2023-05-21

소프트웨어 개발 방법론

: 우리는 팀 프로젝트를 어떻게 진행해야 할까요

김도경

소프트웨어 개발 방법론이란?

소프트웨어를 생산하는 데에 필요한 프로그래밍 개발 과정들을 정리하고 표준화하여

프로그래머들이 프로그래밍 개발과정에서 각개인이 개발과정에서의 일관성을 유지하고

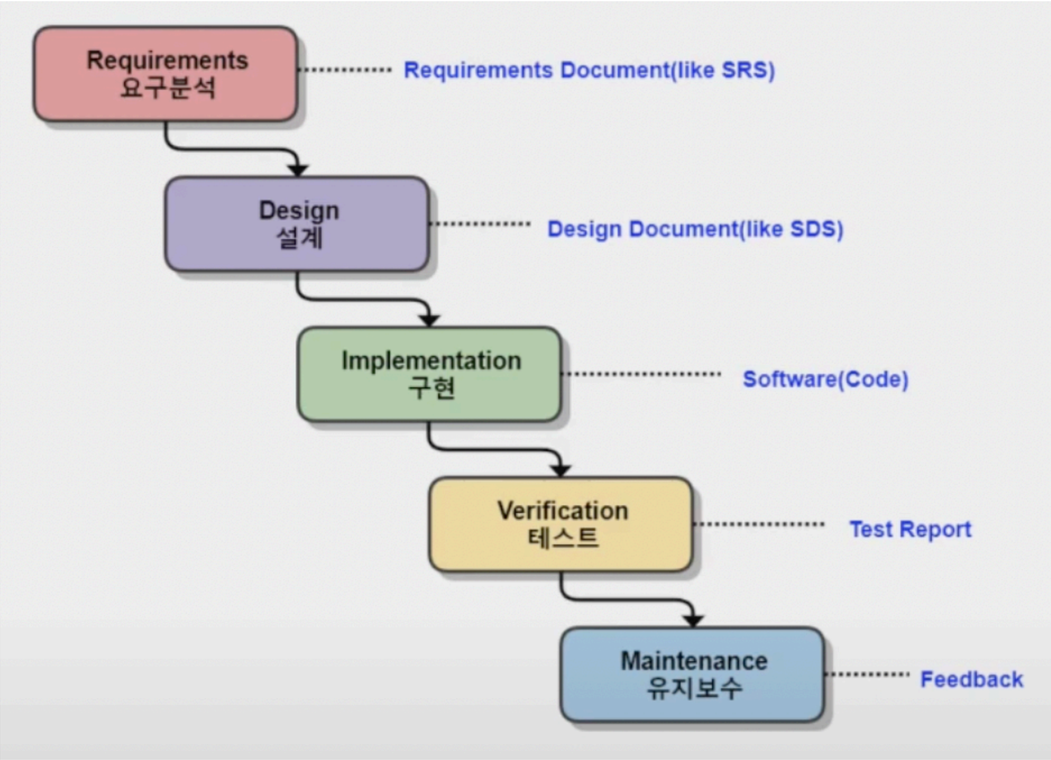
프로그래머들간의 효과적인 협업이 이루어질수 있도록 돕기 위한 방법론

- 구조적 방법론(Structured Development)
- 정보공학 방법론(Information Engineering Development)
- 객체지향 방법론(Object-Oriented Development)
- 컴포넌트 기반 방법론(CBD; Component Based Development)
- 애자일 방법론(Agile Development)
- 제품 계열 방법론(Product Line Development)

- 구조적 방법론(Structured Development)
- 정보공학 방법론(Information Engineering Development)
- 객체지향 방법론(Object-Oriented Development)
- 컴포넌트 기반 방법론(CBD; Component Based Development)
- 애자일 방법론(Agile Development)
- 제품 계열 방법론(Product Line Development)

