2012 International Conference on Radar, Communication and Computing (ICRCC), SKP Engineering College, Tiruvannamalai, TN., India. 21 - 22 December, 2012. pp.48-53.

User Centric Framework of Power Schemes for Minimizing Energy Consumption by Computer Systems

P.K. Gupta[1] and G. Singh[2]

Abstract—**Recently, a lot of work has been carried out on the subject of "Green Computing", which represents an environmentally responsible way for reducing the energy consumption, and also addresses the various environmental related issues, like waste management, greenhouse gases, and so on. Green computing which is also known as sustainable computing also represents the sustainable architecture for software designing and the demand of building sustainable software's is on rise in the global market. This paper proposes an energy sustainable user centric framework for power schemes of operating system to handle the issue of energy consumption by computer system. In comparison to the existing framework of power scheme which is very much operating system oriented and uses the time-out approach of dynamic power management (DPM). The proposed framework is highly user centric and performs the continuous check for user activity on the computer system. This framework implements the three different power saving modes with the help of an algorithm which continuously checks for the power consumption by each running processes on the system.**

***Keywords*— Sustainable computing, power consumption, Shutdown, Hibernate, DVFS.**

I. Introduction

With the proliferation of Information and Communication Technology (ICT) devices, computer systems are considered as a high performance computing devices and consumes the varied amount of energy. The rate at which ICT devices are being produced is proportional to the increase in the energy consumed and heat dissipated by these devices, which poses the problem of an energy crisis and the exacerbation of the greenhouse gas problem and global warming. We cannot escape the fact that the world is becoming more and more dependent upon the use of information and communication technology (ICT), and that personal computers (PC) are one of the means. All over the world, PCs are being increasingly used right from kids to professionals in the course of their everyday lives, and this shows why we see such a phenomenal growth in sales of PCs over the last few years. In the late 1990s, power, energy consumption, and power density had become the limiting factors not only for the system design of portable and mobile devices, but also for high-end systems [1]. The design of computer system had changed from the performance-centric stage to power-aware stage. This scenario becomes disaster if we leave these computer systems running without any work as these systems will not only consume the energy but also produce a enough amount of heat, however both the conditions are not healthy for the environment, as energy consumption puts a lot of pressure to generate more energy and on the other hand heat produced is also responsible for green house gas effect and have various adverse effect on the environment. For example, if we consider the case of Microsoft Windows operating system then these power saving schemes exist from last two or more decade onwards in it with a little or say no change to them. This is also observed that in most of the cases either these power schemes are not properly configured or user has no knowledge of power management feature, which is responsible for power consumption by the computer system.

So, this surge in power consumption leads to a greater demand for power production by most of the developing countries, as their existing power production is insufficient to meet their citizens' demands, and they have to face a number of power cuts and load-shedding situations in order to maintain a fair balance between demand and supply. This emerging issue of power dissipation has imposed a very significant question on the system and software design and it is believed that in the future there will be a great demand for energy-sustainable software. Therefore, the vision of a sustainable planet and the minimization of the energy consumed by computer systems motivated us examine energy-sustainable computing methods [2].

II. Related Work

In early 1990s, PC energy consumption first entered the literature of the energy conservation communities [3]. The power requirements of computer system have changed considerably since the 1980s and are indicated in two modes. Active mode is when the device is in operation and Standby mode refers to a mode which

[1] Department of Computer Science and Engineering, Jaypee University of Information Technology, Waknaghat, Solan, India. Email: pradeep1976@yahoo.com

[2] Department of Electronics and Communication Engineering, Jaypee University of Information Technology, Waknaghat, Solan, India. Email: drghanshyam.singh@yahoo.com

attempts to conserve power with instant recovery. In newer systems the average power requirements is decreased by almost 50 percent, from nearly 100 watts (W) in the mid-1980s to 50 W in the mid-to late 1990s, and standby power consumption stayed constant nearly at 25 W, though overall computer power consumption is on the increase due to latest technology and standby power consumption is decreasing. The Pentium 4 computer systems consume more power than its predecessors at 67 W in active mode, while consuming only 3 W in Standby mode [4, 5].

