**클라우드 #10주차 과제**

21660019 3A김세진

**1. SDLC**

**2. Devops, DevSecOps**

**3. 절차표**

**4. 소프트웨어 공학(개발방법론)**

**5. MECE**

**6. SW테스트**

**7. pair programming, peer review**

**8. 알고리즘**

**9. 튜닝**

**10. db시스템**

**11. 서브네팅, 슈퍼네팅**

**12. RPA, MDD**

**1. 개발단계 SDLC**

1-(1) SDLC의 개념

SDLC는 Software Development Life Cycle, 즉, 소프트웨어 개발 수명주기로써, 시스템 엔지니어링, 정보 시스템, 또는 소프트웨어 공학에서 정보 시스템을 계획, 개발, 시험, 채용하는 과정을 뜻하는 용어이다. 요구사항 분석→설계→개발→테스트→운영 단계로 구성되어있다.

1-(2) 내가 생각하는 SDLC의 개념

새로 만드는 소프트웨어는 일종의 맞춤형 제품이라고 생각할 수 있다. 특정한 수요에 의해 요구되는 사항을 명시적으로 보여주면서 어떤 어떤 기능이 들어가야 한다. 무엇이 꼭 필요하고, 특정 상황에서는 어떻게 작동이 된다는 식의 요구를 분석하고, 그것에 맞게 설계도를 작성하여, 개발하고, 개발 후에는 버그나 보안상 문제점 등이 있는지 테스트한 후, 운영을 하는 방식일 것이다. 이해하기 쉽게 우리가 간단한 움집을 짓는다고 생각해보자, 마구잡이로 짓기 시작한다면 중간에 삐끗하는 순간 모든 것이 무너질 것이고, 시간이 더 걸릴 것이다. 하지만 처음에, 분석을 하고, 설계도를 만들어서 짓기 시작한다면, 처음에는 늦을지라도 후에는 문제가 거의 없이 완벽히 지어질 것이다. 소프트웨어도 마찬가지일 것이다. 처음에는 준비하는 시간이 걸릴테지만, 그만한 완성도를 보여줄 것이다.

1-(3) 현업에서는 어떻게 쓰일지  
 요즘 이슈가 되고 있는 배달어플을 예로 들어보겠다. 처음에 그런 배달어플을 만들어달라고, 소프트웨어를 개발하는 회사에 주문을 했다고 가정해보자, 요구사항 분석으로 우선, 최고 관리자용, 가게주인용, 손님용의 어플이 연결되야 할 것이고, 그것의 정보에 대한 데이터베이스와 연결, 주문, 결제, 위치 등의 많은 요구가 있을 것이다. 이런 요구대로 설계도(UML)를 만들어서, 모든 기능이 어떻게 작동되는지 이해하기 쉽게 만들어서 서로 협업해서 만들고, 만들어진 어플을 알파,베타 서비스 등을 이용해 테스트를 거쳐서 해당 기간동안 문제점을 수정하고, 문제가 없을 시에 완전한 서비스를 실행하는 방식으로 운영할 것 같다.

**2. Devops, DevSecOps**

2-(1) Devops, DevSecOps 의 개념  
 DevOps : 데브옵스(DevOps)는 소프트웨어의 개발(Development)과 운영(Operations)의 합성어로서, 소프트웨어 개발자와 정보기술 전문가 간의 소통, 협업 및 통합을 강조하는 개발 환경이나 문화를 말한다. 데브옵스는 소프트웨어 개발조직과 운영조직간의 상호 의존적 대응이며 조직이 소프트웨어 제품과 서비스를 빠른 시간에 개발 및 배포하는 것을 목적으로 한다.

DevSecOps : 데브섹옵스(DevSecOps)는 개발, 보안, 운영을 위한 줄임말 이라고 간단하게 정의할 수 있다. 설계부터 개발, 테스트, 생산 및 운영까지 애플리케이션 라이프 사이클 전반에 보안을 통합하는 것이다.

