

PC 운영체제 추세 및 발전방향

洪 性 秀

서울大學校 電氣工學部

I. 서 론

80년대 초반 개발되어 소수의 사용자들에게 이용되던 PC가 90년대에 들어 급격히 보급되면서 사회 전반에 걸쳐 큰 영향을 미치고 있다. 이제는 “컴맹”이라는 단어가 일반인 사이에 회자되고, 컴퓨터를 모르는 것이 문맹의 새로운 형태가 될 정도로 컴퓨터에 대한 관심은 급격히 증가하고 있다. 이에 정보가 중요한 무형 자산으로 인식되면서 기업활동에서 뿐만 아니라 개인의 일상 생활에서도 정보 통신의 비중이 크게 증대되고 있다. 더구나 컴퓨터 하드웨어 기술의 발전과 Web과 Netscape으로 대표되는 인터넷에 힘입어 사회의 정보화는 더욱 더 가속화되고 있고, 네트워크 컴퓨터(NC)와 멀티미디어 컴퓨터(MC)와 같이 새로운 개념의 컴퓨터와 서비스들이 속속 출현하고 있다. 하지만 이러한 환경의 변화에도 불구하고 MS-DOS와 Windows로 대표되던 기존의 PC용 운영체제는 단순한 성능 개선 이외에 뚜렷한 변신을 하지 못하는 한계를 보이고 있다.

운영체제는 사용자와 컴퓨터를 연결해주는 역할을 하기 때문에 인간이 컴퓨터를 이용하는데 있어서 운영체제의 역할은 아주 중요하다. 그래서 높은 성능의 하드웨어가 있다고 해도 운영체제의 성능이 발휘되지 못하면 사용자는 큰 만족을 얻을 수 없게 된다. 현재의 운영체제의 설계 개념은 하드웨어의 발전과 주변 환경의 변화를 가로막는 한 요인으로 작용하고 있다. 이러한 환경의 변화를 주도하고 있는 것은 초고속 통신망을 기반으로 한 인터넷과 멀티미디어 환경의 대두로 요약할 수 있다. 이제 운영체제도 기존의 낡은 개념에서 벗어나 새로운 패러다임으로 옷으로 갈아입을 때가 된 것이다.

이러한 요구에 맞추어 본고에서는 차세대 PC용 운영체제의 새로운 패러다임과 그에 해당하는 현재의 기술 동향에 대하여 살펴보기로 한다.

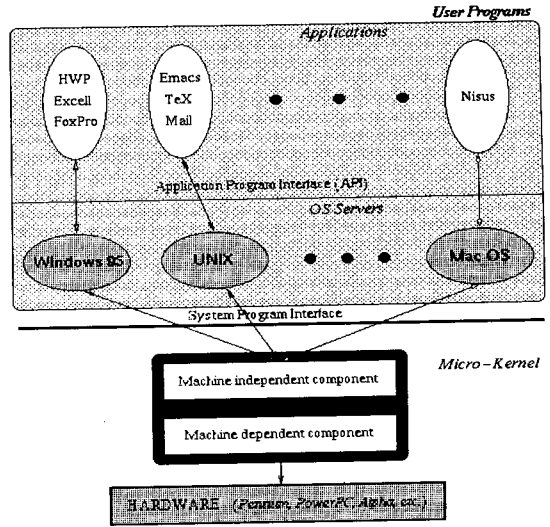
II. 마이크로커널

1986년 Micro-Kernel 개념을 이용한 Mach 운영체제의 출현은 운영체제의 설계와 구성에서 큰 변화를 가져왔다. 현재까지의 가장 일반적인 운영체제의 구조는 통합(integrated) 또는 단일(monolithic) 커널 구조였다. 기존의 Unix, Windows, OS/2 등과 같이 사용자가 일반적으로 접하고 있는 대부분의 운영체제가 여기에 해당한다. 이런 운영체제 하에서는 응용 프로그램에 대한 시스템 서비스와 하드웨어에 대한 제어 기능들이 모두 하나의 주소 공간에서 수행된다. 이에 대응하는 구조로서 마이크로커널 구조가 있다. 여기서는 기존 운영체제의 커널에서 가장 기본이 되는 부분들만이 분리되어 작은 커널을 형성하고, 나머지 부분은 사용자 모드에서 수행하도록 한다.

