항공소프트웨어이해 기말고사 준비자료

운영체제의 처리방식(운영체제/운영체제 개념)

★ 운영체제의 처리방식

① 일괄처리 시스템
- 처리할 작업을 일정 기간 또는 일정량이 될 때까지 모아두었다가 한꺼번에 처리
- 작업 결과를 빠르게 확인할 수 없음

② 다중 처리 시스템
- 두 개 이상의 프로세서(CPU)로 구성되어 다중 작업을 처리하는 시스템

③ 다중 프로그래밍 시스템
- 주기억장치에 여러 개의 프로그램이 상주하면서 자원이 이용 가능할 때만 처리
- 사용자에게 "여러 개의 프로그램이 동시에 실행"되는 것 같은 착각을 줌

④ 시분할 처리 시스템
- 각 사용자들에게 CPU에 대한 일정 시간을 제공하여 주어진 시간 동안 프로그램을 수행
- 많은 사용자들이 하나의 컴퓨터를 공유

⑤ 실시간 처리 시스템
- 처리를 요구하는 자료가 발생할 때마다 즉시 처리하여 정해진 짧은 시간 내에 응답 → 데이터 처리가 엄격한 시간제한을 갖는 시스템
- 변동 사항 발생 시 즉시 수정 가능

⑥ 분산 처리 시스템
- 네트워크를 통해 연결된 여러 컴퓨터 시스템에 작업과 자원을 나누어 처리
- 다수의 컴퓨터에서 각 CPU가 각자 독립적인 메모리를 갖고, 고속의 버스를 이용해서 다른 CPU와 통신을 하면서 작업을 처리

프로세스개념, 병행프로세스(운영체제/프로세스 관리)

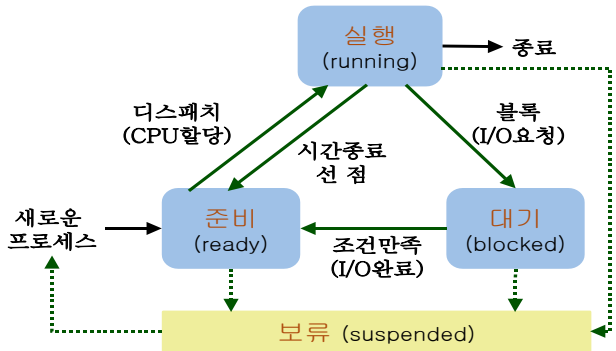
★ 프로세스

- 실행 중이거나 곧 실행이 가능한 프로세스 제어 블록(PCB)을 가진 프로그램

☞ 프로세스에 대한 정보를 저장하고 있는 자료구조 테이블

포인터	프로세스 현재상태
프로세스 식별자	
프로그램 카운터	
프로세스 우선순위	
레지스터	
기억장치 관리정보	
기타	

★ 프로세스 상태

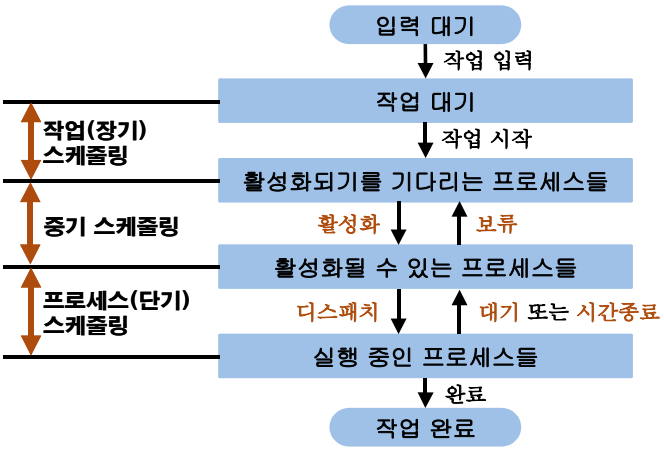


★ 인터럽트
- 프로세서가 명령문을 수행하고 있을 때 예측하지 못했던 사건 등이 발생하여 다른 작업을 처리하기 위해 수행하던 일을 강제로 중단시키는 것
- 종류
  ☞ SVC 인터럽트, 입출력 인터럽트
  ☞ 프로그램 검사 인터럽트, 하드웨어 인터럽트
  ☞ 외부 인터럽트, 재시작 인터럽트

★ 병행 프로세스
- 두 개 이상의 프로세스가 동시에 실행되는 것
- 비동기성: 프로세스 간에 상호 협력하는 것
- 병행 프로세스의 비동기성으로 인한 문제의 해결 방법
  ☞ 동기화(synchronization), 상호배제(mutual exclusion), 임계영역(critical section)

