2022 개인별포트폴리오

시스템분석설계
개인별 포트폴리오

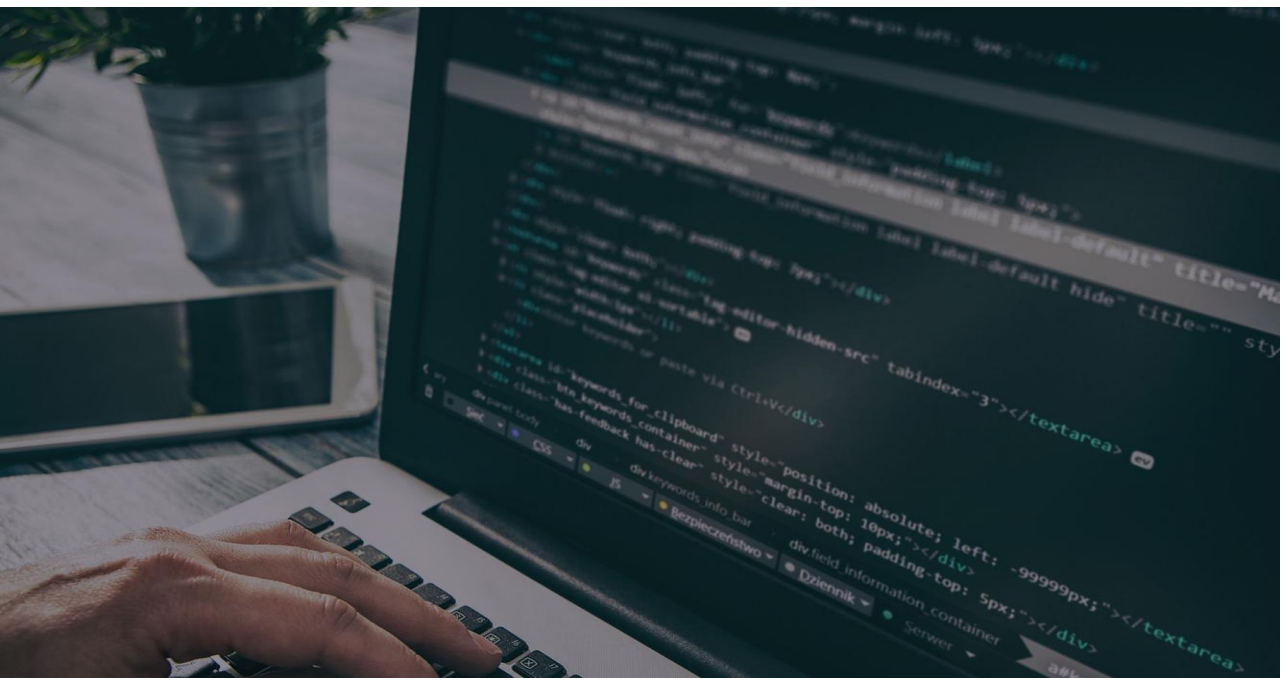
컴퓨터정보공학과 2-B
20203132 이지현

CONTENTS

I. 소프트웨어 공학

소프트웨어 공학
- 소프트웨어 공학의 이해

프로그램이란?



PROGRAM 프로그램, 프로세스

프로그램이란, 적혀진 원시코드와 명령어대로 작동하는 실행파일이다. 크게는 시스템 프로그램과 응용 프로그램(exe)으로 나뉘어져있으며, 이때 프로세스는 프로그램을 실행 시키는 실행 주체(인스턴스)라 표현하는데 즉, 프로그램의 하나의 인스턴스는 프로세스이다.

시스템 프로그램이란, 컴퓨터 시스템과 하드웨어들을 스스로 제어 및 관리하는 프로그램이다.
Ex) 윈도우, 리눅스, 장치 드라이버, 컴파일러, 링커

응용 프로그램이란, 사용자가 원하는 기능을 제공하는 프로그램으로 실행하는 동안 지속적으로 컴퓨터의 성능을 소비한다.
Ex) 워드, 엑셀, 포토샵, 게임, 브라우저

소프트웨어 공학
- 소프트웨어 공학의 이해

소프트웨어란?

SOFTWARE 소프트웨어의 이해

소프트웨어란, **프로그램으로 구성**되어 있다. 프로그래밍 언어로 작성된 코드로 만들어지며 **특정한 목적성을 띄고 개발이 진행**된다.
각 단계마다 문서가 생산되며 **자료구조, DB구조, 테스트 결과 등을 산출물로 확인**할 수 있다.

