# 시스템 분석설계
# 교과목 포트폴리오

20181766 강석주

# 목차

Part 1,

# 소프트웨어 공학

# 1. 소프트웨어의 개요

소프트웨어는 하드웨어를 동작 시켜 사용자가 작업을 편하게 수행하도록 하는 프로그램과 자료구조 등을 총칭한다.

소프트웨어는 프로그램 자체 뿐만 아니라 프로그램의 개발, 운용 및 유지보수에 관련된 모든 문서와 정보를 포함한다

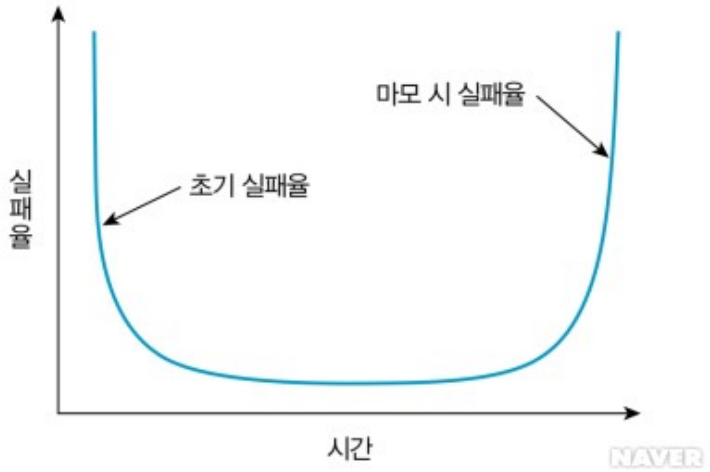
# 2. 소프트웨어의 특징

**제조가 아닌 개발**

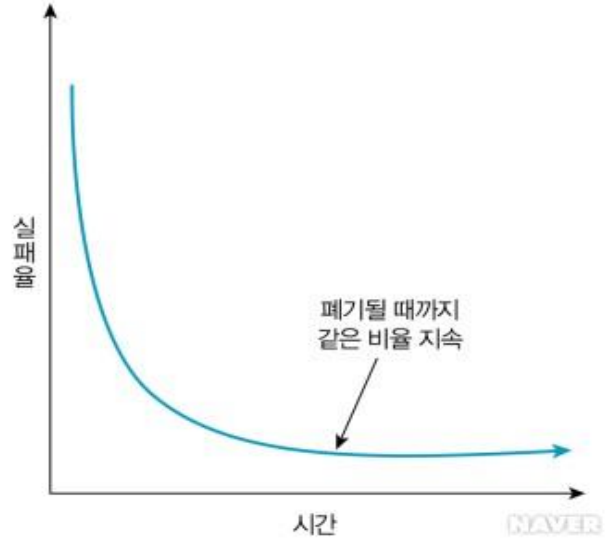
**제조는 정해진 틀에 맞춰 일정하게 생산하는 것을 말하지만 소프트웨어는 개인간의 능력차이에 따라 나오는 결과물이 다르다.**

**소모가 아닌 품질 저하**
**하드웨어 같은 경우는 부품이 닳고 기능이 떨어지지만 소프트웨어는 물건처럼 닳는게 아니라 시간이 지나 품질의 저하가 되는 것 이다.**

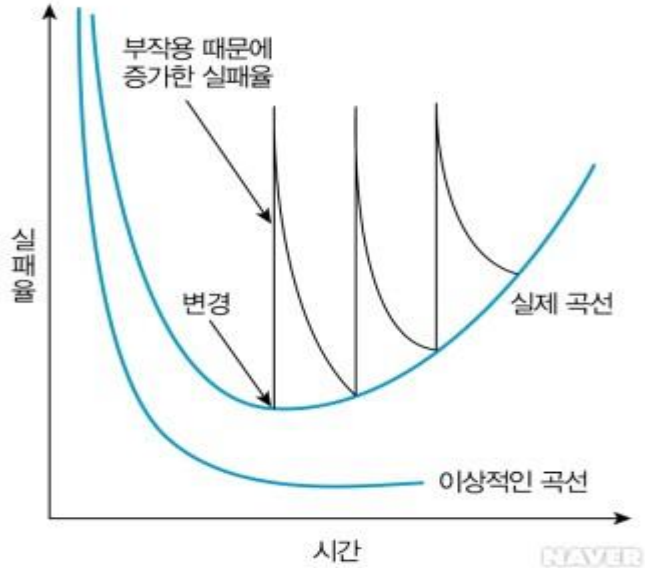
Lorem Ipsum is simply dummy text of the printing and typesetting industry




