<1. 요구사항 확인>

- **프레임워크**: 소프트웨어의 특정 부분 설계 및 구현 시 재사용이 가능하도록 클래스 제공
- 소프트웨어 프레임워크의 특징: **모듈화**(인터페이스에 의한 캡슐화), **재사용성**(반복적으로 사용하는 컴포넌트를 정의할 수 있게 함), **확장성**(다형성을 통해 프레임워크의 인터페이스를 넓게 사용), **제어의 역흐름**(프레임워크 코드가 전체 애플리케이션의 처리 흐름을 제어->제어가 반대로 흐르게 한다)
- **다형성:** 프로그래밍 언어의 요소들이 다양한 자료형에 속하는 것이 허가되는 성질(오버로딩, 오버라이딩)
- **Component:** 특정 기능 수행을 위해 독립적 개발/보급되고 다른 부품과 조립되어 사용되는 S/W 단위
- 4+1뷰: **유스케이스(**아키텍처 설계 작업 주도), **논리**, **프로세스**, **배포**, **구현** 뷰
- **JDBC** : JAVA 언어를 이용하여 DB에 접근하여 관리할 수 있는 인터페이스
- **ODBC** : 응용프로그램에서 DB에 접근하여 데이터를 관리할 수 있는 표준 인터페이스
- **Middleware**: 운영체제와 SW 사이에서 원만한 통신이 이루어질 수 있도록 중개/제어 역할을 하는 SW
- **RPC**(**R**emote **P**rocedure **C**all): 원격 프로시저를 마치 로컬 프로시저처럼 호출하는 방식의 미들웨어
- **MOM**(**M**essage **O**riented **M**iddleware): 메시지 기반의 비동기형 메시지를 전달하는 방식의 미들웨어
- **TP-Monitor**: 온라인 트랜잭션 업무에서 트랜잭션을 처리 및 감시하는 미들웨어
- **ORB**(**O**bject **R**equest **B**roker): 코바(CORBA) 표준 스펙을 구현한 객체 지향 미들웨어
- **WAS**(**W**eb **A**pplication **S**erver): 사용자의 동적인 콘텐츠를 처리하기 위해 사용되는 미들웨어
- **OLTP**(**O**n**L**ine **T**ransaction **P**rocessing): 네트워크에서 여러 이용자가 실시간으로 DB 작업 처리하는 방식
- **tpmC**: 특정 서버 1분당 최대 처리 건수, 객관적인 하드웨어 성능 지표로 사용
- 요구사항 개발 프로세스: 도출->분석->명세->확인
- **Prototyping**: 새로운 요구사항 도출 수단/요구사항에 대해 개발자가 해석한 것을 확인하기 위한 수단
- 하향식 선정방법: 전문가 판단, **델파이 기법**(전문가들의 경험적 지식을 통해 미래 예측)
- 상향식 선정방법: **LOC, Man/Month, Putnam**(개발 주기 단계별 요구+인원 분포**, COCOMO(**프로그램 규모)
- **UML**: 개발자들의 의사소통을 원활히 해주는 객체지향 모델링 언어 - 다이어그램, 사물, 관계
- UML 관계: 연관 —, 집합 ◇, 포함(영향을 끼침) ◆, 일반화 —|>, 의존(일시적) --→, 실체화(행위) ---|>
- **구조적** 다이어그램: 클래스, 객체, 컴포넌트, 배치, 복합체 구조, 패키지
- **행위** 다이어그램: 유스케이스, **시퀀스**(메시지 흐름), 커뮤니케이션, 상태, 활동, 상호작용 개요, 타이밍

<2. 데이터 입출력 구현>

- 데이터 모델링: 개념적->논리적->물리적
- **이상 현상**: 데이터의 중복성으로 인해 테이블을 조작할 때 발생하는 비합리적인 현상(**삽입**, **삭제**, **갱신**)

- **반정규화**: 시스템의 성능 향상을 위해 정규화 원칙을 의도적으로 위배하는 행위
- **정규화**: 관계형 DB 설계 시 중복을 최소화하여 데이터를 구조화하는 과정
- 제1정규화(테이블 내의 속성 값은 **원**자 값), 제2정규화(**부**분 함수 종속성을 제거)
- 제3정규화(**이**행 함수 종속성을 제거), BCNF(**결**정자 함수의 종속성을 제거)
- 제4정규화(**다**중 값 종속을 제거), 제5정규화(**조**인 종속성을 제거)
- **Index**: 검색 연산의 최적화를 위해 키값과 포인터의 쌍으로 구성되는 데이터 구조
- **View**: 사용자에게 허용된 정보만 보여주기 위해 하나 이상의 테이블로부터 유도된 논리적인 가상 테이블
- **Cluster**: 데이터 접근 효율을 높이기 위해 동일한 성격의 데이터를 같은 데이터 블록에 저장하는 방법
- **Partitioning**: 대용량 테이블을 작은 논리적인 단위인 파티션으로 나누는 것(해시,레인지,리스트,컴포지트)
- **PL/SQL**(**P**rocedural **L**anguage for **SQL**) : 표준 SQL을 기반으로 Oracle에서 개발한 데이터 조작 언어
- **APM(A**pplication **P**erformance **M**onitoring**):** 안정적인 시스템 운영을 위한 성능 모니터링 도구
- **Optimizer** : SQL을 가장 빠르고 효율적으로 수행할 수 있는 최적의 경로**(=실행 계획)** 생성
- **RBO(**규칙 기반 옵티마이저-사전 등록 규칙에 따라), **CBO**(비용 기반 옵티마이저-모든 접근 경로를 고려)

