高品質インターネット　5－13
(2002. 10. 11)

分散環境における携帯型計算機の資源管理機構の研究

杉浦　一徳[†]　小川　晃通[††]　中村　修[††]　村井　純[††]

† 通信総合研究所　〒184–8795 東京都小金井市貫井北町 4–2–1
†† 慶應義塾大学　〒252–8520 神奈川県藤沢市遠藤 5322
E-mail: †{uhyo,akimichi,osamu}@sfc.wide.ad.jp, ††jun@.sfc.wide.ad.jp

あらまし　本研究では，広帯域ネットワークに接続された内蔵バッテリによって運用されるノートブックコンピュータ (以下 ノート PC とする) の積極的な資源管理機能を実現する．ノート PC ではバッテリを最大限活用した連続運用が必要となる．しかし，高機能かつ高速なアプリケーション，デバイスを利用すると，プロセッサ，バス，メモリを最大限活用するため消費資源が増加する．ノート PC を特にストリームアプリケーションを焦点とした分散広帯域ネットワーク環境上で利用する場合，オペレーティングシステム，インターフェース，そしてアプリケーションとの有効な資源適応機能が必要となる．UNIX 環境において，資源適応・モニタ機構を実現することによって，ユーザアプリケーションの積極的な資源活用を可能とした．ACPI(Advanced Configuration and Power Interface) を用い，特に消費電力に対しノート PC 内の資源利用率を観測し，アプリケーションとネットワーク，各種利用デバイスの有効な資源活用を実現した．
キーワード　ノートブックコンピュータ，資源管理機構，DVTS

Resource Management Issues for Portable Mobile Computers in Distributed Networks

Kazunori SUGIURA[†], Akimichi OGAWA[††], Osamu NAKAMURA[††], and Jun MURAI[††]

† Communication Research Laboratory　4–2–1 Nukui-Kita Koganei Tokyo 184-8795 Japan
†† keio University　5322 Endo Fujisawa Kanagawa 252–8520 Japan
E-mail: †{uhyo,akimichi,osamu}@sfc.wide.ad.jp, ††jun@.sfc.wide.ad.jp

Abstract　This paper proposes agressive resource management mechanism suitable for trend portable computer connected to the high bandwidth network in battery operation. Note PCs are required to maximize the battery life for limited power supply gained. However in a highly intelligent, high speed devices; they require powerful accessibility to processors, bus, and memories, which consume more energy than the traditional simpler devices, thus reducing battery life. Using portable computers in distributed broadband network environemnt: especially focused on a high bandwidth stream media transport, requires highly redundant resource adaptation mechanism for applications, operating system, and interfaces which are being used. We constructed a resource adaptation and monitoring mechanism to typical Unix Operating system to maximize the resource availability to desired user application. With practical network operation using portable computers, we measured resource utilization especially focused on power consumption and interface utilization using Advanced Configuration and Power Interface (ACPI).
Key words　Notebook Computer, Resource Management, DVTS

1. Introduction

Mobile and portable computers such as portable computers(note PCs) and PDAs, are now becoming main computing and communication infrastructure [1]. Growth of portable devices: note PCs and PDAs, with conjunction of accessing Internet infractructure establishes the birth of ubiquitous environment. Continuous development of portability enhancement and increasing computation power, we see ever growing use of powerful microprocessors running sophisticated, intelligent control software in a vast array of devices including media devices such as digital video cameras, high bandwidth network devices such as a Fast Ethernet. Other than legacy computing devices, for example, cellular phones, information appliances, and digital home game machines are becoming intelligent and equipped themselves with networking facilities.

Unfortunately, there is inherence in the design goal between the portability and performance improvement of the device being used. Note PCs are required to maximize the battery life for limited power supply gained, therefore, reducing cosuming resource is a necessity. However in a highly intelligent, high speed devices; they require powerful accessibility to primary devices, such as processors, bus, and memories, consuming more energy than the traditional simpler devices, thus reducing battery life. In spite of continuous advances in semiconductor and battery capacitance supply technologies that allow microprocessors to provide much greater performance per energy consumption, and highly power consumed devices, to use more efficient power, the fundamental tradeoff between performance and battery life remains influential.

In mobile environment, or in the environment which requires collaboration of shared resources, adaptive and collaborative controls of resources are required. The arrangement or adaptation of applications with collaboration to the operating system, and the awareness of operating system with its components are necessary features for long-term usage under an unstable mobile environment with broadband network application. Without collaborations, packet losses as well as jitters in real-time network traffic will appear resulting flaws in trusted operations. In addition, futile attempt to the resource results waste of power consumption and less battery operation.

