TEAM 2

Optimized and Balanced SW Development Process

강병완 202211248
강현준 202211251
박　완 202211301
정민수 202211365

Table of Contents

- 문제 정의
- 방법론 그림
- 주요 원칙
- Inception
- Elaboration (OOA, OOD)
- Construction (OOI)
- Transition
- Iteration
- DevOps

문제 정의

- AI Coding Agent는 빠르지만
  모호한 요구사항과 무제한 권한에 취약함
- 사람이 만든 OOAD는 처음에는 체계적이지만,
  변경이 생기면 문서 · 설계 · 코드 · 테스트가 서로 어긋나기 쉬움
- Dev와 Ops를 연결하면서 AI의 생산성과
  Human의 책임을 균형 있게 결합해야 함

PAI-DevOps 방법론

Plan-Driven AI DevOps with Human Responsibiity

Ambiguity-Free | Plan-Driven | Architecture-Centric | Iterative & Incremental | Risk-Oriented

1 Inception
- LCO Gate

- Vision
- Project Plan
- Initial UC Model
- FR-NFR Draft
- Risk List

PM / Customer

프로젝트 목적, 범위, 요구사항 초안,
주요 위험, 진행 타당성을 합의하는 단계

2 Elaboration
- OOA & OOD

OOA
- Requirements Analysis Use Case
- Acceptance Criteria
- Domain Model
- SSD
- Traceability
- Analysis Review

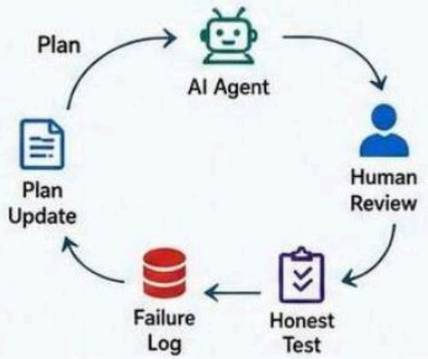
OOD
- Architecture Baseline
- Class & Sequence
- Interface Spec
- Test Strategy
- Construction Plan

Architect | Architect

OOA: 무엇을 만들지 명확화 | OOD: 어떻게 만들지 구조화

3 Construction
- OOI

AI-Human Micro Loop



Code + Tests | PR Review | Test Report | Failure Log

AI generates, Human approves.

Human Reviewer / QA

4 Transition / Operations
- Product Release

- Acceptance Test
- Deployment
- Monitoring
- Runbook
- Incident Response
- Product Release

Ops / DevOps

인수, 배포, 운영, 장애 대응,
운영 데이터 수집 단계

Failure Log + Ops Data + User Feedback → Risk / NFR / Plan Update → Next Iteration

주요 원칙

- **Ambiguity-Free :** Plan, Test, Risk에서 모호성을 배제한다.
- **Plan-Driven :** Plan이 Source of Truth, 모든 활동의 출발점
- **Architecture-Centric :**
  안정된 아키텍처를 먼저 확립하고, 그 위에서 모든 것을 진행한다.
- **Iterative & Incremental :**
  Elaboration→Construction→Transition을 한 사이클로 반복하며,
  매 Iteration 마다 Customer 피드백을 반영하여 수정한다.
- **Risk-Oriented :** Risk를 Structural Risk, Runtime Risk로 나누고,
  Structural Risk는 Inception-Elaboration 단계에서 조기에 해소,
  Runtime Risk는 NFR, Failure Log를 통해 점진적 해소.

주요 원칙

- **Ambiguity Control :**
  Plan : Input과 Output(Goal)이 반드시 명시되어야 한다.
  Test : 검증 대상과 방법을 명시한다.
  LLM AI는 기본적으로 Black-Box 상태 → 같은 input에 대해 같은 output을 기대하기 어려움.
  → 지침과 명확한 목표를 설정하여 과정에 차이가 있더라도, output의 범위를 좁힐 수 있다.