<3. 통합 구현>

- **연계 시스템** 구성: 송신 시스템, 수신 시스템, 중계 서버
- 직접 연계(DB Link, DB Connection, API, JDBC) vs 간접 연계(EAI, ESB/Web Service, Socket)
- **EAI:** 기업에서 운영, 이기종 간 시스템 연계 솔루션(**Point-to-Point**, **Hub&Spoke**, **Message Bus**, **Hybrid**)
- **Adapter:** 이기종 간을 연결하는 EAI의 핵심 장치
- **ESB:** 기업에서 운영, 이기종 간 서비스를 통합->하나의 시스템으로 관리 운영 (SOA의 토대, BUS 이용)
- **SOA(S**ervice-**O**riented **A**rchitecture**):** 느슨하게 결합된 서비스 기반 APP을 구현하기 위한 아키텍처 모델
- **SOAP**: HTTP, HTTPS, SMTP 프로토콜을 사용하여 XML 기반의 메시지를 교환하는 프로토콜
- **WSDL**(**W**eb **S**ervice **D**escription **L**anguage): 웹 서비스에 대한 상세 정보를 기록한 파일
- **UDDI**(**U**niversal **D**escription **D**iscovery **I**ntegrateion): WSDL을 등록하고 검색하기 위한 저장소
- **XML**: HTML 문법이 호환되지 않는 문제와 SGML의 복잡함을 해결하기 위해 고안된 마크업 언어
- **JSON:** 비동기 브라우저, 서버 통신을 위해 '속성-값'의 쌍으로 이루어진 개방형 표준 포맷
- **AJAX:** JS를 이용한 비동기 통신으로 클라이언트와 서버 간 XML 데이터를 주고받는 기술
- **REST:** URL을 통해 자원을 명시하고 HTTP 메서드를 통해 해당 자원을 조작할 수 있는 웹 아키텍처

<4. 서버 프로그램 구현>

- **JVM** : JAVA 기반 APP을 위해 시스템 메모리를 관리하고 실행 환경을 제공하는 가상 머신

- **JDK**: JAVA 기반 APP을 개발하는 데 필요한 툴들을 모아놓은 소프트웨어 패키지(JVM 포함)
- **JUnit** : JAVA용 단위 테스트 도구 ***어노테이션**: 주석을 달아 특별한 의미를 부여하는 메타데이터
- **Module:** 기능 단위로 분해 및 추상화되어 재사용 및 공유가 가능한 단위
- **모듈화:** 모듈을 통해 소프트웨어의 성능을 향상시키고 디버깅, 수정, 통합을 용이하게 하는 설계 기법
- **결합도:** 상호 의존의 정도(data – stamp – control – external – common – content)
- **자료**(데이터 넘겨 주기), **스탬프**(자료구조), **제어**(흐름), **외부**(참조), **공통**(여럿이 공유), **내용**(직접 참조/수정)
- **응집도:** (functional – sequential – communication – procedural – temporal – logical – coincidental)
- **기능**(단일), **순차**(출력->입력), **통신/교환**(동일한 입출력), **절차**(순차 수행), **시간**, **논리**(유사한 성격), **우연**
- **DTO**(**D**ata **T**ransfer **O**bject): 프로세스 사이에서 데이터를 전송하는 객체
- **VO**(**V**alue **O**bject): 고정 클래스를 가지는 객체
- **DAO**(**D**ata **A**ccess **O**bject): 특정 타입의 DB의 추상 인터페이스를 제공하는 객체
- **GoF**(Gang of Four) 디자인 패턴: **생성 패턴**, **구조 패턴**, **행위 패턴**
- **생성 패턴: 추상 팩토리**(의존적인 객체들의 조합), **빌더**(복잡한 인스턴스 조립, 생성과 표기 분리), **팩토리 메소드**(상위클래스-인터페이스 정의/하위클래스-인스턴트 생성), **프로토타입**(일반적인 원형을 복제, 필요한 부분만 수정), **싱글톤**(전역변수 사용 않고 한 클래스에 한 객체만)
- **구조 패턴: 어댑터**(기존 클래스 재사용, 덧씌움), **브리지**(구현부에서 추상층 분리), **컴포지트**(트리 구조, 부분-전체 계층), **데코레이터**(기능을 추가해 나감), **퍼싸드**(거대한 코드에 접근할 수 있는 단순한 인터페이스 제공), **플라이웨이트**(유사 객체들 메모리 사용량 최소화), **프록시**(접근 힘든 객체에 대한 대역 제공)
- **행위 패턴: 커맨드**(캡슐화->재사용성 높은 클래스 설계), **옵저버**(객체 상태 변화시 다른 객체들에게 연락), **템플릿 메소드**(알고리즘 정의), **책임 연쇄**(처리 못하면 다음 객체로), **중재자**(객체간 의존성 줄여 결합도 감소), **전략**(동일 알고리즘들 캡슐화->상호 교환), **메멘토**(Ctrl+Z)
- **팬 인**(어떤 모듈을 제어하는 수), **팬 아웃**(어떤 모듈이 제어하는 수)
- **화이트박스** 테스트(내부 구조와 동작 검사), **블랙박스** 테스트(기능 작동 여부 확인)
- **배치 프로그램:** 유저와 상호작용 없이 일련의 작업을 묶어 정기적으로 반복 수행하는 일괄 처리 방법
- **스프링 배치**: 스프링 프레임워크, 대용량 처리를 제공하는 스케줄러 (ㄴ**정기/이벤트/온디맨드** 배치)
- **쿼츠 스케줄러**: 스프링 프레임워크에 플러그인, job/trigger 분리하여 유연성 제공하는 배치 스케줄러
- **Cron 표현식**: 스케줄러를 실행시키기 위해 작업이 실행되는 주기를 설정하는 표현식

