1장 요구사항 확인

1. 현행시스템 파악

- 시스템 구성 (구성현황, 기능, 인터페이스)

- 소프트웨어 구성, (아키텍처, 소프트웨어)

- 하드웨어 구성 (시스템 하드웨어, 네트워크)

2. 웹 어플리케이션 서버(WAS)

- 사용자의 요구에 따른 변하는 동적인 콘텐츠를 처리하기 위해 사용되는 미들웨어

3. 요구사항

- 기능요구사항, 비기능사항 구분하기

- 개발 프로세스 : 도출 – 분석 – 명세 – 확인

4. 요구공학

- 무엇을 개발해야 하는지 요구사항을 정의하고 분석 및 관리하는 프로세스를 연구하는 학문

- 요구사항 변경의 원인과 처리 방법을 이해하고 요구사항 관리 프로세스의 품질 개선, 프로젝트 실패를 최소화하는 것을 목표로 한다.

5.요구사항 도출

- 인터뷰 브레인스토밍 워크샵 프로토타이핑 유스케이스

6. 요구사항 분석

- 요구사항 중 명확하지 않거나 모호하여 이해되지 않는 부분을 발견, 이를 걸러냄

7. 요구사항 명세

- 요구사항을 체계적으로 분석한 후 문서화하는 것

8. 요구사항 확인

- 요구사항 명세서가 정확하고 완전하게 작성되었는지를 검토하는 활동

9. 요구사항 분석기법

- 개념 모델링 :현실 세계의 상황을 단순화하여 개념적으로 표현하는 과정

- 정형 분석 : 구문과 의미를 갖는 정형화된 언어를 이용해 요구사항을 수학적 기호로 표현 분석

10 요구사항 확인

* 프로토타이핑 : 개발 대상 시스템 또는 일부 기능을 간단하게 만든 원형, 동적인 모형
* 장점 : 의사소통도움, 피드백 가능, 요구사항 개선 가능
* 단점 : 상대성 과대평가, 비용부담, 프토타입 제작에만 집중 될 수 잇음

11. UML

- 시스템 개발자와 고객 간의 상호간의 효율적인 의사소통이 이루어지도록 표준화한 객체지향 모델링 언어

- 연관관계 : 2개 이상의 사물이 서로 관련

- 집합관계 : 하나의 사물이 다른 사물에 포함되어 있는 관계

- 포함관계 : 집합의 특수한 관계, 포함하는 사물의 변화가 포함되는 사물에게 영향을 미침, 생명주기 함께함

- 일반화 관계 : 하나의 사물이 다른 사물에 비해 더 일반적인지 구체적인지를 표현

- 의존관계 : 사물 사이에 연관은 있으나 필요에 의해 서로에게 영향을 주는 짧은 시간 동안만 연관을 유지, 사물의 변화가 다른 사물에도 영향

- 실체화 관계 : 사물이 할 수 잇거나 해야 하는 기능으로 서로를 그룹화 할 수 있는 관계

- 사물에는 구조사물(물리적 요소), 행동사물(행위), 그룹사물(요소들을 그룹화), 주해(설명, 조건)

12. 다이어그램

구조적 다이어그램

* 클래스다이어그램 : 클래스가 가지는 속성, 클래스들 사이의 관계
* 배치 다이어그램 : 구현단계, 프로세스, 컴포넌트 등의 물리적 요소의 위치
* 컴포넌트 다이어그램 : 구현단계, 컴포넌트 간의 인터페이스를 표현

행위 다이어그램

* 유스케이스 : 사용자의 요구를 분석
* 시퀀스 다이어그램 : 객체들이 주고받는 메시지를 표현
* 상태 다이어그램 : 하나의 객체가 속한 클래스의 상태 변화, 객체와의 상호작용 표현
* 활동 다이어그램 : 시스템이 어떤 기능을 수행하는지 객체의 처리의 흐름 표현

13. 유스케이스

- 사용자의 요구사항을 분석하여 필요한 기능들을 정리하고 이를 공유하기 위해 표현하는 것

- 액터 : 시스템과 상호작용하는 외부 요소, 사람

- 유스케이스 : 시스템이 엑터에게 제공하는 서비스 또는 기능을 표현

- 포함관계 : <<include>>, 공통적으로 적용되는 기능을 별도로 분리, 새로운 유스케이스로

확장관계 : <<extends>> 유스케이스의 기능이 확장될 때 원래의 유스케이스와 확장된 유스케이스와 관계

14. 활동 다이어그램

* 액션은 더 이상 분해할 수 없는 단일 작업
* 액티비티는 몇 개의 액션으로 분리될 수 있는 작업
* 스윔레인은 액티비티 수행을 담당하는 주체를 구분한다.
* 포크노드 : 액ㅌ티비티의 흐름이 분리되어 수행
* 조인노드 : 분리되어 수행되던 액티비티의 흐름이 다시 합쳐짐

15. 클래스 다이어그램

- 정적 모델링, 요구한 기능을 구현하는데 필요한 자료들의 논리적인 구조 표현

- 시스템을 구성하는 클래스, 클래스의 특성 속성, 속성과 오퍼레이션에 대한 제약조건

- 클래스 사이의 관계

16. 클래스

- 3개의 구역으로 나뉜다. 이름, 속성, 오퍼레이션

- 이름은 반드시 입력되어야 한다.

- 속성은 클래스의 상태나 정보를 표현한다.

- 접근제어자 (- private , + public , # protected , ~ package)

- 대부분 일반화 관계가 나온다. 상위 클래스의 구체적인 개념을 하위 클래스가 사용하는 것

17. 시퀀스 다이어그램

- 시스템이나 객체들이 메시지를 주고받으며 시간의 흐름에 따라 상호작용 하는 과정을 액터, 객체, 메시지 등의 요소를 사용하여 표현

- 액터는 시스템으로부터 서비스를 요청하는 외부 요소, 사람

- 객체는 메시지를 주고받는 주체 <: >

- 라이프라인은 객체가 메모리에 존재하는 기간으로 객체 아래 점선을 그어 표현한다.

- 메시지 종료( 동기(기다림), 비동기(다른짓), 생성(새로운 객체 생성),

응답(동기 메시지에 대한 수행 결과))

18. 비용산정 모델

- 하향식 선정기법 (경험) : 전문가 판단, 델파이 기법

- 상향식 선정기법 : LOC, ManMonth, COCOMO, FP (기능점수)

19. COCOMO

- 조직형 : 5만이하 중,소규모 소프트웨어

- 중간형 : 30만 이하

- 내장형 : 30만 이상 초대형 규모

20. 스테레오 타입

- 경계 : 시스템과 외부 액터와의 **상호작용**을 담당

- 엔티티 : 시스템이 유지해야 하는 **정보를 관리**하는 기능을 전담하는 클래스

- 제어 : 시스템이 제공하는 기능의 로직 및 제어를 담당하는 클래스

2장 데이터 입출력

1. 데이터 모델

- 현실 세계의 정보들을 컴퓨터에 표현하기 위해서 단순화, 추상화하여 체계적으로 표현한 개념적 모델

2. 데이터모델 구성요소

- 개체 : 데이터베이스에 표현하려는 것, 현실세계의 대상체

- 속성 :데이터의 가장 작은 논리적 단위, 파일 구조상에서 항목또는 필드에 해당

- 관계 : 개체 간의 관계 또는 속성 간의 논리적인 연결

표시 요소

- 구조 : 개체 타입들 간의 관계, 데이터 구조 및 정적 성질 표현

- 연산 : 실제 데이터를 처리하는 작업에 대한 명세, DB조작 기본 도구

-제약조건 : 실제 데이터의 논리적인 제약조건

3. 개념적 데이터 모델

- 현실 세계에 대한 인식을 추상적으로 표현하는 과정

- 대표적인 ER모델

4. 논리적 데이터 모델

- 개념적 구조를 컴퓨터가 이해하고 처리 할 수 있는 컴퓨터 세계의 환경에 맞도록 변환하는 과정

- 데이터 타입들 간의 관계를 이용하여 현실 세계를 표현

5. 이상현상

- 일부 속성들의 종속으로 인해 데이터의 중복이 발생하여 테이블 조작 시 문제가 발생하는 현상

- 삽입이상 : 테이블에 데이터를 삽입할 때 의도와는 상관없이 원하지 않는 값들로 인해 삽입x

- 갱신이상 : 테이블에 속성 값을 갱신할 때 일부 튜플의 정보만 갱신되어 정보 불일치가 일어남

- 삭제이상 : 테이블에서 한 튜플을 삭제할 때 의도와는 상관없는 값들도 함께 삭제, 연쇄삭제

6. 함수적 종속

- X의 값에 따라 항상 Y가 오직 하나만 연관되는 경우 X(결정자) -> Y(종속자)