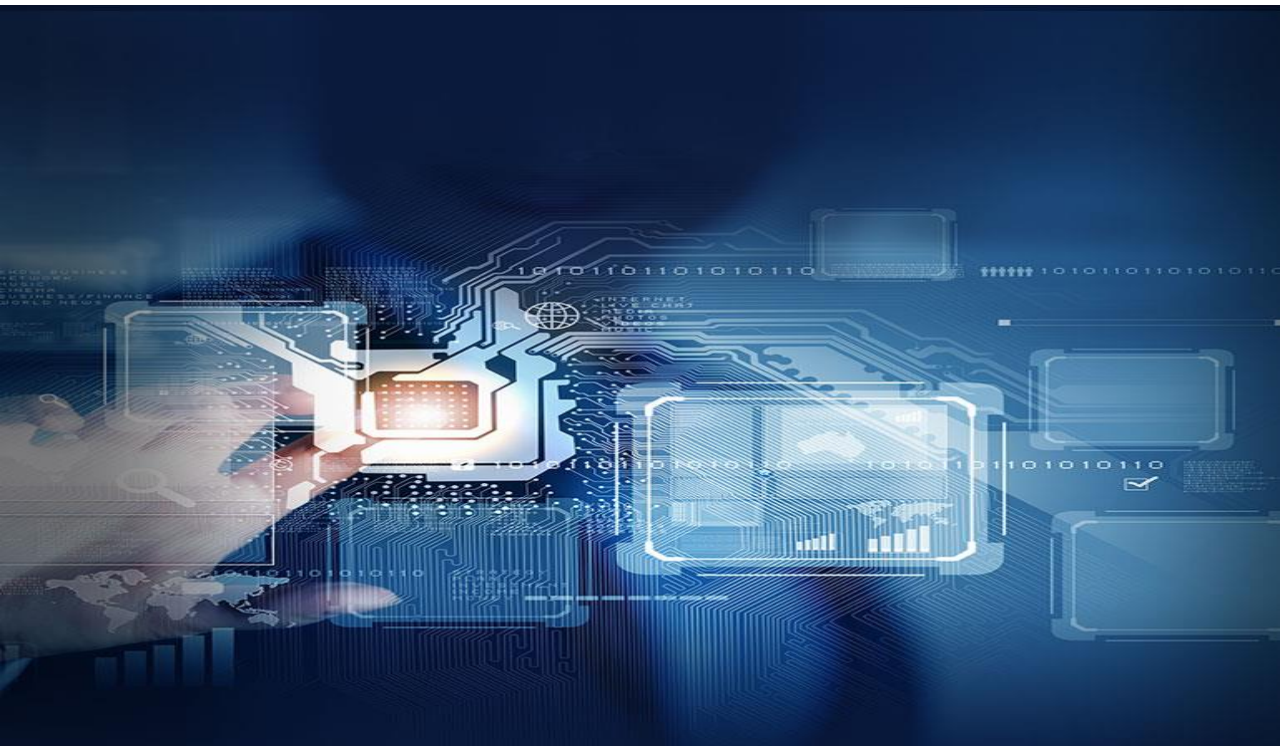
개발 시 소비자들의 요구가 반영되고 **한번 개발한 뒤 적은 비용으로 수정 및 조작이 가능**하다.

소프트웨어는 앞서 말한 **시스템 프로그램(소프트웨어)과 응용 프로그램(소프트웨어), 크게 두가지로 분류**되는데 하드웨어와 다르게 **시대와 환경에 변화에 민감**하다.

소프트웨어를 개발할 때, **컴파일러와 인터프리터를 통해 기계어로 변환되어 사용된다.** 고급 프로그래밍 언어와 다른 **저급 어셈블리어로도 개발이 가능**한데 이때는 **어셈블러를 사용**한다.

소프트웨어 공학
- 소프트웨어 공학의 이해

소프트웨어 공학이란?



