A Sensor-based Power Management System for Sustainable Computing in Large Environments

Simone Brienza
Dept. of Information Engineering
University of Pisa
Pisa, Italy
E-mail: simone.brienza@for.unipi.it

Giuseppe Anastasi
Dept. of Information Engineering
University of Pisa
Pisa, Italy
E-mail: giuseppe.anastasi@iet.unipi.it

***Abstract*— A considerable fraction of the total energy consumed by the Internet is due to an inefficient utilization of computing equipment in homes and offices, such as Personal Computers (PCs), printers, and IP phones. PCs are often left on even when they are not used, while printers and IP phones are typically kept always on, especially in offices. Significant energy and economic savings could, thus, be achieved through a more intelligent utilization of computing equipment, especially in large organizations. In this paper we propose an automated power-management system for large environments, that suspends PCs when they are not in use and re-activate them as soon as the user is supposed to use them again. The proposed solution leverages different sensors for presence detection that are already in the environment and, thus, do not introduce any additional cost.**

Keywords—Power Management, Large Computing Environments, Wake-on-LAN, Low intrusiveness.

I. INTRODUCTION

Recent studies [1] have shown that a large fraction of the total energy consumed by the Internet is due to computing equipment in homes and offices, such as Personal Computers (PCs), printers, displays, and IP phones. PCs are often powered on even when they are not used, while other user equipment (e.g., printers and IP phones) are typically kept always on, especially in public offices. While the power consumption of a single equipment is limited (e.g., 100 W for a desktop PC), the total consumed energy may be remarkable, especially in organizations with a large number of users. A large fraction of this energy could be saved through appropriate power management strategies. However, users generally do not pay so much attention to the energy problem and, so, they typically do not enable power management strategies on their PC (e.g., automatic hibernation after an idle period).

To investigate the PC utilization pattern in working places we monitored, for a period of 8 weeks, the status of networked PCs in our university campus. To this end, we used an approach similar to that described in [2]. We found that a very large number of PCs are *never* turned off by their users. In addition, almost all PCs remain continuously active during the working time, even if their users are involved in a meeting or class. Figure 1 shows three typical utilization patterns. The first user (PC1) never switches her/his PC off because she/he wants to be able to access it from a remote location (e.g., from home),

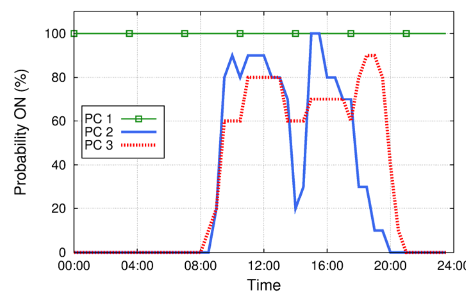

Figure 1 - PC utilization patterns.

when necessary. The other two users regularly switch off their PC when they leave their office. In addition, the user of PC2 typically switches it off also during lunch time. However, in both cases the PC remains active almost continuously during the day.

Our experimental results confirm that significant energy (and economic) savings could be achieved through appropriate power management strategies. However, it is not reasonable to rely only on the user cooperation, especially in large organizations. Hence, automated power management systems are required that allow to switch off PCs when they are not used, with minimum (or no) user involvement. A number of such solutions are already available and are briefly discussed below.

Polisave [2] is a client-server system that allows users to schedule specific actions for automated power management of networked PCs. For each controlled PC, it allows to specify the exact time when the PC should be switched off and on (e.g., when the user typically enters and exits her/his office). The controlled PC (client) periodically queries the power manager (server) to check for possible actions scheduled for it. If a shutdown or hibernation is planned at that time, the PC immediately turns off. On the other side, if a wakeup is scheduled for a PC, the server sends a Magic Packet [3] to it. Polisave allows to define static rules, i.e., the user must specify in advance the exact time when the PC must be switched on/off. Of course, this is very difficult to predict with good accuracy. Polisave can (partially) eliminate energy wastes that occurs during nights or week-ends. However, it has no effect on energy wastes due to idle periods during the working time, e.g., when the user is attending a meeting or teaching a class.