-완전 함수적 종속 :

7. 정규화

1NF : 모든 속성의 도메인이 원자 값

2NF : 기본키가 아닌 모든 속성이 기본키에 대하여 완전 함수적 종속을 만족, 부분 함수적 종속제거

3NF :기본키가 아닌 모든 속성이 기본키에 대해 이행 함수적 종속을 만족하지 않는 정규형, 이행함수적 종속 제거

BCNF : 모든 결정자가 후보키인 정규형

4NF : 테이블 R에 다중 값 종속

5NF : 테이블의 모든 조인종속

8. 테이블

- 로우 : 튜플, 인스턴스, 어커런스

-- 컬럼 : 속성

9. 엔티티 변환

엔티티 -> 테이블

속성 -> 컬럼

주식별자 -> PK

외부식별자 -> FK

관계 -> 관계

10.속성 -> 컬럼 변환 주의사항

Primary UID -> PK

Secondary UID -> Unique key

11. 관계 -> 외래키 시

1:1 -> A의 PK를 B의 FK

1:N -> A의 기본키를 B의 FK, 별도 테이블

N : M -> A의 PK와 .B의 PK를 포함한 별도의 릴레이션(교차 릴레이션)

11. 반정규화

- 시스템의 성능 향상을 위해 정규화된 데이터 모델을 통합, 중복, 분리하는과정, 의도적으로 정규화 원칙을 위배

12. 인덱스

- 데이터 레코드를 빠르게 접근하기 위해 <키 값, 포인터> 쌍으로 구성되는 데이터 구조

- 물리적 구조에 접근하는 방법을 제공

- 클러스터드 인덱스 : 인덱스 키의 순서에 따라 데이터가 정렬되어 저장되는 방식

- 넌클러스터 인덱스 : 인덱스의 키 값만 정렬되잇음

- 랜덤 엑세스가 빈번한 테이블에 용이

- 특정 범위나 특성 순서를 조회

- order by, group by, union이 빈번한 테이블

- 기존 엑세스에 영향을 미친다.

- 저장공간을 테이블과 분리되게 설계

13. 뷰

- 특정 자료만을 보여주기 위해 유도된 가상 테이블, 물리적으로 존재하지 않는다.

- 논리적 독립성을 제공, 삽입 삭제 갱신등에 제약이 따른다.

- 뷰의 정의는 변경할 수 없다

14. 클러스터

- 데이터 저장 시 데이터 엑세스 효율을 향상시키기 위해 동일한 성격의 데이터를 데이터 블록에 저장하는 방식

- 분포도가 넓을수록 용이

- 처리범위가 넓으면 단일 테이블 클러스터링

- 조인이 많으면 다중 테이블 클러스터링

- order by, group by, union이 빈번한 테이블

- 수정이 자주일어나면 안됨

15. 파티션

- 대용량의 테이블을 작은 논리적 단위인 파티션으로 나누는 것

- 범위 분할 : 특정 열의 값을 기준으로 분할

- 해시 분할 : 해시함수를 사용하여 분할, 특정데이터 위치 파악x

- 조합 분할 : 범위 분할 후 해시 분할

- 장점으로는 성능향상, 가용성 향상, 백업 가능, 경합 감소등의 효과가 있다.

3장 통합구현

1. 직접 연계 방식

- 중간 매개체 없이 송수신 시스템이 직접 연계하는 방식

- 단순하고 용이하다, 개발비용이 저렴하다, 개발기간이 짧다

- 결합도가 높다. 환경이 제한적이다.

- DB LINK : db에서 제공하는 DB Link객체를 이용

- API : 데이터를 송신 시 시스템의 DB에서 읽어와 제공하는 앱 프로그래밍 인터페이스

- DB Connection : WAS에서 송신 시스템의 DB로 연결

- JDBC : Java에서 DB에 접근하여 데이터를 작업할 수 있게 자바와 DB를 연결

2. 간접 연계 방식

- 송수신 시스템 사이에 중간 매개체를 두어 연계하는 방식

- 다양한 환경 통합가능

- 복잡하다. 개발기간이 비교적 길다

- 연계솔루션

- ESB : 애플리케이션 간 연계, 데이터 변환, 웹서비스 지원등 표준 기반의 인터페이슬 ㄹ제공

- Socket : 버서는 통신을 위한 소켓을 생성하여 포트를 할당하고 블라블라

- WebService : 웹 서비스에 WSDL, UDDI, SOAP 연결

3. EAI (Enterprise Application Integration)

- 기업 내 각종 애플케이션 및 플랫폼 간의 정보전달, 연계, 통합 등 상호 연동이 가능하게 해주는 솔루션

- point to point : 가장 기본적 앱간 1:1연결

- Hub Spoke : 허브시스템을 통해 데이터를 전송하는 중앙 집중형 방식

- 메시지 버스 : 애플리케이션 사이에 미들웨어를 두어 처리하는 방식

- 하이브리드 : 그룹 내에는 허브앤 스포크,, 그룹간애는 메시지 버스를 사용, 데이터병목현상 최소화

4. ESB

- 애플케이션 간의 연계, 데이터 변환, 웹 서비스 지원하는 솔루션

- 서비스 중심의 통합을 지향, 결합도를 낮추고, 느슨한 결합을 유지

5. XML

- 웹 브라우저 간 HTML 호환, SGML의 복잡함을 해결한 다목적 마크업 언어

- 텍스트 형식의 데이터 포맷, 다양한 플랫폼에 사용

6. JSON

- 속성 – 값 쌍으로 이루어진 개방형 표준 포맷

7. AJAX

- 자바스크립트와 XML이용한 비동기 통신방식

8. WSDL

- 웹 섭비스와 관련된 서식이나 프로토콜 등을 표준적인 방법으로 기술하는 XML

- UDDI의 기초

9. UDDI

- 전 세계 비즈니스 업체에서 자신의 정보를 등록하기 위한 XML기반 규격

10. SOAP

- HTTP/HTTPS, SMTP등을 이용하여 XML을 교환하기 위한 통신 규약

11. Restful

- HTTP/S와 Rest의 원칙을 사용하여 구현되는 웹 서비스 HTTP자원을 관리하는 사용되는 웹 서브 API의 집합

**4장 서버프로그램 구현**

1. 하드웨어 환경

- 웹서버 : 클라이언트로 직접 요청을 받아 처리하는 서버, 대역폭제한(트래픽 방지), 인증, HTTP/S 등을 지원한다.

- WAS : 사용자에게 동적 서비스를 제공하기 위해 웹 서버로부터 요청을 받아 인터페이스 역할

- DB서버, 파일서버

2. 소프트웨어 환경

- 요구사항 관리도구, 설계 모델링도구, 빌드 도구 등등

- 형상관리 도구 : 산출물들을 버전별로 관리하여 품질 향상을 지원하는 도구

3. 모듈

- 단독으로 컴파일이 가능하며 재사용이 가능하다.

