A NETWORK-ORIENTED POWER MANAGEMENT ARCHITECTURE

LUIS F. POLLO and INGRID JANSCH-PÔRTO
Universidade Federal do Rio Grande do Sul – Instituto de Informática
P.O. Box 15064, 91501-970 – Porto Alegre, RS, Brazil
{pollo, ingrid}@inf.ufrgs.br

Abstract: This paper presents the proposal and implementation of a power management architecture for local area network environments. Our main goal is to contribute to more efficient use of electricity, reducing energy waste by facilitating the configuration of power policies and providing a means to automate their execution.

The architecture we propose is based on the SNMP (Simple Network Management Protocol) framework. The management process is coordinated by a central element, which applies configurable power policies to computers and other electronic devices connected to the network, either in accordance to consumption preferences defined by the network administrator, or in response to changes in the supply of electricity, detected through monitoring of UPS (Uninterruptible Power Supply) devices. In the latter case, applying the pre-configured power policies allows the UPS to sustain power to essential parts of the network (e.g. servers) for a longer period of time. Alternatively, the definition of power policies can be based exclusively on administrative preferences, in which case the goal is to minimize consumption of electricity during non-business hours by shutting down inactive equipment or putting it in low-power modes.

We have measured UPS autonomy during utility power outages and found that it can increase by a factor of 7 in a small network setup, by using the management system to automatically power off 90% of the computers. Potential economy resulting from the decrease in consumption during non-business hours alone is estimated to be as high as US$ 36 per computer during the period of a year.

Key words: power management, energy efficiency, network management, SNMP, ACPI, APM.

1. INTRODUCTION

Electric power consumption is increasingly becoming an essential aspect of computing. The wide-spread use of mobile computing devices, such as personal digital assistants (PDAs) and notebooks, has boosted research in the area of power management, where several contributions have been made, most of which focused on reducing consumption in order to allow longer battery-powered operation. However, the impact of energy consumption extends far beyond this category of devices. In the United States, recent studies have shown that the electricity used by office and network equipment corresponds to a considerable percentage of the total amount consumed, with a high estimate of energy waste due to poor use of existing power management mechanisms [1]. Those findings only reinforce the importance of power management for all sorts of computing equipment, from the smallest pocket device to the largest corporate server.

Another recent trend in the computer industry that is likely to have a very significant impact on energy consumption is that of the "appliance-PC" – a personal computer that is always ready to use at the push of a button, the same way a TV or another home appliance would be. Industry giants such as Intel and Microsoft have been working on hardware manufacturing guidelines for the so-called "instantly-available computer", as well as on software architectures that allow the PC to be put into different low-power modes according to the level of system activity, instead of turning it off completely. Their early work resulted in the Advanced Power Management (APM) specification [2], in 1992, which has evolved into a much broader, more complex specification called ACPI (Advanced Configuration and Power Interface), presented in conjunction with other industry leaders in 1996 [3]. Both the APM and ACPI specifications define a set of interfaces between power-manageable hardware and power-aware software. While the "appliance-PC" does not become a reality, these power management mechanisms are being used to help lower energy consumption on existing PCs.

But the power consumption of each computer viewed as a single, stand-alone entity is only the beginning of a much larger problem. With the ever-growing need for interconnection of devices over networks, more and more computers are being left on after regular office-hours, in an attempt to prevent disruptions in network services and inaccessibility to shared resources, or to allow administrative tasks (e.g. backups) to be performed during periods of inactivity. Network operating systems have clearly not been able to keep up with the advances in power management technology, and commonly still require that a host maintains its network connections in order to respond to periodic server queries. That leads to two immediate problems: first, depending on the network hardware installed on a PC, these periodic queries can cause enough activity to keep the computer awake, defeating power management; second, if the PC succeeds in entering a low-power mode, it might be unable to respond to the server, which will then assume the PC is off and terminate network services to it [4]. Certain newer network adaptors are equipped with a technology called "wake on LAN", which allows the computer to be turned on remotely upon reception of a special message (a "magic packet") [5], thus eliminating part of that problem. But the fact still remains that a large portion of networked equipment is unnecessarily left on during non-business hours, whether to comply with corporate guidelines or as result of simple misuse. A study conducted by Lawrence Berkeley National Laboratory (University of California), at two major metropolitan areas of the U.S. showed that only 44% of computers, 32% of monitors and 25% of printers in office environments are turned off at night [6].