Work carried out in the framework of the SmartBuildings project funded by the Regional Government of Sicily (POR FESR Sicilia 2007-2013).

978-3-901882-56-2@IFIP,2013

To this purpose, all modern operating systems allow the user to enable automated power management (e.g., hibernating the PC after a pre-defined idle period). However users typically do not enable such mechanism, in order to avoid its negative side effects.

In this perspective *Gicomp* [4] allows to automatically install/modify power management policies on PCs in large computing environments. Policies are decided by the organization management and installed by the network administrator through the system (i.e., without any user intervention). Specifically, Gicomp allows to define the time to dim and turn off the display, the disk spin down timeout, the suspend timeout, the hibernate timeout, and other options, according to what the operating system offers. Of course, Gicomp can be effective only if power management strategies can be decided centrally and cannot be disabled by the user.

To overcome the limitations of the previous power management systems, in this paper we propose a *E-Net-Manager*, a sensor-based system that leverages information provided by sensors, in addition to user-defined rules. Basically, the PC is switched off when the user is leaving her/his working area, and resumed when she/he approaches it. To this purpose, presence detectors are used. In principle, any kind of device could be used to detect the presence/absence of the user in the working area. In practice, to be cost effective, *E-Net-Manager* relies on devices that are already available in the environment and are typically used for other purposes. Hence, they do not require any additional cost. The list of such sensors includes *attendance recorders*, *card-enabled doors*, *Bluetooth-enabled phones*, and software tools such as *Google Calendar*.

The rest of the paper is organized as follows. Section II presents the power management strategies implemented in *E-Net-Manager*. Section III discusses the system architecture, while Section IV provides some implementation details. Section V concludes the paper.

II. POWER MANAGEMENT STRATEGIES

E-Net-Manager relies on information provided by both the user (or system administrator) and presence detectors. Based on this information, it identifies periods when the user is far from her/his working area (e.g., office or building) and turns off the PC. Similarly, it switches on the PC as soon as the user approaches again the working area. Finally, the system allows the user to remotely check and control the status of her/his PC. More specifically, *E-Net-Manager* leverages the following three classes of events.

- *Static Control Rules*. Each user has an account on the system and can, thus, define rules for all the PCs associated with her/his account. It is possible to define rules to switch on, suspend and turn off a PC [5] at a specific time of the day (for any day of the week). The corresponding actions are executed automatically by the system, i.e., without any user intervention. By properly configuring the system, it is possible to keep the PC on only during working hours, while avoiding, at the same time, to annoy the user with undesired actions. For instance, in case of suspension or shutdown, the system preliminary checks whether the user is still using her/his PC. If so, the operation is deferred.
- *User Presence Detection*. Presence detectors allow to suspend/power off the PC also during idle periods that occur sporadically (e.g., due a meeting or class) and, hence, cannot be predicted in advance and reflected into the system through specific control rules. Information provided by presence detectors is used by the system to determine if the user is within the working area, or not. Whenever the system detects that the user is away from her/his working area, it sends a shutdown (or suspension) command to the PC. Conversely, as soon as the system detects the user presence nearby the working area, it sends a Magic Packet (see below) to her/his PC for powering it on (if it was previously suspended or powered off).
- *Remote Control*. E-Net-Manager also allows users to remotely control their own PCs. To this end, each user can access her/his personal account on the system and check the status of each registered PC (i.e., whether it is on, off, or sleeping). In addition, the user can perform specific actions, depending on the state of the PC. For instance, it is possible to turn off or suspend an active PC, or switch on a sleeping PC.

In addition to implement the above described strategies, *E-Net-Manager* also provides the following features.