In the early age, researchers tried to minimize the energy consumption during the architecture design stage, because the power problem obstructs the development of computer system. Most of the worked out techniques by the researchers are designed to decrease the energy consumption. Some of these techniques include multi-core on-chip processor, dynamic voltage and frequency scaling, clock gating, phase-change memory and solid-state disk drive are few of them used to decrease the power consumption in computer systems. Although these hardware based approaches have been proven useful for reducing the energy consumption of the computer system but it is also found that these techniques alone are not enough, and some higher-level strategies is to be used for reducing the energy consumption [6]. As we can see, that current operating systems are not by considering the power problem as if the system is running in idle mode it consumes a large amount of energy and considered as waste and not used for computing [7]. Zeng *et al.* present Ecosystem [8] which tries to manage power as one kind of system resource. Except for designing new operating systems, some other high-level power aware strategies, like power aware scheduling is also providing a better option for minimizing the energy consumption. In [9] Li *et al.* estimate the power dissipation caused by the operating system and find the power behavior of three types of operating system routines like interrupts, process and inter-process control, and file system. These operating system routines have different power behavior. However, they find that the power of these operating system routines has a linear relationship with instructions executed per cycle (IPC) and build the power model based on IPC. In [10] Do *et al.* build the process-level energy models for three main components CPU, disk, and wireless network interface card (WNIC). Here, the CPU energy model is based on system events like active time of the CPU, the time that the CPU worked on each frequency, and frequency transition time. The energy consumed by the disk and WNIC is computed with the amount of data operated by these devices. Chen *et al.* [11] investigated a user-level simulator at the micro-architectural and memory level and found that operating system activity is not modeled in them. They introduced the tool *SimWattch* – a system-level simulation tool and a flexible user-level simulation tool for predicting performance and power dissipation.

Dynamic power management (DPM) has become essential for modern computer systems as well as for battery driven embedded systems. DPM is an approach by which one can reduce power consumption by switching system components into different states. DPM also refers to the selective shutting or slowing down of computer system components that are idle for a long time or rarely used. Srivastava *et al.* [12] conducted an extensive analysis of various system predictive approaches and proposed a predictive system shutdown strategy for event-driven applications in portable devices. They developed two predictive formulas: one based on general regression-analysis techniques to compare the length of an upcoming off period with that of a previous one and the other obtained by the observation of on-off activity. Hwang and Wu [13] focused on the need to switch off a computer system running in idle or sleep mode and presented a predictive system-shutdown method to avoid sleep mode operations and thereby save energy when running event-driven applications. They used static power management and DPM techniques to define and detect the sleep modes and idle period. Zheng *et al.* [14] modeled the timeout policy driven power managed systems with multiple vacations and an attention span. They have presented their closed form solution based on queue theory. This analysis revealed a traffic load threshold based ‘best’ policy that is obviously equivalent to the optimal deterministic policy and therefore is inefficient to trade-off power for performance with tight constraints.

At last, with the intent of bringing power management into operating system’s control Dynamic voltage and frequency scaling (DVFS) is introduced into Advanced Computer Power Interface (ACPI) [15] and later in 1999, ACPI defines the various performance states to the CPU which is one of the part of operating system power management. From ACPI point of view, this prediction model utilizes the concept of a variety of C-states [15]. These C-states are C0, C1, C2 and C3. Where C0 refers to active state, and rest states are the idle state where power consumption of the CPU decreases as we move from C1 to C2 to C3. Though DVFS is a highly promising technology but system’s software needs to find the correct voltage and frequency to execute the application process on the computer system and if it is not correctly then the overall performance of the systems gets down and this results into more energy consumption by the non CPU components like memory and HDDs.

