운영체제의 역할

안태진, 천현지

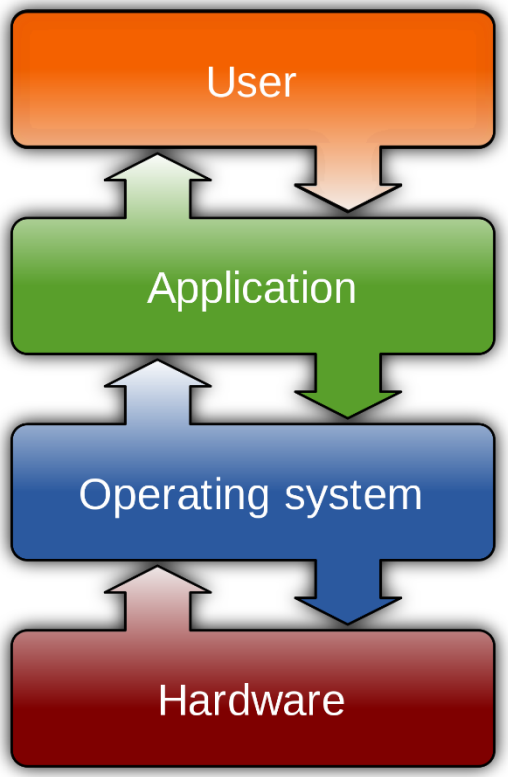
({taejin, hyunji}@codecure.smuc.ac.kr)

상명대학교 보안동아리 CodeCure

목차

운영체제의 정의

- 운영체제란(Operating System)?
  - 사용자 관점
    - 하드웨어와 응용 소프트웨어 중간에 위치하여 사용자들이 컴퓨터를 간편하게 이용할 수 있도록 도와주는 시스템 소프트웨어
  - 시스템 관점
    - 컴퓨터 시스템이 보유하고 있는 자원들을 효율적으로 관리하는 시스템소프트웨어