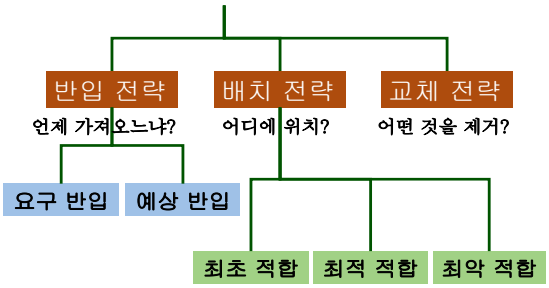
스케줄링(운영체제/프로세스 관리)

★ 스케줄링
- 특정 자원을 요청하는 대상들 중에서 누구에게 먼저 할당해 줄 것인가를 결정하는 일
- 단계별 구분
  ☞ 작업 스케줄링(장기 스케줄링)
    ▶ 디스크 내에서 어떤 작업을 주기억장치로 가져와서 수행시킬 것인가를 결정
    ▶ 보류에서 준비상태로 이동시킴
  ☞ 중기 스케줄링: 프로세스에게 주기억장치의 할당 순서 결정
  ☞ 프로세스 스케줄링(단기, CPU 스케줄링)
    ▶ 프로세스에게 CPU의 할당 순서 결정

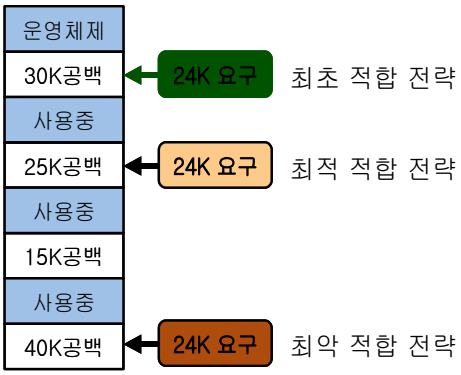


주기억장치관리, 가상기억장치(운영체제/기억장치 관리)

★ 기억장치 관리
- 주기억장치 관리 전략

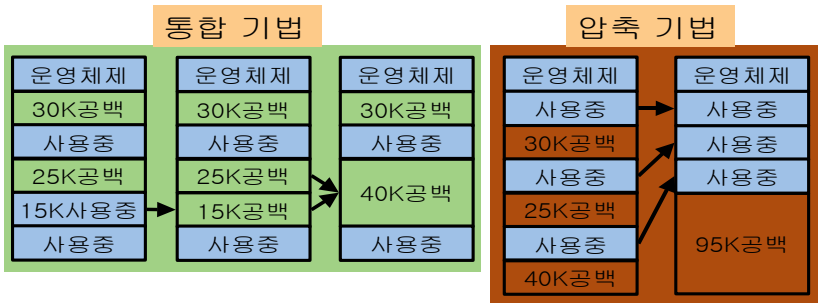


- 배치 전략



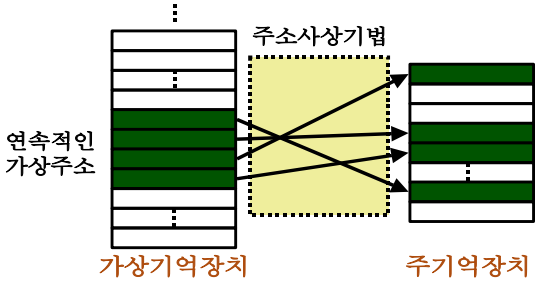
- 주기억장치 관리 기법
  - ☞ 단일 프로그래밍 기법: 시스템 내에 항상 하나의 프로세스만 존재
  - ☞ 고정 분할 다중프로그래밍 기법: 주기억장치를 여러 개의 고정된 크기들로 분할하여 실행 중인 여러 프로세스에게 할당
  - ☞ 가변 분할 다중프로그래밍 기법: 프로그램 크기에 맞게 주기억장치 공간 할당

- 단편화 현상
  - ☞ 주기억장치에서 부분적인 기억공간이 프로그램에 의해 사용되지 않고 낭비되는 현상



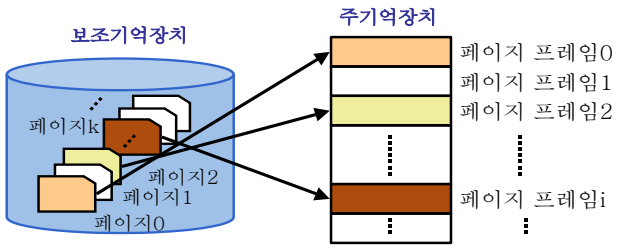
★ 가상기억장치

- 보조기억장치를 주기억장치인 것처럼 사용하기 위한 기억장소 관리 기법
  - ※ **디스크와 같은 장치를 사용해서 프로그램 관점에서 사용 가능한 메모리 영역을 넓게 보이는 기법**
  - ☞ 주기억장치의 용량보다 큰 기억용량의 주소를 지정



