特集

개방형 운영체제의 기술 및 표준화 동향

安 聖 辰, 金 精 玟, 金 東 圭
現代電子産業(株) 소프트웨어 硏究所

Ⅰ. 서언

운영체제(Operating System)는 컴퓨터 소프트웨어 중 가장 핵심이 되는 기술로서 이 기술없이 국산 컴퓨터가 국제적 경쟁력을 갖는다는 것은 어려운 일이다. 이러한 핵심 기술인 운영체제라는 말 앞에 요즈음은 개방형이라는 수식어가 늘 붙어다니는 것을 볼 수 있다. 개방형 운영체제를 논하는데 있어서 우선 개방형 시스템(Open System)의 성격에 대한 정의를 내리는 것이 순서인 것 같아서 필자 나름대로의 정의를 다음과 같이 규정하고자 한다.

첫째, 개방(Open)되어 있어야 한다.

어떤 한 회사나 개인 또는 단체가 소유권을 가지고 독점해서는 안된다.

둘째, 이식성(Portability)이 있어야 한다.

여러 다양한 기종의 하드웨어에 이식될 수 있어야 한다.

세째, 상호운용성(Interoperability)이 있어야 한다.

다른 컴퓨터 제조회사의 컴퓨터와 같이 운용할 때 불편이 없어야 한다.

물론 이 외에 몇 가지 더 언급할 수도 있겠지만 '개방형'의 정의는 위와 같은 세 가지 성격의 내포로 규정하면 되리라 생각한다.

개방형 운영체제와 UNIX를 동의어로 사용하는 경우가 있는데 그 이유는 UNIX만큼 위의 정의를 만족시키는 운영체제가 없기 때문이다. 그래서 본 글에서는 UNIX를 중심으로 하여 90년대 중반 컴퓨터 업계에 영향을 미칠만한 운영체제와 관련기술에 대해 언급하기로 하겠다. 또한 근래에 들어 Microsoft에서 발표한 Windows NT에 대한 관심이 급증하고 있는 바 비록 Windows NT가 개방되어있지는 않지만 향후 운영체제 발전 방향에 많은 영향을 미칠 것이므로 Windows NT의 주요 특징과 기본 개념에 대해 언급하기로 하겠다.

Ⅱ. UNIX 표준화

1. UI 대 OSF

80년대 후반 다양한 UNIX의 변종이 시장에 소개되고 사용되면서 UNIX라는 이름을 달고는 있지만 서로 상호운용이 되지 않거나, 응용 프로그램의 이식이 되지 않는 경우가 생기게 되었다. 이에 대한 해결책으로 UNIX의 소유권자인 AT&T가 워크스테이션 시장의 선두주자인 SUN과 함께 기존의 모든 UNIX 시스템의 장점을 합한 UNIX를 만들기 위해 UI (Unix International)라는 연합체를 만들었다. UI에서는 90년 말에 SVR 4.0을 소개하게 된다. 이러한 AT&T의 UNIX 독점에 대항하기 위해 IBM, DEC, HP등이 OSF(Open Software Foundation)를 결성하여 Carnegie-Mellon의 Mach 커널에 근거한 OSF/1을 만들게 된다. SVR 4.0과 OSF/1과의 경쟁은 근래에 들어 OSF의 붕괴로 인해 자연스럽게 SVR 4.0의 승리로 결론지어졌다.

2. ANSI C

UNIX는 원천 코드의 90% 이상이 C 언어로 되어 있어서 C의 표준화는 UNIX에 적지 않은 영향을 미

친다. 1989년 ANSI에서는 X3.159-1989라는 C 표준을 발표하고, 이는 국제 표준기구인 ISO에 의해 채택되어 ISO/IEC 9899가 되었다. C 언어의 표준은 주로 ISV(Independent Software Vendor)에게 중요한 영향을 미친다. ISV가 다양한 플랫폼에 응용 프로그램을 빨리 이식한다는 것은 곧 사용자에게도 더 좋은 기능의 프로그램을 적시에 제공한다는 것이므로 결과적으로는 사용자 편리성에 기여한다고 볼 수 있다.

3. IEEE POSIX

POSIX(Portable Operating System Interface based on UNIX)는 IEEE(Institute of Electrical and Electronics Engineers)에 의해 정해지는 표준의 집합을 말한다. POSIX는 1003 작업 그룹(Working Group) 내의 10 여개 Sub-Committee에서 해당 표준안을 만드는데 그중 가장 중요한 것이 운영체제 인터페이스를 맡은 P1003.1이다.

POSIX 표준안의 장점은 반드시 UNIX가 아니더라도 C 언어에서 POSIX 표준 안이 지시하는 대로 함수호출을 보장하기만 하면 어떤 운영체제도 POSIX 표준으로 인정받을 수 있다는 점이다. 실제로 UNIX가 아닌 운영체제 이면서 'POSIX Compliance'를 선전문구로 내거는 운영체제가 나오고 있다.

POSIX 표준안은 C 언어를 통한 표준 시스템 서비스의 사용자 인터페이스를 제공하는데 중점을 두고 있으며, 시스템 서비스의 인자로 사용되는 자료 구조, 헤더 화일등의 정의와 오류 처리에 대한 호환성을 제공하고 있다. 이러한 POSIX의 출현으로 컴퓨터에 관련한 사람들은 다음과 같은 이득을 얻을 수 있다.

1) 컴퓨터 제조회사

각 기종에 특수한 시스템 호출등에 노력을 낭비하지 않고 주어진 시스템 인터페이스 환경내에서 어떻게 하면 우수한 기계를 만들것인가만 몰두하면 된다.

2) 소프트웨어 전문회사(ISV)

어떤 하드웨어에 소프트웨어를 만들것인가 걱정할 필요없이 POSIX를 위한 응용 소프트웨어를 만들면서 기능 향상에만 몰두하면 된다.

3) 사용자

하드웨어 기종이 바뀌더라도 응용 프로그램이 POSIX 인터페이스만 사용한다면 운영체제가 바뀌는데 따른 부담이 없다.

참고로 POSIX에서 정의하는 명세들은 다음과 같다.

