M12-01 · GIBALJA VISUAL MANUAL

사용자 문제와 기대 결과 정의하기

한 문장 목표: `user·situation·current alternative → evidence → problem → desired outcome·baseline·target·guardrail → uncertainty·next test → Problem Definition Gate` 를 연결해, 해결책을 만들기 전에 왜 이 문제를 다뤄야 하는지 설명합니다.

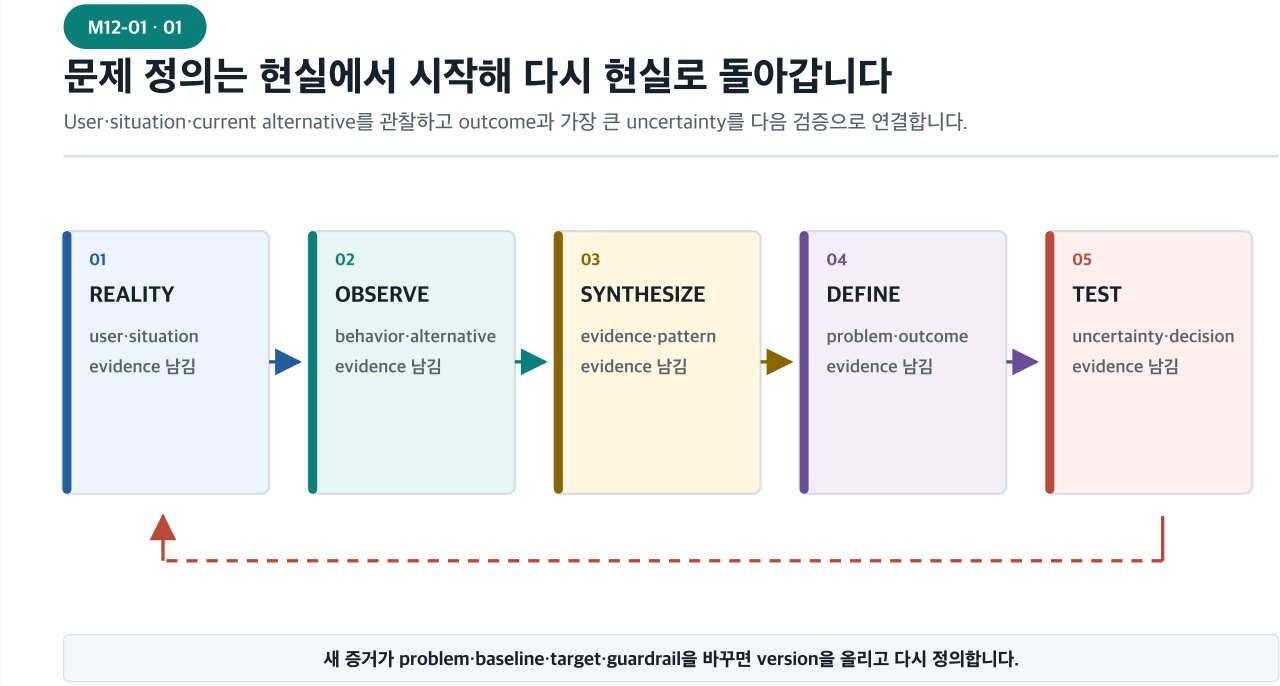
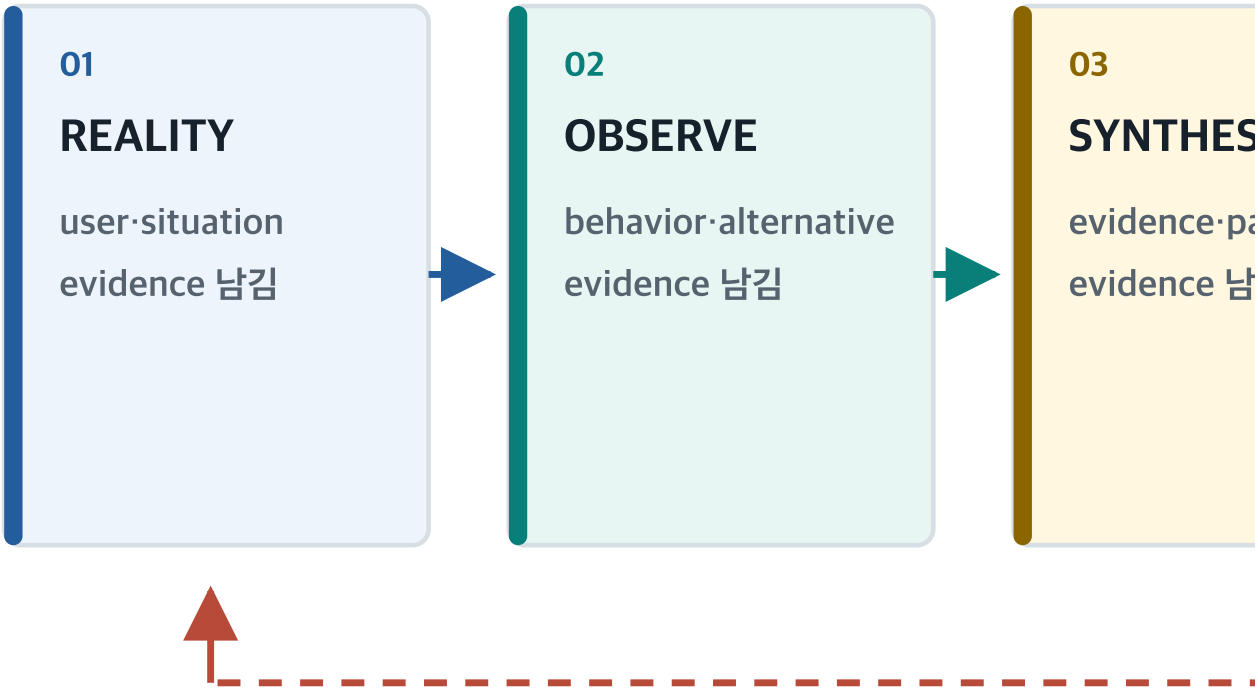
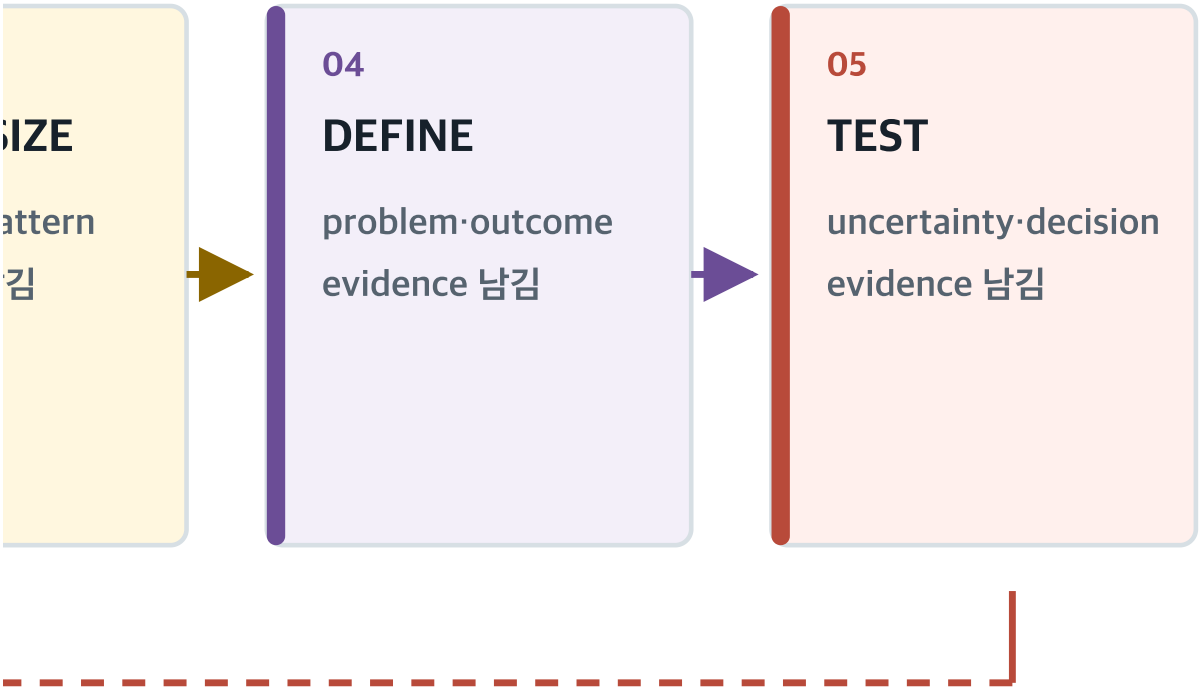


