제 18 회 한국정보처리학회 추계학술발표대회 논문집 제 9 권 제 2 호 (2002.11)

Windows NT 기반의 가상 디스크 설계 및 구현

최수원[0], 고정국
동명정보대학교 컴퓨터공학과
e-mail : ieeo75@empal.com

Design and Implementation of Windows NT-Based Virtual Disk

Su-Won Choi[0], Jeong-Gook Koh
Dept. of Computer Engineering, Tongmyong University of Information Technology

요 약

일반적으로 사용자들은 파일을 효율적으로 관리하기 위해 대용량의 하드 디스크에 다수의 파티션을 생성하여 사용한다. 그러나, 컴퓨터 활용에 익숙치 않은 사용자들의 경우에는 체계적인 파일 관리가 되지 못하여 많은 애로를 겪기도 한다. 특히, 하나의 컴퓨터를 여러 사람이 공동으로 사용할 경우에는 체계적인 파일 관리의 부재로 인한 파급 효과가 더욱 클 것이다.

본 논문에서는 기존의 디스크 파티션을 변경하지 않으면서 사용자 입장에서는 개인별로 별도의 디스크 파티션이 부여된 것처럼 느끼도록 하여 파일 관리에 도움을 줄 수 있는 가상 디스크 프로그램을 설계하고 구현하였다.

1. 서론

컴퓨터 산업과 통신 기술이 빠르게 발전함에 따라 컴퓨터가 널리 보급되고 많은 사람들이 가정과 사무실에서 개인용 컴퓨터를 사용하고 있다. 현재 개인용 컴퓨터의 성능이 나날이 향상되고 있으나 컴퓨터 활용에 익숙치 않은 사용자들은 대용량의 하드 디스크에서 체계적인 파일 관리가 이루어지지 않아서 많은 애로를 겪기도 한다. 특히, 하나의 컴퓨터를 여러 사람이 공동으로 사용하는 경우에는 본인뿐만 아니라 타인에게도 많은 지장을 주게 된다.

만약 하나의 컴퓨터를 여러 사람이 공동으로 사용하는 경우에 사용자 이름의 디스크 파티션이 주어진다면 개인 파일이 여기저기 산재되어 겪게 되는 어려움도 최소화될 것이다. 본 논문에서는 기존의 디스크 파티션을 변경하지 않으면서 사용자 입장에서는 개인별로 별도의 디스크 파티션이 부여된 것처럼 느끼도록 하여 파일 관리에 도움을 줄 수 있는 가상 디스크 프로그램을 설계하고 구현하였다.

본 논문의 구성은 다음과 같다. 2 장에서는 가상 디스크에 관련된 기술들을 기술하고, 3 장에서는 가상 디스크의 기능설계, 동작 방식 및 구현 내역을 설명한다. 마지막으로 4 장에서는 본 논문의 결론 및 향후 연구과제에 대하여 기술한다.

2. 관련기술

2.1 Windows NT Kernel

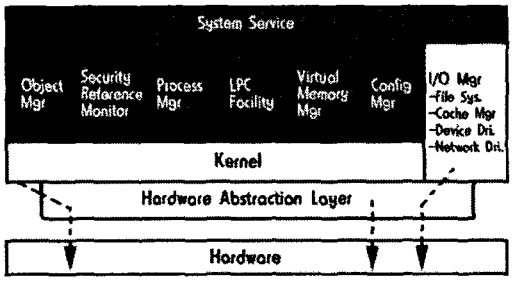
"클라이언트/서버"와 "모듈화" 기반의 윈도우 NT 는 [그림 1]과 같이 하드웨어 추상화 계층(HAL)과 실행부(executive), 마이크로 커널로 구성되어 있다.

하드웨어 추상화 계층은 NT 실행부와 하드웨어 플랫폼 사이에 위치한 코드계층이며, 하드웨어로부터 윈도우 NT 의 나머지 부분을 격리시켜 시스템의 이식성을 제공한다.

윈도우 NT 실행부는 그 자체가 운영체제로서 시스템 서비스와 내부 루틴으로 구성되어 있으며, 사용자 모드 응용 프로그램과 서브 시스템으로부터 보호되어야 한다. 커널 모드에서 실행되는 컴포넌트들은 하드웨어와 소프트웨어 자원을 직접 액세스한다. 윈도우 NT 에서는 시스템의 보안성과 안정성을 유지하기 위해 운영체제의 핵심 부분만을 커널에서 실행하는데,

NT 실행부가 커널 모드에서 실행된다. 나머지 모든 부분은 사용자 모드에서 실행된다.