- P1003.0 - 가이드 프로젝트(POSIX Guide Project)
- P1003.1 - 시스템 인터페이스 정의(System Interface Definition)
- P1003.2 - 쉘과 도구 인터페이스(Shell and Tools Interface)
- P1003.3 - 시험과 검증(Testing and Verification)
- P1003.4 - 실시간 확장(Realtime Extension)
- P1003.5 - Ada 언어와의 바인딩(Ada Language Bindings)
- P1003.6 - 보안기능 확장(Security Extensions)
- P1003.7 - 시스템 관리(System Administration)
- P1003.8 - 네트워킹(Networking)
- P1003.9 - Fortran 언어와의 바인딩(Fortran Language Bindings)
- P1003.10 - 수퍼컴퓨팅(Supercomputing)
- P1003.11 - 트랜잭션 처리(Transaction Processing)

P1003.1 표준은 현재 1990년에 정해진 IEEE Std 1003.1 - 1990이 나와 있고 그 후 많은 첨삭이 가해져서 1993년에 새로운 버전으로의 승인(Approval)을 기다리고 있다. 또한 어떤 운영체제가 P1003.1 사양을 만족하는가 검사하는 검증도구가 있는데 PCTS(Posix Compliance Testing Suite)라고 하며 미국의 NTIS(National Technical Information Service)에서 구입할 수 있다.

4. X/Open XPG3

X/Open은 컴퓨터 제조 회사들의 국제적 모임이다. 이들이 1989년 일곱 권으로 된 XPG3(X/Open Portability Guide, Issue 3)를 내어놓았는데 그 중 가장 중요한 것이 시스템 인터페이스와 헤더를 정의한 Volume 2(XSI System Interface and Headers)이다. 이는 IEEE Std. 1003.1 - 1988에 기초했는데 POSIX에 있지않은 부가 기능들도 많이 언급하고 있다. 현재 XPG4에 대한 작업이 진행중이며 1993년 내에 완결되리라 예측된다.

Ⅲ. 기술 동향

운영체제의 기술 동향에 대해 SVR 4.0을 중심으로 알아보자. 주로 90년대 중반까지를 염두에 두고 언급하겠으며 다음과 같이 몇 가지로 분류하여 설명하겠다.

- 기본적 시스템 기능 향상
- 분산처리 관련 기능 향상
- 멀티 미디어 관련 기능 향상
- 객체 지향 관련 기능 향상
- 트랜잭션 처리 관련 기능 향상

1. 기본 시스템

1) 커널 구조

운영체제 개발에 있어 가장 두드러진 추세 중 하나가 마이크로 커널 기술이다. 마이크로 커널이란 가장 기본적인 기능만을 모은 최소단위(마이크로 커널) 위에 모듈화 되어있고 분산 가능한 여러 시스템 서버를 고속 메세지 전달 기술로 엮는 운영체제 기술을 의미한다. 기존의 커널은 서로 밀결합 되어 있어서 커널 자체가 시스템에 주는 부담이 컸고, 모듈별로 나뉘어져 있지 않았기 때문에 디버깅 및 유지/보수가 어려웠던 것이 사실이다. 커널을 최소화(마이크로화)하는 작업과 병행하여 객체지향 기술을 접목시킴으로써 향후 클러스터 시스템을 만드는 작업이나 객체지향 개발환경을 만드는 작업중 많은 부분을 커널이 책임질 수 있게 될 것이다.

2) 비동기 입출력(Asynchronous I/O)

비동기 입출력이 제공되는 시스템에서는 응용 프로그램이 프로그램과 저장매체간의 데이타 전송을 야기시킨 후 전송이 끝났음을 나중에 통고받음으로써 응용 프로그램에게서 데이타 송수신에 관한 부담을 덜어주게 된다. 응용 프로그램 개발자는 이 기능을 사용함으로써 많은 시간을 기능향상에 쏟을 수 있고, 가장 최소한의 시간만 운영체제 관련 함수를 사용하면 된다.

3) 자동 형상관리(Automatic Configuration)

지금까지의 UNIX는 대부분 새로운 장치를 설치하려는 경우 커널을 재설정(Reconfiguration)해야하는 번거로움이 있었다. 그러한 작업이 그리 어려운 작업은 아니지만 대부분 시스템 관리자가 수행해야하는 작업으로 간주되었고, 시스템을 모르는 일반 사용자가 하기에는 불편함이 있었다. 이러한 사용자의 불편을 없애고자 나온 기능이 '자동 형상관리'로서 시스템이 형상 변경을 알아서 감지하고 필요한 작업수행을 사용자의 간섭없이 실행하는 기능을 한다.

4) 연속 동작(Continuous Operation)

기존 시스템에서는 디스크나 테이프 장치등을 설치하고자할 때 시스템을 정지(Shutdown)시키고서야 작업이 가능하였다. 이로 인해 일정 시간 시스템 사용을 중지시켜야하는 번거로움이 있었다. 이러한 불편을 없애기 위해 시스템 중지없이 여러가지 시스템 관리 동작이 가능하도록 하는 기능이 '연속 동작' 기능이다.

5) 분산형 스트림(Remote Streams)

스트림은 SVR3에서 처음 소개된 프로그램 인터페이스로 통신 프로토콜 모듈과 장치 구동기를 개발하는데 주로 사용되었다. 그러나 기존의 스트림은 하나의 호스트 컴퓨터 및 커널과 밀결합되어 동작하게 되어 있었는데, 이를 원격 호스트에서도 동작할 수 있게 함으로써 호스트의 부하를 덜어주는 기능을 해줄 수 있는 것이 분산형 스트림 인터페이스이다. 이로 인해 UNIX 커널과 통신 프로세스간에 스트림 모듈을 연결시키는 기능을 제공할 수 있게 되었다.

6) 고성능 입출력

최근 고성능의 빠른 매체(Media)와 네트워킹이 급속도로 발전하면서 운영체제 차원에서 이들을 지원해주어야하는 필요가 생기게 되었다. 특별히 실시간 응용과 멀티미디어 응용등에 필수적인 고속 디지탈 네트워크(예: FDDI), 고속 디스크 입출력 채널(예: HPPI)의 사용이 가능하게끔 운영체제 수준에서의 지원이 있을 것이다. 향후 5년간 퍼스널 컴퓨팅 분야에서 가장 주목되는 부분이 멀티미디어 기술이 되리라 판단되는데 운영체제 수준에서의 관련 기술 지원은 반드시 필요하다고 하겠다.

7) 국제화와 지역화

국제화(I18N: Internationalization)에 관련된 이슈는 한글을 고유언어로 사용하는 우리에게는 매우 민감한 문제가 되어왔다. 지금까지는 커널 및 명령어를 원천 코드 수준에서 일일이 한글화하는 단계에서 진일보하여 다중 바이트 처리에 공통적으로 필요한 부분을 합한 다국언어 지원기능(Multi-National Language Supplements)과 각 나라에 특수한 기능

을 구현하는 지역화(L10N: Localization) 패키지로 모듈화하는 단계에 이르렀다. 이에 더하여 부트 및 시스템 초기화, 다양한 명령어 지원, 광폭 문자(Wide Character)에 대한 완벽한 지원이 이루어질 것이다. 이는 UNIX가 여러 국가에 걸쳐 이식성이 뛰어나다고하는 명성을 얻기위해 필수적이라 하겠다.

