응용 SW 기초 기술 활용

1. 운영체제 특징

(1) 운영체제

▼ 개념

- 사용자가 컴퓨터의 하드웨어를 쉽게 사용할 수 있도록 인터페이스를 제공해 주는 소프트웨어
- 한정된 시스템 자원을 효과적으로 사용할 수 있도록 관리 및 운영함으로써 사용자에게 편리성을 제공
- 컴퓨터 시스템과 사용자 간의 인터페이스 기능 담당

▼ 종류

- Windows
  - ▼ 특징
    - GUI 제공
    - 선점형 멀티태스킹 방식 제공 - 동시에 여러 개의 프로그램을 실행하면서 운영ㅊ체제가 각 작업의 CPU 이용 시간을 제어
    - Plug and Play 제공 - 하드웨어 설치 시 필요한 시스템 환경을 운영체제가 자동으로 구성
    - OLE (Object Linking and Embedding) 사용 - 개체를 현재 작성 중인 문서에 자유롭게 연결 또는 삽입하여 편집할 수 있게 하는 기능 제공
- Unix
  - ▼ 특징
    - 대화식 운영체제 기능 제공
    - 다중 작업 기능 제공 - 다수의 작업(프로세스)이 중앙 처리장치와 같은 공용자원을 나누어 사용하여 한 번에 하나 이상의 작업 수행
    - 다중 사용자 기능 제공 - 여러 사람이 동시에 시스템을 사용하여 각각의 작업을 수행할 수 있는 기능 제공
    - 이식성 제공 - 다른 하드웨어 기종으로 쉽게 이식 가능
    - 계층적 트리 구조 파일 시스템 제공 - 파일 관리 용이

  - ▼ 종류

- Linux

  데비안, 레드헷, Ubuntu, CentOS

- Mac
- Android

▼ 명령어

▼ 제어 방법

- CLI (Command Line Interface) - 사용자가 직접 명령어를 입력, 컴퓨터에 명령을 내리는 방식
- GUI (Graphic User Interface) - 그래픽 위주로 컴퓨터를 제어하는 방식

- 리눅스/유닉스

  ▼ 시스템 관련

  - cat - 파일의 내용을 화면에 출력

  ▼ 파일 처리

  - ls, pwd, rm, cp, mv

  ▼ 프로세스

  - ps - 현재 실행되고 있는 프로세스 목록 출력
  - pmap - 프로세스 ID를 기준으로 메모리 맵 정보를 출력
  - kill - 특정 PID 프로세스 종료

  ▼ 파일 권한

  - chmod - 특정 파일 또는 디렉토리의 퍼미션 수정 명령어
  - chown - 파일이나 디렉토리의 소유자, 소유 그룹 수정 명령어

  ▼ 네트워크

  - ifconfig - 네트워크 인터페이스를 설정하거나 확인
  - host - 도메인명은 알고 있는데 ip 주소를 모르거나 혹은 그반대 경우

  ▼ 압축

  - tar - 여러 개의 파일을 하나의 파일로 묶거나 풀 때 사용하는 명령어(압축은 불가)
  - gzip - 압축 담당

  ▼ 검색

  - grep - 입력으로 전달된 파일의 내용에서 특정 문자열을 찾고자할 때 사용

- find - 특정한 파일을 찾는 명령어

▼ 파일 이동

- cp - 디렉토리 복사
- rsync - 로컬 또는 원격에 파일과 디렉토리를 복사하고 동기화

▼ 핵심 기능

- 운영체제는 CPU, 메모리, 스토리지, 주변 기기 등을 적절히 관리
- 최근 들어 운영체제에서 대부분 자동으로 메모리 관리
- 개발 및 시스템 환경이 클라우드화되면서 자원에 대한 관리 노력이 줄음

▼ 메모리 관리

- 프로그램의 실행이 종료될 때까지 메모리를 가용한 상태로 유지 및 관리하는 기능
- 프로그램 실행 중 메모리가 꽉 차게 되면 시스템의 속도가 느려지고 때로는 시스템이 멈추는 현상 발생
- 메모리에 있는 프로그램은 CPU로 이동하여 처리(CPU는 가상주소를, 메모리는 물리주소를 사용하는데 MMU(Memory Management Unit - CPU가 메모리에 접근하는 것을 관리하는 하드웨어, 가상 메모리 주소를 실제 메모리 주소로 변환)가 주소를 매핑하는 역할 수행)

▼ 기법

- 반입 기법

주기억장치(RAM, ROM)에 적재할 다음 프로세스의 반입 시기를 결정, 메모리로 적재 시기 결정(when)

세부 기법 : 요구 반입 기법, 호출 반입 기법

- 배치 기법

디스크에 있는 프로세스를 주기억장치의 어느 위치에 저장할 것인지 결정, 메모리 적재 위치 결정(where)

세부 기법 : 최초 적합(First-fit), 최적 적합(Best-fit), 최악 적합(Worst-fit)

- 할당 기법

실행해야 할 프로세스를 주기억창치에 어떤 방법으로 할당할 것인지 결정, 메모리 적재 방법 결정(how)

세부 기법 : 연속 할당 기법, 분산 할당 기법

- 교체 기법

재배치 기법으로 주기억장치에 있는 프로세스 중 어떤 프로세스를 제거할 것인지를 결정, 메모리 교체 대상 결정(who)

세부 기법 : 프로세스의 Swap In/Out, FIFO, Optimal, LRU, LFU, MFU, 시계 알고리즘

▼ 프로세스 관리

- CPU와 데이터를 송수신하는 상황에서 현재 메모리 사용 부분, 메모리 입출력 순서, 메모리 공간 확보 등 프로세스에 대한 종합적인 관리 기법
- 프로세스(CPU에 의해 처리되는 프로그램) 관리 기법에는 '일시 중지 및 재실행', '동기화', '통신', '교착상태 처리', 프로세스 생성 삭제'가 있음

