쉽게 배우는

소프트웨어 공학

Chatpter 01

소프트웨어 공학 소개

- 소프트웨어의 특징을 살펴본다 .
- 소프트웨어 공학의 뜻을 이해한다 .
- 소프트웨어 개발 단계를 알아본다

Section 01 소프트웨어의 이해

1. 소프트웨어가 사용되는 곳

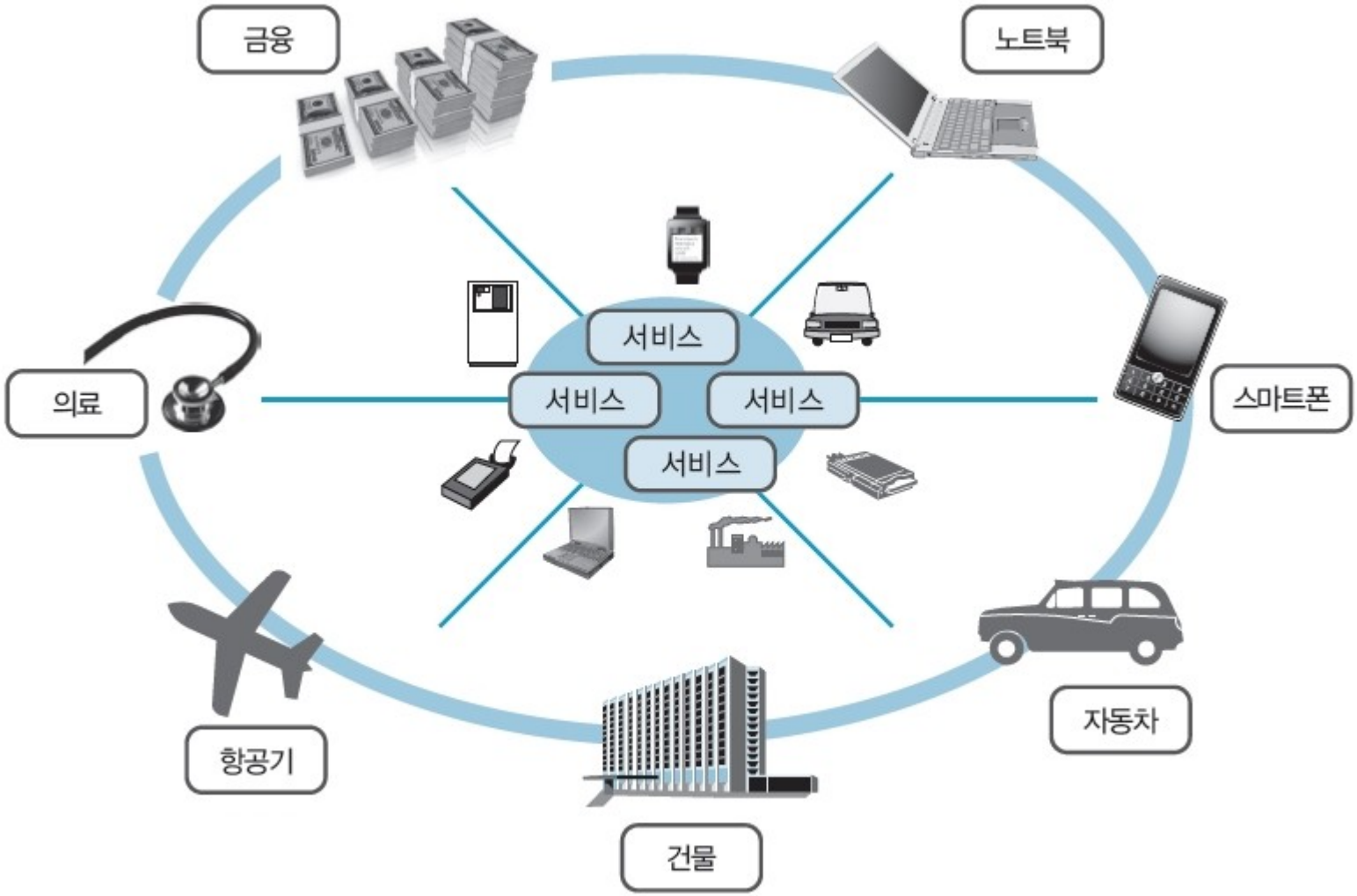


그림 1-1 소프트웨어가 사용되는 곳

2. 프로그램과 소프트웨어

- **프로그램**
  - 원시코드 source code

- **소프트웨어** = 좁은의미 + α ①
  - → 프로그램 뿐만 아니라 그 이상의 것도 포함하는 매우 포괄적인 개념 (넓은 의미)
  - ② 좁은 의미 : 프로그램 (원시코드 source code)
    - 모든 산출물 (자료구조 , DB 구조 , 테스트 결과 등)
    - 각 단계마다 생산되는 문서
    - 사용자 매뉴얼

α (가시적) • 눈에 보이는 결과물.

넓은의미

컴퓨터 < S/W ← data
H/W

3. 소프트웨어의 분류

data 자료 : 측정가능한 값
↳ information 정보
가공처리

관리 소프트웨어

- 자료를 받아들여 가공한 후 정보를 제공하는 소프트웨어
- 주로 DB 에 자료를 저장한 후 검색을 통해 사용자가 원하는 형태로 정보를 제공