- **Risk Control :**
  - AI 사용으로 발생한 Risk의 책임은 해당 AI를 사용해 Output을 만든 사람이 진다.
  - Risk는 2가지로 나뉜다.
    - Structural Risk : 프로젝트 전반적으로 발생할 수 있는 Risk,
      Inception-Elaboration 단계에서 조기에 식별해, 프로젝트를 진행하며 발생하지 않도록 예방
    - Runtime Risk : 초기에 Risk List, NFR 등을 통해 프로젝트 간 발생할 수 있는 Risk를 관리한다.
      특히, AI를 사용하여 발생하는 Hallucination, Prompt Injection 등의 문제를 명시한다.
      이후 초기에 식별하지 않은 Risk가 발생할 때, Failure Log로 기록하고 학습하여 점진적으로 개선시킨다.

주요 원칙

- **Plan :**
  - Source of Truth → 모든 활동의 출발점이자 **지침**
  - 반드시 명확한 input과 output(goal)이 존재해야 함.
  - 실행 전, 실행 중에 Refine 가능
  - Project Plan, Elaboration/Construction/Transition Plan으로 구분
  - Project Plan은 Inception Phase에서 설계되어, 프로젝트 전반적으로 지키는 지침
    AI agent가 항상 동적으로 읽어야 할 지침
  - Elaboration/Construction/Transition Plan은 Phase 실행 시 구체화된다.
  - 모든 Plan은 Iteration 단위로 갱신된다.
  - AI가 초안 작성, 인간이 검토·승인
  - AI가 한 번에 수행 가능한 크기로 유지, 크면 하위 Plan으로 분할
  - Plan은 다음과 같이 상태가 변화한다. :
    Draft(AI 작성 초안) → Approved(by Human) → Executing → Verified

Inception

Inception

- **Inception phase**

- **UP와의 공통점**
  - A short requirements workshop and feasibility study
  - Most actors, goals, and use cases named
  - Most use cases written in brief format
  - Most influential and risky requirements identified.

- **UP와의 차이점**
  - 프로젝트 전반적으로 지켜져야할 지침 Project Plan.
    NFR과 유사하지만, AI agent는 해당 Project Plan을 작업 시작 전에 동적으로 context화 하여 사용.
  - AI Agent가 Document 구조 Refinement, 내용 정리 부분에서 개입할 수 있다.
  - 방법론 자체의 운용 규칙은 AGENTS.md에 정의되어 AI agent가 세션 시작 시 자동 로드한다
    (Project Plan = 프로젝트별 지침, AGENTS.md = 방법론 공통 규칙)

Inception

산출물	설명
Project Plan ★	Input·Goal·Output·Test·NFR과 전체 진행 지침
LCO Review Record ★	검토 결과·미해결 항목·승인 여부·승인자
Vision Document	목적·범위·핵심 기능·제약·성공 기준
Glossary 초안	도메인·프로젝트 핵심 용어 정의
Use Case Model 초안	UC Diagram과 주요 UC 개요
FR/NFR 초안	기능 요구사항과 측정 가능한 품질·운영 제약
Risk List 초안	Structural/Runtime Risk·출처·영향·대응 계획

Elaboration

Elaboration

- **Elaboration phase :**

- **UP와의 공통점**
  - The core, risky software architecture is programmed and tested.
  - The majority of Requirements are discovered and stabilized.
  - The major risks are mitigated or retired.
  - Important Artifacts : Domain models, Design Models, SW Architecture Document

- **UP와의 차이점**
  - AI agent는 Project plan을 기반으로 해당 단계에서 Elaboration plan을 설계, document refinement를 진행
  → 단, Requirement를 완전히 discover, stabilize하는 과정은 Human vs Human (Team vs Customer). 인간이 주도적으로 Requirement를 정의할 수 있어야 한다. (책임 소재 분명)