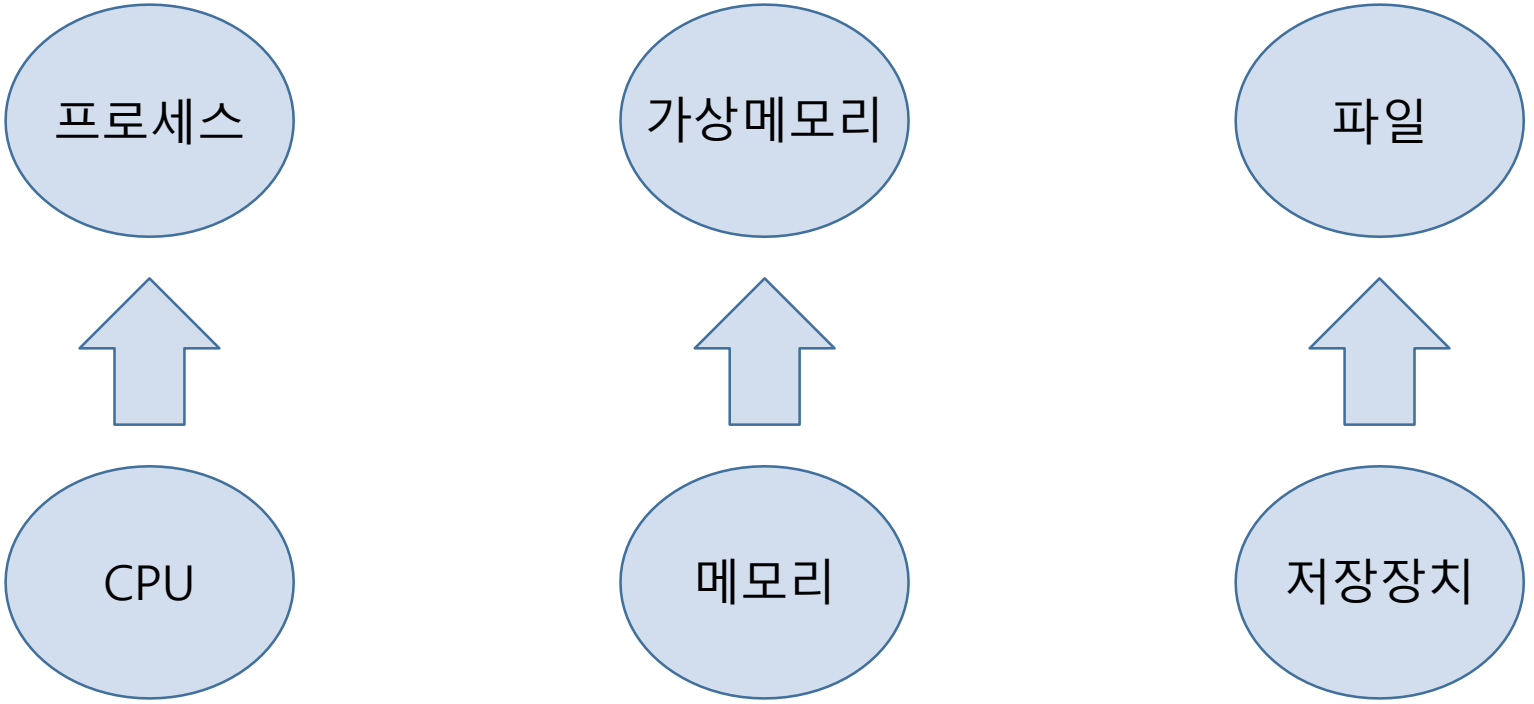


목차

- 운영체제의 정의
- 운영체제의 목적
- 운영체제의 역할
  - 사용자 인터페이스
  - 프로세스 관리
  - 메모리 관리
  - 저장장치 관리
  - 보안
  - 보호

운영체제의 목적

- 응용 프로그램들이 메모리, CPU, I/O 장치 등의 자원을 효율적으로 사용할 수 있도록 함



목차

- 운영체제의 정의
- 운영체제의 목적
- 운영체제의 역할
  - 사용자 인터페이스
  - 프로세스 관리
  - 메모리 관리
  - 저장장치 관리
  - 보안
  - 보호

운영체제의 역할–사용자 인터페이스

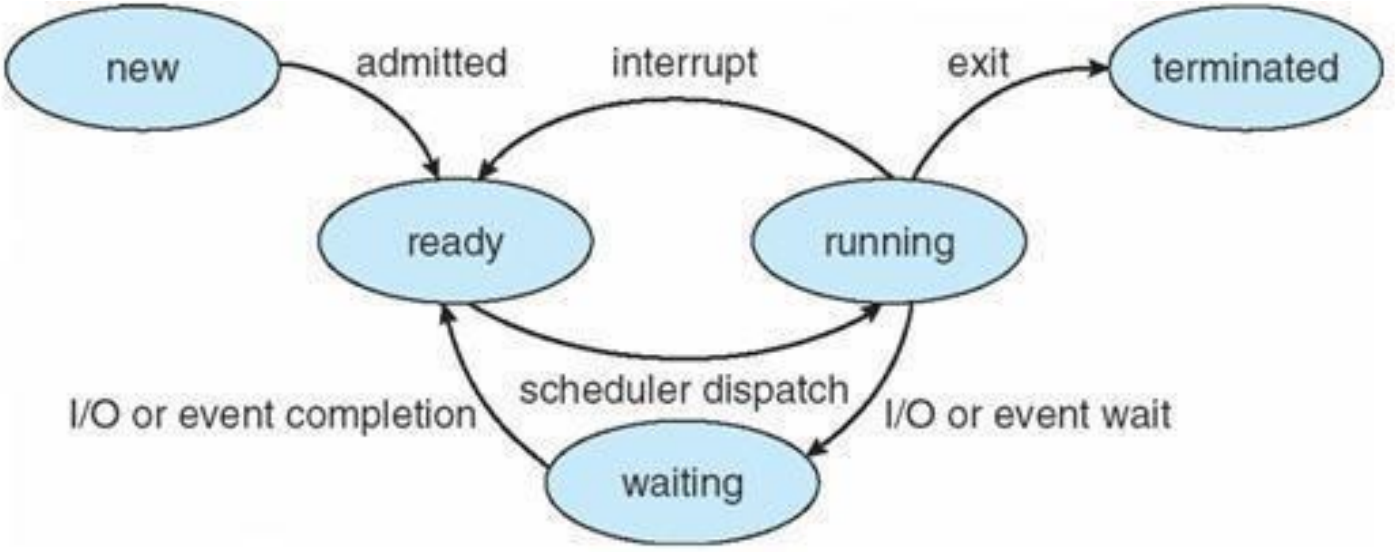
- 사용자 인터페이스(User Interface, UI)
  - 사용자가 컴퓨터와 편하게 의사소통을 할 목적으로 만들어진 매개체
  - e.g., GUI, CUI
    - GUI(Graphic User Interface)
      - 아이콘을선택하여 명령을 실행하는 방식
      - e.g., Windows
    - CUI(Character User Interface)
      - 사용자가 직접 명령을 입력하여 실행하는 방식
      - e.g., DOS, Unix/Linux