★ 가상기억장치 구현 방법

- 페이징(paging) 기법: 동일한 크기의 영역으로 분할
  - ※ **페이지: 프로그램을 동일한 크기의 영역으로 나눈 것**
  - ☞ 프로그램 → "페이지", 메모리 → "페이지 프레임"



- 세그먼테이션(segmentation) 기법
  - ☞ 프로그램과 데이터를 세그먼트 단위로 분할하여 주기억장치에 적재하는 기법
    - ▶ 서로 다른 크기로 분할된 프로그램 블록들
  - ☞ 모듈/함수 단위의 논리적인 개념에 의해 분할

- 페이징 기법과 세그먼테이션 기법의 혼합
  - ☞ 프로그램 → 논리적인 세그먼트 단위로 분할
  - ☞ 각 세그먼트 → 페이지 단위로 분할
  - ☞ 분할된 페이지 단위로 주기억장치에 적재

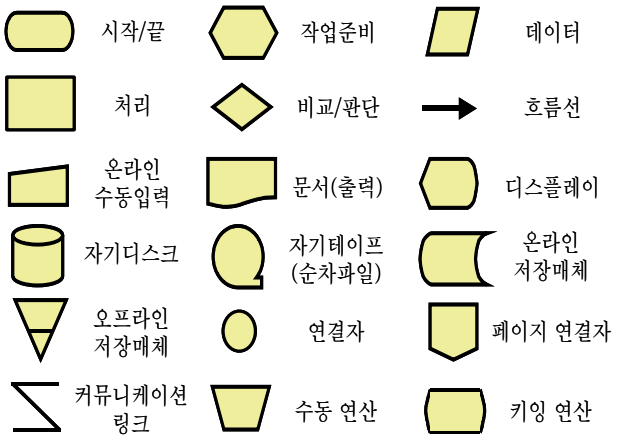
★ 가상기억장치 관리 기법

① 반입 기법
- 언제 보조기억장치에서 주기억장치로 옮길 것인가?
- 요구 반입 기법, 예상 반입 기법

② 배치 기법
- 주기억장치의 어디로 옮길 것인가?
- 페이징 시스템은 특별한 배치 기법이 불필요

③ 교체 기법
- 주기억장치에 있는 어느 것을 교체할 것인가?
- 방법: FIFO, LRU, LFU

알고리즘 설계(순서도) (프로그래밍언어/프로그램 작성 과정)



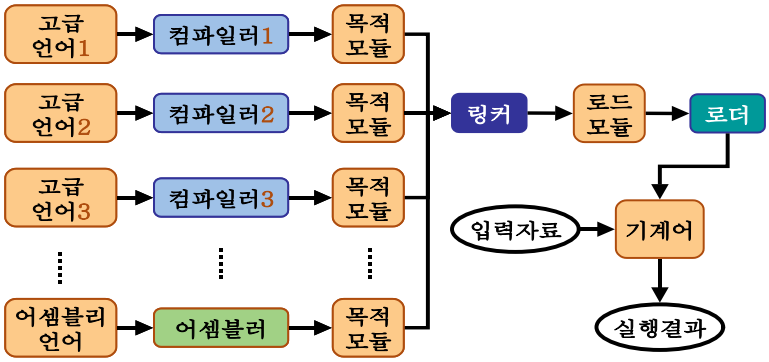
프로그래밍언어 구현기법(프로그래밍언어)

★ 프로그램 언어 구현 기법

① 번역기법
- 주어진 고급 언어로 작성된 프로그램을 실제 주어진 컴퓨터의 기계어로 번역하여 동등한 의미의 기계어 프로그램을 만들어 실행시키는 방법
- 번역기 : **원시 언어로 작성된 프로그램을 입력으로 읽어 들여서 목적 언어로 내용이**

똑같은 프로그램으로 출력해주는 언어 처리기

- 번역기의 종류: 컴파일러, 어셈블러, 링커, 로더, 프리프로세서



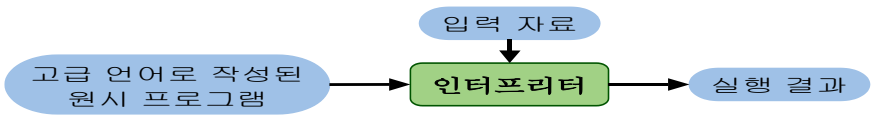
- 컴파일러 언어(종류: 포트란, 알골, 파스칼, 코볼, C, 에이다)

☞ 고급 언어를 번역하여 목적 모듈을 출력시켜서 그 목적 모듈을 링크, 로드하여 실행시키는 방법으로 구현하는 언어(느린 번역, 빠른 실행, 큰 기억장치 요구)