Elaboration: OOA

UP	OOAD	설명
Project Management	Elaboration Plan	Iteration 범위, Goal에 대한 기준
	Risk List	Structural Risk를 면밀히 파악하여 Major Risk가 mitigated 또는 retired 해야 함.
Requirements	Use Case Diagram	Actor, System Boundary, Use case lists
	Use Case Casual Format	Scenario (Common Case, Alternative, Exceptional), Precondition, Goal이 명확해짐. → Majority of Requirements discovered and stabilized
	FR / NFR	Functional Requirement, 측정 가능한 품질 제약
Domain Model	Domain Model (Objects, Concepts)	핵심 도메인 개념, 속성, 관계
System Behavior	System Sequence Diagram	Actor-System 간의 Black-box 이벤트 흐름
	System Operation Contract	System과 Actor 간의 Operation, Operation 호출 precondition, goal

Elaboration: OOD

UP	OOAD	설명
Project Management	Construction Plan	구현 우선순위, Iteration, Risk, 완료 기준 등
	Risk List	Structural Risk를 면밀히 파악하여 Major Risk가 mitigated 또는 retired 해야 함.
SW Architecture (AD)	SW Architecture (AD)	Layer, Component, 기술 구조, 핵심 설계 원칙
Design Model	Sequence Diagram	UC 수행을 위한 Object 메세지, 호출 순서
	Class Diagram	Class Attribute, Operation, Interface, Relationship

Construction

Construction:OOI

- **Construction phase :**

- **UP와의 공통점**
  - Incremental implementation of the remaining lower risk and easier elements
  - Preparation for deployment in the transition phase
  - Applying TDD and Refactoring

- **UP와의 차이점**
  - AI agent가 Plan을 기반으로 Code 및 Test (Unit, System, Integrated)를 생성.
    1개 이상의 모델을 사용하여 Code review, PR까지 진행.
  - AI agent에게 implementation 지시할 때 거대한 하나의 단위가 아닌 의미 단위, 도메인 단위 진행
  - AI agent가 생성하는 Test가 단순 Coverage를 늘리는 것이 아닌, 실질적 기능을 하도록 확인
  - Implementation, CI, 등 발생한 error 및 해결 과정을 Failure Log에 작성
    → 같은 error를 반복하지 않도록 예방하고, Iteration을 반복하며 AI Agent의 Quality를 점진적으로 증가시키기

Construction:OOI

Outputs	설명
Source Code + Automated Tests	Architecture·Coding Standard 준수 구현 및 Unit/Integration/System/NFR Test
Test Report	환경·실행 결과·Coverage·미해결 Failure
PR/Review Record	변경 이유·영향·Test·승인·논의 기록
누적 Failure Log	전체 Iteration에 걸친 Test·구현 실패·패턴·학습 기록
Transition Plan	Release·인수·배포·Rollback·운영 이관 계획
Iteration Plan (per Iteration)	Goal·Input·Output·Scope·Test·NFR·Risk

Transition

Transition

산출물	설명
Acceptance Test Report ★	Customer 인수 기준과 실행 결과
Deployment/Rollback Record ★	배포 환경·절차·결과·복구 기록
Release Package/Note	Version·배포 대상·변경 내용·알려진 문제
Monitoring Dashboard/Alert	주요 NFR 지표·Threshold·Alert
Runbook 및 사용자 문서	운영·장애 대응·사용 절차
Incident Report	장애 영향·원인·복구·재발 방지 대책
갱신된 Failure Log/Risk/NFR ★	운영 학습·패턴·복귀 단계 기록 — 다음 Phase Iteration의 핵심 Input
Post-Deployment Review	Release 평가·다음 Iteration 개선 항목·Phase 재진입 여부 판단