마이크로커널의 기본 개념은 두 가지의 주요 아이디어에 기초하고 있다. 첫째는 아주 최소 부분을 제외한 모든 코드가 하드웨어에 무관하게 작성되어야 한다는 것이며, 둘째는 모든 운영체제에 공통적인 하위 기능들은 운영체제의 특징을 보여주는 상위 요소들로부터 완전히 분리되어야 한다는 것이다. 이러한 두 가지를 실현하기 위해서는 운영체제가 고도로 모듈화되어야 한다.

마이크로커널에 포함되는 기능들의 범위는 구현에 따라 약간씩 차이가 있지만 공통적인 기능들은 '태스크 및 쓰레드 관리'의 하위 부분, 프로세스간 통신(Interprocess Communication : IPC) 및 동기화, 메모리 관리의 하위 부분, 최소의 디바이스 관리, 시스템 운영 중 발생하는 각종 인터럽트 처리 등이다. 파일 서비스, 표준 Unix 인터페이스 등과 같은 운영체제 서비스의 상위 부분들은 사용자 수준의 서버 프로그램으로 구현되며, 하나 이상의

서버가 운영체제에 포함된다. 서버들은 주소공간 사이에 걸치는 IPC를 사용하여 상호 동작함으로써



(그림 1) 마이크로커널을 기반으로 한 운영체제의 구조.

기존의 운영체제 서비스를 실현한다. 그림 1은 전형적인 마이크로커널 구조의 운영체제를 보이고 있다.

마이크로커널을 기반으로 한 운영체제의 가장 중요한 특징은 모듈화로서 시스템 구성요소를 서로 분리하고, 요소들간 통신 구조로서 메시지 전달 방식을 이용한다. 여기서 마이크로커널은 응용 또는 시스템 서버들과 하드웨어 사이에 교통 경찰관의 역할을 수행한다. 즉, 메시지들을 검증하여 요소들간에 전달하며 하드웨어에 대한 접근을 수락한다. 이러한 메시지 전달 방식의 통신 기법은 본질적으로 분산처리, 다중처리, 클러스터링, 대규모 병렬 처리 구조에 아주 적합하기 때문에, 마이크로커널 구조는 기존의 통합 커널에 비하여 효율적인 분산 컴퓨팅 환경 지원한다. 그리고 전체 시스템의 모듈화는 우수한 확장성과 높은 이식성을 제공하는

- 1) 이전과 달리 마이크로커널에서의 프로그램은 기존의 프로세스와 달리, 하나의 주소 공간을 공유하는 여러 개의 쓰레드로 구성된다.
- 2) 운영체제의 크기는 마이크로커널의 크기가 아니라 통합커널에 버금가도록 모든 기능들을 합하였을 때의 크기를 말한다.

다. 하지만 시스템 콜 인터페이스나 운영체제의 서비스 호출이 IPC에 의존하기 때문에 운영체제 성능이 기존의 통합커널에 비하여 다소 감소한다는 측면과 운영체제 크기가 증가한다는 단점이 있다.²⁾

현재 상용화되어 있는 PC용 운영체제들 중 마이크로커널을 이용하는 것으로는 IBM의 Workplace OS, NeXT의 NextStep, QNX Software의 QNX, Be의 Be OS 등이 있다. 또한, 현재 개발되고 있는 네트워크 컴퓨터(NC)도 마이크로커널을 기본으로 하고 있다.

여기서 간단하게 최근 Be, Inc.에서 개발된 Be Operating System(Be OS)³⁾의 구조에 대하여 살펴보기로 하자. 이것은 DOS와 Windows로 대표되는 기존의 PC 운영체제의 틀을 완전히 버린 새로운 개념의 운영체제로 BeBox라는 새로운 PC를 기반으로 한다. Be OS는 마이크로커널, 서버, 도구상자(kit)의 3계층으로 구성된다. Be OS의 마이크로커널은 Windows NT의 HAL(Hardware Abstraction Layer)과 비슷하게 하드웨어와 관련된 부분을 처리하며, 커널 내에 플로피와 하드디스크 드라이버만을 내장하고 다른 디바이스 드라이버들(Ethernet, 오디오, 비디오, 키보드, 마우스 등)은 실행 시에 동적으로 커널에 로드함으로써 작은 크기를 유지한다. Be OS에서는 다수의 서버가 존재하는 클라이언트/서버 모델을 따른다. 즉, 응용의 특성에 따라 서로 다른 서버를 이용할 수 있다. 기본 서버로 응용 서버(application server)와 저장 서버(storage server)가 있으며, 응용의 필요에 따라 새로운 서버가 추가될 수 있다. 저장 서버는 영구적인 데이터에 대한 접근을 관리하며 파일 시스템과 데이터베이스를 관리한다. 응용 서버는 대부분의 하위 레벨의 사용자 인터페이스(키보드와 마우스를 통한 사용자와 응용 프로그램 윈도우의 인터페이스)를 담당한다. 이 서버들이 기존의 통합커널 운영체제에서의 상위레벨의 작업들을 수행한다.