그림 1. 문제 정의는 문서 한 번이 아니라 현실을 관찰하고 증거를 합성해 문제·결과를 정의한 뒤, 가장 위험한 불확실성을 검증하며 version을 올리는 순환입니다.

User·situation·current alternative를 관찰하고 outcome과 가장 큰 uncer



rtainty를 다음 검증으로 연결합니다.



0. 한눈에 보는 두 번째 지도

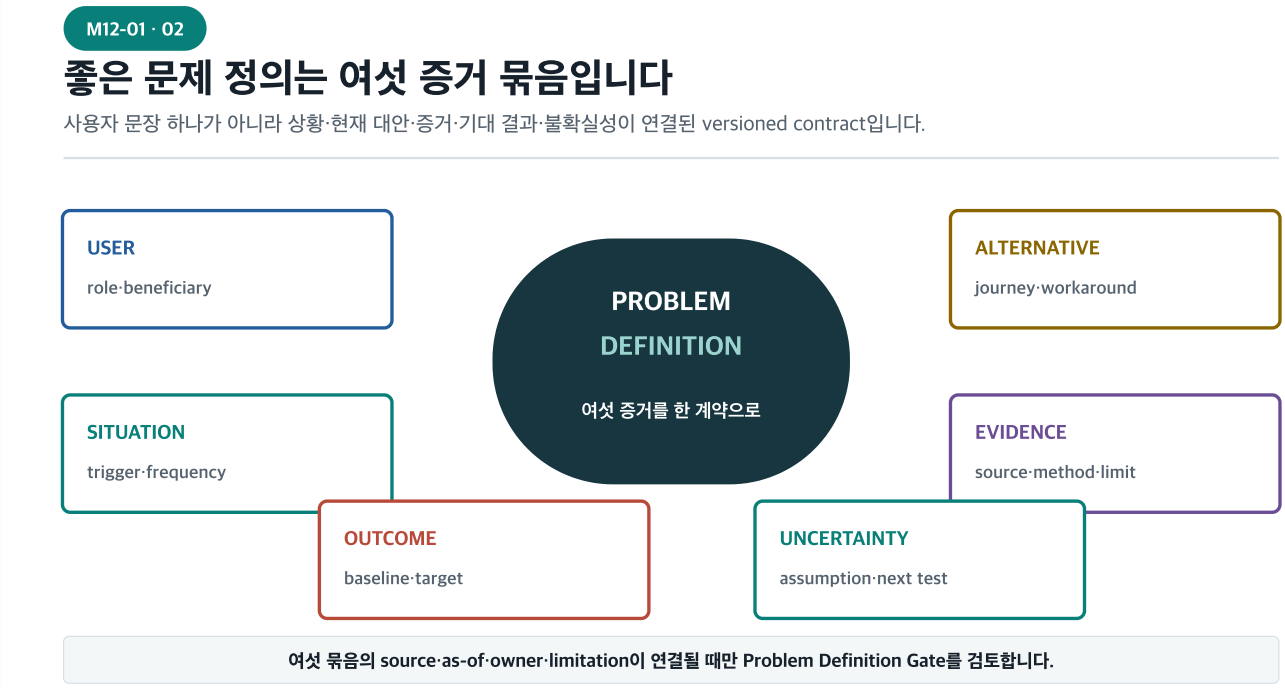
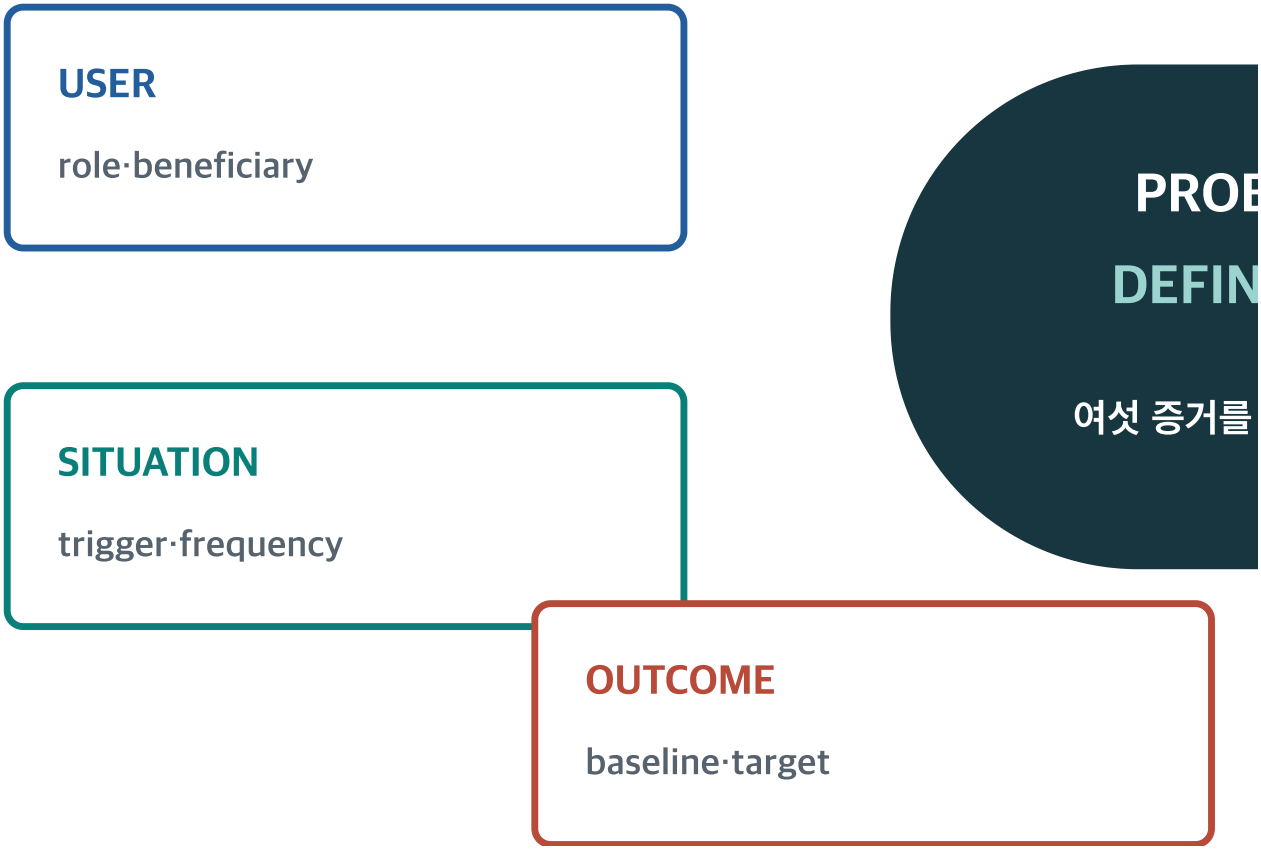


