# System Support for Energy Management in Mobile and Embedded Workloads: A White Paper

*Carla Ellis, Alvin Lebeck, and Amin Vahdat*

Department of Computer Science
Duke University
Durham, NC 27708–0129

October 1999

## 1 Introduction

One of the major challenges of the post-PC environment in which mobile and embedded devices are ubiquitous is the need to reduce the energy consumed in using these devices, thereby extending the lifetime of the batteries that power them as well as reducing the impact of the heat and noise produced by their operation. Energy efficiency of computers is desirable in its own right from both the economic and environmental points of view, especially as computing equipment continues to rapidly proliferate throughout our society.

In the near future, there will be broadband wireless coverage in many areas. Sustained exponential growth in processor performance and memory density means that embedded processors and handheld devices will have performance characteristics comparable to today's workstations. We believe that the availability of a ubiquitous network infrastructure along with powerful processing elements distributed throughout our everyday environment will enable a number of interesting applications. However, the increased dependence on these devices and their increased capabilities comes with a potential cost: increased energy consumption, unless research actively addresses the issue of energy efficiency. To motivate our research, we discuss the vulnerability of one such application to power consumption:

***Disaster Recovery:*** In this scenario, response teams will work to contain situations involving, for example, environmental hazards, natural disasters, or fires. There are a number of critical aspects to this scenario. First, a command center will be responsible for coordinating the efforts of field agents (i.e., firemen, police, the Red Cross, etc.). It will be necessary to know which personnel are available and where they are located at any given time. Next, the agents require mobile access to important data such as street maps, population density, building layout and up-to-date capacity figures of local hospitals. They also need to produce information from on-the-ground observation (e.g. impassable roads) and have access to remote computational resources for predictions of what might happen next (e.g. where fire is likely to spread, where flooding will occur) given field inputs. Emergency generators will adequately supply the command center with electricity, but the agents have no assurance of finding electrical service in the field and must rely on battery power for all their needs. In the near future, it should be possible to quickly deploy a wireless network infrastructure over target zones even when the established infrastructure is damaged. Thus, agents will be able to communicate among themselves and the command center through multi-purpose hand held devices, incorporating the functionality of a cellular phone, a global positioning system, and a personal digital assistant. Small digital video cameras with wireless connectivity and an embedded processor may be deployed throughout an affected zone for gathering important visual data. The viewing angle of these cameras can be controlled remotely (perhaps through an HTTP server running on each camera) with transmissions from select cameras used to deploy available resources. Similarly, sensors with processors and wireless connectivity can be deployed to detect the presence of certain chemicals, heat, or water. These sensors may be placed in hazardous locations and not be serviceable once placed, so the longevity of their operation is limited by the lifetime of their batteries.

The above application embodies a set of challenges that drive our research agenda. Power management, both at the architectural and software levels, will be critical as sensors and handheld devices cannot fail at key moments in a scenario such as disaster recovery. The functionality required in this scenario becomes more demanding just when resources become more limited or unreliable. Remote access to data requiring the composition of complex queries and access to remote computational resources motivates the need for mobile and location-independent code. The diversity of devices and links over which data may be accessed and displayed calls for ways to adapt the presentation and transmission appropriately. The significance of proximity or location (i.e., knowing where one is and what resources are nearby) raises issues of resource discovery and the need for a geographic addressing model. These technologies that are required to contribute to the functionality of the target applications also have potential roles for power management which we will highlight in later sections of this white paper.

The goal of our project is to explore the architectural and software systems challenges to increasing the energy efficiency for such applications. The novel angle of this project is its top-down, application-driven flavor, in contrast to much of the previous work in the field. The top-down approach has the following potential advantages: i) it acknowledges the different qualitative characteristics of applications built specifically for mobile and embedded devices, as opposed to desktop applications, ii) it opens up the possibility of adaptivity within the application itself and ability to exploit application-specific knowledge that may have subtle power implications, and iii) its explicit goal is to identify and attack the source of highest energy use first given the usage patterns of the application. By contrast, the bottom-up approach may focus on a power-hungry component of the system and reduce its power consumption. However, since energy consumption depends on both the power and the time spent using it, unless the component in question is actually active during the execution of the particular tasks our device performs, it did not play a part in the energy used. Thus, those optimization efforts targeted at that component may not realize the hoped for payoffs.

We focus on the following specific directions:

- *Experimental Toolkits and Measurement Methodology:* Understanding the power consumption profile of these devices and the unique characteristics of these workloads are prerequisites for targeting research efforts. We are developing representative workloads that are appropriate for performance evaluation studies of mobile, wireless, and embedded systems. We are building a tool package for power measurement that is analogous to debugging or performance monitoring tools. The goal is to identify the areas in a system design that will yield the biggest payoff in energy savings by focusing optimization efforts there. We are developing a common simulation framework. Simulation studies will precede any full-scale implementation efforts to first assess the feasibility and sensitivity of proposed techniques. The simulation approach is essential for studying novel architectural features.