마이크로커널을 기반으로 한 운영체제는 점차

차세대 운영체제의 구조화 기법으로 자리잡게 될 것이라고 기대된다. University of Washington의 Brian Bershad 교수는 "Windows NT를 사용하기 위해 최소 16MB의 메모리가 필요하지만, 실제로 사용되지 않는 운영체제의 기능들이 컴퓨터의 프로세싱 능력의 많은 부분을 차지한다."⁴⁾라고 말한다. 이것은 마이크로커널 방식의 구조화에 시사하는 바가 크다고 할 수 있다. 실제로 현재 운영체제 오버헤드의 30% 정도가 메모리 관리와 네트워크 처리 때문에 발생하기 때문이다. 그러므로, 마이크로커널 기술을 통한 운영체제의 모듈화로 이런 문제점을 개선해 나갈 수 있다.

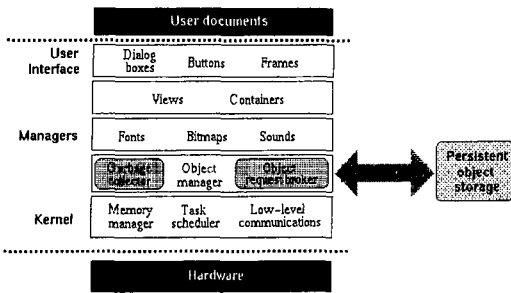
III. 객체 지향 운영체제

최근 운영체제를 포함한 많은 소프트웨어들의 기능이 다양해지고 복잡해지면서 기존의 시스템 개발 방식에 한계가 발생하였다. 이에 전체 시스템을 객체로 분할을 하여 설계하는 객체 지향 개념이 대두되었다. 운영체제는 응용 프로그램들이 시스템의 자원을 이용하기 위한 다리 역할을 하기 때문에, 운영체제에서의 객체 모델의 도입은 전체 시스템 설계가 완전한 객체 모델을 따를 수 있도록 한다는 면에서 큰 의미를 갖는다. 즉, 전체 시스템 설계와 관리에서 객체 지향 모델의 장점들인 빠른 이식성, 재사용 가능성, 내부 구성의 캡슐화 등을 완벽하게 얻을 수 있게 된다. 이에 따라 객체 지향 운영체제 구성 방법에 대한 많은 연구들이 진행되고 있다. 본 절에서는 이러한 객체 지향 운영체제의 형태와 현재의 기술 동향에 대하여 간략하게 살펴보기로 한다.

가장 순수한 형태의 객체 지향 운영체제(Object Oriented Operating System : OOOS)는 컴퓨터 하드웨어 인터페이스를 담당하는 작은 마이크로커널과 재사용 가능한 객체들로 구성된다. 이 때, 응

3) Be-All of Operating Systems의 약자. <http://www.be.com>와 96년 BYTE 5월호 참조.

4) BYTE 1995년 11월호 참조



(그림 2) 객체 지향 운영체제를 기반으로 한 시스템의 한 예

용 프로그램과의 상호 인터페이스를 담당하는 운영체제의 모든 부분들이 객체로 구현된다. 따라서 상위레벨을 구현하는 운영체제와 응용 프로그램 사이에 아무런 차이가 존재하지 않게 된다. 그림 2는 이러한 OOOS의 구조를 보여준다.