SOFTWARE ENGINEERING 소프트웨어 공학의 이해

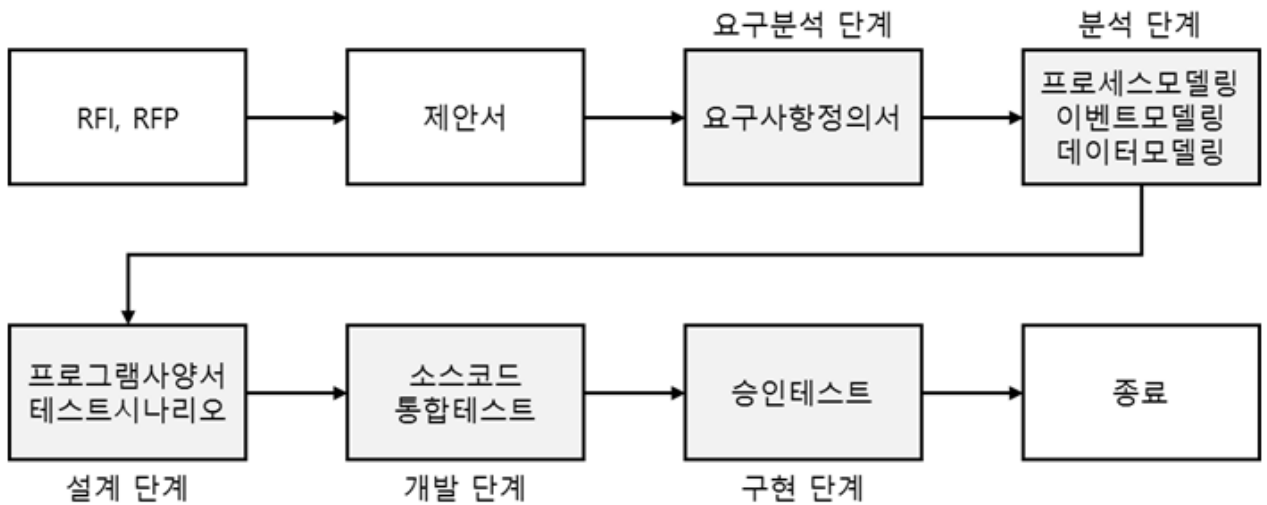
소프트웨어 공학의 정의
인류의 이익을 위해 과학적 원리, 지식, 도구 등을 활용하여 **새로운 제품이나 도구 등을 만드는 것**이다. **계획, 요구사항 분석, 설계와 구현, 출시 및 유지보수**까지의 과정을 거치며 이론과 개념, 주기의 전반을 다루는 학문이다.

소프트웨어 공학의 목표로는 **복잡도 저하, 비용 최소화, 개발기간 단축, 대규모 프로젝트 관리, 고품질 소프트웨어, 효율성**이 궁극적인 목표이다.

소프트웨어 개발의 단계별 목표
분석 : 무엇을 만들 것인가? → 요구 사항 정의서, 요구 분석 명세서
설계 : 어떻게 구축할 것인가? → 설계 사양서
구현 : 코딩과 단위 시험 → 모듈별 코드
테스팅 : 요구에 맞게 실행되나? → 테스팅 결과 보고서