- *Energy-Aware APIs:* By exposing power management mechanisms to software and by exploiting information from applications, we intend to raise power to a first-class operating system managed resource. We must determine what is needed in the programming interface to empower applications to affect energy usage. We will design and prototype an energy-aware API and use the experience of developing applications with it to evaluate its effectiveness.

- *System Support for Energy Management:* A key component of our project is to design and evaluate the systems support necessary to enable sophisticated power management policies to be implemented in software. This work encompasses both application-driven and hardware/software cooperative power management mechanisms. We are investigating policies and mechanisms to utilize mobile code and remote resources, to take advantage of low power modes of various (existing or proposed) hardware components, to exploit and enhance idleness in workload behavior, and to adapt to varying conditions in the operating environment.

## 2 A Preliminary Case Study

In our preliminary investigations of this emerging research area, we explored opportunities for application-based adaptation for energy conservation in the context of a concrete example. Our first-hand experiences allowed us to identify many shortcomings with existing operating systems but suggested promising energy savings by allowing the application knowledge to be exploited. These insights were reported on at the recent HOTOS Workshop [8].

As an exercise intended to gain an appreciation for the system support needs of mobile applications developers, one of the PIs (Ellis) chose to build an application for the Palm PDA. Hiker's Buddy takes NMEA protocol strings sent by a Global Positioning System (GPS) receiver and plots the hiker's current location on a topographical map found by searching a database of map segments that have been transcoded for the Palm's limited display and then downloaded from a server into the PDA prior to starting out on the hike. This application is a particularly appealing vehicle for study of PDA application development: it is necessarily mobile and battery-powered; the small form factor and light weight of the device are assets to this task and not simply constraints of today's technology; it involves an external, also battery-powered peripheral; and it is a natural for disconnected operation although plausible roles for intermittent wireless connectivity present themselves. It is rich in issues to explore.

In particular, Hiker's Buddy was ideal as a vehicle for investigating application-directed power management. Many of the first problems that arose in programming Hiker's Buddy had power-management implications. For example, there was an incompatibility between control of the GPS function through the serial line and the PalmOS default power management strategy for putting the device into sleep mode after a pause in user input events.

Solving power management problems requires an understanding of what different operations cost in terms of energy consumption. Thus, we designed an experimental methodology to quantify the power consumption of the Palm device, including both generic operations and functions specific to Hiker's Buddy, such as interactions with the GPS unit. The measurements were done by developing a driver application that put the Palm into a stable power state and then measuring current with a multimeter during each test period. Substituting a power supply for the batteries ensured constant voltages. In order to evaluate the power usage of specific algorithms, a program module implementing the algorithm could be dropped into the driver, allowing its power consumption to be measured during execution. This framework has served as a starting point for developing more general measurement tools (see next section).

Finally, we were interested in seeing whether design decisions made at the application-level could have a substantial impact on the energy consumption of using Hiker's Buddy. Intuitively, applications should hold an important key, in the application-specific constraints and opportunities for saving energy that can be known only at that level. To test this, we instrumented Hiker's Buddy to capture the time spent in various power states during execution, which combined with the measured power levels, allowed an estimation of the energy expended. We designed experiments comparing different design decisions at the application-level that might have an impact on energy efficiency. Among the issues we have investigated are different map search algorithms and database structures in memory, variations on the polling frequency for the GPS data, and adaptive strategies for managing continuous use of the GPS receiver. Our experience convinced us that the application designer can, in fact, make a difference if given the tools to tune the energy behavior of the application program. This firsthand experience validated our higher-level perspective on the energy use problem.

In the Hiker's Buddy exercise, we were often frustrated in our attempts to tailor the energy use to the specific needs of the application by the existing weak support for energy tuning: a lack of feedback on what effect various design decisions might have on energy use, an inadequate programming model of power consumption in the target platform, a gap between our vision of how the application should control its energy use and the ability to express that to the system, and limitations on the flexibility of interactions between the operating system and the given architecture. Identifying and alleviating these drawbacks in the state-of-the-art of energy conservation has motivated our current and future projects.

## 3 Experimental Tools

The first requirements are to understand how devices currently use power in order to form a baseline model to direct our efforts and to identify the energy needs of those applications we envision running on such devices. This implies an ability to target the "most common case" behaviors and to identify the "power hogs" in a system both in terms of device components and usage patterns. Thus an important research activity becomes the development of *experimental tools and measurement methodologies* for evaluating designs and providing feedback to applications developers on the power consumption impact of design decisions.

The following activities support the evaluation needs of this project:

**Development of a representative workload.** Simulation studies and controlled measurement experiments of systems yield results only as good as the workloads that drive them. There are benchmark suites and collections of trace data that have served the systems research community and gained the status of "standards", but they capture computational behavior radically different from our expectations of "typical" behavior for mobile and embedded devices. In current studies, we are using one set of traces for desktop productivity applications [13], available from [25], as respresentative of a laptop workload. We are adopting some of the C3I Non-realtime benchmarks [1] for their similarity to some of the processing of sensor data that would arise in our disaster recovery scenario.