This paper proposes the practical resource adaptation mechanism for the Unix operating system, application, and application client using note PC in a broadband network infrastructure. Four demands for Adapting power management feature is especially focused on this research to maximize the effect of limited resources.

(1) Device Awareness

Specific signals, indicators, quantities, qualities, and status of the devices should be aware to the client using its devices: or client should have rights to examine, reallocate, and reassign the device status.

(2) Application Awareness

Application are to collaborate to operating system to adept changes in device states, such as a critical power conditions in battery life.

(3) User Awareness

User should also be aware of the system, and collaborate to application, and operating system for conditions that require physical changes: plugging the AC Adapters, replacing batteries, etc.

(4) Developer Awareness

Application developer should be aware of both device and user. For solution, common API is required for programming benefits.

2. Resource Management

Trend portable devices, especially note PCs, has embedded features to allow power savings for each devices installed. Power savings features are implemented in device basis, and can be controlled with simple access to the devices.

Abstraction of accessing devices for power savings and management are implemented as APM (Advanced Power Management). ACPI (Advanced Configuration and Power Interface) is developed as a standardization for recognition of device status and further device configuration [3]. However, common Unix operating system does not collaborate fully with its unique features available in ACPI, causing futile operations.

Focusing on the trend note PC processors, power management feature enables decreasing and increasing the processor voltage and frequency in steps to adjust the power consumption. Figure 1 shows the graph of processor performance and power consumption decrease. We used simple

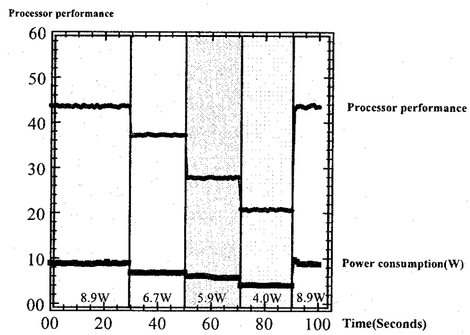

Figure1: Processor performance and power consumption

dhrystone based benchmark to measure the processor performance differentials running FreeBSD 4.2. When reducing the frequency of the processor, voltage will decrease in optimal steps automatically. A graph show reducing processor frequency and voltage effectively reduces the power consumption of note PC, somehow, results as a performance decrease. It is clear that optimizing the processor running frequency steps with processor utilization of the operating environment will effectively consume power consumption. Automatic frequency/voltage adjustment mechanism hard wired to Crusoe processors is one example of autonomous optimization method thus it can also be managed manually.

Spinning down the drive of the HDD when not in use is active solution to reduce power consumption. Douglis et. al. [2] shows that the disk subsystem on note PC consumes a major portion of available energy. Autonomous disk spin down by the countdown timer of the drive has been implemented by hardware since the early phase of the note PC production. Thus, autonomous disk spin down is an unfriendly solution for operating system and application, mostly results as an increase of power consumption.

When transferring or receiving considerable amount of network traffics; not rare under broadband network infrastructure, bus interrupts may occur per traffic packet basis. Especially in mobile environment, reducing the packet as well as a mechanism to decrease the interrupt per received packet is a major solution for resource savings. Reducing bus interrupt generations will effectively spare the utilization of the bus as well as the processor and memory, outcomes with less power consumption. Newly production of network interface chips considered for note PCs installs such method while it needs to be activated manually.

On a software basis, optimizing the packet sizes to MTU will effectively reduce interrupts generated by network interface, While in the Internet infrastructure, reassembling of packet may occur. Also, note that aggregation of packet size to MTU may also appear as a delay of incoming packet.

We have focused on above three devices to minimize the power consumption when they are used. Resource adaptation mechanism is introduced to monitor and configure the device management state with applications being used.

3. Resource adaptation

Implementation of resource adaptation provides direct and exclusive control over the power management and motherboard device configuration functions of a computer applying ACPI as an existing interface. Our goal is to have an interface capable of accessing throughout device, operating systems , application and application users. Applying ACPI is one simple method to have a bridge to device configuration and status recognition.

Implementation of resource adaptation was done based on FreeBSD 4. FreeBSD 5.0 covers ACPI implementation, we modified the 5.0 ACPI drivers for accessibility in 4.4. DVTS application is also modified with ACPI aware libraries managing ACPI devices.

"State handler daemon" is implemented as a Bi-directional transparent bridge between the device , operating system and application to establish monitoring and managing specific environmental changes in its devices. We defined the function of the circumstances in device as a "device state" or "state". Applications as well as operating system can be aware to "state" with monitoring methods description inside the kernel. Figure 2 lays out the software and hardware components relevant to resource management and how they relate to each other.