2-(2)내가 생각하는 Devops DevSecOps의 개념  
 DevOps : 소프트웨어를 개발하고 운영하는 일종의 방법 중 하나인 것 같다. 학교와 관련한 일로 설명을 해보자면, 학교의 종합행정시스템이라는 시스템이 있다. 이것은 학생의 학점, 데이터를 얻을 수 있는 시스템인데, 종종 이러한 시스템에 접근이 안되는 경우가 있었다. 나는 학교에 전화를 했고, 학교 홈페이지를 관리하는 쪽에서 전화를 받아 문제를 인지했다. 그리고 개발팀? 서버팀에 건의가 되었고 다음날 이 시스템에 다시 접근할 수 있게 되었다. 이 예시에서 운영쪽은 사용자의 편의를 위한 쪽으로 업무가 집중되어 개발쪽의 일은 모르며, 개발팀은 사용자의 잘못된 사용이나, 불만점을 직접적으로 받을 수 없게 되어있다. 따라서 DevOps라는 것은 개발과 운영을 따로 두는 방식이 아니라, 하나로 유기적으로 연결되게 꾸린 팀, 혹은 방법을 의미하는 것 같다.

DevSecOps : 위의 DevOps가 개발과 운영 사이의 모든 부분에서 보안을 신경쓰는 것을 의미하는 것 같다. 더 찾아보니 보안을 개발자나 운영자만의 단순한 책임을 지는 것이 아닌 누구든 이 소프트웨어와 관련된 사람이던지 보안에 책임을 가지고 개발 운영하자는 취지의 개발론인 것 같다.

2-(3) 현업에서 어떻게 쓰일지

데브옵스 또는 데브섹옵스의 가장 큰 챌린지(Challenge)는 데브옵스를 할 수 있는 역량을 지닌 IT 조직이 현실적으로 존재할 수 있는가 이다. 비용을 절감하다는 이유로 인원을 줄이고 외부 업체로 하청을 주고 있으며 나간 인력도 보충해주지 않아 허덕거리는 현실에서, 원가 절감을 이유로 필요한 툴(소프트웨어)도 사주지 않는 현실에서 데브옵스와 오픈소스소프트웨어로 IT 시스템을 구축하고 운영 하는 게 가능한 일인지 묻고 싶다.

**4. 소프트웨어 개발방법론**

4-(1) 소프트웨어 개발방법론의 개념

소프트웨어 개발 방법론은 소프트웨어를 생산하는 데에 필요한 프로그래밍 개발 과정들을 정리하고 표준화하여 프로그래머들이 프로그래밍 개발과정에서 각개인이 개발과정에서의 일관성을 유지하고 프로그래머들간의 효과적인 협업이 이루어질수 있도록 돕기 위한 방법론이다.

4-(2) 내가 생각하는 소프트웨어 개발방법론의 개념

우리가 서울로 가야 할 일이 생겼다고 할 때, 우선 들어가는 비용과 시간, 편리함 등을 잘 분석하여, 비행기, 기차, 자가용, 버스 등을 고민하다가 자기와 잘 어울리는 하나를 선택한다. 아주 간단한 선택을 함에도 여러 것을 분석하여 정하는데, 다양한 기능을 가진 요즈음의 커다란 소프트웨어를 생각을 해보면, 여러 방법론이 있을 것이고, 그런 방식에서 자신이 개발하려는 소프트웨어에 맞게 방법론을 선택하여 개발한다면, 시간이나 자원을 절약할 수 있다고 생각한다.