In addition, we plan to gather a new set of workloads designed specifically for palmtops and handheld devices, including some that creatively exploit wireless communication or interesting peripherals such as GPS receivers from which arise some of the most exciting uses of small battery-powered devices. We have source code for many of the most popular Palm applications and we have a number of locally developed applications to broaden the collection. Many of the applications were generated as a result of a project-oriented Topics course [7] offered by one of the PIs (Ellis) during Spring semester 1999. These include prototypes of embedded applications such as a trip logging tool to be mounted in an auto dashboard for recording various sensor data, a web-based stock trading application using agents for disconnected access, and support for private authenticated beaming. The exercise of designing new applications helps to pinpoint the needs of programmers who wish to express power interactions appropriate to their application domain. Generating a home-grown workload is an activity that can naturally involve students at both the graduate and undergraduate level. The creativity of a large number of students can be harnessed to benefit this research while providing educational benefits to them and a sense of contributing to a larger research effort.

Traces derived from such applications (at the granularity of system calls and events) will serve as a benchmark suite for simulations and other analyses and will be made available to the research community for related studies. We will initially instrument the source code of selected applications while finalizing the details of the event tracing (e.g. what information is required), then we will investigate the use of "Hackmaster" intercepts for PalmOS code to trace significant events on the Palm in a more automatic fashion. Finally, we will investigate instrumenting code of other "typical" applications running on platforms for which we have appropriate binary rewriting tools to obtain finer grain traces (e.g. memory references).

**Development of a measurement methodology and tools for power consumption.** One of the prerequisites for making progress in energy conservation by software systems is giving programmers a better understanding of how energy is consumed by their applications running on a particular device. We are building a tool package for power measurement analogous to debugging or performance monitoring tools. The goal, of course, is to identify the areas in a design to focus optimization efforts. Two challenges are especially important in the design of our tool: i) capturing the cause-and-effect appropriately so that observations of high energy use are properly assigned to devices or semantic constructs in the program that can serve as targets for optimization, and ii) low interference by the measurement setup in the power used by the system under test. Our starting point involves generalizing and improving upon the techniques employed fairly successfully in the preliminary work on the Palm PDA.

The power state model embedded in the measurement tool was influenced by several considerations: i) implementation - the ability to capture a trace and correlate the events with current measurements, ii) causality - the ability to identify events that can be viewed as responsible for changes in power levels, and iii) usability for the programmer - the ability to map the measured events to meaningful program behavior. This should provide a framework for formulating power reduction strategies: within a state there is a choice of optimizing time spent there or the underlying technology that determines the power consumed. Since our model bears some resemblance to the instruction-level power estimation model of [24] (at a different level of abstraction, but for similar experimental reasons), we plan to pay particular attention to its development and use. Integrating these two models may yield a hierarchical approach. We need to validate that this model, developed for convenience of the empirical power measurement methodology and based on traceable software constructs, can be generally useful. One current activity of our group is porting our measurement infrastructure from the Palm platform to WinCE platforms. This effort will strengthen our case for the generality of our approach.

This work is related to the PowerScope toolkit [9] which combines hardware instrumentation with CPU statistical profiling techniques to map power consumption to program structure. In their method, the multimeter interrupts the processor under test which seems intrusive and assuming a causal relationship between program counter and measured power may be problematic. Our approach is qualitatively different in the relationship between the multimeter and device (i.e. triggering method) and that the events traced represent power state transitions at a certain level of abstraction. This allows us to account for power consumption levels caused by an earlier, perhaps unrelated, activity but still in effect and observable (e.g. an earlier disk spinup for which the timeout has not yet expired).

**Simulation Infrastructure for Energy Analysis** To complement the above power profiling tools, we will also develop a simulation environment that allows us to investigate novel power management mechanisms and their impact on overall system energy efficiency. The proposed simulation infrastructure will allow complete system simulation, like SimOS [20], but with an emphasis on power consumption.

Our infrastructure will augment the traditional performance models for each system component (e.g., display, CPU, memory hierarchy, disk, network interface) with analogous power consumption models. Developing these models is a major undertaking, and we expect our models to evolve over time. Initial models can be based on technical specifications and empirical data obtained using the above profiling tools and microbenchmarks. Subsequent models can be refined to include more detailed information and new power management mechanisms. Furthermore, the specific mobile device can have significant impact on overall power consumption. For example, the Intel Pentium used in laptops has very different power characteristics than the Motorola DragonBall used in the Palm. Therefore, it is crucial that our simulation infrastructure support multiple models for each system component.

## 4 Energy-Aware Application Interface

An important aspect of our research is to bring the issue of energy efficiency to the higher levels of system design, including allowing applications to direct the power management policies. A key requirement is to provide applications with an API that allows expression of application-specific energy management policies and exposes power-related information through the OS to the application.

