Chapter XII
Device Driver Based Computer in Broadband Age

Yoshiyasu Takefuji
Keio University, Japan

Koichiro Shoji
SciencePark Corporation, Japan

Takashi Nozaki
SciencePark Corporation, Japan

ABSTRACT

In this chapter, we present a device-driver-based computer that realizes the reduction of mode (domain or vertical) switching overheads between user and kernel mode with innovative attributes, including shared keyboards and mice, access-controlled files, and timed files. Experimented results show that old personal computers can revive again with the proposed Driverware technology. The proposed Driverware can improve the CPU resource utilization by three times.

BACKGROUND

On April 19, 1965, Moore predicted that the number of transistors per integrated circuit would double every 18 months (Moore, 2001). Ruley showed that the speed of CPU has been double every 18 months (Ruley, 2001). Although the recent progress of semiconductor technology has been providing us over 2 GHz (gigahertz) CPU devices, you may not be able to obtain such dramatic improvements with your personal computer because you do not sense the high-speed CPU devices. In existing popular operating systems, including Microsoft Windows, Macintosh, Linux, and FreeBSD, your user software programs do not maximize the performance of such a high-speed CPU and other resources. In this chapter, a device driver-based computer is proposed, where

the "Driverware" software program plays a key role. "Driverware" allows you to dramatically improve user software programs in your personal computer without replacing the current operating system. The technical aspects of the root of all evil that waste your CPU resources in user software programs are caused by device driver programs. A device driver is a software program to control a hardware component or peripheral device of a computer, such as a hard disk, a CD/DVD player, a network device, a mouse, keyboard, a video card, and so on. Driverware provides the current operating system a thin layer closest to the hardware layer, where communications and controls among device drivers can be established. Overheads involved in device drivers are significantly reduced by the driver-to-driver communications and controls. When playing a CD/DVD audio/video player or a network video stream on your GHz CPU-based personal computer in the broadband network, you may still face frame drop or abrupt audio skips if you do Web browsing simultaneously. Four device drivers, including a network device driver, a video device driver, audio device driver, and a display device driver, are involved in this network video stream processing. If the priority of network device driver is lower than that of the other drivers, you may face the frame drop. If four device drivers are properly balanced and their priorities are taken care of, then you will be satisfied with the current network video stream without using a special hardware such as a network video stream receiver. Driverware allows device drivers to communicate each other, and to efficiently control computer resources in order to reduce overheads involved in device drivers. Frame drops and abrupt audio skips are minimized by the proposed Driverware. Until the advent of Driverware, there has been no general-purpose device-driver-based computer. The key element of Driverware is based on distributed delegation of authority, while conventional user programs are based on the centralized control. In user programs, overheads are caused by unnecessary thread* switching in a process, context (horizontal) switching between processes, and mode (domain or vertical) switching between kernel and user mode. The proposed Driverware contributes to reduce the mode-switching overheads. We have measured the mode-switching overheads involved in device drivers for the first time in the world. Based on the measured result, your old personal computers in the garage can revive again.

The framework for minimizing overheads of vertical and horizontal switching was proposed in 1994 (Inohara & Masuda, 1994). Horizontal switching problems in user mode were discussed in Borg (2001). Ousterhout has measured the mode and context-switching overheads in RISC/CISC machines under the Unix operating system in 1990 (Ousterhout, 1990). In 1996, Lai and his colleagues measured the mode and context-switching overheads in Intel x86-based machines under Linux/FreeBSD/Solaris operating systems, respectively (Lai & Baker, 1996). Based on the existing results, reducing the context-switching overheads plays a key role in overall system performance, where the context switching overhead is larger than the mode-switching overhead by 10 times. Ultimately optimized context-switching operating system, RT-Linux has been introduced (Ramamritham & Stankovic, 1994; Zhou & Petrov, 2006). Based on the context-switching overhead reduction, TUX Threaded Linux Web server (Bar, 2000)), and other Web servers have shown the high performance, respectively (Quynh & Takefuji, 2006).

In the real world, the majority of users use Microsoft operating systems for personal use (Sun, Lin, & Wu, 2006). Based on our study, the mode-switching overhead reduction is more important than the context-switching overhead reduction under Microsoft operating systems. Figure 1 shows the table of the mode-switching and context-switching overheads, where more than 2 months were used for switching overhead measurement. In Windows 2000 operating system (Microsoft, 2006), 7.4 microseconds are required