(예) 인터넷뱅킹 시스템 , 대학의 종합정보 시스템 , 예약 시스템 등

제어 소프트웨어 → 실시간 (기기 제어 + 센서를 이용)

- 각종 센서를 이용하거나 기기들의 동작을 제어하는 소프트웨어

(예) 교통 신호 제어 , 의료기기 제어 공장장비 제어 등

임베디드 소프트웨어 * embedded

- 장비나 기기에 내장된 형태의 소프트웨어

C언어로 많이작성

(예) 가전제품내의 소프트웨어 , 각종 공정제어 시스템내의 소프트웨어

4. 소프트웨어의 특징

- **제조가 아닌 개발**
  - 제조 : 정해진 틀에 맞춰 일정하게 생산하는 것으로 , 많은 인력이 필요하고 능력별 결과물 차이가 근소함
  - 개발 : 개인 능력 별 결과물 차이가 매우 큼

- **소모가 아닌 품질 저하**
  - H/W : 오래 사용하면 부품이 닳고 , 고장 발생 빈도 높고 , 기능도 떨어짐
  - S/W : 오래 사용해도 닳지 않고 , 고장 발생 빈도 낮고 , 기능도 동일 함

5. H/W 실패 곡선 (욕조 곡선) 의 특징

p. 21 bathtub curve ↓ SW 특징 : 사용자의 요구 발생 → 변경사항 발생 → 변경사항 적용

초기 실패율 높음

↓

오류 해결

↓

오랜 기간 동안 사용

↓

주변 환경 문제 발생
(먼지 , 진동)