Our experience with Hiker's Buddy provided promising results at tuning energy consumption with minimal OS support. However, the system interface stifled further efforts to fully exploit the power savings opportunities offered by our application. Requirements that were straightforward to articulate could not be simply implemented because of the assumed coupling between components in the available power management system calls. For example, Hiker's Buddy had no need for the display to be on while the GPS was acquiring satellites, but this came with the requirement to keep the serial line open to the GPS. From this experience, we have begun to compile a wish-list of capabilities for a power-wise OS and what should be exposed to the applications layer. The ability to independently control individual components is a first step. There also needs to be some way to represent the power model of external devices "hiding" behind a standard interface such as the serial line. In the Palm device, separately powered devices ranging from GPS receivers to wireless modems are all serial devices. Extending the power management to incorporate the battery life of peripheral devices is needed to make a comprehensive solution. Notifications of power-related events (e.g., imminent entry into sleep mode) from the system to applications would give applications the option of adapting. Reliable reporting of energy availablity is necessary for informed decisions. Applications should be able to provide either hints or hard requirements, specify either actions to trigger wakeup from a sleeping state or actions that can be deferred until wakeup due to other causes. There must be a recognition of foreground and background activity since background activities have residual power needs. Each of these features are motivated and could have been directly exploited by our example application. We are planning to prototype an energy-aware API to allow further evaluation of the utility, drawbacks, and omissions in this set of features.

The key point is that reducing energy consumption should be raised to first-class status among the performance goals of OS design and application-involvement should be encouraged from the outset by the mechanisms available. Furthermore, low-power architecture design should provide the appropriate mechanisms for effective operating system and application needs.

# 5 System Support

The final goal of this research is to provide the system support needed to implement energy-efficient policies at the higher-levels of system design.

Reducing the energy consumed by applications on small devices can be addressed at various levels: by improving battery technology, by engineering more efficient electronics and components, and by designing computer architectures and software with power as a primary measure of performance. Our research focuses on the latter approach in which the needs of the applications become the driving force for the development of power management functions in the operating system and architecture.

We begin by considering coarse-grained strategies that delegate power-hungry work to remote, energy-rich resources, thereby conserving local battery power. Then, we address tasks for which that choice is not viable and which necessarily invoke local processing and consume local power. We consider more energy-efficient ways to perform these tasks, including more fully exploiting hardware power modes.

## 5.1 Dynamic Recruitment of Remote Resources

Mobile code is emerging as the enabling technology for transparent access to Internet resources. An important aspect of this project is to leverage our previous work in mobile code to support intelligent program placement to improve both performance and energy consumption.

### 5.1.1 Previous Work: WebOS

One of the PI's (Vahdat) earlier research investigated wide-area persistence, naming, and security for wide-area applications given the availability of remotely programmable computational resources. WebOS [26] provides access to global system resources by adapting operating system abstractions to the unique requirements of wide-area applications. The principal goal of this research was to provide a convenient set of abstractions for access to remote resources.

There are a number of similarities between wide-area and mobile/embedded systems—for instance, highly unpredictable network and computational characteristics and the associated need to use mobile code to address such variability. Thus, many of the earlier results developed within the context of wide-area systems are directly applicable to the applications targeted by this current project. Contributions of the individual WebOS components which are relevant to this work are summarized below.

- *Cache Coherent Global Namespace:* To support transparent access to remote resources, applications require consistent access to global data, independent of where the data is stored or replicated. To this end, WebFS [26] implements a global file system that supports flexible caching policies for different application classes.

- *Locating Wide-Area Resources:* As programs and data migrate and are replicated across the network, one challenge is to locate the instance of a resource able to deliver the best quality of service to a given client. For wide-area and mobile systems, tracking the current location of objects and services is another challenge. The performance available from a server is dependent on both server load and the characteristics of the connection between the server and client, both of which can change rapidly over time. Active Names [27] allow location-independent programs to perform name resolution in a client and service-specific manner. A key goal of this project is to develop techniques for placing mobile code across both mobile and wide-area systems in a scalable fashion.

### 5.1.2 Support for Placement of Mobile Code

Our target applications depend upon energy efficient and location-independent access to Internet resources. One approach is to achieve such efficiency through mobile code—programs able to dynamically migrate and replicate in response to the local and wide-area environment. Rather than statically assigning program execution to a local host (such as a PDA), the runtime environment may dynamically determine that it is more efficient to migrate a program to a remote site (e.g., with line power or with more computational resources) and to retrieve its results when execution completes. If the time/power saved on the mobile device is more than than the time/power consumed to: i) send the process, ii) idle until results can be retrieved, and iii) receive the results, then this method may be effective for conserving either time or power. Interestingly, tradeoffs may be inherently different when optimizing for time versus power. Thus, application-specific knowledge may be required to determine the proper tradeoff. For example, in a disaster-recovery situation, it may be critical to retrieve all available information quickly, independent of the power consumption costs. In other cases, timely evaluation may not be as important as maximizing battery life.

