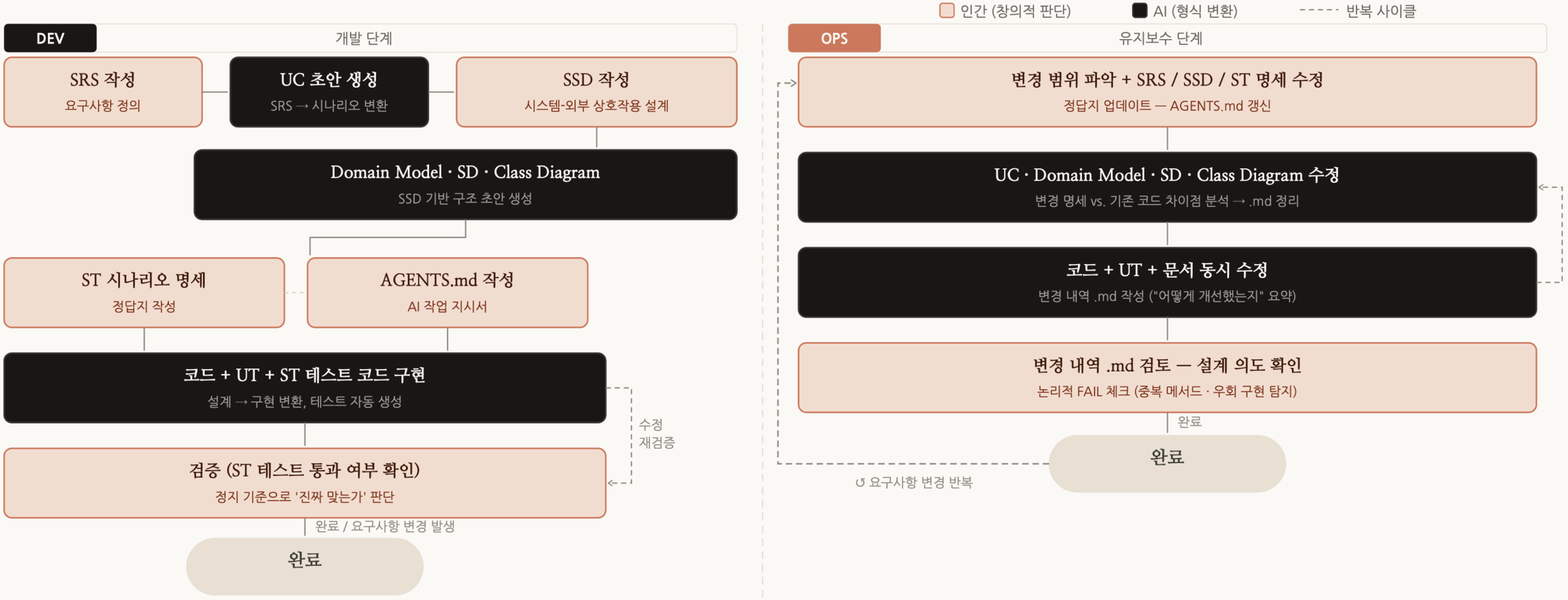
Human-Anchored AI Development

인간 중심의 AI 협업 개발 방법론

"창의적 판단은 인간이, 형식 변환은 AI가" — 단, 정답지는 항상 인간이 들고 있어야 한다

6팀 — 정상훈 · 정수혁 · 하재아 · 한지훈

Human-Anchored AI Development — 전체 흐름도



핵심 원칙 | "창의적 판단은 인간이, 형식 변환은 AI가" — 단, 정답지는 항상 인간이 들고 있어야 한다

01

핵심 철학

왜 인간이 정답지를 들고 있어야 하는가

핵심 철학

역할 분담의 원칙

창의적 판단 vs. 형식 변환

인간이 담당하는 것

창의적 판단

- SRS 작성 — 비즈니스/도메인 맥락 이해 필요
- SSD 작성 — 시스템과 외부의 상호작용 설계
- ST 시나리오 명세 — 정답지는 인간이 작성
- AGENTS.md 작성 — AI 작업 지시서 정의
- 검증 — 정답지 기준으로 '진짜 맞는가' 판단

AI가 담당하는 것

형식 변환

- UC 초안 생성 — SRS를 시나리오 형식으로 변환
- Domain Model / SD / Class Diagram 초안
- 코드 + UT 구현 — 설계를 코드로 변환
- ST 테스트 코드 변환 및 실행
- 수정 → 재검증 반복