8) 다중 프로세싱(Multiprocessing)

프로세서를 여러 개 사용하기 위해서 공유 메모리를 이용한 밀결합 다중 프로세싱(Tightly Coupled Multiprocessing)이 향후 몇 년간은 중요한 이슈가 될 것이며 아마도 퍼스널 컴퓨팅 환경에서도 다중 프로세서 시스템을 사용하는 시기가 올 것이다. 그 후 loosely coupled 다중 프로세싱 기술에 관심이 모아질 것이다.

우선 밀결합 다중 프로세싱을 지원하기 위해서 운영체제 수준에서 다음과 같은 기능들이 제공되리라 보인다.

- 다중 프로세싱의 기능을 십분 이용하게 해주는 다중 쓰레딩(Threading)
- 동시 사용 가능하게 디자인된 시스템 라이브러리 함수
- 응용 프로그램 디버깅 도구
- 다중 프로세서들의 부하 균등분배를 보장하는 스케줄링
- 병렬 프로그래밍 인터페이스(Parallel Programming API)

이러한 기능들이 제공된다면 고성능 서버 응용 프로그램, 고성능 OLTP, 그래픽, 윈도우 프로그램들의 수행에 많은 도움이 될 것이다.

9) 보안 기능

보안 기능 확장은 미국 국방성(DoD)에서 정의한 오렌지 북(Orange Book)에 명시된 C1 수준부터 A1 수준까지의 요구 사항을 만족시키는 것을 말한다. 현재의 UNIX로는 C1 수준을 만족하며 B1/B2 수준을 만족시키기 위해 커널의 많은 부분을 고쳐야 한다. SVR4 ES/MP 작업이 진행 중인데 ES (Enhanced Security) 패키지에서 B1/B2 수준을 만족시켜줄 것이다. 참고로 가장 높은 보안 수준인 A1은 B3 수준보다 요구사항이 더 많은 것은 아니고 단지 B3 수준의 디자인을 공식적으로 증명해 보여야 한다. A1 수준의 운영체제로는 Honeywell의 SCOMP가 있다.

10) POSIX

가까운 미래에 UNIX에서 지원할 예정인 POSIX는 2.3절에서 언급한 것들 중 다음의 표준을 지원할 것이다.

- P1003.4 실시간 확장
- P1003.4a 다중 쓰레드
- P1003.6a, b & c 보안 인터페이스 확장
- P1003.8 화일 접근 투명성 확장(Transparent File Access)

11) 복구(Recovery)

재난복구(Recovery)는 위급상황이 발생했을 때 시스템이 견디어내거나 복구하는 기능을 말하는데, 이러한 위급상황으로 부터의 복구는 사람의 간섭이 최소화되면서 자동적으로 수행되어야 할 것이다. 기본 운영체제 수준에서 하드웨어적 재난처리기능(Fault Handling Mechanism)을 구현한다는 것은 비용이 너무 많이 들며 새로운 시스템 소프트웨어를 릴리스하는데도 번거로움이 많다. 그래서 한단계 위인 컴퓨터 구조(Architecture) 수준에서 구현을 하고자 하는데 이 경우 시스템 제조회사마다 다양한, 하지만 일관성이 상실될 수 있는, Fault Tolerancy가 구현될 것이다.

12) 확장성(Scalability)

UNIX가 현재 가장 이식성이 있고, PC에서 메인프레임에 이르기까지 다양한 선형성을 자랑하지만, 32 비트 이상의 어드레스를 처리한다거나 많은 프로세서를 지원한다거나 하는 면에서 부족한 점이 많다. 어떤 경우 중,소 시스템에서는 좋은 성능을 내는 알고리즘이 다중 프로세서상의 높은 부하상태에서는 제대로 동작하지 못하는 일도 있다. 그래서 선형성을 유지하기위해 여러 조건을 만족시켜주는 방향으로 성능 향상을 꾀하고 있는데 주로 다음과 같은 것을 포함한다.

- 대형 시스템 지원

64 비트 프로세서를 사용하는 대형 시스템에서 시스템 자원을 효율적으로 사용하고 높은 부하상태에서도 잘 동작하도록 지원한다.

- 확장된 시스템 자원 지원

화일의 최대 크기, 메모리 주소 크기등이 제약을 받아왔는데 이러한 제한은 기술의 진보로 극복할 수 있게 되었다. 즉 화일 크기 제한은 논리적 볼륨 관리자(Logical Volume Manager)와 RAID(Redundant

Array of Inexpensive Disks) 기술로 극복이 가능할 것이다. 또한 가상 메모리 크기도 현재의 4 기가 바이트에서 확장될 것이다.

- 장치 지원

기존의 UNIX에서 장치 구동기에는 주번호와 부번호(Major & Minor Number) 가 할당되었는데 시스템이 지원해야하는 장치의 수가 늘어남에 따라 이러한 방법의 한계가 드러나게 되었다. 새로운 방법으로 장치의 번호(ID)를 구분할 수 있는 방법이 개발되어 매우 큰 장치 형상관리를 가능하게 할 것이다.

13) XPG4

XPG4가 93년도에 발표되면 그 이후 나오는 운영체제는 모두 이를 만족시키기 위해 노력할 것이다. UNIX 향후 버전에서는 주로 XPG4 시스템 인터페이스와 XPG4 국제화 관련 요구사항들을 만족시킬 것이다.

2. 분산 처리

1) DCE

DCE(Distributed Computing Environment)란 서로 다른 이기종 시스템 간의 상호운용성을 높이기 위해 OSF에서 개발한 프로토콜 및 서비스를 말한다. 이를 UI에서 채택하여 SVR4 시리즈에 이식하게 될 것이다. DCE로 인하여, DCE 프로토콜만 지원하는 시스템이라면 이기종이더라도 사용자에게 투명하게 상호운용될 수 있을 것이다. DCE에는 여러가지 기능이 있는데 몇가지만 언급하자면,

- 셀 디렉토리 서비스(Cell Directory Service)

Internet의 Domain Name Service와 X.500을 포괄한 서비스.

- RPC(Remote Procedure Call)

NIDL(Network Interface Definition Language) 컴파일러 및 DCE RPC 프로토콜의 실행환경(Run-time Environment)을 제공한다.

- 보안

Kerberos 프로토콜을 사용하여 클라이언트/서버 양자간 진위확인(Authentication)을 통한 보안기능 강화.

- Time Service

네트워크상의 시스템 시간을 동기시키는 기능.

2) 분산 응용 프로그램 지원

응용 프로그램 개발자가 분산 응용 프로그램을 쉽게 만들 수 있도록 여러가지 측면에서 지원한다. 그 한가지로 개발환경(Development Framework)을 만들어주고 실행 환경(Run-time Framework) 서비스를 제공하여 응용 프로그램이 다른 응용 프로그램과, 또는 다른 시스템과 잘 융화되도록 도와준다.