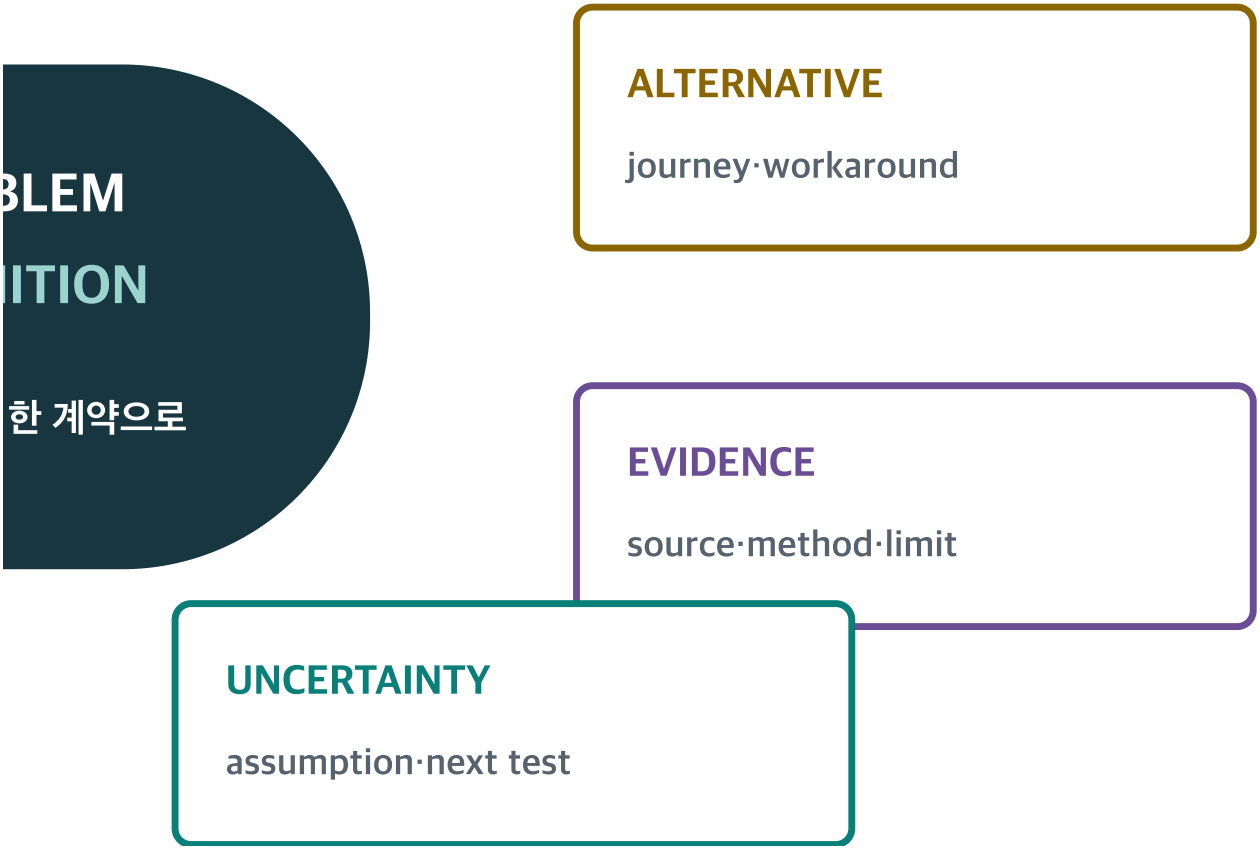
그림 2. 좋은 문제 정의는 멋진 한 문장이 아니라 여섯 증거 묶음입니다. 각 묶음에 source·as-of·owner·limitation이 있어야 Gate에서 다시 확인할 수 있습니다.

[illegible]

사용자 문장 하나가 아니라 상황·현재 대안·증거·기대 결과·불확실성이 연결



결된 versioned contract입니다.



학습 순서	시간	무엇을 남기나
그림 16장	40분	문제 정의 전체 지도
개념·사례	80분	사용자·상황·증거·결과 계약
실습 스튜디오	90분	24 scenario·Problem Gate
셀프 테스트	30분	30문항·모범 답안

> 본문의 사용자·세션·수치·인용·비용은 전부 허구의 합성 학습 자료입니다. 실제 참여자 연구, 개인정보·녹음·동의 기록, 통계적 일반화, 법률·윤리·정책 승인, 제품 구축 결정을 대신하지 않습니다.

0.1 첫 두 장에서 기억할 열두 문장

```
요청한 사람과 실제 사용자는 다를 수 있다.
문제는 사람의 성격이 아니라 특정 상황의 마찰이다.
현재 대안은 경쟁자이자 문제 증거다.
관찰·인용·해석·의견은 다른 증거다.
숫자에는 분모·출처·시점·한계가 필요하다.
증상·문제·원인·해결책을 한 문장에 섞지 않는다.
기능은 output이고 기대 결과는 사용자 변화다.
Target은 baseline과 기간이 있어야 한다.
평균 개선은 guardrail을 통과해야 한다.
Fact·inference·assumption·unknown은 서로 다른 행동을 요구한다.
다음 연구는 가장 영향이 크고 불확실한 가정부터 한다.
problem_ready는 제품을 만들라는 승인이 아니다.
```

1. 이 PDF를 공부하는 방법

1.1 1회차 · 그림과 caption만 읽기 · 40분

그림 제목을 따라 `현실 → 증거 → 문제 → 결과 → 불확실성 → Gate`를 소리 내어 설명합니다. 각 그림마다 다음 빈칸 하나만 채웁니다.

> 이 그림에서 내가 아직 가정으로 두고 있는 것은 _____이고, 틀리면 바뀌는 결정은 _____이다.

1.2 2회차 · 합성 사례로 손을 움직이기 · 90분

문제 정의 실습 생성기를 실행합니다.

```
./02_Labs/G12_Product_Definition/L12-01_create-problem-definition-practice.sh
```

세 version을 비교합니다.

Version	Pass	Solution term	Guardrail	Decision
stakeholder-request-v1	6/24	1	0	blocked_solution_request
evidence-without-outcome-v2	16/24	0	2	blocked_outcome_ambiguity
verified-problem-definition-v3	24/24	0	5	problem_ready

1.3 3회차 · 내 문제 정의 한 장 만들기 · 110분

사용자 문제·기대 결과 정의 템플릿을 다음 순서로 채웁니다.