목차

- 운영체제의 정의
- 운영체제의 목적
- 운영체제의 역할
  - 사용자 인터페이스
  - 프로세스 관리
  - 메모리 관리
  - 저장장치 관리
  - 보안
  - 보호

운영체제의 역할 – 프로세스관리

- 프로세스
  - 메인 메모리에 적재되어 있는 프로그램
  - 다섯 가지 상태(new, ready, running, waiting, terminated)중 한가지 상태로 존재



운영체제의 역할 – 프로세스 관리

- 운영체제의 프로세스 관리
  - 프로그램의 실행을 관리
    - 프로세스간 통신(InterProcess Communication, IPC)
    - 프로세스 생성 및 제거, 동기화
    - 프로세스 중지와 재수행
    - 프로세스 스케줄링

운영체제의 역할 – 프로세스관리

- 프로세스간 통신 (IPC)
  - 프로세스간 데이터 공유와 교환을 뜻함
  - IPC 방법
    - PIPE
      - 두 개의 프로세스를 연결하는 pipe를 생성, 하나의 프로세스는 데이터를 쓰기만, 다른 하나는 읽기만가능
      - 읽기와 쓰기 둘 다하려면 두개의 pipe 생성해야함
    - 메시지 큐 (MessageQueue)
      - 데이터에 번호를 붙여 커널 영역에 저장 후 여러 개의 프로세스가 동시에 데이터를 다룰 수 있음

운영체제의 역할 – 프로세스관리

- 공유 메모리 (Shared Memory)
  - 공유 메모리를 커널에 요청해 메모리 공간을 할당 받고 데이터를 그 공간에 공유
  - 중개자가 없어 IPC중 가장 빠르게 작동
    - 중개자?
      - 프로세스들이 메시지를 주고 받는데 도움을 주는 것들
      - e.g., 커널
- 소켓 (Socket)
  - 컴퓨터끼리의 통신을 도와주는 방법, 네트워크방법
  - 원격에서 프로세스간 데이터 공유 시 사용
    - e.g., 네트워크

운영체제의 역할 – 프로세스관리

- 프로세스 생성 및제거, 동기화
  - 프로세스 생성의 종류
    - 사용자가 직접 프로그램실행
      - e.g., ./Hello, Hello
    - fork() 사용
      - 프로세스가 자신의 프로세스 복제하여 다른 프로세스 생성
      - e.g., 업무용 소프트웨어에서 출력 프로그램실행
      - 이때 생성된 프로세스가 자식 프로세스, 생성한 프로세스가 부모 프로세스임

운영체제의 역할 – 프로세스관리

- 프로세스의 제거
  - 프로세스에 할당된 모든 자원 회수, 디스크로 돌려보냄
  - 자식 프로세스가 종료를 위해 종료 상태를 부모 프로세스에게 보내고, 부모 프로세스는 이 정보를 받으면 자식 프로세스 제거

- 올바르지 못한 제거
  - 자식 프로세스의 종료 상태를 부모 프로세스가 얻어가지 않는 경우 좀비 프로세스 생성
    - 좀비 프로세스?
      - 실행이 종료되었지만, 제거되지 않은 프로세스
    - 이 경우 방지 위해 프로세스 동기화 필요
      - 프로세스 동기화?
        - 자식 프로세스가 완전히 종료되었는지 특정 함수로 확인