3) 분산 시스템 관리

원격 시스템 관리가 가능하게 해주는 여러가지 서비스를 제공하는데 가음과 같은 것들이 있다.

- Backup 및 Restore

시스템 관리자와 일반 사용자가 쉽게 사용할 수 있는 Backup/Restore 환경을 제공한다. 분산 환경하에서 데이타 유실(Loss)을 막아주는 것이 가장 중요한 기능이다.

- 장치 관리

물리적/논리적 장치를 관리하고 할당하는 기능과 사용자 프로세스와 장치 접근을 연결시켜주는 기능을 제공한다. 이러한 기술은 향후 계층형 저장 장치(Hierarchical Storage), Tape Carousels, 광디스크 쥬크박스 (Jukebox)를 지원하기 위한 기본 기술이다.

- 네트워크 관리

기존의 거의 모든 프로토콜(SNMP, CMIP, SNA, etc.)을 지원하며 향후 새로운 프로토콜을 지원하기 쉽게끔 만들어진다.

- 프린트 관리

레이져 프린터, 라인 프린터, 팩스, 35mm 슬라이드 카메라를 포함하는 다양한 출력장치를 모두 관리하는 기능을 제공한다.

4) Data Center

분산 시스템 관리의 일환으로 다음과 같은 분야에 대해 연구가 진행중이다.

- 구좌(Account) 관리
- 배치(Batch) 관리
- DCE 관리
- 메일 관리
- 문제(Problems) 관리
- 성능(Performance) 관리
- 원격/국지보안(Remote/Local Security) 관리

3. 멀티미디어

UNIX는 멀티태스킹과 네트워킹을 통해 멀티미디어 관련 서비스를 어느 정도 지원해 왔다. 향후에는

멀티미디어 통합환경(Framework)을 통해 관련 서비스를 확장할 것이다. UNIX 멀티미디어 환경이 지원하는 서비스는 다음과 같다.

- 드라이버

open/read/write 확장, 드라이버간 통신기능 보완, JPEG/MPEG 지원을 위한 기능 보완을 포함한다.

- 화일 시스템

2 메가 바이트보다 큰 화일을 지원하기 위해 CD-ROM 화일 시스템 및 표준 멀티미디어 포맷을 지원한다.

- 응용 인터페이스

X11 확장기능(이미지, 오디오, 비디오), 폰트 및 텍스트 지원, 쓰레드 라 이브러리 지원, 데이타 압축 및 객체 클래스를 지원한다.

- 통신

현존하는 멀티미디어 응용 프로그램 및 데이타와 상호운용성을 유지한다.

가까운 미래에는 데스크탑 PC 산업이 멀티미디어 솔루션을 제공하면서 이 분야를 선도하리라 보인다.

4. 객체 지향

1) 객체 지향 환경

객체 관리 환경(Object Management Environment)을 통하여 분산형 객체지향 응용 프로그램을 만들고 실행하는데 필요한 도구를 제공할 것이다. 이 기술은 OMG(Object Management Group)의 CORBA(Common Object Request Broker Architecture)에 바탕하여 만들어 지고 객체간 인터페이스는 인터페이스 정의언어(IDL: Interface Definition Language)에 의해 이루어 진다. IDL 컴파일러는 객체정의를 컴파일하여 동작(Operation)을 위한 Method를 수행하는 Server Stub을 만들어 낸다. 클라이언트는 ORB(Object Request Broker)를 통해서 동작을 야기시킨다. ORB는 클라이언트의 요구를 서버 객체에게 보내고 동작을 수행하기 위한 적절한 Method를 야기시킨다.

이러한 객체지향 환경을 지원하기 위해 다음과 같은 서비스를 제공한다.

- IDL 컴파일러
- ORB(Object Request Broker)
- Object Adaptor(Server Stub을 야기시킨다)
- 객체 저장 시스템
- 객체 인스턴스를 만들고 삭제하는 서비스

또한, 객체지향 응용 프로그램 개발을 지원하기 위해 객체지향 컴파일러 및 디버거를 제공한다.

2) OLE/DDE

Microsoft의 OLE(Object Linking & Embedding) 및 DDE(Dynamic Data Exchange) 표준을 수용하여, 한 응용 프로그램의 데이타 화일이 다른 응용 프로그램의 데이타로 사용되며 한 프로그램이 데이타를 변환시키면 여러 데이타 인스턴스의 무결성을 보장해 주는 동작을 수행한다.

5. 트랜잭션 처리

1) OSI-TP

OSI TP(Transaction Processing)는 이기종 트랜잭션 처리 시스템 간의 상호운용성(Peer-to-Peer Interoperability)을 위한 프로토콜 정의이다. 이 프로토콜은 이기종 트랜잭션 시스템이 광역 트랜잭션(Global Transaction)안에서 조화되도록 해준다. X/Open에서는 현재 OSI-TP를 위한 프로그래밍 인터페이스(API)를 만들고 있다. OSI-TP를 지원하는 다른 시스템과 상호운용시 문제가 없도록 UNIX에서도 운영체제 수준에서 OSI-TP를 지원할 것이다.

2) X/Open 트랜잭션 사양

X/Open에서 분산 트랜잭션 처리와 관련하여 만들었거나 가까운 장래에 완성되어질 사양은 다음과 같다.

- X/Open Peer-to-Peer, RPC, XATMI 트랜잭션 처리 인터페이스.
- X/Open XA: 트랜잭션 모니터와 자원 관리자간의 통신 사양.
- X/Open TX: 응용 프로그램과 트랜잭션 모니터간의 통신 사양.

이러한 사양을 UNIX가 지원함으로써 다른 모든 개방형 시스템 플랫폼과 트랜잭션 처리가 호환되도록 할 것이다.

Ⅳ. Windows NT

Microsoft에서 새롭게 발표한 Windows NT는 Microsoft의 Windows 운영체제들 중에서 완전한

32-bit 체제를 지원하는 Preemptive Multitasking 시스템이다. 이 시스템은 계속적으로 변화, 발전하는 하드웨어 기술의 잇점을 최대한 활용하기 위하여 Windows의 그래픽 인터페이스 사용편리성 및 사용자들에 의해 검증된 생산성을 운영체제의 새로운 기술들과 결합하고 있다. 본 장 이후에서는 이러한 배경을 가지고 탄생된 Windows NT의 주요 특징과 시스템 전반에 걸친 기술적 사항에 대해서 알아보도록 하겠다.