Our goal is to investigate and explore exactly the network-related aspects of power management. We believe that energy, as much as any other shared network resource, should be managed from a global perspective of the entire LAN, and not only from the isolated view of each device. In this paper, we propose centralizing the configuration of power policies at a management station in order to minimize energy waste due to badly configured devices. The possibility of configuring power policies for several devices from a single point can be a valuable asset to network administrators, who are in the best position to decide which parts of the network are required to stay on during non-business hours, tailoring energy consumption down to the absolutely necessary. Furthermore, we believe that this capability to oversee and configure power consumption by networked devices should seamlessly integrate into existing network management platforms. Our approach is to utilize the Simple Network Management Protocol (SNMP [7]), the *de facto* standard management framework, and the existing power management capabilities of computer hardware, as the basis for a simple, yet efficient, network-oriented power management architecture.

This paper is organized as follows. Section 2 discusses pertinent related work. Section 3 presents our proposed architecture. Section 4 gives details of the implemented prototype environment. Section 5 presents a few experimental results. Section 6 concludes the paper.

2. RELATED WORK

It is important to point out that the existing power management features of computer hardware already make a significant contribution to energy savings. Since the creation of the Energy Star program by the United States Environmental Protection Agency (EPA), in 1992, a long way has been covered, and the vast majority of PCs manufactured today ships with some sort of power management capability. According to estimates made by the Energy Analysis Department of the Environmental Energy Technologies Division, at LBNL, based on data for 1999, power management saves about 23 TWh per year in the United States [1], which would correspond to US$ 1.95 billion at an average price of 8.5 cents per KWh. To give a clearer picture of just how much electricity that is, it would be enough to serve over 2 million average American households for an entire year [8].

The availability of power management features in computer hardware has allowed researchers to concentrate on the conception of mechanisms through which the low-level physical components can be driven to achieve the greatest possible savings [9, 10, 11, 12]. Others have explored the collaboration between different levels of software (e.g. operating system, device drivers and applications) to allow more efficient power management [13, 14, 15, 16]. But, even though those areas have been extensively investigated, there appears to be little work concentrating on ways to overcome the obstacles that network environments pose to power management.

There is, however, a research area where the paths of network and power management do cross. For many years, the UPS (Uninterruptible Power Supply) industry has been providing their customers with data protection systems that can trigger the unattended shutdown of computers in the event of utility power failures. These systems all share a common event-driven structure and have an inherent notion of power policies, even though those policies are based on a simplistic on-or-off approach. Their configuration typically associates the occurrence of energy-related events, detected through UPS monitoring, to the execution of a set of predefined actions, such as remotely shutting down specified computers or notifying a network administrator via e-mail. Communication between the monitoring station and the client computers is typically done via a proprietary protocol.

3. OVERVIEW OF THE POWER MANAGEMENT ARCHITECTURE

What we have done is to identify the potential of this industry-standard data protection architecture for adaptations that would make it suitable to perform power management tasks in a network environment. If we could extend the event association capabilities of the original architecture to allow the execution of predefined actions at specified times (e.g. "after 6 P.M., everyday"), we would be able to use the same communication infrastructure to perform coarse-grained (on-or-off based) remote power management of the computers in the network, reducing consumption during periods of inactivity. Additionally, if we could map this communication infrastructure to a standard network management protocol (such as SNMP), we would guarantee compatibility with existing management platforms and tools. Finally, if we could extend the set of possible actions to include intermediate low-power states for the remote hosts, we would have fairly flexible, fine-grained control over the overall use of energy in the network, which could have a positive effect not only on reducing energy waste but also on extending backup-power autonomy during utility power outages.