4-(3) 각 방법론에 대한 설명

1. 정보공학 개발 방법론 : 비즈니스 시스템 규모 성장과 소프트웨어 공학 발전에 따라 등장한 방법론으로 기업의 전사적인 관점과 데이터 중심으로 시스템을 구축하는 방법론이다.  
    먼저 기업의 정보화 전략을 수집하고, 업무별 DB, 프로세스 모델링, 모델의 실제 설계, 데이터 상세 설계, 프로그램 코딩 순서대로 진행한다.
2. 객체지향 개발 방법론 : 객체지향 개발 방법론에서는 분석, 설계, 구현의 전 과정을 객체 중심으로 진행한다. 심지어 데이터를 저장하는 테이블도 따로 설계하지 않고 데이터 객체로 설계한다. 데이터는 결국 데이터베이스에 저장되는데 만일 데이터베이스가 객체형 DB라면 별다른 변환과정 없이 데이터 객체를 그대로 저장하면 되지만, 관계형 DB를 사용한다면 객체를 관계형 테이블로 변환하는 과정이 필요하다. 이 과정을 객체-관계 매핑(Object Relation Mapping)이라 하며 현재 대부분의 회사에서 관계형 데이터베이스를 사용하고 있기 때문에 객체-관계 매핑은 필수적인 과정이라 할 수 있다.
3. CBD 개발 방법론 : CBD 개발 방법론은 크게 컴포넌트를 개발하는 CD(Component Development) 단계와 개발된 컴포넌트를 사용해서 개발을 진행하는 CBD(Component Base Development) 단계로 나눌 수 있다. CD 단계에서는 도메인을 분석해 컴포넌트 대상 업무를 선별하고 컴포넌트를 개발해 저장소에 입력한다. CBD 단계에서는 요구 분석을 통해 컴포넌트 기반으로 설계하고 필요한 컴포넌트를 저장소에서 찾아서 조립하는 방식으로 프로그램 개발을 진행한다. 만일 필요한 컴포넌트가 저장소에 없다면 CD 단계로 돌아가서 컴포넌트를 개발하고 이를 사용해 개발을 계속 진행한다.
4. 애자일 개발 방법론 : 애자일 개발 방법론은 기존 방법론들이 너무 절차를 중시한 나머지 변화에 대응하기 어려웠던 단점을 개선하기 위해 나왔다. 애자일 방법론은 절차보다는 사람을, 문서보다는 작동하는 소프트웨어를, 미리 철저하게 계획하기 보다는 변화에 대한 민첩한 대응을, 계약과 협상에 얽매이기 보다는 고객과의 협력을 중요하게 생각한다.

**5. MECE**

5-(1) MECE란?

상호 중복이 없고 전체적으로 누락이 없는 (Mutually Exclusive Collectively Exhaustive) 것을 의미한다. MECE는 어떠한 사항을 하나의 기준에 의해 분류하기 때문에 머리가 복잡하거나 정리가 쉽게 되지 않을 경우 전체를 명료하게 이해할 수 있게 도와 준다. 또한 상황을 파악하거나 해결 방안을 도출할 때 한쪽에 편향되지 않고, 전체를 파악하여 문제를 바라볼 수 있게 한다.

5-(2) 내가 생각하는 MECE의 개념

데이터에서 중복이나 누락이 없게 데이터 구분을 잘 정의해 놓는 것을 의미하는 것 같다. 예를 들면, 데이터에 사람을 넣는데, 010-3333-4444라는 번호를 넣은 사람이 있었는데, 다시 010-3333-4444라는 번호를 가진 사람이 또 가입하려고 한다면, 데이터 제약사항에 의해 등록이 안되게, 중복이 안되게 막는 방식으로 방지한다거나, 010-111111-2222222라는 식의 기준과 다른 번호가 들어오면 가입이 불가능 하게 정해놓는 것인 것 같다.

5-(3) MECE의 사례와 현업

사업을 하면서 기업은 고객의 정보를 잘 정리해야 한다. 예를 들면, 3개월 구매금액이 100만원 미만의 고객은 브론즈, 100이상 300이하는 실버, 300이상은 골드라고 정했을 때, 이 테이블에 들어가는 모든 고객데이터에는 누락이 들어가서 null이 입력되거나 중복이 되어 같은 고객의 데이터지만, 어떤곳에는 브론즈, 어떤 곳에는 실버 이렇게 중복되어 데이터의 무결성이 파괴되는 시스템을 만들면 안된다. 따라서 MECE라는 용어에 맞게 기업은 데이터를 저장하고 사용하는데, 누락없이, 중복없이 활용해야 한다.

**6. SW테스트**

6-(1) Test란 시스템이 완전히 통합되어 구축된 상태에서 시스템이 총체적으로 잘 작동 되는지 검사하는 것으로 검사를 통과한 소프트웨어 컴포넌트와 필요한 하드웨어를 통합한 소프트웨어 시스템 전체를 대상으로 한다. 이 과정에서 소프트웨어가 정상 운행 되는지. 하드웨어와의 결합에서 이상이 생기지 않는지를 검사하는 것이다.

6-(2) 테스트 기법

화이트 박스 테스트(코드기반) :   
 statement coverage : 모든 코드가 한번은 실행되게 입력. 이 방식은 코드의 에러여부를 파악한다.