※ 많은 횟수 반복되는 프로그램의 경우에 효율적

② 인터프리터 기법

- 고급 언어로 된 프로그램을 자료로 읽어 들여서 기계어 수행과 동일한 알고리즘으로 프로그램의 각 문장을 디코딩하고 실행시키는 방법

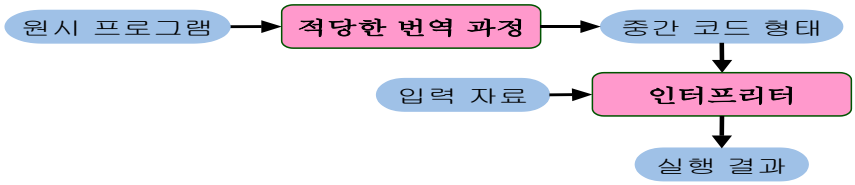


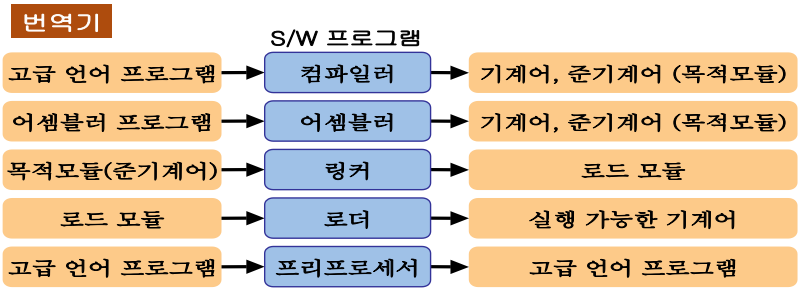
- 인터프리터 언어(종류: 리스프, 스노볼4, APL, 프롤로그)

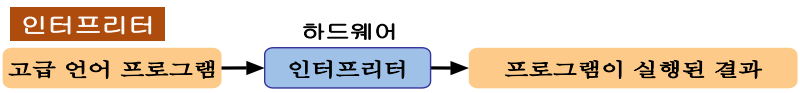
☞ 고급 언어를 적당한 중간 코드까지만 번역해서 곧바로 실행시키는 방법을 택하는 언어(실행 중에 자료의 동적 변화나 사용자와의 대화가 용이)

③ 하이브리드 구현기법

- 프로그램을 좀더 실행시키기 쉬운 형태로 번역한 후, 그 번역된 형태의 프로그램을 디코드해서 시뮬레이션으로 실행하는 구현 기법







☞ **컴파일러 언어**

고급 언어를 번역하여 목적 모듈을 출력시켜서 그 목적 모듈을 링크, 로드하여

실행시키는 방법으로 구현하는 언어

☞ **인터프린터 언어**

고급 언어를 적당한 중간 코드까지만 번역해서 곧바로 실행시키는 방법을 택하는 언어

객체지향프로그래밍(프로그래밍언어)

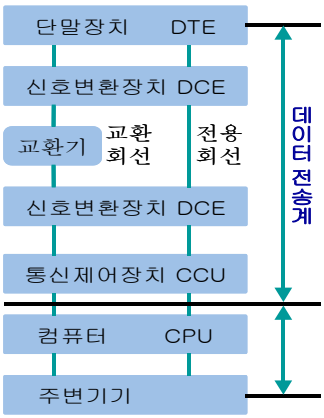
★ 객체지향 프로그래밍

- 모든 데이터 항목을 "객체"로 취급
  - ▶ 능동적 객체 (객체 = 속성 + 함수)
  - ▶ 클래스: 같은 속성과 함수를 갖는 객체들의 모임
  - ▶ 한 클래스는 여러 하위 클래스로 구성 가능
  - ▶ 다른 객체와 메시지 교환을 통해 통신
- 상속 ➔ 소프트웨어 재사용성 향상
  - ▶ 새로운 객체가 기존 객체의 서브 클래스로 정의되면 기존 객체의 데이터와 연산을 그대로 전달 받아서 사용할 수 있는 것
- 다형성 ➔ 부모클래스는 서브클래스에 의해 무시될 수도 있는 메소드를 정의
  - ▶ 서브클래스의 어떤 객체든 참조될 수 있도록 부모클래스의 타입을 다형성 변수로 정의
- 동적 바인딩 ➔ 다형성 지원
  - ▶ 객체가 필요로 하는 데이터 타입이 무엇인지 미리 정하지 않고, 메소드 호출 시 적절한 클래스의 메소드에 동적으로 바인딩되는 개념