↓

다시 실패율 증가

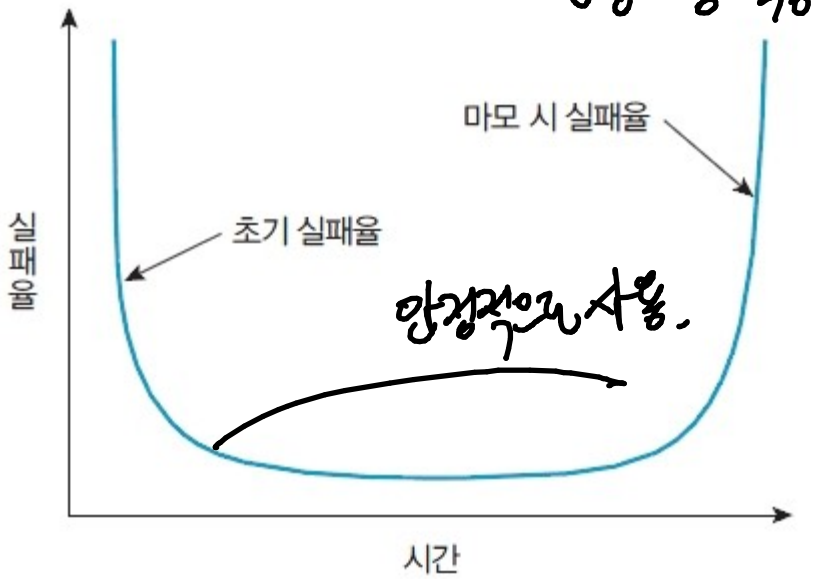


그림 1-2 하드웨어의 실패 곡선 //

6. 이상적인 소프트웨어 실패 곡선

특징 : 이상적인 상황

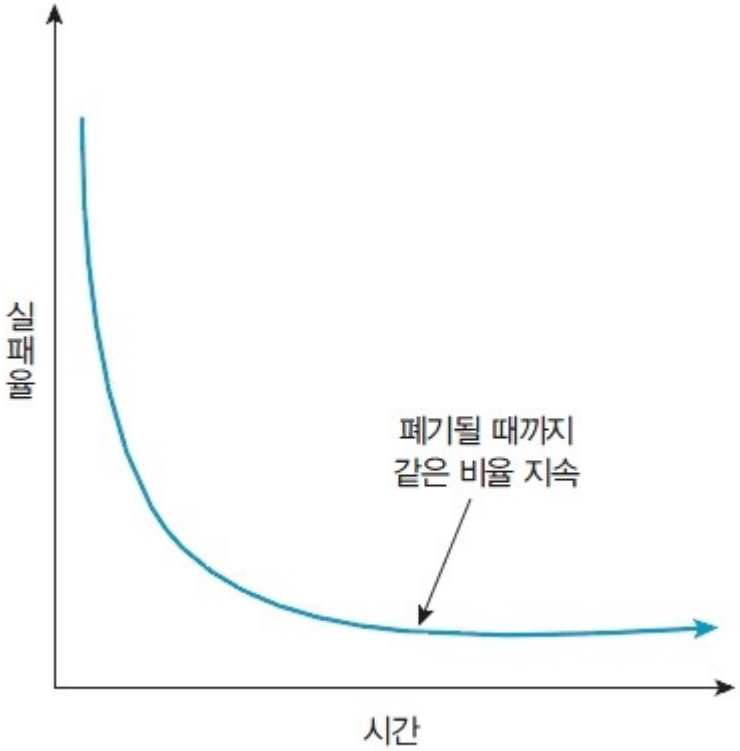
- 개발 완료 후 변경 사항 없어야 함
- 개발 완료 후 환경 변화 없어야 함