1. 역할과 구체 상황
2. 현재 journey와 대안
3. evidence ledger와 문제 규모
4. problem·cause·solution 가설 분리
5. desired outcome과 outcome contract
6. guardrail과 M11-04 제약
7. assumption·unknown·next test
8. Problem Definition Gate와 M12-02 handoff

1.4 막힐 때

사용자 문제·기대 결과 용어집 300에서 지금 장과 같은 번호의 20개만 읽습니다. 300개를 먼저 외우지 않습니다.

2. 문제 정의를 다시 정의하기

2.1 문제 정의는 무엇인가

PROBLEM DEFINITION 특정 사용자가 구체 상황에서 현재 방법으로 목표를 달성하려다 반복해 겪는 마찰과 영향을, 추적 가능한 증거로 설명하고 기대 결과·측정·guardrail·불확실성·다음 결정을 연결한 versioned contract입니다.

2.2 비슷해 보이지만 다른 여섯 문장

문장	실제로 말하는 것	아직 빠진 것
고객이 불편해한다	평가	누가·언제·무엇을 하다
회의 기록이 누락된다	증상	상황·현재 행동·영향
채널 전환 때문에 누락된다	원인 가설	대안 원인·검증
AI 봇이 필요하다	해결책 가설	문제와 결과
완전 기록률을 높인다	결과 방향	baseline·target·기간
4주 안에 58%에서 85%	target	guardrail·source·owner

2.3 M07-02·M10-01과 다른 점

M07-02는 이미 선택한 요구사항을 작은 작업과 완료 기준으로 나눴고, M10-01은 요구사항을 scenario와 test case로 바꿨습니다. M12-01은 그보다 앞에서 **어떤 사용자 문제와 결과를 다뤄야 하는지**를 증거로 확인합니다. 아직 상세 기능 요구사항을 쓰지 않습니다.

2.4 최소 문제 정의 계약

```
user + beneficiary + situation + trigger + frequency
+ current journey + alternative + barrier + consequence
+ evidence(source·method·sample·as-of·consent·limitation)
→ anti-solution problem statement
→ desired outcome + baseline + target + period + measure
+ quality·safety·privacy·access·time·cost guardrails
+ fact·inference·assumption·unknown + next test + owner
→ Problem Definition Gate + M12-02 handoff
```

2.5 합성 사례의 경계

이 매뉴얼의 예시는 `5~20명 팀의 운영 조정 담당자가 회의 직후 결정·담당·기한을 여러 채널에 옮기는 상황`입니다. 36개 관찰 세션, 18명 likely user, 6개 evidence source, 4개 user group, baseline 58%와 14분은 모두 허구의 고정값입니다.

---

3. 먼저 넓게 발견하고 증거로 좁히기

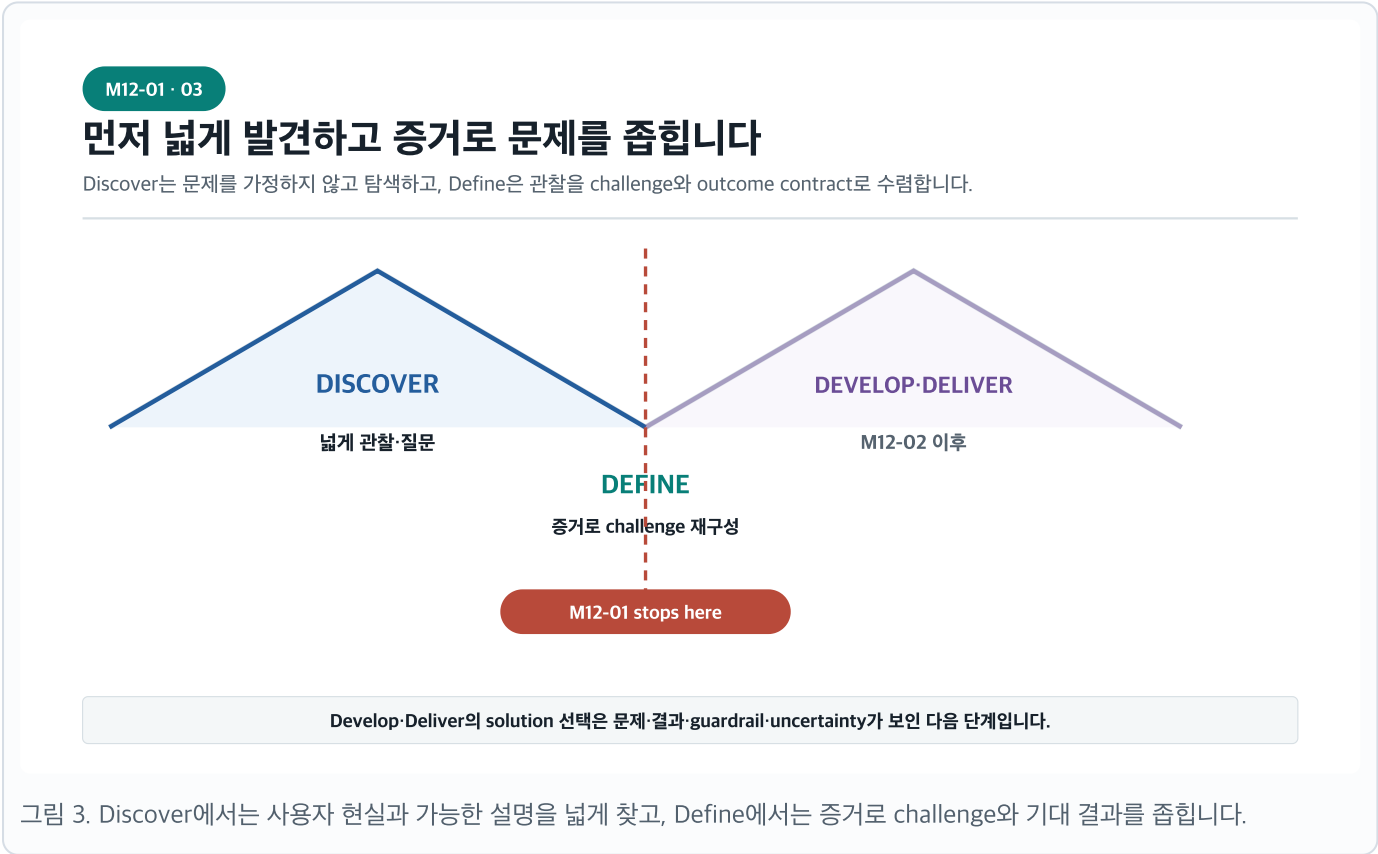


그림 3. Discover에서는 사용자 현실과 가능한 설명을 넓게 찾고, Define에서는 증거로 challenge와 기대 결과를 좁힙니다.

Discover는 문제를 가정하지 않고 탐색하고, Define은 관찰을 challenge와

· outcome contract로 수렴합니다.

재구성

here

3.1 그림을 읽는 질문

Discover에서는 사용자 현실과 가능한 설명을 넓게 찾고, Define에서는 증거로 challenge와 기대 결과를 좁힙니다. 아래 표를 읽고 각 행에서 `evidence → decision` 연결을 말해 봅니다.

요소	합성 사례	확인할 것
Discover	사용자를 정해 놓지 않고 actual·likely user를 탐색	관찰·질문·대안·반대 사례
Define	패턴과 차이를 근거로 문제 경계를 수렴	문제 문장·outcome·guardrail·uncertainty
Develop 이후	해결책 후보와 요구사항을 비교	M12-02 이후 수행

3.2 흔한 실패

- 한 사람의 강한 요청을 여러 사용자의 행동 증거처럼 씁니다.
- 맥락이나 분모를 지우고 인용·평균 숫자 하나만 남깁니다.
- 아직 확인하지 않은 원인과 해결책을 fact처럼 씁니다.
- 누가 어떤 증거를 언제 다시 확인할지 적지 않습니다.