1. Windows NT의 시스템 목표

Microsoft의 NT 개발그룹이 파악한 90년대의 운영체제에 대한 사용자 요구사항은 다음과 같다.

첫째, 운영체제의 이식성(Portability)은 우선적으로 요구되는 중요 사항이다. 이를 위해 이식성이 강한 프로그래밍 언어를 사용하는 것이 요구된다.

둘째, 운영체제는 응용 프로그램들이 다양한 하드웨어의 장점을 충분히 활용할 수 있도록 다중처리기능(Multiprocessing)과 시스템의 확장성(Scalability)을 지원해야 한다.

세째, 컴퓨팅 환경이 다운사이징 되어가는 추세를 반영하여 하드웨어 자원의 공유및 분산 처리 환경을 위한 네트워크 기능이 증대되어야 한다.

네째, 새로운 운영체제는 POSIX 표준을 만족해야 한다.

다섯째, 다중 사용자 환경에서 적절한 보안 기능은 필수적이다.

이와 같이 파악된 사용자 요구사항을 바탕으로 하여 NT 개발그룹은 Windows NT를 Windows의 그래픽 환경을 지원하며, 새로운 응용 프로그램 개발에 대한 32-bit 프로그래밍 인터페이스인 Win32 API를 지원하는 Microsoft 최초의 Windows-based 운영체제로 성격을 규정하였다. 여기서 Win32 API는 다중 쓰레딩(Multithreaded Process), 동기화, 보안기능, 입출력, 객체관리(Object Management)등과 같은 특성을 이용하여 운영체제의 고급 기능들을 응용 프로그램에 가능하게 한다. 이렇게 해서 설정된 디자인 목표의 우선순위와 이를 실현하려하는 대략적인 전략은 다음과 같다.

첫째, 시스템이 지속적으로 개량, 발전되기 위해서는 추후의 확장및 개선이 용이해야 한다. 이를 위해 Windows NT에서는 안정적으로 작동해야 하는 기반 부분인 Executive와 요구사항의 변화와 더불어 계속적으로 성능이 개선되어야하는 서버부분인 Protected Subsystem 부분으로 시스템을 양분하고 있다. Protected Subsystem 외에 확장성을 위해 Executive의 모듈화, 객체개념의 사용, 적재가능한 입출력 디바이스 드라이버 지원, RPC 지원등과 같은 기능들을 통해 시스템의 추후 확장이 용이하도록 하고 있다.

둘째, 사용자 요구사항에서 파악된 바와 같이 하드웨어 플랫폼간의 이식이 용이해야 한다. 이를 위해서는 다양한 시스템에서 사용가능한 프로그래밍 언어를 이용하는 것이 필요하고, 시스템 구축시에 이식하고자 하는 대상 시스템에 대한 고려가 수반되어야 하며, 하드웨어 종속적인 부분을 최소화하고, 피할 수 없는 부분은 별도로 쉽게 관리할 수 있도록 해야 한다. Windows NT는 용이한 이식을 위하여 시스템의 대부분을 C 언어를 이용하여 작성되었고, Windows 환경의 그래픽 부분과 네트워킹 사용자 인터페이스 부분은 C++을 이용하여 작성되었다. 어셈블리 언어는 하드웨어와의 통신이 필요한 부분과 최적의 속도가 요구되는 부분에만 사용하고, 그 부분은 고립화되어 관리된다.

또한 프로세서에 종속적인 부분은 독립 모듈로 구성하여 다른 프로세서에 대한 유사한 모듈로 대치하는 것이 가능하도록 하고 있으며 동일한 프로세서를 사용하는 시스템간의 하드웨어 플랫폼에 종속적인 부분은 HAL(Hardward Abstraction Layer)이라는 DLL(Dynamic Link Library)로 구성함으로써 이식을 위한 배려를 하고 있다.

세째, 시스템은 내부적인 오동작, 외부 환경의 변화와 무관하게 예측가능하게 작동해야 하며, 사용자 프로그램의 오동작으로 부터 운영체제 자신 및 사용자를 보호해야 한다. 이를 위해 Windows NT에서는 소프트웨어 및 하드웨어의 오류에 대해 일관되게 대처하기 위한 방법으로 Structured Exception Handling을 사용한다. 시스템 안정성을 위한 다른 요소로는 Executive의 분할 독립요소로의 구성, 모든 종류의 디스크 에러로부터 복구가 가능한 NT File System(NTFS), 자원 배당, 객체 보호등을 지원하는 보안유지 구조등을 들 수 있다.

네째, 기존의 Microsoft 시스템들과의 호환이 가능해야 한다. Windows NT에서는 Protected

Subsystem을 이용하여 Win32 API 이외의 실행환경을 제공하는데, Intel 프로세서에서 작동하는 경우에 Protected Subsystem은 MS-DOS, 16-bit Windows, OS/2, LAN Manager를 포함하는 기존의 Microsoft 응용 프로그램들과의 호환을 제공하고, MIPS 프로세서에서 작동하는 경우에는 MS-DOS, 16-bit Windows, LAN Manager 응용 프로그램(에뮬레이터를 사용)의 호환을 제공한다.

다섯째, 위에서 언급한 디자인 목표를 만족하는 범위내에서 시스템은 각 하드웨어 플랫폼에서 최대한의 성능을 나타내야 한다. Protected Subsystem 중에서 클라이언트 어플리케이션과 빈번한 통신을 수행해야 하는 부분은 고속 메시지 패싱 메커니즘인 LPC (Local Procedure Call)를 이용하여 성능을 보장한다.

2. Windows NT의 주요 특징

Windows NT는 완전한 32-bit 연산, Preemptive Multitasking, 향상된 보안성과 신뢰성, 내장된 네트워크 기능, 고성능 개인용 컴퓨터 및 워크스테이션, 서버등의 다양한 하드웨어 플랫폼에 걸친 선형성을 제공한다. 또한, Intel x86 CPU, MIPS RISC CPU등에 기반한 하드웨어등을 포함하는 개인용 컴퓨터의 능력을 사용가능하게 하고 있으며, 대칭적 다중처리 기능도 제공한다. 게다가 이러한 하드웨어를 기반으로 하는 MS-DOS나 Windows 운영체제를 위한 기존의 응용 프로그램을 수정없이 실행하는 것이 가능할 뿐만 아니라 강력한 32-bit Windows 응용 프로그램도 실행할 수 있다.

주요 특징을 부연해서 살펴보면, 우선 Windows NT는 천만 카피이상 공급된 대중적인 Windows의 잇점-사용 편리성과 배우기 용이함-을 취하고 있으며 기존의 Intel과 MIPS 기반 시스템에서 동작하는 많은 응용 프로그램을 수정없이 실행시킬 수 있고, 또한 새로운 32-bit 윈도우즈 응용 프로그램도 실행하는 것이 가능하도록 하였다. 여기에 OS/2 Server와 POSIX 응용 프로그램을 지원한다.