발견되지 않은 오류로 초기 실패율 높음

오류발견

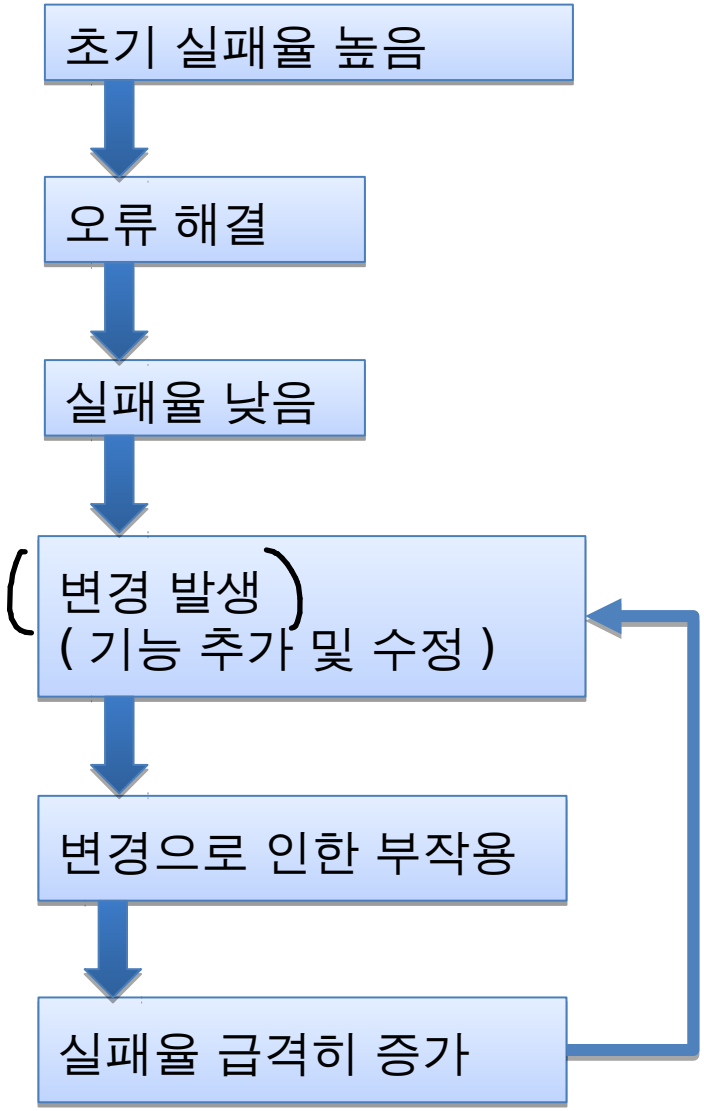
오류 해결

오랜 기간 동안 사용

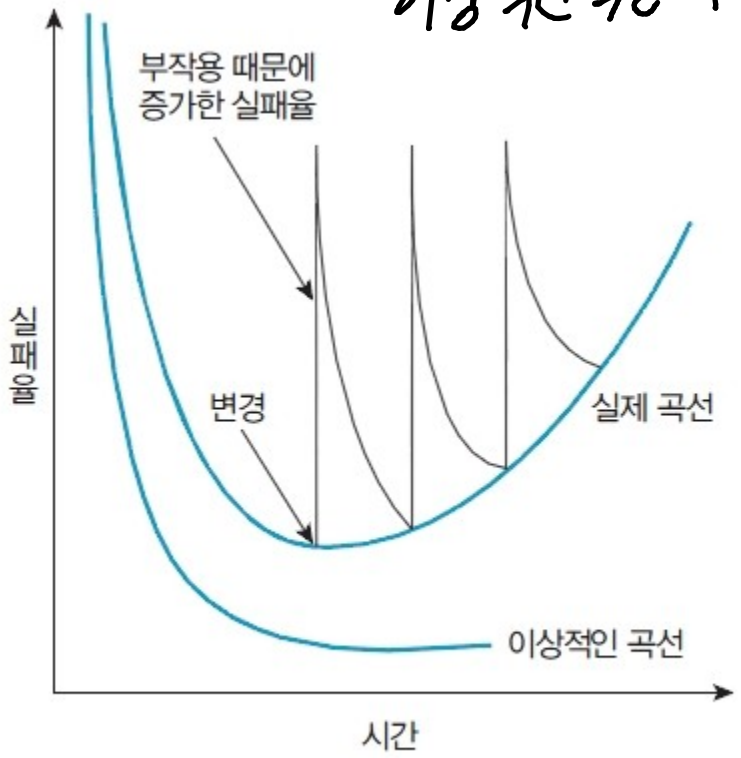


(a) 이상적인 소프트웨어의 실패 곡선

7. 실제 소프트웨어 실패 곡선



(변경사항이 발생하기 때문에 이상적인 곡선과 다름)



(b) 실제 소프트웨어의 실패 곡선

8. 소프트웨어의 당면 과제 (1)

1) 소프트웨어 개발의 느린 발전 속도

발전속도
H/W > S/W

- H/W 의 발전 : PC 및 스마트폰의 발전 속도 (크기 , 속도 , 성능)
- S/W 발전 속도 : DOS ~ Windows 10

2) 새로운 소프트웨어에 대한 사용자 요구의 증가

- S/W 의 발전 속도가 미처 따라가지 못함
- H/W 와 S/W 의 개발 방법의 근본적인 차이 때문
  - H/W: 검증 받은 부품을 조립하는 형태의 생산
  - S/W: 처음부터 만들어가는 개발 형태
- (해결 방안) CBD 개발 방법론

component based development . 부품기반개발

8. 소프트웨어의 당면 과제 (2)

③ 관리 기술의 부분적 활용

- 기계 : 닦고 , 조이고 , 기름치고 => 수명 연장

- S/W 개발에도 관리가 필요 (비용 · 일정)
  - 비용 관리
  - 일정 관리
  - 개발자 관리

  → PMBOK 를 활용한 적극적인 프로젝트 관리 필요

9. 소프트웨어 개발의 어려움 (1)

p.23.