Those are the basic characteristics of the power management architecture we propose. We have been able to validate the feasibility of the model by implementing it in a prototype environment, which will be discussed in detail in Section 4. For now, it is enough to define the elements of the architecture, which can be seen as a specialized derivation of the traditional SNMP model, with *agents* residing on every manageable electronic device of the network and a *manager* that oversees their operation and determines when to apply changes in their power consumption, according to the pre-defined policies mentioned before. The agents hold the specific knowledge about the energy consumption characteristics of the devices they control, and communicate with the manager through SNMP, exchanging information as defined in a *management information base* (MIB) designed specifically for power management. Figure 1 depicts the main elements of the architecture.

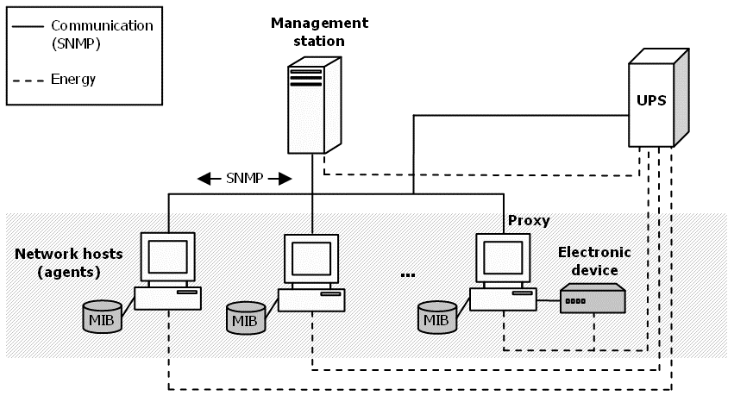

Figure 1. SNMP-based power management

In theory, any electronic device connected to the network can be managed, as long as it fulfills two basic requirements: first, it must be accessible via the TCP/IP network (either directly or through a proxy); second, it must provide some sort of power management capability that can be controlled by the agent (even if it is as simple as turning the device on or off, for example).

3.1 Power policies and the event-driven operation model

The basis for the operation of our proposed power management system is an event-driven model that is inspired in the reactive operation model of data protection systems. As we mentioned before, those systems act upon the occurrence of special conditions in the supply of energy to guarantee data integrity across the network. We call those conditions *energy events*. Energy events have intrinsic associated semantics. For example: "the external power has been disrupted" or "the level of the UPS batteries is low". We extend that model by defining *programmed events*, which can trigger the execution of actions at specified times. A programmed event does not have an inherent "type" like an energy event, but instead allows us to use an "alarm-clock" abstraction to initiate the execution of actions based on time criteria (e.g. "turn off the displays of all computers at 6:30 PM, everyday"). By adapting this abstraction to the original model, we are able to use a single configuration structure for both purposes. In other words, we can specify power policies that are applied in a reactive fashion either in response to energy events (to improve data protection) or according to predefined time criteria (to reduce energy waste).

Having a uniform association paradigm for both purposes also facilitates the dispatch of actions to the remote hosts, which can be implemented in a single component that handles the conversion of configured actions to their corresponding SNMP requests and transmits them to the appropriate destinations.

In our approach, power policies are specified as lists of actions that can be associated to either energy or programmed events. Five different types of actions are supported, as indicated in Table 1. Each action in a power policy is destined to a specific *target* – a device or group of devices where the action must be executed[1]. Except for the `WAKEUP` action, all others are mapped to an SNMP `Set` request intended to alter a specific object in the remote agent's MIB. The remote agent is responsible for interpreting the received value and executing the appropriate action, which is why we say the operation model is based on the association of events to *remote actions*.