구조적 방법론

: 폭포수 모델



전체 시스템을 기능에 따라 나누어 개발하고,
이를 통합하는 분할과 정복 접근 방식의 방법론

프로세스 중심의 하향식 방법론

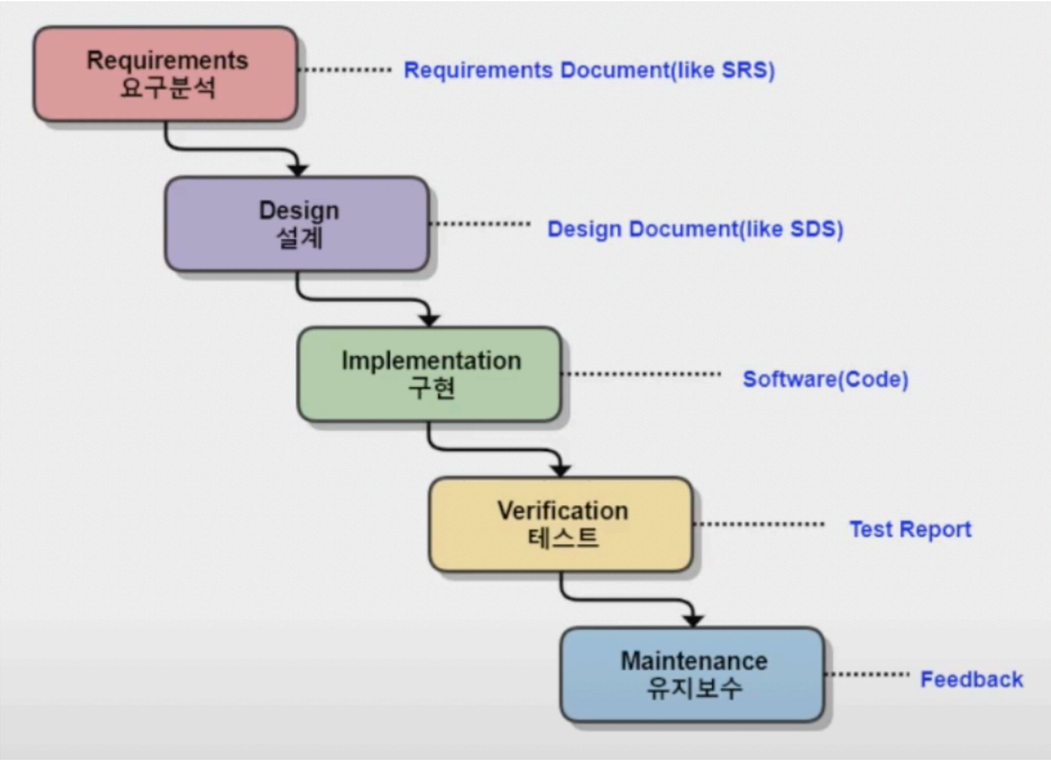
명확한 요구사항을 추출하여 설계에 반영 가능

대규모 프로젝트도 관리하기 쉬움

관리자와 개발자 모두에게 쉬운 개발 방법

구조적 방법론

: 폭포수 모델



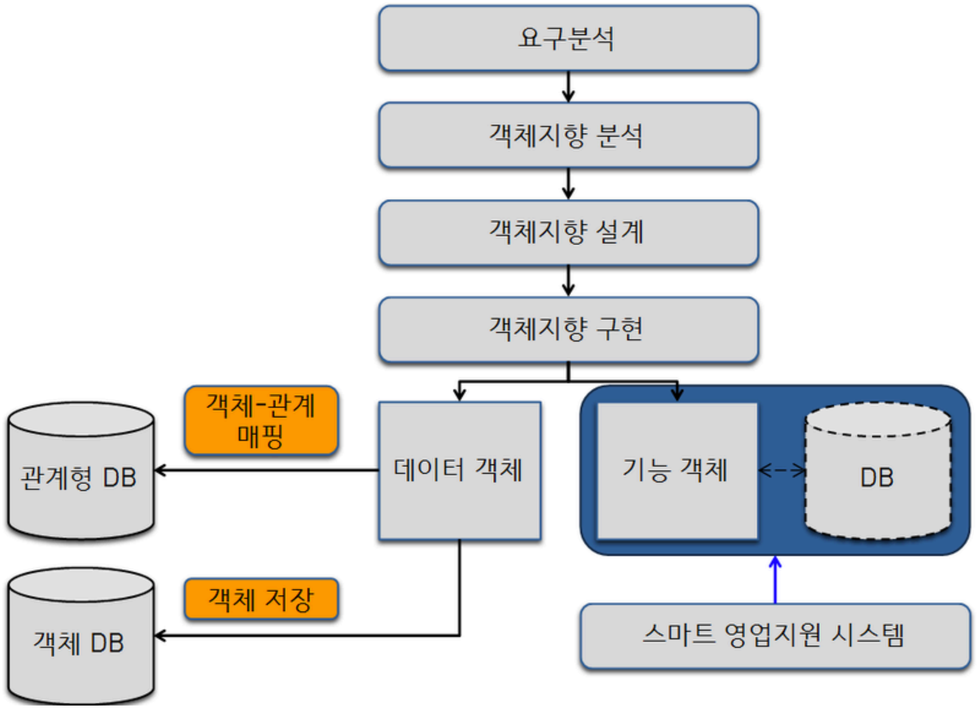
하지만,

요구 사항은 애초에 명확하지 않다.