<5. 인터페이스 구현>

- **인터페이스 설계서**: 이기종 시스템 및 컴포넌트 간 데이터 교환 및 처리를 위해 각 시스템이 교환되는 데이터, 업무, 송수신 주체 등이 정의된 문서
- **스니핑:** 직접 공격하지 않고 네트워크 중간에서 남의 패킷의 정보를 몰래 도청하는 공격 기법

- 인터페이스 보안 취약점**: 입**력데이터 검증/표현(검증 누락, 잘못된 형식), **보**안기능(인증/암호화/권한 관리 등 기능을 부적절하게 구현), **시**간 및 상태(부적절 관리), **에**러 처리(부적절 예외처리), **코**드 오류(널 포인터 역참조, 부적절 자원 해제), **캡**슐화(잘못된 세션, 디버그 코드, 시스템 데이터 정보 노출), **A**PI 오용
- DB 암호화 기법: **API** 방식, **Plug-In** 방식, **Hybrid** 방식
- **IPSec**: 무결성과 인증을 보장하는 인증 헤더와 기밀성을 보장하는 암호화를 이용한 IP 보안 프로토콜
- **SSL/TLS:** APP과 TCP/IP 계층 사이에서 데이터를 암호화하고 기밀성 보장하는 공개키 기반 보안 프로토콜
- 인터페이스 구현 검증을 위한 **테스트 프레임워크**: **xUnit**(다양한 언어), **STAF**(서비스 호출/컴포넌트 재사용/다양한 환경), **FitNesse**(웹 기반), **NTAF**(FitNesse+STAF, NHN), **Selenium**(다양한 브라우저), **Watir**(Ruby)
- **스카우터:** APP 모니터링 및 DB 모니터링, 인터페이스 감시 기능 제공하는 인터페이스 도구

<6. 화면 설계>

- **UI**: 유저와 시스템 사이에서 의사소통을 할 수 있도록 고안된 물리적 가상의 매개체(CLI, GUI, NUI)
- UI 설계 원칙: **직관성**(누구나 쉽게 이해하고 사용 가능), **유효성**(정확하고 완벽하게 사용자의 목표에 달성), **학습성**(누구나 쉽게 배우고 사용 가능), **유연성**(사용자의 인터랙션를 최대한 포용하고 실수 방지)
- **리치 클라이언트**(SW 실행을 클라이언트가 책임) vs **씬 클라이언트**(SW 실행을 전적으로 서버가 책임)
- **SSO**(**S**ingle **S**ign **O**n): 한 번의 로그인을 통해 다른 시스템에도 자동으로 접속하여 이용하는 방법
- **사용성 테스트:** 사용자가 직접 제품을 사용하면서 미리 작성된 시나리오에 맞춰 과제를 수행하는 테스트
- **페르소나:** 잠재적 사용자의 다양한 목적과 관찰된 행동 패턴을 응집시켜놓은 가상의 사용자
- **요구사항 매트릭스**: 페르소나의 목적을 기준으로 데이터 요구, 기능의 기반으로 만든 요구사항 표
- **3C 분석**(고객/경쟁사/자가 분석), **SWOT**(강점/약점/기회/위협 분석), **목업**(와이어프레임보다 더 실제처럼)
- **와이어프레임**: 이해관계자들과 화면 구성을 협의하거나 화면 단위로 대략적인 레이아웃만 구성한 문서
- **스토리보드**: UI 화면설계를 위해 와이어프레임과 DB연동,정책 등 구축하는 서비스의 정보가 수록된 문서
- **프로토타입:** 정적 화면으로 설계된 와-프, 스-보에 동적인 요소를 적용하여 만든 시뮬레이션 가능한 모형

<7. 애플리케이션 테스트 관리>

- **살충제 패러독스**: 동일한 TC로 테스트를 진행하면 더 이상 새로운 결함을 찾을 수 없다. 새로운 TC 필요
- **오류-부재의 궤변**: 사용자의 요구사항이 충족되지 않으면 결함이 없다해도 품질이 높다고 볼 수 없다.
- **워크스루**: 검토 회의 전 요구사항 명세서를 미리 배포하여 사전 검토 회의를 통해 결함 발견
- **인스펙션**: 요구사항 명세서 작성자를 제외한 다른 검토 전문가들이 결함 발견
- **동료 검토**: 요구사항 명세서 작성자가 내용을 직접 설명, 동료들이 이를 들으면서 결함 발견
- **테스트 커버리지**: 주어진 TC에 의해 수행되는 SW의 테스트 범위를 측정하여 테스트의 정확성과 신뢰성을 향상시키는 역할을 수행하는 테스트 품질 측정 기준