소프트웨어 공학
- 소프트웨어 공학의 이해

소프트웨어 개발



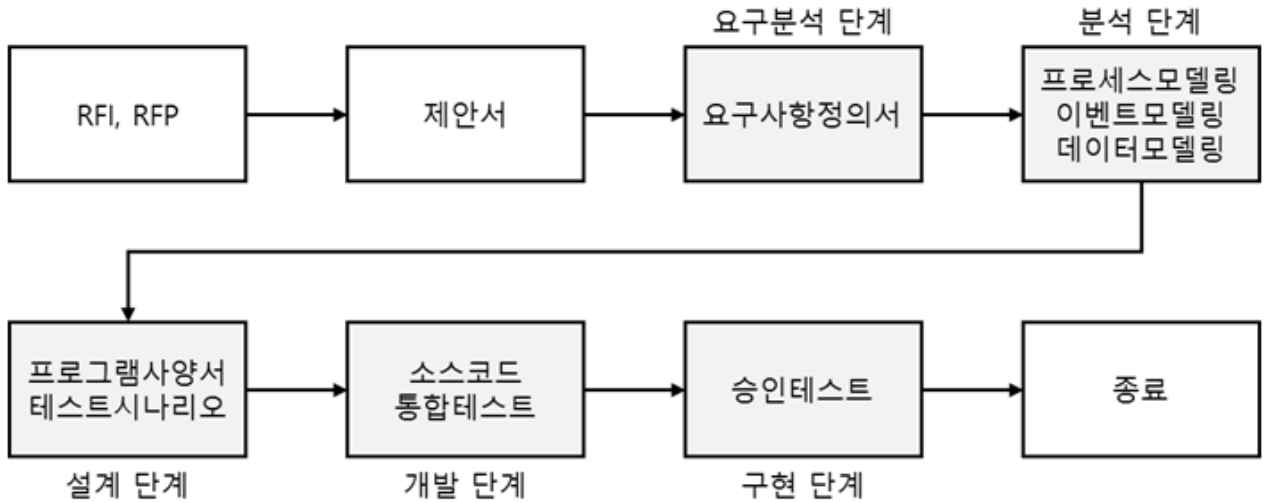
SOFTWARE DEVELOPMENT 소프트웨어 개발 단계

I. 요구사항 분석
문제 분석 단계, 개발하는 소프트웨어의 기능과 **제약조건, 목표등 사용자와 함께 설정하는 단계**이다. 어떤 소프트웨어를 개발할지 개발자와 사용자가 **소프트웨어의 성격을 정확하게 이해**해야하며, 이를 토대로 **필요자원 및 예산 예측 후 요구 사항 정의서, 요구 분석 명세서**를 작성한다.

II. 설계단계
앞선 단계에서 **논의한 기능들을 실제로 수행하기 위한 방법을 결정하고 설계 사양서를 작성**한다. 크게 **시스템, 프로그램, UI**로 나뉜다.

시스템 구조설계 : 내부 프로그램 및 모듈 간의 관계 구조 설계
프로그램 설계 : 프로그램 내 각 모듈 간 처리 절차, 알고리즘 설계
UI 설계 : 사용자가 시스템을 사용하기 위한 인터페이스 설계

소프트웨어 개발



SOFTWARE DEVELOPMENT 소프트웨어 개발 단계

III. 구현 단계

설계 단계에서 설정한 기능들의 구현 및 문제 해결 방법을 프로그래밍 언어를 사용하여 실제 프로그램과 모듈별 코드를 작성한다.

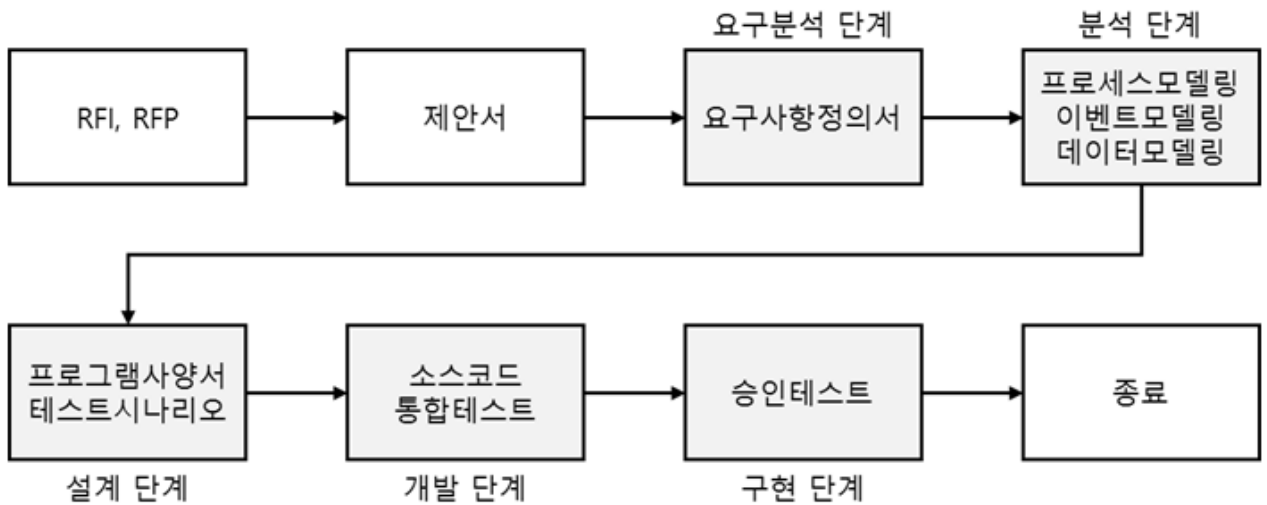
이때 구현 기법은 구조화 프로그래밍과 모듈러 프로그래밍 두개로 나뉜다. 구조화 프로그래밍은 정확성 검증과 테스트 및 유지보수가 쉬운 장점이 있고, 모듈러 프로그래밍은 모듈의 재사용이 가능하다는 장점이 있다.