Thus, we are developing mechanisms for placement and migration of programs originating from diverse sources (e.g., handheld computers to servers). We will also develop policies to determine the conditions under which a program should migrate. Programs will cooperate with the runtime system to make more intelligent placement decisions while avoiding oscillations. One important factor that the runtime must consider is varying levels of confidence in the information available for different nodes. For example, recently acquired information from a node in the same administrative domain is likely to be considered more accurate than older information from a distant, untrusted node.

Consider the example of a program running on a handheld computer with a slow and expensive wireless connection to the global Internet. To optimize both performance and power consumption, it may be beneficial to locate a nearby workstation with available cycles and to then retrieve the results upon the program's completion. In this case, the application must evaluate the cost (both in time and energy) of transferring the computation over a slow link versus continuing to run it locally. The evaluation becomes more complex with the periodic availability of relatively high-bandwidth, energy-efficient radio links with short range (e.g., Bluetooth bandwidth is 700 kbps at up to 10 meters [2]). One possibility is for applications to wait a threshold period of time for a high speed/energy efficient link to become available before deciding it is preferable to transmit over a slower and more power-hungry link.

This study will initially use simulation techniques to evaluate various placement policies. Existing network simulators, such as ns [14], are largely designed to evaluate lower level protocol issues. Thus, we will build a custom simulator to focus on placement of computation in large with a particular focus on the energy costs associated with a particular placement policy. We will consider the computational and energy costs of individual server workloads (e.g., for dynamically generated content) and the client's ability to choose among available replicas within the Active Names framework. After this initial validation, the simulation environment will be enhanced to consider more complex scenarios involving, for example, embedded processors, wireless connections, and handheld computers.

While simulation is an important technique for conducting controlled experiments in complex systems, it is equally important to deploy the policies developed through simulation in working systems. Only a real deployment can verify that the simulator captured the necessary levels of complexity. Thus, the research will be conducted in an iterative fashion, using simulation results to implement policies within the existing Active Names prototypes ported to a wireless infrastructure.

#### 5.1.3 Case Studies of Remote Execution

**Mobile Agents.** We have performed some initial experiments on the energy-saving potential of remote execution for the Palm. Our experiments were based on an IBM Workpad running PalmOS version 3.0 and a Dell Dimension PII 400 running Sun OS 5.7. The transportation and execution of our remote processes were handled by Tcl agents [16] that were part of the D'Agents system–a mobile-agent system developed at Dartmouth College [4, 3]. D'Agents is a system already established for handling remote executions from a Palm device using a program called Pilot Launcher. Communication with the server was over a 56K modem. Measured power consumption values for the Palm device and the modem were: CPU executing a loop with register data consumed 128 mW, sending on the serial line consumed 143 mW, receiving consumed 162 mW, and allowing the CPU to idle (with display on) consumed 41mW. The modem itself consumed approximately 387 mW.

We tested two scenarios: i) the sizes of the data sent and the results returned were independent of the amount of computation to be performed which was varied by specifying a number of loop iterations, and ii) the data size transmitted was proportional to the computational intensity of the process. For the first case, the results were as expected: as the computation increased, the energy consumed from local execution eventually began to overcome the constant network costs of remote execution and remote execution steadily became more advantageous. On the other hand, our experiments with the second scenario never showed an advantage for remote execution, illustrating the importance of selectivity. We are further exploring the space between these two scenarios to characterize the relationship between the rate of growth in data size to computation as it affects the potential for energy savings. Several other studies have investigated remote execution for power management, focusing primarily on laptop-class computers [15, 21, 22]. In particular, experiments with applications such as compilation, Gaussian, and text formatting in [21] support our conclusions that the key to effectively using remote execution is to be very selective in choosing the processes which are likely to benefit, where the computation is significant relative to the data shipped and the processor on the server receiving the task is significantly faster than on the mobile device. These experiments provide insight on the criteria that must go into an informed migration policy.

One of the outcomes of this study was an assessment of the suitability of the D'Agents and Pilot Launcher support for the purpose of saving energy by remote execution. The support for Tcl agents in D'Agents was not designed with energy conservation in mind and several sources of overhead were identified (e.g. impact of interpretation, support for more functionality than was needed). These insights will inform the development of our mobile code environment.

**Transcoding.** One special case of exploiting remote execution is the use of a transcoding proxy for transforming multimedia web content to save bandwidth (and thus energy consumed by the local device by receiving less data) or to specialize image data for display, given the destination device characteristics (thus saving energy for rendering). Transcoding is a well-known technique that can be used to provide differentiated service, essentially serving the same multimedia object at different quality levels for different situations. Transcoding frequently involves trading off object fidelity for size. Unfortunately, one danger with aggressively transcoding is that the result delivered has degraded in quality so much as to be unusable. We have developed a transcoding technology that addresses this problem and allows transcoding to be selectively applied.