- 모듈화를 통해 분리된 시스템의 각 기능, 작은 작업 단위

- 결합도는 약하게 응집도는 강하게 크기는 작게

4. 결합도 (내공외제스자)

- 모듈 간에 상호 의존하는 정도, 모듈 사이의 연관관계

- 높을수록 안좋고 낮을수록 좋다

- 자료 결합도 : 모듈 간의 인터페이스가 자료요소로만 사용

- 스탬프 결합도 : 모듈간의 인터페이스로 배열이나 자료구조가 전달

- 제어 결합도 : 뫃듈이 다른 모듈 내부의 논리적인 흐름을 제어하기 위해 제어신호를 이요ㅕㅇ하여 통신하거나 제어요소를 전달하는 결합도

- 외부 결합도 : 어떤 모듈에서 선언한 데이터 변수를 외부의 다른 모듈에서 참조

- 공유 결합도 : 공유되는 공통데이터 영역을 여러 모듈이 사용, 전역변수

- 내용 결합도 : 한 모듈이 가른 모듈의 내부 기능 및 그 내부자료를 직접 참조하는 경우

5. 응집도 (우논시절통순기)

- 정보은닉 개념 확장

- 모듈이 독집적인 기능으로 정의되어 있는 정도, 요소들이 서로 관련되어 있는 정도

- 기능적 응집도 : 모듈 내부의 모든 기능요소들이 단일 문제

- 순차적 응집도 : 모듈 내 하나의 활동으로 나온 출력 데이터가 다음 작업 입력데이터로 활용

- 교환적 응집도 : 동일한 입력과 출력을 사용하여 서로 다른 기능을 수행하는 구성 요소

- 절차적 응집도 : 모듈안의 구성 요소들이 그 기능을 순차적으로 수행할 겨웅

- 시간적 응집도 : 특정 시간에 처리되는 몇 개의 기능

- 논리적 응집도 : 유사한 성격을 갖거나 특정 형태로 분류

- 우연적 응집도 : 관련없음

6. 팬인/ 팬아웃

- 들어오면 팬인, 나가면 팬아웃, 숫자로 작성

7. 공통모듈

- 여러 프로그램에서 공통적으로 사용할 수 있는 모듈들

- 정확성 : 해당 기능이 필요하다는 것을 작성

- 명확성 : 해당 기능을 중의적으로 해석x

- 완전성 : 시스템 구현시 필요한 모든 것을 기술

- 일관성 : 공통 기능들 간 상호 충돌 발생x

- 추적성 : 기능에 대한 요구사항의 출처, 시스템간의 관계 파악정도

8. 재사용

이미 개발된 기능들을 파악하고 재구성하여 새로운 시스템 또는 기능 개발에 사용하기 적합하도록 최적화, 누구나 이해할 수 있고 사용이 가능하도록 사용법 공개

* 함수와 객체
* 컴포넌트
* 애플리케이션

9. 효율적인 모듈

- 낮은 결합도 높은 응집도

- 복잡도와 중복성을 줄이고 일관성 유지

- 유지보수 용이

- 하나의 입구와 하나의 출구

10. DBMS 접속

- 사용자가 데이터를 사용하기 위해 응용 시스템을 이용하여 DBMS에 접근

- 웹 응용시스템은 웹 서버와 웹 애플리케이션 서버로 구성된다. 사용자와 웹 서버에 접속하는 인터페이스 역할을 한다.

-

11. JDBC

- java언어로 데이터베이스에 접속하고 SQL문을 사용

- 특정 드라이버가 필요하다

12. ODBC

- 개발 언어에 관계 없이 사용가능

- ODBC문장을 사용하여 ODBC에 포함된 드라이버 관리자가 DBMS의 인터페이스에 맞게 연결해줌

13. Mybatis

- JDBC 코드를 단순화하여 사용할 수 있는 SQL Mapping 기반 오픈소스 접속 프레임워크

- SQL문장을 분리하여 XML 파일을 만들고 매핑

14. 동적SQL

- 다양한 조건에 따라 SQL구문을 동적으로 변경하여 처리할 수 있는 SQL처리방식

- 문자열 변수에 담아 동적 처리

- 속도가 느리다

15. 정적SQL

- 커서를 통해 정적처리를 하며 속도가 빠르다

16. 서버개발 프레임워크

-서버 프로그램 개발 시 다양한 네트워크 설정,요청 등을 손쉽게 처리할 수 있도록 클래스나 인터페이스를 제공하는 소프트웨어

- 대부분은 MVC패턴을 사용한다

- Spring – JAVA

- NodeJs – Javascript

- Django – Python

- Codeeigniter – PHP

- Ruby 머ㅏ시기 - .Ruby

17. 프레임워크 특징

- 모듈화 : 캡슐화를 통해 모듈화를 강화하고 변경에 따른 영향을 최소화

- 재사용성 : 재사용 가능한 모듈들을 제공함으로써 개발자의 생산성 향상

- 확장성 : 다형성을ㄹ 통한 인터페이스 확장이 가능하여 다양한 개발 가능

- 제어의 역행 : 객체들의 제어 권한을 프레임워크가 가져 생산성 향상

18. 배치프로그램

- 사용자와의 상호 작용 없이 여러 작업들을 미리 정해진 일련의 순서에 따라 일괄적으로 처리

- 정기배치 : 정해진 기간에 정기적으로 수행

- 이벤트성 배치 : 특정 조건을 설정해두고 조건이 충족 시

- 온디멘드 배치 : 사용자의 요청 시

19. 배치 스케쥴러

- 스프링배치 : 스프링사의 오픈소스 프레임워크, Job, Job런처, Step

- 쿼츠 : 일괄 처리;를 위한 다양한 기능을 제공하는 오픈소스, Trigger가 있다.

**5장 화면설계**

1. 사용자 인터페이스 UI

- 사용자와 시스템 간의 상호작용이 원할하게 이뤄지도록 도와주는 장치나 소프트웨어를 의미한다.

- CLI : 명령과 출력이 텍스트 형태로 이뤄지는 인터페이스

- GUI : 아이콘이나 메뉴를 마우스로 선택하여 작업을 수행하는 그래픽 환경의 인터페이스

- NUI : 사람의 말이나 행동으로 기기를 조작

- OUI : 모든 사물과 사용자 간의 상호작용을 위한 인터페이스

2. UI 기본원칙

- 직관성 : 누구나 쉽게 이해하고 사용할 수 있어야 한다.

- 유효성 : 사용자의 목적을 정확하고 완벽하게 달성

- 학습성 : 누구나 쉽게 배우고 익힐 수 잇어야 한다.

- 유연성 : 사용자의 요구사항을 최대한 수용하고 실수를 최소화

- 사용자 중심으로 이루어진다.

3. UI 설계도구

- 와이어프레임 : 페이지에 대한 개략적인 레이아웃이나 UI요소 등에 대한 뼈대를 설계, 화면설계

- 목업 : 와이어프레임보다 좀 더 실제 화면과 유사하게 만든 정적인 형태의 모형

- 스토리보드 : 와이어프레임에 대한 설명, 페이지 간 이동 흐름등을 추가한 문서

- 프로토타입 : 와이어프레임이나 스토리보드 등에 인터랙션을 적용함으로써 실제 구현된 것처럼 테스트가 가능한 동적인 형태의 모형

- 유스케이스 : 사용자의 요구사항을, 사용자가 원하는 목표를 달성하기 위해 수행할 내용을 기술

4. 웹의 3요소

- 웹 표준 : 웹에서 사용되는 규칙 또는 기술을 의미

-웹 접근성 : 누구나 어떠한 환경에서도 웹 사이트에서 제공하는 모든 정보를 접근

-웹 호환성 : 하드웨어나 소프트웨어 등이 다른 환경에서도 모든 이용자에게 동등한 서비스 제공

5. 네비게이션

- 사용자가 원하는 정보를 빠르게 찾을 수 잇도록 안내하는 것

6. 정황시나리오

- 정황시나리오는 사용자의 요구사항을 도출하기 위해 작성하는 것으로, 사용자가 목표를 달성하기 위해 수행하는 방법을 순차적으로 묘사한 것이다. 이야기 형식으로 표현

7. UI 프로토타입

- 사용자 요구사항을 기반으로 실제 동작하는것처럼 만든 동적인 형태의 모형, 테스트가 가능

- 장점 : 사용자를 설득하고 이해시키기 쉽다, 사전에 오류 발견이 가능하다.