▼ 상태

- 생성 상태

사용자에 의해 프로세스가 생성된 상태

- 준비 상태

CPU를 할당받을 수 있는 상태

준비 리스트: 각각 우선순위를 부여하여 가장 높은 우선순위를 갖는 프로세스가 다음 순서에 CPU를 할당 받음

- 실행 상태

프로세스가 CPU를 할당 받아 동작 중인 상태

- 대기 상태

프로세스 실행 중 입출력 처리 등으로 인해 CPU를 양도하고 입출력 처리가 완료까지 대기 리스트에서 기다리는 상태

대기 리스트: 우선순위가 존재하지 않음

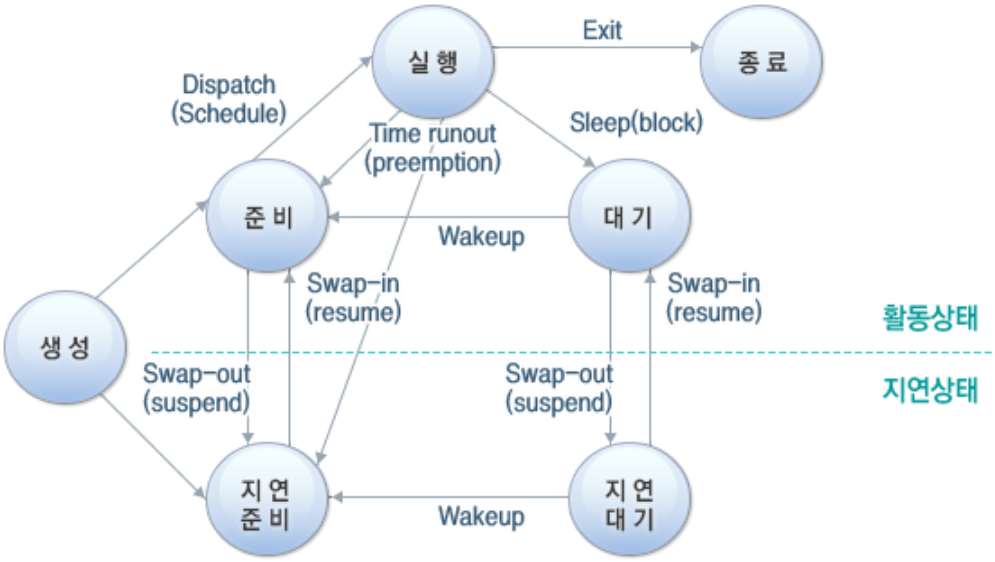
- 완료 상태

프로세스가 CPU를 할당 받아 주어진 시간 내에 완전히 수행을 종료한 상태

▼ 상태 전이

- 하나의 작업이 컴퓨터 시스템에 입력되어 완료되기까지 프로세스의 상태가 준비, 실행 및 대기 상태로 변하는 활동
- 활동 상태: 프로세스가 기억장치를 할당받은 상태

- 지연 상태: 프로세스가 기억장치를 할당받지 못한 상태



- 디스패치

준비 상태에 있는 여러 프로세스(Ready List) 중 실행될 프로세스를 선정(Scheduling)하여 CPU를 할당(Dispatching) → 문맥교환(CPU가 현재 실행하고 있는 프로세스의 문맥 상태를 프로세스 제어블록(PCB)에 저장하고 다음 프로세스의 PCB로 부터 문맥을 복원하는 작업) 발생

프로세스: 준비 상태 → 실행 상태로 전이

- 타이머 런 아웃

CPU를 할당 받은 프로세스는 지정된 시간이 초과되면 스케줄러에 의해 PCB저장, CPU 반납 후 다시 준비 상태로 전이

프로세스: 실행 상태 → 준비 상태

타임 슬라이스 만료, 선점 시 타임아웃 발생

- 블록

실행 상태에 있는 프로세스가 지정된 할당 시간을 초과하기 전에 입출력이나 기타 사건이 발생(Block)하면 CPU를 스스로 반납하고 입출력이 완료될 때까지 대기 상태로 전이

프로세스: 실행 상태 → 대기 상태

즉시 실행 불가능한 시스템 콜, I/O 작업 시작, 프로세스간 통신 시 Block 발생

- 웨이크 업

어느 순간에 입출력이 종료되면 대기 상태의 프로세스에게 입출력 종료 사실을 wait&signal 등에 의해 알려주고, 준비 상태로 전이

프로세스: 대기 상태 → 준비 상태

- Swap-in

프로세스에게 다시 기억장치가 할당될 경우

지연 준비 상태 or 지연 대기 상태 → 준비 상태 or 대기 상태

- Swap-out

프로세스가 기억장치를 잃은 경우

준비 상태 or 대기 상태 → 지연 준비 상태 or 지연 대기 상태

▼ 프로세스 스케줄링

▼ 개념

- CPU를 사용하려고 하는 프로세스들 사이의 우선순위를 관리하는 작업
- 처리율과 CPU 이용률을 증가시키고 오버헤드, 응답시간, 반환시간, 대기시간을 최소화 시키기 위한 기법
- 특정 프로세스가 적합하게 실행되도록 프로세스 스케줄링에 의해 프로세스 사이에서 CPU 교체가 일어남
- 프로세스 스케줄링을 실행하는 스케줄러의 유형에는 장기, 중기, 단기 스케줄러 존재

▼ 주요 용어

- 서비스 시간

프로세스가 결과를 산출하기까지 소요되는 시간

- 응답시간(반환시간)

프로세스들이 입력되어 수행하고 결과를 산출하기까지 소요되는 시간

응답시간 = 대기시간 + 수행시간

- 평균 응답 시간(평균 반환 시간)

대기 큐의 프로세스가 결과를 산출하기 소요되는 시간 평균

- 대기시간

프로세스가 프로세서에 할당 대기까지 큐에 대기하는 시간

- 평균 대기 시간

프로세스가 대기 큐에서 대기하는 평균 시간

- 종료 시간
- 시간 할당량

한 프로세스가 프로세서를 독점하는 것을 방지하기 위해 서비스되는 시간 할당량

- 응답률

(대기시간+서비스시간) / 서비스 시간

HRN(Highest Responce ratio Next) 스케줄링에서 사용

HRN 스케줄에서 응답률이 높으면 우선순위가 높다

▼ 유형

▼ 선점형 스케줄링 (Preemptive Scheduling)

- 개념

하나의 프로세스가 CPU를 차지하고 있을 때, 우선순위가 높은 다른 프로세스가 현재 프로세스를 중단시키고 CPU를 점유하는 스케줄링 방식

- 장점

비교적 빠른 응답

대화식 시분할 시스템에 적합

- 단점

높은 우선순위 프로세스들이 들어오는 경우 오버헤드 초래

- 활용

실시간 응답 환경, Deadline 응답 환경

▼ 알고리즘

▼ 라운드 로빈

- 동작 방식

프로세스는 같은 크기의 CPU 시간을 할당(시간 할당량), 프로세스가 할당된 시간 내에 처리 완료를 못하면 준비 큐 리스트의 가장 뒤로 보내지고, CPU는 대기 중인 다음 프로세스로 넘어감

- 특징

균등한 CPU 점유 시간

시분할 시스템(CPU 스케줄링과 다중 프로그래밍을 이용해서 각 사용자들에게 컴퓨터 자원을 시간적으로 분할하여 사용할 수 있게 해 주는 대화식 시스템) 사용

- 계산법

도착 시간, 서비스 시간 주어짐

1초마다 큐 상태, 프로세스 실행 상태 체크해서 계산

응답시간(반환시간) = 종료 시간 - 도착 시간

대기시간 = 응답 시간 - 서비스 시간

▼ SRT(Shortest Remaining Time First)

- 동작 방식

가장 짧은 시간이 소요되는 프로세스를 먼저 수행하고, 남은 처리 시간이 더 짧다고 판단되는 프로세스가 준비 큐에 생기면 언제라도 프로세스가 선점됨

