<Chapter1. 소개>

운영체제 : 컴퓨터 하드웨어와 사용자간의 매개체역할(연결역할)을 하는 시스템 소프트웨어 (결과를 사용자에게 직접적인 전달x) (EX. 컴파일러, interpreter).

운영체제의 목적 : 컴퓨터 시스템을 편리하게 이용, 컴퓨터 하드웨어를 효율적으로 관리

컴퓨터 시스템의 구성 : 1. 하드웨어 2. 운영체제 3. 사용자 4. 응용프로그램으로 나눌 수 있다.(그림 p.14)

-하드웨어 : 중앙처리장치, 기억장치와 입출력장치로 구성되어 계산을 하기 위한 기본적인 자원제공.

-응용프로그램 : 사용자가 제시한 문제를 풀기 위해 필요한 자원의 사용방법을 정의.

운영체제를 보는 두가지 관점 : 1. 운영체제는 자원할당자다. 2. 운영체제는 통제자다.

-자원할당자로 보는 관점 : 자원을 공정하고 효율적으로 관리하고 할당하니까 자원할당자다! (여기서 할당의 의미 : 여러개의 프로세스가 자원을 차지하려고 경쟁하는 중에 특정 프로세스에게 자원을 지정해주는 것)

-통제자로 보는 관점 : 다양한 입출력 장치, 사용자 프로그램을 통제하니까 통제자다!

운영체제가 충돌 문제를 고치기위한 문제점들 : 1.어떤 응용프로그램이 우선순위가 가장 높은가? 2. 어떤 파일이 하드 드라이브 상의 가장 빠른 검색이 가능한 위치를 부여받을것인가? 3. CPU의 처리속도와 디스크 접근속도 중 어떤 쪽이 더 중요한가? 4. 상대적으로 더 우대되어야 할 사용자의 일인 경우, 그 처리를 위한 대기시간의 최소화가 가능한가?

운영체제가 접촉해야하는 인터페이스 : 1. DB 2.문서편집기 3. 웹브라우저 4. 통꼐분석패키지

운영체제의 유형 : 1. 일괄처리 시스템 2. 다중프로그래밍 시스템3. 시분할 시스템 4. 실시간 시스템 5. 개인용 컴퓨터 시스템 6. 분산처리 시스템 7. 멀티미디어 시스템 8. 임베디드 시스템

-일괄처리 시스템(=batch processing system) : interactive의 반대개념. cpu의 유휴상태(idle state)의 시간을 줄이기 위하여 작업 순서의 자동화 개념 도입. 한 프로그램에서 다음 프로그램으로 제어를 자동적으로 넘기기 위해 상주 모니터(=원시os)를 둔다. 이를 통해 작업 간의 전이를 오퍼레이터의 개입 없이 자동적으로 수행하고자 했다. 그러나 하나의 작업이 시작되면 그 작업이 모든 시스템 자원을 독점해서 여러 시스템 자원,특히 중앙처리장치가 빈번한 유휴 시간을 가지게 되는 문제점이 있음.

-다중 프로그래밍 시스템(=multiprogramming system) : cpu안에 여러 프로그램이 동시에 실행되면서 cpu가 쉬는시간이 없게끔 만든다. 순차 실행하면 입출력처리 때문에 유휴가 발생하는데 다중 프로그래밍실행은 유휴시간 발생할때 다른 프로그램을 할당해서 결국 cpu가 놀지 못하게 한다. (여러개의 program을 준비상태로둔다) 하지만 이로인해 여러 작업들중 어느 작업을 실행할지 별도의 기억장치 관리 기법이나 중앙처리장치 관리 기법등이 필요하고 장치 스케줄링, 교착 상태(deadlock)문제, 병행 제어(concurrency control)및 보호(protection)문제등을 고려해야함. (그림 p.18)

-시분할 시스템(=time-sharing system) : 여러 사용자들이 짧은 시간단위만큼 공유한다.(보통 1/1000초, ms단위. Timeslot). 실제로는 아니지만 각 사용자로 하여금 자신만이 컴퓨터 시스템을 독점하여 사용하고있는것처럼 느끼게한다. 자원 분배가 공정하다. 사용자는 대화식(interactive) 단말장치를 이용해 시분할 시스템과 인터페이스하기때문에 즉각 답변을 받는다. 그러나 자원 제어에 대한 대부분의 책임을 운영체제에 전가시키기때문에 운영체제를 복잡하게 한다는 단점이 있다.