객체는 단순히 주어진 데이터 타입에 어떤 동작을 수행하는 코드와 해당 데이터 타입에 속하는 데이터의 묶음이다. 객체들이 서로 메시지를 전송할 때, 커널이 해당 코드를 수행하게 함으로써 프로그램의 수행이 이루어진다. 즉, 모든 것이 객체들간의 데이터의 흐름에 의하여 이루어진다. 이러한 OOOS는 앞에서 설명한 바와 같이 마이크로커널을 그 기반으로 하며, 마이크로커널은 메모리 할당과 태스크 스케줄링과 같은 하드웨어에 의존적인 일들만을 처리하며, 사용되지 않는 객체들로 인하여 메모리가 낭비되는 것을 방지하기 위해 자동적으로 쓰레기 모음(garbage collecting) 작업을 수행한다. 객체 관리자는 커널 서비스를 이용하여 동적으로 일시적인 객체를 만들거나 제거하며, 사용자 문서 객체와 같이 앞으로 필요한 것들을 유지하는 작업을 한다.

이와 같은 순수한 개념의 OOOS는 현재까지 PDA(Personal Digital Assistant)와 같은 시스템에서만 적용되어 왔다. 이것은 PDA가 완전히 새로운 시스템으로 기존의 것과의 호환을 전혀 고려할 필요가 없기 때문이다. 하지만 이와는 달리 PC에서는 순수한 OOOS가 구현되기 힘들다고 많은 사람들은 생각하고 있다. 이것은 PC용 운영체제는 이전에 존재하던 것들과의 호환성을 고려해야만

하기 때문이다. 그래서 현재 객체 지향 개념을 차용한 대부분의 PC 운영체제들은 기존의 일반적인 파일 시스템 위에 객체 개념을 얹어 놓은 과도기적인 시스템의 형태를 보이고 있다. 예를 들어, NextStep은 Mach 마이크로커널을 기반으로 하고 Unix 파일 시스템을 채용하면서, Objective-C로 작성된 객체들로 상위 구조를 구성한다. 이와는 달리 Be OS는 기존의 접근 방법을 완전히 탈피하면서 객체 개념을 도입하였다. Be OS의 파일 시스템은 기존과는 완전히 다른 형태로 관계 데이터베이스(relational database)로 구성되며, 데이터는 시스템이나 응용 프로그램에서 지정된 형태로 저장된다. 이러한 파일 시스템을 근간으로 커널, 서버, 디바이스 드라이버와 상호 작용을 하는 객체 지향 개념의 소프트웨어 키트가 존재한다. 이것에는 User Interface Kit, Multimedia Kit, MIDI Kit, Networking Kit, Database Kit 등이 포함되며, 객체 개념을 이용하여 다양한 응용을 개발할 수 있도록 한다. 이렇게 현재의 PC용 운영체제 구조에서는 이 시스템이 OOOS의 기본 구조를 가장 충실히 따르고 있다.

PC에서 완전한 OOOS가 언제 출현할 지는 예측하기 힘들지만 객체 지향 개념으로 운영체제를 제작해야한다는 것은 이론의 여지가 없다. 실제로 이미 많은 운영체제 제작사들이 서로 연합하여 OOOS를 개발하고 있다. 이것은 객체 개념을 이용하면 운영체제와 같이 복잡한 시스템을 효율적으로 개발할 수 있고, 개발된 시스템을 쉽게 유지 관리할 수 있기 때문이다.

IV. 멀티미디어 컴퓨팅

“컴퓨터로의 모든 기능 통합”이라는 이름 하에 대두되는 멀티미디어 컴퓨팅은 현재의 컴퓨터 기술을 이끌어가고 있는 중심 분야라고 할 수 있다. 이전의 텍스트 위주의 작업 환경에서 동화상과 음향을 포함한 멀티미디어 기술은 원격 교육 시스템과 같은 교육적 측면과 맞물리면서 빠른 속도로

발전하고 있다. 이러한 멀티미디어 응용 프로그램의 특성은 기존의 응용 프로그램들과 많은 차이를 보이기 때문에, 기존의 운영체제들은 새롭게 대두되는 멀티미디어 응용 프로그램들을 적절히 지원하지 못하는 취약점을 보이고 있다. 멀티미디어 응용 작업의 특징을 살펴보면 다음과 같이 네 가지로 요약할 수 있다.