- *Security and privacy*. Each communication within the system is secured through encryption and authentication so as to prevent malicious users from accessing critical information (e.g., presence of a user or status of a PC).
- *Low intrusiveness*. The system operates transparently to the user (i.e., without any interaction) while trying, at the same time, to avoid any possible discomfort to her/him.
- *Interoperability*. The system can manage any PC, irrespective of its operating system.

III. SYSTEM ARCHITECTURE

In this section we describe the architecture of *E-Net-Manager*. We assume that the system has to work in the context of a large network composed of many hosts and organized into several subnets. Also, the network can extend over multiple buildings, such as a university or corporate campus. Finally, the power management system is expected to work also in networks using dynamic assignment of IP addresses (DHCP), private IP addresses (NAT), or in the presence of Firewalls.

The system architecture follows the client-server paradigm and includes several components, as shown in Figure 2. Before describing the various components, however, it may be worthwhile introducing two basic technologies used by *E-Net-Manager*, namely *Wake-on-LAN* (WoL) and *eXtensible Messaging and Presence Protocol* (XMPP).

Wake-on-LAN [3] is a mechanism designed for Ethernet network interface cards (NICs) and allows the system to remotely turn on a powered off or suspended PC. To this end, a component of the NIC remains always active – only consuming a small amount of energy – so as to receive the "power on" command from the server. This command consists of a special packet called Magic Packet containing the MAC address of the target PC.

The *eXtensible Messaging and Presence Protocol* (XMPP) [6] is a message-oriented communication protocol based on XML. XMPP is used for communication between the different components of the system (i.e., PCs, server, proxies, and sensors). Specifically, each PC is an XMPP independent node with its own identifier (called *JID*). Similarly, the server has its own JID. Hence, all information exchanged within the system forms XML streams. In XMPP, connections are started by clients and are persistent. This allows a bidirectional communication also in networks with NAT servers or Firewalls. The connection between a PC and the Server can be created with both SSL and TLS encryption, thus achieving communication confidentiality and server authentication. Clients are authenticated using SASL (*Simple Authentication and Security Layer*) through JID-password pairs.

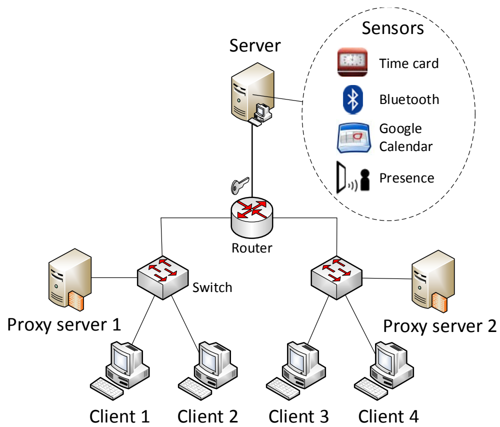

Figure 2 - Architecture of E-Net-Manager.

Let us now turn our attention to the various components of *E-Net-Manager* that are depicted in Figure 2. The *Server* side provides the power management service to controlled PCs. It consists of a number of software processes that are listed below, along with a brief description of their service.

- *Power Manager*. Manages static rules, information coming from sensor, and remote control requests; in addition, it sends commands to controlled PCs to change their energy status.
- *Database server*. Stores data about PCs and rules defined by users and the system administrator.
- *Web server*. Allows users to (**i**) check the status of their PCs, (**ii**) control them remotely, and (**iii**) define rules for "power on", "suspension" and "power off" actions.
- *XMPP server*. Allows the communication between the server and the PCs to control.

If the network consists of many subnets, the Server may not be on the same subnet of the PC to control. In such a case, a *Proxy server* is placed in each subnet, so as to remotely turn on PCs in that subnet. This is because the Magic Packet [3] is transmitted to the broadcast address of the subnet where the target PC resides and, hence, it requires that the sender and receiver are located on the same subnet. A Proxy is a host with a permanent connection to the Server. Whenever a "power on" rule for a PC becomes active, the Power Manager running on the server machine sends a "power on" command to the Proxy. Then, the Proxy sends the Magic Packet to the target PC.