다음으로 생각해 볼 수 있는 것으로서 Windows NT의 확장 아키텍처는 90년대의 하드웨어 플랫폼상에서 응용 프로그램을 보다 효율적으로 실행할 수 있도록 해준다는 것이다. 즉, 완전한 32-bit 내부 아키텍처는 데이타를 더욱 효율적으로 이동하게 하며 32-bit 플랫 메모리 모델은 개발자로 하여금 응용 프로그램당 최대 2 GB의 메모리를 접근할 수 있도록 해줌으로써 많은 양의 데이타를 처리하는 프로그램의 개발을 용이하게 해준다. 대칭형 다중처리에 대한 지원은 Windows NT를 확장가능하게 하고, 메모리 보호기법은 운영체제와 응용 프로그램을 각각 다른 영역으로 로드함으로써 데이타의 충돌을 방지하고 안정성을 보장하려 하고 있다. Preemptive Multitasking 방식은 운영체제로 하여금 각 응용 프로그램에 프로세싱 타임을 효율적으로 할당할 수 있게 하고 확장된 메모리와 디스크 공간은 복잡한 응용 프로그램 수행을 가능하게 한다.

또한, Windows NT는 워크그룹 컴퓨팅을 통해 화일 공유기능과 프린트 공유기능을 내장하고 있으며 강력한 워크그룹 응용서비스 기능을 제공한다. 여기에 프로세스간 통신을 위하여 RPC(Remote Processing Call)를 이용하여 분산 응용 프로그램을 개발하도록 한다. 보안성에 있어서는 C2 등급의 보안환경을 제공하고 있다.

3. Windows NT의 기반 모델

1) 클라이언트/서버 모델(Client/Server Model)

운영체제는 다양한 방법으로 구조화될 수 있다. 그 한 가지 방법은 MS-DOS와 같은 작은 운영체제에서 흔히 사용되는 방법으로서 운영체제를 일련의 프로시쥬어로 구성하고 임의의 프로시쥬어가 다른 임의의 프로시쥬어를 호출할 수 있게 구성하는 방법이 있다. 이 경우 일반적으로 운영체제는 커널 모드에서 동작하고, 응용 프로그램은 사용자 모드에서 동작하게 함으로써 서로를 분리하고 응용 프로그램이 시스템 서비스를 요구하는 경우에는 적절한 조치를 하게 된다. 이런 획일적인 시스템은 확장하는데 다소 문제가 생기게 된다.

이와 다른 방법으로는 운영체제를 모듈과 층으로 구분하고, 각 층간에 계층 구조를 형성하는 방법이 있다.

각 모듈은 다른 모듈이 호출할 수 있는 일련의 함수를 제공하고 특정 계층의 코드는 하위 계층의 코드만 호출할 수 있도록 하는 것이다. 이러한 구조는 「그림 1」과 같이 표현할 수 있으며, 시스템의 기능 확장이 용이하고, 계층간의 호출을 제한함으로써 디버깅이 용이하다는 장점을 가진다.

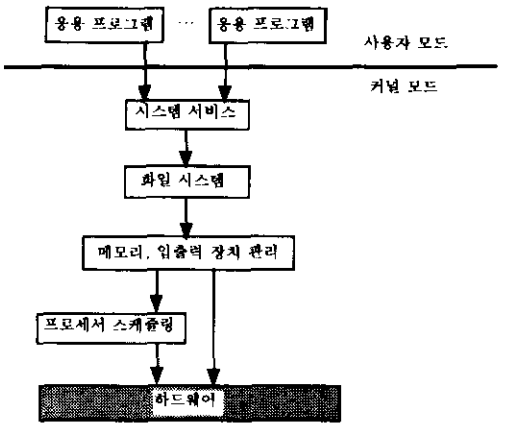


그림 1. 계층화된 운영체제

또 다른 방법으로는 운영체제를 클라이언트/서버 모델로 구성하는 것이다. 이러한 아이디어는 운영체제를 각각 특정 서비스를 구현하는 일련의 프로세스 서버- 예를 들면, 메모리 서비스, 프로세스 생성 서비스, 프로세서 스케쥴링 서비스 등 로 구성하고, 각 서버들은 사용자 모드에서 대기를 하면서 클라이언트가 요청하는 요구를 받아들여 서비스를 수행하는 과정을 거치게 된다. 이러한 과정은 「그림 2」와 같이 표현할 수 있는데 이것은 이상적인 모형이고, 실제는 커널 모드에서 수행하는 작업의 종류와 양에 따라 다양한 스펙트럼을 가진다. 클라이언트/서버 모델 접근 방식을 사용하게 되면, 운영체제를 독립적인 작은 모듈 성분들로 구성할 수 있게 되고 분산 환경에 적합시키는 것이 용이하게된다.

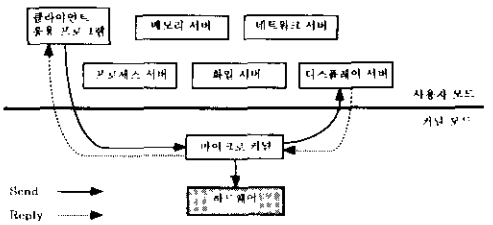


그림 2. 클라이언트/서버 운영체제

Windows NT는 계층화된 모델과 클라이언트/서버 모델로부터 개념을 빌려오고 있다. Windows NT의 커널모드 부분을 NT Executive라 하는데, 이 부분은 가상 메모리 관리, 객체 관리, 입출력, 화일 시스템, 프로세스간 통신, 보안 시스템등을 구현하는 일련의 부분들로 구성이 된다. 이러한 부분들 상호간은 모듈화된 방식으로 상호 작용을 하지만, NT Executive의 최하위 입출력 부분, NT 커널, HAL에서는 계층화된 모델이 사용된다. 즉, 다른 부분들이 이 두 부분위에 계층화되게 된다. 「그림 3」에서와 같이, Windows NT에서는 API와 운영체제 환경이라고 여기는 기능들을 제공하기 위해 클라이언트/서버 모델을 사용한다. 클라이언트/서버 모델은 NT Executive를 간단하게 하는데, Windows NT로 하여금 Win32, MS-DOS, 16-bit Windows, POSIX, OS/2 API를 각각의 서버로 구성하여 제공함으로써 충돌을 방지하고 확장을 용이하게 하며 또한 신뢰성을 높인다. 아울러 분산 컴퓨팅 모델에 적합한 잇점을 제공한다.