윈도우 NT 의 마이크로 커널은 NT 실행부의 일부로서 NT 실행부와 저수준의 운영체제 프리미티브를 통해 통신한다[9].



[그림 1] Windows NT 의 구조

2.2 윈도우 NT Kernel 드라이버

윈도우 NT 의 Kernel 드라이버는 Non PnP, Non Power Driver 로서 장치에 종속적이지 않으며, CPU 의 특권 레벨 또는 Supervisor 레벨(인텔 아키텍처의 Ring 0)에서 실행되는 32 비트 모듈 컴포넌트이다. VxD(Virtual Device Driver)가 메모리에 적재되면 VMM(Virtual Machine Manager)의 일부가 되는 것처럼 윈도우 NT 드라이버도 적재 후에는 Kernel 의 일부가 되어 Kernel 과 동일한 권한을 갖는다. 또한, 윈도우 NT 드라이버는 적재된 후 I/O Manager 의 관리를 받는데, I/O Manager 는 드라이버를 적재하고 관리하는 환경을 제공한다[8]. 윈도우 NT Device Driver 의 Kernel Object 들은 <표 1>과 같다[3,4,5].

<표 1> Kernel Object 요소

Kernel Object	설 명
DriverObject (PDRIVER_OBJECT)	장치 객체와 관련된 장치 구동기를 나타내는 객체
DeviceObject (PDEVICE_OBJECT)	장치 구동기가 관리하는 장치에 대한 장치 Object 자료구조의 리스트를 가리킴
DeviceExtension (PDRIVER_EXTENSION)	AddDevice(PDRIVER_ADD_DEVICE) 멤버만 액세스할 수 있는 작은 하위 구조체를 가리킴
IRP (I/O Request Packet)	Kernel 에서 실행되는 입출력관리자 루틴의 일부분으로 정의된 자료구조의 커널 Object
Device Queue Object	장치 Object 는 세 가지 상태를 가짐. 이 상태가 장치 큐의 지원 루틴을 구동하는데 영향을 미침
Interrupts Object	데이터 전송이 종료되면 ISR 이 H/W 인터럽트를 알림
DPC(Deferred Procedure Calls) Object	DpcForIsr 루틴을 사용하며, ISR 이 DISPATCH_LEVEL 에서 제어권을 넘겨받을 때 요청됨. 가장 최근의 인터럽트에서 발생한 IRP 를 처리함
Dispatcher Object (Timer,Event,Semaphore,Mutex)	일반적으로 생성, 폐쇄, 읽기, 쓰기, IOCTL, 내부 IOCTL IRP 처리 Object 를 말함
Controller Object	데이터 읽기나 쓰기에 사용됨
Adapter Object	DMA 어댑터 채널 이용시 사용됨

3. 가상 디스크의 기능 설계 및 구현

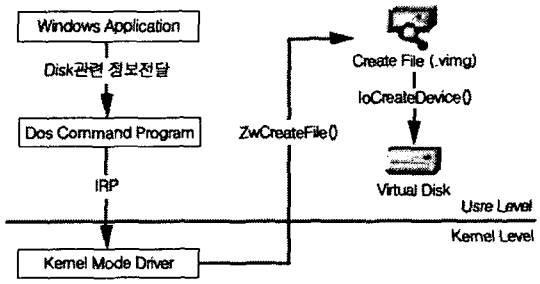
3.1 가상 디스크의 기능 설계

본 논문에서는 컴퓨터 사용에 익숙치 않은 초보자나 하드 디스크의 파일 관리에 어려움을 겪는 사용자에게 편리한 파일 관리 기능을 제공할 수 있는 가상 디스크(Virtual Disk) 프로그램을 설계하고 구현하였다. 설계된 가상 디스크 프로그램은 윈도우 NT/2000 에서 동작하며 디스크 이미지 파일(.vimg)을 이용하여 손쉽게 가상의 디스크를 생성하고 삭제할 수 있다.

가상 디스크 프로그램은 사용자 프로그램의 하부 구조를 숨김으로써 사용자에겐 사용자 응용 프로그램과 생성된 가상 디스크 이미지 파일만이 보여진다. 즉, 사용자는 가상 디스크에 대해 자세히 알지 못하더라도 본 논문에서 구현한 가상 디스크 프로그램을 이용하면 기존의 디스크 파티션을 변경하지 않고도 가상 디스크를 생성하여 사용하고 필요가 없으면 이를 삭제할 수 있다.