구조화 프로그래밍 : 조건문, 반복문으로 프로그램 작성 후 순차구조, 선택구조, 반복구조로 표현

모듈러 프로그래밍 : 프로그램을 여러 개의 작은 모듈로 나누어 계층을 관리

소프트웨어 공학
- 소프트웨어 공학의 이해

소프트웨어 개발



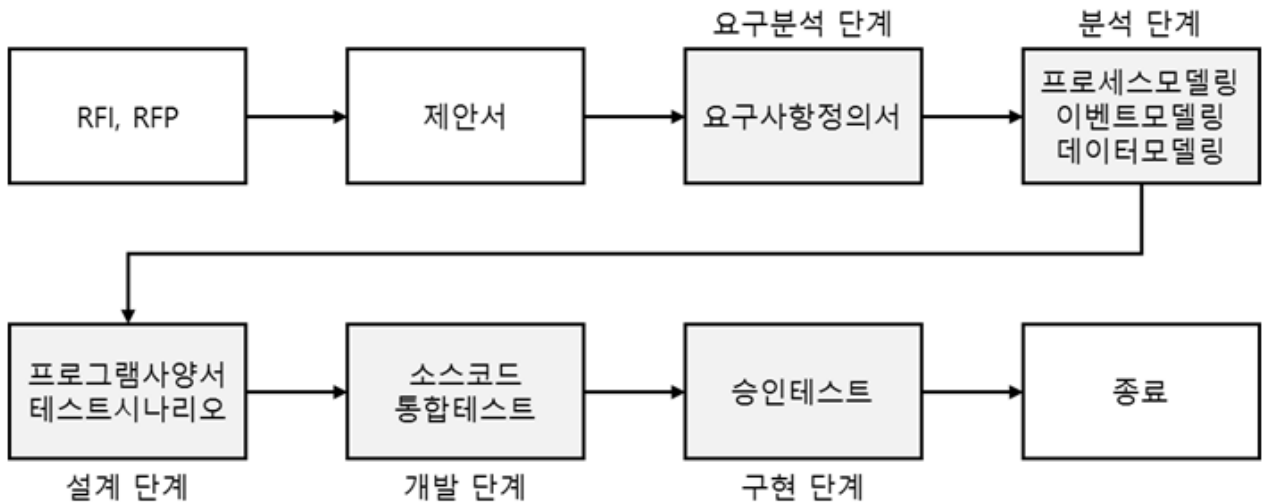
SOFTWARE DEVELOPMENT 소프트웨어 개발 단계

IV. 테스트 단계

테스트 단계에서는 **개발한 소프트웨어가 사용자의 요구사항을 만족하는지, 실행 결과가 예상 결과와 맞는지 검사하고 평가**하는 과정이다. 미처 발견하지 못한 오류를 테스트 단계에서 발견할 확률이 높다.테스트 단계로는 **단위 테스트, 통합테스트, 인수테스트**가 있다.

