networking & Computing, Lausanne (CH), 2002.

[LNT02] Henrik Lundgren, Erik Nordström, Christian Tschudin, "Coping with Communication Gray Zones in IEEE 802.11b based Ad hoc Networks", Proc. of the Fifth International Workshop on Wireless Mobile Multimedia (WoWMoM 2002), Sept. 2002.

[LS01] J.R. Lorch, A.J. Smith, "Improving Dynamic Voltage Scaling with Pace", Proc. of the *ACM Sigmetrics 2001 Conference*, Cambridge (MA), June 2001, pp. 50-61.

[LS97] J.R. Lorch, A.J. Smith, "Scheduling Techniques for Reducing Processor Energy Use in MacOS", *ACM/Baltzer Wireless Networks*, pp.311-324, 1997.

[LS98] J.R. Lorch, A.J. Smith, "Software Strategies for Portable Computer Energy Management", *IEEE Personal Communications*, Vol. 5, No. 3, pp. 60-73, June 1998.

[MACM00] D. Mosse, H. Aydin, B. Childers, R. Melhem, "Compiler-assisted Dynamic Power-aware Scheduling for Real-time Applications", Proc. of Workshop on Compiler and Operating Systems for Low Power (COLP 2000), Philadelphia (PA), October 2000.

[MCD+03] S. Mohapatra, R. Cornea, N. Dutt, A. Nicolau, N. Venkatasubramanian, “Integrated Power Management for Video Streaming to Mobile Handheld Devices”, Proceedings of the ACM International Conference on Multimedia 2003, Berkeley (CA) November 2 - 8, 2003.

[MFHH03] S.R. Madden, M.J. Franklin, J.M. Hellerstein, W. Hong, “The Design of an Acquisitional Query Processor for Sensor Networks”, Proc. SIGMOD, June 2003.

[MMA+01] S. Meninger, J.O. Mur-Miranda, R. Amirtharajah, A.P. Chandrakasan and J.H. Lang, “Vibration-to-Electric Energy Conversion”, *IEEE Trans. on VLSI Systems*, Vol.9, No.1, 2001.

[MOI+97] H. Mehta, R. Owens, M. Irwin, R. Chen, D. Ghosh, “Techniques for Low Energy Software”, *International Symposium on Low Power Electronics and Design*, pp. 72-75, Aug 1997.

[MT] Maxwell Technologies, at http://www.maxwell.com/.

[NKSK02] S. Narayanaswamy, V. Kawadia, R. S. Sreenivas, and P. R. Kumar, “Power control in ad-hoc networks: Theory, architecture, algorithm and implementation of the COMPOW protocol”, Proceedings of *European* Wireless 2002, pp. 156–162, February 2002.

[PCD+01] D. Panigrahi, C.F. Chiasserini, S. Dey, R.R. Rao, A. Raghunathan and K. Lahiri, “Battery Life Estimation of Mobile Embedded Systems”, *Int'l Conference on VLSI Design,* 2001.

[PCS02] B.J. Prahbu, A. Clockalingam and V. Sharma, “Performance Analysis of Battery Power Management Schemes in Wireless Mobile Devices”, *Proc. of the IEEE Wireless Communication and Networking Conference (WCNC’02),* 2002.

[PEBI01] S.H. Phatak, V. Esakki, R. Badrinath and L. Iftode, “Web&: An Architecture for Non-Interactive Web”, in *Proc. of the IEEE Workshop on Internet Applications,* (*WIAPP'01*), July 2001.

[PS01] P. Pillai and K.G. Shin, “Real-Time Dynamic Voltage Scaling for Low-Power Embedded Operating Systems”, *Proceedings of the 18th Symposium on Operating Systems Principles (SOSP’01),* 2001.

[PS02] C. Poellabauer and K. Schwan, “Power-Aware Video Decoding using Real-Time Event Handlers”, *Proc. of the 5th ACM Int’l Workshop on Wireless Mobile Multimedia* (*WoWMoM’02*), pp. 72-29, Sep. 2002.

[PS98] E. Pitoura, G. Samaras, Data Management for Mobile Computing, Kluwer Academic Publishers, 1998.

[PSS01] S. Park, A. Savvides and M.B. Srivastava, “Battery Capacity Measurement and Analysis using Lithium Coin Cell Battery”, *ACM Int'l Symposium on Low Power Electronic Design (ISLPED '01),* August 2001.

