MSAP.ai

msap.ai

AI 시대 엔지니어의 문제 정의 역량

구현을 AI 가 맡을 때 국내 개발자가 길러야 할
요구 정의와 아키텍처 역량



사람이 이끄는 구간

요구 정의 **설계**

AI 가 이끄는 구간

구현 시험

CONTENTS

Agenda

KEY INSIGHT

구현을 AI 가 맡을수록 사람의 일은
요구 정의와 품질 판단으로 옮겨 감

— 이 두 역량이 국내 개발자의
커리어 방향을 정함

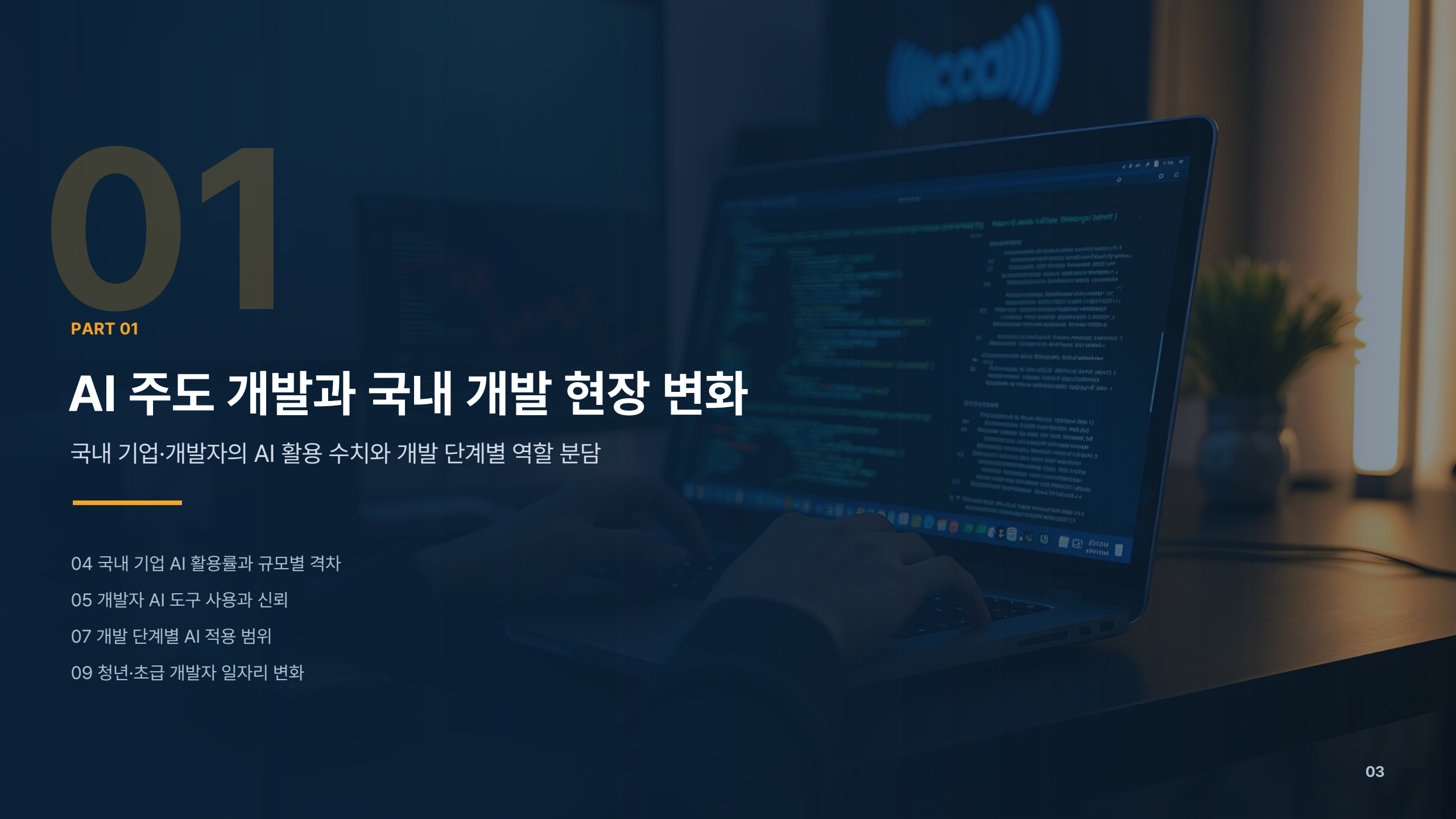
01 **AI 주도 개발과 국내 개발 현장 변화** 03–09
국내 기업·개발자의 AI 활용 수치와 개발 단계별 역할 분담