개집 짓기

- 필요 도구 : 망치 , 톱 , 줄자 등
- 설계 도면 필요 없음 , 머릿속 구상만으로도 충분
- 혼자 가능 , 만드는 과정 단순

그림 1-4 개집 짓기

9. 소프트웨어 개발의 어려움 (2)

단독주택 짓기

- 필요 도구 : 레미콘과 같은 장비 , 시멘트 등의 수 많은 자재
- 설계 도면 , 건축 설계사 필요
- 많은 사람 참여 , 만드는 공정 과정 필요

그림 1-5 단독주택 짓기

9. 소프트웨어 개발의 어려움 (3)

대형 빌딩 짓기

- 필요 도구 : 레미콘뿐만 아니라 크레인과 같은 대형 장비
- 설계 도면 , 건축 설계사뿐만 아니라 내진 설계 필요
- 많은 사람이 참여할 뿐만 아니라 통제와 조정할 수 있는 조직 (부서) 이 필요
- 하중 문제 등 고려 사항이 많음

그림 1-6 대형 빌딩 짓기

9. 소프트웨어 개발의 어려움 (4)

대형 민정 잘기억 바람.

개발 과정이 복잡하다

무엇이든지 복잡하면 문제가 많이 발생할 수 있는데 소프트웨어 개발도 예외가 아니다. 그래서 소프트웨어 공학에서는 개발의 복잡함을 줄이기 위한 방법과 기술을 제시한다.

참여 인력이 많다

인력이 많으면 의사소통 경로가 많아져 의사 결정 과정도 복잡할 것이다. 또한 협력도 쉽지 않다. 그리고 중간에 이직하는 사람, 새로 투입되는 사람 등 변화도 많이 발생한다. 그래서 소프트웨어 공학에서는 개발에 참여하는 팀을 구성하고 관리하는 효율적인 방법을 제시한다.

개발 기간이 길다

개발 기간이 길면 프로젝트 진행 상황을 파악하기 쉽지 않고 개발 비용 산정도 어렵다. 그래서 소프트웨어 공학에서는 프로젝트를 효율적으로 관리하기 위한 프로젝트관리지식체계PMBOK를 소개한다.

그림 1-7 대규모 소프트웨어 개발의 어려움과 소프트웨어 공학

Section 02
공학과 소프트웨어 공학의 이해

1. 공학

- **공학이란? 공학은 '기술적 문제'를 대상으로 하는 학문으로 '문제를 발견하고 이에 대한 기술적 해결책을 제시하는 학문'**

공학의 사용 예

- 전기공학과, 건축공학과, 토목공학과 등의 대학교에서 학과 명으로 사용

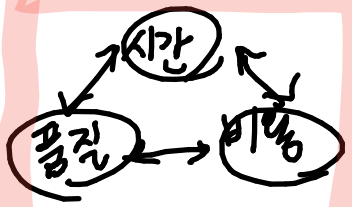
공학의 특성

- 제약 사항: 정해진 기간, 주어진 비용

  → 과학적 지식을 활용하여 문제를 해결하는데 한정된 기간과 비용의 제약을 받음

소프트웨어 공학

- 소프트웨어 + 공학
- 취지: '소프트웨어 개발 과정에 공학적인 원리를 적용하여 소프트웨어를 개발'
- 목적:
  - S/W 개발의 어려움 해결
  - 효율적 개발을 통한 생산성 향상
  - 고품질 소프트웨어 제품



2. 소프트웨어 개발 과정

- **소프트웨어 개발 생명주기 (SDLC Software Development Life Cycle)**
  - 계획 단계에서 유지보수 단계에 이르기까지 일어나는 일련의 과정