한편 가상 디스크가 생성되어도 시스템에는 가상 디스크에 해당하는 물리적인 장치가 존재하지 않는다. 사용자는 가상 디스크 이미지 파일과 가상 디스크의 생성을 위해 가상 디스크 프로그램을 이용하는데, 내부적으로는 ZwCreateFile() 함수와 IoCreateDevice() 함수를 이용하여 가상 디스크 이미지 파일에 접근하게 된다.

가상 디스크의 생성 방식은 [그림 2]와 같다. 가상 디스크를 새로 생성하거나 기존의 가상 디스크를 개방하려는 사용자는 가상 디스크 프로그램에서 가상 디스크 생성 항목들을 선택한다. 그러면 가상 디스크 프로그램이 디스크 생성 정보를 DOS 명령어 프로그램에게 전달해 주며, DOS 명령어 프로그램은 관련 정보를 수록한 IRP(I/O Request packet)를 커널 내의 가상 디스크 구동기(Virtual Disk Driver)에게 전달한다. 가상 디스크 구동기는 IRP 를 수신한 후 가상 디스크 이미지 파일을 생성하거나 개방하기 위해 ZwCreateFile() 함수를 이용하여 핸들을 오픈한다. 이후 IoCreateDevice() 함수를 이용하여 가상 디스크를 생성하게 된다. 가상 디스크 구동기는 사용자가 가상 디스크를 제거할 때까지 생성된 가상 디스크의 이미지 파일을 이용하여 가상 디스크를 가리키므로 가상 디스크를 사용할 수 있다.



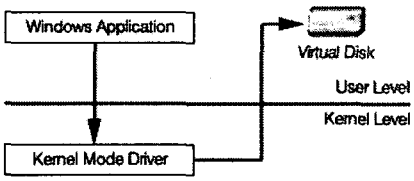
[그림 2] 가상 디스크의 생성 방식

가상 디스크를 제거할 때는 응용 프로그램에서 ZwClose(hfile) 함수를 이용하여 가상 디스크 구동기가 가리키던 이미지 파일의 핸들을 닫음으로써 가상 디스크를 제거한다.

3.2 가상 디스크의 동작 방식

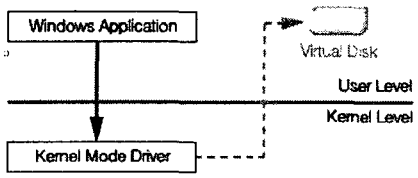
본 논문에서 설계한 가상 디스크는 윈도우 NT/2000이 초기화되면 가상 디스크 구동기 서비스를 시작한다. 시스템이 종료되거나 재부팅되면 가상 디스크 구동기 서비스도 중단되어 가상 디스크가 제거된다. 그러나, 가상 디스크에 저장된 사용자 데이터는 가상 디스크 이미지 파일 속에 이진 데이터 형태로 저장되어 있기 때문에, 필요하면 언제든지 "가상 디스크 열기" 기능을 이용하여 가상 디스크를 복원할 수 있다.

[그림 3]은 가상 디스크의 생성 과정을 나타낸다. 사용자가 가상 디스크 프로그램을 이용하여 일정 용량의 이미지 파일을 생성하게 되면, 가상 디스크 구동기가 이를 이용하여 가상 디스크를 생성한다.



[그림 3] 가상 디스크의 생성 과정

사용자가 가상 디스크를 의도적으로 제거하거나 시스템을 종료하더라도 가상 디스크 구동기와 가상 디스크간의 링크만 끊어질 뿐 사용자가 생성한 가상 디스크 이미지 파일과 가상 디스크에 저장되어 있던 데이터는 그대로 남아있다. 따라서, 기존의 가상 디스크 이미지 파일을 이용하여 가상 디스크 생성시 지정해 준 디스크 용량으로 가상 디스크를 다시 개방하여 가상 디스크 구동기와의 링크를 설정하면 가상 디스크 제거시와 동일한 가상 디스크를 복원할 수 있다.



[그림 4] 가상 디스크의 제거 과정

3.3 가상 디스크의 구현

본 논문에서는 가상 디스크의 기능을 설계하고 구현한 후 사용자들이 가상 디스크 기능을 편리하게 사용할 수 있도록 그래픽 사용자 인터페이스도 제공하였다. 가상 디스크를 구현하고 기능을 시험한 환경은 다음과 같다.