하드웨어의 실패 곡선




이상적인 소프트웨어 실패 곡선




실제 소프트웨어의 실패 곡선

# 3. 소프트웨어의 개발 속도

**하드웨어보다 소프트웨어의 개발 속도가 느리다**

**하드웨어와 소프트웨어의 근본적인 개발 방법의 차이**

**하드웨어 : 검증받은 부품을 조립하는 형태의 생산**
**소프트웨어 : 처음부터 만들어가는 개발 형태**

**해결방안으론 CBD 개발 방법론이 있다**

* CBD 개발 방법론이란
컴포넌트를 조합해 재사용 함으로서 개발 생산성과 품질을 높이고 시스템 유지보수 비용을 최소화할 수 있는 개발방법론

# 4. 소프트웨어의 개발 생명주기(SDLC)

**폭포수모델(Waterfall)**
**정해진 단계 선형 순차적 모델, 분석, 설계, 구현, 시험 및 유지보수 과정으로 접근한다**

**계획 -> 요구분석 -> 설계 -> 구현 -> 테스트 -> 유지보수로 이루어져 있다**

# 5. 프로젝트 관리와 지식 체계의 9가지 관점

프로젝트 관리
형상 관리: 문서 관리를 중심으로 하는 관리

PMBOK(Project Management Body of Knowledge)

1. 프로젝트 통합 관리
2. 프로젝트 범위 관리
3. 프로젝트 일정 관리
4. 프로젝트 비용 관리
5. 프로젝트 품질 관리
6. 프로젝트 인적자원 관리
7. 프로젝트 의사소통 관리
8. 프로젝트 위험 관리
9. 프로젝트 조달 관리

# 6. WBS(Work Breakdown Structure)

‘작업 분할 구조도’ 라고 하며 프로젝트 목표 달성을 위하여 필요한 업무들과 액티비티들을 세분화 하는 작업이다. 계층 구조 중 최하위에 있는 항목은 작업 패키지라고 하며, 이 작업 패키지는 담당자를 할당할 수 있을 정도로 작게 나눠야 한다.

목적
사용자와 개발자 간의 의사소통 도구로 사용함
프로젝트 업무 내역을 가시화 할 수 있어 관리가 용이함
프로젝트 팀원의 책임과 역할이 분명함
필요 인력과 일정 계획을 세우는 데 기초로 활용함

Part 2

# 깃과 깃허브

# 1. Git이란

컴퓨터 파일의 변경사항을 추적하고 여러 명의 사용자들 간에 해당 파일들의 작업을 조율하기 위한 형상 관리 도구이다. 간단히 말하자면 git을 통해 파일의 변경사항을 확인 및 적용하고 그 차이점을 확인할 수 있다.

Git을 사용하는 이유
많은 사람들과 협업 시 각각의 사람들이 branch를 따고,
각각의 branch에서 변경사항을 문제없이 적용하여, main에 병합하는 것 등 개발에 매우 유용하다

# 1. Git이란

컴퓨터 파일의 변경사항을 추적하고 여러 명의 사용자들 간에 해당 파일들의 작업을 조율하기 위한 형상 관리 도구이다. 간단히 말하자면 git을 통해 파일의 변경사항을 확인 및 적용하고 그 차이점을 확인할 수 있다.

Git을 사용하는 이유
많은 사람들과 협업 시 각각의 사람들이 branch를 따고,
각각의 branch에서 변경사항을 문제없이 적용하여, main에 병합하는 것 등 개발에 매우 유용하다

# 2. Git관련 주요 용어

**저장소(Repository)**
소스코드가 저장되어 있는 여러 개의 branch들이 모여 있는 디스크상의 물리적 공간

**Checkout**
특정 시점이나 브랜치의 소스코드로 이동하는 것을 의미

**Stage**
작업한 내용이 올라가는 임시 저장 영역

# 2. Git관련 주요 용어

**Commit**
작업한 내용을 로컬 저장소에 저장하는 과정

**Tag**
커밋의 임의 위치에 쉽게 찾아갈 수 있도록 붙여 놓은 이정표

**Push**
로컬 저장소의 내용 중 원격 저장소에 반영되지 않을 커밋을 원격 저장소로 보내는 과정

# 2. Git관련 주요 용어

**Pull**
푸시와 반대로 원격 저장소에 있는 내용 중 로컬 저장소에 반영되지 않은 내용을 가져와서 로컬에 저장

**Branch**
커밋을 단위로 구분된 소스코드 타임라인에서 분기해서 새로운 커밋을 쌓을 수 있는 가지를 만드는 것

**Merge**
브랜치와 반대되는 개념으로 하나의 브랜치를 다른 브랜치와 합치는 과정

# 3. Git의 기초

Git의 스냅샷
Git은 데이터 파일 시스템 스냅샷의 연속으로 취급하고 크기가 아주 작다
파일이 달라지지 않으면 git은 성능을 위해서 파일을 새로 저장하지 않는다(이전파일의 링크만 저장)
Git은 데이터를 스냅샷의 스트림처럼 취급