- 단점 : 필요 이상으로 자원을 소모 할 수 잇다. 중요한 작업이 생략될 수 잇다.

- 페이퍼 프로토타입 : 스케치등을 하여 손을 직접 작성하는 방법, 비용이 저렴하며 즉시 생성이 가능하며 변경도 가능하다. 테스트하기에 부적당하며 공유가 어렵다

- 디지털 프로토타입 : ppt와 같은 프로그램을 사용하여 작성하는 방법이다. 최종 제품과 비슷하게 테스가 가능하며 수정, 재사용이 쉽지만 사용법을 알아야한다.

8. 아이콘

텍스트 박스 : 입력이 가능함

콤보 박스 : 목록에서 항목을 선택

라디오 박스 : 여러 개 중 하나만을 선택

체크 박스 : 여러 개의 값 중 하나 이상 선택

9. 사이트맵

-- 화면의 정보를 한눈에 파악하기 위한 시각적인 컨텐츠 모형

- 일반적으로 테이블 형태도 되어있고, 위에서부터 아래로 내려가며 정보를 찾을 수 잇다.

- 사이트에 표시할 콘텐츠를 한 눈에 알아 볼 수 있도록 메뉴별로 구분한 모형

**6장 앱 테스트 관리**

1. 앱 테스트의 개념

- 애플리케이션에 잠재되어 있는 결함을 찾아내는 일련의 행위

- 소프트웨어가 고객의 요구사항을 만족시키는지 확인하고

- 소프트웨어가 기능을 정확히 수행하는지 검증 한다

2. 앱 테스트 기본원리

- 완벽한 테스트 불가능 : 잠재적인 결함을 줄일수는 있지만 결함이 없다고 증명 할 수 없다. 완벽한 테스팅은 불가능하다.

- 결합 집중 : 애플리케이션의 20%에 해당하는 코드에서 전체 결함의 80%가 발견된다고 하여 파레토 법칙을 적용한다.

- 살충제 패러독스 : 동일한 테스트 케이스로 동일한 테스트를 반복하면 더 이상 결함이 발견되지 않는다. 테스트 케이스를 지속적으로 보완 및 개선해야 한다.

- 테스팅은 정황의존 : 테스트는 정황에 따라 테스트 결과가 달라질 수 있다. 정황에 따라 테스트를 다르게 수행

- 오류-부재의 궤변 : 결함을 모두 제거해도 사용자의 요구사항을 만족시키지 못하면 해당 소프트웨어의 품질은 높지않다.

- 테스트와 위험은 반비례, 점진적 확대, 별도 팀 수행

3. 정적 테스트

- 프로그램을 실행하지 않고 명세서나 소스 코드를 대상으로 분석하는 테스트

- 워크스루 : 개발자가 모집한 전문가들이 검토, 오류의 조기검출 목적

- 인스펙션 : 워크스루 발전형태, 개발 단계에 산출된 결과물의 품질을 평가하며 이를 개선

- 코드검사

4. 동적 테스트

- 프로그램을 실행하여 오류를 찾는 테스트, 개발의 모든 단계에서 진행

- 블랙박스, 화이트박스

5. 명세기반 테스트

- 명세를 빠짐없이 테스트 케이스로 구성하여 확인하는 테스트

- 동치분할, 경계 값 분석

6. 구조기반 테스트 : 소프트웨어 내부의 논리 흐름에 따라 테스트 케이스를 작성

7. 검증테스트

- 개발자의 시각에서 제품이 명세서대로 완성됐는지를 테스트

8. 확인 테스트

- 사용자의 시각에서 사용자가 요구한대로 제품이 완성됐는지

9. 목적에 따른 테스트

- 회복 테스트 : 여러 가지 결함을 주어 실패하도록 한 후 올바르게 복구되는지

- 안전 테스트 : 침입으로부터 시스템을 보호할 수 잇는지를 확인하는 테스트

- 강도 테스트 : 과도한 정보량을 부하하여 과부하 시에도 소프트웨어가 정상적으로 실행되는지를 확인하는 테스트

- 구조 테스트 : 소스코드의 복잡도를 평가

- 회귀 테스트 : 변경 또는 수정된 코드에 새로운 결합이 없는가 확인, 테스트를 반복한다.

- 병행 테스트 : 기존 소프트웨어에 동일한 데이터를 입력하여 결과를 비교하는 테스트

10. 화이트 박스 테스트

- 모듈의 원시 코드를 오픈 시킨 상태에서 원시 코드의 논리적인 모든 경로를 테스트

- 모듈안의 작동을 직접 관찰한다. 모든 문장을 한 번 이상 실행함

- 기초 경로 검사, 조건검사, 루프검사, 데이터 흐름 검사 등이 있다.

11. 화이트 박스 테스트 검증 기준

- 구문 검증 기준 : 소스 코든 모둔 구문이 한 번 이상 수행

- 결정 기반 : 모든 조건문이 한 번 이상 수행되도록 테스트 케이스 설계

- 조건 검증 기준 : 모든 조건무에 대해 조건이 T/F 경우 한번 이상

12. 블랙박스 테스트

- 수행할 특정 기능을 알기 위해서 각 기능이 완전히 작동되는 것을 입증하는 테스트,

- 기능 테스트, 구현된 기능을 테스트한다.

- 소프트웨어 인터페이스에서 테스트한다.

- 동치분할검사, 경계값 분석, 오류예측검사, 비교검사

13. 개발 단계에 따른 앱 테스트 순서

단위 테스트 – 구조기반 테스트, 명세기반 테스트

통합 테스트 – 하향식 상향식

시스템 테스트

인수 테스트 – 알파테스트, 베타테스트

검증테스트 – 단위테스트, 통합테스트, 시스템테스트

확인테스트 - 인수 테스트

14.단위 테스트

- 설계의 최소 단위인 모듈이나 컴포넌트에 초점을 맞춰 테스트하는 것

15. 통합테스트

- 단위 테스트가 끝난 모듈을 통합하는 과정에서 발생하는 오류 및 결함을 찾는 테스트기법

16. 하향식 통합 테스트

- 상위모듈에서 하위모듈 방향으로 통합한다.

- 깊이 우선 통합법 (수직위주), 넓이 우선 통합법 (수평위주)

- 상위 모듈에서는 테스트 케이스를 사용하기 어렵다.

- 시스템 구조를 보여줄 수 있다.

- 스텁을 사용한다, 스텁은 일시적으로 필요한 조건만을 가지는 시험용 모듈이다.

17. 상향식 통합 테스트

- 하위 모듈에서 상위 모듈 방향으로 통합하면서 테스트

- 제어 모듈과 종속 모듈의 그룹 클러스터

- 클러스터 단위로 테스트한다.

- 테스트가 완료되면 클러스터는 프로그램 구조의 상위로 이동하여 결합

- 드라이버는 실제 모듈로 대체된다.