- 테스트: 검증/확인, 단위->통합->시스템->인수, 정적/동적, 구조 기반/명세 기반/경험 기반 로 분류 가능.
- **검증 테스트**: 개발자 시각 vs **확인 테스트**: 사용자 시각
- **단위 테스트**: 사용자 요구사항에 대한 단위 모듈, 서브루틴 등을 테스트 (인터페이스/자료구조/실행경로)
- **통합 테스트**: 단위 테스트를 통과한 SW/HW 컴포넌트 간 인터페이스 및 연동 기능을 구조적으로 테스트
- **하향식 통합**(깊이 or 너비 우선, Stub), **상향식 통합**(Cluster->Driver), **빅뱅 통합**(모든 모듈을 한번에 통합)
- **시스템 테스트**: 실제 환경과 최대한 유사한 환경에서 진행 (단위/통합 테스트 통과 후)
- **인수 테스트**: 사용자의 입장에서 테스트. **알파 테스트**와 **베타 테스트**가 존재
- **정적 테스트:** 명세서나 소스코드를 대상으로 분석하는 테스트 (워크스루, 인스펙션, 코드 검사 등)
- **동적 테스트**: 프로그램을 실행하여 오류를 찾는 테스트(블랙/화이트 박스 테스트)
- **구조 기반 테스트**(화이트박스 테스트): **구문 커버리지**(모든 명령문 한번씩), **결정 커버리지**(모든 결정문이 참과 거짓 수행), **조건 커버리지**(모든 결정문 내 각 조건이 참과 거짓 수행), **조건-결정 커버리지**(개별 조건식도 참과 거짓 수행), **다중 조건 커버리지**(모든 개별식 조건의 모든 조합 고려)
- **명세 기반 테스트**(블랙박스 테스트): **동등 분할**(그룹핑), 경계값 분석, 결정 테이블, 유스케이스, 상태 전이
- **경험 기반 테스트**: 테스터의 경험을 기반으로 수행(에러 추정, 체크 리스트, 탐색적 테스팅)
- **회복**(결함->실패 후 올바르게 복구), **안전**(침입으로부터 보호), **강도**(과부하 시 정상 실행), **구조**(논리적 경로, 복잡도), **회귀**(변경된 코드에 새로운 결함 여부), **병행**(변경/기존 SW에 동일한 데이터 입력 후 비교)
- **맥케이브 순환복잡도:** 제어 흐름의 정보를 정량적을 표시하는 기법, 간선 수 – 노드 수 + 2
- **테스트 하네스:** 테스트를 지원하기 위한 코드와 데이터를 의미, 개발자가 테스트를 위해 작성함
- **테스트 오라클**: 테스트의 결과값이 참인지 거짓인지 판단하기 위해 사전에 정의한 참 값과 비교

  참(모든 입력값), **샘플링**(몇 개 입력값만), **휴리스틱**(나머진 추정), **일관성 검사**(변경 전후 비교) 오라클
- **성능 테스트 도구:** APP 처리량, **응답 시간**(작업요청~응답도착), **경과 시간**(작업의뢰~처리완료) 테스트
- **자원 사용률**: APP이 작업을 처리할 동안의 CPU, MEM, DISK 등의 사용량
- **외계인 코드**: 아주 오래되거나 참고문서 또는 개발자가 없어 유지보수 작업이 매우 어려운 코드
- **스파게티 코드**: 실행은 되지만 소스 코드가 얽혀 있어 구조를 파악하기 힘든 코드

<8. SQL 응용>

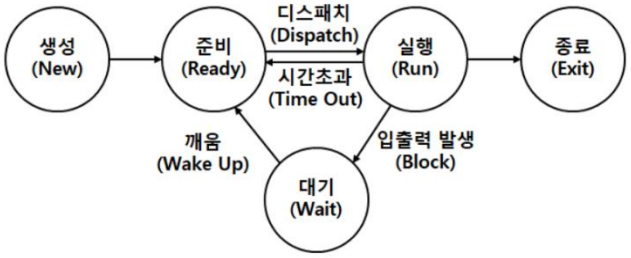
- **절차형 SQL**: 특정 사용자가 실행 순서를 정해놓고 사용하는 SQL문
- **프로시저**: 사전에 정의해놓은 SQL문을 호출할 경우 특정 작업을 수행하는 서브 프로그램(값 반환 X)
- **트리거**: 테이블에 수정/삭제/삽입 등의 이벤트 발생 시 자동으로 DBMS에서 실행되도록 한 프로그램
- **사용자 정의 함수:** 일련의 SQL 작업을 연속적으로 처리, 처리 결과를 단일 값으로 반환하는 절차형 SQL
- **MyBatis**: SQL 쿼리를 별도 XML 파일로 분리, 매핑을 통해 SQL 실행하는 Spring/자바 프레임워크