Table 1. Supported action types and parameters

Action type	Parameters	Meaning
`SHUTDOWN`	-	Turn the device off
`WAKEUP`	-	Turn the device on
`SET_POWER_STATE`	`component, state`	Change the power state of the specified `component` to the specified `state`
`RUN_COMMAND`	`command`	Execute the specified `command`
`SHOW_MESSAGE`	`message`	Show the specified `message` to the user(s)

`WAKEUP` actions are used to remotely power on a device or bring it to an active state if it had been "sleeping". In those cases, the remote agent will not be running, therefore the correspondence with an SNMP message does not exist. Instead, a *Wake-on-LAN magic packet* is used to wake up the corresponding host. Obviously, only hosts that have a compatible network adapter will be able to detect the packet. Wake-on-LAN enabled adapters remain powered when the computer is turned off or put in a low-power state, and continually scan incoming network packets for a special data pattern. When that pattern is detected, the adapter triggers a boot sequence in the BIOS.

`SET_POWER_STATE` actions also deserve a little more attention. This category of actions was designed with flexibility in mind. Instead of defining a large fixed (and thus limited) set of state-changing actions for all possible components of a computer (e.g. one to turn off the monitor, another to spin down the hard disk, and so on), we chose to define a single, configurable type of action that can be used to request changes to the state of any component in a flexible manner. For each `SET_POWER_STATE` action, a hardware component name and the desired state must be specified. This approach allows the system to be universally compatible with different types of power management standards, such as APM or ACPI, for which specific agents could be implemented. There is no constraint on the names of components or states used in the configuration, except that every agent must support at least one component named "`global`", which represents the entire device. That means that the successful execution of an action depends on the correct

[1] We use the term "device" to refer to each *networked device* (e.g. a computer or another manageable electronic device), and "component" to refer to *hardware components* that may be individually controlled *within* a device.

interpretation of the messages by the agent, and on the semantic equivalence between the actions configured in the NMS and those supported by the agent.

3.2 Manager-agent communication

In typical SNMP applications, the network management station (NMS) is responsible for initiating communication with the remote agents to update its view of the network. In order to monitor the status of the network, the NMS is usually configured to perform periodic queries of every host's agent to determine its current state, including possible error conditions. This procedure is commonly referred to as *polling*.

However, this strategy would not be appropriate in our power management architecture, since the agents might frequently be unresponsive due to their hosts entering low-power states. On the other hand, as we have mentioned before, intensive incoming traffic could prevent the network hosts from entering such low-power states, which is another reason why polling is not an adequate approach. Instead, we use the alternative method: agent-initiated notification. In other words, whenever a relevant change in a computer's power state occurs, the agent must send an unsolicited notification to the manager. This avoids interference with the local execution of power management on each computer, and also reduces SNMP traffic considerably. The agents of those devices that are in an active state are also required to send periodic *"I'm alive"* notifications so that the manager can update its network map.

This passive behavior of the manager is only used for monitoring purposes, of course. Actual power management of the remote hosts requires that the NMS sends an SNMP `Set` request to the corresponding agents, which triggers the remote execution of the desired state change.

A thorough description of the Power Management MIB, which must be implemented by the agents, is included in [19].

3.3 Security

Unauthorized access to the power management agent on a host must be prevented, since it allows the requesting entity to perform operations that could result in unavailability of services provided by that host or to execute other potentially harmful actions. Luckily, version 3 of the Simple Network Management Protocol [17] provides both authentication and confidentiality, thus guaranteeing that only authorized requestors can access objects on an agent. Therefore, as long as an agent accepts only valid SNMPv3 requests, its presence in the system is no more of a threat than any other secure remote operation service (e.g. an SSH server).

After authorization, further processing of a request is entirely up to each agent's implementation. For example, an agent could make sure that only harmless commands are allowed to execute in response to a `RUN_COMMAND` action, preventing the management station to delete files, stop network services, and so forth.

4. PROTOTYPE ENVIRONMENT

Given the prohibitive cost of commercial network management software and the additional complexity of integrating a customized solution into such a platform, our approach was to develop a stand-alone management application, entirely in Java, using publicly available libraries. Additionally, a prototype agent has been developed for the Microsoft Windows 2000 platform, which provides one of the richest power management APIs amongst all PC operating systems. This section details the implementation of the prototype manager and agent.