We have done preliminary work [5] to characterize the information quality tradeoffs, the computational requirements and the potential space gain of a transcoding that changes the JPEG [17] compression metric. We show that, for a transcoding that changes the JPEG compression metric, the change in JPEG Quality Factor directly corresponds to the information quality lost. We describe an algorithm to compute the Independent JPEG Group's (IJG) [12] equivalent of the JPEG Quality Factor of any JPEG image. Reconstructing the initial image quality is required in order to quantify the quality loss due to transcoding operations. To understand the overhead involved in performing a transcoding, we develop a predictor that estimates the computational overhead with a high degree of accuracy. We also develop a predictor to determine if an image will lose more in storage space for a corresponding loss in information quality. We refer to the technology we have developed as *quality-aware transcoding*. Preliminary results [6] show how to utilize this technology to allow network proxy servers to customize objects for the network bandwidth available on the wireless "last hop" to a mobile client. Experiments performed in that work identify policies that selectively transcode JPEG images to deliver good quality with significant savings in bandwidth consumption and, consequently, energy for receiving. Knowledge of the characteristics of the link to the client can eliminate as many as 40% of unnecessary transcodings that would otherwise degrade the usefulness of the data with little performance advantage. We continue to pursue ways to utilize informed, quality-aware transcoding. An earlier transcoding study within the context of Active Names [27] demonstrated significant performance improvements available from adaptively placing transcoding code in wide-area networks in response to changing CPU and network characteristics. We experimented with a common scenario where network bandwidth is scarce and computational power is abundant. In this case, it may be beneficial to perform transcoding at a server, rather than to transmit a large file across a high-power, low-bandwidth link, only to discard most of the transmitted bits through a transcoding operation at a Web proxy (as is currently done [10]). We plan to further pursue such strategies while balancing the potentially conflicting optimizations in time, power, and network bandwidth.

### 5.2 Proximity-Based Communications

In the previous section, there was an adaptive decision to be made about whether or not to use remote resources and incur the communication costs. However, many applications need data originating in the Internet and, for them, avoiding communication is not an option. The question then becomes how to make the essential communication as energy efficient for the application running on the mobile device as possible. We describe one approach which may leverage other features inherent in our disaster recovery scenario.

In wireless mobile computing, the network interface can be a large component of overall power consumption [11, 23]. However, in many cases transmitter power consumption may be larger than required. Many devices transmit a signal strong enough to reach the receiver from their maximum range independent of their actual distance from the receiver.

In free space, the transmitted signal strength decreases logarithmically with distance [19]. For a given transmitted signal strength, a distance $d_0$ produces a 20dB stronger signal at the receiver than sending the same signal from a distance $10 * d_0$. If the minimum required signal strength at the receiver is the same as that received from the $10 * d_0$ transmitter, then the closer transmitter is consuming roughly 20dB more power than necessary. For example, if the receiver threshold is 1mW, then at the receiver the closer transmitter's signal strength is 100mW and consumes 210mW of power.[1] However, the closer transmitter could achieve the 1mW signal at the receiver with only 2.1mW in power consumption.

We plan to investigate techniques for exploiting the proximity of communicating devices to reduce energy consumption. In particular, we are investigating the benefits of using GPS data to predetermine transmit power requirements. This differs from the dynamic techniques used by cell phones, and has potential benefits in scenarios where the base station does not have line power and cannot afford to be in continuous contact with the remote devices, or situations where there is not a defined base station. For example, many ad hoc networks allow communication among many nodes in the network, not just a device and base station.

The potential benefit of proximity-based transmit power depends on several factors: 1.) The sensitivity of the receiver, 2.) power loss within the transmitter, 3.) signal propagation loss (instead of free space), and 4.) the potential benefit of increasing the communication bandwidth at close range instead of decreasing the transmit power. We plan to investigate each of these factors by examining in detail two wireless LAN implementations (WaveLAN [28] and Bluetooth [2]) and implementing a detailed simulator based on the ns [14] network simulation infrastructure.

### 5.3 Memory Systems

Whenever local computation is the method of choice (or, when disconnected, the only option), the question becomes how to make the basic interactions of hardware and software as energy efficient as possible. Memory instructions are among the more power-hungry operations on embedded processors [24], making the hardware/software of memory management a good candidate for optimization.

We initially plan to investigate various memory system components (both hardware and software) for opportunities to eliminate redundant or unnecessary operations, and to reduce the overhead of required

[1] From basic circuit design we assume the power required to generate a transmitted (radiated) signal of $P$ Watts is $2.1 * P$ Watts. $dB = 10log(\frac{P_1}{P_0})$

operations through hardware support. By focusing on the memory system we can build on our expertise in memory hierarchies and virtual memory to develop techniques to improve energy efficiency.

Our work in memory systems is built on the assumption that main memory for portable systems will be composed of one or more DRAM devices with the following properties: i) power management mechanisms, ii) burst transfer capability, iii) several internal banks, and iv) the requirement of refresh operations to maintain data.