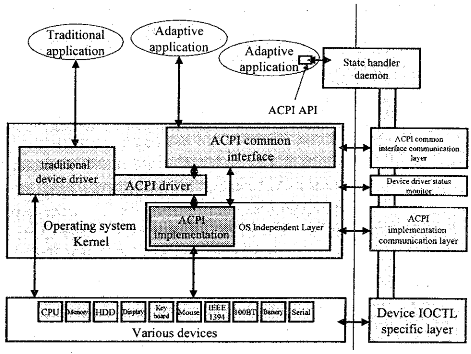

Figure2: Resource Management Interface

"ACPI common interface" implements and holds ACPI system description table enabling adaptive application to directly reconfigure the state of the devices. "State handler daemon" is an application-application interface enabling the monitoring and managing of the ACPI access device and also custom devices which are not implemented as an ACPI feature; for example, battery charge preset commands available on specific note PC with custom device drivers such as IBM Thinkpad. Traditional application, which is independent to power management, may directly accesses to the Unix legacy device drivers. With cooperation of "State handler daemon" monitoring application, users can manually assist traditional application to use resources effectively. In addition legacy "ACPI driver" defined as **/dev/acpica** is implemented as a Unix device driver, for traditional application with access to **/dev/acpica**.

Unix operating system abstracts configuration and querying kernel specific identifiers by users with "sysctl" interface. "sysctl" utility retrieves kernel state and allows processes with appropriate privilege to set kernel state. We have applied simple battery management features to comply "sysctl" interface. But applying "sysctl" interface to other device specific feature, especially network and processor interfaces, results as a rough and application un-aware design. For example, one application may request the processor to be running in lowest power mode available when other process requests the highest performance, "sysctl" may not be a applicable interface to cover such requests.

3.1 ACPI

The Advanced Configuration and Power Interface (ACPI) specification was developed to establish industry common interfaces enabling robust operating system to motherboard device configuration and power management of both devices and entire systems. ACPI is the key element in operating system directed configuration and Operating System Power Management(OSPM). ACPI includes the traditional power management BIOS code, Advanced Power Management (APM) application programming interfaces (APIs, PNPBIOS APIs, Multiprocessor Specification(MPS)) tables into a well-defined power management and configuration interface specification. ACPI provides the means for an orderly transition from existing legacy hardware to ACPI hardware, and it allows for both ACPI and legacy mechanisms to exist in a single machine and to be used as needed.

We implemented ACPI access to the devices described below to enable specific resource management.

- System power management

A mechanism capable of turning the system back and forth from sleeping states, including the device to wake the system itself.

- Device power management

Device table managing the power states, controls for putting devices into other power states. Enabling the system to put devices into different power modes.

- Processor power management

Manage the processor power state when system is idle.

- Device and processor performance management

Controls device and processor performance to achieve a desirable balance between performance and energy conservations.

- Dynamic configuration(Plug and Play)

Dynamic configuration of device arrangements.

- Device alert event handling
- battery Management
- thermal Management

Resources definition are stored as "ACPI namespace"; A hierarchical tree structure containing named objects.

3.1.1 ACPI common interface

ACPI drivers are implemented in some current operating systems such as Windows 2000, XP and FreeBSD 5.0. Though bi-directional device, operating system, application and user interface are not neatly assimilated. It was important to former operating system to support interoperability with traditional applications, causing complex interactions and operation against expectation such as sudden device status change during access. To establish a bi-directional mechanism, we implemented "ACPI common interface" inside the kernel independent to traditional ACPI driver. "ACPI common interface" supports ACPI layer and level access in common format per device basis via "ACPI System Description Table" in its name space. Description tables containing "Definition Blocks" is stored based on ACPI Machine Language(AML) under ACPI specifications.

When resource adaptation aware application accesses the device, device states are refreshed and hand shaken to application for the state disclosure, while resource adaptation unaware application with acpi support may access the traditional device driver, or in this case, client user may specifically notify or listen to device states by "ACPI Aware State handler daemon".