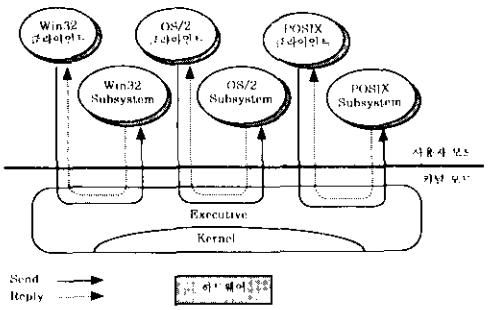


그림 3. Windows NT의 클라이언트/서버 구조

2) 객체 모델(Object Model)

운영체제를 객체지향 방법론적 시각으로 바라보면, 시스템이 다루는 여러 가지 형태의 데이타에 우선 관심을 갖게 된다. 이 경우의 데이타는 보통 시스템 자원의 형태를 띠게 되는데, 데이타를 중심으로한 시스템의 목표는 변화시키기가 용이한 시스템을 구축하는 것이다. 이 경우, 변화를 최소화할 수 있는 방법은 데이타의 물리적 표현을 객체내에 숨기는 것이다.

Windows NT는 완전한 객체 지향적 시스템은 아니지만 시스템 자원을 표현함에 있어서 객체를 사용하고 있다. 화일이나, 공유 메모리, 물리적 디바이스 등 하나 이상의 프로세스에 의해서 공유될 수 있는 시스템 자원은 객체로서 구현되고 객체가 제공하는 서비스에 의해 다루어진다. 이러한 접근 방식은 시간

이 경과함에 따라 시스템에 생기게 되는 변화의 영향을 최소화할 수 있게 해준다. 또한 시스템 자원을 접근하고 다룸에 있어서의 일관된 방식이 가능하며 이를 통해 보안 기능에 도움을 받을 수 있다.

3) 대칭적 다중처리(Symmetric Multiprocessing)

일반적으로 다중처리 시스템은 비대칭적 다중처리 방식과 대칭적 다중처리 방식으로 구분된다. Windows NT를 포함하는 대칭적 다중처리 방식에서는 운영체제로 하여금 임의의 가능한 프로세서에서 동작하도록 하거나 또는 모든 프로세서에서 동시에 그들 간의 메모리를 공유하며 동작하는 것을 가능하게 한다. 이 접근 방식은 비대칭적 다중처리 방식에 비해 다중 프로세서의 능력을 더욱 효과적으로 사용가능하게 한다. Windows NT는 다중처리 운영체제로서의 능력에 도움이 되는 다른 특성도 가지고 있는데, 우선순위가 높은 작업이 요구될 때, 커널 부분을 제외하고는 Preemption이 가능하다는 점과 한 프로세스 내에 여러 개의 쓰레드를 둠으로써 여러 프로세서에서 동시에 작업하는 것이 가능하게 한다는 점, 서버에 여러 쓰레드를 둠으로써 다수의 클라이언트로 부터의 요구를 처리하게 한다는 점, 프로세스간 통신간에 객체를 편리하게 공유하는 메카니즘이 존재한다는 것등을 들 수 있다.

V. Windows NT의 구조

Windows NT의 구조는 크게 커널 모드에 해당하는 부분인 NT Executive와 사용자 모드에 해당하는 부분인 Protected Subsystem으로 양분할 수 있다. 간략화된 Windows NT의 블럭 다이어그램은 「그림 4」와 같다.

여기서 Protected Subsystem은 서버 역할을 하는 부분으로 독립된 프로세스 내에 존재하는데, 이 프로세스는 NT Executive에 의해 보호되는 메모리 내에 존재하게 된다. 클라이언트와 서버간의 처리는 메시지 전달에 의해 이루어지는데 이 경우 항상 NT Executive를 거치게 되어있다. NT Executive는 운영체제의 엔진에 해당하는 부분으로서 임의의 수의 서버 프로세스를 지원하는데 서버들은 NT Executive에게 사용자 및 프로그래밍 인터페이스를 제공하고 다양한 응용 프로그램 실행환경을 제공한다.

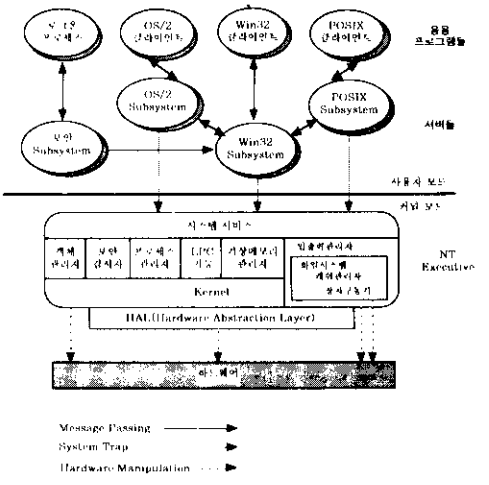


그림 4. Windows NT의 블럭 다이어그램

1. Protected Subsystems

Windows NT의 용어에서는 protected subsystems를 서버라고도 칭한다. 서버라는 호칭이 암시하듯이 protected subsystems는 프로그램이 호출할 수 있는 API를 제공하게 된다. 응용 프로그램이나 다른 서버가 API 루틴을 호출하게 되면 그 API 루틴을 구현하고 있는 서버에게로 NT executive의 LPC (Local Procedure Call)기능을 통해서 메시지가 전달되게 된다. 서버는 처리된 결과를 메시지를 통해서 다시 클라이언트에게로 보내게 된다. Windows NT는 두가지 종류의 protected subsystems를 가지고 있는데 environmental subsystems와 integral subsystems가 그것이다.

Environmental subsystems는 사용자 모드의 서버로서 운영 체제에 고유한 API를 제공하는 부분으로서 대표적인 것으로는 응용 프로그램으로 하여금 32-bit Windows API를 가능하게 하며 NT의 그래픽 사용자 인터페이스를 제공하는 Win32 subsystem이 있다. 이외에도 POSIX subsystem, 16-bit Windows subsystem, MS-DOS subsystem 등이 있는데, 이것들은 각각의 API를 제공하지만 사용자의 입력을 제공받고 결과를 출력하는 역할은 Win32 subsystem을 이용하여 수행한다.

Integral subsystems는 주요한 운영 체제의 기능

을 수행하는 부분으로서 Windows NT에 존재하는 것은 보안 기능을 담당하는 security subsystem과 네트워킹 서비스를 담당하는 network subsystem 등이 있다.

2. NT Executive

NT executive는 Windows NT의 커널 모드에 해당하는 부분으로서 사용자 인터페이스를 제외하고는 자체적으로 완전한 운영 체제이다. NT executive는 각각 두 가지 기능을 구현하는 일련의 요소로 구성이 되어있는데 그 두 가지 기능이란 environmental subsystems이나 다른 executive의 구성 요소가 호출할 수 있는 시스템 서비스와 executive 내의 구성 요소에게만 가능한 내부적인 루틴이다. 이를 그림으로 표현하면 「그림 5」과 같다. 각각의 구성 요소에 대한 특징 및 내용은 다음 절에서 설명한다.