테스트 단계의 순서
단위 테스트 → 통합 테스트 → 인수 테스트

소프트웨어 개발



SOFTWARE DEVELOPMENT 소프트웨어 개발 단계

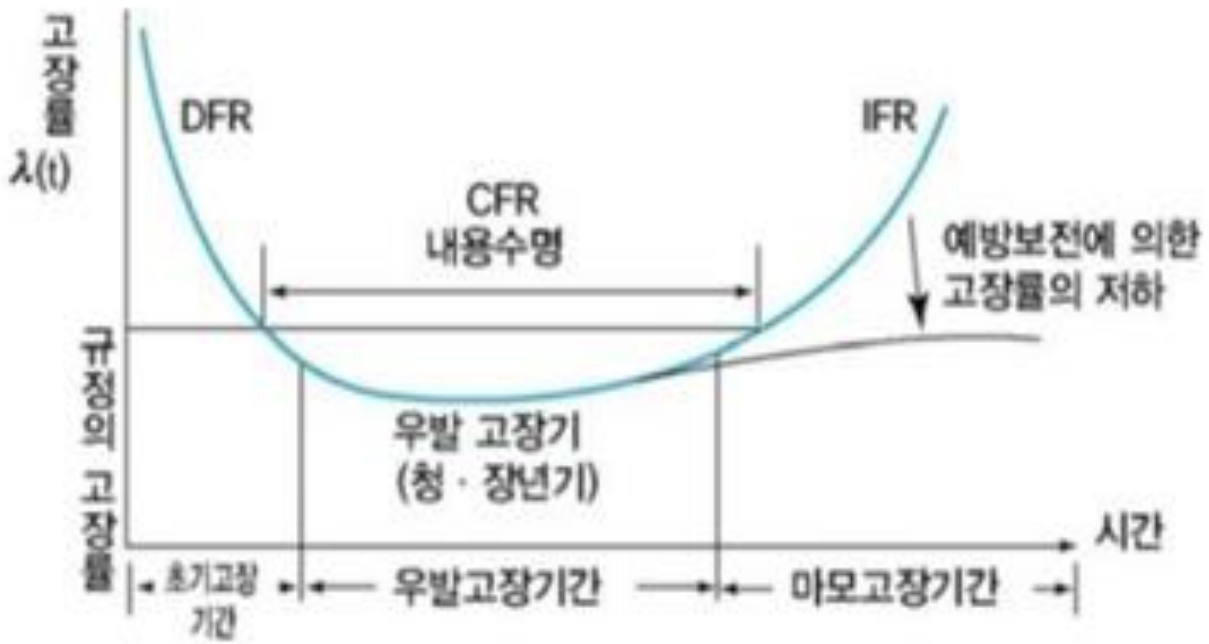
V. 유지보수 단계

유지보수 단계는 **소프트웨어가 출시된 후 일어나는 모든 활동을 지칭**한다. 커스터마이징, 구현, 테스트 등 모두 이 단계에 포함되며 **소프트웨어 생명주기에서 가장 긴 기간을 차지**한다.**유지보수 유형에는 수정형, 적응형, 완전형, 예방형**이 있다.

수정형 : 사용 중 발견한 프로그램 오류 수정 작업 진행
적응형 : 소프트웨어의 환경적 변화를 위한 재조정
완전형 : 소프트웨어의 성능 향상 및 개선 작업
예방형 : 앞으로 변경 사항을 수용하기 위한 대비

소프트웨어 공학
- 소프트웨어 공학의 이해

소프트웨어의 실패 곡선



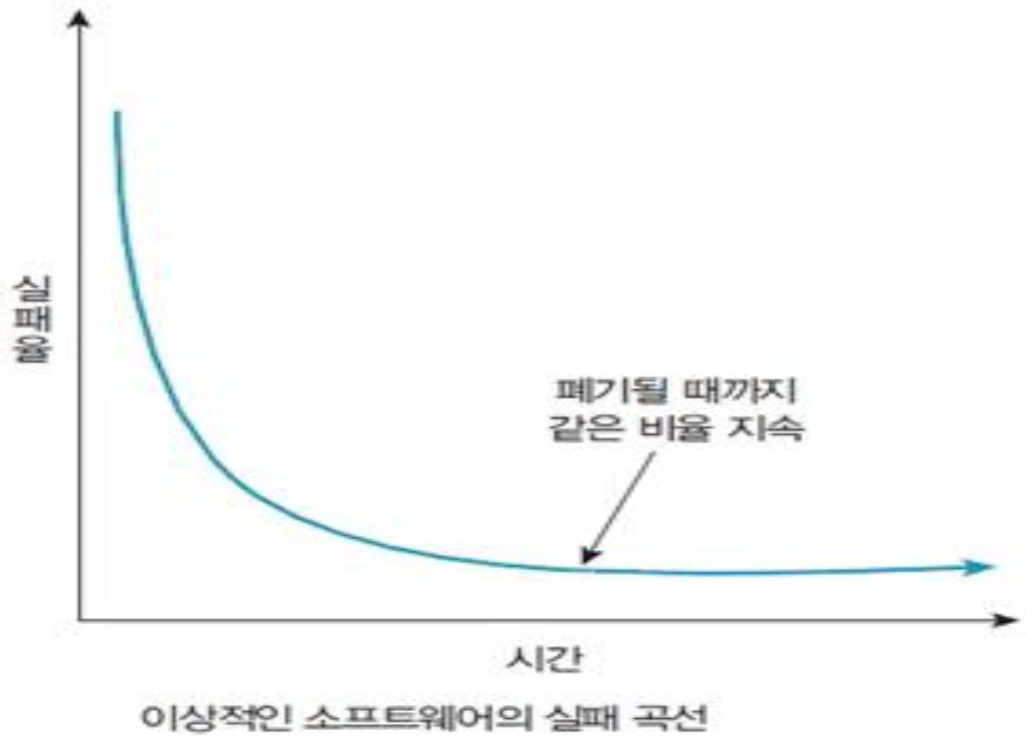
BATHTUB CURVE H/W 욕조 곡선

H/W는 오래 사용 시 부품 닳게 되어 고장 발생 빈도가 높아지며 그에 따라 기능도 떨어진다.