1) 운영체제
   - Windows 2000 Professional (구현/기능시험)
   - Windows 2000 Server (기능시험)

2) 개발도구
   - Visual Studio 6.0
   - Windows 2000 DDK

3) 디버깅도구
   - WinDbg 6.0
   - DebugView 4.13

가상 디스크 프로그램은 ZwCreateFile() 함수를 사용하여 가상 디스크 이미지 파일을 생성한다. [그림 5]는 가상 디스크 이미지 파일을 생성할 때 VdiskOpenFile() 함수에서 ZwCreateFile() 함수의 사용 예를 보여준다.

```
NTSTATUS
VDiskOpenFile (IN PDEVICE_OBJECT DeviceObject, IN PIRP Irp)
{
    ...
    status = ZwCreateFile(
        &device_extension -> file_handle,
        device_extension->read_only ? GENERIC_READ :
        GENERIC_READ | GENERIC_WRITE,
        &object_attributes,
        &Irp -> IoStatus,
        NULL,
        FILE_ATTRIBUTE_NORMAL,
        0,
        FILE_OPEN,
        FILE_NON_DIRECTORY_FILE |
        FILE_RANDOM_ACCESS |
        FILE_NO_INTERMEDIATE_BUFFERING |
        FILE_SYNCHRONOUS_IO_NONALERT,
        NULL,
        0 );
    ...
```

[그림 5] 가상 디스크 이미지 파일 생성 코드

I/O Manager는 구동기가 시스템에 적재될 때 구동기 객체를 만들고 구동기의 초기루틴을 요청한다. 적재된 구동기는 장치나 구동기에 대한 인터페이스를 나타내기 위해 IoCreateDevice 를 요구함으로써 장치 객체를 생성할 수 있다. [그림 6]은 장치를 만들기 위해 IoCreateDevice() 함수를 호출하는 코드이다.

```
status = IoCreateDevice(
        DriverObject,
        sizeof(DEVICE_EXTENSION),
        &device_name,
        FILE_DEVICE_DISK,
        0,
        FALSE,
        &device_object
        );
    ...
```

[그림 6] 가상 디스크 장치 생성 코드

IoCreateDevice() 함수를 호출하여 생성할 수 있는 장치의 종류는 <표 2>와 같다[10].

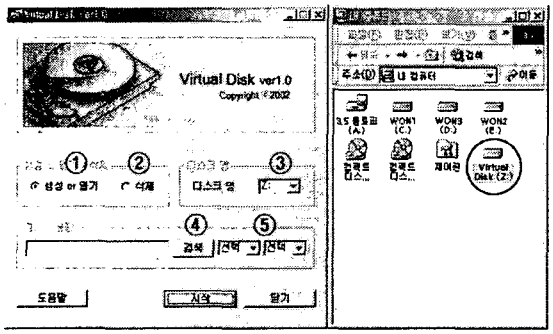
<표 2> IoCreateDevice() 함수로 생성할 수 있는 장치들

BEEP	MOUSE	BATTERY	VIRTUAL_DISK	INPORT_PORT
TAPE	MIDI_IN	MIDI_OUT	KEYBOARD	PARALLEL_PORT
NULL	PRINTER	SCANNER	WAVE_OUT	SERIAL_PORT
DVD	SOUND	CD_ROM	UNKNOWN	SERIAL_MOUSE_PORT
DISK	WAVE_IN	VIDEO	8042_PORT	MASS_STORAGE
KS	CHANGER	SCREEN	CONTROLLER	

가상 디스크 프로그램은 가상 디스크 이미지 파일

을 이용하여 가상 디스크를 생성하거나 삭제할 수 있다. 가상 디스크 프로그램의 활용 방법은 다음과 같다. 사용자가 가상 디스크를 생성할 때는 ①번 항목의 "생성 or 열기"를 선택한 후 ③번의 "디스크 명"을 지정한다. 그리고 가상 디스크 이미지 파일이 저장될 경로를 ④번의 "검색" 버튼을 이용하여 지정하고 ⑤번에서 생성될 "디스크 크기"를 선택한다. 마지막으로 "시작" 버튼을 누르면 가상 디스크가 생성된다.

본 논문에서는 가상 디스크 프로그램의 기능 시험을 위해 디스크 명이 "Z"인 가상 디스크를 생성하였다. 가상 디스크 프로그램을 이용하여 가상 디스크를 생성한 결과는 [그림 7]과 같다.