[QP00] Q. Qiu and M. Pedram, “Dynamic power management of complex systems using generalized stochastic petri nets” in *Proc. Design Automation Conf.*, pp. 352–356, June 2000.

[QP99] Q. Qiu and M. Pedram, “Dynamic power management based on continuous-time Markov decision processes” in *Proc. Design Automation Conf.*, pp. 555–561, June 1999.

[R04] R. Ramanathan “Antenna Beamforming and Power Control for Ad Hoc Networks” in

Ad hoc networking (S. Basagni, M. Conti, S. Giordano and I. Stojmenovic Editors), IEEE Press and John Wiley and Sons, Inc., New York, 2004.

[RB96] M.Rulnick, N. Bambos "Mobile Power Management for Wireless Communication Networks", *Wireless Networks*, Vol. 3, No. 1, Mar. 1996.

[RM99] Volkan Rodoplu, Teresa H.-Y. Meng "Minimum energy mobile wireless networks", IEEE Journal on Selected Areas in Communications, 17(8), August 1999, pp. 1333–1344.

[RRH00] R. Ramanathan, R. Rosales-Hain, "Topology control of multi-hop wireless networks using transmit power adjustment", *Proceedings of IEEE INFOCOM* 2000, Tel Aviv, Israel (March 2000).

[RSPS02] V. Raghunathan, C. Schurgers, S. Park and M.B. Srivastava, "Energy-Aware Wireless Microsensor Networks", *IEEE Signal Processing Magazine*, pp. 40-50, March 2002.

[S03] T. E. Starner, "Powerful Change Part 1: Batteries and Possible Alternatives for the Mobile Market", *IEEE Pervasive Computing*, October-December 2003, pp. 86-88.

[S93] M. Satyanarayanan et al., "Experience with Disconnected Operation in a Mobile Computing Environment", in *Proc. of Usenix Mobile and Location-independent Computing Symposium*, pp. 11-28, Aug. 1993.

[SAA03] Y.Sankarasubramaniam, O. B. Akan, I. F. Akyildiz, "ESRT: event-to-sink reliable transport for wireless sensor networks", Proceendings of ACM MobiHoc'03, Annapolis, Maryland, June 2003.

[SBGD01] T. Simunic, L. Benini, P. Glynn and G. De Micheli, "Event-driven Power Management", *IEEE Transactions on CAD-ICAS*, Vol. 20, No 7, July 2001.

[SC00] V. Swaminathan, K. Chakrabarty, "Real-time Task Scheduling for Energy-aware Embedded Systems", Proc. of the IEEE Real-time Systems Symposium, Orlando (FL), November 2000.

[SCB96] M. Srivastava, A. Chandrakasan and R. Brodersen, "Predictive System Shutdown and Other Architectural Techniques for Energy Efficient Programmable Computation", *IEEE Transactions on VLSI Systems*, Vol. 4, pp. 42-55, Mar. 1996.

[SK97] Stemm, M. and Katz, R. H. (1997) Measuring and reducing energy consumption of network interfaces in handheld devices. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science*, Special Issue on Mobile Computing, Vol. 80, No. 8, 1125-1131.

[SL00] I. Stojmenovic, X. Lin, "Power-Aware Localized Routing in Wireless Networks," Proc. IEEE Symp. Parallel and Dist. Processing Sys., May 2000.

[SR03] P. Shenoy, P. Radkov, "Proxy-assisted Power-friendly Streaming to Mobile Devices", Proceedings of SPIE -- Volume 5019, *Multimedia Computing and Networking 2003*, January 2003, pp. 177-191

[STD94] C. Su, C. Tsui, A. Despain, "Saving Power in the Control Path of Embedded Processors", *IEEE Design and Test of Computers*, vol. 11, no. 4, pp. 24--30, Winter, 1994.

[STGM02] C. Schurgers, V. Tsiatsis, S. Ganeriwal, M. Srivastava, "Optimizing Sensor Networks in the Energy-Latency-Density Design Space", *IEEE Transactions on Mobile Computing*, Vol 1. N.1, 2002.

[T01] C.-K. Toh, "Maximum battery life routing to support ubiquitous mobile computing in

wireless ad hoc networks", *IEEE Communications Magazine*, 39(6), June 2001.