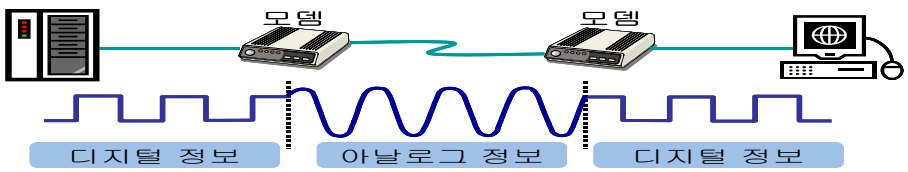
데이터통신 시스템(네트워크와 정보통신/데이터통신)

★ 데이터 통신

- 단말 장치(DTE, Data Terminal Equipment)
  - ▶ 송신자나 수신자의 끝에서 신호를 보내는 장치
  - ▶ 컴퓨터가 통신 장치들을 통해 데이터를 교환하기 위한 인터페이스

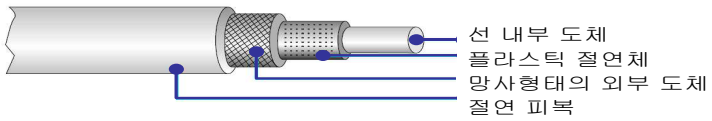


- DCE, Data Communication Equipment
  - ◉ 모뎀(MOdulation and DEModulation)

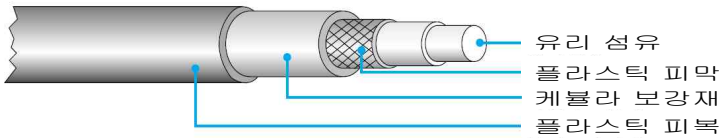


  - ◉ 디지털 서비스 장치(Digital Service Unit)
    - ▶ 디지털 신호를 변조하지 않고 그대로 디지털 전송 매체를 이용해 고속으로 전송

- DCL, Data Communication Line: 송수신자 사이를 연결해주는 물리적인 통로
  ◉ 유선 통신: 이중나선(twisted-pair wire), 동축케이블(coaxial cable), 광섬유(optical fiber)
  ◉ 무선 통신: 라디오파, 마이크로파, 위성 마이크로파
  ※ 이중 나선: 절연된 두 가닥의 구리선이 균일하게 감겨 있는 형태의 통신 회선
  ※ 동축 케이블



  ※ 광섬유: 빛을 전송하는 머리카락보다 가늘고 유연한 실린더형 매체



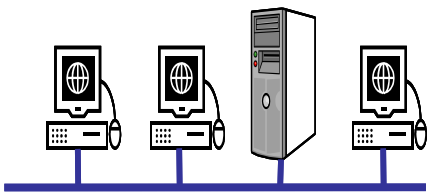
  ※ 라디오파: 방송 형태의 통신(FM 라디오 채널, TV 채널)
  ※ 마이크로파: 접시형 안테나 사용
  ※ 위성 마이크로파: 위성에 사용하는 전파는 1~10GHz의 주파수 범위를 사용하므로 구분상 마이크로파에 해당
- 통신제어장치(CCU, Communication Control Unit)
  ▶ 컴퓨터와 통신 회선 사이에서 서로 다른 데이터 취급 방법과 전송 속도 등을 조절하는 장치
  ▶ 통신 회선으로의 접속과 종료, 데이터 전송 방식과 전송 상태 보고, 다중 접속 제어 등을 담당
- 통신 소프트웨어
  ▶ 데이터 통신망의 기능을 관리
  ▶ 데이터 통신 시스템 장치들 간의 하드웨어 제어와 데이터 송수신 기능을 담당
  ▶ 사용자와 통신 시스템 간의 인터페이스를 제공

회선망의 유형(네트워크와 정보통신/데이터전송)

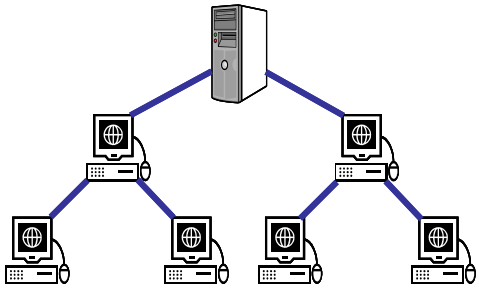
★ 회선망의 유형

☞ 데이터 통신망을 통신 회선을 이용하여 공간적으로 어떻게 구성하느냐에 따라

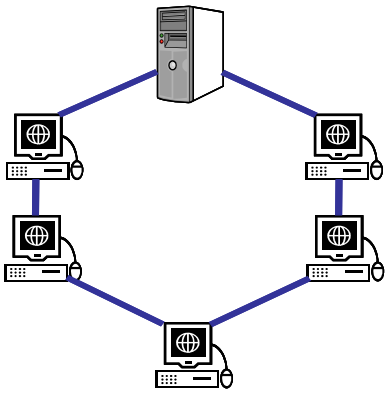
1) 버스(bus)형: 하나의 통신 회선에 여러 대의 노드를 연결한 형태
   - 데이터가 있을 때 방송(broadcast) 방식으로 연결된 모든 노드에게 전송
   - 한 노드의 고장이 전체 통신망에 영향을 미치지 않음