III. User Centric Power Management

Accurate and detailed monitoring of user activity is the basis for continuing to improve performance and energy efficiency of computing systems. Understanding user interaction can provide valuable information about which resources will be needed ahead of time. This leads to performance optimizations such as better resource allocations for applications that can utilize a given resource more productively. Energy efficient design requires systematic optimization at all levels of design abstraction: from process technology and logic design to architectures and algorithms [16,17]. There are various static techniques, applied during the design phase like at the algorithmic level – strength reduction [18], algorithmic and algebraic transformation [19, 20], and retiming, nd at logic level – logic minimization [21]. An alternative approach is to adjust the system operation and energy consumption to application workload dynamically, that is, during system operation and the method like application-driven management could be used.

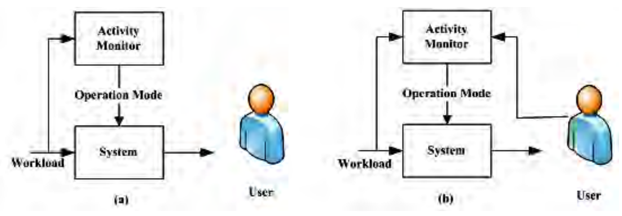


Fig. 1. Energy Management Schemes (a) Existing (b) Proposed

Despite differences, these dynamic methods exploit the same idea; namely to keep the system in lowest power mode whenever there is no activity inputs, and activate the system whenever the input signals change. To implement the idea, the system incorporates an extra unit that constantly monitors the input activity or workload and based on it determines the new operational mode for the system, as shown in Figure 1(a**).** Depending on the application, the workload can be measured by different metrics like the average rate, at which events arrive at processor, and idling time per sample interval. However, it is not always possible to make correct predictions due to peculiarities of application, operational environment, and/or user demands, which are varying in time. Existing energy management policies are device centric; that is they either ignore the user, assuming unchangeable operational environment for the device, or rely on very simplified policies, which causes to large energy losses. From here, it is clear if one want to reduce the energy losses then the device energy management must be user centric, that is, adaptable to varying user behavior. In the next section we have proposed the user centric approach to energy management, which monitors not only the system workload but also the user behavior and the environment. The main idea of our approach is to extend controller functionality to monitor the demands on system operation imposed by the user and adjust the system performance to the variation in these demands which is shown in Figure 1(b).

IV. Proposed User Centric Framework

This section discusses the proposed user centric framework of power schemes for Windows Operating Systems. This user centric framework as shown in Figure 2, starts its working with the user login into the system. In this framework repository is designed for operating system which plays the main role and keeps the record of all installed software and also assigns the various actions to installed software's. This repository also maintains the record of power consumed by each running process on the machine. Proposed framework implements the various techniques 1) smart wait (*i-W*ait) where a prompt is invoked to the user to input new login duration time when the previous duration gets over, 2) smart shutdown (*i-S*hut), where computer system gets shutdown to minimize the power consumption, and 3) smart hibernate (*i-H*ibnet), where computer system gets hibernated to minimize power consumption.

A. Proposed algorithm

Here, proposed algorithm computes the power consumption of each running process on the computer system and whenever it is found that the power consumption is below the defined threshold value which represents the condition that either machine is running in idle mode or user is not performing any operation on the computer system. Our proposed framework finds this functioning of the machine and stopped it to minimize power consumption. We have selected two different power saving modes known as hibernate and shutdown. While switching the state of the machine we have considered that the users should not lose their data so,