- 특징

짧은 수행시간 프로세스 우선 수행

- 계산법

도착시간, 서비스 시간 주어짐

도착하는 프로세스와 기존의 프로세스 시간 남은 것 비교해서 계산

응답시간(반환시간) = 종료 시간 - 도착 시간

대기시간 = 응답 시간 - 서비스 시간

▼ 다단계 큐(Multi Level Queue)

- 동작 방식

작업들을 여러 종류 그룹으로 분할, 여러 개의 큐를 이용하여 상위 단계 작업에 의한 하위 단계 작업이 선점 당함

각 큐는 자신만의 독자적인 스케줄링 가짐

- 특징

독립된 스케줄링 큐

▼ 다단계 피드백 큐 (Multi Level Feedback Queue)

- 동작 방식

입출력 위주와 CPU 위주인 프로세스의 특성에 따라 큐마다 서로 다른 CPU 시간 할당량 부여

FCFS(FIFO)와 라운드 로빈 스케줄링 기법을 혼합, 새로운 프로세스는 높은 우선순위, 프로세스의 실행시간이 길어질수록 점점 낮은 우선순위 큐로 이동하고 마지막 단계는 라운드 로빈 방식 적용

- 특징

큐마다 다른 시간 할당량

마지막단계는 라운드 로빈 방식 처리

▼ 비선점형 스케줄링 (Non Preemptive Scheduling)

- 개념

한 프로세스가 CPU를 할당받으면 작업 종료 후 CPU 반환 시까지 다른 프로세스는 CPU 점유 불가능한 스케줄링 방식

- 장점

응답시간 예상이 용이

모든 프로세스에 대한 요구를 공정하게 처리

- 단점

짧은 작업을 수행하는 프로세스가 긴 작업 종료 시까지 대기

- 활용

처리시간 편차가 적은 특정 프로세스 환경

▼ 알고리즘

▼ 우선순위

- 동작 방식

프로세스 별로 우선순위가 주어지고, 우선순위에 따라 CPU를 할당

동일 순위는 FCFS

- 특징

주요/긴급 프로세스에 대한 우선 처리

설정, 자원 상황 등에 따른 우선순위 선정

▼ 기한부(Deadline)

- 동작 방식

작업들이 명시된 시간이나 기한 내에 완료되도록 계획

- 특징

요청에 명시된 시간 내 처리를 보장

▼ FCFS(First Come First Service)

- 동작 방식

프로세스가 대기 큐에 도착한 순서에 따라 CPU를 할당

FIFO

- 특징

도착한 순서대로 처리

- 계산법

도착시간, 서비스 시간, 종료 시간 주어짐

응답시간(반환시간) = 종료 시간 - 도착 시간

대기시간 = 응답 시간 - 서비스 시간

▼ SJF (Shortest Job First)

- 동작 방식

프로세스가 도착하는 시점에 따라 그 당시 가장 적은 서비스 시간을 갖는 프로세스가 종료 시 까지 자원 점유

준비 큐 작업 중 가장 짧은 작업부터 수행, 평균 대기 시간 최소

CPU 요구 시간이 긴 작업과 짧은 작업 간의 불평등이 심해, CPU 요구 시간이 긴 프로세스는 기아 현상 발생

- 특징

기아 현상 발생 가능성

- 계산법

우선순위가 높은 프로세스가 자원 선점 (서비스 시간 짧은 것)

도착시간, 서비스시간 주어짐

종료 시간 계산 (우선순위 나눈 후 계산)

응답시간(반환시간) = 종료 시간 - 도착 시간

대기시간 = 응답 시간 - 서비스 시간

▼ HRN (Highest Response Ratio Next)

- 동작 방식

대기 중인 프로세스 중 현재 응답률이 가장 높은 것 선택

SJF의 약점인 기아 현상을 보완한 기법으로 긴 작업과 짧은 작업 간의 불평등 완화

HRN 우선순위 = (대기시간+서비스시간)/서비스시간

- 특징

기아 현상(시스템 부하가 많아서 준비 큐에 있는 낮은 등급의 프로세스가 무한정 기다리는 현상 - 에이징 활용하여 해소) 최소화기법

(2) 가상화, 클라우드

1) 가상화

▼ 개념

- 물리적인 리소스들을 사용자에게 하나로 보이게 하거나, 하나의 물리적인 리소스를 여러 개로 보이게 하는 기술
- 가상화를 통해 서버의 가동률을 60~70% 이상 올릴 수 있다

▼ 종류

- 플랫폼 가상화

하드웨어 플랫폼 위에서 실행되는 호스트 프로그램이 게스트 프로그램을 만들어 마치 독립된 환경을 만들어 낸 것 처럼 보여주는 기법

- 리소스 가상화

게스트 소프트웨어 위에서 사용자는 독립된 하드웨어에서 소프트웨어가 실행되는 것 처럼 활용하는 기법

메모리, 저장 장치, 네트워크 등을 결합하거나 나누기 때문에 사용자는 가상화 된 물리적 장치들이 어떤 위치에 있는지 알기 어려움

▼ 기술 요소

- 컴퓨팅 가상화

물리적으로 컴퓨터 리소스를 가상화하여 논리적 단위로 리소스를 활용할 수 있도록 하는 기술

서버 가상화를 통해 하나의 시스템에서 1개 이상의 운영체제를 동시에 가동시킬 수 있으므로, 서버 이용률이 크게 향상

→ 하이퍼 바이저

- 스토리지 가상화

스토리지와 서버 사이에 소프트웨어/하드웨어 계층을 추가하여 스토리지를 논리적으로 제어 및 활용할 수 있도록 하는 기술

이기종 스토리지 시스템의 통합을 가능하게 하는 기술

→ 분산 파일 시스템

- I/O 가상화

서버와 I/O 디바이스 사이에 위치하는 미들웨어 계층, 서버의 I/O자원을 물리적으로 분리하고 케이블과 스위치 구성을 단순화하여 효율적인 연결 지원

→ 가상 네트워크 인터페이스 카드

- 컨테이너

컨테이너화된 애플리케이션들이 단일 운영체제상에서 실행되도록 해주는 기술

하이퍼바이저 없이 운영체제가 격리된 프로세스로 동작하기 때문에 오버헤드가 낮음

→ 도커

- 분산처리 기술

여러대의 컴퓨터 계산 및 저장 능력을 이용하여 커다란 계산 문제나 대용량의 데이터를 처리하고 저장하는 기술

- 네트워크 가상화 기술

물리적으로 떨어져 있는 다양한 장비들을 연결하기 위한 수단으로 중계장치(라우터, 스위치 등)의 가상화를 통한 가상 네트워크를 지원하는 기술

→ SDN, NFV

2) 클라우드 컴퓨팅

▼ 개념

- 인터넷을 통해 가상화된 컴퓨터 시스템 리소스를 제공하고, 정보를 자신의 컴퓨터가 아닌 클라우드에 연결된 다른 컴퓨터로 처리하는 기술
- 구성 가능한 컴퓨팅 자원(컴퓨터 네트워크, 데이터베이스, 서버, 스토리지, 애플리케이션, 서비스)에 대해 어디서나 접근 가능