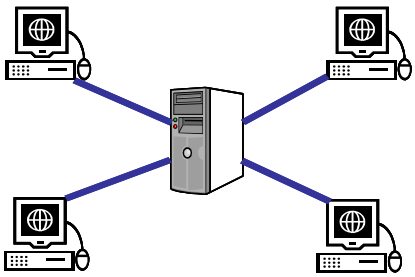
2) 트리(tree)형: 원점(headend)에서 한 개 이상의 가지를 뻗어 나가는 구조
   - 버스형의 확장 형태
   - 데이터는 양방향으로 모든 노드에게 전달



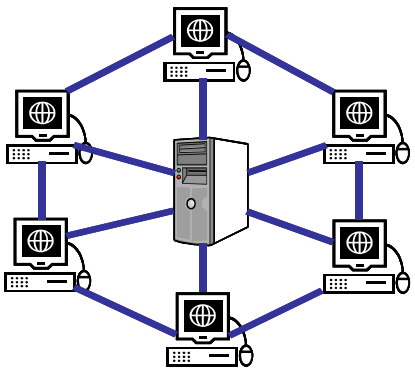
3) 링(ring)형: 원형의 형태로 인접한 두 개의 노드를 점대점 방식으로 연결
   - 단방향 링, 이중 링



4) 성(star)형: 중앙의 제어 노드(호스트)에 다른 모든 노드들을 점대점 방식으로 연결한 형태
   - 중앙 집중식 구조로 유지보수 및 전송제어 용이
   - 호스트 장애 발생 → 전체 통신망의 동작 중단



5) 매시(mesh)형: 통신의 신뢰도가 중요시되는 구성 형태
   - 주 컴퓨터의 중계 없이 직접 노드들 상의 통신이 가능하므로 통신망의 효율이 향상



OSI모델(네트워크와 정보통신/인터네트워킹)

★ OSI(Open System Interconnection) 모델

- 개방형 시스템 간 상호 접속

① 물리 계층(physical layer, L1)
- 물리적인 매체를 통한 비트 스트림의 전달 기능 제공
- 인터페이스의 기계적, 전기적, 기능적, 절차적 특성에 대한 규정을 다룸

② 데이터링크 계층(data link layer, L2)
- 같은 네트워크 상에 있는 두 시스템 간의 패킷 전달과 이에 대한 흐름 제어 및 오류 제어 등을 제공

③ 네트워크 계층(network layer, L3)
- 패킷을 송신측으로부터 여러 네트워크를 통하여 수신측까지 전달

④ 전송 계층(transport layer, L4)
- 송신측에서 수신측까지의 전체 메시지 전달 기능, 오류 제어 및 흐름 제어 제공
- 전체 메시지가 올바른 순서대로 도착하는 것 보장

⑤ 세션 계층(session layer, L5)
- 두 시스템 간의 통신을 위해 대화 제어
- 사용자 위주의 논리적인 연결 서비스 제공
- 통신 시스템 간의 상호 대화가 질서 있게 이루어질 수 있도록 설정 및 유지하고 동기화함

⑥ 표현 계층(presentation layer, L6)
- 응용 계층 간에 전송하는 데이터를 표현함에 있어서 의미에 따른 구문의 차이를 해소하기 위해 데이터 변환, 데이터 압축 및 암호화 등의 기능을 제공

⑦ 응용 계층(application layer, L7)
- 사용자나 응용 프로그램 사이에서 통신을 위한 서비스 제공
- 전자우편, 원격 파일 접근 및 전송 등의 서비스 제공

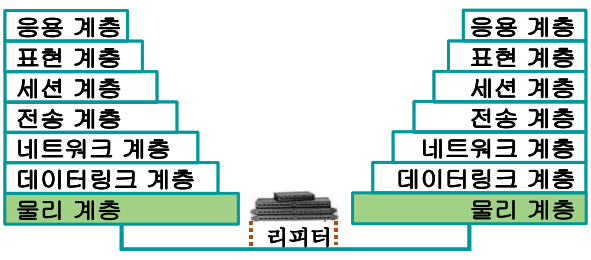
◉ 계층별 주요 기능(★★★ 순서가 매우 중요함)

계층		기능
7	응용 계층	응용 프로세스 간의 정보 교환
6	표현 계층	정보 표현 방식 관리, 암호화, 정보 압축
5	세션 계층	응용 프로세스 간의 대화 관장
4	전송 계층	호스트 간의 메시지 단위의 정보 교환 및 관리
3	네트워크 계층	통신 전담 프로세서 간의 네트워크를 통한 패킷 전송
2	데이터링크 계층	링크 상에서의 패킷 전송
1	물리 계층	링크 상에서의 신호 전송