H/W 실패 곡선(욕조 곡선)의 특징
초기 실패율 높음 → 오류 해결 → 오랜 기간 동안 사용 → 주변 환경 문제 발생 → 다시 실패율 증가

소프트웨어 공학
- 소프트웨어 공학의 이해

소프트웨어 실패 곡선



이상적인 소프트웨어의 실패 곡선

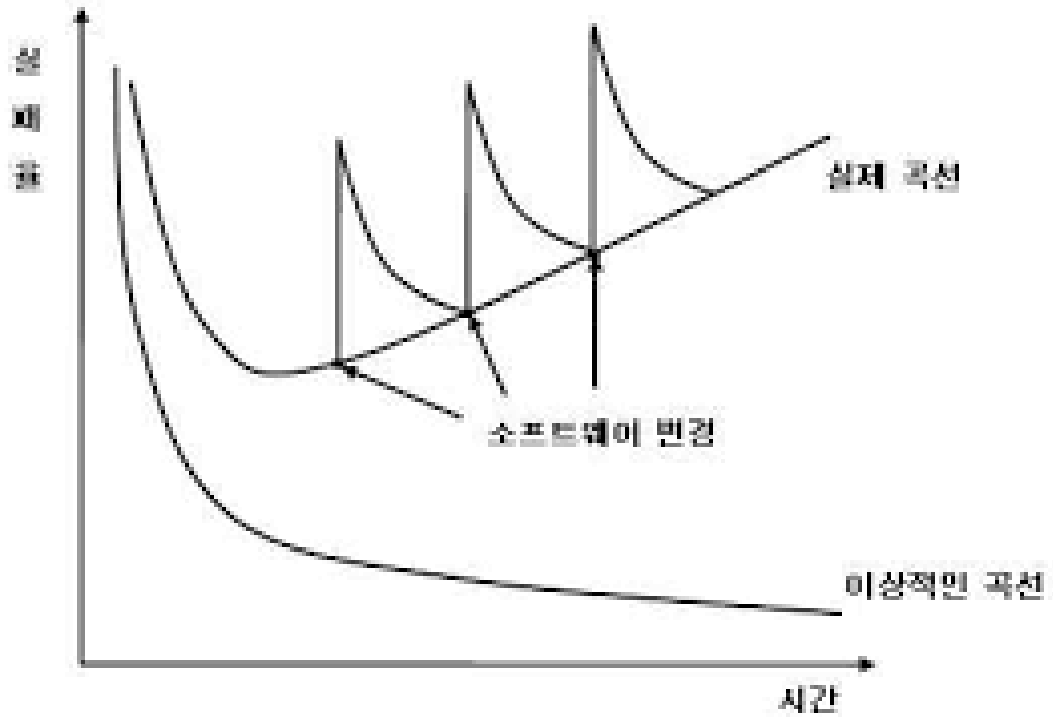
FAILURE CURVE 이상적인 실패 곡선

이상적인 실패 곡선으로는 **개발 완료 후 환경변화와 변경사항이 없어야한다.**

발견되지 않은 오류로 인한 초기 실패율 높음 → 오류 해결 → 오랜 기간 동안 사용

소프트웨어 공학
- 소프트웨어 공학의 이해

소프트웨어 실패 곡선



FAILURE CURVE 실제 소프트웨어 실패 곡선

실제 소프트웨어 실패 곡선은 **이상적인 실패 단계와 비슷하지만 수정 등 변경의 문제로 실패율이 급격하게 증가한다.**

초기 실패율 높음 → 오류 해결 → 실패율 낮음 → 기능 추가 및 수정 변경 발생 → 변경으로 인한 부작용 → 실패율 급격히 증가 → 반복

II. 깃과 깃허브

깃과 깃허브
– 깃허브의 이해

깃과 깃허브란?

GITHUB 깃과 깃허브

깃이란, 로컬에서 관리가능한 **분산 버전 제어 시스템**이며 **변경 사항을 추적하고 여러 사용자들 간에 파일에 대한 작업을 조율**할 수 있다.

깃허브란. 분산 버전 관리 툴인 **깃을 지원하는 웹 호스팅 서비스**이다.

깃과 깃허브
- 깃허브의 이해

깃허브 주요용어

GITHUB 깃허브 주요용어

- Fork : 다른 깃허브 저장소(오픈 소스 프로젝트)를 복사하는 작업

- Upstream : 타인의(오픈소스 프로젝트) 깃허브 저장소

- Origin : 나의 깃허브 저장소
  (GITHUB 안 내 계정에 있는 fork한 저장소)
- 로컬 저장소 : 내가 자신의 컴퓨터에 복사한 지역 저장소