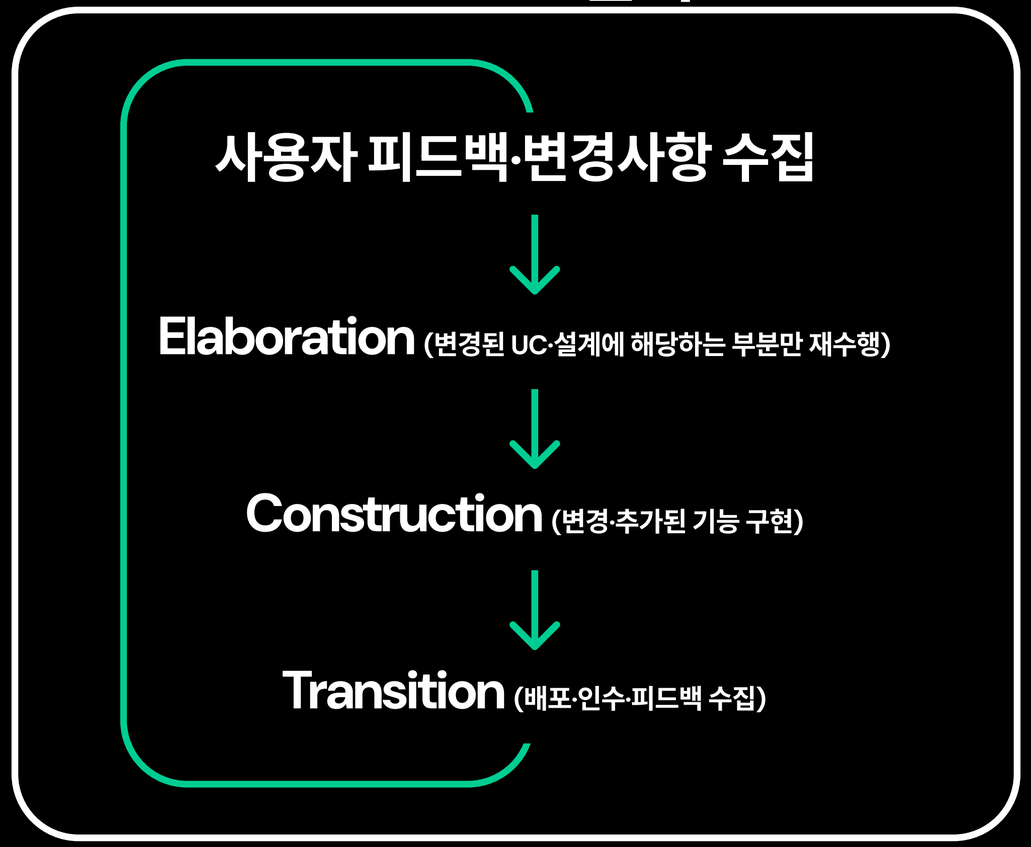
Iteration

Iteration 구조

Inception (최초 or 큰 범위 변경 시)