3.2 State handler daemon

"State Handler daemon" is a daemon process resident to operating system. Daemon encapsulates the clients ACPI access similar to traditional "sysctl" and system accounting interface. "State handler daemon" is a bi-directional interface capable of notifying events to applications or the application users. Figure 3 shows the devices supported by the "State handler daemon". Resource adaptation aware applications

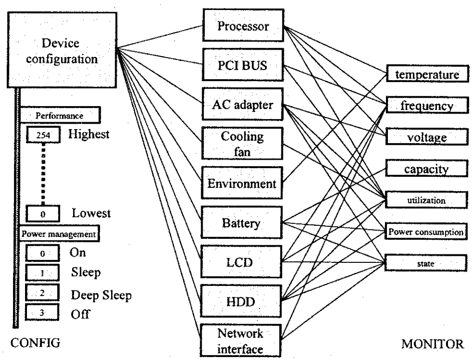

Figure3: State Handler Daemon

may access to the daemon to monitor and reconfigure the state. Some functions, such as access to utilization of cooling fan, network interface interrupt decrease state, and battery charging method, which are unique features implemented by note PC vendor, are accessible as a vendor unique configuration for device independency. We designed daemon to easily extend access to device dependent configuration method of an interface directly by ACPI implementation communication layer, while traditional ACPI implementation may have to implement a specific application to access the feature.

Configuration of devices based on performance is divided in 8bit priority as an abstraction. Power management features can be configured in 4 conditions.

- On
- Sleep State where the interface consumes less power than the "On" state. It requires minimal latency in order to

return to the "On" state.

- Deep Sleep State where the interface consumes a minimal amount of power. It requires large latency in order to return to the "On" state.
- Off

User can manually access to "State Handler Daemon" with simple access utility.

4. Evaluations

We modified DVTS as a sample application with resource adaptation by collaborating with "State handler daemon" [4] [6] [5]. Processor utilization and processor frequency is monitored through daemon to adaptively reconfigure the state. Figure 4 shows the DVTS implementation sample with adaptation to processor speed. DVTS receiver application requires processor resource due to its implementation in shared memory process, while sender application does not.

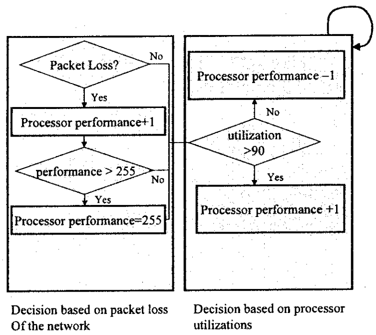

Figure4: Processor utilization adaptat ion in DVTS

For exclusive usage of note PC as a DVTS broadband conference system and to examine the efficiency of resource adaptation, we added an option to perform maximum efficiency. This option enables to run system fully optimized for DVTS. Devices that are not required to DVTS such as LCD panel and drive, are tuned to deep sleep state. Furthermore, vendor unique configuration, interrupt decreasing method in network interface, and processor fan configuration coupled with processor temperature and environmental temperature are implemented and managed via daemon.

To evaluate the effectiveness of resource adaptation mechanism, we constructed a variety of test-bed using note PCs. Also demonstration evaluation was held to realize the scalability issues. Table 1 shows the note PCs that are used for evaluations in this section.

4.1 Processor performance adaptation

To examine the manageability of processor performance adaptation of our resource adaptation mechanism implementation, simple packet loss evaluation is tested. Two note PCs are located at each end considering one as a DVTS sender application, and another as a DVTS receiver application connected via 100Base-T Fast Ethernet network. DVTS sender application sends images captured by DV Camera, and DVTS receiver receives packets from networks, and sends DV packet to IEEE1394-NTSC media converter.

Table1: Note PC used for evaluations

Model	CPU	Clk	Mem	NIC	IEEE1394
SHARP1	M-P3	600	128	RealTek	OHCI
IBM1	M-P3M	1133	640	Intel	PCMCIA
FUJITSU1	Crusoe	533	128	Intel	OCHI
CASIO1	Crusoe	600	192	RealTek	OHCI
IBM2	M-P3	800	640	Intel	OHCI
DESK1	P3	1000	256	Intel	OHCI

We chose "IBM2", "CASIO1" and "FUJISU1" model to evaluate the feature of processor performance adaptations. Other model does not implement frequency adjusting mechanism in its hardware. FreeBSD with resource adaptation mechanism is used as an operating system.

Figure 5 shows packet losses created by the manual resource reconfiguration of processor frequency in receiver note PC "FUJITSU1" which equips with the slowest processor. When the processor frequency is configured as a maximum performance mode, there are no packet losses in DVTS receiver, while in minimum duty mode, average of 38 packet losses are recognized. Other model; "CASIO1" generated an average of 14 packets in slowest frequency while "IBM1" did not generate any packet losses in slowest configurable frequency.