3.3 3분 연습

1. 내 사례에서 이 그림의 빈칸 하나를 고릅니다.
2. 지금 아는 내용을 fact와 inference로 분리합니다.
3. 필요한 source·method·owner를 한 줄로 씁니다.
4. 답이 나오면 바뀌는 continue·reframe·stop 결정을 표시합니다.

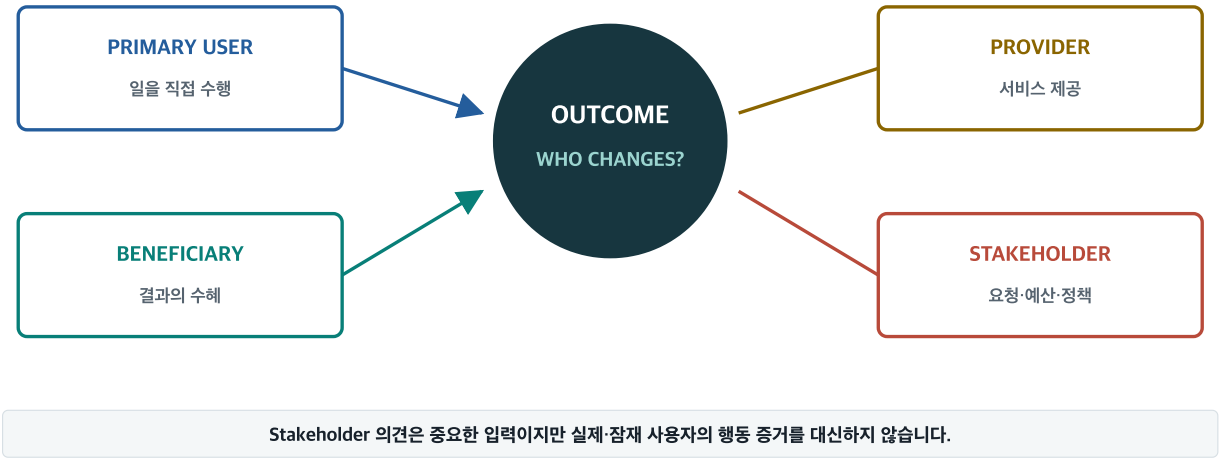
> 좋은 문장은 길어서가 아니라 source·상황·행동·결과·불확실성이 분리되어 다시 검토할 수 있어서 강합니다.

4. 사용자·수혜자·이해관계자 구분하기

M12-01 · 04

요청한 사람과 사용하는 사람과 혜택받는 사람은 다를 수 있습니다

Primary user·beneficiary·provider·supporter·stakeholder의 역할과 서로 다른 성공 결과를 분리합니다.

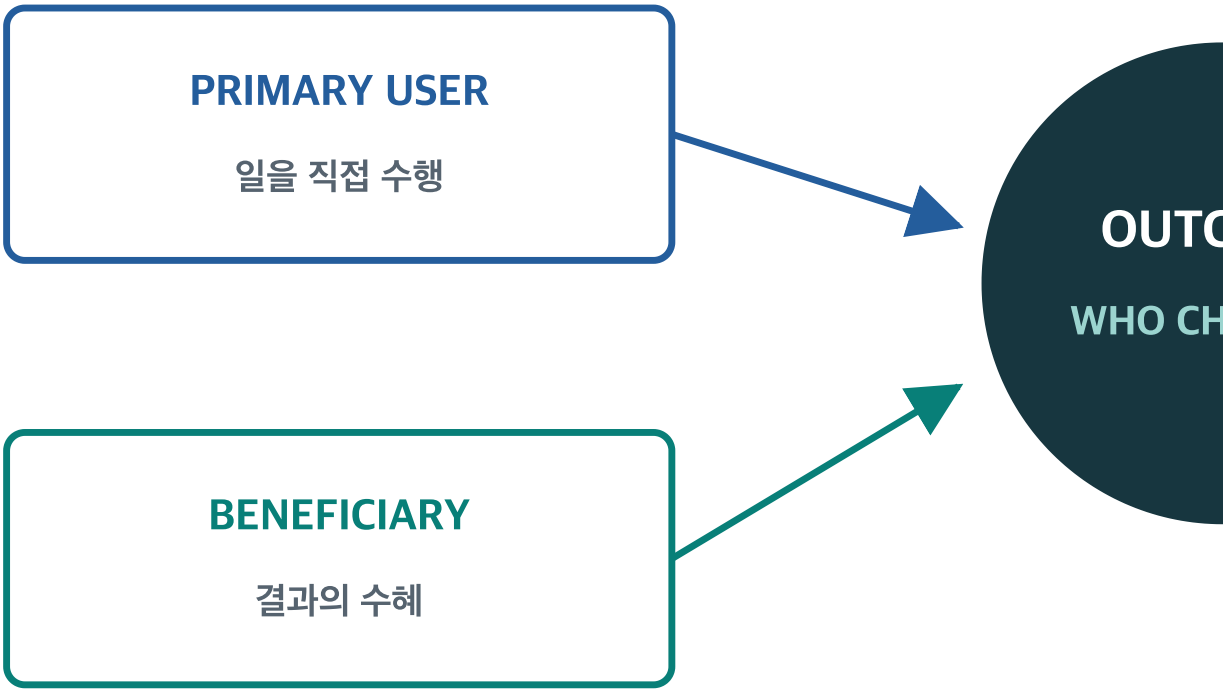


Stakeholder 의견은 중요한 입력이지만 실제·잠재 사용자의 행동 증거를 대신하지 않습니다.

그림 4. 요청한 사람, 사용하는 사람, 혜택받는 사람, 제공하는 사람의 성공 기준을 각자 적습니다.

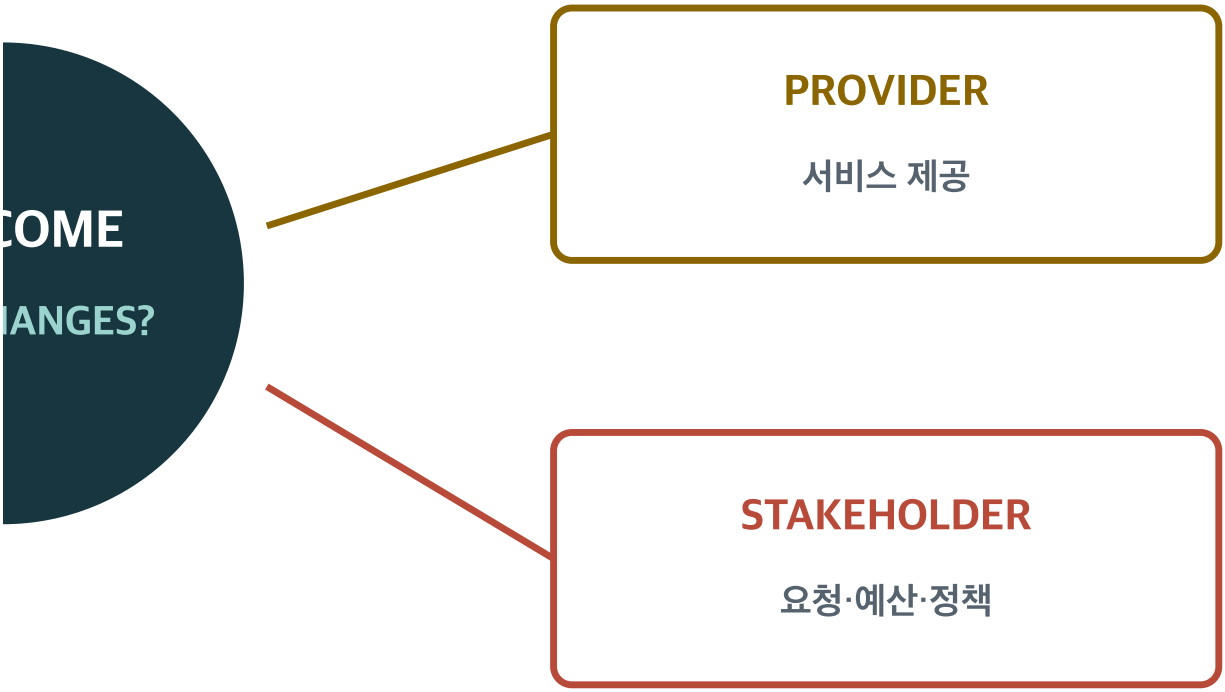
[illegible]