- 드라이버는 하위 모듈을 호출하고 파라미터를 전달하는 도구

18. 테스트 산출물

- 테스트 계획서 : 테스트 수행을 계획한 문서

- 테스트 케이스 : 테스트를 위한 입력 값, 조건, 실행 결과 등으로 만들어진 명세서

- 테스트 시나리오 : 테스트를 수행할 여러 개의 테스트 케이스의 동작 순서를 기술한 문서

- 테스트 결과서 : 테스트 결과를 비교 분석한 내용을 정리한 문서

19 결함 추적 및 관리

에러 발견 – 등록- 분석 – 결함확정 – 할당 – 조치 – 검토 및 승인

ㅂㄷㅂㅎㅎㅈ

20. 테스트 케이스

-구현된 소프트웨어가 사용자의 요구사항을 정확하게 준수했는지를 확인하기 위해 값, 조건, 결과 값 등으로 구성된 테스트 항목에 대한 명세서

21. 테스트 시나리오

- 테스트 케이스를 적용하는 순서에 따라 여러 개의 테스트 케이스들을 묶은 집합,. 테스트 케이스들을 적용하는 구체적인 절차

22. 테스트 오라클

- 테스트 결과가 올바른지 판단하기 위해 사전에 정의된 참 값을 대입하여 비교하는 기법

- 참 오라클 : 테스트 케이스의 입력 값에 대해 기대하는 결과를 제공하는 오라클

- 샘플링 오라클 :입력 값들에 대해서만 기대하는 결과를 제공하는 오라클

- 추정 오라클 : 특정 테스트 케이스의 입력 값에 대해 기대하는 결과제공, 나머지 추정처리

- 일관성 검사 오라클 : 변경이 있을 떄 테스트 케이스의 수행 전과 후의 결과 값이 동일한지 확인

23. 테스트 하네스 도구

- 앱 컴포넌트 및 모듈을 테스트하는 환경의 일부분, 테스트를 지원하기 위해 생성된 코드와 데이터를 의마한다.

- 테스트 드라이버,

- 테스트 스텁,

- 테스트 슈트(시스템에 사용되는 테스트 케이스 집합),

- 테스트 케이스, 테스트 스크립트(테스트 실행절차), -

- 목 오브젝트(상황에 맞는 예정된 행위를 수행하는 주체)

24. 결함관리

- 프로세스 : 결함 관리 계획 – 결합 기록 – 결함 검토 – 결함 수정 –결함 재확인 – 결함 상태 추적 및 모니터링 - 최종확인

25 결합관리 측정 지표

* 결함 분포 : 특정 속성에 해당하는 결함 수 측정
* 결함 추세 : 테스트 기간에 따른 결함 수의 추이 분석
* -결함 에이징 : 결함이 지속되는 시간 측정

26. 앱 성능

- 경과시간 : 작업을 의뢰한 시간부터 처리가 완료될 때까지 걸리는 시간

- 응답 시간 : 요청을 전달한 시간부터 응답이 도찰할 때까지 걸린 시간

- 자원 사용률 : 작업을 처리하는 동안의 CPU 사용량, 메모리량 등 자원 사용률

- 처리량 : 일정 시간 내에 앱이 처리하는 일의 양

27.클린 코드

- 누구나 쉽게 이해하고 수정 및 추가 할 수 있는 단순, 명료한 코드

**8장 소프트웨어 개발 보안**

1. 개발보안 관련 기관

행정안정부 – 개발 보아누 정책을 총괄, 관련 법규 정비

한국인터넷 진흥원 – 보안 정책 및 가이드를 개발, 기술 지원, 자격제도 운영

발주기관 – 개발 보안의 계획을 수립, 감리법인 선정

사업자 – 계획 등을 명시한다, 개발

감리법인 – 감리계획을 수립한다.

2. Secure SDLC

- 보안상 안전한 소프트웨어를 개발하기 위해 SDLC에 보안 강화를 위한 프로세스를 포함하는 것

3. 각 단계의 보안 활동

분석 단계 – 보안 항목에 해당하는 요구사항을 식별하는 작업 수행

설계 단계 – 식별된 보안 요구사항들을 소프트웨어 설계서에 반영, 보안 설계서 작성

구현 단계 – 보안 가이드를 준수하며 보안 요구사항을 구현한다. 시큐어 코딩

테스트 단계 – 보안 설계서를 바탕으로 보안 사항들이 정확히 반영하고 동작되는지 점검

4. 시큐어 코딩

소프트웨어의 구현 단계에서 발생 할 수 잇는 보안 취약점들을 최소화 하기위해 보안요소를 교려하며 코딩하는 것

5. 세션통제

- 불충분한 세션관리 : 특정규칙이 있는 세션아이디를 발급하거나 타이아웃이 너무 길 경우

- 잘못된 세션에 의한 정보노출 : 멀티스레드 환경에서 멤버 변수에 정보를 저장할 떄 발생가능

6. 입력 데이터 검증 및 표현

- SQL삽입공격 : 입력란에 SQL을 삽입하여 무단으로 DB를 조회하거나 조작하는 보안약점

입력 데이터에 예약어 및 특수문자 필터링하여 방지

- XSS : 웹 페이지에 악의적인 스크립트를 삽입하여 방문자들의 정보를 탈취하거나 비정상적 기능을 유발, HTML 태그 제한, 스크립트 삽입방지, 문자 치환으로 방지

- 운영체제 명령어 삽입 : 외부 입력값을 통해 시스템 명령어의 실행을 유도함으로써 권한을 탈취하거나 시스템 장애 유발, 웹 인터페이스를 통해 명령어 전달 차단, 외부 입력값 검증

- 신뢰되지 않은 URL연결 : 입력 값으로 사이트 주소를 받는 경우 이를 조작하여 피싱 사이트로 유도, 외부 사이트의 주소를 화이트리스트로 관리하여 방지

7. 보안 기능

- 적절한 인증 없이 중요기능 허용

- 부적절한 인가

- 취약한 암호화 알고리즘 사용

- 하드코드된 비밀번호

- 중요정보 평문 저장 및 전송

8. 시간 및 상태

- TOCTOU 경쟁조건 : 검사시점과 사용시점을 고려하지 않고 코딩하는 경우 발생

- 종료되지 않는 반복문, 재귀함수

9. 에러처리

- 오류 메시지를 통한 정보노출

- 오류 상황 대응 부재

10. 코드 오류

- 널 포인터 역참조 : 널 포인터가 가리키는 메모리에 어떠한 값을 저장할 때 발생, 널이 될 수 있는 포인터를 이용하기 전에 널 값을 가지고 있는 검사함으로써 방지

11. 캡슐화

- 잘못된 세션에 의한 정보 노출 : 다중스레드 환경에서 멤버 변수에 정보 저장시 발생, 지역변수를 활 용하여 변수의 범위를 제한함으로써 방지

- 제거되지 않고 남은 디버그 코드 : 디버그 코드로 인해 노출가능, 코드 검사를 통해 남아잇는 디버그 코드 삭제

12. 개인키 암호화 기법

- 양방향 방식

- 동일한 키로 데이터를 암호화, 복호화

- 대칭 암호기법, 단일키 암호화 기법,

- 암호화 복호화 속도가 빠르며, 알고리즘이 단순하다, 파일의 크기가 작다

- 관리해야 할 키의 수가 많아진다

- DES : 75년 미국에서 개발, 64비트 블록 56비트 키 길이

- AES : 01년 미국에서 개발, 128비트 블록, 128,192,256비트 키 길이

- SEED : 99년 국내에서 개발, 128비트 블록, 128,256비트 키 길이

- ARIA : 04년 국내에서 개발, 128비트 블록, 128,192,256비트 키 길이

- A로 시작하는 암호화 알고리즘은 128비트의 블록, 128.192,256비트의 키 길이를 가진다.