Direct Rambus DRAM (RDRAM) [18] is a concrete example that satisfies our assumptions. We note that RDRAM is not the predominant memory technology in portable systems today. However, most memories exhibit similar properties, with the possible exception of controlling the power state of individual memory chips. RDRAM's novel communication protocol allows individual devices to be in any of the following power states, in decreasing power consuming and increasing access time order: Active, Standby, Nap, and Powerdown.

**Smart physical page allocation** An important operating system function is the allocation of physical pages as backing store for virtual memory. We are investigating intelligent page allocation for energy efficiency by grouping active pages into the smallest number of DRAM devices. This technique enables more devices to be in a low-power state while providing performance close to that achieved by placing all DRAM devices in the highest power state. Further, this technique reduces the energy overhead of reading data from memory.

We can extend the above approach to eliminate unnecessary operations by including information on which bank within the DRAM chip pages are allocated. By packing live pages into a small number of DRAM banks, we can reduce the number of banks requiring refresh. The data on pages that will not be accessed do not need to be retained. This can reduce power consumption for portable devices that have a significant amount of idle time (e.g., PDAs), since even when they are "turned off", the DRAM must be refreshed to maintain state.

**Lazy bzero** A common operation performed by many operating systems and applications is initializing data with zeros. For sufficiently large regions of memory the energy overhead to perform this initialization occurs in instruction execution and the transfer of data between the data cache and main memory. We are investigating the energy efficiency of hardware support to zero initialize data. One approach is to modify page table entries to indicate when a page is to be zero initialized before user-level access. The hardware support consists of examining TLB entries on a data cache miss, zeroing the cache block if needed, and clearing the bit for the cache block so it is not zeroed again. This approach can potentially eliminate redundant operations when both the operating system and user-level code zero data. It also reduces the overhead of zeroing data by reducing instruction execution, data cache writebacks and fetches for sufficiently large data regions, while maintaining the potential advantages of having the initalized data in cache since hardware initialization takes place in the cache without main memory access.

**Burst Transfers** The final aspect of power consumption in memory system design we plan to investigate is the use of burst transfers to reduce the overhead of data transfer between cache and main memory. Burst transfers are a common approach to reducing the latency of data transfers. We plan to investigate the potential benefits in terms of energy efficiency. We will investigate both hardware and software techniques to identify when multiple cache blocks can be transferred to/from main memory. The energy savings occur because we can amortize the control overhead across several data items. This technique is similar to using file prefetching to amortize disk spinup overhead over more data. In general, time-shifting to increase burstiness of access patterns is valuable when the overhead of transitions between power modes are a factor.

### 5.4 Operating System Internals

Recognizing energy as a fundamental system resource requires incorporating policies within the OS to explicitly allocate its use. Whenever a device is operating on battery power, the goal may be to expend a fixed energy budget over the longest possible session. As an example, consider the impact of this requirement on OS scheduling policy. This seems to share aspects of real-time scheduling in that there is a certain amount of essential work to accomplish before a deadline (i.e., a drained battery) and some low priority tasks may be forced to starve. We plan to investigate this analogy with ideas borrowed from real-time scheduling to manage a fixed energy budget.

Finally, we need to reconsider the internal structure and activity of the OS itself from this new perspective as it is likely to be wasteful of energy. For example, daemon processes often perform periodic maintenance functions (e.g. compaction, garbage collection); however, any nonessential activity is an energy drain and should be deferred until the device is once again plugged in and fully charged unless the energy costs of operating in a degraded state (e.g. highly fragmented disk) outweigh the overhead. Various such tradeoffs will be examined for common maintenance activities.

## 6 Summary and Conclusions

Our goal is to develop prototype systems and applications for next generation mobile and wireless environments. These future environments will support applications with demanding requirements such as disaster recovery. One of the central issues is energy conservation, especially for mobile and embedded devices, which promises to have significant economic, environmental, and societal impacts.

We focus on three key directions: i) the development of power measurement tools, workloads, and experimental methods to evaluate energy consumption, ii) the energy-aware APIs to allow application-directed power management, and iii) the development of system support for high-level solutions.

These research projects all rely on experimental techniques for evaluating ideas. By making empirical measurements and observations on device and workload characteristics, we pinpoint the problem areas of greatest potential. By initially formulating simulation models, we narrow the solution space and allow consideration of new architectures. By ultimately constructing working prototypes, we can observe all activity associated with real operating environments and gain deeper insights into their behavior. The popularity and accessibility of the palmtop and handheld platforms we are using gives this research significant potential for immediate technology transfer.

## 7 References

[1] Air Force Research Lab. C3i non real-time parallel benchmarks. http://www.rl.af.mil/programs/hpcbench/summ_nrt.htm, 1998.

[2] Bluetooth. Technology Overview. `http://www.bluetooth.com`, 1999.

[3] J. Bredin, D. Kotz, and D. Rus. Market-based Resource Control for Mobile Agents. In *Proceedings of the Second International Conference on Autonomous Agents*, pages 197–204, May 1998.