4.1 The manager

The manager we have implemented performs the following basic tasks:

– monitoring of energy events reported by UPS devices;
– generation of programmed events at specified times;
– dispatch of remote actions in response to energy or programmed events according to the configuration;
– monitoring of power management agents for network map update.

Each of these tasks is delegated to a specialized component of the manager, as depicted in Figure 2. We will discuss the manager's components next.

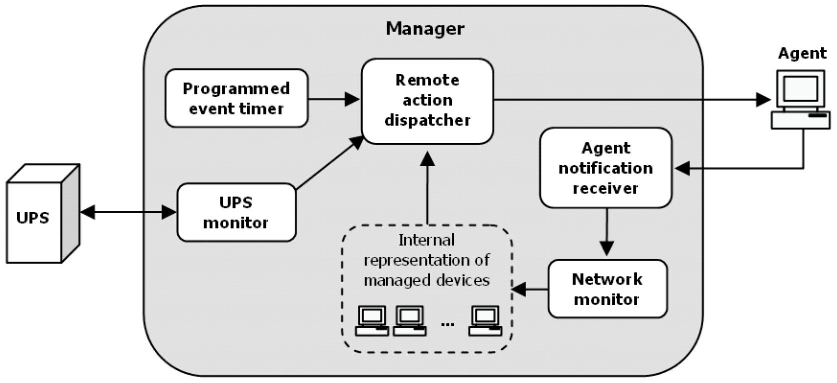

Figure 2. Composition of the manager

4.1.1 UPS monitoring

UPS devices can be monitored in several different ways, but the most typical are either via a serial line or via SNMP. Many vendors support the standard UPS-MIB [21] for SNMP management; others support their own specialized MIBs. We have implemented SNMPv1-based UPS monitors for the standard MIB and for two other vendor-specific MIBs that are used for supervision of UPS models available in our laboratories. In order to organize shared access to a common UDP port used for trap reception, we have implemented a single trap handling component that analyses incoming traps, identifies the source UPS agent, and forwards them to the appropriate monitor. This is easily accomplished by having all monitor classes implement a common interface – each monitor class “understands” a certain type of MIB, and is responsible for creating one or more monitor instances to handle several UPSs that can be managed through that MIB, at its own discretion. Using this approach, we can dynamically install and uninstall UPS monitors as small software plug-ins, without having to recompile the manager.

4.1.2 Programmed events

The other component capable of triggering the dispatch of actions is the programmed event timer. This timer is initialized from information in the manager’s configuration file that describes the dates and times when specified actions should be automatically executed. Besides a scheduled execution time, a repetition rate can be specified for each programmed event. An event can be scheduled to repeat daily, weekly or monthly at the same time, which facilitates the configuration of typical power saving policies such as “automatically turn off all computer monitors after work, everyday”.

4.1.3 Dispatch of remote actions

The remote action dispatcher is the central point for event processing in the manager. As we have briefly described before, it operates in response to either energy events reported by UPS monitors or programmed events generated by a timer. The basic task of the dispatcher, upon detection of an event, is to identify the actions associated with that event, convert them to the appropriate SNMP commands, and send them out to their respective targets. Since actions might be configured for delayed execution, the dispatcher also handles possible conflicts between pending actions and those scheduled for immediate execution.

4.1.4 Network monitoring

As we have mentioned before, the process of updating the network map in the manager is based on the reception of agent notification. The network monitor is the component responsible for processing incoming notifications in order to maintain an updated view of the network. Depending on the type of notification received and the current state of the originating device, the network monitor executes a full SNMP query of the agent’s MIB to fill in relevant information about the device. The work performed by this monitor is important because the internal representation of the devices instruments the decision making process in the action dispatcher. Maintaining a faithful view of all power-managed devices also facilitates graphical representation of the network for administrative purposes.