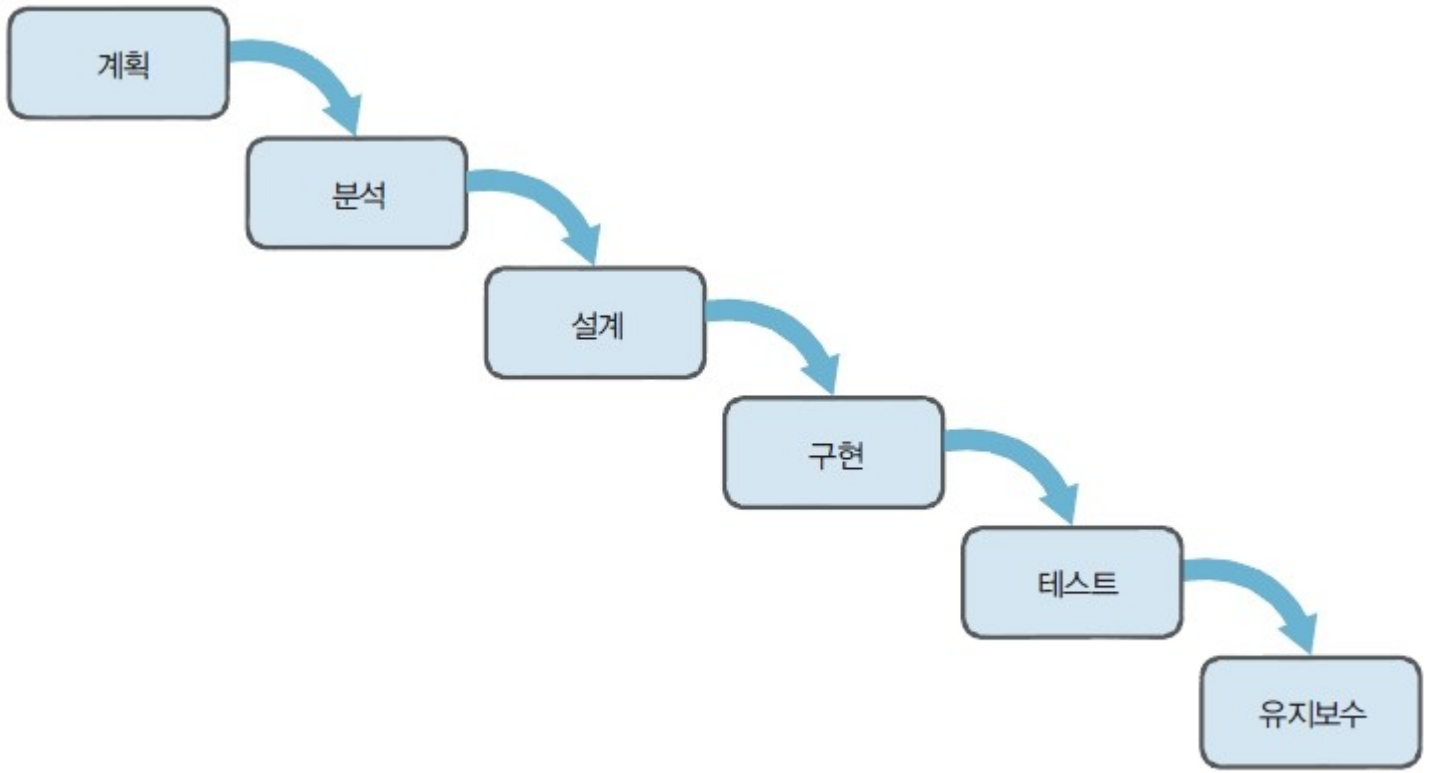


그림 1-8 소프트웨어 개발 생명주기 SDLC: Software Development Life Cycle

3. 소프트웨어 공학

정의

품질 좋은 소프트웨어를 경제적으로 개발하기 위해
계획을 세우고, 개발하며,
유지 및 관리하는 전 과정에서
공학, 과학 및 수학적 원리와 방법을 적용하여
필요한 이론과 기술 및 도구들에 관해
연구하는 학문

목표

- 개발 과정에서의 생산성 향상
- 고품질의 소프트웨어 생산 → 사용자 만족

Section 03
소프트웨어 개발 단계의 소개

1. 소프트웨어 개발 단계 p.28

SPLC // 폭포수형.

① ■ 소프트웨어 개발 프로세스

- 1 단계 : 계획
- 2 단계 : 요구분석
- 3 단계 : 설계
- 4 단계 : 구현
- 5 단계 : 테스트
- 6 단계 : 유지보수

영-유-청소-청년-중장년-노년.-탐욕-사망.