▼ 분류

- 사설 클라우드

기업 또는 조직 내부에서 보유하고 있는 컴퓨팅 자원을 사용하여 내부에 구축되어 운영되는 클라우드

자체 컴퓨팅 자원으로 모든 하드웨어, 소프트웨어, 데이터 수용

직접적인 통제가 가능하며 보안성 높일 수 있음

- 공용 클라우드

클라우드 서비스 제공 업체에서 다중 사용자를 위한 컴퓨팅 자원 서비스를 제공하는 클라우드

일정한 비용을 지불하고 하드웨어, 소프트웨어 사용

확장성, 유연성 등이 뛰어남

- 하이브리드 클라우드

사설 클라우드 + 공용 클라우드

사설 클라우드의 약점인 구축 비용 문제와 공용 클라우드 약점인 보안성 확보 문제 해결

사용 업무의 중요도, 보안성 확보의 중요도 등에 따라 이용 형태 변경 가능

▼ 유형

- 인프라형 서비스(IaaS; Infrastructure as a Service)

서버, 스토리지 같은 시스템 자원을 클라우드로 제공하는 서비스

컴퓨팅 자원에 운영체제나 애플리케이션 등의 소프트웨어 탑재 및 실행

하위의 클라우드 인프라를 제어하거나 관리하지 않지만 스토리지, 애플리케이션에 대해서는 제어권 가짐

- 플랫폼형 서비스 (PaaS; Platform as a Service)

인프라를 생성, 관리 하는 복잡함 없이 애플리케이션을 개발, 실행, 관리할 수 있게 하는 플랫폼을 제공하는 서비스

SaaS 개념을 개발 플랫폼에도 확장한 방식, 개발을 위한 플랫폼을 구축할 필요 없이, 필요한 개발 요소를 웹에서 빌려 사용하는 모델

OS, 애플리케이션과 애플리케이션 호스팅 환경 구성의 제어권 가짐

- 소프트웨어형 서비스 (SaaS; Software as a Service)

소프트웨어 및 관련 데이터는 중앙에 호스팅되고 사용자는 웹 브라우저 등의 클라이언트를 통해 접속하여 소프트웨어를 서비스 형태로 이용하는 서비스

주문형 소프트웨어

2. 데이터베이스 기초 활용

(1) 데이터베이스

▼ 개념

- 다수의 인원, 시스템 또는 프로그램이 사용할 목적으로 통합하여 관리되는 데이터의 집합
- 데이터에 대한 효과적인 관리를 위해 **자료의 중복성 제거, 무결성 확보, 일관성 유지, 유용성 보장이 중요**

▼ 종류

▼ 파일 시스템

- 데이터베이스 전 단계의 데이터 관리 방식.
- 파일에 이름 부여, 저장이나 검색 위해 논리적으로 위치시킬 곳 정의한 뒤 관리
- ISAM (Indexed Sequential Access Method)
- VSAM (Virtual Storage Access Method

▼ 계층형 데이터베이스 관리시스템 (HDBMS)

- 데이터를 상하 종속적인 관계로 계층화하여 관리
- 데이터 접근 속도 빠름, 종속적인 구조로 변화하는 데이터 구조에 유연한 대응 어려움
- IMS (Information Management System)

▼ 망형 데이터베이스 관리시스템 (NDBMS)

- 데이터 구조를 네트워크상의 망상 형태로 논리적으로 표현한 데이터 모델
- 트리 구조나 계층형 데이터베이스 보다 유연, 설계 복잡

▼ 관계형 데이터베이스 관리시스템 (RDBMS)

- 관계형 모델 기반
- 데이터를 저장하는 테이블의 일부를 다른 테이블과 상하 관계로 표시하며 상관 관계를 정리
- 변화하는 업무나 데이터 구조에 대한 유연성이 좋아 유지 관리가 용이

- Oracle, SQL Server, MySQL, Maria DB, MS-SQL

▼ 관리 툴

- DB 관리자(DBA)들이 데이터베이스를 편리하고 쉽게 다룰 수 있도록 도와주는 도구

▼ 기능

- 데이터베이스 생성, 삭제
- SQL 명령어 작성 및 실행
- 상태 모니터링 - 받은 데이터양, 보낸 데이터양, 동시 연결 수, 실패한 시도 등의 상태 표시
- 사용자 계정 관리 - 최상위 SYS 계정, SYS로 부터 권한 받은 SYSTEM 계정, 일반 사용자 계정
- 데이터베이스 내보내기/가져오기
- 환경 설정 - 버퍼 크기, 동시 접속 클라이언트 숫자, 스레드 숫자 등의 환경 변수 설정

▼ DBMS

- 데이터 관리 복잡성 해결, 데이터 추가, 변경, 검색, 삭제 및 백업, 복구, 보안 등의 기능을 지원하는 소프트웨어

▼ 유형

- Key-Value DBMS

키 기반 GET, PUT, DELETE 제공, 메모리 기반에서 성능 우선 시스템 및 빅데이터 처리 가능 DBMS

Unique 한 키에 하나의 값을 가지고 있는 형태

Redis, DynamoDB

- Column Family Data Store DBMS (컬럼 기반 데이터 저장)

Key 안에 (Column, Value) 조합으로 된 여러 개의 필드를 갖는 DBMS

테이블 기반, 조인 미지원, 컬럼 기반으로 구글의 Blgtable 기반으로 구현

HBase, Cassandra

- 문서 저장 DBMS

값의 데이터 타입이 문서라는 타입을 사용하는 DBMS

문서 타입은 XML, JSON과 같이 구조화된 데이터 타입으로, 복잡한 계층 구조 표현 가능

MongoDB, Couchbase

- 그래프 DBMS

시맨틱 웹과 온톨로지 분야에서활용되는 그래프로 데이터를 표현하는 DBMS

노드와 엣지로 특징되는 요소 특화

노드 간 관계를 구조화하여 저장

Neo4j, AllegroGraph

▼ 특징

- 데이터 무결성

부적절한 자료가 입력되어 동일한 내용에 대하여 서로 다른 데이터가 저장되는 것을 허용하지 않는 성질

- 데이터 일관성

삽입, 삭제, 갱신, 생성 후에도 저장된 데이터가 변함없이 일정

- 데이터 회복성

장애가 발생했을 시 특정 상태로 복구되어야 하는 성질

- 데이터 보안성

불법적인 노출, 변경, 손실로 부터 보호되어야 하는 성질

- 데이터 효율성

응답 시간, 저장 공간 활용 등이 최적화되어 사용자, 소프트웨어, 시스템 등의 요구 조건을 만족시켜야 하는 성질

(2) 관계형 데이터베이스 활용

▼ ERD 개념 (ER-Diagram)

- ERD는 업무 분석 결과로 도출된 실체(엔티티)와 엔티티 간의 관계를 도식화한 다이어그램
- ERD로 요소들 간 연관성을 도식화하여 데이터베이스 관리자, 개발자, 사용자 모두 데이터의 흐름과 연관성을 공통적으로 쉽게 확인 가능