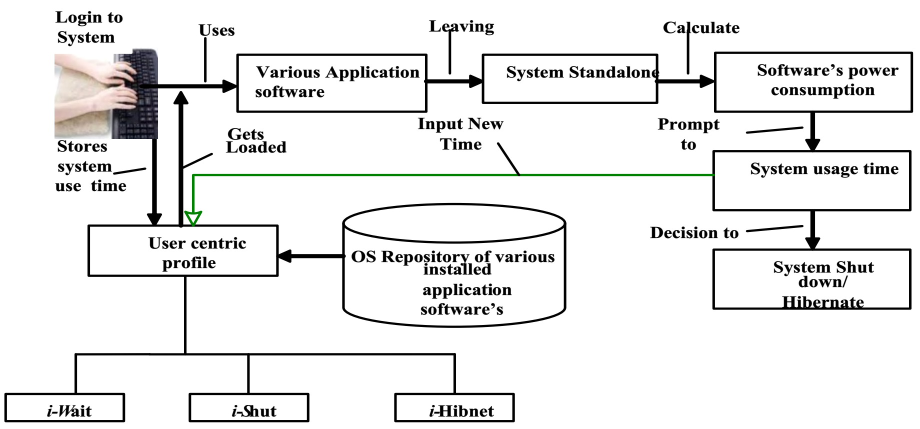


Fig. 2. Proposed user centric frameworks

this decision is based on the repository maintained as a part of operating system and categorize each running software process in to two categories known as lossless and lossy. User maintains this repository according to their choice.

Algorith m: Compute power consumption

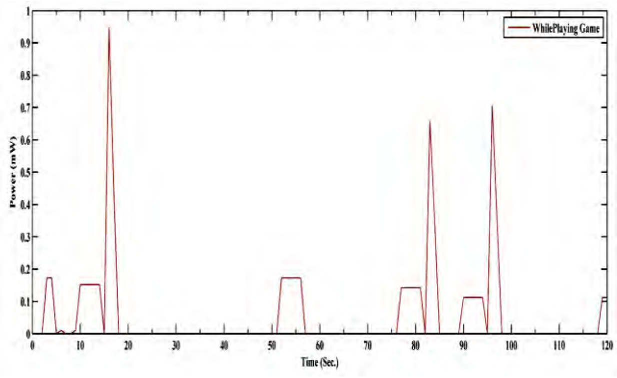
Symbols used:

Ξ	=	CPU Usage
Δ	=	UserTime
Δ_{NEW}	=	NewUserTime
Δ_{OLD}	=	OlduserTIme
η	=	KernelTime
η_{NEW}	=	NewKernelTime
η_{OLD}	=	OldKernelTime
τ	=	Threshold
ξ	=	Updatedelay
Λ	=	Rawpower

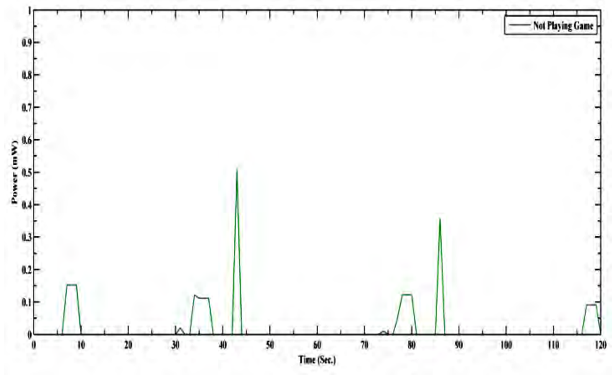
```
Begin:
Loads the user centric framework
Input the usage time $T_u$ and Threshold $\tau$
While (Tu)
iWait ← $T_u$
Start calculating the Rawpower or each running processes
Calculate total UserTime
$\Delta \leftarrow \Delta_{NEW} + \Delta_{OLD}$;
Calculate total KernelTIme
$\eta \leftarrow \eta_{NEW} + \eta_{OLD}$;
Calculate the Rawpower of each running processes;
$\Lambda \leftarrow (((\Delta + \eta) * 100) / \xi) * \Xi$;
Store the value of $\Lambda$ for each second into a file
   For (Rawpower consumption of each running process P1, P2…Pn)
   If ($\Lambda < \tau$)
   OS repository is checked for running process;
   If (Any application process is running)
      HIBERNATE;
      Else      SHUTDOWN;
      EndIf
   EndIf
   If (usage time has reached)
   Update usage time ($T_u$)
   EndIf
If (any action has not taken to update $T_u$)
   Wait for a minute;
   OS repository is checked for running process;
   If (Any application process is running)
      HIBERNATE;
      Else      SHUTDOWN;
      EndIf
Endif
End while
End:
```

V. Results

This section presents the detailed overview of results obtained for different scenarios like when there is no power consumption and when there is power consumption by the running processes on computer system. These results are obtained in real time environment for time period of 120sec. Here, Fig. 3(a) and Fig. 3(b) represents the power consumption by Bubble shooter Deluxe game while playing game and not playing game respectively. In detail analysis, while playing game then there is continuous power consumption by the game whereas if it is not being played then also a small amount of power is also consumed by the game. This may be because of various multimedia related elements available in the game and they force to consume the power. This is the scenario when one can minimize the power consumption. Our proposed algorithm and framework is capable to handle such scenario in effective manner.



