시스템분석설계 개인 포트폴리오

컴퓨터정보공학과 2-A 20200824 현지예

CONTENTS

01 소프트웨어공학의 이해

프로그램이란?

- 컴퓨터에서 실행될 때 특정 작업을 수행하는 일련의 명령어들의 모음(집합체)이다.
  - 원시 코드 (Source Code)
    - C언어, JAVA, Python 등의 언어로 쓰인다.
- 컴파일된 결과물 뿐만 아니라, 프로그래머가 작성한 소스 코드까지도 포함한다.
- 원시코드, 모든 산출물(자료구조, DB구조, 테스트 결과 등), 각 단계마다 생산되는 문서, 사용자 매뉴얼 등이 있다.

소프트웨어(Software)란?

- 응용 프로그램과 데이터처럼, 컴퓨터의 하드웨어(Hardware) 상에서 구동되거나 처리되는 무형물을 지칭하는 말이다.
- 컴퓨터의 시스템을 구성하는 주요 요소 중 하나이다.
  (컴퓨터 = 하드웨어 + 소프트웨어 + 펌웨어)
  - 펌웨어란?
    - 컴퓨터의 CPU가 아니라 그보다 한참 하위 단계의 장치들을 제어하는 특수한 소프트웨어
- 원시코드, 모든 산출물(자료구조, DB구조, 테스트 결과 등), 각 단계마다 생산되는 문서, 사용자 매뉴얼 등이 있다.
  - 소프트웨어는 프로그램 뿐만 아니라 그 이상의 것도 포함하는 매우 포괄적인 개념이다.
  - 범위 : 프로그램 < 소프트웨어

소프트웨어 공학이란?

- ***정의***
  - 소프트웨어 + 공학
  - 소프트웨어의 개발, 운용, 유지보수 및 폐기에 대한 체계적인 접근 방법을 연구하는 학문이다.
- ***특징***
  - 소프트웨어 개발 전 과정에 걸쳐 필요한 이론, 개념 및 기술을 다룬다.
  - 소프트웨어 개발 과정에서 생성되는 모든 산출물이 그 대상이 된다.
- ***목적***
  - S/W 개발의 어려움을 해결한다.
  - 효율적 개발을 통한 생산성을 향상시킨다.
  - 고품질 소프트웨어 제품을 만든다.

소프트웨어의 특징

● **제조가 아닌 개발을 필요로 한다.**

- **제조**
  - 정해진 틀에 맞춰 일정하게 생산하는 것이다.
  - 많은 인력이 필요하며, 능력별 결과물의 차이가 근소하다.
- 개발
  - 새로운 것을 창조해내는 것이다.
  - 개발하는 개인 능력에 따라 결과물의 차이가 매우 크다.

● **많이 사용해도 소모되지 않는다.**

- H/W
  - 오래 사용하면 부품이 닳고, 고장 발생 빈도가 높아지며, 기능도 떨어진다.
- S/W
  - 오래 많이 사용한다 해도 닳지 않고, 고장 빈도도 낮다.
  - 시간이 지나면서 다른 좋은 소프트웨어가 개발되어 사용 빈도가 줄어드는 것이지, 기능은 동일하다.

소프트웨어가 사용되는 곳

- *금융*
- *노트북*
- *의료*
- *항공기*
- *건물*
- *스마트폰*
- *자동차*

● 스마트폰

- **앱**
  - 스마트폰의 앱은 컴퓨터의 소프트웨어와 같다.
  - IOS 체제 & 안드로이드(JAVA) 체제가 존재한다.
  - objective-c 라는 언어로 개발되었다.
  - 이제는 swift 언어로 바뀌고 있다. (애플에서 개발)
  - 각종 앱은 온라인 스토어에서 다운받아 사용할 수 있다. (애플의 앱스토어, 안드로이드의 T스토어)

● 자동차

- **휘발유 자동차**
  - 휘발유 차에는 S/W가 10% 정도만 사용된다.
  - 시간이 지날수록 휘발유 차의 수요가 줄어들 것으로 예상된다.
- **전기 자동차**
  - 전기 차에는 S/W가 95% 정도가 사용된다.
  - 앞으로 전기 차의 수요는 늘어날 것으로 예상된다.