02 **기술 관점에 머무는 엔지니어의 사고 습관** 10–14
요구 정의로 옮겨 갈 때 걸림돌이 되는 습관과 그 원인

03 **요구를 정확히 정의하는 사고 기법** 15–20
구체와 추상, 목적·목표·수단, 요구사항 계층, 용어 사전

04 **아키텍트 역량과 기술 기본기** 21–28
품질 특성 선택과 그 바탕이 되는 기술 지식, 개발팀 점검표

출처: 장별 바닥글과 29장 참고 자료



01

PART 01

AI 주도 개발과 국내 개발 현장 변화

국내 기업·개발자의 AI 활용 수치와 개발 단계별 역할 분담

04 국내 기업 AI 활용률과 규모별 격차

05 개발자 AI 도구 사용과 신뢰

07 개발 단계별 AI 적용 범위

09 청년·초급 개발자 일자리 변화

국내 기업 AI 활용률과 규모별 격차

KEY INSIGHT 한 해 만에 9.4%p 오름 — 다만 제조 대기업 49.2%, 중소 4.2% 로 회사 규모에 따라 개발 방식이 갈라지는 중

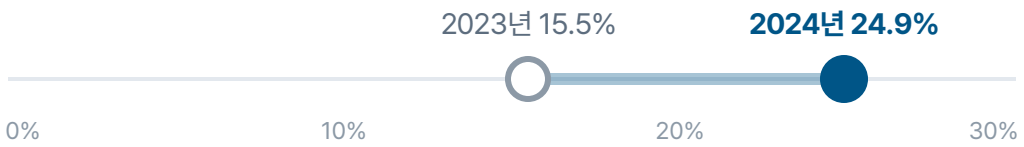
사업체 AI 업무 활용률

10,323개 사업체

24.9%

2024년 기준

전년 대비 +9.4%p



최근 3년 도입한 디지털 신기술 가운데

AI 비중 43.5% — 신기술 도입 1순위

제조기업 AI 활용률

504곳 · 보도 기준



같은 국내 시장 안에서 활용률 차이 약 12배

※ 두 조사는 대상과 질문이 달라 수치를 한 축에 겹쳐 비교하지 않음

출처: 과기정통부·KISDI "2025 디지털산업 실태조사"(2026.7) · 대한상공회의소 "기업의 AI 전환 실태와 개선방안"(2025.11, 보도 인용)

개발자 84% 가 AI 도구 사용, 신뢰는 33%

KEY INSIGHT 쓰는 사람은 많지만 결과를 그대로 믿지 못함 — 거의 맞는 답을 걸러 내는 검증이 개발자의 핵심 업무로 남음

AI 도구 사용 또는 사용 계획

84%

전문 개발자 매일 사용 50.6%

AI 출력 정확도를 신뢰

33%

세 명 중 한 명만 결과를 믿음

AI 출력 정확도를 불신

46%

신뢰보다 불신이 13%p 많음

개발자가 꼽은 가장 큰 불만

66% **거의 맞지만 틀린 답**

겉보기엔 동작하지만 요건과 다른 코드

45% **AI 코드 디버깅이 더 오래 걸림**

직접 짠 코드보다 원인 찾기가 어려움

※ 국내 개발자만 따로 조사한 공개 자료가 없어 글로벌 응답으로 표시

출처: Stack Overflow “Developer Survey 2025” AI 항목(글로벌 응답, 2025.7)

AI 도입 효과는 체감과 측정이 다름

KEY INSIGHT AI 는 팀이 이미 가진 기반을 키움 — 요건·리뷰·테스트 기반이 약하면 속도보다 배포 불안정이 먼저 커짐

DORA 2025 설문

응답자 약 5,000명

90%
업무에 AI 사용

80%+
생산성 향상을 체감

AI 도입과의 관계

↑ **처리량 · 제품 성과** — 양의 관계

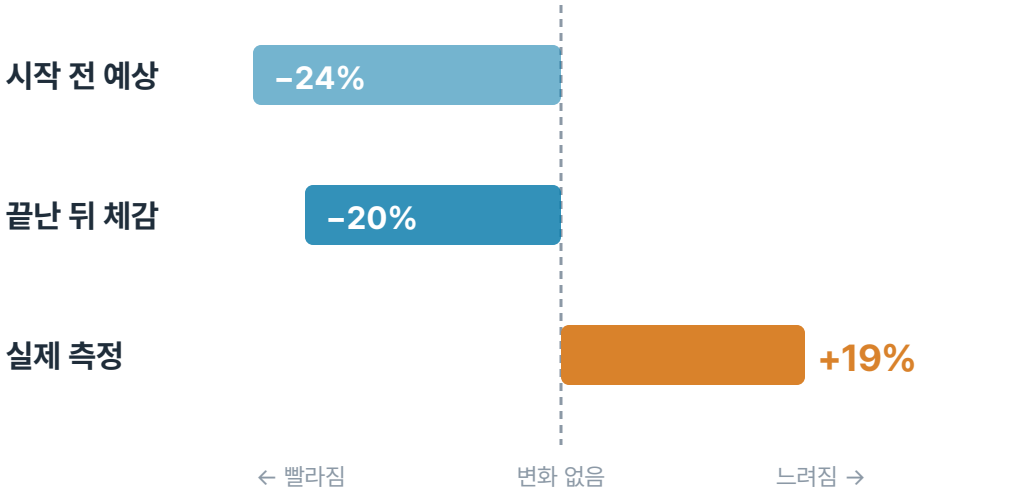
↓ **배포 안정성** — 음의 관계

DORA 결론: AI 는 팀을 고치지 않고
이미 있는 것을 증폭함

METR 무작위 대조 실험

숙련 개발자 16명 · 이슈 246건

AI 사용 시 작업 시간 변화



2025년 초 도구 기준 수치임. METR 은 2026.2 재측정에서
빨라지는 방향이 나왔으나 신호가 약하다고 발표

출처: Google Cloud “Announcing the 2025 DORA Report”(2025.9) · METR “Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity”(2025.7)