로컬에서 실행 가능
프로젝트의 모든 히스토리를 로컬 디스크에서 관리
거의 모든 명령이 로컬 파일과 데이터로만 사용 가능하다
네트워크에 있는 다른 컴퓨터는 필요가 없다

# 3. Git의 기초

Git의 스냅샷
Git은 데이터를 저장하기 전에 항상 체크섬을 구하고 그 체크섬으로 데이터를 관리한다
체크섬은 git에서 사용하는 가장 기본적인 데이터 단위이자 git의 기본 철학이다
체크섬 없이 어떠한 파일이나 디렉토리도 작업할 수 없다
Git은 sha-1 해시를 사용하여 체크섬을 만든다

Git은 데이터를 추가하는 방식
Git으로 무엇을 하던지 git 데이터베이스에 데이터가 추된다
되돌리거나 데이터를 삭제할 방법이 없다

# 4. Git의 상태

Git은 파일을 Committed, Modified, Staged 이렇게 세가지 상태로 관리한다.

Committed : 데이터가 로컬 데이터베이스에 안전하게 저장되었다는 것을 의미
Modified : 수정한 파일을 아직 로컬 데이터베이스에 커밋하지 않은 것을 말한다
Staged : 현재 수정한 파일을 곧 커밋할 것이라고 표시한 상태

# 5. Git으로 하는 일

워킹 트리에서 파일을 수정
내용 및 개발 수정 작업
수정했지만, 아직 Staging Area에 추가하지 않은 상태이기에 Modified

Staging Area에 파일은 Stagego서 커밋할 스냅샷 생성
Add index로 staged changes 부분에 커밋할 내용을 올리는 작업(Staging area에 추가했다면 Staged)

Staging Area에 있는 파일들을 커밋해서 Git 디렉토리에 영구적인 스냅샷으로 저장한다
Add index 할 파일들을 commit하는 작업(git 디렉토리에 있는 파일들은 Committed 상태)

# 6. GitHub란

분산 버전 컨트롤 소프트웨어(git)을 기반으로 소스 코드를 호스팅하고, 협업 지원 기능들을 지원하는 마이크로소프트의 웹서비스이다.

GitHub는 소스코드가 공개되는 경우 무료로 사용할 수 있다.

비공개 저장소를 유료 사용자만 사용할 수 있었지만 마이크로소프트에 인수된 후 무료 사용자도 비공개 저장소를 사용할 수 있도록 개편되었다.

# 6. GitHub의 서비스들

1. 저장소
깃허브의 핵심 기능은 git원격 저장소 호스팅이다. 깃허브를 통해 로컬 개발 환경과 온라인에서 안전하게 깃 저장소에 접근할 수 있다.

2. 페이지
페이지는 깃허브 저장소를 기반으로 정적 파일들을 호스팅할 수 있는 서비스이다. 파일을 웹상에 공개하거나 저장소의 특정 브랜치에 파일들을 올려두고 웹사이트로 공개할 수 있다.