요구 사항은 바뀔 수 있다.

나중 단계에서 앞 단계의 문제가 드러난다.

객체지향 방법론

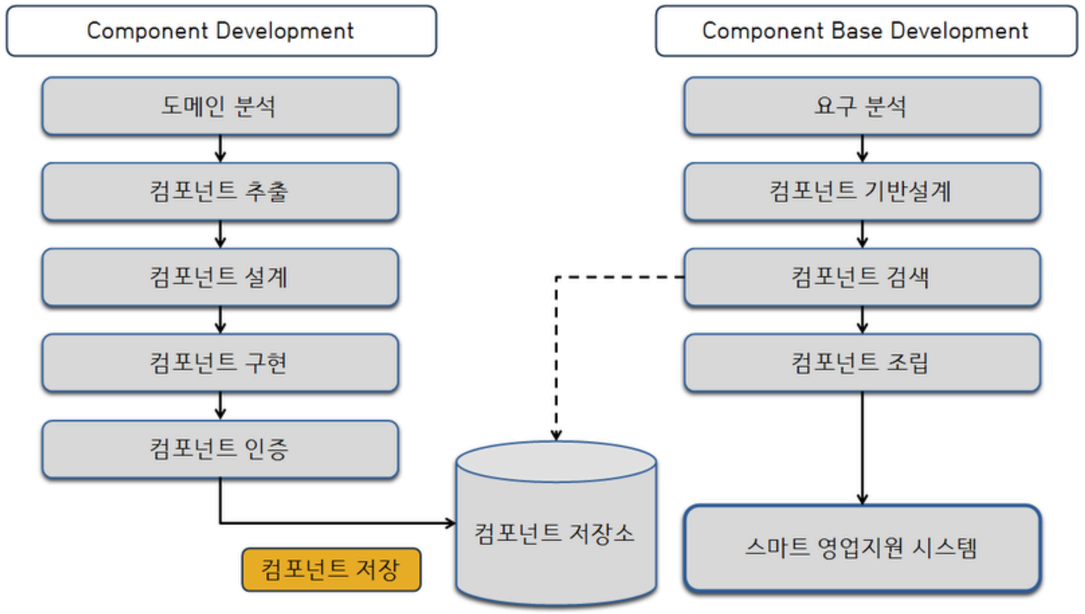


'객체'라는 기본 단위로 시스템을 분석 및 설계하는 방법론

복잡한 현실 세계를 사람이 이해하는 방식으로 시스템에 적용하는 방법론

객체, 클래스, 메시지를 사용

컴포넌트 기반 방법론(CBD)



소프트웨어를 구성하는 컴포넌트를 조립해서
하나의 새로운 응용 프로그램을 작성하는 방법론

문제를 조각으로 나누어 각각 컴포넌트를 생성한 후,
다시 조합하는 재사용성에 초점을 둔 방식

재사용 가능한 컴포넌트 개발 또는 상용 컴포넌트를
조합해 어플리케이션 개발

컴포넌트 : 인터페이스로 접근 가능하고 독립적인 기능을 수행하는 모듈로써 교체가 가능한 소프트웨어 부품

애자일 방법론

절차보다는 사람이 중심이 되어 변화에 유연하고 신속하게 적응하면서
효율적으로 시스템을 개발할 수 있는 신속 적응적 경량 개발 방법론

기존 방법론들이 절차를 중시한 나머지 변화에 빠른 대응을 할 수 없는 단점 개선을 위해 등장

개발 범위 안에 있는 요구사항 분석 후 우선순위 순으로 개발

애자일 방법론

: 애자일 소프트웨어 개발 선언

> 우리는 소프트웨어를 개발하고, 또 다른 사람의 개발을 도와주면서
> 소프트웨어 개발의 더 나은 방법들을 찾아가고 있다.
> 이 작업을 통해 우리는 다음을 가치 있게 여기게 되었다.
>
> - 공정과 도구보다 개인과 상호작용을
> - 포괄적인 문서보다 작동하는 소프트웨어를
> - 계약 협상보다 고객과의 협력을
> - 계획을 따르기보다 변화에 대응하기를
>
> 가치 있게 여긴다. 이 말은, 왼쪽에 있는 것들도 가치가 있지만,
> 우리는 오른쪽에 있는 것들에 더 높은 가치를 둔다는 것이다.[4]

애자일은 방법론이 아니라 정신(핵심 가치)

애자일 방법론

: 애자일 소프트웨어 개발 선언

우리는 소프트웨어를 개발하고, 또 다른 사람의 개발을 도와주면서
소프트웨어 개발의 더 나은 방법들을 찾아가고 있다.
이 작업을 통해 우리는 다음을 가치 있게 여기게 되었다.