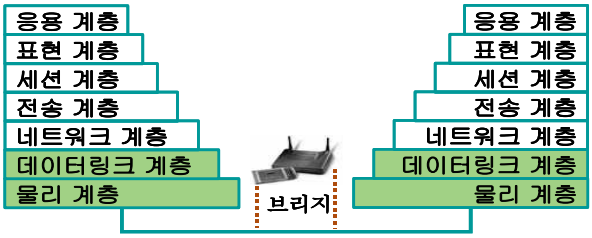
인터네트워킹 장비(네트워크와 정보통신/인터네트워킹)

★ 인터네트 워킹 장비

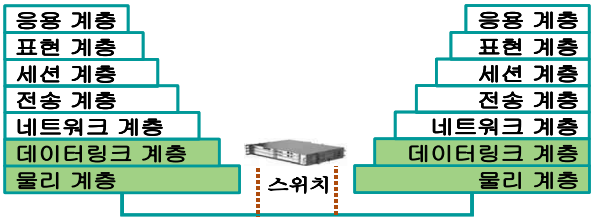
① 리피터
- OSI 물리 계층에서 동작하는 장치
- 송신측의 원래 비트 형태를 재생하여 전송
- 네트워크의 물리적인 길이 확장 역할



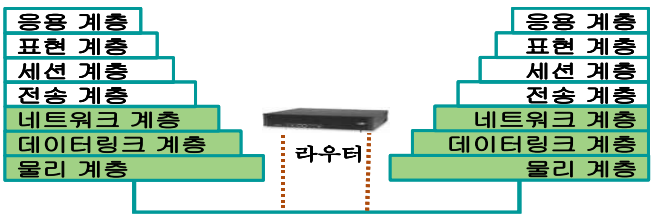
② 브리지(bridge)
- OSI 데이터링크 2계층인 MAC 계층에서 동작
- 수신한 데이터 형식/내용 변화 없이 네트워크를 연결해줌



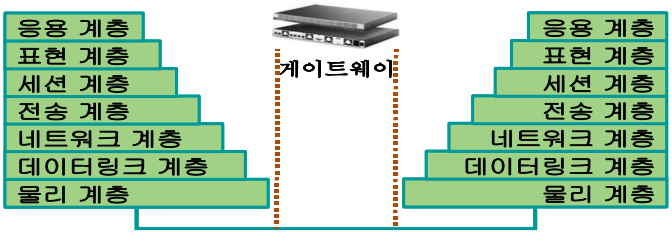
③ 스위치(switch)
- OSI 데이터링크 계층에서 동작하는 연결 장치
- HW로 데이터 처리 → 비교적 빠른 속도 제공, 서로 다른 속도의 다수 개의 포트들도 제공



④ 라우터(router)
- OSI 네트워크 계층에서 동작하는 장치
- 최적 경로 설정 및 관리, 데이터 전송
- 라우팅 프로토콜을 통한 라우팅 테이블 운영



⑤ 게이트웨이(gateway)
- OSI 4~7계층에서 네트워크의 연결 기능 담당
- 사용하는 매체나 각종 프로토콜이 일치하지 않는 서로 다른 네트워크 구조 연결



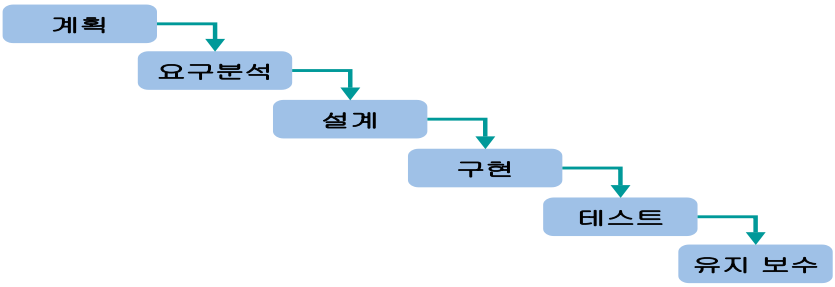
소프트웨어 개발 프로세스 모형(소프트웨어공학)

★ 소프트웨어 개발 프로세스 모형

◉ 일반적인 소프트웨어 생명주기(3단계)