Primary user·beneficiary·provider·supporter·stakeholder의 역할과 서로



른 사람은 다른 무엇을 원하는가

를 다른 성공 결과를 분리합니다.



4.1 그림을 읽는 질문

요청한 사람, 사용하는 사람, 혜택받는 사람, 제공하는 사람의 성공 기준을 각자 적습니다. 아래 표를 읽고 각 행에서 `evidence → decision` 연결을 말해 봅니다.

요소	합성 사례	확인할 것
Primary user	결정·담당·기한을 실행 기록으로 옮기는 운영 조정 담당자	작업 성공·시간·통제
Beneficiary	후속 조치를 실행하고 상태를 확인하는 팀원	정확한 책임·기한
Provider	업무 흐름과 지원을 운영하는 담당자	지원 부담·복구 가능성
Stakeholder	예산·정책·성과를 책임지는 요청자	비용·위험·조직 결과

4.2 흔한 실패

- 한 사람의 강한 요청을 여러 사용자의 행동 증거처럼 씁니다.
- 맥락이나 분모를 지우고 인용·평균 숫자 하나만 남깁니다.
- 아직 확인하지 않은 원인과 해결책을 fact처럼 씁니다.
- 누가 어떤 증거를 언제 다시 확인할지 적지 않습니다.

4.3 3분 연습

1. 내 사례에서 이 그림의 빈칸 하나를 고릅니다.
2. 지금 아는 내용을 fact와 inference로 분리합니다.
3. 필요한 source·method·owner를 한 줄로 씁니다.
4. 답이 나오면 바뀌는 continue·reframe·stop 결정을 표시합니다.

좋은 문장은 길어서가 아니라 source·상황·행동·결과·불확실성이 분리되어 다시 검토할 수 있어서 강합니다.

5. 상황·시작 사건·장벽을 한 장면으로 쓰기

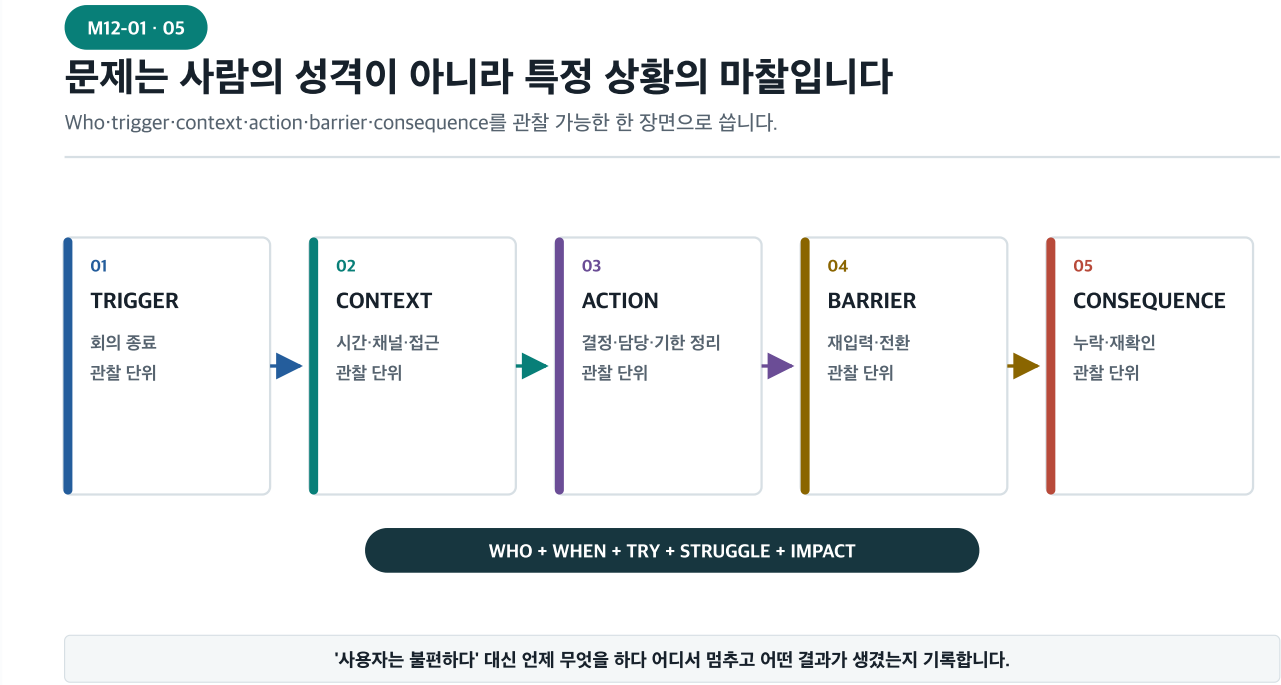


그림 5. 사용자의 성격이 아니라 언제 무엇을 하다가 어디서 막히고 어떤 결과가 생기는지 씁니다.

[illegible]

Who·trigger·context·action·barrier·consequence를 관찰 가능한 한 장면

WHO + WHEN + TRY +

[illegible]으로 씁니다.

- STRUGGLE + IMPACT

5.1 그림을 읽는 질문

사용자의 성격이 아니라 언제 무엇을 하다가 어디서 막히고 어떤 결과가 생기는지 씁니다. 아래 표를 읽고 각 행에서 `evidence → decision` 연결을 말해 봅니다.

요소	합성 사례	확인할 것
Who	5~20명 팀의 운영 조정 담당자	역할과 실제·잠재 사용 여부
Trigger	주 3회 이상 회의가 끝난 직후	행동을 시작시키는 사건
Trying	결정·담당·기한을 실행 가능한 기록으로 남김	사용자가 하려는 일
Barrier	채팅·표·개인 메모 사이 재입력	관찰 가능한 마찰
Consequence	담당 누락·재확인·실행 지연	사용자와 팀에 생긴 영향

5.2 흔한 실패

- 한 사람의 강한 요청을 여러 사용자의 행동 증거처럼 씁니다.
- 맥락이나 분모를 지우고 인용·평균 숫자 하나만 남깁니다.
- 아직 확인하지 않은 원인과 해결책을 fact처럼 씁니다.
- 누가 어떤 증거를 언제 다시 확인할지 적지 않습니다.