요구 정의 | 설계 | 구현 | 시험

개발 단계별 AI 적용 범위

KEY INSIGHT 구현과 시험은 AI 가 빠르게 맡고 요구분석은 사람 판단이 남음 — 사람의 일이 요구사항을 정의하는 쪽으로 옮겨 감

사람 판단이 남는 구간 | AI 가 빠르게 맡는 구간

요구사항 분석	설계	구현	시험
AI 적용 정도			
낮음	중간	높음	최고
생성형 AI 활용이 쉽지 않은 단계	초안은 AI 가 쓰고 결정은 사람이 내림	코드 생성·수정을 AI 가 대부분 맡는 추세	생성형 AI 가 가장 효과적인 단계

> **요구분석·설계 단계는 생성형 AI 의 활용이 쉽지 않음**
> **시험 단계가 생성형 AI 가 가장 효과적인 단계**
>
> 소프트웨어정책연구소(SPRi), 생성형 AI 에 의한 소프트웨어 개발자 업무 영향 분석 요지

출처: SPRi "생성형 AI에 의한 소프트웨어 개발자 업무 영향 분석"(IS-179, 2024.8). 단계별 적용 정도는 이 분석을 바탕으로 한 상대 표시

요구 정의 | 설계 | 구현 | 시험

문제 정의는 사람, 문제 해결은 AI 가 주도

KEY INSIGHT AI 결과물을 리뷰하려면 요건을, 요건을 판단하려면 고객 요구를 알아야 함 — 리뷰 역량과 요구 이해 역량은 한 묶음임

문제 정의 구간

요구 정의 · 요건 정의

Pilot
사람

Copilot
AI

사람이 고객 요구와 제공할 가치를 정의하고
AI 가 자료 정리·초안 작성으로 보조

문제 해결 구간

설계 · 구현

Pilot
AI

Copilot
사람

AI 가 설계안과 코드를 만들고
사람이 그 결과물을 리뷰

AI 결과물 리뷰가 요구 이해로 이어지는 연쇄

1 결과물이 맞는지 리뷰 → **2** 맞음의 기준인 요건과 대조 → **3** 요건이 맞는지 보려고 고객 요구 이해 → **4** 정의한 요구 자체의 타당성 검증

출처: SPRi IS-179(2024.8) 단계별 분석 · Google Cloud “2025 DORA Report”(2025.9)를 바탕으로 역할 구분을 정리

청년·초급 개발자 일자리가 먼저 영향받음

KEY INSIGHT 경력이 짧은 사람의 정형화된 구현 업무부터 줄어듦 — 경력 초기부터 요구 이해와 검증 경험을 쌓아야 경쟁력이 남음

4년간 줄어든 청년(15~29세) 일자리 28.5만 개

94%

AI 고노출 업종에서 감소

28.5만 개 가운데 26.8만 개

업종별 청년 고용 변화



⚠ 한국은행은 이 감소를 AI 단독 효과로 보기는 신중해야 한다고 단서를 붙임

SW 개발 전문가 26명 델파이 조사

초급 개발자 일자리가 생성형 AI 의 영향을 가장 크게 받음

SPRi, SW개발자 채용 변화 전망과 생성형 AI

경력 편향 기술 변화

3년간 청년 일자리 21.1만 개 감소,
같은 기간 50대 일자리 20.9만 개 증가

경력이 짧은 사람의 정형화된 업무부터 대체

출처: 한국은행 BOK 이슈노트 제2026-19호(2026.8)·제2025-30호(2025.10) · SPRi "SW개발자 채용 변화 전망과 생성형 AI"(RE-192, 2025.4)



02

PART 02

기술 관점에 머무는 엔지니어의 사고 습관

요구 정의 단계로 옮겨 갈 때 걸림돌이 되는 습관과 그 원인

11 어떤 문제를 푸느냐는 질문과 기술 답

12 현업 요구를 테이블 설계로 바꿔 말하는 습관

13 공공 SW 사업의 과업 변경

14 아는 틀로 해석하는 스키마

어떤 문제를 푸느냐는 질문에 기술 답만 나옴

KEY INSIGHT 고객이 겪는 문제까지 말하지 못하면 AI 에 줄 요건도 정할 수 없음 — 사안을 정리해 결론을 내는 논리적 사고가 필요함

질문: 이 작업으로 고객의 어떤 문제가 풀리나요?