# 7. Fork

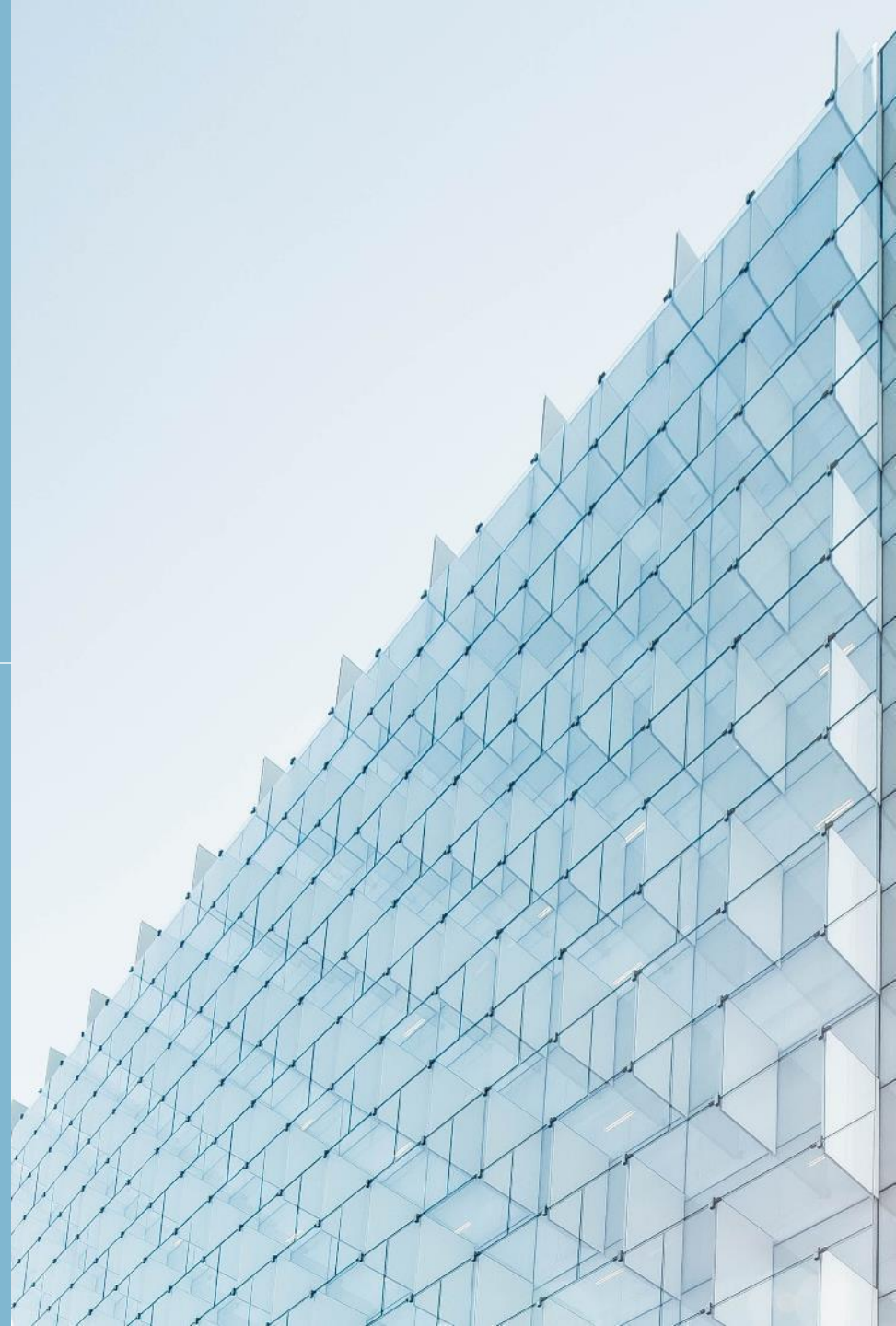
1. Fork란 다른 사람의 github 저장소를 복제하여 내가 어떤 부분에 수정, 추가, 삭제를 용이하도록 나의 Github 저장소로 그대로 복제하는 기능이다.(다른 사람의 github 저장소와 연결)

2. 이후 원래 저장소에 변경사항을 적용하고 싶으면 해당 저장소에 pull reques를 해야한다.
한 후 관리자로부터 승인되었으면 내가 만든 코드가 commit, merge되어 원본에 반영된다.

Part 2,

# HTML 개발 절차

Lorem Ipsum is simply dummy text of the printing and typesetting industry



# 1. 서비스 기획

1. 아이디어 구체화
아이디어를 구체적으로 정리를 한다. 사용자의 needs에 맞게 또한 어떠한 페이지를 원하는지를 정리한다.

2. 페이지의 틀 구상
페이지에 들어갈 내용들이 어떠한 것들이 있는지 구상을 하고 어디에 들어갈지 구상을 한다.

3. 프로젝트 인원 구성
웹 개발은 보통 여러명이 협업하여 진행하기 때문에 프로젝트에 필요한 역할인 css, 구조, 서버, 테스터를 따로 두기도 한다.

# 2. 역할

구조 : 전체 큰 페이지들의 구조를 짜며 각각 요소들이 들어갈 div, header등 페이지의 틀을 구성한다

Css : 웹 페이지를 디자인하며, 각각 페이지에 맞는 스타일 요소들을 생성한다.

서버 : 제작한 페이지를 서버와 연동하며 가져오거나 내보내는 정보들을 서버에 저장하는 역할을 한다.

테스터 : 완성되기 전 페이지를 사용하며 추가사항이나 오류점을 찾아내 주는 역할을 한다.

Part 3

# HTML 제작 절차

| STEP 1 | STEP 2 | STEP 3 | STEP 4 |
| --- | --- | --- | --- |
| 클라이언트에게 수주를 받고 제작에 대한 견적을 짠다 | 웹 구조를 짠 후 css적용을 하여 서버와 연동을 하여 페이지를 제작한다. | 테스터가 테스트를 하여 추가 수정사항 본을 하여 최종 검토를 한다 | 클라이언트에게 페이지를 전달하고 필요한 수정부분 및 페이지 작업을 완료한다. |

Part 3,

# HTML 개발 절차

## 구조

- 웹 페이지의 큰 틀 구성
- 여러 웹 페이지를 구성

## 서버

- 각종 데이터 연동 및 구성
- 받아온 데이터들 출력

## css

- 웹 내부 디자인
- Css파일 용량 최적화
- Css파일 분할

## 테스터

- 웹 시연
- 수정사항 및 오류사항 찾기


감사합니다