깃허브 주요용어

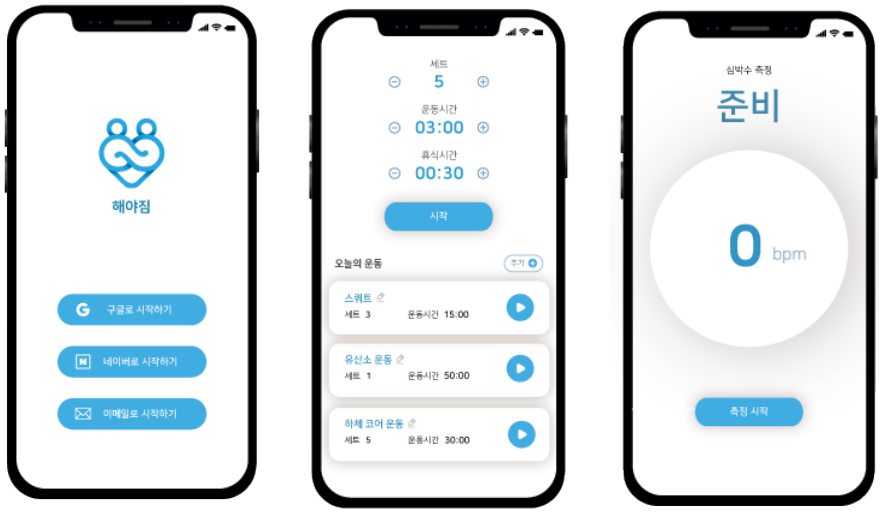
GITHUB 깃허브 주요용어

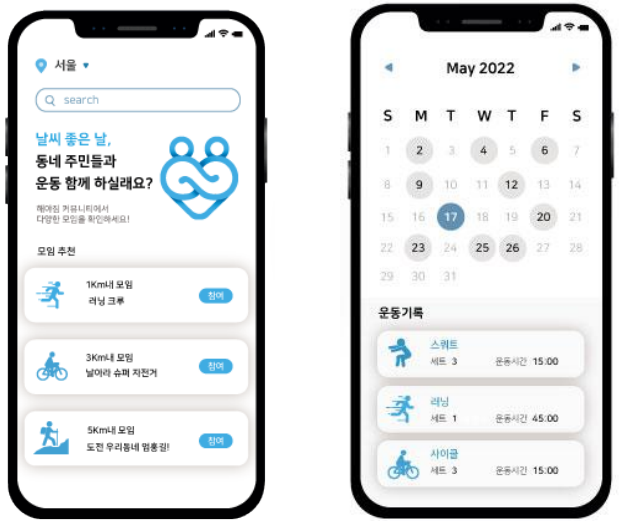
- Repository : 히스토리, 태그 , 소스의 가지치기 등 작업자가 변경한 모든 히스토리를 확인가능
- Merge : 다른 저장소의 내용을 현재 저장소로 가져와 합치는 작업
- Push : 로컬 컴퓨터에서 서버로 변경사항을 적용
- Pull : 서버 저장소로부터 최신 버전을 가져옴
- Clone : 나의 저장소를 로컬 저장소에서 가져올 때 사용

III. 안드로이드 스튜디오

안드로이드 스튜디오
- 기능 및 코드 리뷰

주요기능 및 코드





ANDROID STUDIO 해야짐 UI

운동 어플리케이션 해야짐 UI 구조도

로그인 기능, 타이머 기능, 운동달력, 커뮤니티, 심박수 기능

안드로이드 스튜디오
- 기능 및 코드 리뷰

코드

ANDROID STUDIO 해야짐 메인 페이지 구현

로그인 기능 구현
해야짐 앱 등록 및 설정 → 기능 구현 → 릴리즈 키 등록

안드로이드 스튜디오
- 기능 및 코드 리뷰

코드

릴리즈 인증서 지문

```
keytool -list -v \
-alias <your-key-name> -keystore <path-to-production-keystore>
```

디버그 인증서 지문

```
keytool -list -v \
-alias androiddebugkey -keystore %USERPROFILE%\.android\debug.keystore
```

Gradle 서명 보고서 사용

```
$ ./gradlew signingReport
```

Keytool aab에서 사용

```
keytool -printcert -jarfile app.aab
```

Keytool apk에서 사용

```
keytool -printcert -jarfile app.apk
```

ANDROID STUDIO 해야짐 메인 페이지 구현

해야짐 앱 등록 및 설정
SHA-1(릴리스 및 디버그 인증서 지문)을 가져옴

SHA-1 가져오는 방법
- Keytool 사용
- Gradle 서명 보고서 사용

두가지 방법 중 하나 선택하여 코드 작성 후 Google Console API에 프로젝트 등록과 SHA-1 지문 등록을 하면 구글 로그인이 가능한 Oauth Client 구성

Oauth Client : 인증을 위한 개방형 프로토콜

Thank you