<9. 소프트웨어 개발 보안 구축>

- 보안의 3요소: **기밀성**(인가자만 접근), **무결성**(인가자만 수정), **가용성**(인가자는 언제나 사용 가능)
- **인증**: 시스템 내 정보/자원을 사용하려는 사람이 합법적인 사용자인지 확인하는 모든 행위
- **부인 방지:** 데이터를 송수신한 자가 송수신 사실을 부인할 수 없도록 송수신 증거 제공
- **nmap:** 서버에 열린 포트 정보를 스캐닝해서 보안 취약점을 탐지하는 도구, 해커들이 공격 전 주로 사용
- **DOS 공격**: 시스템을 악의적으로 공격해 해당 시스템의 자원을 부족하게 하여 사용 못하게 하는 공격
- **DDOS 공격:** 분산 배치된 공격자가 동시에 특정 시스템 공격(핸들러, 에이전트, 마스터, 데몬 프로그램)
- **XSS(크로스 사이트 스크립트):** 사용자가 해당 웹페이지를 열람함으로써 악의적인 스크립트가 실행되도록
- **SQL삽입:** 유효성 검증을 하지 않을 경우, 입력창/URL에 SQL 삽입하여 DB열람/조작을 가하는 공격
- **CSRF:** 사용자가 자신의 의지와는 무관하게 공격자가 의도한 행위를 웹사이트에 요청하도록 만드는 공격
- **SYN 플러딩:** TCP의 3-Way-Handshake의 약점을 노린 공격자가 다량의 SYN 패킷을 보내는 공격
- **TFN**(**T**ribe **F**lood **N**etwork): 많은 소스에서 여러 목표 시스템에 서비스 거부 공격하는 분산 공격 도구
- **스머핑:** 고성능 컴퓨터로 제3의 사이트를 이용해 엄청난 양의 ICMP Echo 패킷을 보내 마비시키는 공격
- 헷갈림 주의: **스니핑**(도청), **스누핑**(가로채감), **스머핑**(마비시키기), **스푸핑**(승인받은 것처럼 IP 위조)
- **PoD(Ping of Death)**: 허용 범위 이상 크기의 패킷을 고의적으로 전송하는 서비스 거부 공격
- **RUDY**: 요청 헤더의 Content-length를 비정상적으로 크게, 메시지 body 부분은 매우 소량으로 연결
- **Land Attack**: 출발지IP와 목적지IP를 같은 패킷주소로 만들어 보냄->자기 자신에게 응답을 보내게 함
- **Tear Drop**: IP패킷 재조합 과정에서 잘못된 Fragment Offset 정보로 인해 문제를 발생시키도록 하는 공격
- **Worm**: 네트워크를 통해 연속적으로 자신을 복제하여 시스템의 부하를 높이는 바이러스
- **Key Logger Attack**: 컴퓨터 사용자의 키보드 움직임을 탐지해 개인 정보를 몰래 빼가는 해킹 공격
- **스턱스넷:** 독일 지멘스의 SCADA 시스템의 제어 SW에 침투하여 시스템을 마비시킨 악성 코드
- **Crimeware**: 온라인에서 불법 활동을 조장하기 위해 만들어진 프로그램. =공격용 툴킷
- **Honey Pot:** 침임자를 속이는 침입탐지기법, 공격을 당하는 것처럼 보이게 하여 크래커를 추적한다.
- **Buffer Overflow:** 정해진 메모리의 범위를 넘치게 해서 임의의 프로그램이나 함수를 실행시키는 공격
- **MAC(**강제접근통제, 주체/객체 등급/권한을 비교), **DAC**(임의접근통제, 신분), **RBAC**(역할기반, 역할에 따라)
- **WAF**(**W**eb **A**pplication **F**irewall): 웹방화벽, 웹APP 보안에 특화되어 개발된 솔루션. 웹 공격 탐지 및 차단
- **벨 라파듈라 모델**: 미 국방부를 위해 개발된 모델, 기밀성 강조(No Read Up, No Write Down)
- **비바 모델**: 벨 라파듈라 모델의 단점인 무결성을 보장할 수 있는 모델(No Read Down, No Write Up)
- **DLP**(**D**ata **L**oss **P**revention): 데이터 유출 방지. 기업 내에서의 정보가 외부로 유출되는 것을 방지

- **APT**(**A**dvanced **P**ersistent **T**hreat, 지능형 지속 공격): 조직이나 기업을 표적으로 정한 뒤 장기간에 걸쳐 다양한 수단을 총동원하는 지능적 해킹 방식. 지속적으로 정보를 수집하고 취약점을 분석하여 공격함.
- **사이버 킬체인**: 록히드 마틴의 APT 공격 방어 분석 모델. 7단계 프로세스별 공격분석 및 대응을 체계화
- **대칭키 - 블록암호**: **DES**(56Bits키+64Bits블록)/**AES**(DES 대체)/**SEED**(KISA가 구현, **128Bits**)/**AREA**(국정원)
- **대칭키 - 스트림암호**: **RC4**(취약한 것으로 밝혀짐)
- **비대칭키(공개키+개인키): RSA**(소인수 분해의 어려움을 이용하여 암호화). *대칭키/비대칭키 모두 양방향
- **단방향 암호화:** 암호화 수행은 가능하지만 절대로 복호화가 불가능한 알고리즘(HASH).
- **HASH: MD4**, **MD5**, **SHA**(미국 국가 표준), **HAS-160**(MD5+SHA1 장점 합침, 국내 표준 서명 알고리즘)
- **솔트**: 단방향 해시 함수에서 추가 입력으로 사용되는 임의의 문자열(원문에 가미)
- **Seven Touchpoint**: 7가지 보안 강화 활동을 정의한 소프트웨어 개발 보안 방법론
- **Secure Coding**: 설계 및 구현 단계에서 보안 취약점을 사전에 제거하고 안전한 SW를 만드는 코딩 기법

<11. 응용 SW 기초 활용>



- **Shell**: 커널과 사용자 간을 연결하는 명령어 창
- **커널**: 프로세스/파일/입출력 관리 등 여러 기능을 수행
- **PCB**(**P**rocess **C**ontrol **B**lock): OS가 프로세스의 정보를 표현
- **MMU**(**M**emory **M**anagement **U**nit): 가상 메모리를 실제 메모리 주소로 변환해주는 장치
- **문맥 교환**: CPU가 실행중인 프로세스의 상태를 PCB에 저장하고 다음 프로세스의 PCB로부터 문맥 복원
- **응답률**: (서비스 시간+대기 시간) / 서비스 시간 -> **HRN**(**H**ighest **R**esponse-ration **N**ext) 응답률 순
- **선점형 스케줄링**: 우선순위가 높은 다른 프로세스가 현재 프로세스 중단 후 CPU를 점유하는 방식
- **라운드 로빈**: 프로세스가 할당 시간동안 처리되지 않으면 큐의 맨 뒤로 이동하고 다음 프로세스 할당
- **SRT**(**S**hortest **R**emaining **T**ime): 가장 짧은 시간이 소요되는 프로세스를 먼저 처리 (SJF의 선점 형태)
- **LRU**(**L**east **R**ecently **U**sed), **NUR**(Not Used Recently, 최근에 사용하지 않은 페이지 교체)
- 비선점형 스케줄링: 우선순위, **FCFS**, **SJF(S**hort **J**ob **F**irst**)**, **HRN**(응답률이 가장 높은 프로세스 먼저)
- **Thrashing**(스래싱): 페이지 부재의 지속으로 실제 처리 시간보다 페이지 교체 시간이 더 길어지는 현상
- **Woking Set**: OS의 가상기억장치 관리에서 프로세스가 일정 시간동안 자주 참조하는 페이지들의 집합
- **교착 상태(Deadlock):** 두 개 이상의 프로세스들이 서로의 자원을 요구하며 무한정 기다리는 현상
- 교착 상태의 발생 조건: 상호배제, 점유와 대기, 비선점, 환형 대기
- **은행가 알고리즘**: 교착 상태 회피 기법, 안정 상태 일때만 자원을 프로세스에 할당
- **컴퓨터 클라우딩**: 네트워크를 통해 어디서든 자원에 접속할 수 있는 기술(**IaaS**, **PaaS**, **SaaS**)