운영체제의 역할 – 프로세스관리

- 자식 프로세스보다 부모 프로세스가 먼저 종료된 경우 고아 프로세스 생성
  - 고아 프로세스?
    - 부모 프로세스가 자식 프로세스보다 먼저 종료 되었을 때의 자식 프로세스
  - 고아 프로세스는 init 프로세스의 자식 프로세스로 등록
    - init 프로세스?
      - 모든 프로세스의 부모 프로세스, 다른 모든 프로세스를 실행시키는 프로세스

운영체제의 역할 – 프로세스 관리

- 프로세스의 중지와 재수행
  - 목적
    - 전체적인 부하 감소, 시스템 자원 관리
  - 중지의 경우
    - 시스템 기능에 이상 발생
      - 프로세스 중단, 시스템 기능 정상회복후 프로세스 재수행
    - 의심되는 부분이 있는 프로세스
      - 사용자가 실행중인 프로세스 중지, 점검후 재수행 여부 결정
    - 시스템에 부하가 걸린 경우
      - 몇 개의 프로세스 중단 후 시스템 정상화 시 프로세스 재수행

운영체제의 역할 – 프로세스관리

- 프로세스 스케줄링
  - 준비(Ready)상태에 있는 프로세스들은 CPU할당을 받기 위해 레디 큐(Ready Queue)라는 공간에서 기다리고, 이 공간은 CPU 스케줄러에 의해관리됨
  - 운영체제가 여러가지 큐를 나누고 각 큐에 맞는 스케줄링 알고리즘 선택후 실행
  - 디스패처(Dispatcher)에 의해 CPU 할당
    - 디스패처?
      - 프로세스에게 CPU를 할당해주는 작업을 디스패치라고 하는데, 이 작업을 수행하는 모듈을 디스패처라고함

운영체제의 역할 – 프로세스관리

- 적용 시점에 따른 구분
  - 비선점형 스케줄링
    - 프로세스가 CPU를 할당 받으면 프로세스가 종료되거나 I/O이벤트로 자발적으로 중지 될 때까지 실행 보장
  - 선점형 스케줄링
    - 프로세스가 CPU를 할당 받아 실행중이어도 프로세스 중지시키고 CPU 강제 점유 가능

- 우선 순위 변동 여부에 따른 구분
  - 정적 스케줄링
    - 프로세스에 부여된 우선 순위 바뀌지 않음
  - 동적 스케줄링
    - 프로세스의 우선 순위 변동 가능

운영체제의 역할 – 프로세스관리

- CPU 스케줄러의 알고리즘
  - 우선 순위 스케줄링(Priority Scheduling)
    - 우선 순위가 높은 프로세스에게 CPU 먼저 할당
    - 시간 제한, 메모리 요구량, 프로세스의 중요성 등 판단해 우선순위 정의
    - 선점형, 비선점형 스케줄링
    - 단점
      - 한 프로세스보다 우선 순위가 높은 프로세스가 계속 들어오면 그 프로세스는 실행되지 않음

운영체제의 역할 – 프로세스관리

- 우선 순위 스케줄링(Priority Scheduling)

프로세스	버스트 시간	우선순위[1]	턴어라운드 시간	대기 시간
P1	10	3	16	6
P2	1	1	1	0
P3	2	4	18	16
P4	1	5	19	18
P5	5	2	6	1
평균	-	-	12	8.2

운영체제의 역할 – 프로세스관리

- FCFS 스케줄링(First-come, FirstServed, 선입선출)
  - CPU 요청 순서대로 할당
  - 비선점 스케줄링
  - 단점
    - CPU의 평균 대기시간 김

프로세스	버스트 시간	대기 시간	턴어라운드 시간
P2	3	0	3
P3	3	3	6
P1	24	6	30
평균	-	17	27

운영체제의 역할 – 프로세스관리

- SJF 스케줄링(Shortest-job-first, 최단 작업 우선)
  - 가장 짧은 CPU 할당 시간 가진 프로세스 처리
  - 선점, 비선점 스케줄링
  - 문제점
    - CPU할당 시간을 예측 불가, 다음 할당 시간이 이전 할당 시간과 비슷할거라고 예측