4.1.5 Configuration

Configuration of the manager is done via the Extensible Markup Language (XML), which has become a widely used format for this purpose because of its hierarchical structure and broad support in all major programming languages. Besides providing a relatively large set of application options (e.g. UDP ports, timeout limits, logging options, etc.), the main goal of the configuration file is to describe the elements of the network and the associations between events and action lists (i.e. power policies). A group metaphor is provided to facilitate the configuration process – devices can be arranged according to arbitrary logical criteria, such as hardware similarity, common power management features, etc. Figure 3 shows sample sections of the configuration file illustrating the general process for defining a time-based power policy.

```
...
<device name="pc1" address="10.0.0.5"/>
<device name="pc2" address="10.0.0.6"/>
<device name="pc3" address="10.0.0.7"/>
...
<group name="pcsGroup">
    <group-member ref="pc1"/>
    <group-member ref="pc2"/>
    <group-member ref="pc3"/>
</group>
...
<programmed-event name="Automatic monitor night turn-off"
                  startTime="07/01/2002 19:00:00"
                  repeat="DAILY">
    <action target="pcsGroup" type="SET_POWER_STATE">
        <param name="component" value="display"/>
        <param name="state" value="off"/>
    </action>
</programmed-event>
...
```

Figure 3. Sample manager configuration

4.1.6 Graphical operation

Besides operating in the unattended mode described so far, the manager can also be controlled through a graphical tool, via RMI (Java's Remote Method Invocation API). This allows the administrator to visualize the network map and execute remote actions on demand.

4.2 The agent

In order to test and evaluate the power management architecture, we have also implemented a prototype agent for the Microsoft Windows 2000 operating system. This agent can perform the following basic functions:

– detection of power management-related system messages;
– notification of the manager upon system power-on, transition to low-power states, and resumption from low-power states (wake-up);
– processing of SNMP `Set` requests that allow the manager to command the host's transition to three different states: power-off, standby, and hibernation.

Starting with Windows 2000, Microsoft's operating systems have vast support for power management operations. Applications are notified of changes in the power status of the computer through a specific system message, `WM_POWERBROADCAST`, which can indicate several different events (e.g. "system entering low-power mode", "system resuming from low-power mode", etc.). Our prototype agent simply interprets these messages as they are received from the OS, sending out the appropriate SNMP notifications when necessary.

The agent understands three different values for the global state of the computer: "off"; "standby" (also referred to in the literature as "suspend to RAM", a state in which memory contents might be lost due to abrupt loss of power); and "hibernate" (or "suspend to disk", when memory contents are safely transferred to non-volatile storage and the PC is completely powered down).

We have also tested the ability to remotely power on the machine by sending it a "magic packet" from the management station, and ensured proper agent behavior upon successful remote wake-up.

5. EXPERIMENTAL RESULTS

A series of experiments allowed us to validate the feasibility of the proposed architecture and to estimate the potential economy that may result from effectively deploying the system. The most important results are summarized in the following sections.

5.1 Estimates of energy savings during non-business hours

Perhaps the most appealing reason for a power management system that can be integrated into and take advantage of existing network management platforms is the enormous potential for power savings during non-business hours. As we have mentioned previously, an LBNL study released in 2001 estimates nightly turn-off rates for office equipment to be considerably low (44% for PCs and 32% for monitors) [6]. That same study estimates that only 3 to 8% of computers and 38% of monitors are put into low-power states after office-hours, leaving 30% of monitors and over 50% of computers that are potential candidates for power management, not to mention other equipment such as printers, copiers, fax machines and so forth, all of which could potentially be power-managed remotely.

Considering a typical 9-to-5 workday[2], an average of 250 workdays per year, and an average price for electricity of 8.5 cents per KWh, we can make the following (quite obvious, yet frequently overlooked) additional observations:

– each desktop PC station (CPU and monitor) consumes about 1.2 KWh during a typical workday, if it is permanently active[3]; that amount of energy equals a cost of US$ 25.50 per year;

[2] We consider the actual length of a "9-to-5 workday" to be 9.5 hours for more realistic estimates.