- 실시간 처리 특성 : 동화상을 재생할 때, 각 프레임에 늦어도 1/30초 내에 처리해야 하며, 또한 여러 작업들을 서로 정해진 시간 순서대로 처리해야 한다.
- 서비스 품질(Quality of Service)의 절충 가능성 : 멀티미디어 처리 작업의 경우에는 시스템의 상태에 따라 데이터의 양을 조절할 수가 있으며, 이 데이터의 양은 서비스의 품질을 결정하게 된다. 멀티미디어 데이터의 소비자와 제공자에 해당하는 응용 프로그램들이 서비스 품질에 대해 상호 절충함으로써 보다 효율적인 멀티미디어 처리작업이 이루어질 수 있도록 할 수 있다.
- 대량의 데이터에 대한 단순한 입출력 : 동영상과 같은 미디어의 데이터 크기는 일반적인 텍스트에 비하여 상당히 크며, 일반 데이터와는 달리 데이터의 압축과 재생을 제외하고는 다른 처리 없이 입력장치로부터 출력장치로 직접 전달된다. 즉, CPU의 개입이 필요 없는 경우가 대부분이다.
- 비 멀티미디어 작업과의 공존 : 워드프로세싱이나 스포레드쉬트와 같은 기존의 응용 프로그램들도 멀티미디어 응용 프로그램들과 함께 사용된다.

현재의 운영체제는 기본적으로 단일 미디어를 전제로 하여 설계되어 있기 때문에, 멀티미디어 응용을 지원하는데는 큰 한계가 존재한다. 이러한 멀티미디어 서비스의 특성을 만족시키기 위해서는 우선 첫째로, 기존의 실시간 시스템 분야에서 연구되고 있는 실시간 스케줄링의 개념이 프로세스 스케줄링에 적용되어야 한다. 두번째로는 운영체제

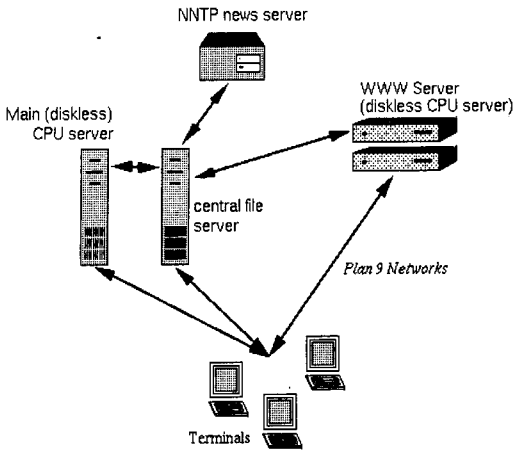
내에 QOS 관리자를 두고 시스템의 부하에 따라 적절하게 서비스의 품질을 응용 프로그램과 절충하는 방법이 제공되어야 한다. 세번째로 불필요한 데이터의 복사를 막기 위하여 입력장치에서 직접 출력장치로 데이터가 흘러갈 수 있도록 하는 방법이 포함되어야 한다. 마지막으로 운영체제가 여러 종류의 서비스 정책을 지원해야 한다. 이것은 한 시스템에 멀티미디어와 비멀티미디어 응용들이 동시에 존재하기 때문에 어느 하나를 위한 시스템을 구성할 수 없기 때문이다. 그러므로 마이크로커널 구조를 이용하여 응용의 특성에 맞는 다양한 정책들을 지원해야 한다.

이러한 멀티미디어의 특성을 완벽히 지원하는 범용 PC용 운영체제는 아직 존재하지 않으며, 멀티미디어 응용 프로그램들이 이런 작업들을 자체적으로 처리하고 있다. 또한, 멀티미디어를 위한 미들웨어(middleware)를 중심으로 많은 개발이 이루어지고 있다. 한편, 대학의 연구실을 중심으로 실시간 기능과 멀티미디어 기능을 갖춘 운영체제의 개발이 진행중이며 Real-Time Mach에 멀티미디어 서비스를 지원하기 위한 기능들을 첨가하는 작업들이 이루어졌다.

현실로 나타나고 있는 “컴퓨터로의 모든 기능의 통합”이 어느 수준에 이를지는 예측하기 힘들지만, 멀티미디어가 인간 생활에 미치는 영향을 생각해 본다면 인간과 많은 연관을 맺고 있는 PC에서의 양질의 멀티미디어 서비스 지원은 필수적이라고 할 수 있다.