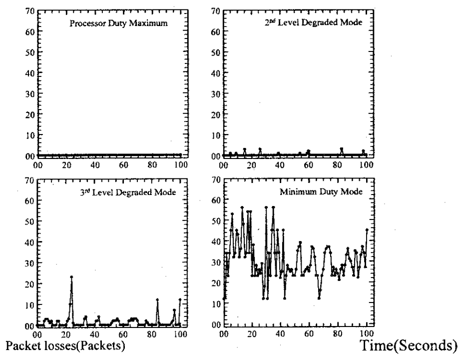

Figure5: Packet losses in DVTS and processor frequency relations

4.2 Power consumption

We evaluated our system compared to other operating system internal implementation; Windows environment. Windows environment implements ACPI and custom device driver provided by the note PC manufacturer which enables power management scheme. Though this evaluation is hard to compare because of totally different operating system architecture, we decided to evaluate the power consumption during the stable DVTS operation. DVTS does not support receiving application in Windows environment, we compared results in sending applications only. Table 2 shows the average running time in battery operation under different operating system platforms. We used "FUJITSU1" note PC for evaluation, We did not enable the LCD and drive power

Table2: Evaluation of dvsend in different operating systems

OS	Time(min.)
Our Imp.	84
FreeBSD 4	75
NetBSD 1.5	74
Linux 2.4	76
WIN ME	80
WIN 2000	78
WIN XP	77

management in our implementation. Windows architecture shows slight advantages than traditional FreeBSD and other Unix based operating system due to custom device drivers.

Figure 6 shows the difference of decrease in battery capacity between traditional and resource adaptive model of same note PC. This evaluation is tested with simple processor fre-

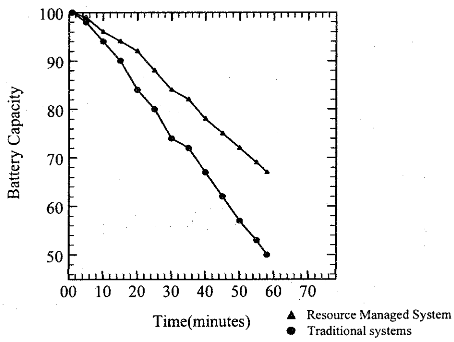

Figure6: Battery capacity decrease

quency adaptation only. compared with traditional system, with autonomous device management not linked with operating system or the application, about 15% of battery life is extended with processor adaptation. We evaluated all note PC and average of 15% extension in battery operation is realized.

5. Contributions and future works

Managing computer resource is a basic principle of operating system architecture, while current environment does not fully guarantee the effective application and user collaborations under variety of devices and networks. Resource adaptation mechanism is a necessity for collaborating with devices, operating systems, applications, clients and developers.

Providing users with applications and services by the cooperation of various software components taking advantage of the deployment of multiple communication facilities including the component's mobility on top of aforementioned hardware platforms is our contribution.

6. Conclusions

In this paper, we implemented agressive resource management mechanism suitable for trend portable computer connected to the high bandwidth network in battery operation. In such environment, where occasional environmental changes take in place rapidly, adaptive and collaborative management of resources are required. Optimization of resource based on power supply management is dedicated to the operating environment. Bi-directional, user, application, operating system, and device aware interface for resource configuration/management is extended to the current Unix operating system by implementing application programming interface of Advanced Configuration and Power Interface(ACPI). "State handler daemon" interface enables applications and application users to monitor and adapt the resources available during the operation. Thick collaboration between application and devices that are reserved in its limited environments economizes the consuming device utilization. Contribution of our implementation shows: for example, life extension of battery life and effective network bandwidth adaptation.

Bibliography

[1] Kazunori Sugiura, Kimio Ishii, and Jun Murai, "Implementation and Evaluation of Mobile Computing Environment", IPSJ-92-DPS-60, Information Processing Society of JAPAN, March, 1993.

[2] Fred Douglis, P. Krishnan, and Brian Marsh, "Thwarting the Power Hungry Disk", USENIX Conference, January, 1994.

[3] Intel, Microsoft, Compaq, and Toshiba, "Advanced Configuration and Power Interface Specifications", ACPI Group, 2001.

[4] Kazunori Sugiura, Takeshi Sakurada, and Akimichi Ogawa "Demonstration of High Quality Media Transport System Using Internet", IPSJ Journal, Vol.41, No12, pp.1321-1326, 2000.

[5] Akimichi Ogawa, "DVTS (Digital Video Transport System) WWW page", "WWW page", Nov. 2001.

[6] Akimichi Ogawa, Katsushi Kobayashi, Kazunori Sugiura, Osamu Nakamura, and Jun Murai, "Design and Implementation of DV Stream over Internet", IWS99, Feb. 1999.