▼ ER 모델

- ERD의 구성 요소인 개체, 관계, 속성을 추출하기 위해 업무나 시스템에 대한 명확한 정의가 있어야 함
- ERD로 도식화하기 전 각 개체를 사각형, 화살표, 마름모로 표기한 형태를 ER 모델

▼ 구성

- 엔티티(Entity)

사물 또는 사건으로 정의되며 개체라 부름

Peter Chen Model - 사각형, Crow's Foot Model - 표 형식

- 속성(Attribute)

엔티티가 가진 요소 또는 성질

단수형으로 명명

엔티티명 사용X

속성 Null, Not Null 고려하여 작성

Peter Chen Model - O, Crow's Foot Model - 표 내부에 표시

- 관계

두 엔티티 간의 관계

1:1, 1:m, n:m

Peter Chen Model - 마름모, Crow's Foot Model - ——, ——-<, >——<

(3) 데이터베이스 관리

▼ 트랜잭션

▼ 개념

- 인가 받지 않은 사용자로 부터 데이터를 보장하기 위해 DBMS가 가져야 하는 특성, 데이터베이스 시스템에서 하나의 논리적 기능을 정상적으로 수행하기 위한 작업의 기본 단위

▼ 특성

- 원자성(Atomicity)

분해가 불가능한 작업의 최소 단위

연산 전체가 성공 또는 실패

하나라도 실패 할 경우 전체가 취소

Commit/Rollback, 회복성 보장

- 일관성(Consistency)

트랜잭션이 실행 성공 후 항상 일관된 데이터베이스 상태를 보존해야 함

무결성 제약 조건, 동시성 제어

- 격리성(Isolation)

트랜잭션 실행 중 생성하는 연산의 중간 결과를 다른 트랜잭션이 접근 불가

Read Uncommit, Read Commit, Phantom Read, Serializable

- 영속성(Durability)

성공이 완료된 트랜잭션의 결과는 영속적으로 데이터베이스에 저장

회복 기법

▼ 상태 변화

- 활동 상태 (Active)

초기 상태, 트랜잭션이 실행 중일 때 가지는 상태

- 부분 완료 상태 (Partially Committed)

마지막 명령문이 실행된 후에 가지는 상태

- 완료 상태 (Committed)

트랜잭션이 성공적으로 완료된 후 가지는 상태

- 실패 상태(Failed)

정상적인 실행이 더 이상 진행될 수 없을 때 가지는 상태

- 철회 상태(Aborted)

트랜잭션이 취소되고 데이터 베이스가 트랜잭션 시작 전 상태로 환원된 상태

부분 완료 →(commit) 완료

실패 →(rollback) 철회

▼ 제어 (TCL - Transaction Control Language)

- COMMIT

트랜잭션 확장

트랜잭션을 메모리에 영구적으로 저장

- ROLLBACK

트랜잭션 취소

트랜잭션 내역을 저장 무효화시킴

- CHECKPOINT

저장 시기 설정

ROLLBACK을 위한 시점을 지정

▼ 기본 연산

- CRUD

Create - 삽입 - INSERT

Read - 조회 - SELECT

Update - 갱신 - UPDATE

Delete - 삭제 - DELETE

▼ 수행 상세

- SELECT

```
-- ALL : 모든 튜플 검색, 명시 안하면 자동으로 ALL
-- DISTINCT : 중복된 속성이 조회될 경우 그 중 한 개만 검색(SELECT 뒤에 명시된 속성이 중복될 경우 한 개만 검색_
-- DISTINNCTROW : SELECT 뒤에 속성들과 상관없이 튜플 전체가 중복된 튜플 제거
SELECT [ALL | DISTINCT | DISTINCTROW] 속성1, 속성2, ...
  FROM 테이블명1, ...
[WHERE 조건]
[GROUP BY 속성1, 속성2, ...]
[HAVING 그룹조건]
[ORDER BY 속성 [ASC|DESC];
```

▼ WHERE 조건절

- 비교 - = <>(다름) < ≤ > ≥
- 범위 - BETWEEN
- 집합 - IN, NOT IN
- 패턴 - LIKE (NAME LIKE '정보%' - 이름이 '정보'로 시작되는 문자열

와일드 문자

+: 문자열 연결

%: 0개 이상의 문자열과 일치

[ ]: 1개의 문자와 일치 → [0-8]% : 0~8 사이의 숫자로 시작하는 문자열

[^]: 1개의 문자와 불일치 → [^0-8]%: 0~8 사이의 숫자로 시작하지 않는 문자열

_: 특정 위치의 1개의 문자와 일치 → _동%: 두 번째 위치에 '동'이 들어가는 문자열

- NULL - IS NULL, IS NOT NULL

- 복합 조건 - AND, OR, NOT

```
SELECT 직책, COUNT(직책), SUM(급여)
  FROM 급여
GROUP BY 직책;

SELECT 부서, SUM(급여)
  FROM 급여
GROUP BY 부서;

SELECT 직책, 부서, SUM(급여)
  FROM 급여
GROUP BY 직책, 부서;

-- GROUP BY 절이 없을 경우 전체 테이블이 하나의 그룹이 되고, 그룹에 해당하는 총 튜플의 수 출력
SELECT COUNT(*)
  FROM 급여
```

- INSERT

```
-- 속성과 데이터 개수, 타입이 일치해야 함, 속성명 생략 가능
INSERT INTO 테이블명 (속성1, ...)
VALUES (데이터1, ...)
```

- UPDATE

```
UPDATE 테이블명
  SET 속성명=데이터, ...
WHERE 조건;
```

- DELETE

```
DELETE FROM 테이블명
  WHERE 조건;
```

(4) 기술 트렌드

1) 빅 데이터

- 수십 페타바이트(PB) 크기의 비정형 데이터

▼ 특성

- 데이터 양(Volume)

페타바이트(10**15) 수준

분석 규모에 관련된 특성

디지털 정보량이 기하 급수적으로 폭증

- 데이터의 다양성(Variety)

정형, 비정형, 반정형의 다양한 데이터

자원 유형에 관련된 특성

로그, 소셜, 위치 등 데이터 유형이 다양해지는 것을 의미

- 데이터 속도(Velocity)

빠르게 증가하고 수집되며, 처리되는 데이터

수집, 분석, 활용 속도와 관련된 특성

가치 있는 정보 활용을 위해 실시간 분석이 중요해지는 것

▼ 수집, 저장, 처리 기술

- 비정형/반정형 데이터 수집

내외부 정제되지 않은 데이터를 확보, 이를 통해 필요 정보를 추출하여 활용하기 위해 효과적으로 수집 및 전송하는 기술

척와(Chukwa), 플럼(Flume), 스크라이브(Scribe)

- 정형 데이터 수집

내외부 정제된 대용량 데이터의 수집 및 전송 기술

ETL, FTP, 스쿱(Sqoop), 하이호(Hiho)