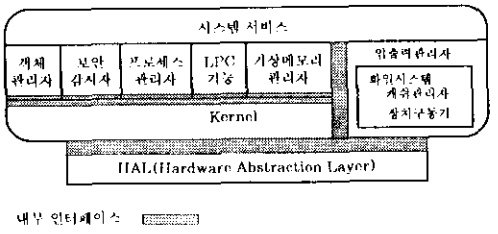


그림 5. 시스템 인터페이스

VI. Windows NT Executive 주요 구성요소의 특징

1. 커널(Kenrel)

NT 커널은 쓰레드를 스케줄링하고 배당하는 프로세스 제어를 하고, 인터럽트 및 예외상황 처리, 프로세스간 동기화, 그리고 NT Executive의 나머지 부분이 상위 계층의 객체를 구현하는데 사용하는 기본적인 객체와 인터페이스의 제공과 같은 작업을 담당한다. 특히 커널은 입출력 장치 구동기로 하여금 다중 프로세서에 걸쳐 실행을 동기화할 수 있게 하고 전원고장이 발생하여도 입출력을 복구할 수 있게하는 객체와 함수를 지원한다.

2. 프로세스 관리자(Process Manager)

NT에서는 프로세스를 독립적으로 실행가능한 단위인 쓰레드로 나누는데, 이러한 쓰레드는 다중 처리를 가능하게 한다. 프로세스 관리자는 프로세스와 쓰레드를 생성하고 종료하며, 쓰레드의 실행을 보류하거나 재기시키기도 하고 프로세스 및 쓰레드에 관한 정보를 저장하거나 추출하는 일을 담당하기도 한다.

3. 객체 관리자 (Object Manager)

NT에서 객체는 시스템 자원을 일관되게 관리할 수 있는 기본을 제공하는데 자원의 명명, 공유, 보호와 같은 작업의 중심이 되는 부분이 객체 관리자이다. 또한 객체는 환경 서브 시스템이 자신의 객체나 객체 타입의 자원을 구현하고자 할때 사용할 수 있는 기본 요소를 제공한다. 객체 관리자는 운영체제의 자원을 표현하는데 사용되는 NT Executive의 객체와 추상화된 데이타형을 생성, 관리, 삭제하는 등의 역할을 담당한다.

4. 가상 메모리 관리자(Virtual Memory Manager)

가상 메모리 관리자는 페이징 기법을 이용하여 가상 메모리를 구현함으로써 각각의 프로세스에게 메모리 제약을 덜어주고 프로세스간의 메모리 충돌을 방지한다. 가상 메모리 관리자의 구현은 부득이한 경우를 제외하고는 불필요한 시간소요 작업을 피하기 위해 Lazy-Evaluation 기법을 사용하고 있다.

5. 입출력 시스템(I/O System)

입출력 시스템은 입력을 처리하는 부분과 다양한 장치로의 출력을 담당하는 부분등의 일련의 구성요소로 이루어져 있는데, 이를 위해 입출력 관리자, 화일 시스템, 네트워크 리디렉터 및 서버, NT Executive 장치 구동기, 캐쉬 관리자 등의 부요소를 갖는다. 입출력 관리자는 입출력 처리 모델을 정의하고 하나 이상의 구동기에 공통으로 존재하거나 또는 요구되는 기능들을 수행하는 기능과 입출력 요구사항을 표현하는 IRP(I/O Request Packet)를 생성하고, 이를 구동기로 인도, 결과를 송부하는 등의 역할을 담당한다.

여기서의 구동기는 기존의 개념인 장치 구동기만을 의미하지는 않고 화일 시스템이나 네트워크 리디렉터 및 서버등도 포함하는 개념이다.

Ⅵ. 결론

지금까지 90년대 중반 컴퓨터 업계를 선도할 운영체제의 기술동향에 대해 UNIX 및 Windows NT를 중심으로 하여 살펴보았다. 이를 크게 보자면 향후의 운영체제는 첫째, 다양한 하드웨어에 이식이 가능한 방향으로 갈것이며, 둘째 상호운용성 증대를 위한 네트워크 및 분산처리에 치중할 것이고, 세째 여러가지 표준에 부합(Compliant)되는 방향으로 나아갈 것이다. 이에 대비하기 위해 우리로서는 각 분야에 전문적인 지식을 가진 엔지니어를 양성하는 일이 시급할 것으로 생각되며 이를 적절히 시행하지 못하게 되면 시스템 및 응용 소프트웨어 산업의 존립마저 위태롭지 않을까 생각된다

參考文獻

[1] Helen Custer, "Inside Windows NT", Microsoft Press, 1992.

[2] Boykin, Joseph, and Susan Loverso, "Recent Developments in Operating Systems", Computer, 1990.

[3] Charles Petzold, "Programming Windows", Microsoft Press, 1991.

[4] Unix International, "1993 Roadmap for UNIX Systems and Related Technologies", Unix International, 1993.

[5] IEEE, "Portable Operating System Interface for Computer Environments - IEEE Standard 1003.1-1988", IEEE, 1988.

筆者紹介

安 聖 辰

1952年 4月 18日生

1979年 6月 Ohio State University computer science Master

1974年 2月 서울대학교 공과대학 응용수학과 학사

1979年 7月 ~ 1986年 3月 Sperry Univac, OS 개발

1986年 4月 ~ 1990年 7月 AT&T Bell Labs FT/RT unix 개발

1990年 8月 ~ 현재 현대전자, S/W 연구소, 핵심기술분야 담당 이사

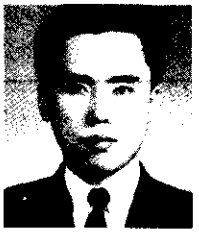
주관심분야 : OS, Computer Architecture, DBMS, OLTP 통신 S/W, Multimedia, CRS

筆 者 紹 介

金 精 玟
1964年 2月 12日生
1986年 2月 서울대학교 전자계산기 공학과 학사
1988年 6月 美 NORTHWESTERN UNIV 전산학 석사

1988年 7月 ~ 현재 현대 전자 근무

주관심분야 : Operating System



金 東 圭
1968年 10月 21日生
1991年 2月 서울대학교 계산통계학과 졸업(이학사)
1993年 2月 서울대학교 대학원 전산과학 전공(이학석사)

1993年 1月 ~ 현재 현대 전자 소프트웨어 연구소 기초기술 개발실 근무중

주관심분야 : 컴퓨터 비전, 멀티미디어