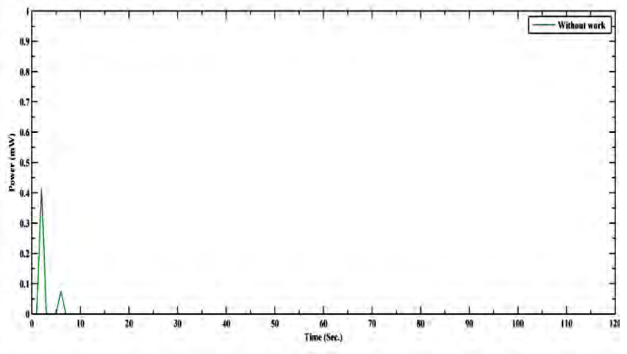
(a)



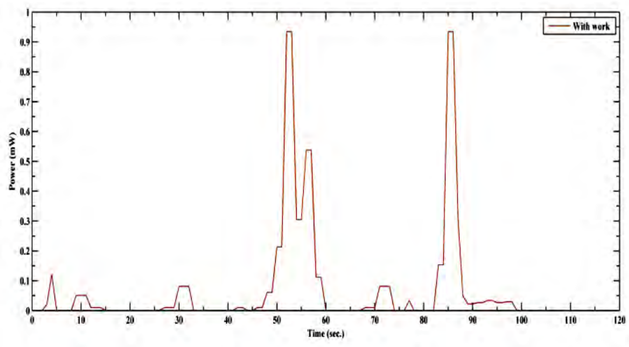
(b)

Fig. 3. Power consumption by Bubble Shooter Deluxe game (a) mode playing game (b) mode not playing game

Fig. 4(a) and Fig. 4(b) represent the power consumption scenario for Microsoft word in non-working mode and in working mode. From here, we gets a clear picture that when there is no working on the software, no power is consumed at all and during its working mode we can observe the various peaks.



(a)



(b)

Fig. 4. Power consumption by Microsoft Word (a) in non-working mode (b) in working mode

VI. Conclusion and Future Directions

In this paper we have suggested user centric approach to minimize the power consumption by computer systems for which we have proposed and implemented algorithm that continuously checks for power consumption of each running application process on the system and when it is found that there is no power consumption by the running processes or when it is found below the threshold then user centric profile takes the decision with the help of designed repository to switch the computer system into power saving mode. Here, the obtained results show the running state of each application processes when there is continuous processing on the machine as well as when there is no processing at all. We can easily find the human inactivity on the machine and switch the computer system to shutdown/hibernated mode to minimize the energy consumption by computer system. In future work, we can measure the performance of the framework with operating system and then necessary changes could be done in the proposed framework. Moreover, some other user centric methods could be evolved to minimize power consumption by computer systems.

References

[1] Ishfaq Ahmad and Sanjay Ranka, "Handbook of Energy-Aware and Green Computing" Chapman & Hall/CRC Computer and Information Science Series, CRC Press, 2012, pp. 1 – 1196.

[2] Sandeep K.S. Gupta, Tridib Mukherjee, Georgios Varsamopoulos and Ayan Banerjee, "Research directions in energy-sustainable cyber–physical systems," Sustainable Computing: Informatics and Systems, vol. 1, no. 1, pp. 57 – 74, 2011.

[3] Ruediger Kuehr and Eric Williams, "Computers and the Environment: Understanding and Managing their Impacts," Kluwer Academic Publishers, October 2010, pp. 1-285.