소프트웨어 개발 단계

●1단계 : 계획

- 개발 비용 산정
  - COCOMO모델, 기능점수(FP)모델 사용
  - COCOMO모델
    + 비용추정 모델로서 소프트웨어 개발비, 유지보수의 비용 견적, 개발 단계 및 업무 활동 등 비용 견적의 유연성이 높아 가장 널리 통용된다.
    + Basic cocomo, Intermediate cocomo, Detailed cocomo로 나뉜다.
  - 기능점수(FP)모델
    + 소프트웨어의 생산성을 측정하기 위해 개발되었다.
    + 모델 구성을 위한 방법은 시스템에서 발생하는 5가지 항목을 조사하여 작성한다.
      (입력 유형의 수, 출력 유형의 수, 사용자 명령어의 수, 데이터 파일의 수, 인터페이스의 수)
- 일정 계획
  - 작업분할구조도(WBS), CPM 사용
  - WBS
    + 프로젝트 관리 및 시스템 엔지니어링에서 프로젝트를 소규모의 구성 단위까지 나누어 전달할 수 있는 기능을 말한다.
  - CPM
    + 캠페인 기간 동안 일어난 노출 1,000회당 지불하는 비용을 뜻한다.

- **SLDC**

1. **계획**
2. 요구분석
3. 설계
4. 구현
5. 테스트
6. 유지보수

소프트웨어 개발 단계

●2단계 : 요구분석

- 기존 시스템의 문제점 파악 -> 새로운 요구사항 도출
  -> 다이어그램 작성
- 개발 방법론에 따른 표현 도구
  - 구조적 방법론 : DFD, DD, Mini Spec
  - 정보공학 방법론 : E-R 다이어그램
  - 객체지향 방법론 : UML의 유스케이스 다이어그램
- 최종 산출물 : 요구 분석 명세서

- **SLDC**

1. 계획
2. **요구분석**
3. 설계
4. 구현
5. 테스트
6. 유지보수

소프트웨어 개발 단계

●3단계 : 설계

- 설계 원리
  - 분할과 정복, 추상화, 단계적 분해, 모듈화, 정보은닉
  - S/W 설계는 표준화가 잘 되어있지 않음
    + 설계도면만으로는 제대로 구현 할 수 없다.
      (S/W의 큰 문제점이다.)
  - 분할과 정복
    + 어려운 큰 문제를 작고 쉬운 문제로 분할하여 정복한다.
  - 추상화
    + 고수준일수록 사람(예 : 음료자판기 사용자)에 가깝고,
      저수준일수록 기계(예 : 자판기 수리기사)에 가깝다.
- 소프트웨어 아키텍처, 객체지향 설계
- 아키텍처 스타일
- GoF의 디자인 패턴
- 모듈 평가 기준 : 응집도와 결합도

- **SLDC**

1. 계획
2. 요구분석
3. **설계**
4. 구현
5. 테스트
6. 유지보수

소프트웨어 개발 단계

●4단계 : 구현

- 간략한 프로그래밍 언어의 역사
- 표준 코딩 규칙

●5단계 : 테스트

- 테스트의 절차
- 개발자 또는 사용자 시각에 따른 분류
- 사용되는 목적에 따른 분류
- 품질에 따른 분류
- 소프트웨어 개발 단계에 따른 분류

●6단계 : 유지보수

- 수정 유지보수
- 적응 유지보수
- 기능보강 유지보수
- 예방 유지보수

- **SLDC**

1. 계획
2. 요구분석
3. 설계
4. **구현**
5. **테스트**
6. **유지보수**

하드웨어와 소프트웨어의 실패 단계

- **H/W의 실패 단계**
  (그래프는 욕조 모양의 곡선을 띈다.)
  - 초기 실패율이 높다.
  - 오류가 해결된다.
  - 오랜 기간동안 사용한다.
  - 주변 환경 문제가 발생한다.
    - 먼지가 쌓이거나, 부품이 부식되는 등의 문제가 발생한다.
  - 다시 실패율이 증가한다.

- **S/W의 이상적인 실패 단계**
  (그래프가 지속적으로 감소하는 모양의 곡선을 띈다.)
  - 발견되지 않은 오류들로 초기 실패율은 높다.
  - 오류가 해결되면 폐기될 때까지 실패율이 지속된다.
  - 이상적인 상황
    - 개발 완료 후 변경 사항과 환경 변화가 없어야 한다.

- **S/W의 실제 실패 단계**
  (그래프가 감소하다가 다시 급격히 증가하는 모양의 곡선을 띈다.)
  - 실패율이 감소하는 부분까지는 이상적인 실패 단계와 비슷하다.
  - 실제로는 변화(기능 추가 및 수정)의 부작용으로 실패율이 급격하게 증가한다.