Decision coverage : 코드의 흐름에서 모든 진행을 테스트. 이 방식은 if, for 등의 구문에서 모든 흐름의 입력을 해보는 것이다.

Condition coverage : 조건문이 있을 때 조건과 맞지않는 input을 넣어본다.

Multiple condition coverage : 조건문의 true, false에 모든 상황을 고려, 조건이 (A&B)일 경우 T,T T,F F,F F,T 같이 매개변수가 여러 개인 if문에 사용

블랙 박스 테스트(input, output 기반의 시험)

Syntax testing : 입력에 올바른 값과 올바르지 않는 값을 넣어서 테스트한다

Equivalent partitioning : 입력을 동등하게 쪼개서 각 영역의 대표값을 입력하여 테스트한다. 예시로 성적이 0~100일 때, 0~20은 10, 20~40은 30 을 넣어서 테스트한다.

Boundary testing : 입력값의 경계의 값을 넣어 테스트한다.

Decision table : 입력값의 종류를 결정하여 테스트한다.

6-(3)알파, 베타 테스트

게임 광고에서 종종 듣던 말이 있는데 바로 베타테스트, 베타테스터라는 말이었다. 이 테스트는 게임이 실제 시장에 나오기 전에 게임을 테스트하는 방식이다.

알파와 베타테스트의 차이는 실시하는 시기와 대상에서 차이가 나는데, 알파 테스트의 경우, 게임을 다 만들고 회사 내 관련자들이 실제 게임에서 프로그램이 버그없이 잘 작동하는지 테스트를 실행하는 것이고, 베타 테스트는 정시로 출시하기 전에 사용자들이 사용함에 있어 불편한 점, 버그 등을 찾는 테스트이다. 또한 베타 테스트를 한정된 인원으로 진행하면 클로즈 베타, 모든사람들에게 자유롭다면 오픈베타라고 부른다.

6-(4) 내가 생각하는 테스트

실제로 작년에 안드로이드 프로그램으로 간단한 앱을 만든적이 있다. 실제로 내가 만들고 동작의 방식을 알고 있기에 내가 짯던대로 실행을 시킬때는 이상이 없었다. 하지만, 사용자들은 나와 다르다는 것을 잊었었다. 실제로 발표 때는 이거 눌러봐요 저거 눌러봐요 했을 때, 어? 이게 아닌데? 이런 생각이 많이 들었다. 조그마한 프로그램도 내 생각대로 작동시키지 않는데, 큰 프로그램에서는 얼마나 다양하게 사용할것인가. 그런 점에서 테스트는 잘 작동 되는지를 판단하는 것이 아니라, 얼마나 오류가 있는지를 살피는 것이 테스트인 것 같다.

**7. Pair Programming, Peer Review**

7-(1) 개념  
 Pair Programming : 두 사람이 한 짝이 되어서 같이 프로그래밍을 한다.

Peer Review : 피어리뷰란 특정 학문 영역의 동료 전문가들의 연구를 평가하는 것이다.

7-(2) 내가 생각하는 Pair Programming, Peer Review의 개념   
프로그래밍을 하면서 초보자 뿐만 아니라 전문가조차 타자를 치면서, 혹은 실수로 어떤 잘못된 코드가 삽입이 될 수 있다. 모든 프로그래밍이 완료된 후에 이러한 코드를 찾으려 한다면, 과장이아니라 한달이 넘어가는 시간이 걸리기도 한다고 한다. 따라서 이러한 결함을 줄이기 위하여, 안전하게 2명이 코드를 번갈아 가면서 실수를 줄이는 방식으로 프로그래밍을 하는 것을 Pair Programing이라고

Peer Review는 전문적인 논문에서 그 논문에 이상이 없는지에 대한 토론을 하기도 하고, 이러한 방법을 통하여, 평가를 한 사람과 평가를 받는 사람 모두의 실력이나, 이해의 면에서 상호작용을 통하여 좋은 방향으로의 시너지 효과를 기대할 수 있을 거 같다.