13. 공개키 암호화 기법

- 양방향 방식

- 암호화는 공개키를 복호화는 비밀키로 한다.

- 키의 분배가 용이하고, 관리해야할 키의 개수가 적다

- 속도가 느리며 알고리즘이 복자하다,. 개인키 암호화보다 크기가 크다

- RSA : 소수를 이용

14. 해시

- 단방향 알고리즘, 고정된 길이의 값이나 키로 변환

- SHA, MD5

**10장 SW기초**

1. 운영체제

- 컴퓨터 시스템의 자원들을 효율적으로 관리하며 , 사용자가 컴퓨터를 편히하고 효과적으로 사용할 수 있도록 환경을 제공하는 프로그램의 모임

2. 운영체제의 목적 (처반사신)

- 처리능력 향상 : 시스템이 처리하는 일의 양

- 반환시간 단축 : 작업의뢰 시간 ~ 처리완료까지의 시간

- 사용가능도 향상 : 필요 시 사용가능한 정도

- 신뢰도 향상 : 주어진 문제를 정확하게 해결하는 정도

3. 운영체제의 기능

- 자원을 효율적으로 관리하기 위한 자원 스케줄링

- 사용자와 시스템 간의 편리한 인터페이스

- 입출력에 대한 보조 기능

- 자원 보호 기능-

- 시스템 오류 검사, 복구

4. 윈도우즈

- GUI

- 선점형 멀티태스킹

- PnP : 자동감지, 필요한 시스템 환경을 OS가 자동으로 구성

- OLE : 그림 등의 개체를 자유롭게 연결하거나 삽입가능

- 싱글유저 시스템

- 트리구조

5. 유닉스

- 트리구조

- 멀티유저 멀티태스킹

- 대부분 C로 작성

6. 커널

- 유닉스의 핵심적인 부분

- 프로그램과 하드웨어 간의 인터페이스 역할

- 프로세스 관리, 기억장치관리, 입출력 관리

7. 쉘

- 명령어 해석기

- 시스템과 사용자 간의 인터페이스

8. 리눅스

- 유닉스 기반 오픈소스

9. MacOS,

- 애플 사가 유닉스 기반으로 개발한 OS

10. 기억장치 관리

반입전략, 배치전략, 교체 전략

11. 반입 전략 (Fetch)

- 프로그램이나 데이터를 언제 주기억장치로 적재할 것인지를 결정

- 요구반입, 예상 반입

12. 배치 전략 (Placement)

-새로 반입되는 프로그램이나 데이터를 주기억장치의 어디에 위치 시킬 것인지를 결정

- 최초 적합(들어갈 수 있는 맨 처음), 최적 적합(단편화 가장 적은 곳), 최악 적합(단편화 심한 곳)

13. 교체 전략 (Replacement)

- 이미 사용되고 있는 영역 중에서 어느 영역을 교체하여 사용할 것인지를 결정

- FIFI, OPT, LRU, LFU, NUR, HRN((대기시간+서비스시간)/서비스 시간))

14. 리눅스 명령어

- cat : 내용 확인

- chmod : 허가권 변경 chmod u=rwx abc.txt

- chown : 소유자와 그룹 변경 chown member abc.txt

- find : 파일을 찾는다.

- ps : 현재 실행중인 프로세스를 표시한다,

- top : 시스템의 프로세스와 메모리 사용 현황을 표시한다.

- meminfo : 메모리 상태 확인

15. 프로세스

- pcb를 가지는 프로그램, 프로세서에 의해 처리되는 사용자 프로그램, 실행중인 프로그램

- 비동기적 행위를 일으키는 주체

- 프로시저가 활동중인 것

16. 프로세스 상태전이

- 준비 : 프로세스가 프로세서를 할당받기 위[해 기다리고 있는 상태

- 실행 : 프로세스가 프로세서를 할당받아 실행되는 상태

- 대기 : 입출력 처리 필요하면 현재 실행중인 프로세스가 중단되고 입출력 처리가 완료될 떄까지의 상태

- 디스패치 : 준비 상태에서 대기하고 있는 프로세스 중 하나가 프로세서를 할당받아 실행 상태로 전이 (준비 -> 실행)

- 웨이크업 : 입출력 작업이 완료되어 프로세스가 대기 상태에서 준비 상태로 전이되는 과정

(대가 -> 준비)

17. 데이터베이스

- 통합된 데이터 : 중복이 최소화된 데이터의 모임

- 저장된 데이터 : 컴퓨터가 접근 할 수 잇는 저장 매체에 저장된 데이터

- 운영 데이터 : 존재가치가 확실하고 없어서는 안 될 반드시 필요한 데이터

- 공용 데이터 : 여러 응용 시스템들이 공동으로 소유하고 유지하는 데이터

18. DBMS 기능

- 정의 기능 : 데이터의 형과 구조에 대한 정의, 제약 조건등을 명시

- 조작 기능 : 데이터의 검생 갱신 삽입 삭제 등을 체계적으로 처리하기 위해 사용자와 데이터베이스 사이의 인터페이스 수단을 제공

- 제어 기능 : 작업이 정확하게 수행되어 데이터의 무결성이 유지되도록 제어하는 기능

19. ER도형

- 개념적 데이터 모델으로 개체 속성 관계 등에 대하여 용이하게 표현할 수 있는 데이터 모델이다.

사각형 – 개체타입

마름모 - 관계타입

타원 – 속성

타원에 밑줄 – 기본키 속성

직선 – 개체와 속성을 연결

쌍타원 – 다중 값 속성

20. 관계형 데이터 구조 , 릴레이션

- 릴레이션의 구조 ,속성들의 집합 : 릴레이션 스키마

- 튜플들의 집합 : 렐리이션 인스턴스

- er모델을 릴레이션 스키마로 변환하는 것을 매핑룰이라고 한다.

- 속성은 컬럼으로, 식별자는 PK로 관계는 FK로 표현

21. 키

- 데이터베이스에서 조건에 만족하는 튜플을 찾거나 순서대로 정렬한 기준이 되는 속성

22. 슈퍼키

- 유일성은 만족하지만 최소성은 만족하지 않는 키

23. 후보키

- 릴레이션을 구성하는 속성들 중에서 튜플을 유일하게 식별하기 위해 사용되는 속성들의 부분집합, 유일성과 최소성을 모두 만족한다.

24. 기본키

- 후보키 중에서 선정된 키로 중복된 값을 가질 수 없다.

25. 대체키

- 후보키 중 선정된 기본키를 제외한 나머지 후보키

26. 외래키

- 다른 릴레이션의 기본키를 참조하는 속성 또는 속성들의 집합을 의미한다.

27. 무결성

- 데이터베이스에 저장된 데이터 값과 그것이 표현하는 현실 세계의 실제값이 일치하는 정확성

28. 개체 무결성

- 기본키를 구성하는 어떤 속성도 NULL 값이나 중복값을 가질 수 없다

29. 도메인 무결성

- 주어진 속성 값이 정의된 도메인에 속한 값이어야 한다

30. 참조 무결성

- 외래키 값은 NULL이거나 참조 릴레이션의 기본키 값과 동일해야한다.

31. 기타무결성

사용자 정의 무결성 -속성 값들이 사용자가 정의한 제약조건에 만족해야한다.

NULL 무결성 – 특정 속성 값이 널이 될수 없다.

고유 무결성 – 특정 속성에 대해 각 튜플이 갖는 속성값들이 서로 달라야한다.

키 무결성 – 하나의 릴레이션에는 적어도 하나의 키가 존재

32. LAN

근거리 통신망 , 속도가 빠르며 에러율이 낮다, 거리가 짧다, 주로 학교 회사 등, 버스형이나 링형을 사용한다.

33.WAN

- 광대역 통신망, 멀리 떨어진 사이트들을 연결한다. 속도가 느리며 에러율이 높다. 주로 망형을 사용한다.