- 분산데이터 저장/처리

대용량 파일의 효과적인 분산 저장 및 분산 처리 기술

HDFS(Haddop Distributed File System) - 대용량 데이터의 집합을 처리하는 응용 프로그램에 적합하도록 설계된 하둡 분산 파일 시스템

Map Reduce - 구글에서 대용량 데이터 처리를 분산 병렬 컴퓨팅에서 처리하기 위한 목적으로 제작하여 발표한 소프트웨어 프레임워크

- 분산데이터 베이스

HDFS의 칼럼 기반 데이터베이스로 실시간 랜덤 조회 및 업데이트가 가능한 기술

HBASE

▼ 분석, 실시간 처리 및 시각화 위한 주요 기술

- 빅 데이터 분석

빅데이터를 분석하기 위한 데이터의 가공과 분류, 클러스터링, 패턴 분석을 처리하는 기술

데이터 가공을 위한 대표적인 솔루션에는 Pig, Hive가 있고, 데이터 마이닝을 위한 대표적 솔루션으로 Mahout가 있음

- 빅 데이터 실시간 처리

하둡 기반의 실시간 SQL 질의 처리와 요청된 작업을 최적화하기 위한 워크플로우 관리 기술

실시간 SQL 질의를 위한 대표적인 솔루션은 Impala

워크플로우 관리 위한 솔루션 Oozie

- 분산 코디네이션

분산 환경에서 서버들 간에 상호 조정이 필요한 다양한 서비스를 분산 및 동시 처리 제공 기술

대표 솔루션: Zookeeper

- 분석 및 시각화

빅데이터 분석 기술을 통해 분석된 데이터의 의미와 가치를 시각적으로 표현하기 위한 기술

대표 솔루션: R

2) NoSQL

▼ 개념

- 데이터 저장에 고정된 테이블 스키마가 필요하지 않고 조인 연산을 사용할 수 없으며, 수평적으로 확장이 가능한 DBMS

▼ 특성(BASE)

- Basically Available

언제든지 데이터는 접근할 수 있어야 하는 속성

분산 시스템이기 때문에 항상 가용성 중시

- Soft-State

노드의 상태는 내부에 포함된 정보에 의해 결정되는 것이 아니라 외부에서 전송된 정보를 통해 결정되는 속성

특정 시점에서는 데이터의 일관성이 보장되지 않음

- Eventually Consistency

일정 시간이 지나면 데이터의 일관성이 유지되는 속성

일관성을 중시하고 지향

▼ 유형

- Key-Value DBMS

키 기반 GET, PUT, DELETE 제공, 메모리 기반에서 성능 우선 시스템 및 빅데이터 처리 가능 DBMS

Unique 한 키에 하나의 값을 가지고 있는 형태

Redis, DynamoDB

- Column Family Data Store DBMS (컬럼 기반 데이터 저장)

Key 안에 (Column, Value) 조합으로 된 여러 개의 필드를 갖는 DBMS

테이블 기반, 조인 미지원, 컬럼 기반으로 구글의 Bigtable 기반으로 구현

HBase, Cassandra

- 문서 저장 DBMS (Document Store)

값의 데이터 타입이 문서라는 타입을 사용하는 DBMS

문서 타입은 XML, JSON과 같이 구조화된 데이터 타입으로, 복잡한 계층 구조 표현 가능

MongoDB, Couchbase

- 그래프 DBMS

시맨틱 웹과 온톨로지 분야에서활용되는 그래프로 데이터를 표현하는 DBMS

노드와 엣지로 특징되는 요소 특화

노드 간 관계를 구조화하여 저장

Neo4j, AllegroGraph

3) 데이터마이닝

▼ 개념

- 대규모로 저장된 데이터 안에서 체계적이고 자동적으로 통계적 규칙이나 패턴을 찾아내는 기술
- 대규모 데이터에서 의미 있는 패턴을 파악하거나 예측하여 의사결정에 활용하는 기법
- 데이터의 숨겨진 정보를 찾아내어 이를 기반으로 서비스와 제품에 도입하는 과정

- 통계 분석은 가설이나 가정에 따른 분석, 검증을 하지만 데이터 마이닝은 수리 알고리즘을 활용하여 대규모 데이터에서 의미있는 정보 찾아냄

▼ 절차

1. 목적 설정 - 목적에 따라 사용할 모델과 필요 데이터 정의
2. 데이터 준비 - 운영 데이터 접근에 따른 부하 고려, 데이터 정제를 통해 데이터의 품질을 보장하고, 필요 시 데이터 추가 등을 통해 충분한 양의 데이터 확보
3. 가공 - 모델링 목적에 따라 목적 변수 정의
4. 마이닝 기법 적용
5. 정보 검증 - 테스트 데이터와 과거 데이터를 활용하여 최적의 모델 선정

▼ 주요 기법

- 텍스트 마이닝

대량의 텍스트 데이터로부터 패턴 또는 관계를 추출하여 의미 있는 정보를 찾아내는 기법

비정형 반정형 데이터에 대하여 자연어 문서 처리 기술을 적용하여 의미 있는 정보를 추출

- 웹 마이닝

웹으로 부터 얻어지는 방대한 양의 정보로 부터 유용한 정보를 찾아내기 위해 분석하는 기법

웹 자원으로 부터 의미 있는 패턴, 프로 파일, 추세 등을 발견하기 위해 데이터 마이닝 기술을 응용

- 분류 규칙(Classsification)

과거 데이터로 부터 특성을 찾아내어 분류 모형을 만들어 이를 토대로 새로운 레코드의 결과 값을 예측하는 기법

마케팅, 고객 신용 평가 모형에 활용

→ 우수 고객의 분류 모형 구축으로 마케팅 활용

- 연관 규칙(Association)

데이터 안에 존재하는 항목들 간의 종속 관계를 찾아내는 기법

제품이나 서비스의 교차 판매, 매장 진열, 사기 적발 등 다양한 분야에 활용

→ 넥타이 구매 고객의 50% 이상이 셔츠 구매, 매장의 상품 진열 변경

- 연속 규칙 (Sequence)

연관 규칙에 시간 관련 정보가 포함된 형태의 기법

개인별 트랜잭션 이력 데이터를 시계열적으로 분석하여 트랜잭션의 향후 발생 가능성 예측

→ A 품목을 구매한 회원이 B 품목을 구매할 확률은 75%

- 데이터 군집화(Clustering)

대상 레코드들을 유사한 특성을 지닌 몇 개의 소 그룹으로 분할하는 작업으로 작업의 특성이 분류 규칙과 유사

정보가 없는 상태에서 데이터를 분류하는 기법

분석 대상에 결과 값이 없으며, 판촉 활동이나 이벤트 대상을 선정하는 데 활용

→ 고객의 지역/연령/성별에 따른 차별화 홍보 전략

3. 네트워크 기초 활용

(1) 네트워크

▼ 개념

- 원하는 정보를 원하는 수신자 또는 기기에 정확하게 전송하기 위한 기반 인프라

▼ 분류

- WAN (광대역 네트워크)