소프트웨어 개발의 어려움

- **물리적인 형태가 없는 무형의 논리적인 요소이다.**
  - 개발 과정에 대해 정확하게 이해하기 어렵다.
  - 개발 진행 상황을 파악하기도 어렵다.
- **최종 산출물이 개발 과정에서 확인되지 않는다.**
  - 오류를 발견해야 할 시기를 놓치거나, 오류에 대한 해결책을 못 찾는 경우가 발생한다.
  - 참여 인력도 많기 때문에 개발 중 의사소통이 복잡하다.
- **프로젝트 개발 기간이 길다.**
  - 개발을 진행하다 보면 지연이 되거나, 예상 범위가 초과되기도 한다.
  - 기간이 길어질수록 진행 관리와 개발 비용 선정도 어려워진다.


FINANCE REPORT
ACCOUNT REPORT
DASHBOARD > INCOME
DAILY WEEKLY MONTHLY
5M$
4M$
3M$
2M$
1M$
MON TUE WED THU FRI SAT SUN
TOTAL INCOME
37.91%
31.86%
30.23%
LINE ITEMS 16.15 M$
SHIPPING 0.15 M$
TAXES 0%
TOTAL 16.3 M$
LINE ITEMS 13.5 M$
SHIPPING 0.2 M$
TAXES 0%
TOTAL 13.7 M$
LINE ITEMS 13.00 M$
SHIPPING 0 $
TAXES 0%
TOTAL 13.00 M$

02 깃과 깃허브

Git이란?

- 로컬에서 관리되는 버전 관리 시스템
  (VCS : Version Control System)
- 소스코드 수정에 따른 버전을 관리해주는 시스템

Github란?

- 클라우드 방식으로 관리되는 버전 관리 시스템
  (VCS)
- 자체 구축이 아닌 빌려쓰는 클라우드 개념
- 오픈소스는 일정 부분 무료로 저장 가능
  아닐 경우 유료 사용

버전 관리란?

- 소스 하나 또는 묶음을 하나의 버전으로 간주한다.
- 파일이나 폴더를 추가, 수정, 삭제하며 사람이 직접 관리한다.
- 원할 때 예전 버전 내용 전체를 되돌려 볼 수 있으며 복잡한 코드를 개발할 때 이전 버전과 비교하기 쉽다.

버전 관리가 필요한 이유

- 개발자 사이의 협업에 필요하다.
- 전체 개발 소스를 공유하면서 개발 피트를 나눌 수 있고 같은 모듈을 개발하더라도 소스를 서로 공유하며 개발할 수 있습니다.

- 옥토캣
- 깃허브(Github)의 마스코트

깃허브의 오픈 소스 프로젝트

아톰 (Atom)

- 아톰은 깃허브에서 주도해서 개발하고 있는 오픈소스 에디터이다.
- 2014년 처음 공개되었으며, GUI 텍스트 에디터로 많은 주목을 받았다.
- 2018년 깃허브가 마이크로소프트에 인수되면서 마이크로소프트에서 개발한 비주얼 스튜디오 코드와 포지션이 겹치지만 현재도 개발 중이다.

일렉트론 (Electron)

- 웹에서 주로 사용하는 자바스크립트, HTML, CSS를 사용해 크로스 플랫폼 데스크탑 애플리케이션을 만들 수 있는 프레임워크이다.
- 대표적으로는 비주얼 스튜디오 코드, 슬랙, 노션, 디스코드 앱에서 사용 중이다.

깃허브의 오픈 소스 프로젝트

허브 (Hub)

- 깃 명령어를 확장해 CLI에서 깃허브를 사용할 수 있도록 도와주는 명령어입니다.

시멘틱 (Semantic)