34. IP

- IP주소는 인터넷에 연결된 모든 컴퓨터 자워을 구분하기 위한 고유한 주소이다.

- IPv4는 8비트 씩 4부분 32비트로 구성되며 A클래스부터 E클래스 까지 5부분 존재

- A클래스 (0~127), B클래스 (128~191), C클래스(192~223)

- D클래스 (224~239) 멀티캐스트 용도

-E클래스 (240~) 실험용

35. IPv6

- v4에 비해 속도가 빠르며 호환성이 좋다. 보안문제를 해결할 수 잇다.

- 16비트 씩 8부분 총 128비트, 16진수로 표현, 콜론으로 구분한다.

- 유니캐스트 : 단일송신자와 단일 수신자의 통신

- 멀티캐스트 : 단일 송신자와 다중 수신자간의 통신

- 애니캐스트 : 단일 송신자와 가까이 있는 단일 수신자간의 통신

36. DNS

- 숫자로 된 IP주소를 사람이 이해하기 쉬운 문자 형태로 표현한 것

37. OSI 참조모델

- 하위계층 (1~3), 상위계층(4~7)

- 서로다른 시스템 간을 상호 접속하기 위한 개념

38 물리 계층

* 비트 단위 전송
* 계능기차
* 리피터, 허브 사용,

39. 데이터링크 계층

- 프레임 단위 전송

- 두 개의 인접한 개방 시스템간의 효율적 정보 전송

- 브릿지, 스위치

- HDLC, LLC, MAC, PPP

40. 네트워크 계층

- 패킷 단위

- 결오설정, 트래픽 제어 등을 수행

- 라우터

- IP, ICMP, IGMP, ARP, RARP

41. 전송 계층

- 세그먼트 단위

- 종단 간 시스템 간의 데이터 전송

- 상위 계층과 하위계층의 인터페이스

- 흐름제어, 오류제어, 다중화

- TCP/UDP

- 게이트웨이 사용

42. 세션 계층

- 메시지 단위

- 인증, 동기화, 데이터 교환 관리, 대화 제어

- 동기점 뭐시기 나오면 세션계층

43. 표현 계층

- 압축, 암호화, 구문 검색, 코드 변환 등

44. 응용계층

- 사용자가 OSI환경에 접근할 수 잇도록 서비스를 제공한다.

45. 허브

가까운 거리의 컴퓨터들을 연결하는 장치로 각 회선을 통합적으로 관리, 리피터 역할도 수행한다.

더미 허브 : 데이터를 단순히 연결하는 기능만을 수행한다.

스위치 허브: 네트워크 상에 흐르는 데이터의 유무 및 흐름을 제어,대역폭을 사용하는 지능형허브

물리계층에서 사용

46. 리피터

- 물리계층에서 사용

- 원래의 형태와 다르게 왜곡되거나 약해질 경우 원래의 신호형태로 재생하여 전송하는 역할

47. 브릿지

- LAN과 LAN을 연결하거나 LAN안에서 컴퓨터 그룹을 연결하는 기능을 수행

- 트래픽 병목현상을 줄인다. 보안성을 높인다. 회선 수 n(n-1)/2

48. 스위치

- LAN과 LAN을 연결하거나 더 큰 LAN을 만든다.

- 전송속도가 빠르다.

49. 라우터

- 데이터 전송의 최적 경로를 선택할 수 있다. 3계층 장비

50. 게이트 웨이

- 전 계층의 프로토콜 구조가 다른 네트워크의 연결을 수행한다.

51. TCP/IP

- 인터넷에 연결된 서로 다른 기종의 컴퓨터들이 데이터를 주고 받을 수 잇도록 하는 표준 프로토콜

- TCP는 신뢰성있는 연결형 서비스, 스트림 전송기능, 다중화, 흐름제어, 오류제어 등 수행

- IP는 데이터그램을 가반 비연결형 서비스,

52 프로토콜 3요소

* 구문 : 데이터의 형식, 부호화 등을 규정
* 의미 : 정보전송을 위한 협조 사항과 오류관리를 위한 제어 정보
* 시간 : 통신속도, 메시지의 순서제어

53. TCP/IP 계층

- 물리 + 데이터링 -> 네트워크 엑세스 계층 : 이더넷,IEEE, X-25

- 네트워크 계층 -> 인터넷 계층 : 데이터 전송을 위한 주소지정, 경로설정

- 전송계층 -> 전송계층 : 호스트들 간의 신뢰성잇는 통신

-> 응용 + 표현 + 세션 -> 응용계층 : 응용 프로그램 간의 송수신

54. 응용계층 프로토콜

FTP : 원격 파일전송 시스템, 21번

SMTP : 전자 우편을 교환하는 서비스 25번

TELNET : 원격접속, 가상터미널 기능 23번

SNMP : 네트워크 관리 프로토콜,

DNS : 도메인 네임을 IP주소로 매핑

HTTP : www에서 HTML 문서를 송수신 하기 위한 표준 프로토콜

TCP기반 :나머지

UDP기반 : DNS, TFTP, SNMP

55. TCP

- 전송계층 프로토콜

- 스트림 위주의 전송

- 신뢰성있는 연결

- 투명성이 보장된다

56. UDP

- 비연결형 서비스

- 비신뢰성 연결

- 오버헤드가 적으며 속도가 빠르다

- 실시간 전송에 유리하다.

57. RTCP

- RTP 패킷의 전송 품질을 제어하기 위한 프로토콜

- 다중화 머시기

- RTP는 UDP기반의 유니캐스트 등의 통신서비스를 통한 실시간 머시기

58. IP

- 전송할 데이터에 주소를 지정, 결로를 설정

- 비연결형 데이터그램 방식, 신뢰성 보장하지 않는다

59. ICMP

- 제어 메시지 관리 역할

60. IGMP

- 멀티캐스트를 지원하는 호스트나 라우터 사이에서 멀티캐스트 룹을 유지

61. APR, RARP

IP -> MAC = ARP

MAC -> IP = RARP

62. 라우팅 알고리즘

- 송수신 측 간의 전송 경로 중에서 최적 패킷 교환 경로를 결정

- RIP : 소규모 동족의 네트워크에서 효율적인 방법, 홉을 사용한다. 거리벡터 알고리즘 사용

- OSPF : 대규모 네트워크에서 많이 사용, 링크 상태 알고리즘 사용

- IGRP : 네트워크 상태를 고려하여 라우트 거리벡터 알고리즘 사용

- BGP : 자율 시스템 간의 라우팅 프로토콜

**11장 패키징**

1. 소프트웨어 패키징

- 모듈별로 생성한 실행 파일들을 묶어 배포용 설치파일을 만드는 과정

- 사용자 중심

2. 릴리즈 노트

- 개발 과정에서 정리된 릴리즈 정보를 고객과 공유하기 위한 문서

- 헤더에는 버전, 이름, 날짜 등의 정보가 포함된다.