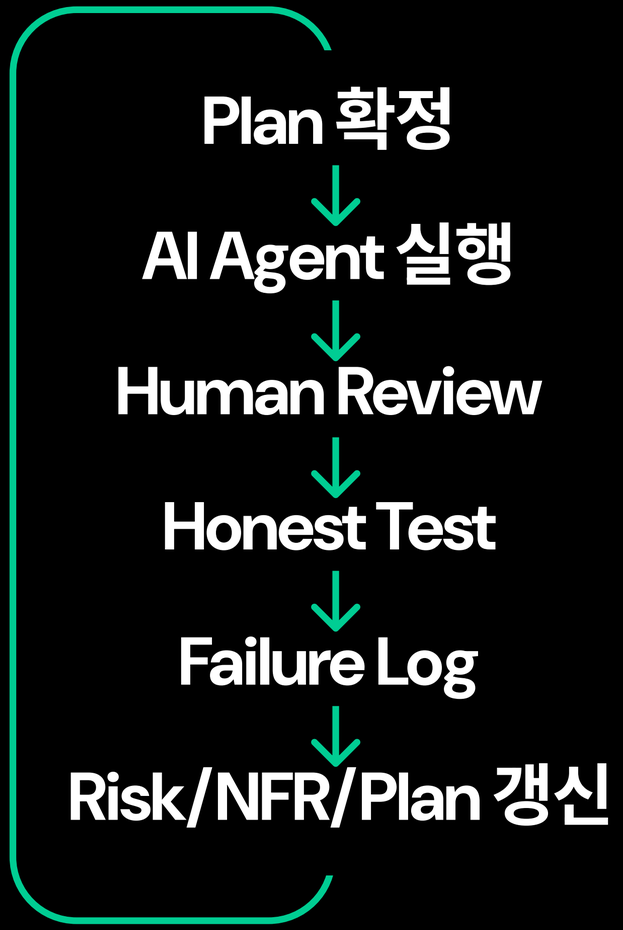
↓

Iteration 반복



Iteration 실행 흐름

단계	Human	AI Agent	완료 증거
Plan 확정	Goal·Scope·Test·Risk·완료 기준 승인	누락·모호성·충돌 검사	승인된 Iteration Plan
AI Agent 실행	Harness·제약 제공, 작업 범위 통제	Plan(Harness) 내 Code·Test·문서 초안 생성	변경 초안·실행 기록
Human Review	Architecture·보안·로직·추적성·범위 검토	Review 후보·영향 분석 제공	Review Record
Honest Test	QA 독립 검증, 실패 경로 확인	Test 실행·실패 원인 분석 지원	Test Report
Failure Log 갱신	원인·조치·재현·Test·소유자 확정	Failure 구조화, 연결 항목 제안	갱신된 Failure Log
Plan 갱신	Risk/NFR/우선순위·다음 Goal 확정	갱신 후보·다음 작업 제안	다음 Iteration Plan