- 데이터베이스의 종류: **HDBMS**(계층형), **NDBMS**(망형), **RDBMS**(관계형)
- **개체-관계 다이어그램**(**ERD**, E-R 다이어그램) 기본 요소: **개체(엔티티)**, **속성**, **관계**
- **트랜잭션**: DB시스템에서 하나의 논리적인 기능을 수행하기 위한 작업의 기본 단위
- **SUPER KEY**:유일성은 만족, 최소성은 불만족
- **ACID: A**tomicity(**원자성**, 전체가 실패 or 성공), **C**onsistency(**일관성**, 트랜잭션 후 항상 일관된 DB 상태 유지), **I**solation(**독립성**, 트랜잭션 실행 중 다른 트랜잭션 접근 불가), **D**urability(**영속성**, 영구적으로 반영됨)
- 트랜잭션 용어: **COMMIT**(트랜잭션 확정), **ROLLBACK**(작업 취소->전 상태로), **CHECKPOINT**(트랜잭션 저장)
- **병행제어** 기법: 로킹, 낙관적 검증(일단 수행부터), 타임 스탬프(부여된 시간에 따라)
- **Locking**: 하나의 트랜잭션이 데이터를 액세스 하는 동안 다른 트랜잭션이 액세스할 수 없도록 제어
- **HINT:** SQL문제 액세스 경로 및 조인 순서 등의 정보를 사전에 줘서 빠른 실행 결과를 만드는 기법
- A에게 B테이블 SELECT문 이용할 수 있는 권한 부여(+다른사람에게 권한 부여할 수 있는 권한)

  GRANT SELECT ON B TO A WITH GRANT OPTION;
- **NoSQL**: 전통적 RDBMS와 달리 테이블 스키마가 필요하지 않고 수평적 확장이 가능한 DBMS
- **Ontology**: 실존하는 개념/속성/관계 및 여러 개념을 컴퓨터가 처리할 수 있는 형태로 추상적으로 표현
- **시맨틱 웹**: Ontology를 활용하여 서비스 검색/조합 기능들을 자동화하는 웹
- **데이터 마이닝**: 대규모로 저장된 데이터에서 체계적이고 자동적인 방법으로 규칙/패턴을 찾아내는 기술
- **WAN**: 전송 거리가 넓고 라우팅 알고리즘이 필요한 광대역 네트워크(vs LAN)
- **백본망**(**Backbone Network**): 각기 다른 LAN이나 부분망 간에 정보를 교환하기 위한 경로를 제공하는 망
- **OSI 7계층:** 국제표준화기구(ISO)에서 개발, 네트워크 프로토콜 디자인과 통신을 계층으로 나눠 설명한 것
- 1. **물리**: 실제 장치들 연결 – 비트 - 리피터(입력 신호 증폭), 허브(가까운 컴퓨터들을 연결)
- 2. **데이터링크**: 각종 흐름, 순서 제어 – 프레임 - **브리지**(LAN 연결), **스위치**(큰 LAN 생성) – PPP, Ethernet
- 3. **네트워크**: 라우팅, 혼잡 제어 – 패킷 – **라우터**(경로 설정) – IP, IGMP, BGP, ARP, RARP
- 4. **전송**: 종단 시스템 간 투명한 데이터 전송 – 세그먼트 – **게이트웨이**(서로 다른 프로토콜) - TCP, UDP
- 5. **세션**: 동기점(오류가 있는 데이터의 회복), 대화 제어자 – RPC, SSL/TLS(인증/암호화/무결성)
- 6. **표현**: 통신에 적당한 형태로 변환, 데이터의 암호화/복호화 – SSL/TLS, JPEG
- 7. **응용**: 사용자에게 편리한 환경 제공 – HTTP, HTTPS, **SMTP**(이메일), POP3, DNS, **FTP**(파일 전송), SSH
- **프로토콜**: 이기종 시스템 간 데이터 교환을 할 수 있도록 하는 표준화 통신 규약 (**구문**, **의미**, **타이밍**)
- **TCP**: 근거리 통신망이나 인터넷에서 신뢰성있는 전송을 수행하는 프로토콜
- **UDP:** 비연결성이고 신뢰성이 없으며 순서화되지 않은 데이터그램 서비스를 제공하는 통신 프로토콜