프로세스	버스트 시간	턴어라운드 시간	대기 시간
P1	**6**	**9**	**3**
P2	**8**	**24**	**16**
P3	**7**	**16**	**9**
P4	**3**	**3**	**0**
평균	**-**	**13**	**7**

운영체제의 역할 – 프로세스관리

- RR 스케줄링(Round-robin)
  - 일정 시간을 선정하여 CPU 할당을 모든 프로세스에게 동일 분담
  - 선점 스케줄링
  - 시간 할당량에 따라 알고리즘의 성능 좌우

프로세스	버스트 시간	턴어라운드 시간	대기 시간
P1	24	30	6
P2	3	7	4
P3	3	10	7
평균	-	15.67	5.67

운영체제의 역할 – 프로세스관리

- SRT 스케줄링(Shortest-remaining-time , 최단 잔여시간)
  - 진행 중인 프로세스 있어도 최단 잔여시간 프로세스에게 우선권 부여
  - 선점형 스케줄링

프로세스	도착시각	버스트 시간	종료시각	턴어라운드 시간	대기 시간
P1	00:00	8	00:17	17	9
P2	00:01	4	00:05	4	0
P3	00:02	9	00:26	24	15
P4	00:03	5	00:10	7	2
평균	-	-	-	13	6.5

목차

- 운영체제의 정의
- 운영체제의 목적
- 운영체제의 역할
  - 사용자 인터페이스
  - 프로세스 관리
  - 메모리 관리
  - 저장장치 관리
  - 보안
  - 보호

운영체제의 역할 – 메모리 관리

- 메모리 관리
  - 프로세스들을 위해 메모리 할당, 제거, 보호하는 활동

- 메모리 관리 기법
  - 반입 전략
    - 메인 메모리에 적재할 다음 프로세스의 반입시기를 결정
    - 요구 반입 전략, 예상 반입 전략이 있음
      - 요구 반입 전략
        - CPU가 요구하는 내용을 요구시 반입하는 방식
      - 예상 반입 전략
        - 운영체제가 스스로 예상하여 미리 반입하는 방식
        - 요구 반입 전략보다 시간은 덜 들지만, 공간 낭비 가능성 있음

운영체제의 역할 – 메모리 관리

- 메모리 관리 기법
  - 배치 전략
    - 반입한 프로세스를 메인 메모리 어느 위치에 저장할지 결정
    - 최초 적합, 최적 적합, 최악 적합 등이 있음
      - 최초 적합 (First-fit)
        - 메인 메모리 공간 중 프로그램을 저장할 수 있는 가장 첫번째의 공간에 할당
      - 최적 적합 (Best-fit)
        - 메인 메모리 공간 중 가장 알맞은 공간에 할당
      - 최악 적합 (Worst-fit)
        - 메인 메모리 공간 중 가장 큰 공간에 프로그램을 할당
  - 교체 전략
    - 메인 메모리의 모든 영역이 사용중인 상태에서 어느 영역을 삭제하고 교체할 것인지 결정