7-(3) 현업에서 어떻게 쓰일지

현업에서는 어떻게 쓰이는지는 잘 찾지 못했지만, 내 생각으로만 적어보자면, 회사 내에서 분업할 때, 팀을 나눈다면, 2인 1조로 나누어서 1팀이 특정 부분의 프로그래밍시에 코드의 문제를 줄이기 위하여 쓸거 같다. 조그마한 한글자라도 틀리다면, 고치는데 오래걸릴 것이기 때문에, 즉시 즉시 고치는 것이 나을 것이기 때문이다.

Peer Review는 공부를 하면서 좋은 방식의 공부 방법일 것 같다. 지금 공부하는 spring의 경우에서도 내가 생각해봤을 때는 이런식으로 작동이 안되야 되는데 잘 되서 교수님에게 물어봤을 때, 해결이 되고 나서 이해가 더 잘된 것처럼, 더 상위의 논문의 경우 이해가 어렵기 때문에, 이러한 토론으로 통해, 이해를 통해 더욱 넓은 견해를 가지게 될 것 같다.

**8. 알고리즘**  
 8-(1) 알고리즘의 개념  
 알고리즘 : 수학과 컴퓨터 과학, 언어학 또는 관련 분야에서 어떠한 문제를 해결하기 위한 일련의 절차나 방법(finite sequence)을 공식화한 형태로 표현한 것, 계산을 실행하기 위한 단계적 절차를 의미한다.

8-(2) 내가 생각하는 알고리즘의 개념  
 알고리즘 : 매우 쉽게 말하면, 라면 끓이는 방법을 설명하면, 1. 물을 끓인다. 2. 스프를 넣는다. 3. 면을 넣고 3분30초를 끓인다. 이렇듯 어떠한 절차나 방법을 이해하기 쉽고, 누구도 부정할 수 없게끔 정의하여, 나타내 놓은 것을 말한다.

8-(3) 현업에서 어떻게 쓰일지

코드의 경우를 살펴보면

K=0;

if(i <= n){

if(j <= n){

while(k <= n){

k = k+1;

}

}

}

이러한 경우에 알고리즘은

1. i가 n보다 적은가?

2. j가 n보다 적은가?

3. k가 n만큼 반복하였는가?

의 알고리즘을 걸쳐서 진행되는데, 컴퓨터가 일단 절차에 따라 앞에서부터 진행되므로, 프로그래밍에서 알고리즘이라는 것은 매우 중요하다고 볼 수 있다.

**9. 튜닝**  
 9-(1) 튜닝의 개념  
 튜닝 : 조율·개조가 기본 뜻.기성 제품 외관을 바꾸거나 성능을 변화시키는 것을 말한다.  
  
 9-(2)내가 생각하는 튜닝의 개념

IT쪽에서 튜닝이라는 것은 기존의 시스템이나 DB와 같은 서버, 하드웨어 같은 것의 성능을 변화 시키기 위하여, 바꾸거나 개조하거나 업그레이드 하는 것을 튜닝이라고 하는 것 같다.  
   
  
 9-(3) 튜닝의 사례

일단 DB의 경우를 살펴보면, 시스템 성능저하의 요인 중 DB와 관련된 것은 90%, 시스템 디자인이 7.5%이며, 하드웨어의 요인이 2.5%로 DB가 거의 대부분의 요인이라고 볼 수 있는데, 이러한 경우에 DB의 응용, DB자체등을 통하여, 최적의 성능을 얻을 수 있다.

DB 튜닝의 경우에는 각각 DB설계 영역, DBMS 영역, SQL 영역으로 나뉘는데,

DB설계 영역에서 테이블 분할 및 통합, 효율적 인덱스 설정, 데이터 모델링 등을 효율적으로 바꿈으로써 성능을 향상시킬 수 있으며,

DBMS 영역에서는 Buffer Pool 튜닝, Thread, 트랜잭션 등의 설정이 가능하다.

마지막으로 SQL 영역에서는 옵티마이저, 조인방식/순서, 다중처리, 병렬처리를 함으로써 향상시킬 수 있다.