- **ARP:** IP 주소를 물리적 주소(=MAC 주소)로 변경 vs **RARP**: 물리적 주소(=MAC 주소)를 IP 주소로 변경
- **IGMP**(**I**nternet **G**roup **M**anagement **P**rotocol): 멀티캐스트 그룹 유지를 위해 사용되는 프로토콜
- **ICMP**(**I**nternet **C**ontrol **M**essage **P**rotocol): 오류 메시지 등 여러 정보를 전달/컨트롤하는 프로토콜
- **LDAP**(**L**ightweight **D**irectory **A**ccess **P**rotocol): 디렉터리 서비스의 등록/수정/삭제 및 검색 프로토콜
- **DHCP:** Dynamic Host Configuration Protocol): IP 주소 및 설정 정보를 클라이언트에게 동적으로 할당
- **SSH**(**S**ecure **SH**ell): 포트번호 22번, 높은 안정성을 보장하는 원격 접속 프로토콜
- **IPv4**: 데이터 교환을 위해 32bit 주소 체계를 갖는 네트워크 계층의 프로토콜(유니/멀티/브로드캐스트)
- **IPv6**: 차세대 인터넷 프로토콜, IPv4 문제점 해결, 128bit 주소체계(유니/멀티/애니캐스트)
- **유니캐스트:** 단 하나의 수신자에게 1대1로 정보를 전송
- **멀티캐스트:** 같은 내용의 데이터를 여러 명의 특정한 일부 그룹의 수신자들에게 동시에 전송하는 기술
- **브로드캐스트:** 하나의 송신자가 같은 서브네트워크 상의 모든 수신자에게 데이터를 전송하는 기술
- **애니캐스트**: Topology상의 수신자 그룹 안에서 가장 가까운 노드로 데이터그램을 연결시키는 프로토콜
- **DNS**(**D**omain **N**ame **S**ystem): 도메인 네임을 IP 주소로 매핑하는 시스템
- **네트워크 슬라이싱**: 하나의 네트워크를 다수의 독립적 가상 네트워크로 분리, 전용 네트워크 제공
- **패킷 스위칭**: 작은 블록의 패킷으로 데이터 전송, 전송 동안만 네트워크 자원 사용(X.25, ATM, 프-릴)
- **X.25**: 두 단말 장치가 패킷 통신망을 통해 원활한 패킷 전송을 하기 위한 프로토콜
- **프레임 릴레이**: 프로토콜 처리 간략화, 데이터프레임들의 중계/다중화 기능만 수행->처리 속도 향상
- **ATM**: 53바이트 셀 단위로 전달하는 비동기식 시분할 다중화 방식의 패킷형 전송 기술
- **서킷 스위칭**: 네트워크 리소스를 특정 사용 계층이 독점하여 통신하는 방식
- **SAN**(**S**torage **A**rea **N**etwork): 여러 OS의 기종들이 동일 저장장치의 데이터 공유->백업 장비 단일화
- **NAT**(**N**etwork **A**ddress **T**ranslation): 사설 IP 주소를 공인 IP주소로 바꿔주는 네트워크 주소 변환 기술
- **MQTT(M**essage **Q**ueuing **T**elemetry **T**ransport): 제한된 대역폭의 푸시기술 기반 경량 메시지 전송 규약
- **라우팅 프로토콜**: 최종 목적지까지 패킷을 최소 비용/홉수로 적절한 경로를 설정해주는 프로토콜
- **거리 벡터 알고리즘:** 인접 라우터와 정보를 공유하여 목적지까지의 방향과 거리를 결정(RIP, IGRP)
- **RIP:** 거리 벡터 알고리즘(벨만 포드), 최대 홉수를 15개로 제한, 자율시스템(Auto System) 내부
- **IGRP:** 거리 벡터 알고리즘, RIP의 문제점 개선, 대규모 통신망
- **OSPF**: 링크 상태 알고리즘, 모든 라우터에 정보 전달, 다익스트라, 대규모 네트워크, 홉수 제한X
- **BGP:** 경로 벡터 알고리즘**,** 대규모 네트워크, 도달 가능성 및 라우팅 정보 교환
- **VPN(V**irtual **P**rivate **N**etwork**)**: 인터넷과 같은 공중망을 마치 전용선으로 사설망을 구축한 것처럼 사용

<12. 제품 소프트웨어 패키징>

- **릴리즈 노트**: 최종 사용자인 고객에게 잘 정리된 릴리즈 정보를 현재 시제로 제공하는 문서(헤더+개요)
- 릴리즈 노트 작성 프로세스: 모듈 식별 > 릴리즈 정보 확인 > 릴리즈 노트 개요 작성 > 영향도 체크 > 정식 릴리즈 노트 작성 > 추가 개선 항목 식별
- **DRM**(**디지털 저작권 관리**): 디지털 매체를 통해 유통되는 데이터의 저작권을 보호하기 위한 시스템
- **Clearing House:** 저작권에 대한 사용 권한, 라이선스 발급, 결제 관리 등을 수행하는 곳
- **PKI(P**ublic **K**ey **I**nfrastructure**)**: 공개키 암호화 방식 기반 디지털 인증서를 활용하는 SW, HW 등을 총칭
- **DOI**(**D**igital **O**bject **I**dentifier): 디지털 저작물에 특정 번호를 부여하는 일종의 바코드 시스템
- **URI**(**U**niform **R**esource **I**dentifier)**:** 인터넷에 있는 자원을 나타내는 유일한 주소
- **핑거프린팅**: 멀티미디어 콘텐츠에 저작권 정보와 구매한 사용자 정보를 삽입->불법 배포자 추적 기술
- **MPEG-21**: 멀티미디어 프레임워크 표준 규격 * **XrML**: XML + 권리 조건 표현
- **코드 난독화**: 역공학을 통한 공격을 막기 위해 프로그램 소스를 알아보기 힘든 코드로 바꾸는 기술
- **역공학**: 시스템의 구조 분석을 통해 기존에 개발된 시스템의 기술적인 원리를 도출해내는 작업
- **Secure DB**: 커널 암호화 방식으로 데이터베이스 파일을 직접 암호화, 접근제어/보안적 요소 추가된 기술
- **CMS**: 다양한 미디어 포맷에 따라 각종 콘텐츠 생산에서 활용, 폐기까지 전 공급 과정을 관리하는 기술
- **형상 관리**: SW 변경 사항을 관리하기 위해 개발된 일련의 활동(**체크아웃**, **체크인**-업로드, **커밋**-서버 반영)
- 형상 관리 절차: 형상 식별 -> 형상 통제(Base line 조정) -> 형상 감사(공식적으로 승인) -> 형상 기록
- **Base line**: SW 변경 통제 시점
- 버전 관리 도구: 공유 폴더(RCS), 클라이언트/서버(CVS, SVN), 분산 저장소(Git, Bitkeeper) 방식
- **SVN(Subversion)**: CVS를 개선, 아파치에서 발표, 개발 작업은 trunk 디렉터리에서 수행, revision 1씩 증가