운영체제의 역할 – 메모리 관리

- 메모리 할당 방법

그림 7-8 메모리 할당 방법

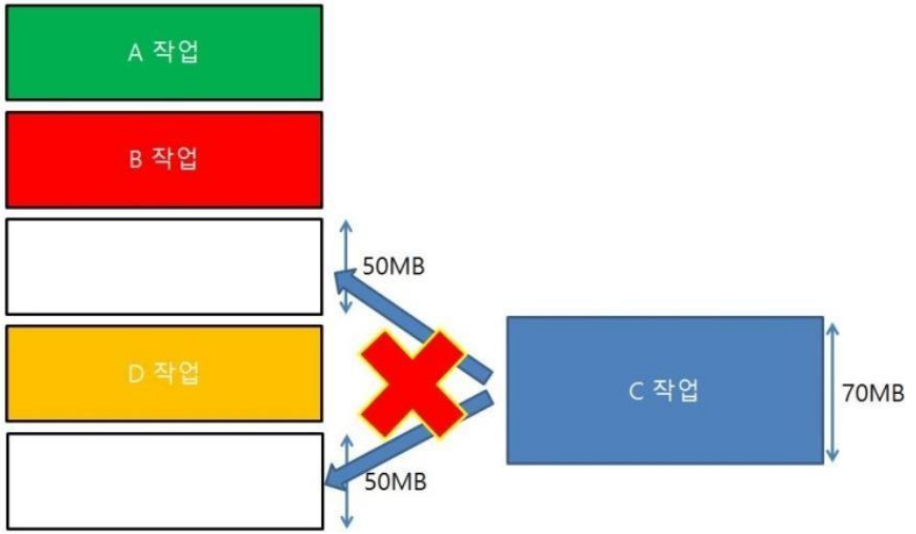
- 연속 메모리 할당
  - 프로그램을 메모리에 전체 올려 놓고 관리
- 불연속 메모리 할당
  - 프로그램을 특정단위 조각으로 나누어 분산하여 할당

운영체제의 역할 – 메모리 관리

- 메모리 할당 방법
  - 단일 프로그래밍
    - 1개의 프로그램만 주기억장치에 적재하여 실행
    - 프로그램 크기가 작으면 사용자 영역 낭비
  - 다중 프로그래밍
    - 여러개의 프로그램을 동시에 주기억장치에 적재하여 실행
    - 고정 분할, 동적분할로 나눌 수 있음
      - 고정 분할 방식(정적할당)
        - 주기억장치의 사용자 영역을 여러 개의 고정 크기로 분할
        - 내부 단편화가 발생할 수 있음
      - 동적 분할 방식(동적할당)
        - 프로그램을 주기억장치에 적재할 때 필요한 크기로 영역을 분할
        - 외부 단편화가 발생할 수 있음

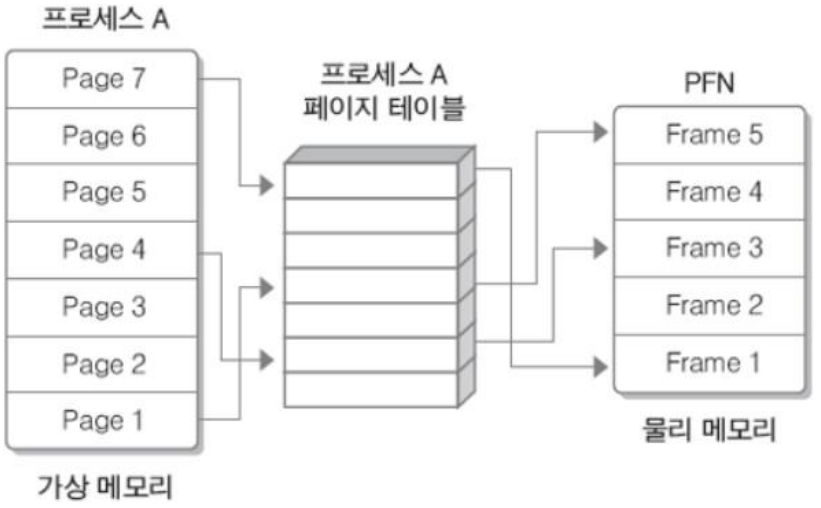
운영체제의 역할 – 메모리 관리

- 메모리 할당 방법
  - 내부 단편화
    - 프로세스가 필요한 양보다 더 큰 메모리 공간 할당
    - 프로세스에서 사용하는 메모리 공간이 낭비
  - 외부 단편화
    - 메모리 할당, 해제 과정에서 작은 메모리들이 존재 하게 됨
    - 총 메모리 공간은 충분하지만 실제로 할당할 수 없는 상황



운영체제의 역할 – 메모리 관리

- 메모리 할당 방법
  - 페이징(Paging)
    - 가상 메모리를 모두 같은 크기의 블록으로 분할하여 운용
    - 일정한 크기를 가진 블록을 페이지(Page)라고 함
    - 하나의 프로세스는 하나의 페이지 테이블을 가짐
      - 페이지 테이블
        - 프로세스의 페이지 정보를 저장하고 있는 테이블