5.3 3분 연습

1. 내 사례에서 이 그림의 빈칸 하나를 고릅니다.
2. 지금 아는 내용을 fact와 inference로 분리합니다.
3. 필요한 source·method·owner를 한 줄로 씁니다.
4. 답이 나오면 바뀌는 continue·reframe·stop 결정을 표시합니다.

> 좋은 문장은 길어서가 아니라 source·상황·행동·결과·불확실성이 분리되어 다시 검토할 수 있어서 강합니다.

6. 현재 여정과 대안에서 문제 찾기

그림 6. 화면 하나가 아니라 before·during·after, online·offline·support, 사람·채널·증거를 end-to-end로 봅니다.

[illegible]

Before·during·after의 online·offline·support handoff와 workaround를

	BEFORE · 회의 준비	
PERSON 담당자·팀원	agenda 확인	
CHANNEL 회의·채팅·표·메모	calendar·chat	
WORK 검색·전환·재입력	자료 찾기	
EVIDENCE 시간·오류·지원	준비 시간	

end-to-end로 펼칩니다.

DURING · 결정 발생	AFTER · 실행 기록
결정 메모	담당 재확인
회의·개인 note	chat·sheet
기록 분산	14분 재입력
누락 관찰	58% 완전 기록

6.1 그림을 읽는 질문

화면 하나가 아니라 before·during·after, online·offline·support, 사람·채널·증거를 end-to-end로 봅니다. 아래 표를 읽고 각 행에서 `evidence → decision` 연결을 말해 봅니다.

요소	합성 사례	확인할 것
Before	회의 준비와 이전 결정 검색	calendar·chat·문서
During	결정과 담당 후보를 개인 메모에 남김	회의·구두 확인
After	채팅과 표에 재입력하고 담당을 재확인	14분·58% complete
Recovery	누락 발견 뒤 사람에게 다시 묻고 수정	지연·방해·신뢰 하락

6.2 흔한 실패

- 한 사람의 강한 요청을 여러 사용자의 행동 증거처럼 씁니다.
- 맥락이나 분모를 지우고 인용·평균 숫자 하나만 남깁니다.
- 아직 확인하지 않은 원인과 해결책을 fact처럼 씁니다.
- 누가 어떤 증거를 언제 다시 확인할지 적지 않습니다.

6.3 3분 연습

1. 내 사례에서 이 그림의 빈칸 하나를 고릅니다.
2. 지금 아는 내용을 fact와 inference로 분리합니다.
3. 필요한 source·method·owner를 한 줄로 씁니다.
4. 답이 나오면 바뀌는 continue·reframe·stop 결정을 표시합니다.

> 좋은 문장은 길어서가 아니라 source·상황·행동·결과·불확실성이 분리되어 다시 검토할 수 있어서 강합니다.

7. 참여자를 존중하는 연구 경계

7.1 동의는 서명 한 번이 아닙니다

참여자는 연구 목적, 하게 될 활동, 기록 여부, 자료 사용·공유·보관 범위, 예상 위험, 철회 방법을 이해해야 합니다. 연구팀은 필요한 정보만 수집하고 동의한 범위 밖으로 재사용하지 않습니다.

7.2 합성 실습과 실제 연구의 경계

합성 실습에서 하는 것	실제 연구에서 추가로 필요한 것
허구의 사용자·수치·인용	모집 기준·참여자 보호·조직 승인
metadata-only trace	원자료 접근 통제·보관·삭제 계획
consent 필드 모형	실제 informed consent와 철회 절차
접근 필요 시나리오	필요한 조정과 포용적 모집
학습용 Gate	법률·윤리·정책·제품 책임자의 판단

7.3 연구 전 최소 질문

1. 이 질문에 사람의 개인 경험이 꼭 필요한가?
2. 덜 민감한 자료나 기존 증거로 답할 수 있는가?
3. 누가 참여에서 배제되기 쉬운가?
4. 참여 중 불편·압박·피해가 생기면 어떻게 멈출 것인가?
5. 무엇을 언제 삭제하고 누가 접근하는가?

> 이 실습 앱에는 실제 이름·연락처·녹음·transcript·동의 기록을 넣지 않습니다. 실제 연구 자료는 조직의 승인된 저장소와 정책을 사용합니다.

8. 증거는 종류가 아니라 출처·방법·한계와 함께 강해집니다

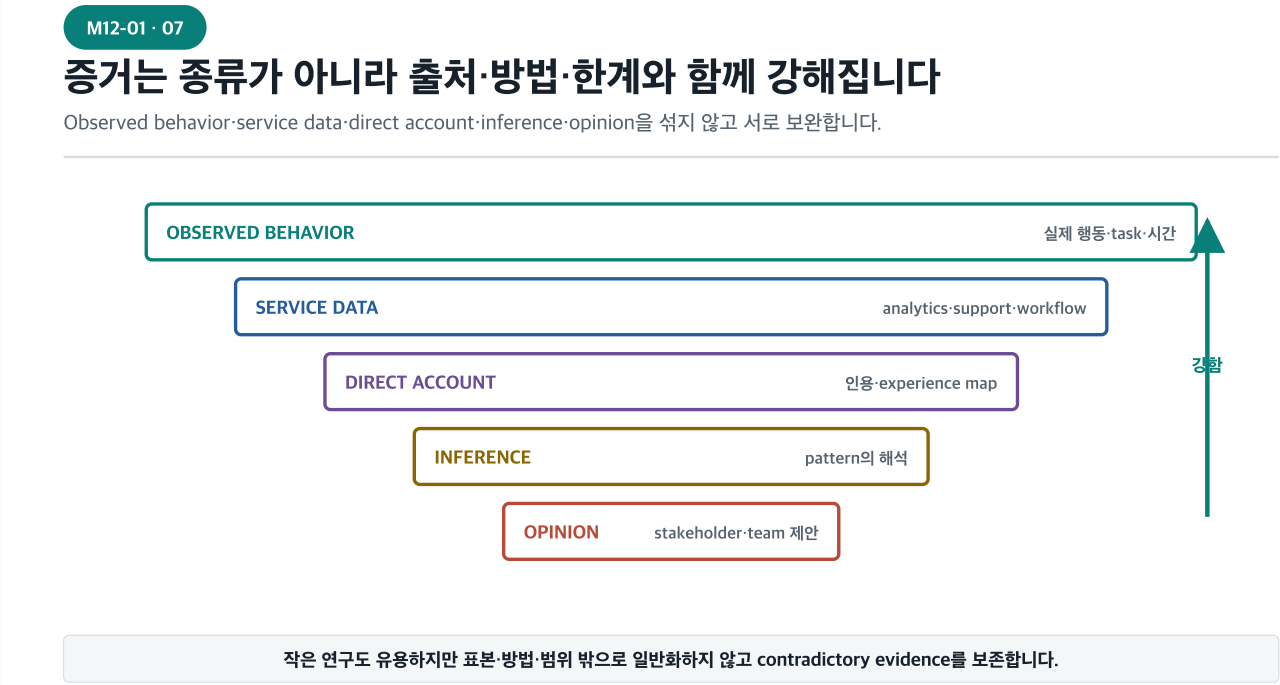
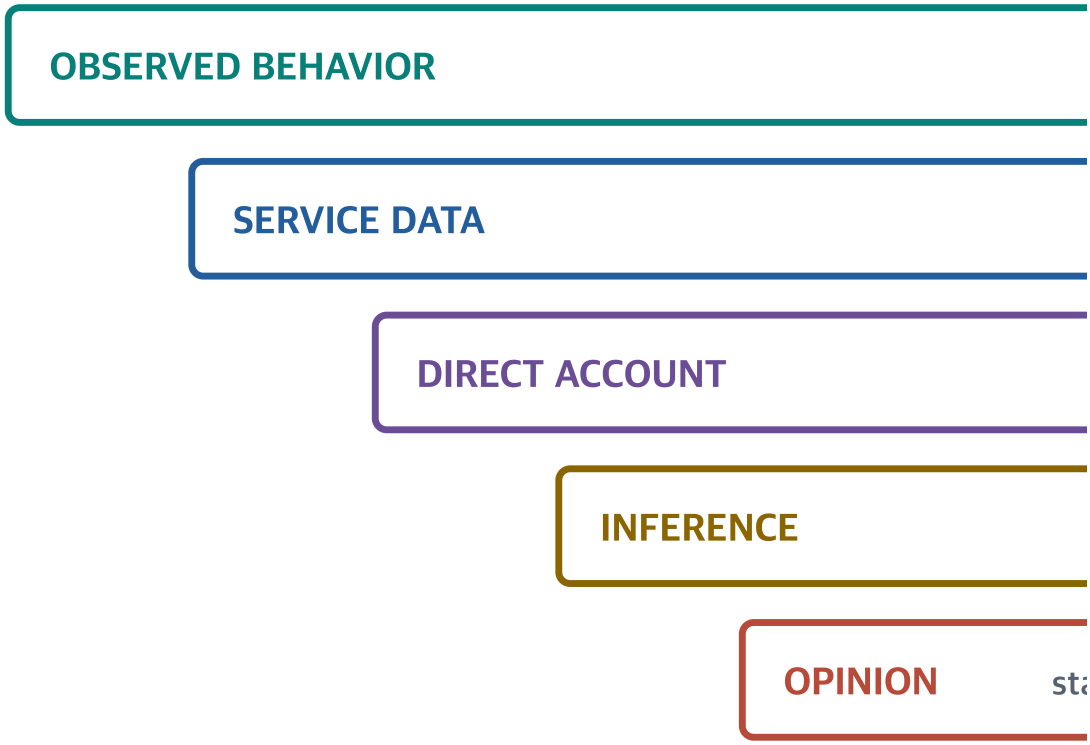