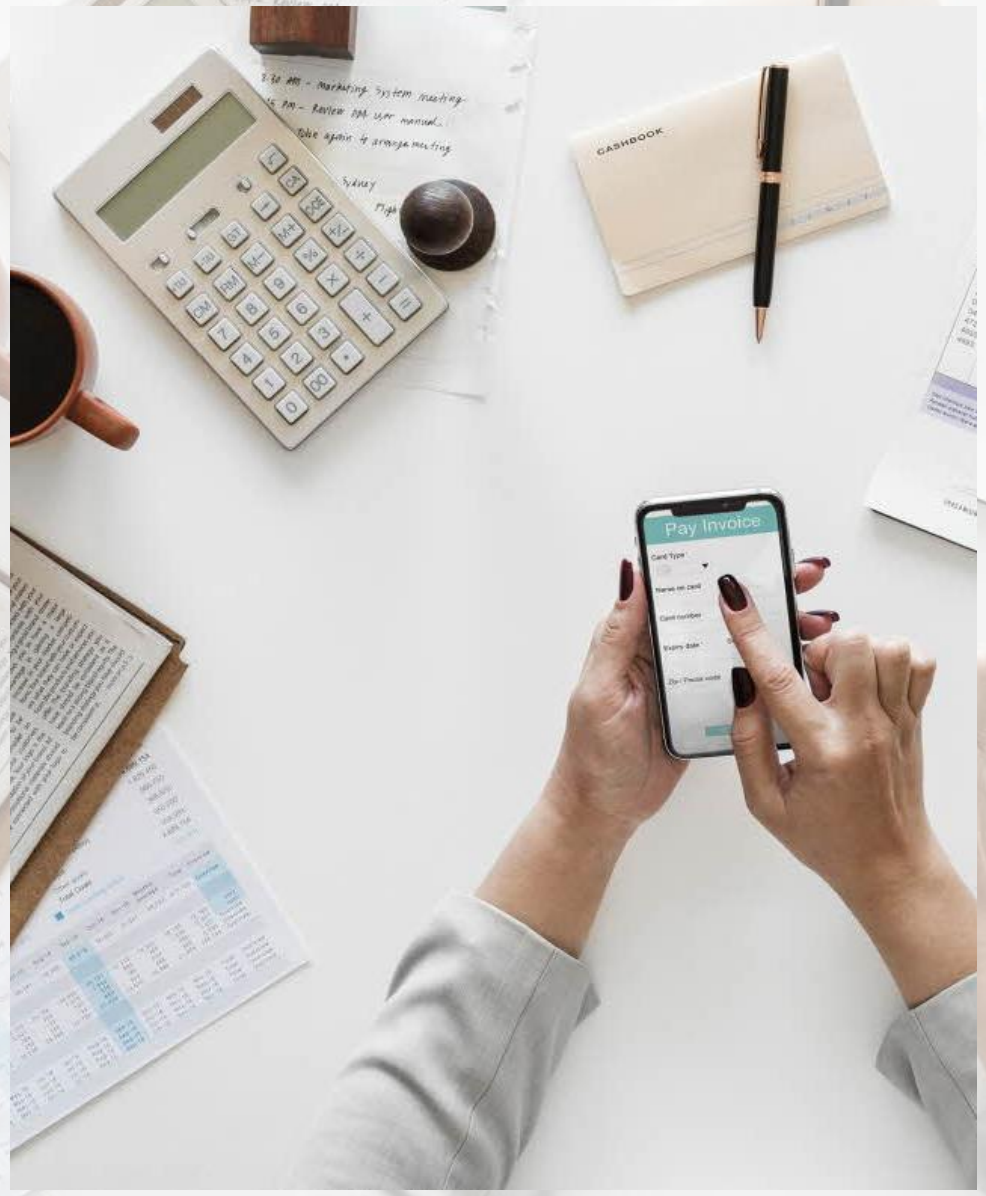
- 언어 실행기없이 소스 코드를 파싱, 분석, 비교할 수 있는 도구이다.
- 루비, 자바스크립트, 타입스크립트, 파이썬, 고 등을 지원하며 구현은 하스켈로 되어있습니다.

03 공용 API의 사용 (네이버 지도)

Open API

(Open Application Programming Interface)

- 인터넷 이용자가 일방적으로 웹 검색 결과 및 사용자인터페이스(UI) 등을 제공받는데 그치지 않고 직접 응용 프로그램과 서비스를 개발할 수 있도록 공개한 API를 뜻한다.
  - 시간 및 비용이 절감된다.
  - 제공 내용
    + 지도, SNS, 음악, 공공데이터, 비즈니스, 날씨, 쇼핑 등등
  - 제공처
    + 네이버, 다음 카카오, 구글, 페이스북, 정부
  - 승인된 인증키가 있어야만 Open API 사용 가능



네이버 지도의 활용

- 네이버 지도 API 활용 절차
  - 네이버 지도를 활용하는 프로젝트 개발을 위해서는 네이버 클라우드 플랫폼에 회원가입을 하고 개발 앱에 대한 API키를 발급받아야 한다.
  - 회원 정보는 한 번만 등록하면 되지만, API키는 프로젝트마다 새로 신청해야 한다.
- 네이버 지도 활용의 원리
  - 네이버 클라우드 플랫폼에서 지도 API를 발급받는다.
  - 엑티비티의 레이아웃 속에 MapFragment를 생성한다.
  - Fragment를 인식하면 지도 API 가 보여진다.

THANKS!