운영체제의 역할 – 메모리 관리

- 메모리 할당 방법
  - 세그먼테이션(Segmentation)
    - 가상 메모리를 서로 크기가 다른 논리적 단위인 세그먼트(Segment)로 분할하고 메모리 할당
    - 페이징과 다르게 세그먼트들은 크기가 서로 달라 미리 분할해 두기 불가능
    - 메모리에 적재될 때 빈 공간을 찾아 할당하는 사용자 관점의 가상 메모리 관리 기법

목차

- 운영체제의 정의
- 운영체제의 목적
- 운영체제의 역할
  - 사용자 인터페이스
  - 프로세스 관리
  - 메모리 관리
  - 저장장치 관리
  - 보안
  - 보호

운영체제의 역할 – 저장장치관리

- 파일 시스템
  - 파일 시스템
    - 컴퓨터에서 파일을 기록하고 사용하는 모든 작업의 조직적인 체계
  - 파일 시스템 기능
    - 파일에 대한 접근 제어 방법 제공
    - 파일의 생성, 변경, 삭제 관리
    - 파일의 무결성과 보안 유지 방안 제공
    - 데이터의 백업 및 복구 기능 제공
    - 데이터의 효율적인 저장과 관리

운영체제의 역할 – 저장장치관리

- 파일 시스템 종류
  - FAT(File Allocation Table)
    - Windows NT가 지원하는 파일시스템중 가장 간단한 시스템
      - Windows NT
        - 마이크로소프트사의 서버급 운영체제
    - 연결 리스트를 이용한 자료구조로 인해 검색시간이 오래 걸림
    - 파일 데이터 블록들이 여기저기 흩어지는 단편화 현상 발생
  - HPFS(High Performance FileSystem)
    - IBM의 OS/2 1.2부터 사용된 파일 시스템
    - 제작 당시부터 대용량 디스크에 적합한 구조
    - FAT파일 시스템에 비해 파일 손실과 단편화가 적음
    - 대용량 저장 장치를 타겟으로 사용했기 때문에 200MB미만의 저장장치에서는 성능 저하

운영체제의 역할 – 저장장치관리

- 파일 시스템 종류
  - NTFS(New Technology FileSystem)
    - FAT와 HPFS에 있던 제약 사항들을 개선
    - Windows NT 및 2000 이상의 OS에서 대표적인 파일 시스템으로 자리잡아 서버 시스템은 물론 일반 PC 에서도 사용
  - UFS(Unix FileSystem)
    - 유닉스의 대표적인 파일시스템
    - 빠른 속도와 높은 안정성을 목표로 만들어짐
    - BSD계열(FreeBSD, NetBSD, OpenBSD 등)은 물론 HP-UX, Apple OS X 등 많은 유닉스 계열의 OS들이 각각 OS에 맞게 변형하여 사용

운영체제의 역할 – 저장장치관리

- 파일 시스템 종류
  - EXT2
    - UFS에 영향을 받은 리눅스 파일 시스템
    - UFS에서 유명무실한 구조들은 제거
    - 현재 배포되고 있는 모든 리눅스 배포판의 기본을 이룸
    - 파일 복구에 매우 강함

운영체제의 역할 – 저장장치 관리

- 대용량 저장장치 관리
  - 보조 저장장치
    - 메인 메모리를 초과하는 데이터 저장
    - 오랜 시간 보관되어야 하는 데이터 저장
    - 주로 디스크 장치가 사용됨
  - 대용량 저장장치 관리 기능
    - 자유 공간 관리(free space management)
    - 저장 공간 할당(storage allocation)
    - 디스크 스케줄링(disk scheduling)

운영체제의 역할 – 저장장치관리

- 입출력 시스템
  - 입출력 시스템
    - 컴퓨터에 연결된 다양한 입출력 하드웨어 장치들과 소통
  - 입출력 시스템 관리 기능
    - 버퍼링, 캐싱, 스풀링의 기능을 제공
      - 버퍼링(buffering)
        - 전송되는 동안 임시적으로 데이터를 저장
      - 캐싱(caching)
        - 성능을 위해 데이터 일부분을 빠른 저장장치에 저장
      - 스풀링(spooling)
        - 어떤 작업의 출력과 다른 작업의 입력이 겹치는 것
    - 일반적 장치 드라이버 인터페이스 제공
    - 특정 하드웨어 장치에 대한 드라이버 제공
      - 드라이버?
        - 특정 하드웨어나 장치를 제어하는 프로그램