3. DRM

- 디지털 저작권

- 아날로그이면 디지털로 변환 후 패키저에 의해 DRM패키징 수행

- 클리어링 하우스 : 저작권에 대한 사용 권한, 라이선스 발급, 등 결제 관리를 수행

- 패키저 : 컨텐츠를 메타데이터와 함께 배포 가능한 형태로 묶어 암호화

- DRM 컨트롤러 : 배포된 컨텐츠의 이용권한을 통제

- 보안 컨테이너 : 컨텐츠 원본을 안전하게 유통하기 위한 전자적 보안장치

4. DRM 기술요소

- 암호화 : 라이선스 암호화, 전자서명

- 키관리 : 암호화한 키에 대한 저장 및 분배기술

- 크랙방지 : 크랙에 의한 컨텐츠 사용 방지 기술

- 인증 : 라이선스 발급 및 사용의 기준이 되는 사용자 인증

5. 형상관리

- 소프트웨어의 개발 과정에서 소프트웨어의 변경 사항을 관리하기 위해 개발된 일련의 활동

- 소프트웨어의 변경의 원인을 알아내고 제어하며, 적절히 변경되고 잇는 확인

- 개발의 전 단계에 적용,

6. 버전관리

- 프로그램의 모든 프로그래머 및 개발자들이 프로그램을 개정한 사항들을 추적하는 데 사용되는 소프트웨어

7. 공유 폴더 방식

- 로컬 컴퓨터의 공유 폴더에 저장되어 관리되는 방식

- 구닥다리 방식

- RCS

8. 클라이언트/서버 방식

- 버전 관리 자료가 중앙 시스템(서버)에 저장되어 관리되는 방식

- 모든 버전 관리는 서버에서 수행한다.

9. SVN

- 클라이언트 서버 방식으로 아파치에서 만들었다.

- 오픈소스

- 서버의 자료를 클라인트로 복사해와 작업한 후 변경 내용을 서버에 반영

10. 분산 저장소 방식

- 버전 관리 자료가 하나의 원격 저장소와 분산된 개발자 PC의 로컬 저장소에 함께 저장되어 관리되는 방식

- GIT

11. 빌드 자동화 도구

- 소스 코드 파일들을 컴파일한 후 여러 개의 모듈을 묶어 실행파일로 만드는 과정

- Jenkins : 자바 기반의 오픈소스, 웹 UI, 대부분의 형상관리 도구와 연동 가능

- Grandle : 그루비기반 안드로이드 개발 환경

**8장 SQL**

1. DDL

- 데이터 정의어, 데이터 형식, 접근 방식 등 DB를 구축하거나 수정할 목적으로 사용하는 언어

- Alter, Create, Drop

2. Create table

- 테이블을 정의하는 명령어

- 외래키 설정은 foreign key() references 참조테이블(참조 속성)

- check는 제약조건을 표시한다.

- 외래키 옵션 중 on delete, on update는 참조 테이블의 속성이 변경 시에 일어나며 주로 cascade옵션을 사용한다.

3. Create view

- Create view 뷰명 AS select ~

- Create view 고객(성명, 전번) as select 성명, 전번 from 고객 where ~

4. Create index

- 특정 속성을 기준으로 한다.

- Create (unique) index 고객\_idx ON 고객(고객번호 desc)

- 유니크 속성은 중복 값이 없는 속성으로 인덱스를 생성한다.

5. Aler table

- 속성 추가 add : alter table 테이블 ADD 속성명 타입

Alter table 학생 ADD 학년 varchar(3);

- 특정속성 디폴값 변경 Alter : Alter Table 테이블 Alter 속성명

Alter table 학생 ALTER 학생 varchar(10) not null;

- 특정속성 삭제 : Drop Column : Alter Tabe 테이블 DROP COLUMN 속성명

Alter table 학생 Drop Column 학생

6. DROP

- 도메인 테이블등을 삭제할 때 사용함 cascade옵션을 자주 사용

7. DCL

- 데이터 제어어로 데이터의 보안, 무결성, 회복, 병행제어 등을 정의하는데 사용하는 언어

8. Grant

-권한을 부여할 때 사용하는 명령어

- Grant 권한 ON 개체 TO 사용자 [WITH GRANT OPTION],

- 부여받은 권한을 다른 사용자에게 다시 부여하는 옵션 WITH GRANT OPTION

- Grant ALL ON 고객 TO 나비 WITH GRANT OPTION

9. Revoke

- 권한을 취소하는 명령어

- REVOKE [GRANT OPTION FOR] 권한 리스트 ON 개체 FROM 사용자 [CASCADE]

- GRANT OPTION FOR 다른 사용자에게 권한을 부여할 수 잇는 권한을 취소함

- CASCADE는 권한 취소 시 권한을 부여받은 사용자가 다른 사용자에게 부여한 권한도 연쇄취소

10. COMMIT

- 트랜잭션이 성공적으로 끝나면 일관성있는 상태를 가지기 위해 변경된 모든 내용을 디비에 반영하는 명령어

- COMMIT;

11. ROLLBACK

- 커밋되자 않은 변경된 모든 내용들을 취소하고 이전 상태로 되돌리는 명령어

- ROLLBACK; , ROLLBACK TO S1;

12. SAVEPOINT

- ROLLBACK 위치 지정하는 명령어

- SAVEPOINT S2;

13. DML

- 데이터 조작어, 데이터베이스 사용자가 데이터를 실질적으로 관리하는데 사용되는 언어

- SELECT, INSERT, DELETE, UPDATE

14. SELECT

- 튜플을 검색하는데 사용하는 명령어

15. INSERT

- 기본 테이블에 새로운 튜플을 삽입할 때 사용한다.

- INSERT INTO 테이블명 VALUES

- INSERT INTO 사원(이름, 부서) VALUES(‘김’,’컴’);

- INSERT INTO 사원 VALUES (~);

- INSERT INTO 편집부원(이름, 생일, 주소, 기본급) SELECT 이름, 생일,주소,기본급 FROM 사원 WHERE 부서 = ‘편집’;

16. DELETE

- 기본 테이블에 잇는 튜플들 중에서 특정 튜플을 삭제할 때 사용한다.

- DELETE FROM 테이블명 [WHERE]

- DELTE FROM 사원 WHERE 이름=’김’;

- DELETE FROM 사원; 사원 테이블의 모든 레코드를 삭제

17. UPDATE

- UPDATE 테이블 SET 속성명 = 데이터 WHRER ~

- UPDATE 사원 SET 주소 = ‘수색동’ WHRER 이름 = ‘홍길동’ 홍길동주소를 수색동으로 수정

- UPDATE 사원 SET 부서 =’기획’, 기본급 = 기본급+500 WHERE 이름 = ‘황진이’;

18. 프로시저

- 특정 기능을 수행하는 절차형SQL, 호출을 통해 실행되어 미리 저장해 놓은 SQL작업을 수행

- DECLARE, BEGIN, CONTROL, SQL, EXCEPTION, TRANSACTION, END 순서

- CREATE OR REPLACE 옵션은 선택적으로 동일한 프로시저 이름이 이미 존재하면 덮어쓰기한다.

- 파라미터 IN – 프로시저에게 값을 전달

- OUT – 호출 프로그램에게 값을 반환할 떄

- INOUT – 둘 다

- 실행하는 명령어는 EXECUTE, EXEC, CALL 등이 있다.

- EXEC EMP\_CHANGE(32);

- 삭제는 DROP PROCEDURE ~

19. 트리거

- 데이터베이스 시스템에서 삽입 갱신 제어 등의 이벤가 발생할 때마다 관련 작업이 자동으로 수행되는 절자형SQL

- CREATE [OR REPLACE] TRIGGER 이름 [동작시기][동작옵션] ON 테이블명 REFERENCING

[NEW |OLD] AS 테이블

FOR EACH ROW

- 동작시기에는 변견되기 전에 수행하는 BEFORE, 변경된 후에 실행하는 AFTER

- 동작옵션에는 INSERT, DELETE, UPDATE가 있다.

- NEW 옵션은 추가되거나 수정에 참여할 튜플들의 집합을 의미한다.

- OLD 옵션은 수정되거나 삭제 전 대상이 되는 튜플들의 집합

- FOR EACH ROW : 각 튜플마다 트리거를 적용한다는 의미.

- 삭제는 DROP TRIGGER 학년정도\_TRI;

20. 사용자 정의함수

- 프로시저와 유사하게 SQL을 사용하여 일련의 작업을 연속으로 처리하고 종료 시 처리결과로 단일값을 반환하는 절차형SQL

- RETURN이 필수로 들어간다.