V. 분산 컴퓨팅 운영체제

미국의 클린턴 행정부가 조인했던 기가비트 네트워크와 같은 초고속 통신망이 일반 사용자들이 사용할 수 있을 정도로 보편화된다면 “과연 내가 사용하고 있는 컴퓨터에서 모든 기능들을 직접 수행할 필요가 있는가” 하는 의문이 생기게 된다. 네트워크를 통해 컴퓨터들이 연결되어 있는 환경에서는 멀리 존재하는 컴퓨터에서 프로그램을 수행



〈그림 3〉 MPlan 9으로 구성된 분산 컴퓨팅 환경의 한 예

시키거나, 프로그램을 가져와 자신의 컴퓨터에서 수행시킬 수 있기 때문이다. 이러한 생각을 실천에 옮기는 작업이 80년대 말부터 현재까지 진행 중에 있으며, AT&T의 Plan 9, SunSoft의 Spring과 같은 실험적인 운영체제들이 그 중간 결과라고 할 수 있다. 현재까지의 이러한 분산 컴퓨팅 환경은 많은 메모리와 빠른 네트워크 성능을 요구하고 있기 때문에 개인용 운영체제에서 각광을 받고 있지 못하고 있는 실정이다. 하지만 이미 이런 제약점들을 극복하기 위해 많은 연구가 진행되고 있다. 또한 값싼 메모리와 빠른 네트워크 환경의 구축이 눈앞에 다가와 있는 현실을 볼 때, 분산 환경에서의 컴퓨터의 이용은 빠른 속도로 우리에게 다가올 것으로 생각된다. 더구나 현재 Oracle과 함께 많은 업체들이 추진하고 있는 NC(Network Computer)와 SunSoft의 네트워크 언어 자바(Java)는 이러한 분산 컴퓨팅 환경에 추진력을 불어넣고 있다.⁵⁾

본 절에서는 현재 개발되어 있는 분산 운영체제인 Plan 9에 대하여 간략하게 살펴보고 분산 운영체제의 전망에 대하여 살펴보기로 한다.

Plan 9은 1987년에 AT&T의 Computing Science Research Center에서 시작한 프로젝트로 기

존의 Unix 시스템에서 가장 부족한 기능인 네트워킹을 보강하여 운영체제를 만드는 것을 목표로 시작되었다. Plan 9은 분산 운영체제로서 그림 3과 같이 사용자 터미널, 데이터를 저장하는 파일서버, 응용 프로그램들을 수행시키기 위한 빠른 CPU를 제공하고, 사용자를 검증(authentication)하고, 네트워크 게이트웨이를 제공하는 기타 서버들을 지원한다. 이들 구성 요소들은 기존의 LAN과 WAN 등으로 연결되어 있으며, 사용자는 터미널이나 CPU 서버에서 수행되는 응용 프로그램과 상호작용을 하고, 응용 프로그램을 통해 파일서버로부터 데이터를 얻어온다. 이러한 분산 운영체제에서는 자신의 컴퓨터에 존재하는 파일과 서버에 존재하는 파일간에 차이가 없기 때문에 (물리적으로 떨어져있다는 것 이외에) 프로그램이 어느 곳, 어떤 컴퓨터에서 수행되는가에 관계없이 응용 프로그램을 사용할 수가 있다. 응용 프로그램과 데이터들이 여러 곳에 분산되어 존재하기 때문에 보안 문제가 Plan 9에서의 가장 중요한 부분이 된다. Plan 9에서는 Kerberos와 비슷한 방법을 사용하여 이를 처리하고 있다. Plan 9에서는 패스워드가 네트워크를 통해 절대로 전달되지 않도록 하며, 대신에 응용 프로그램은 사용자 검증서버(authentication server)가 발행하는 암호화된 표(ticket)를 이용한다. 또한 Plan 9은 데이터가 저장되는 파일 서버에서는 절대로 응용 프로그램이 수행될 수 없다는 제약을 가한다.