[TMCP01] D. Teasdale, V. Milanovic, P. Chang and K.S.J. Pister, "Microrockets for Smart Dust", *Smart Materials and Structures*, Vol. 10, pp. 1145-1155, 2001.

[TMW94] V. Tiwari, S. Malik, A. Wolfe, "Power Analysis of Embedded Software: A First Step Towards Software Power Minimization", *IEEE Transactions on VLSI Systems*, vol. 2, no.4, pp.437--445, Dec. 1994.

[US96] S. Udani and J. Smith, "The Power Broker: Intelligent Power Management for Mobile Computing", Tech. Rep. MS-CIS-96-12, Dept. of Computer Information Science, University of Pennsylvania, May 1996.

[VA03] M.C. Vuran, I.F. Akyildiz, "Correlation-based Collaborative Medium Access in Wireless Sensor Networks", November 2003. http://www.ece.gatech.edu/research/labs/bwn/publications.html.

[VL03] T. van Dam, and K. Langendoen, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks", ACM SenSys, Nov. 2003.

[W91] M. Weiser, "The Computer for the Twenty-First Century", Scientific American, September 1991.

[W96] M. Wolfe, "High Performance Compilers for Parallel Computing". Reading, MA: Addison-Wesley, 1996.

[WC02] IEEE Wireless Communications, Special issue on "Energy-Aware Ad Hoc Wireless Networks", Volume 9, Number 4, August 2002.

[WCK02] C. Y. Wan, A. T. Campbell and L. Krishnamurthy, "PSFQ: A Reliable Transport Protocol for Wireless Sensor Networks," In Proc. WSNA'02, September 2002, Atlanta, GA

[WDGS02] Jie Wu, Fei Dai, Ming Gao, Ivan Stojmenovic, "On calculating poweraware connected dominating sets for efficient routing in ad hoc wireless networks", IEEE/KICS Journal of Communications and Networks, 4(1), March 2002, pp. 59–70.

[WEC03] C.-Y. Wan, , S.B. Eisenman, A.T. Campbell, "CODA: Congestion Detection and Avoidance in Sensor Networks," in Proc. ACM SENSYS 2003, November 2003.

[WESW98] Hagen Woesner, Jean-Pierre Ebert, Morten Schlager, and Adam Wolisz. Power saving mechanisms in emerging standards for wireless LANs: The M level perspecitve. IEEE Personal Communications, 5(3):40-48, June 1998.

[WL99] J. Wu and H. Li, "On calculating connected dominating set for efficient routing in ad hoc wireless networks", Proceedings of the Third International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, Seattle, WA (August 1999).

[WLBW01] Roger Wattenhofer, Li Li, Paramvir Bahl, Yi-Min Wang, "Distributed topology control for wireless multihop ad-hoc networks" Proc. of IEEE Infocom, pages 1388–1397, April 2001.

[WWDS94] M. Weiser, B. Welch, A. Demers and S. Shenker, "Scheduling for Reduced CPU Energy", in *Proc. of the First Symposium on Operating Systems Design and Implementation* (*OSDI*), pp. 13-23, Nov. 1994.

[WWRF] Wireless World Research Forum (WWRF): http://www.ist-wsi.org.

[XHE00] Ya Xu, John Heidemann, Deborah Estrin. "Adaptive energy conserving routing for

multihop ad hoc networks", Technical Report 527, USC/Information Sciences Institute, October 2000.

[XHE01] Ya Xu, John Heidemann, Deborah Estrin, "Geography-informed energy conservation for ad hoc routing", Proc. of 7th Annual International Conference on Mobile Computing and Networking, pages 70–84, July 2001.

[YHE02] W. Ye, J. Heidemann and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks", Proc. of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002), New York, NY, USA, June 2002.

[ZD04] G. Zaruba, S. Das, "Off-the-Shelf Enablers of Ad Hoc Networks", in *Ad hoc networking*, S. Basagni, M. Conti, S. Giordano, I. Stojmenovic (Editors), IEEE Press and John Wiley and Sons, Inc., New York, 2004.

[ZR96] M.Zorzi e R.R.Rao, "Energy Constrained Error Control for Wireless Channels", *Proceeding of IEEE GLOBECOM '96*, pp.1411-1416, 1996.