**10. 서브네팅, 슈퍼네팅의 관계**

10-(1) 서브네팅,슈퍼네팅의 개념  
 서브네팅 : 하나의 LAN을 여러 개의 작은 네트워크로 분할 한다는 뜻이다.   
 슈퍼네팅 : 여러개의 네트워크를 하나의 네트워크 형태로 합친 네트워크로 합치는 것  
  
10-(2) 내가 생각하는 서브네팅, 슈퍼네팅의 개념  
 정보통신을 배우면서 한 가정, 한 회사에 한 개의 IP만이 설정된다고 들어간다고 들은 적이 있다. 내가 하는 IP라는 것은 컴퓨터의 주소를 나타내는 유일한 주소라는 것이었는데, 회사나 집에 여러개의 컴퓨터를 사용한다면, 이런 경우에는 어떻게 할까? 이 부분에서 서브네팅이라는 것을 하는 것 같다. 하나의 IP를 여러 개로 쪼개는 것이다. 그러면서 이러한 쪼개져서 들어간 호스트를 구분하기 위해서는 서브넷 마스크를 쓰는 것 같다.

슈퍼네팅의 경우에는 부족한 네트워크에 대하여 여러 네트워크를 묶어서 느리게 처리되는 네트워크를 빠르게 연결 할 수 있는 방법이라고 생각한다.

10-(3) 현업에서 어떻게 쓰일지

서브네팅의 경우에는 하나의 IP에서 여러 호스트 컴퓨터에게 할당해주는데, 이는 하나의 IP를 하나의 컴퓨터에게 할당해주면, 전세계에 42억개밖에 없는 IP의 개수에 한계가 있기에, 문제가 발생한다. 따라서 네트워크 단의 A,B,C클래스를 두고 호스트를 지정하는 호스트 네트워크와 서브넷 마스크를 이용하여 IP를 쪼개고 호스트 컴퓨터를 구분하는 방식으로 쓰일 것 같다.

**11. RPA, MDD**

11-(1) RPA, MDD 의 개념  
 RPA : 사용자 PC 상에서 이뤄지는 단순 반복 업무를 미리 정해진 Workflow에 따라 자동으로 수행하게 해 주는 소프트웨어이다.

MDD : 사람이 프로그램 작성의 기본 틀이 되는 모델을 만들면, 이후 프로그램 코딩 등의 세부작업을 알아서 자동으로 처리하는 프로그래밍 기법이다.

11-(2) 내가 생각하는 RPA, MDD 의 개념  
 RPA : 매크로 프로그램이 등장하게 된 배경과 비슷하다고 생각한다. 본디 매크로라는 것은 단순반복을 사람이 하기 귀찮아서 미리 정해진 그대로, 반복하여 동작하게 만드는 것이었는데, 그 부분에서 더 발전시키거나 아이디어를 얻어서 업무에 적용한 소프트웨어이지 않을까 싶다.

MDD : 2학년 여름방학의 경험에 의하면, MDD는 원래 코드기반의 프로그래밍을 그래픽기반의 프로그래밍으로 사용자가 눈에 보이게 모델객체를 끌어다가 다른 객체와의 관계를 설정하고, 동작을 지정해 놓으면, 그 지정한 대로, 프로그램이 짜여진 그대로 프로그램을 만드는 방식으로 프로그래밍을 하는 방법이다.

11-(3) RPA, MDD 사례   
 RPA 사례 : 금융 B사의 경우 부서 40개 이상의 업무를 자동화했다. RPA 고도화를 통해 향후 5년간 최소 92억원 이상의 비용을 절감할 수 있을 것으로 예상된다. 적용된 부분으로는 업무량 상위의 외환업무부, 퇴직연금부, 금융결제부 등의 부서의 자동화 요건을 충족시키기 위해 우선 적용하였으며, 이후에 AI와 접목하여 한단계 더 진화된 자동화를 추진하려고 한다.

RPA의 적용에서 주의할 점은 라이선스 비용 효율화, 운영 안정화 및 관리, OCR적용에 있어서의 이슈 사항등이 있으며,

MDD 사례 : Conquest, Inc and Popkin, Inc(미국 IT서비스 업체)기업은 미국 정보 공공기관에서 SI시스템과 SW솔루션을 제공하는 분야에서 MDD기법을 적용하였으며, 이는 기존 시스템을 모델기반의 아키텍처로 구축하였으며, 시스템에 대한 문서를 자동 생성 다이어그램을 적극 활용하여 커뮤니케이션 효율을 높였다.