[3] We consider the following average consumption values in active, low-power, and off modes for a common desktop and a 15 inch monitor, respectively: 50/75 Watts; 25/5 Watts; and 1.5/0.5 Watts.

- if each PC is kept completely active during non-business hours, it consumes an additional 1.8 KWh of electricity per day – an unnecessary expense of US$ 38 per year;
- if the monitor alone is put into a low-power state after office-hours, US$ 21.50 can be saved each year, per PC;
- if the entire PC is put into a low-power state at the end of a workday, a total of US$ 29 can be saved per PC each year;
- turning off the PC completely allows even further savings: U$ 36.50 a year.

5.2 Increase in UPS autonomy during power outages

We have conducted measurements in a typical network setup in order to determine the potential increase in UPS autonomy during utility power outages. Our experimental environment consisted of a small Ethernet LAN of 10 PCs (one of which acted as a server, running the manager), all drawing power from a 2 kVA UPS equipped with 12 batteries that can supply 9Ah each.

These measurements correspond to backup power autonomy during a blackout, in three different scenarios. In the first scenario, the manager was not configured to react to the start of battery-powered operation, and all 10 PCs remained active until the batteries were depleted. In the second scenario, the manager was instructed to put the 9 client PCs in standby mode. In the third and last scenario, the manager was configured to power down the same 9 PCs. The average measured consumption per PC is 125 Watts in active mode, 30 Watts in standby mode, and 6.5 Watts in soft-off mode.

The results of those three rounds of measurements are presented in Figure 4. The graphic shows that autonomy increases by a factor of 3.5 between scenarios *a* and *b*, and by a factor of 7 between *a* and *c*. It should be noted that we have omitted possible user intervention in the process of altering the power state of client PCs in this experiment. In other words, the picture represents an ideal case, in which no users were present at the time of the power failure, or in which all users peacefully accepted the power management agent's request to power off the PC or put it in standby mode. It is likely that, in a real-life situation, most stations would be in use and would have to remain active for some time so that users could finish pending tasks. Nevertheless, it is clear that the potential for reduced consumption during a power failure exists, and could be better explored through the use of an automated solution, such as the one we have presented. Assuming a power failure during regular business-hours, the increase in autonomy would most probably appear in intermediate ranges of the graphic, but would still help amplify the chances of utility power returning before UPS batteries were depleted, avoiding a complete network shutdown. We have yet to conduct measurements in such circumstances to obtain precise numbers.

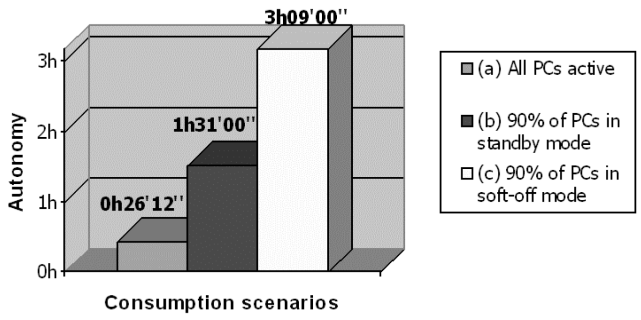

Figure 4. UPS autonomy in different consumption scenarios

6. CONCLUSION

This paper described the design and implementation of a power management architecture for local area networks. The architecture is based on the SNMP management framework, relying on *agents* that reside on each power-manageable device of the network, and whose operation is supervised by a central element, the *manager*. Manager and agents communicate via SNMPv3, a standard protocol which provides the required security characteristics, and exchange information according to a *management information base* specifically designed for power management.

Our main goal is to facilitate power management in network environments, where many factors might interfere with successful execution of power management on each device, particularly poor configuration and operation patterns of network systems and services. We believe that the network administrator is in a privileged position to determine which devices are best suited for power management, given his knowledge of the services that run on the network and their requirements, and would be in an even better position to define power consumption policies for the various devices if it could be done from a central point, which provides a global view of the entire LAN.