The *Client* part of the system consists of all those PCs that are under the control of the power management system. Our system requires that a simple software module (*client module*) is installed on each PC. The behavior of the client module is fully transparent to the user. Whenever a rule defined for a PC becomes active, or a sensor detects the user's absence, the Power Manager sends a "power off" ("suspension") command to the PC. The client module on the PC is in charge of receiving and executing this command. However, to limit the intrusiveness of power management, it preliminary checks whether the command can be actually executed. For instance, a "power off" (or "suspension") command is aborted, and deferred, if the client module detects a mouse or keyboard activity.

The last, but not least important, component of our system is made up by presence detectors. While, in principle, any device can be potentially used in *E-Net-Manager* to detect the presence/absence of the user within/from the working area, the system is conceived to leverage presence detectors that are already available in the environment for other purposes, e.g., attendance recorders, card-enabled doors, Bluetooth-enabled phones, as well as software tools such as Google Calendar. Of course, it may happen that different detectors must be used depending on the context and users (e.g., attendance recorders are typically not used by faculty members).

IV. IMPLEMENTATION

We have completely implemented the server part of the system, the proxy, the client module for Unix and (Unix-like) operating systems, and the user interface. In addition we have integrated in our system two different kinds of presence detector, namely attendance recorders and Bluetooth phones. We provide below some implementation details about the different system components.

For the server part of the system, we used *MySQL* as a relational database to store data about users and PCs, as well as to store static rules defined by users. The XMPP service is provided by *eJabberd*, an open-source XMPP server. The Power Manager is structured as a *multi-threaded* (C++) program that uses the *Boost Asio library* to create timers for triggering actions defined by users (through static rules). The Power Manager periodically (once in a day) reads the rules in the database and activates specific timers for triggering commands at scheduled times. The Power Manager uses the *Gloox library* to implement the necessary XMPP functionalities.

Also the Proxy has been completely implemented. Currently it can register itself to the Server and communicate its own subnet address. In addition, it can receive "power on" commands from the Power Manager and send Magic Packets to the target PC.

The client module is currently available only for Unix (and Unix-like) hosts. It is implemented as a *daemon* starting at boot time and is able to (**i**) receive commands from the Power Manager, and (**ii**) power off or suspend the PC. It also monitors the mouse and keyboard activity in order to check the user's presence. Both the client module and the Proxy are programmed in C++ and use the *Gloox library*.

As far as presence detectors, presently we have integrated in our system the following devices.

- *Attendance Recorders*. We have managed the attendance recorder system in such a way that, whenever a user clocks in or out, a special message – containing the user id and the performed action (i.e., clock in/out) – is sent to the Power Manager. Since the user is entering/exiting his working area, the Power Manager will send an appropriate command to the user's PC (or corresponding Proxy).
- *Bluetooth-enabled phones*. We use an approach similar to that proposed in [7] that leverages the discovery mechanism used by Bluetooth devices. Specifically, we assume that the user has a Bluetooth-enabled mobile phone and the PC is equipped with a Bluetooth interface (both assumptions are quite common nowadays). In addition, the mobile phone and PC are configured in such a way that they establish a Bluetooth connection, whenever they happen to come in contact. A specific software module, running on the PC, periodically checks whether the mobile phone is still in contact (i.e., the user is within the coverage range of the Bluetooth receiver, which is in the order of 10 m). If the mobile phone is not detected for a certain time interval, the software module assumes that the user is far and puts the PC into the sleep state. Since the PC is in sleep state, the user can quickly resume it (manually) when she/he comes back. We verified that the reactivation process is very fast as the PC is again fully active in a few seconds. However, we are currently trying to automate the re-activation operation as well.