흔히 돌아오는 답

- API 를 바르게 동작시키려고
- 성능 요건을 맞추려고
- 그 칼럼이 NULL 이면 장애가 나서
- 요구사항 정의서 항목을 맞추려고

고객 문제로 답한 예

주문을 취소한 고객이 다시 구매하지 않아 매출이 줄어드는 문제

누가 · 무엇 때문에 · 어떤 손해를 보는지

논리적 사고 복잡한 사안을 정리해 순서대로 이해하고 결론을 얻는 사고 방식

기술 답은 수단을 말하고, 고객 문제 답은 목적을 말함

출처: 설명용 예시(특정 조사 수치 아님)

현업 요구를 바로 테이블 설계로 바꿔 말함

KEY INSIGHT 해결책을 먼저 내면 요구 확인 단계를 건너뜀 — 요구가 틀리면 구현이 맞아도 다시 만들어야 함

요구사항 협의 회의

현업

주문을 취소한 고객에게
다시 연락하고 싶어요

개발자

그럼 주문 테이블에 취소 사유 칼럼과
알림 플래그를 추가하면 됩니다

해결책을 바로 제안

개발자의 의도와 실제 결과

의도: 빠르게 해결 방법을 제안

결과: 목적을 묻지 않은 채 설계가 먼저 정해짐

현업은 테이블·칼럼 용어를 쓰지 않아 대화가 어긋남

건너뛴 질문

목적

왜 연락하려는가

재구매 유도인지, 불만 대응인지

성공 기준

무엇으로 성공을 판단하는가

재구매율인지, 응답률인지

대상 범위

누구에게 연락하는가

모든 취소 고객인지, 특정 사유인지

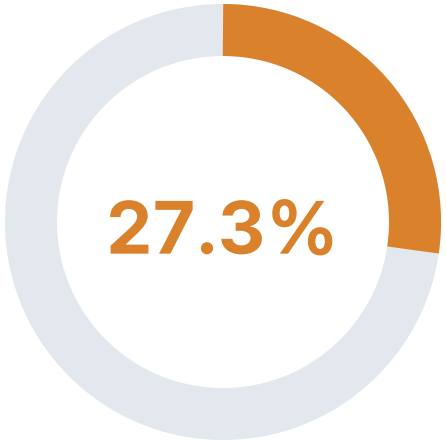
출처: 설명용 예시(국내 커머스 SI 상황 가정)

공공 SW 사업 27.3% 가 과업 변경을 경험

KEY INSIGHT 요구사항이 구축 도중에 구체화되는 구조가 원인 — 법으로 요구사항 상세화를 의무화할 만큼 반복된 약점임

공공 SW 사업 참여 기업의 과업 변경 경험

2018년 실태조사



정부가 밝힌 원인

시스템이 구축되면서 요구사항이 구체화되는 특성

착수 때 정한 요구가 개발 중에 바뀜

과기정통부는 2020년 과업 변경 절차를 정리한 공공SW사업 과업변경 가이드를 배포

해외 조사 · PMI 2014

47%

목표에 못 미친 프로젝트 가운데 부정확한 요구사항 관리가 원인

분모는 목표에 못 미친 프로젝트 수

소프트웨어 진흥법 제44조

발주 시 요구사항을 상세하게 작성해 공개하도록 의무화

분석·설계 사업을 분리 발주할 수 있는 근거도 포함

출처: 과기정통부 "공공SW사업 과업변경 가이드" 보도자료(2020.4) · PMI "Pulse of the Profession: Requirements Management"(2014) · 소프트웨어 진흥법 제44조(2020.12 시행)

아는 틀로 해석하는 스키마

KEY INSIGHT 기술 지식이 깊을수록 요구도 기술 틀로 바꿔 듣게 됨 — 이 경향을 알아야 상대의 말을 그대로 받아 적을 수 있음

스키마 경험으로 쌓인 지식의 틀 — 사람은 새 정보를 이 틀에 맞춰 해석함

바틀렛의 기억 실험 (1932)

1. 입력
낯선 문화권의 민담을 읽힘

↓

2. 반복 회상
시간을 두고 여러 번 다시 말하게 함

↓

3. 결과
낯선 부분은 빠지고
익숙한 표현으로 바뀜

같은 구조

엔지니어가 요구를 들을 때

1. 현업의 말
고객이 다시 찾아오게 하고 싶음

↓

2. 기술 틀로 해석
무의식중에 데이터·기능 단위로 쪼갬

↓

3. 결과
방문 이력 테이블·알림 API 로 기록
기술력 자체는 AI 결과물 판단에 계속 필요

출처: F. C. Bartlett, Remembering: A Study in Experimental and Social Psychology(Cambridge University Press, 1932). 오른쪽은 설명용 예시

03

PART 03

요구를 정확히 정의하는 사고 기법

구체와 추상, 목적과 목표와 수단, 요구사항 계층, 용어 사전



16 구체와 추상을 오가는 두 질문

17 목적과 목표, 수단의 구분

18 ISO/IEC/IEEE 29148 요구사항 계층

19–20 상위 목적 찾기와 팀 용어 사전

구체와 추상을 오가는 두 질문

KEY INSIGHT 매일 하는 일이 구체화라 추상화 연습이 부족함 — AI 가 구체화를 맡을수록 위로 묻는 질문이 사람 몫으로 남음

추상화 질문 · 위 방향

"즉, 무엇을 위해?"

이 구현이 채우는 요구는
무엇인가

경험이 적어 서툰 방향

추상화의 뜻

대상의 여러 특징 가운데
목적에 맞는 특징만
뽑아내는 일



구체화 질문 · 아래 방향

"예를 들면?"