-실시간 시스템(real-time system) : ex.자율주행, vx works. 시스템 내부/외부에서 이벤트가 발생했을 때 지연시간과 결과 출력의 처리시간이 제한된 시간 안에 처리된다.

-하드 실시간 시스템(경성) : 엄격하게 지켜야한다.

-소프트 실시간 시스템(연성) : 약간은 차이나도 괜찮다.

-다중처리 시스템(=multiprocessing system) : 공유 기억장치 하나를 cpu들이 공유해서 쓴다(one memory many cpu). (다중 프로그래밍시스템은 program이 많은 것!!)

-밀착 결합(=tightly copled) : 프로세스들이 매우 밀접하게 동기화(=시간을 맞춤)

-느슨 결합(=loosely coupled) : 프로세서간의 동기화가 그렇게 엄격하지 않다. 2개 혹은 그 이상의 프로세서를 결합해서 처리율 높음.

다중처리 시스템의 문제점으로는 어떤 job이 들어왔을 때 어떻게, 어떤 cpu가 처리할지 해결책 제시해줘야한다.

(p.20그림)

-개인용 컴퓨터 시스템(=personal computer system) : 대형 시스템보다 작고 값이 싼 초소형 컴퓨터. 편리성, 응답성이 좋음.

-분산 처리 시스템(=distributed processing system) : 하나의 시스템처럼 보이는 독립된 컴퓨터들의 집합. 통신망에 연결된 여러 컴퓨터들의 처리 능력을 이용해 메세지 전달(message passing)을 통해 거대한 계산 문제를 해결하려는 분산처리 모델.

-

-종류 : 1.밀착된 결합 시스템에서는 프로세서들이 기억장치와 출력을 공유, 2. 느슨한 결합 시스템에서는 프로세서들이 기억장치와 클럭을 공유하지 않고 각 프로세서들은 자신의 지역(local) 기억장치를 가짐. 프로세서들은 고속의 버스(bus)나 전화선과같은 다양한 통신 라인을 통해 서로 통신

-운영체제 : 1. 네트워크 운영체제 2. 분산 운영체제

-네트워크 운영체제 : 노드 간 기종의 차이가 심하고 대규모 네트워크 시스템에 사용. 각 노드들은 독자적인 운영체제가짐, 통신기능은 기존 운영체제 위에 존재하며 자원 공유 곤란.

-분산 운영체제 : 각 노드들을 통괄할 수 있는 하나의 운ㅇ여체제로 운영, 통신 기능은 운영체제 설계시 고려, 미니, 마이크로컴퓨터등으로 연결된 근거리 네트워크 시스템에서 많이 사용되며 자원 공유가 용이.

-멀티미디어 시스템(=multimedia system) : 멀티미디어 콘텐츠를 저작할 수 있는 기능과 환경을 제공하는 시스템.

-저작 시스템(=authoring system) : 멀티미디어 콘텐츠를 제작할 수 있는 기능과 환경을 제공

* 재생 시스템(=presentation system) : 단순히 멀티미디어 콘텐츠를 이용하고 재생

멀티미디어 플랫폼(multimedia platform) : 멀티미디어 콘텐츠를 제작하고 재생하는 하드웨어 환경과 소프트웨어 환경

-임베디드 시스템(=embedded system) : 마이크로프로세서 또는 마이크로 컨트롤러를 내장하여 시스템 제작자가 의도한 몇 가지 혹은 특수한 기능만을 수행하도록 제작된 시스템. 임베디드 시스템은 전체 장치의 일부분으로 구성되며 제어가 필요한 시스템을 위한 두뇌 역할을 하는 특정 목적의 컴퓨터 시스템

운영체제에 대한 관점 : 1. 자원 관리자 관점 2. 프로세스 관점 3. 계층 구조 관점

-1.자원 관리자 관점 : 자원은 유한하기 때문에 경쟁, 충돌이 일어나기때문에 자원관리를 해줘야한다. (여기서 말하는 자원은 메모리, 디스크, IO device) 자원의 상태를 추적, 저장,

어떤 프로세스가 언제, 어떤 자원을, 얼마나 쓸건지 결정, 자원 할당, 자원 회수 과정을 수행한다.