In addition to the above mentioned hardware detectors, we are currently trying to integrate also software detectors, such as Google Calendar. The idea is to exploit information stored in Google Calendar to trigger appropriate power management actions. To this purpose, when inserting an event in her/his calendar, the user should indicate – e.g., by selecting a specific color – whether the PC must be switched off (or suspended) during that event. In case, the system will automatically turn off the PC at the start time.

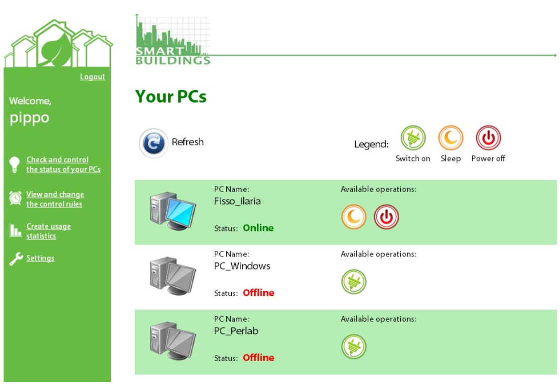

Figure 3 - Example of Web interface.

Finally, some few words about the user interface. *E-Net-Manager* is designed to be unobtrusive and, thus, it limits the user interaction to the minimum. The client module simply displays a notification message if a "power off" or "suspension" command has been received, so that the user can decide to abort it. In addition, the user can interact with the power management system through a Web interface. In our implementation we have used *Apache* as web server, and all web pages have been implemented in *PHP* and *Javascript* (using *jQuery*). The user can log in to the web server and perform all the supported actions, i.e., (**i**) check the status of her/his PCs and remotely control them; (**ii**) view/change the static rules defined for her/his PCs; (**iii**) view/change dynamic rules that leverage information from supported sensors. As an example, Figure 3 shows the page of a generic user. For each user's PC, the system displays its status, through proper icons. The user can perform specific actions, by clicking on specific buttons, so as to switch on, switch off or suspend her/his PC.

V. Conclusions

In this paper we have presented *E-Net-Manager*, an automated system for power management of networked PCs in large environments, that leverages information provided by presence sensors, in addition to static rules defined by the user. In principle, any presence detection system can be used. However, *E-Net-Manager* is intended to use presence detectors that are already available in the environment, so as to be cost effective and low intrusive. We have already implemented all the system components and we have integrated a couple of presence detectors in our system. As a further step, we plan to integrate additional hardware and software detectors. Also, we plan to deploy and test our system in a large computing environment (e.g., our university college).

References

[1] R. Bolla, R. Bruschi, K. Christensen, F. Cucchietti, F. Davoli, S. Singh, "The potential impact of green technologies in next generation wireline networks – Is there room for energy savings optimization?", IEEE Communication Magazine, Vol. 49, N. 8, August 2011.

[2] L. Chiaraviglio, M. Mellia, "Polisave: efficient power management of campus PCs", Proc. International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2010), Split,, Croatia, September 23-25, 2010.

[3] Magic Packet Technology, White Paper, Publication# 20213, Rev: A, Amendment/0, November 1995.

[4] K. Kurowski, A. Oleksiak, M. Witkowski, "Distributed power management and control system for sustainable computing environments", Proc. International Conference on Green Computing (IGCC 2010), Chicago, Illinois, USA, August 15-18, 2010.

[5] "Advanced Configuration and Power Interface Specification, revision 5.0", Hewlett-Packard, Intel Corporation, Microsoft, Phoenix Technologies, Toshiba, December 6, 2011 [Online]. Available at: http://acpi.info/DOWNLOADS/ACPIspec50.pdf

[6] P. Saint-Andre, "Extensible messaging and presence protocol (XMPP): core," RFC 3920, Internet Engineering Task Force, Oct. 2004 [Online]. Available at: http://www.ietf.org/rfc/rfc3920.txt

[7] C. Harris, V. Cahill, "Power management for stationary machines in a pervasive computing environment", Proc. Hawaii International Conference on System Sciences (HICSS 2005), Hawaii, USA, January 3-6, 2005.