LAN에 비해 전송 거리가 넓음, 라우팅 알고리즘 필요

LAN 대비 에러율이 높고 전송 지연이 큼

- LAN (근거리 네트워크)

한 건물 또는 작은 지역을 커버하는 네트워크

▼ OSI 7 Layer

- 계층을 지날 때마다 헤더가 붙는데, 해당 계층의 기능과 관련된 제어 정보가 포함
- 제어 정보들은 모두 운영체제가 제공하는 프로토콜에 의해 송신 측에서는 계층을 지날 때마다 덧붙여서 추가되고, 수신 측에서는 계층을 지날 때마다 제거

▼ 물리 계층

- 실제 장비들을 연결하기 위한 연결 장치
- 허브: **여러 대의 컴퓨터를 연결**하여 네트워크로 보내거나, 하나의 네트워크로 수신된 정보를 여러 대의 컴퓨터로 **송신**하기 위한 장비

- 리피터: **디지털 신호를 증폭**시켜 주는 역할을 하여 신호가 약해지지 않고 컴퓨터로 수신되도록 하는 장비

▼ 데이터 링크 계층

- 오류와 흐름을 제거하여 신뢰성 있는 데이터를 전송
- 브리지: 두 개의 근거리 통신망을 서로 연결해 주는 통신망 연결 장치
- L2 스위치: 느린 전송 속도 브리지, 허브 단점 개선위해 출발지에서 들어온 프레임을 목적지 MAC 주소 기반으로 빠르게 전송시키는 데이터 링크 계층의 통신 장치
- NIC(Network Interface Card) : 외부 네트워크와 접속하여 가장 빠른 속도로 데이터를 주고 받을 수 있게 컴퓨터 내에 설치 되는 장치

▼ 네트워크 계층

- 다수의 중개 시스템 중 올바른 경로를 선택하도록 지원
- 라우터

LAN과 LAN 연결, LAN과 WAN 연결하기 위한 네트워킹 장비

패킷의 위치를 추출하여, 그 위치에 대한 최적의 경로를 지정하며, 이 경로를 따라 데이터 패킷을 다음 장치로 전송

라우팅 프로토콜은 경로 설정을 하여 원하는 목적지까지 지정된 데이터가 안전하게 전달되도록 함

- 게이트웨이

프로토콜을 서로 다른 통신망에 접속할 수 있게 해주는 장치

LAN에서 다른 네트워크에 데이터를 보내거나 다른 네트워크로 부터 데이터를 받아 들이는 출입구 역할

▼ 전송 계층

- 송신, 수신 프로세스 간의 연결
- L4 스위치

▼ 세션 계층

- 송신, 수신 간의 논리적 연결
- 호스트(PC)

▼ 표현 계층

- 코드 문자 등을 번역하여 일관되게 전송하고 압축, 해제, 보안 기능도 담당
- 호스트(PC)

▼ 응용 계층

- 사용자 친화 환경 제공(이메일, 웹 등)

- 호스트(PC)

(2) 네트워크 프로토콜

1) 프로토콜

▼ 개념

- 서로 다른 시스템이나 기기들 간의 데이터 교환을 원활히 하기 위한 표준화 된 통신 규약
- 데이터 처리 기능, 제어 기능, 관리적 기능 필요

▼ 기본 요소

- 구문(Syntax) - 시스템 간의 정보 전송을 위한 **데이터 형식, 코딩, 신호 레벨 등의 규정**
- 의미(Semantic) - 시스템 간의 정보 전송을 위한 제어 정보로 **조정과 에러 처리**를 위한 규정
- 타이밍(Timing) - 시스템 간의 정보 전송을 위한 **속도 조절과 순서 관리 규정**

2) 네트워크 프로토콜

- 컴퓨터나 원거리 통신 장비 사이에서 메시지를 주고 받는 양식과 규칙의 체계
- 전달 방식, 통신 방식, 자료 형식, 오류 검증 방식, 코드 변환 규칙, 전송 속도 등을 정함
- 다른 기기 간 정보의 전달을 표준화 가능

▼ 특징

- 단편화 - 전송이 가능한 작은 블록으로 나누어지는 기법
- 재조립 - 단편화된 데이터를 원래 데이터로 복원
- 캡슐화 - 상위 계층의 데이터에 각종 정보를 추가하여 하위 계층으로 보내는 기법
- 연결 제어 - 데이터 전송량이나 속도 제어
- 오류 제어 - 전송 중 잃어버리는 데이터나 오류가 발생한 데이터를 검증
- 동기화 - 송신과 수신 측의 시점을 맞추는 기법
- 다중화 - 하나의 통신 회선에 여러 기기들이 접속
- 주소 지정 - 송신과 수신지의 주소를 부여하여 정확한 데이터 전송 보장

3) TCP (Transmission Control Protocol)

▼ 개념

- 인터넷 프로토콜 스위트의 핵심 프로토콜

- 전송 계층에 위치, 근거리 통신망이나 인트라넷, 인터넷에 연결된 컴퓨터에서 실행되는 프로그램 간에 일련의 옥텟을 안정적으로, 순서대로, 에러 없이 교환할 수 있게 해주는 프로토콜

▼ 특징

- 신뢰성 보장 - 패킷 손실, 중복, 순서 바뀜 등이 없도록 보장
- 연결 지향적 특징
- 흐름 제어 - 송신(송신 전송률) 및 수신(수신 처리율) 속도를 일치시킴
- 혼잡 제어 - 혼잡 제어 기법을 통해 송신율 감속

▼ 헤더 구조

- Source/Destination Port Number (각 16Bit)
- Squence Number(32Bit) - 신뢰성 및 흐름 제어 기능 제공
- Acknowledgement Number(32Bit) - 확인응답번호/승인번호
- HLEN(4Bit) - 헤더 길이
- Flag bit(6개)

URG(Urgent)

ACK

PSH(Push)

RST(Reset)

SYN

FIN(Finish)

- Window size(16bit)
- Checksum(16bit)
- Urgent pointer(16bit)
- Options and Padding

4) UDP (User Datagram Protocol)

▼ 개념

- 비연결성, 신뢰성 없으며, 순서화되지 않은 데이터그램 서비스 제공하는 전송(4계층) 계층의 통신 프로토콜

▼ 특징

- 비신뢰성
- 순서화되지 않은 데이터그램 서비스 제공
- 실시간 응용 및 멀티 캐스팅 가능
- 단순 헤더

▼ 헤더 구조

- Source Port Number(16bit)
- Destination Port Number(16bit)
- UDP Length(16bit)
- UDP Checksum(16bit)
- Data(가변)

5) IPv4

- 패킷 교환 네트워크 상에서 데이터를 교환하기 위한 32bit 주소 체계를 갖는 네트워크 계층의 프로토콜
- 4부분 10진수
- 멀티캐스트, 유니캐스트(1:1), 브로드 캐스트