[4] J. Bredin, D. Kotz, and D. Rus. Utility Driven Mobile-Agent Scheduling. Technical Report PCS-TR98-331, Dartmouth College, Computer Science, Hanover, NH, October 1998.

[5] S. Chandra and C. S. Ellis. JPEG Compression Metric as a Quality Aware Image Transcoding. In *2nd Symposium on Internet Technologies and Systems*, Boulder, CO, October 1999. USENIX.

[6] S. Chandra, C. S. Ellis, and A. Vahdat. Multimedia Web Services for Mobile Clients Using Quality Aware Transcoding. In *Proceedings of the Second ACM/IEEE International Conference on Wireless and Mobile Multimedia (WoWMoM'99)*, Seattle, WA, August 1999. ACM SIGMOBILE.

[7] C. Ellis. Powerpoint presentation from palm programming topics course. `http://www.cs.duke.edu/~carla/palm/index.htm`, 1999.

[8] C. S. Ellis. The Case for Higher-Level Power Management. In *Proceedings of the 7th Workshop on Hot Topics in Operating Systems*, Rio Rico, AZ, March 1999.

[9] J. Flinn and M. Satyanarayanan. PowerScope: A tool for profiling the energy usage of mobile applications. In *Workshop on Mobile Computing Systems and Applications (WMCSA)*, pages 2–10, February 1999.

[10] A. Fox, S. Gribble, Y. Chawathe, and E. Brewer. Cluster-Based Scalable Network Services. In *Proceedings of the 16th ACM Symposium on Operating Systems Principles*, Saint-Malo, France, October 1997.

[11] R. Kravets and P. Krishnan. Power Management Techniques for Mobile Communication. In *Proc. of the 4th International Conf. on Mobile Computing and Networking (MOBICOM98)*, pages 157–168, October 1998.

[12] T. Lane, P. Gladstone, L. Ortiz, J. Boucher, L. Crocker, J. Minguillon, G. Phillips, D. Rossi, and G. Weijers. The Independent JPEG Group's JPEG Software Release 6b. ftp.uu.net/graphics/jpeg/jpegsrc.v6b.tar.gz.

[13] D. Lee, P. Crowley, J.-L. Baer, T. Anderson, and B. Bershad. Execution Characteristics of Desktop Applications on Windows (NT). In *International Symposium on Computer Architecture (ICSA)*, June 1998.

[14] S. McCanne, S. Floyd, and K. Fall. ns - LBNL Network Simulator. See `http://www-nrg.ee.lbl.gov/ns/`, 1996.

[15] M. Othman and S. Hailes. Power conservation strategy for mobile computers using load sharing. *Mobile Computing and Communications Review, SIGMOBILE*, 2(1):44–50, January 1998.

[16] J. K. Ousterhout. An Introduction To Tcl and Tk. Forthcoming book from Addison Wesley Publishing, 1993.

[17] W. B. Pennebaker and J. L. Mitchell. *JPEG - Still Image Data Compression Standard*. Van Nostrand ReinHold, New York, 1993.

[18] Rambus. *RDRAM*, 1999. `http://www.rambus.com/`.

[19] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, 1996.

[20] M. Rosenblum, S. A. Herrod, E. Witchel, and A. Gupta. Complete computer simulation: The simos approach. *IEEE Parallel and Distributed Technology*, 1995.

[21] A. Rudendo, P. Reiher, G. Popek, and G. Kuenning. Saving portable computer battery power through remote process execution. *Mobile Computing and Communications Review, SIGMOBILE*, 2(1):19–26, January 1998.

[22] A. Rudendo, P. Reiher, G. Popek, and G. Kuenning. The remote processing framework for portable computer power saving. In *Proceedings of ACM Symposium on Applied Computing*, Feb 1999.

[23] M. Stemm and R. Katz. Measuring and Reducing Energy Consumption of Network Interfaces in Hand-Held Devices. In *Proceedings of 3rd International Workshop on Mobile Multimedia Communications (MoMuC-3)*, September 1996.

[24] V. Tiwari, S. Malik, and A. Wolfe. Power analysis of embedded software: A first step towards software power minimization. *IEEE Transactions on Very Large Scale Integration*, 2(4):437–445, December 1994.

[25] University of Washington Memory Systems Group. The Scribe Project: Windows NT application traces. `http://memsys.cs.washington.edu/memsys/html/traces.html`, 1998.

[26] A. Vahdat, T. Anderson, M. Dahlin, E. Belani, D. Culler, P. Eastham, and C. Yoshikawa. WebOS: Operating System Services for Wide-Area Applications. In *Proceedings of the Seventh IEEE Symposium on High Performance Distributed Systems*, Chicago, Illinois, July 1998.

[27] A. Vahdat, M. Dahlin, T. Anderson, and A. Aggarwal. Active Names: Flexible Location and Transport of Wide-Area Resources. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems (USITS)*, October 1999.

[28] WaveLAN. Technical Specification. `http://www.wavelan.com`, 1999.