DevOps

DevOps Pipeline

Plan

→ Issue/Task

→ Branch

→ AI 구현

→ Local Build/Test

→ PR

→ Human Review

→ CI Build/Test

→ Merge

→ Deploy

→ Monitor

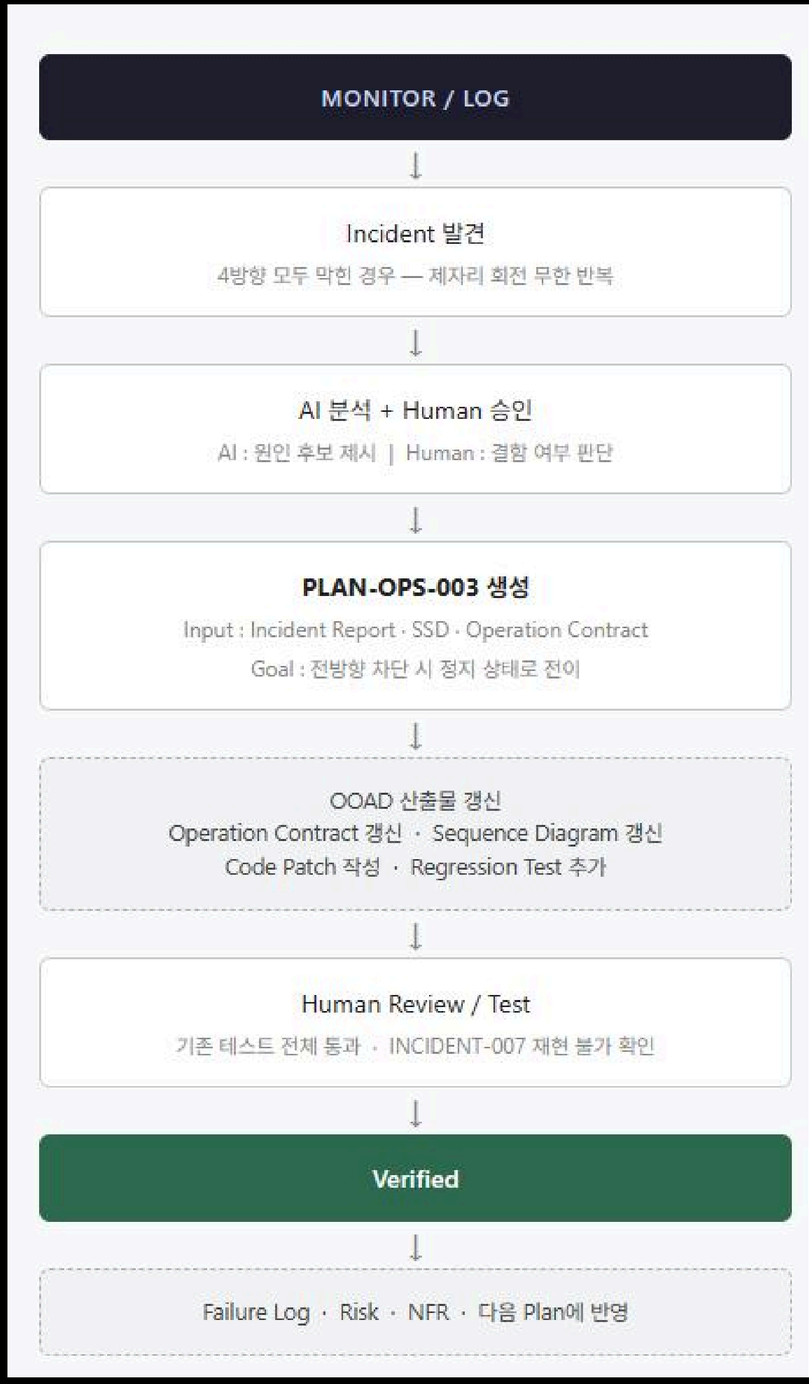
→ Failure Log

→ Risk/NFR/Plan 갱신

→ 다음 Plan

규칙	내용
1	모든 변경은 승인된 Plan/Issue와 연결
2	main 직접 수정 금지 — Branch·PR 사용
3	Build·필수 Test 실패 시 Merge 금지
4	AI 생성 Code·Test는 Human Review 필수
5	PR에 변경 이유·영향 산출물·Test 결과·Risk 기록
6	배포는 승인된 Release·Rollback Plan 존재 시에만
7	운영 Failure는 Failure Log 기록 후 다음 Plan으로 환류

DevOps Example



Plan Example

id	PLAN-OPS-003
phase	Transition/Operations
status	Approved
author	AI (draft) / 강병완 (approved)
trigger	운영 중 결함 발견 (Failure Log + Ops Data 분석으로 AI 자동 감지)
inputs	• INCIDENT-007 • ops/failure-log.md • UC-04-포위상태탈출.md • ooa/ssd/SSD-UC04.md • `ooa/contracts.md (OP-05: escapeSurrounded 계약)` • risks/risk-list.md
goal	전.후.좌.우 4방향이 모두 막힌 상황에서 회전만 무한 반복하는 결함 수정 (정지 조건 누락)
outputs	• ooa/contracts.md 갱신 • ood/sd/SD-UC04.md 갱신 • 패치 PR • tests/TC-15 • risks/risk-list.md 갱신
acceptance	• 4방향 차단 시나리오에서 정지 상태 전이 확인 • 기존 TC 전체 회귀 통과 (Honest Test) • INCIDENT-007 재현 불가 확인

수행 단계 (AI-Human Micro Loop)

#	작업	담당
1	INCIDENT-007 재현 테스트 작성: 4방향 차단 → 무한 회전 (실패 확인, Failure Log 기록)	AI
2	원인 추적: OP-05 계약이 "막히면 회전"만 정의, 전방향 차단 케이스 미정의 → 설계 결함	AI
3	계약에 정지 상태 추가 → SD-UC04 수정 → 코드 패치 → TC-15 통과	AI
4	PR Review + Test Report 확인 → Verified 전환, 포스트모템 기록	강병완

추적 체인

INCIDENT-007 → PLAN-OPS-003 → UC-04 → OP-05 → SD-UC04 → PR #42 → TC-15

Thank you!