그 요구를 채우는 기능은
어떤 모습인가

매일 하는 일이라 익숙한 방향

AI 와의 분담

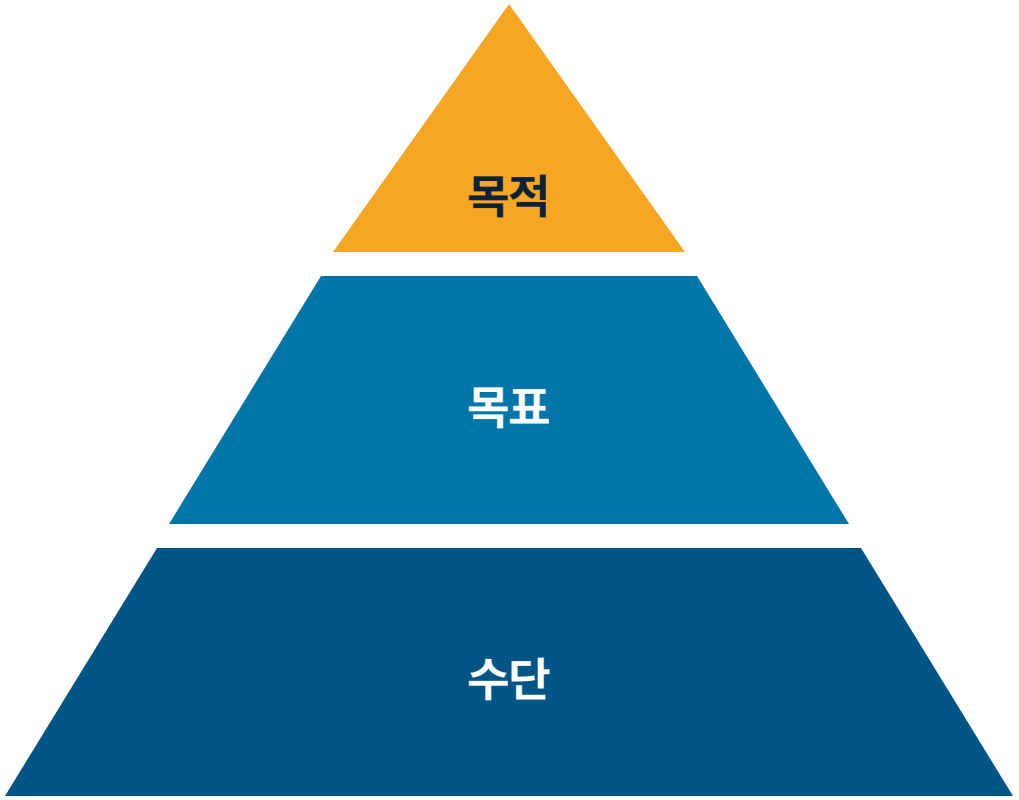
요건을 코드로 옮기는
구체화는 AI 가 빠르게
처리하는 영역

목적을 기준으로 위아래를 오가야 구현과 요구가 한 줄로 이어짐

출처: 호소야 이사오, 구체와 추상(원서 dZERO, 2014)의 추상화 개념을 요지로 정리. 계층 예시는 설명용

목적과 목표, 수단의 구분

KEY INSIGHT 목표가 없으면 AI 결과물이 맞는지 판단할 기준이 없음 — 요건과 인수 기준이 AI 에 주는 합격선 역할을 함



도달하려는 상태 — 요구 정의

고객 요구

예: 주문 수량을 잘못 넣는 일을 막고 싶음

목적을 이뤘는지 판단하는 조건 — AI 에 주는 합격선

기능·비기능 요건, 인수 기준

예: 주문 수량은 1~200 사이만 허용

목적을 이루는 방법·도구 — AI 가 빠르게 만듦

소스 코드

예: 수량 검증 로직과 테스트 코드

셋이 맞물려야 노력이 결과로 이어짐 — 목표가 빠지면 끝났는지 판단할 수 없음

출처: ISO/IEC/IEEE 29148:2018 요구사항 정의 개념을 바탕으로 정리. 주문 수량 예는 설명용

ISO/IEC/IEEE 29148 요구사항 계층

KEY INSIGHT 국내 RFP 의 기능 요구사항은 맨 아래 층에 해당함 — 위 층의 사업 목적을 모르면 항목을 다 채우고도 사업 목표를 놓침

표준 문서	ISO/IEC/IEEE 29148 의 층	국내 공공 SW 사업에서 부르는 말
BRS	Business Requirements Specification **비즈니스 요구**	사업 목적·성과 지표 (제안요청서 사업 개요)
StRS	Stakeholder Requirements Specification **이해관계자 요구**	현업·사용자 요구 (인터뷰·요구사항 협의)
SyRS	System Requirements Specification **시스템 요구**	시스템이 갖출 기능·품질 (요구사항 정의서)
SRS	Software Requirements Specification **소프트웨어 요구**	**기능 요구사항(SFR) · 성능(PER) · 보안(SER) 등** **제안요청서 요구사항 항목은 이 층**

출처: ISO/IEC/IEEE 29148:2018 Requirements engineering(요구사항 문서 구성) · 국내 공공 SW 사업 제안요청서 요구사항 분류 관행

포인트 적립 기능의 상위 목적 찾기

KEY INSIGHT 포인트는 재구매를 이루는 수단 중 하나임 — 쿠폰·재구매 알림과 비교해 더 나은 방법을 고를 수 있게 됨

즉, 무엇을 위해?