▼ 헤더

- 주소 등 각종 제어 정보를 담고 있는 부분
- Version(4bit)
- IHL(IP Header Length) (4bit)
- ToS(Type of Service) (8bit)
- Total length (16bit)
- Identification (16bit) - 단편화 시에만 필요
- Flag(3bit)
- Fragment Offset(13bit) - 위치 값
- TTL(Time To Live) (8bit) - 수명
- Protocol(8bit)
- Header checksum(16bit)
- Source address(32bit)

- Destination Address(32bit)

6) IPv6

- IPv4의 주소 고갈, 보안성, 이동성 지원 등의 문제점 해결. 128bit 주소 체계
- 8부분 16진수
- 멀티 캐스트, 유니 캐스트, 애니 캐스트

▼ 특징

- IP 주소의 확장
- 이동성 - 네트워크의 물리적 위치에 제한 받지 않고 같은 주소를 유지하면서도 자유롭게 이동가능
- 인증 및 보안 기능
- 개선된 QoS 지원 - 특정 트래픽은 별도의 특별한 처리를 통해 높은 품질의 서비스 제공
- Plug&Play 지원 - IPv6 호스트는 네트워크 접속 시 순간 자동적으로 네트워크 주소 부여받음
- Ad-hoc 네트워크 지원 - 자동 네트워킹 및 인터넷 연결 지원
- 단순 헤더 적용
- 실시간 패킷 추적 기능

▼ 헤더

- IHL, Identification, Flag, Fragment Offset, Header Checksum 삭제
- Type of Service, Total length, Time to live, Protocol 변경
- Version(4bit)
- Traffic Class(8bit)
- Flow Label(20bit)
- Payload length(16bit)
- Next header(8bit)
- Hop limit(8bit)
- Source address(128bit)
- Destination address(128bit)

(3) 네트워크 전달 방식

1) 패킷 스위칭

- 컴퓨터 네트워크와 통신의 방식 중 하나로 작은 블록의 패킷으로 데이터를 전송하며 데이터를 전송하는 동안만 네트워크 자원을 사용하도록 하는 통신 방식
- WAN을 통해 데이터를 원격지로 송부하기 위해 X.25, 프레임 릴레이 및 ATM 기술 사용
- X.25

통신을 원하는 두 단말 장치가 패킷 교환망을 통해 패킷을 원활히 전달하기 위한 프로토콜

고정된 대역폭, 패킷 사용, 1~3 계층 담당

- 프레임 릴레이

ISDN 사용 위한 프로토콜

1~2 계층 담당, 가격 저렴, 기능 단순화, 유연한 대역폭

- ATM(Asynchronous Transfer Mode)

비동기 전송 모드라고 하는 광대역 전송에 사용하는 스위칭 기법

동기화를 맞추지 않아 보낼 데이터가 없는 사용자의 슬롯은 다른 사람이 사용할 수 있도록 하여 네트워크 상의 효율성 높임

AAL(ATM Adaptation Layer), ATM 계층, 물리 계층으로 계층 구성

- 의미: 데이터를 패킷 단위로 보내는 방식
- 장점: 회선 효율이 우수, 비동기 전송이 가능, 연결 설정이 필요 없고 다중 전달 용이
- 단점: 실시간 전송에 부적합, 네트워크 지연 발생
- 활용: 이메일, 메시지

2) 서킷 스위칭

- 네트워크 리소스를 특정 사용 층이 독점하도록 하는 통신 방식
- 전송 보장, 서킷 확보 작업 - 다른 기기들은 해당 경로 사용 불가

- 의미: 전송 경로를 설정한 뒤 데이터를 송수신하는 방식
- 장점: 경로 접속 시간 빠름, 전송 제어 절차와 형식에 제약 받지 않음
- 단점: 송수신 측 모두 데이터 교환 준비가 완료되어야 함, 회선이 독점되어 있음
- 활용: 영상, 비디오 등

(4) 라우팅 알고리즘

- 데이터는 송신 측으로부터 수신 측까지 데이터를 전달하는 과정에서 다양한 물리적인 장치들을 거침
- 목적지까지의 최적 경로를 산출하기 위한 법칙

▼ 거리 벡터 알고리즘

- 인접 라우터와 정보를 공유하여 목적지까지의 거리와 방향을 결정하는 알고리즘
- 벨만-포드 알고리즘 활용
- 각 라우터가 업데이트될 경우마다 전체 라우팅 테이블을 보내라고 요청하지만 수신된 경로 비용 정보는 이웃 라우터에게만 보내짐

▼ 링크 상태 알고리즘

- 링크 상태 정보를 모든 라우터에 전달하여 최단 경로 트리를 구성하는 알고리즘
- 다익스트라 알고리즘 활용
- 네트워크를 일관성 있게 파악 가능, 계산이 복잡하고 트래픽을 광범위한 범위까지 전달

▼ 라우팅 프로토콜 종류

- RIP - 거리 벡터 알고리즘 활용, 느린 업데이트
- IGRP - 네트워크 상태를 고려하여 라우팅 (RIP 진화 버전)
- OSPF - 링크 상태 알고리즘 활용, 빠른 업데이트
- BGP - 규모가 큰 네트워크 상호 연결

4. 기본 개발 환경 구축

(1) 운영체제 선택

- 윈도우 계열, 리눅스/유닉스 계열

▼ 운영체제 운용

- 외부의 침입이나 바이러스로 인해 시스템이 통제 불능의 상태가 되어 불필요한 리소스를 낭비하거나 중요한 데이터의 유실을 방지하기 위해 지속적으로 운용
- 서버 운영체제 운용 기준

  운용 아키텍처 및 기능 파악, 백업 주기, 보안 업데이트 주기 설정 및 점검

- 개별 PC용 운영 체제 운용 기준

정기적 데이터 백업, 보안 업데이트, 시스템 백업

(2) 개발 도구 설치 및 운용

▼ 프로그래밍 언어

- 언어 타입 - 정적(Java, C#, C++, COBOL) / 동적(SQL, PHP, Python) 개발 언어
- 시스템 특징 - 일반 / 도메인 특화 시스템
- 언어 특징 - 객체지향, 명령형, 순차적, 선언형(SQL)
- 지원 - 관리 도구 지원형, 언어 독립형

▼ 개발 지원 도구

- 요구사항 관리
- 설계
- 구현
- 테스트
- 빌드 - Jenkins
- 형상 관리
- 품질 관리
- 이슈 관리
- 프로젝트 관리

(3) 응용 시스템 개발 인프라 구축

▼ 방식

- 온프레미스(On-Premise) 방식 - 외부 인터넷망 차단된 상태에서 인트라넷망만 활용하여 개발 환경 구축 방식
- 클라우드 방식

컴퓨팅 환경 - 하드웨어 장비 세팅, 웹 서버, DBMS 서버

스토리지

데이터베이스

네트워킹 전송

개발자 도구

보안 환경 구축

응용 기술 세팅

생산성 향상

- 하이브리드 방식