[4] J. Roberson, G. Homan, A. Mahajan, B. Nordman, C. Webber, R, Brown, M. McWhinney and J. Koomey, "Energy use and power levels in new monitors and personal computers," Berkeley, CA: LBNL, 2002.

[5] R. Howarth, B. Haddad, and B. Paton, "The economics of energy efficiency: Insights from voluntary participation programs," Energy Policy, vol. 28, pp. 477-486, 2000.

[6] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A framework for architectural level power analysis and optimizations," Proc. of 27th Annual International Symposium on Computer Architecture, Vancouver, Canada, pp. 83-94, 2000.

[7] P. K. Gupta, G. Singh, "Minimizing Power Consumption by Personal Computers: A Technical Survey", IJITCS, vol.4, no.10, pp.57-66, 2012.

[8] H. Zeng, C.S. Ellis, A.R. Lebeck, and A. Vahdat, "Ecosystem: Managing energy as a first class operating system resource," SIGPLAN Notices, vol. 37, no. 10, pp. 123-132, 2002.

[9] T. Li, and L.K. John, "Run-time modelling and estimation of operating system power consumption," SIGMETRICS Performance Evaluation Review, vol. 31, no. 1, pp. 160-171, 2003.

[10] T. Do, S. Rawshdeh and W. Shi, "ptop: a process-level power profiling tool", in Proceedings of the 2nd Workshop on Power Aware Computing and Systems (HotPower'09), October 2009.

[11] J. Chen, M. Dubois and P. Stenström, "Simwattch: integrating complete-system and user-level performance and power simulators", IEEE Micro vol. 27, no. 4, pp. 34–48, 2007.

[12] M. B. Srivastava, A. P. Chandrakasan, and R. W. Brodersen. "Predictive System Shutdown and Other Architecture Techniques for Energy Efficient Programmable Computation", IEEE Transactions on VLSI Systems, vol. 4, no. 1, pp. 42-55, March 1996.

[13] Chi-Hong Hwang and Allen C.H. Wu, "A Predictive System Shutdown Method for Energy Saving of Event-Driven Computation", ACM Transactions on Design Automation of Electronic Systems, vol. 5, no. 2, pp. 226–241, April 2000.

[14] Zheng R., Hou Jennifer C., Sha L.: 'On timeout driven power management policies in wireless networks'. Proc. IEEE Global Telecommunications Conf., Dallas, 2004, vol. 6, pp. 4097–4103.

[15] Hewlett-Packard Corporation, Intel Corporation, Microsoft Corporation, Phoenix Technologies Ltd. and Toshiba Corporation, "Advanced Configuration and Power Interface Specification", Rev – 5, December 2011, pp. 1 – 958.

[16] P. K. Gupta, G. Singh,"Energy-Sustainable Framework and Performance Analysis of Power Scheme for Operating Systems: A Tool," IJISA, vol.5, no.1, pp.1-15, 2013.

[17] P. K. Gupta, G. Singh, "Energy-sustainable snapshot algorithm for operating systems to minimize power consumption," Sustainable Computing: Informatics and Systems, pp.1-21, 2012. (Under review)

[18] A.P. Chandrakasan, M. Potkonjak, R.Mehra, J. Rabaey, and R.W.Brodersen, "Optimizing power using transformations," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol.14, no.1, pp. 12-31, January 1995.

[19] K.K. Parthi, "Algorithm transformation techniques for concurrent processors," Proceedings of IEEE, vol 77, no.12, pp. 1879-1895, December 1989.

[20] M. Potkonjak and J. Rabaey, "Fast implementation of recursive programs using transformations," Proceedings of the 1992 IEEE international Conference on Acoustics, Speech and Signal processing (ICASSP), vol.5, San Francisco, CA, pp. 569-572, March 1992.

[21] S. Malik and S. Devadas, "A survey of optimization techniques targeting low power VLSI circuits," Proceedings of the 32nd ACM/IEEE Design Automation Conference (DAC'95), San Francisco, CA, pp. 242-247, 1995.