4 사업 목적
재구매율을 높여 고객 이탈을 줄임

3 사용자 목적
고객이 혜택을 받고 다시 방문

2 기능 목적
구매 금액만큼 포인트를 적립하고 결제에 사용

1 기술 목적 · 엔지니어가 처음 말하는 층
포인트 적립·사용 내역을 DB 에 저장

→

재구매율을 높이는 수단 후보

- **포인트 적립** — 지금 만드는 것
- 재방문 쿠폰 발행
- 장바구니 재구매 알림
- 멤버십 등급 혜택

비용·효과를 같은 목적 기준으로 비교 가능

출처: 설명용 예시(국내 커머스 포인트 서비스 가정)

같은 낱말도 문맥마다 뜻이 다름

KEY INSIGHT 팀마다 다른 뜻을 용어 사전으로 맞춰야 요구가 정확히 전달됨 — AI 에 주는 지시문도 같은 용어를 써야 결과가 맞음



언어 게임 · 비트겐슈타인

낱말의 뜻은 그 말이 쓰이는 방식이 정함

철학적 탐구(1953) §43

유비쿼터스 언어 · DDD

개발자와 현업 전문가가 함께 쓰는 공통 언어

에릭 에반스, 도메인 주도 설계(2003)

팀 용어 사전 예시

용어	쓰는 문맥	뜻	업무 규칙
주문	물류	창고에 내리는 출고 지시	결제 완료 뒤에만 생성
주문	회계	매출로 잡는 거래	출고 완료 시점에 매출 인식

출처: L. 비트겐슈타인, 철학적 탐구(1953, 한국어판 아카넷 2016) · E. Evans, 도메인 주도 설계(2003, 한국어판 위키북스 2011). 주문 예시는 설명용



04

PART 04

아키텍트 역량과 기술 기본기

품질 특성을 고르고 구조를 정하는 판단과 그 바탕이 되는 기술력

22–23 PM·아키텍트 책임과 ISO/IEC 25010:2023

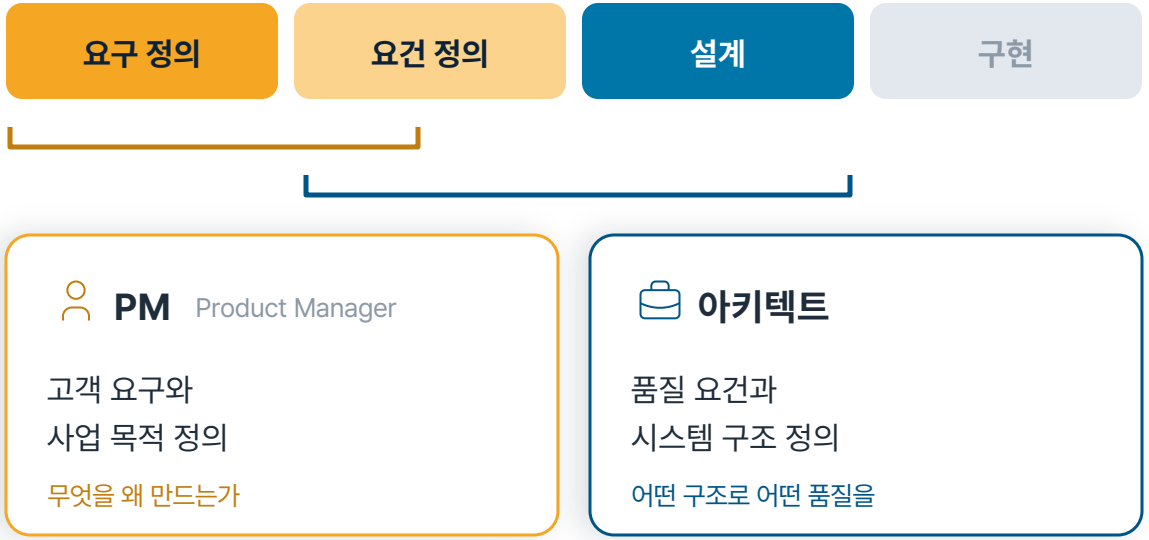
24–25 아키텍처 선택과 품질 특성 트레이드오프

26–27 품질 판단에 필요한 기술 지식과 개발팀 점검표

28 AI 활용의 전제

PM 과 아키텍트의 책임 범위

KEY INSIGHT 둘 다 제품 가치가 목적이고 책임지는 층이 다름 — 기술에서 출발한 엔지니어에게는 아키텍트가 가까운 경로임



구현을 AI 가 맡을수록 두 직무의 판단이
엔지니어의 일상 업무로 내려옴

아키텍처란 무엇인가

> **Michael Keeling, Design It! (2017)**
>
> **소프트웨어를 어떻게 구성할지에 관한 중요한 설계 결정의 묶음**
>
> 한국어판: 개발자에서 아키텍트로(한빛미디어, 2021)