목차

- 운영체제의 정의
- 운영체제의 목적
- 운영체제의 역할
  - 사용자 인터페이스
  - 프로세스 관리
  - 메모리 관리
  - 저장장치 관리
  - 보안
  - 보호

운영체제의 역할 – 보안

- 보안
  - 적절한 보호 시스템 뿐만 아니라, 시스템이 동작하는 외부 환경에 대해서도 고려
  - 외부 보안, 내부 보안, 사용자 인터페이스보안의 세가지로 구별
    - 외부 보안
      - 외부의 침입자나 천재지변으로부터 컴퓨터시스템보호
    - 내부 보안
      - 컴퓨터 하드웨어나 운영체제가 불법 침입자로부터 데이터를 보호하기위한 접근 제어 코드를 내장
        - 접근 제어 코드?
          - 누군가가 무언가를 사용하는 것을 허가하거나 거부하는 기능을 제공하는 코드
    - 사용자 인터페이스 보안
      - 사용자의 신원을 운영체제가 먼저 확인 후 시스템의 데이터 자료들을 접근할 수 있게 함

목차

- 운영체제의 정의
- 운영체제의 목적
- 운영체제의 역할
  - 사용자 인터페이스
  - 프로세스 관리
  - 메모리 관리
  - 저장장치 관리
  - 보안
  - 보호

운영체제의 역할 – 보호

- 시스템 보호
  - 프로세서, 기억장치, 파일 기타 자원에 대한 적절한 접근 권한을 부여

- 보호의 영역
  - 프로세스는 하나의 보호 영역을 갖고, 이 영역에서만 동작
  - 접근 권한
    - 어떤 프로세스가 객체에 대한 조작을 수행할 수 있는 능력
  - 접근 제어의 목적
    - 컴퓨터 시스템을 이루고 있는 컴퓨팅 자원, 정보 자원 등에 대해 허가 되지 않은 접근을 방해
    - 허가 받은 사람만 접근하기 위함

운영체제의 역할 – 보호

- 접근 행렬
  - 자원에 대한 접근 권한을 행렬로 표시
  - 행에는 사용자, 프로세스
  - 열에는 파일, 메모리, 프린터 등의 자원
  - 사용자는 자신의 객체에 대해 다른 사용자에게 접근 권한을 부여, 취소할 수 있음

객체 / 영역	F1	F2	F3	CD-ROM	프린터
D_1	읽기		읽기		
D_2				읽기	인쇄
D_3		읽기	실행		
D_4	읽기 쓰기		읽기 쓰기		

운영체제의 역할 – 보호

- 접근 행렬의 구현
  - 전역 테이블
    - 가장 단순한 형태
    - <영역, 객체, 권한 집합>들의 집합으로 구성

영역	객체	권한집합
D_1	F_1	READ
D_1	F_2	READ/WRITE
D_2	CD-ROM	READ
D_2	프린터	PRINT
D_3	F_2	READ
D_3	F_3	EXECUTE
.....	.....	.....
D_4	F_1	READ/WRITE

운영체제의 역할 – 보호

- 접근 행렬의 구현
  - 접근 제어 리스트
    - 파일에 접근 할 수 있는 사용자 리스트를 나타냄
    - 각 영역은 <사용자명, 사용자 그룹 명>의 쌍

종류	접근 제어 리스트
파일 0	(lee, 교수, rw-)
파일 1	(kim, 교수, rwx)
파일 2	(park, *, rw-)(choi, 대학원, r--)(kim, 학생, --x)
파일 3	(*, 학부, r-x)
파일 4	(*, *, ---), (*, 교수, r--)

운영체제의 역할 – 보호

- 접근 행렬의 구현
  - 권한 리스트
    - 접근 제어 행렬에 있는 각 행을 해당 영역과 결합한 것
    - 각 영역에 대한 권한 리스트는 그 자원에 허용된 조작 리스트로 구성

감사합니다!