- 공정과 도구보다 개인과 상호작용을
- 포괄적인 문서보다 작동하는 소프트웨어를
- 계약 협상보다 고객과의 협력을
- 계획을 따르기보다 변화에 대응하기를

가치 있게 여긴다. 이 말은, 왼쪽에 있는 것들도 가치가 있지만,
우리는 오른쪽에 있는 것들에 더 높은 가치를 둔다는 것이다.[4]

애자일은 방법론이 아니라 정신(핵심 가치)

애자일 방법론

: 익스트림 프로그래밍(XP)

의사소통 개선과 즉각적 피드백으로 소프트웨어 품질을 높이기 위한 방법론

- 용기(Courage): 용기를 가지고 자신감 있게 개발(코드를 작성하기 전에 테스트, 빠르게 피드백, 테스트에 부합하지 못하는 코드를 리팩토링할 수 있는 용기)
- 단순성(Simplicity): 필요한 것만 하고 그 이상의 것들은 하지 않음
- 의사소통(Communication): 개발자, 관리자, 고객 간의 원활한 소통
- 피드백(Feedback): 의사소통에 대한 빠른 피드백
- 존중(Respect): 팀원 간의 상호 존중

애자일 방법론

: 익스트림 프로그래밍(XP)

XP의 12가지 기본 원리

- 짝 프로그래밍(Pair Programming): 개발자 둘이서 짝으로 코딩하는 원리
- 공동 코드 소유(Collective Ownership): 시스템에 있는 코드는 누구든지 언제라도 수정 가능하다는 원리
- 지속적인 통합(CI; Continuous Integration): 매일 여러 번씩 소프트웨어를 통합하고 빌드해야 한다는 원리
- 계획 세우기(Planning Process): 고객이 요구하는 비즈니스 가치를 정의하고, 개발자가 필요한 것은 무엇이며 어떤 부분에서 지연될 수 있는지를 알려주어야 한다는 원리
- 작은 릴리즈(Small Release): 작은 시스템을 먼저 만들고, 짧은 단위로 업데이트한다는 원리
- 메타포어(Metaphor): 공통적인 이름 체계와 시스템 서술서를 통해 고객과 개발자 간의 의사소통을 원활하게 한다는 원리
- 간단한 디자인(Simple Desing): 현재의 요구사항에 적합한 가장 단순한 시스템을 설계한다는 원리
- 테스트 기반 개발(TDD; Test Driven Develop): 작성해야 하는 프로그램에 대한 테스트를 먼저 수행하고 이 테스트를 통과할 수 있도록 실제 프로그램의 코드를 작성한다는 원리
- 리팩토링(Refactoring): 프로그램의 기능을 바꾸지 않으면서 중복제거, 단순화 등을 위해 시스템 재구성한다는 원리
- 40시간 작업(40-Hour Work): 개발자가 피곤으로 인해 실수하지 않도록 일주일에 40시간 이상을 일하지 말아야 한다는 원리
- 고객 상주(On Site Customer): 개발자들의 질문에 즉각 대답해 줄 수 있는 고객을 프로젝트에 풀타임으로 상주시켜야 한다는 원리
- 코드 표준(Coding Standard): 효과적인 공동 작업을 위해서는 모든 코드에 대한 코딩 표준을 정의해야 한다는 원리

애자일 방법론

: 익스트림 프로그래밍(XP)