~~■ 소프트웨어 개발 프로세스~~

② ■ 품질 관리

③ ■ 프로젝트 관리

2. 계획 / 요구분석 단계

- **1 단계 : 계획 (3 장에서 구체적으로 설명)**
  - 개발 비용 산정 : COCOMO 모델 , 기능점수 (FP) 모델 사용
  - 일정 계획 : 작업분할구조도 WBS, CPM 사용
  - 위험 관리
- **2 단계 : 요구분석 (4 장에서 구체적으로 설명)**
  - 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 : UML 의 유스케이스 다이어그램
  - 최종 산출물 : 요구 분석 명세서

3. 설계 / 구현 단계

(개념적인 단계)

- **3 단계 : 설계 (5-6 장에서 구체적으로 설명)** 요구분석 → 설계
  - 설계 원리 : 분할과 정복 , 추상화 , 단계적 분해 , 모듈화 , 정보은닉
  - 소프트웨어 아키텍처 , 객체지향 설계
  - 아키텍처 스타일
  - GoF 의 디자인 패턴
  - 모듈 평가 기준 : 응집도와 결합도

- **4 단계 : 구현 (7 장에서 구체적으로 설명)**
  - 간략한 프로그래밍 언어의 역사
  - 표준 코딩 규칙

4. 테스트 / 유지보수 단계

- **5 단계 : 테스트 (8 장에서 구체적으로 설명)**
  - 테스트의 절차
  - 개발자 또는 사용자 시각에 따른 분류
  - 사용되는 목적에 따른 분류
  - 품질 특성에 따른 분류
  - 소프트웨어 개발 단계에 따른 분류

- **6 단계 : 유지보수 (10 장에서 구체적으로 설명)**
  - 수정 유지보수 (에러 발생시)
  - 적응 유지보수 (기종변경시) maintenace (유지보수)
  - 기능보강 유지보수 (업그레이드)
  - 예방 유지보수

5. 소프트웨어 개발 프로세스

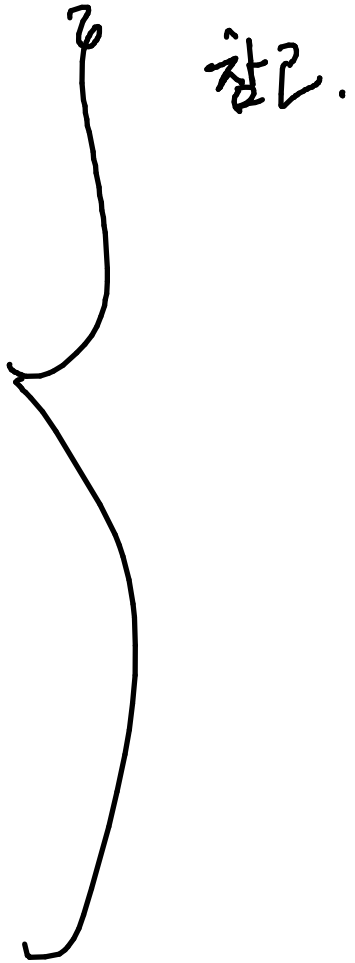
SDLC의 모형 .

- **소프트웨어 개발 프로세스 (2 장에서 구체적으로 설명)**
  - 주먹구구식 개발 모델
  - 선형순차적 모델 (폭포수 모델)
  - V 모델
  - 진화적 프로세스 모델 (프로토타입 모델)
  - 나선형 모델
  - 단계적 개발 모델
  - 통합 프로세스 모델 (UP)
  - 애자일 프로세스 모델

+ 3개 → 11개의 모형 암기

6. 품질 관리

- **품질 관리 (9 장에서 구체적으로 설명)**
  - 제품 품질 특성 평가
    - ISO/IEC 9126 모델
    - ISO/IEC 9126 모델
    - ISO/IEC 9126 모델
    - ISO/IEC 9126 모델



- **프로세스 품질 특성 평가**
  - ISO/IEC 9000 모델
  - ISO/IEC 12207 모델
  - CMMI 모델
  - SPICE 모델

7. 프로젝트 관리 (10장)

- **프로젝트 관리 (10 장에서 구체적으로 설명)**
  - 형상 관리

- **PMBOK 프로젝트관리지식체계의 9 가지 관점**
  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

Thank You