핵심 요약 | AI는 형식을 변환하고, 인간은 창의적 판단과 검증으로 품질을 보장한다

02

개발 단계 흐름

SRS부터 검증까지 — 단계별 인간/AI 역할 구분

개발 단계 흐름

단계별 프로세스

인간과 AI의 협업 흐름도

01	인간	SRS 작성	비즈니스/도메인 맥락 이해, 요구사항 정의
02	AI	UC 초안 생성	SRS를 유스케이스 시나리오 형식으로 변환
03	인간	SSD 작성	시스템-외부 상호작용 설계 (창의적 판단)
04	AI	Domain Model · SD · Class Diagram	SSD 기반 구조화 및 코드 수준 변환
05	인간	ST 시나리오 명세 + AGENTS.md	정답지 작성 + AI 작업 지시서 정의
06	AI	코드 + UT + ST 테스트 구현	설계→구현 변환, 테스트 코드 자동 생성
07	인간	**검증 및 완료**	정답지 기준으로 '진짜 맞는가' 판단

핵심 원칙

- **정답지**
  ST 시나리오 명세는 반드시 인간이 작성
- **검증 기준**
  AI 코드 기준이 아닌 정답지 기준으로 판단
- **모호함 금지**
  AGENTS.md는 명확하고 구체적으로 작성
- **단계별 지시**
  한 번에 모든 것을 시키지 않는다

03

유지보수 단계

변경 사이클 — 코드와 문서를 동시에 수정하는 AI 활용법

유지보수 단계

변경 사이클 흐름

요구사항 변경 발생 시 반복되는 프로세스

인간 **변경 범위 파악 + SRS/SSD/ST명세 수정**
정답지 업데이트 포함

AI UC + Domain Model + SD + Class Diagram 수정

AI 명세 vs. 기존 코드 차이점 분석 → .md 정리
"구현 불가 이유", "없는 인터페이스" 등 명시

AI 코드 + UT + 문서 동시 수정 + 분석 .md 작성
기존과 달라진 점 요약

인간 **AI 변경 내역 검토 — 설계 의도 확인**
논리적 FAIL 주의, 우회 구현 체크

AI 수정 → 재검증 반복 → 완료

유지보수의 AI 장점

코드 + 문서 동시 수정
코드만 고치고 문서를 잊어버리는 전통적인 문제를 해결

인간 검증 부담 절감
AI가 요약한 변경 내역을 보고 판단하므로 프로젝트가 커도 검증 가능

논리적 FAIL 방어
중복 메서드·우회 구현을 정답지 기준으로만 걸러낼 수 있음

04

AGENTS.md

AI 에이전트에게 주는 작업 지시서 작성법

AGENTS.MD

작성 가이드

반드시 포함해야 할 6가지 항목

작업 범위
어떤 파일을 수정할지 명시

"기존 파일만 수정하고 새 파일을 만들지 마라"

인터페이스 제약
기존 설계를 존중하도록 지시

"기존 인터페이스를 재사용해라"

품질 기준
AI가 작업 완료 여부를 스스로 판단할 수 있는 명확한 조건

"빌드와 테스트가 통과하는지 확인해라"

문서화 규칙
변경사항 추적

"수정 부분에 [추가]/[삭제]/[변경] 태그 표시"

검증 방법
AI가 스스로 검증하게 유도

"CI를 통해 모든 테스트가 통과하는지 확인"

참고 문서
AI가 읽어야 할 문서 경로 제시

"설계 문서를 먼저 읽고 그 내용으로 작업"

핵심 원칙 | 모호한 표현 금지 · 단계별 지시 · AI 초안은 반드시 설계 의도에 맞는지 확인

05

검증 전략

개발 단계와 유지보수 단계의 검증 방법론

검증 전략

검증 프로세스 비교

개발 단계와 유지보수 단계의 검증 흐름은 다르다

개발 단계 검증

- ST 테스트 실행 → 통과 여부 확인 → 완료
- 인간이 미리 만든 정답지(ST 명세) 기준으로 통과 여부만 보면 됨
- 명세가 촘촘할수록 테스트 통과만으로 충분한 검증이 됨

포인트

기존 코드가 없는 개발 단계에서는 AI가 우회할 여지가 없으므로 논리적 FAIL이 원천적으로 발생하기 어렵다

유지보수 단계 검증

- 변경된 명세 vs. 기존 코드 차이점 분석 → .md로 정리
- AI가 코드·UT·문서 수정 후 변경 내역 .md 작성
- 인간이 변경 내역 .md 검토 → 논리적 FAIL 체크 → 완료

논리적 FAIL이란

빌드·테스트는 통과했지만 기존 코드를 건드리기 싫어 중복 메서드로 우회한 경우 — 변경 내역 문서를 인간이 직접 읽어야만 잡아낼 수 있다

핵심 차이 | 개발은 테스트 통과로 충분 — 유지보수는 변경 내역 문서를 인간이 반드시 읽어야 한다

06

CONCLUSION

"AI를 잘 쓰려면 오히려 도메인을 더 잘 알아야 한다"

1. AI 초안을 검토하려면 내가 먼저 알아야 함
2. AI가 틀렸을 때 잡아내려면 정답을 알아야 함
3. SRS 작성 수준의 도메인 이해로도 충분히 시작 가능

감사합니다