> **Bass 외, Software Architecture in Practice 4판**
>
> **시스템을 따져 보는 데 필요한 구조의 묶음**
>
> 한국어판: 소프트웨어 아키텍처 이론과 실제 4/e(에이콘, 2022)

출처: M. Keeling, Design It!(Pragmatic Bookshelf, 2017) · L. Bass·P. Clements·R. Kazman, Software Architecture in Practice 4판(2021). 정의는 요지 번역

ISO/IEC 25010:2023 품질 특성 9가지

KEY INSIGHT 기능 외 여덟 특성이 품질을 좌우함 — AI 에 기능만 지시하면 나머지 특성은 아무도 정하지 않은 채 남음

2011판 (8개)	2023판 (9개)	변화
기능 적합성	기능 적합성	유지
성능 효율성	성능 효율성	유지
호환성	호환성	유지
사용성	**상호작용 능력 (Interaction capability)**	**이름·범위 변경**
신뢰성	신뢰성	유지
보안성	보안성	유지
유지보수성	유지보수성	유지
이식성	**유연성 (Flexibility)**	**이름·범위 변경**
—	**안전성 (Safety)**	**신설**

AI 지시에 넣을 품질 요건 예

성능 효율성
피크 시 응답 2초 이내

보안성
개인정보 칼럼은 암호화 저장

유지보수성
모듈 사이 순환 의존 없음

상호작용 능력
키보드만으로 주문 완료 가능

요건 예시는 설명용

출처: ISO/IEC 25010:2023 Systems and software engineering — Product quality model(2판, 2023.11, iso.org/standard/78176)

아키텍처 선택과 품질 특성의 관계

KEY INSIGHT 어떤 구조도 모든 특성을 함께 올리지 못함 — MSA 는 변경 용이성을 얻는 대신 공격 표면과 운영 부담이 늘어남

아키텍처 선택	신뢰성	유연성	유지보수성	보안성	성능 효율성
클라우드 네이티브 컨테이너·쿠버네티스 기반	● 가용성	● 확장성			
MSA 서비스 단위로 나눠 배포		● 독립 배포	● 변경 용이성	▲ 공격 표면 증가	▲ 호출 지연·운영 부담
제로 트러스트 절대 믿지 말고 계속 검증				● 접근 통제	
관측성 APM·OpenTelemetry	● 장애 원인 추적				● 병목 측정

● 높이는 특성 ▲ 부담이 늘어나는 특성 빈칸: 직접 관계 적음

클라우드 네이티브는 컨테이너 전환이 기본, 가상화는 예외

출처: 과기정통부·KISA “제로트러스트 가이드라인 2.0”(2024.12) · MSAP.ai “MSA 설계 전략” · Cloud Native Forum “클라우드 네이티브 보안”(29장 URL)

품질 특성 트레이드오프 판단

KEY INSIGHT 제품 가치를 기준으로 어느 특성에 비용을 쓸지 정하는 일이 아키텍트의 판단임 — AI 가 대신 정할 수 없는 결정임

제약 개발 자원은 유한해 모든 특성을 최대로 만들 수 없음 · 품질 특성끼리 서로 부딪침

ATAM 으로 판단하는 순서

1. **품질 시나리오 정의**
   예: 피크 시 응답
   2초 이내
2. **아키텍처 방식 분석**
   후보 구조가 시나리오를
   만족하는지 확인
3. **트레이드오프점 식별**
   한 특성을 올리면
   다른 특성이 내려가는 지점
4. **위험 정리·결정 기록**
   무엇을 취하고 버렸는지
   근거와 함께 남김

예시 · 결제 서비스

보안성(결제마다 추가 인증) ↔ 상호작용 능력(결제 단계 수)

결정: 고액 결제에만 추가 인증 — 이탈률과 사고 위험을 같은 제품 가치 기준으로 비교

출처: SEI "The Architecture Tradeoff Analysis Method"(CMU/SEI-98-TR-008) · L. Bass 외, Software Architecture in Practice 4판(2021). 결제 예시는 설명용

품질 판단에 필요한 기술 지식

KEY INSIGHT 좋은 구조를 아는 사람만 AI 코드의 문제를 찾아냄 — 기술을 내려놓지 않고 그 위에 질문하는 사고를 더해야 함

리팩터링

겉보기 동작은 그대로 두고
내부 구조를 이해하기 쉽고
고치기 쉽게 바꾸는 일

AI 결과물 리뷰 활용

중복·긴 함수 등 구조 문제 판별

테스트

요건과 인수 기준을
실행할 수 있는 검사로
바꿔 두는 일

AI 결과물 리뷰 활용

요건 충족 여부를 자동으로 검증

DDD

업무 개념을 도메인 모델과
유비쿼터스 언어로
코드에 옮기는 설계 방법

AI 결과물 리뷰 활용

지시문과 코드의 용어 일치 확인

DORA 가 꼽은 AI 효과를 키우는 역량 7가지

명확한 AI 활용 방침 | 건강한 데이터 환경 | AI 가 접근 가능한 내부 데이터 | **탄탄한 버전 관리** | **작은 단위로 나눠 작업**