<신기술 동향 & 기타>

- **Map reduce**: 구글에서 대용량 데이터를 분산 병렬 컴퓨팅에서 처리하기 위한 목적으로 제작(Hadoop)
- **RTO**(**R**ecovery **T**ime **O**bjective): 복구 목표 시간, 서비스를 복구하는 데까지 걸리는 최대 허용 시간
- **RPO**(**R**ecovery **P**oint **O**bjective): 복구 목표 시점, 감내 가능한 손실 허용 시점, 어느 시점으로 백업할지
- **BCP**(**B**usiness **C**ontinuity **P**lanning): 장애 및 재해로부터 조직의 생존을 보장하기 위한 예방/복구 계획
- **DRP**(**D**isaster **R**ecovery **P**lanning): **재해 복구 계획**, 재해에 대비하여 빠른 복구를 위한 제반 계획
- **DRS**(**D**isaster **R**ecovery **S**ystem): **재해 복구 시스템**, 재해 복구 계획의 원활한 수행을 지원하기 위해 평상시에 확보하여 두는 인적/물적 자원 및 이들에 대한 지속적인 관리 체계가 통합된 것
- **Mirror Site:** 주 센터와 데이터복구센터 모두 운영 상태로 실시간 동시 서비스가 가능한 재해 복구 센터

- **Hot Site**(동일한 수준 Stanby), **Warm Site**(중요한 자원만, 수 시간) **Cold Site**(최소한, 수 일 주기 백업)
- **디지털 포렌식(**디지털 법의학): 범죄 사실에 대한 디지털 증거자료를 수집/복사/제출하는 일련의 과정
- **애자일 개발 방법론:** 변화에 유연하고 절차보다는 사람 중심인 신속 적응적 경량 개발 방법론
- **럼바우 방법:** 분석 활동을 객체 모델링/동적 모델링/기능 모델링 으로 나누어 수행하는 방법
- **Tailoring:** 프로젝트의 필요에 따라 SW 개발 프로세스/기법 등을 비즈니스/기술적 요구에 최적화
- **NAS(N**etwork **A**ttached **S**torage): 서버와 저장 장치를 네트워크로 연결하는 방식
- **MVC 패턴:** 모델, 뷰, 컨트롤러로 구성되어 비즈니스 로직을 서로의 영향 없이 쉽게 고칠 수 있는 패턴
- **멀티 프로그래밍**: 2개 이상의 프로그램을 CPU가 번갈아 사용해가며 자원 활용률을 극대화하는 기법
- **ETL**(**E**xtraction **T**ransformation **L**oading): 데이터를 추출하고 변환하고 전송 및 로딩하는 이동 프로세스
- **XP:** 비즈니스 요구 변동이 심할 때 적합(원리- 짝프로그래밍, 리팩토링, 테스트 기반 개발, 작은 릴리즈)
- XP의 특징**:** 의사소통, 단순성, 피드백, 용기, 존중
- **Secure SDLC**(Secure **S**ofrware **D**evelopment **L**ife **C**ycle): 보안 소프트웨어 개발 생명 주기
- **HA(고가용성)**: 서버와 네트워크, 프로그램 등 정보시스템이 오랜 기간 동안 계속 정상 운영 가능한 성질
- **TELNET:** 원격의 컴퓨터에 접속하여 자신의 컴퓨터처럼 사용할 수 있도록 해주는 프로토콜
- **하이퍼바이저:** 하나의 컴퓨터에서 동시에 다수의 OS를 구동시킬 수 있는 SW 가상화 플랫폼
- **데이터 웨어하우스**: 다량의 데이터를 효과적으로 분석하여 정보화하고 효율적으로 사용할수 있게 된 DB
- **에이징 기법:** 기다린 시간에 비례하여 프로세스의 우선 순위를 높여주는 기법
- **리팩토링:** 기능은 수정하지 않고 복잡한 소스코드를 수정/보완하여 가독성/유지보수성 높이는 기법
- **NAC**(**N**etwork **A**ccess **C**ontrol): 내부 PC의 MAC 주소를 IP관리 시스템에 등록 후 보안 관리 기능 제공
- **쿠버네티스**: 컨테이너화된 APP의 자동 배포, 스케일링 등 제공하는 오픈소스 관리시스템(리눅스 재단)
- **CLASP**(**C**omprehensive, **L**ightweight **A**pplication **S**ecurity **P**rocess): 이미 운영중인 시스템에 적용하기 좋음
- **척와:** 비정형 데이터 수집기술, 분산된 각 서버에서 에이전트를 실행하고, 컬렉터가 에이전트로부터 데이터를 받아 HDFS(하둡 File system)에 저장하는 기술
- **스쿱**: 정형 데이터 수집기술, 커넥터를 이용하여 RDBMS에서 HDFS로 데이터를 수집하는 기술
- **MEMS:** 초정밀 반도체 제조기술 바탕, 초미세 장치
- **Stack Guard**: 메모리 상에서 프로그램 복귀 주소와 변수 사이에 특정 값 저장->변경되면 실행 중단
- **HIPO**: 하향식 소프트웨어 개발을 위한 문서화 도구