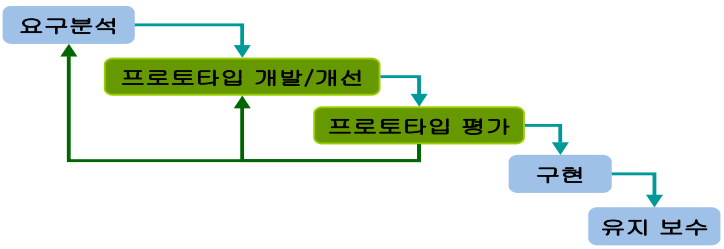
1) 정의 단계: 사용자의 요구사항 정의 (시스템 분석 활동 포함)
2) 개발 단계: 요구사항과 명세서에 따라 소프트웨어 개발
   시스템 설계, 코딩, 테스트 등의 활동 포함
3) 유지 보수 단계: 운용 도중 생기는 결점(error)을 수정, 개선하는 활동
   수정형, 적응형, 완전형, 예방형 유지보수를 포함

◉ 폭포수 모형



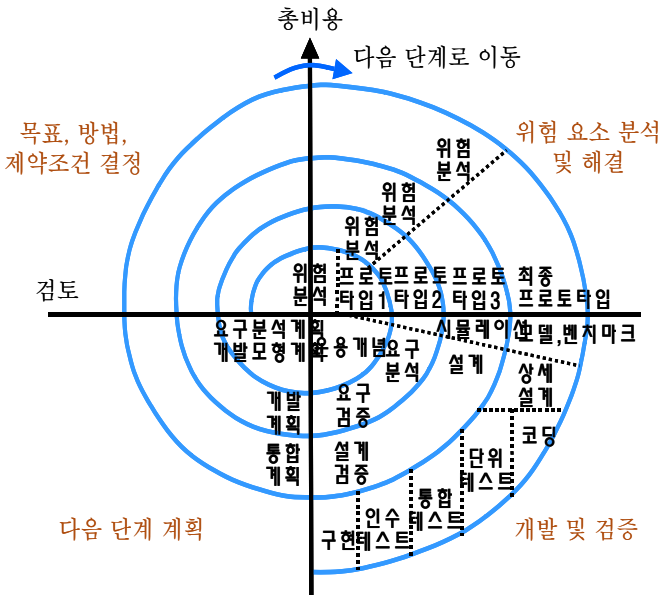
◉ 프로토타입 모형

- 개발자는 추후에 개발될 소프트웨어의 시범 케이스/본보기로 프로토타입을 만들어 제시



◉ 나선형 모형

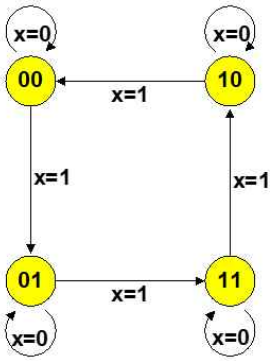
- 목표, 방법, 제약조건 결정
- 위험 요소 분석 및 해결
- 개발 및 검증
- 다음 단계 계획



서술형

ARINC429, ARINC653 , 소프트웨어 테스트, 소프트웨어 유지보수

다음 상태변이도를 참고하여 상태표/여기표을 작성하고 카르노캡을 이용하여 출력을 간소화 시킨후 순서 논리회로를 완성하시오. (단, JK F/F을 사용)

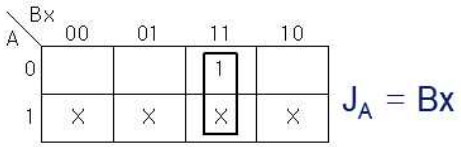


답안작성 방법

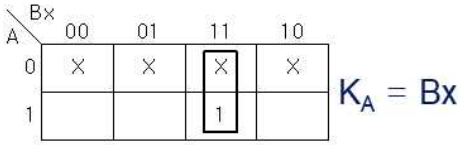
1. 상태표/여기표 작성

Present State		Input	Next State		F/F Input(여기표)			
A	B	x	A	B	J_A	K_A	J_B	K_B
0	0	0	0	0	0	x	0	x
0	0	1	0	1	0	x	1	x
0	1	0	0	1	0	x	x	0
0	1	1	1	0	1	x	x	1
1	0	0	1	0	x	0	0	x
1	0	1	1	1	x	0	1	x
1	1	0	1	1	x	0	x	0
1	1	1	0	0	x	1	x	1

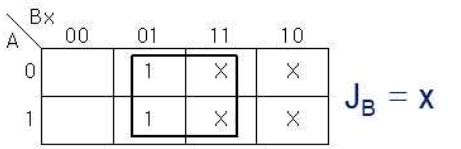
2. 맵을 이용한 간소화



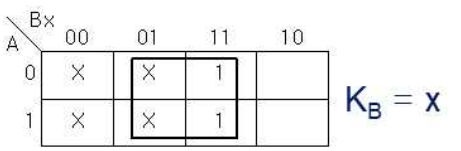
$J_A = Bx$



$K_A = Bx$



$J_B = x$



$K_B = x$

3. 논리회로 작성