-프로세스 관리 기능 : cpu와 프로세스 상태추적하고 저장, 어떤 작업에게 프로세스를 사용하도록 할건지 결정, 필요한 하드웨어 레지스터 설정하고 프로세스에게 cpu할당 (=디스패처)(=dispatcher)

-기억장치(=memory)(메모리) 관리 기능 : 어떤 프로세스에게 언제, 얼마의 기억장치를 할당할지 결정함.

-장치(device) 관리 기능 : 입출력장치 상태를 추적하고 저장한다. 입출력 스케줄링 : 장치할당 효율적으로 할 방법 결정. 해당 장치를 할당하고 입출력 동작을 시작하고 입출력이 종료되면 자동적으로 자원 회수.

-정보관리기능 : 파일시스템(File system) = 정보 위치, 사용여부, 상태를 추적하고 관리한다. 어떤 작업에게 정보 자원을 사용하도록 할지를 결정하고 정보 보호를 위한 대책을 수립하고 접근 루틴제시.

-2. 프로세스 관점 : 하나의 작업이 완료될때까지 상태 변경 관리한다.

-3. 계층구조 관점 : OS의 커널 = 기계 개념으로 운영되는 모든 운영체제 모듈을 총칭. ( OS = 커널 + UX) ( p.29그림 참고 – 계층적 기계 개념의 구성도)

입출력 프로그래밍 : CPU와 I/O device 속도차이가 문제 -> I/O device는 CPU와 비동기적으로(=따로) 수행.(=병행적으로 수행) (입출력장치와 CPU는 상호작용을 위해 입출력 프로그래밍이 필요함). 1.BIOS 2.부트진행과정 3. POST 4.버퍼링 5. 스풀링 6. 채널 7. 인터럽트

-1. BIOS(basic input/output system) : 부트 프로세스를 실행시키는 명령어들을 포함하고 있으며 이러한 형태의 컴퓨터 칩(Rom)에 저장된 명령어들을 펌웨어(firmware)라고 한다.

-2. 부트 진행 과정(boot process) : 컴퓨터 가동을 위해 파워 버튼을 누르면 부터 로더가 초기화 됨. POST(power on self test) 수행시작. 다른 BIOS들이 가동을 위해 초기화됨. 유저는 필요에 따라 BIOS접근을 위한 비밀키 입력을 요구 받음. 간단한 메모리 테스트가 수행되고 여러 파라메타들이 세트 됨. 프러그앤플레이 디바이스들이 초기화 됨. DMA(Direct Memory Access)채널을 위한 자원들과 IRQ(Interrupt Request)가 할당됨. 부트 디바이스들이 정해지고 초기화 됨. OS가 초기화 됨.

-3. POST(Power on self Test) : 컴퓨터에 전원이 공급되었을 때, 컴퓨터 키보드, 램, 디스크 드라이브 그리고 기타 하드웨어 등이 바르게 동작하는지를 확인하기 위해 컴퓨터 BIOS가 동작시키는 일련의 진단 시험 과정

-4.버퍼링(buffering) : cpu와 입출력장치의 속도차이를 보완하기 위한 방법. Cpu는 출력할 데이터를 버퍼에 넣고 다 채워지면 채널은 출력 지시 받고 출력해서 버퍼를 비우고 cpu에게 알려준다.

-이중 버퍼링 : cpu는 A버퍼, 채널은 B버퍼로 번갈아가며 작업 가능.

-5. 스풀링(SPOOLing) : 버퍼링과 똑같은데 하드디스크에 굉장히 큰 버퍼를 만든 것임.

-스풀링 vs 버퍼링 : 버퍼링은 주기억장치가 버퍼, 스풀링은 주기억장치가 디스크.

-6. 채널(channel) : cpu와 입출력장치 사이의 입출력 제어 장치.

cpu지시 -> 채널 작업 끝 -> polling 이용해서 cpu에게 보고하든가 interrupt이용해서 cpu에게 보고하든가 함. Polling이용하면 직접 다했는지 물어보기때문에 시간낭비가 되는 반면 interrupt는 채널 작업완료되면 interrupt 신호 발생시키는 방식(번호와 함께) 혹은 시스템에 예기치 않은 상황이 발생했을 때 OS에게 알려줌. - > cpu는 프로그램 정지 시키고 바로 interrupt처리 시작

-7. 인터럽트(interrupt) : 인터럽트는 시스템에 예기치 않은 상황이 발생했을 때 os에게 알려준다.

인터럽트 종류 : p.38~39참고.

구조 및 처리 : p.40참고