XP의 12가지 기본 원리

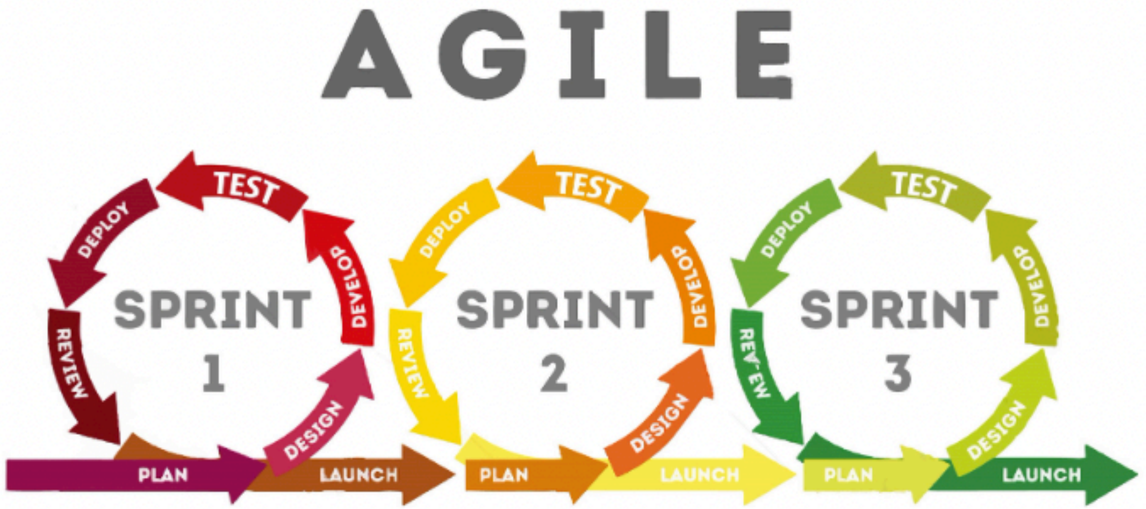
- 짝 프로그래밍(Pair Programming): 개발자 둘이서 짝으로 코딩하는 원리
- 공동 코드 소유(Collective Ownership): 시스템에 있는 코드는 누구든지 언제라도 수정 가능하다는 원리
- 지속적인 통합(CI; Continuous Integration): 매일 여러 번씩 소프트웨어를 통합하고 빌드해야 한다는 원리
- 계획 세우기(Planning Process): 고객이 요구하는 비즈니스 가치를 정의하고, 개발자가 필요한 것은 무엇이며 어떤 부분에서 지연될 수 있는지를 알려주어야 한다는 원리
- 작은 릴리즈(Small Release): 작은 시스템을 먼저 만들고, 짧은 단위로 업데이트한다는 원리
- 메타포어(Metaphor): 공통적인 이름 체계와 시스템 서술서를 통해 고객과 개발자 간의 의사소통을 원활하게 한다는 원리
- 간단한 디자인(Simple Desing): 현재의 요구사항에 적합한 가장 단순한 시스템을 설계한다는 원리
- 테스트 기반 개발(TDD; Test Driven Develop): 작성해야 하는 프로그램에 대한 테스트를 먼저 수행하고 이 테스트를 통과할 수 있도록 실제 프로그램의 코드를 작성한다는 원리
- 리팩토링(Refactoring): 프로그램의 기능을 바꾸지 않으면서 중복제거, 단순화 등을 위해 시스템 재구성한다는 원리
- 40시간 작업(40-Hour Work): 개발자가 피곤으로 인해 실수하지 않도록 일주일에 40시간 이상을 일하지 말아야 한다는 원리
- 고객 상주(On Site Customer): 개발자들의 질문에 즉각 대답해 줄 수 있는 고객을 프로젝트에 풀타임으로 상주시켜야 한다는 원리
- 코드 표준(Coding Standard): 효과적인 공동 작업을 위해서는 모든 코드에 대한 코딩 표준을 정의해야 한다는 원리

애자일 방법론

: 스크럼

매일 정해진 시간, 장소에서 짧은 시간의 개발을 하는 팀을 위한 프로젝트 관리 중심 방법론

스프린트를 중심으로 하는 개발 방법론



애자일 방법론

: 스크럼 주요 용어

백로그(Backlog)
: 제품과 프로젝트에 대한 요구사항

스크럼 마스터(Scrum Master)
: 프로젝트 리더, 스크럼 수행 시 문제를 인지 및 해결하는 사람

스프린트(Sprint)
: 반복적인 개발 주기 (계획 회의부터 제품 리뷰가 진행되는 날짜 까지의 기간이 1스프린트)

일일 스크럼 회의(Daily Scrum Meeting)
: 날마다 진행되는 미팅 (어제 한일, 오늘 할일, 장애 현상 등을 공유)
To-Do-List 계획 수립

애자일 방법론

: 단점

빠르기만 한 배포

- 짧은 개발로 완성도가 미흡하거나 충분히 테스트 되지 않은 기능이 출시됨

잦은 요구사항 변경

- 잦은 재설계 발생 -> 개선해야하는데 리펙토링 생략
- 지속적으로 코드 품질 하락(나쁜 구조에 계속 기증 추가)

이름만 애자일인 막 코딩

- 개발 단계를 생략한 채 그냥 개발하고 고치는 관행으로 복귀
- 아무런 계획도 문서도 테스트도 없음

소프트웨어 개발 방법론

: 그래서

애자일이 무조건 옳다?

NO NO!

소프트웨어 개발 방법론

: 그래서

프로젝트 참여자의 수준,

프로젝트 특성 및 규모,

요구사항의 명확도 등을 고려해서

적절한 방법을 선택

2023-05-21

감사합니다.

김도경