사용자 중심 | 내부 플랫폼 품질

진한 칩은 개발 기본기에 가까운 항목

출처: M. Fowler “DefinitionOfRefactoring”(martinfowler.com) · E. Evans, 도메인 주도 설계(2003) · Google Cloud “DORA AI Capabilities Model”(2025.9)

개발팀 문제 정의 점검표

KEY INSIGHT 기존 산출물에 항목을 더하는 것으로 시작 가능함 — 요구사항 정의서와 리뷰 기준부터 바꾸면 AI 결과물 품질이 함께 올라감

요구사항 정의서에 "목적" 열 추가

항목마다 상위 목적을 한 줄로 적음

요구사항 정의서

기능 요구사항마다 인수 기준 작성

AI 결과물의 합격선으로 사용

요구사항 정의서

비기능 요구사항을 9개 특성으로 점검

ISO/IEC 25010:2023 기준으로 빠진 특성 확인

비기능 요구사항

팀 용어 사전 운영

AI 지시문에도 같은 용어를 사용

용어 사전

AI 코드 리뷰에 "요건 대비 확인" 추가

동작 확인에 더해 요건 충족 여부까지 확인

리뷰 체크리스트

주니어 개발자를 요구 분석 회의에 참여시킴

구현보다 요구 이해 경험을 먼저 쌓게 함

팀 운영

새 절차를 만들지 않고 기존 산출물에 항목을 더하는 수준

출처: 앞 장 근거 종합(ISO/IEC/IEEE 29148:2018 · ISO/IEC 25010:2023 · DORA 2025 · 한국은행·SPRi 초급 개발자 분석)

AI 활용의 전제는 요구 정의와 기본기

KEY INSIGHT AI 는 팀의 강점과 약점을 함께 키움 — 기본기를 다진 팀일수록 AI 로 얻는 이득이 커짐

기본기 없이 AI 로 속도만 올리면
배포 불안정이 먼저 커짐

01

요구 정의

상위 목적과 인수 기준을
사람이 정함

16–20장

02

품질 판단

우선할 품질 특성과
구조를 사람이 고름

22–25장

03

기술 기본기

리팩터링·테스트·도메인 모델로
AI 결과물을 검증

26–27장

출처: Google Cloud “Announcing the 2025 DORA Report”(2025.9) — AI 도입과 배포 안정성의 음의 관계, 증폭 효과

REFERENCES

참고 자료

KEY INSIGHT 수치는 조사마다 대상과 질문이 다름 — 인용할 때는 발행처와 조사 대상을 함께 밝힘

국내 공공·연구기관

과기정통부·KISDI, 2025 디지털산업 실태조사
2026.7 · 04장

대한상공회의소, 기업의 AI 전환 실태와 개선방안
2025.11, 보도 인용 · 04장

한국은행, BOK 이슈노트 제2025-30호·제2026-19호
2025.10·2026.8 · 09장

SPRi, IS-179 · RE-192
2024.8·2025.4 · 07–09장

과기정통부, 공공SW사업 과업변경 가이드
2020.4 보도자료 · 13장

소프트웨어 진흥법 제44조
2020.12 시행 · 13장

과기정통부·KISA, 제로트러스트 가이드라인 2.0
2024.12 · 24장

해외 조사·국제 표준

Stack Overflow, Developer Survey 2025
2025.7 · 05장

Google Cloud, 2025 DORA Report
AI Capabilities Model 포함, 2025.9 · 06·26·28장

METR, Early-2025 AI 개발자 생산성 실험
2025.7, 2026.2 갱신 글 · 06장

PMI, Pulse of the Profession
Requirements Management, 2014 · 13장

ISO/IEC 25010:2023 Product quality model
23·27장

ISO/IEC/IEEE 29148:2018 Requirements engineering
17·18장

SEI, Architecture Tradeoff Analysis Method
CMU/SEI-98-TR-008 · 25장

도서·공개 글

F. C. Bartlett, Remembering(1932)

L. 비트겐슈타인, 철학적 탐구(1953)

E. Evans, 도메인 주도 설계(2003)

M. Fowler, 리팩터링 2판(2018)

M. Keeling, Design It!(2017)

L. Bass 외, Software Architecture in Practice 4판

호소야 이사오, 구체와 추상(원서 2014)

msap.ai/msa/msa-design-strategy
cncf.co.kr/cloud-native/cloud-native-security
cncf.co.kr/ai/ai-era-developer-survival-strategy

착안: MinoDriven, AI 시대의 커리어 플랜(TECH PLAY, 2026.2). 이 자료의 수치·사례·해석은 위 자료를 근거로 새로 정리함

출처: 각 장 바닥글에 발행처·자료명·연도를 표기

Q & A

질의응답

AI 시대 국내 개발자의 요구 정의·아키텍처 역량과 커리어 방향 논의

MSAP.ai

msap.ai

홈페이지 https://www.msap.ai/

메일 hello@msap.ai 전화 02-6953-5427

© MSAP.ai. All rights reserved.