Plan 9 시스템은 상업적이기 보다는 분산처리 운영체제의 가능성을 시험하는 시스템이기 때문에 일반 사용자가 이용할 수 있는 시스템이 되기까지는 많은 시간이 걸릴 것으로 예상된다. 반면, 현재 노벨을 중심으로 한 네트워크 컴퓨터(NC)와 Microsoft를 중심으로 한 멀티미디어 컴퓨터(MC)⁶⁾는 기존의 분산 컴퓨터의 한 극단적인 형태이기 는 하지만 Plan 9과는 달리 상업성을 전제로 하고 있기 때문에 일반 사용자가 시스템을 접하기까지 많은 시간이 걸리지 않을 것이다. WWW의 빠른

5) BYTE 1996년 3월호 참조

6) NC에는 디스크가 없는 반면, MC에는 디스크가 존재한다.

보급과 SunSoft의 자바 언어와 초고속 통신망의 등장으로 인하여 이들의 현실성이 더욱 증가되고 있다.

한편 분산 컴퓨팅은 네트워크의 속도에 크게 종속되기 때문에, 분산처리 컴퓨터의 미래가 네트워크 기술에 달려 있다고 해도 과언이 아닐 것이다. 아울러 분산 환경에서 PC의 사용이 갖는 많은 장점들 때문에 분산환경에서의 PC용 운영체제의 개발은 활기를 띠고 있다.

VI. 보안을 기본으로 하는 운영체제

현재까지 컴퓨터에서의 보안은 대형 컴퓨터나 워크스테이션과 같이 여러 명의 사용자가 하나의 컴퓨터를 공유하는 시스템에서 자신의 데이터를 지키는 것으로 인식되어 왔으며, PC와 같이 개인용 시스템에서는 바이러스와 같은 위협으로부터의 보호만을 의미했다. 점차 PC들이 네트워크에 연결되어 사용되고 다수의 사용자와 컴퓨터 자원을 공유하게 됨에 따라 PC에서도 좀 더 넓은 의미의 보안이 적용되어야 할 필요성이 대두하게 되었다. 특히 네트워크를 통하여 물리적인 접근 없이도 통신망에 존재하는 소프트웨어 자원들을 접근할 수 있기 때문에, 즉, 물리적으로 자원이 저장되어 있는 매체로부터 멀리 떨어져 있는 곳에서도 통신망을 통해 불법복사, 내용변조, 내용파괴 등의 행위가 가능하기 때문에, 앞으로는 개인용 컴퓨터도 이러한 행위들을 막을 수 있는 기능들을 반드시 포함하여야 한다. 이를 위해 운영체제에서는 시스템 자체에서의 보안과 네트워크에서의 보안 두 가지 모두를 고려해야만 한다.

시스템 자체에서의 보안은 기존의 Unix 시스템과 같이 여러 명의 사용자가 하나의 시스템을 공유할 때 발생하는 보안의 문제를 다루게 된다. 예를

들어, 특정 개인의 파일이나 공용의 프로그램, 사용자의 패스워드 등을 다른 사용자들로부터 보호한다거나 악의를 가진 응용 프로그램으로부터 시스템을 보호하는 것 등이다. 한 예로, Windows NT에서는 사용자가 컴퓨터를 사용하기 위해 반드시 로그인을 해야 하며, 접근허가⁷⁾가 된 파일만을 접근할 수 있도록 하고 있다. 그리고 시스템의 사용과정을 기록하는 감사(audit) 기능을 두어 문제의 원인을 추적할 수 있도록 하며, 직무에 기초한(role-based) 접근 방법을 제공한다.

이와는 달리, 네트워크에서의 보안은 네트워크를 통해 전송되는 자료들의 변조 위험 등에 대처하는 문제를 다룬다. 예를 들면, 네트워크 컴퓨터의 가장 기본이 되는 자바⁸⁾의 보안기능은 인상적이다. 자바에서는 수험코드 모듈(applet)들이 네트워크를 통해 전송된 후 원격 컴퓨터에서 수행되기 때문에 애플릿을 통한 해킹을 방지할 수 있도록 파일에 대한 접근과 처리(이름 변경, 삭제, 생성 등)에 많은 제약들을 가한다. 또한, 네트워크를 통해 전송되는 자바 바이트 코드가 변조되는 것을 방지하기 위해 자바 바이트 코드를 수행하기 전에 검증한다. 이 때 자바 가상 기계 코드의 유효성 여부, 스택의 과사용 여부, 레지스터의 부적절한 사용 여부, 데이터 타입의 불법적인 변경 여부 등이 검사된다. 이러한 검사는 로드된 바이트 코드가 시스템에서 악의적으로 오동작을 할 수 없도록 하기 위한 것이다.