We do not intend to replace local power management (i.e. power management that is executed on each device independently of the influence of the manager). In fact, we encourage power management features to be enabled and functional on all electronic devices of the network to obtain the greatest possible energy savings. Our goal is simply to aid in the configuration of power policies in the network environment, possibly acting on those devices that have not been correctly set up for local power management. The architecture is designed so as to prevent interference with local execution of power management.

There are several possible areas for further exploration within the context of this research. The most prominent ones are probably the development of full-featured agents for a wider variety of operating systems and integration of the stand-alone management application with an existing network management platform, such as Hewlett-Packard's OpenView, for example.

REFERENCES

[1] K. Kawamoto et al. Electricity Used by Office Equipment and Network Equipment in the U.S.: Detailed Report and Appendices. LBNL-45917. Lawrence Berkeley National Laboratory, University of California. February 2001.

[2] Intel Corp. and Microsoft Corp. Advanced Power Management (APM) BIOS Interface Specification. Rev. 1.2. 1996. http://www.microsoft.com/hwdev/archive/BUSBIOS/amp_12.asp

[3] Compaq Computer Corp., Intel Corp., Microsoft Corp., Phoenix Technologies Ltd., Toshiba Corp. Advanced Configuration and Power Interface specification. 1996. http://www.acpi.info/index.html

[4] B. Nordman et al. User guide to power management for PCs and monitors. LBNL-39466/UC-1600. Lawrence Berkeley National Laboratory, University of California. January 1997.

[5] ADVANCED MICRO DEVICES, INC. Magic Packet technology white paper. 1995. http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/20213.pdf

[6] C. A. Webber et al. Field surveys of office equipment operating patterns. LBNL-46930. Lawrence Berkeley National Laboratory, University of California. September 2001.

[7] J. Case et al. A Simple Network Management Protocol (SNMP). RFC 1157.

[8] Energy Information Agency, United States Department of Energy. A Look at Residential Energy Consumption in 1997. DOE/EIA-0632 (97), p.17. November 1999.

[9] F. Douglis, P. Krishnan and B. Marsh. Thwarting the Power Hungry Disk. In Proceedings of the 1994 Winter USENIX Conference, pp.293-306, January 1994.

[10] R. Kravets and P. Krishnan. Power Management Techniques for Mobile Communication. In Proceedings of the 4th International Conference on Mobile Computing and Networking (MOBICOM98), pp.157-168, October 1998.

[11] J. Lorch and A. J. Smith. Reducing processor power consumption by improving processor time management in a single-user operating system. In Proceedings of the 2nd ACM International Conference on Mobile Computing (MOBICOM96), pp.143-154, November 1996.

[12] M. Stemm and R. Katz. Measuring and Reducing energy consumption of network interfaces in hand-held devices. In Proceedings of the 3rd International Workshop on Mobile Multimedia Communications (MoMuC-3), September 1996.

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

[14] J. Lorch and A. J. Smith. Software Strategies for Portable Computer Energy Management. IEEE Personal Communications Magazine, v.5, n.3, pp.60–73, June 1998.

[15] Y. Lu, T. Simunic and G. De Micheli. Software controlled power management. In Proceedings of the 7th International Workshop on Hardware/Software Codesign, pp.157-161, Rome, Italy, May 1999.

[16] A. Vahdat, A. R. Lebeck, C. S. Ellis. Every Joule is precious: the case for revisiting operating system design for energy efficiency, In Proceedings of the 9th ACM SIGOPS European Workshop, September 2000.

[17] J. Case et al. Introduction to version 3 of the Internet-standard Network Management Framework. Request for Comments 2570. 1999.

[18] J. Case et al. UPS Management Information Base. RFC 1628. 1994.

[19] L. F. Pollo. Power management system for local area networks (in Portuguese). Master's thesis. PPGC – UFRGS. Porto Alegre, Brazil, 2002. http://www.inf.ufrgs.br/~pollo/netpower/