[그림 7] 가상 디스크 프로그램의 기능 시험

가상 디스크를 생성할 때는 가상 디스크가 생성될 물리적인 실제 디스크의 여유 공간을 고려해야 한다. 즉, 사용자는 가상 디스크의 크기를 실제 디스크의 여유 공간보다 작게 설정해야 한다. 만일 여유 공간보다 가상 디스크를 크게 설정하게 되면 0 바이트 크기의 가상 디스크가 생성되며, 생성된 가상 디스크가 나중에 제거되지 않는 문제가 발생하였다. 따라서, 본 논문에서는 이러한 문제에 대처할 수 있도록 사용자가 실제 디스크의 여유 공간을 초과하는 가상 디스크를 생성하고자 할 때 가상 디스크의 크기를 재입력하도록 구현하였다. [그림 8]은 가상 디스크의 생성 가능 여부를 검사하기 위해 실제 디스크의 여유 공간을 확인하는 부분이다.

```
GetDiskFreeSpaceEx( Storage1,&Storage2,&Storage3,&Storage4);

Low = Storage2.LowPart;
Upper = Storage2.HighPart;
Storage5 = (Upper * 2);
Storage5 = pow(Storage5, 32);
Storage6 = Storage5 + Low;
...
```

[그림 8] 실제 디스크의 여유 공간 검사 코드

한편, 생성된 가상 디스크를 제거할 때는 ②번의 "삭제" 항목을 선택하고, ③번의 "디스크 명"을 지정한 후 "시작" 버튼을 누른다.

4. 결론 및 향후 연구과제

본 논문에서는 컴퓨터 사용에 익숙치 않은 초보자나 하드 디스크의 파일 관리에 어려움을 겪는 사용자에게 편리한 파일 관리 기능을 제공할 수 있는 가상 디스크 프로그램을 설계하고 구현하였다.

하나의 컴퓨터를 여러 사람이 공동으로 사용하는 경우에 사용자들은 대개 자신의 폴더를 생성한 후 그곳에 파일들을 저장한다. 그런데, 컴퓨터 활용에 미숙하거나 파일 관리에 익숙치 않은 사용자의 경우 동일한 파티션 내에 위치한 본인 폴더뿐만 아니라 타인의 폴더에도 파일을 생성하거나 저장하여 파일 관리에 지장을 주게된다.

그러나, 본 논문에서 구현한 가상 디스크 프로그램을 활용하여 별도의 가상 디스크를 생성한다면 디스크 파티션을 변경하지 않더라도 사용자 입장에서는 개인별로 별도의 디스크 파티션이 부여된 것처럼 느끼기 때문에 파일 관리에 많은 도움이 될 것이다.

한편, 현재 통용되는 가상 CD-ROM 프로그램들은 다수의 가상 CD-ROM 을 생성할 수 있으나, 본 논문에서 구현한 가상 디스크 프로그램은 하나의 가상 디스크가 생성되어 Mount 되어 있을 때에는 새로운 가상 디스크를 생성할 수가 없다. 또한, 가상 디스크를 계속 사용하려면 생성된 가상 디스크가 계속 유지되어야 하는데, 현재 구현된 가상 디스크 프로그램에서는 시스템 종료시 가상 디스크가 제거되는 단점이 있다. 따라서, 앞으로 이러한 문제점들을 보완하고, 가상 디스크 이미지 파일을 암호화하여 본인 외 타인의 접근을 막는 암호화된 가상 디스크 생성 기능을 추가시켜 가상 디스크 프로그램의 기능을 향상시키고 안정성도 높일 예정이다.

참고문헌

[1] David A. Solomon et al., Inside Windows 2000 3rd Ed., 정보문화사, 2001

[2] Art Baker Jerry Lozano, The Windows 2000 Device Driver Book 2nd Ed., Prentice-Hall Computer Books, 2001

[3] Walter Oney, Programming the Windows Driver Model, 정보문화사, 2001

[4] Chris Cant, Writing Windows WDM Device Drivers, 에이콘출판사, 2000

[5] 하제소프트, Windows 2000/NT Device Driver, 2002

[6] 영진소프트테크, Basic Technics for Developing Windows Driver Model, 2002

[7] Andrew S. Tanenbaum, MODERN OPERATING SYSTEMS 2nd Ed., 사이텍미디어, 2002

[8] Windows System Programmer, http://cafe.daum.net/programmer

[9] 김상욱, 윈도우 NT Windows 4.0, 21 세기사, 2000

[10] Windows 2000 DDK Documentation