PC 환경에서의 보안을 증진시키는 문제는 컴퓨터의 사용에 있어서 이전에 사용자들이 느끼지 못한 불편함과 같은 새로운 문제를 야기한다. 그러므로, 운영체제에는 이러한 불편을 최소로 줄이면서 가능한 최대로 안전한 상태를 유지할 수 있도록 하는 기능들이 계속해서 추가되어야 한다. 사회의 정보화가 가속화되고 있는 현실에서 개인의 자료가 타인에게 공개되고 변조되기를 원하는 사용자는 아무도 없을 것이다. 따라서 보안의 문제는 지속적

7) Windows NT는 Unix에서와 같이 읽기, 쓰기, 실행허가를 모든 화일에 각각 설정할 수 있으며 NT는 "take ownership"과 "change permission"을 추가로 두고 있다. 또한, 접근-제어 리스트를 이용하여 화일 접근에 대한 좀 더 세세한 제어를 할 수 있도록 한다. 접근-제어 리스트는 화일뿐만 아니라 NT에서 관리하는 모든 객체(object)들에 대해 설정할 수 있다. (1996년 BYTE 5월호 참조)

8) 본고의 자바에 대한 기사 참조

인 관심과 연구의 대상이 되어야 하며 기능과 성능의 개선 이전에 먼저 고려되어야 할 사항이다.

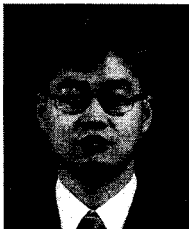
VII. 결 론

지금까지 현재 PC용 운영체제의 개발 기술과 앞으로의 기술 동향에 대하여 간략하게 살펴보았다. 위에서 제시한 기술들은 PC용 운영체제뿐만 아니라 대부분의 운영체제가 가져야할 기능들이다. 이 모든 것들은 값싸면서도 성능 좋고 안전한 컴퓨터를 사용자들이 사용할 수 있도록 해야 한다는 하나의 일관된 목표에 기초를 두고 있다. 하지만 이러한 기능들이 실제로 우리가 사용하는 PC의 운영체제에 포함되기까지는 많은 시간이 걸릴 것이다. 그 시간의 길고 짧음은 운영체제를 제작하는

업체와 개발자들의 생각여하에 따라 많이 차이가 날 것이다. SunSoft의 Jim Mitchell은 “대부분의 프로젝트가 Unix의 성능을 향상시키는 관점에서 진행되어 왔다. 아무도 근본적으로 다른 시스템을 개발할 때 오는 위험을 안으려 하지 않고 있다.”⁹⁾고 말한다. 이것은 왜 현재까지 빠른 컴퓨터 하드웨어 성능 향상에 비해 운영체제의 성능이 상대적으로 느린 향상을 보이고 있는가를 잘 설명해 주고 있다. 이것은 현재의 운영체제 제작 기술이 기존의 운영체제와의 연계에 너무나 큰 집착을 보이고 있기 때문이다.

멀티미디어, 초고속 통신망, 빠른 하드웨어의 성능 향상과 같은 주변 환경의 변화에 의해 이러한 생각에 많은 변화의 조짐이 보이고 있는 현 시점에서 지금까지 설명한 객체 지향 개념이나 분산 컴퓨팅과 같은 개념의 도입은 앞으로의 운영체제 기술 발전에 큰 힘이 될 수 있을 것으로 기대된다.

저 자 소 개



洪 性 秀

1963年 10月 11日生

1986年 2月 서울대학교 공과대학 컴퓨터 공학과(학사)

1988年 2月 서울대학교 공과대학 컴퓨터 공학과(석사)

1994年 12月 University of Maryland at College Park, Dept. of Computer Science(박사, Ph.D.)

1988年 2月~1989年 7月 연구원 한국전자통신연구소

1994年 12月~1995年 3月 Faculty Research Associate(Univ. of Maryland)

1995年 4月~1995年 8월 Member of Techical Staff Silicon Graphics Inc.

1995年 9月~현재 전임강사, 서울대학교 공과대학 전기공학부

주관심 분야 : Real-Time Systems, Operating Systems, Distributed Control Systems, Software Engineering

9) BYTE 1995년 11월호 참조