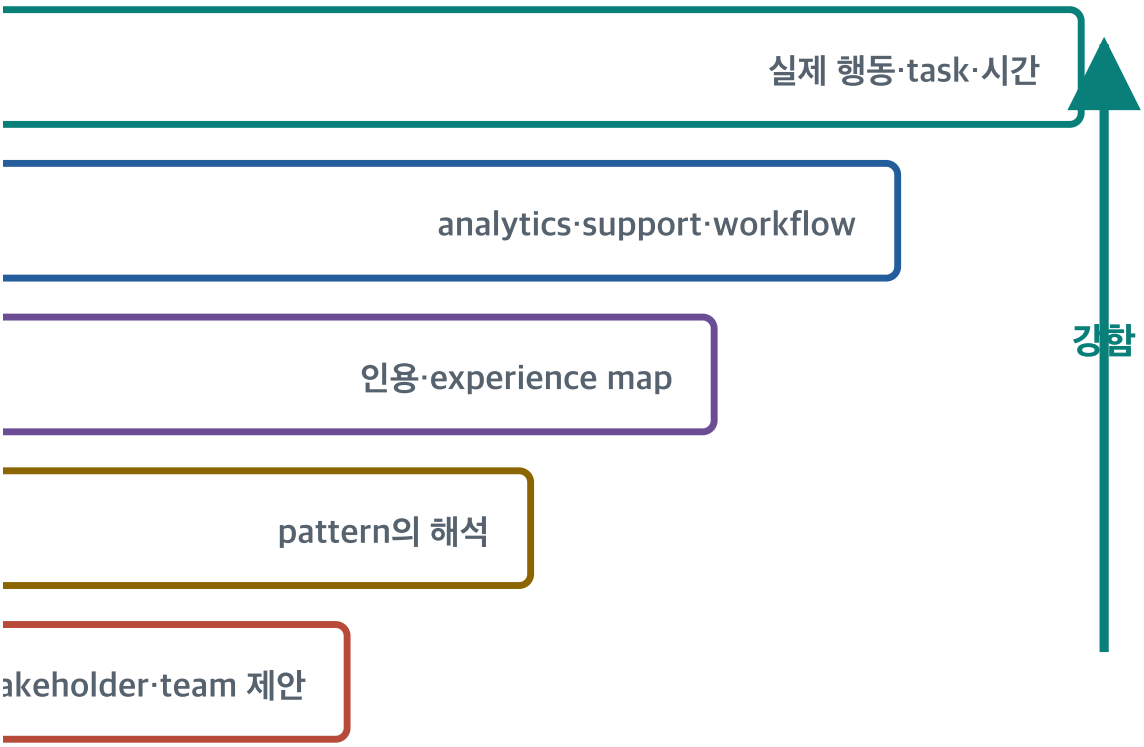
그림 7. 관찰 행동·서비스 자료·직접 진술·해석·의견을 구분하고 서로 다른 증거가 같은 패턴을 지지하는지 확인합니다.

Observed behavior·service data·direct account·inference·opinion을 수



업계 용례집 구성

서지 않고 서로 보완합니다.



8.1 그림을 읽는 질문

관찰 행동·서비스 자료·직접 진술·해석·의견을 구분하고 서로 다른 증거가 같은 패턴을 지지하는지 확인합니다. 아래 표를 읽고 각 행에서 `evidence → decision` 연결을 말해 봅니다.

요소	합성 사례	확인할 것
Observed behavior	실제 행동과 환경	가장 직접적이지만 관찰 범위 한계
Service data	반복되는 수치와 사건	무엇을 기록했는지 정의 필요
Direct account	참여자의 구체적 경험	회상·표현 한계
Inference	연구자의 패턴 해석	대안 설명과 confidence 필요
Opinion	선호·요청·평가	행동 증거와 분리

8.2 흔한 실패

- 한 사람의 강한 요청을 여러 사용자의 행동 증거처럼 씁니다.
- 맥락이나 분모를 지우고 인용·평균 숫자 하나만 남깁니다.
- 아직 확인하지 않은 원인과 해결책을 fact처럼 씁니다.
- 누가 어떤 증거를 언제 다시 확인할지 적지 않습니다.

8.3 3분 연습

1. 내 사례에서 이 그림의 빈칸 하나를 고릅니다.
2. 지금 아는 내용을 fact와 inference로 분리합니다.
3. 필요한 source·method·owner를 한 줄로 씁니다.
4. 답이 나오면 바뀌는 continue·reframe·stop 결정을 표시합니다.

> 좋은 문장은 길어서가 아니라 source·상황·행동·결과·불확실성이 분리되어 다시 검토할 수 있어서 강합니다.

9. 정성 증거를 합성하는 법

9.1 한 세션에서 바로 결론 내리지 않기

세션마다 `관찰 → 직접 인용 → 해석 → 질문`을 분리합니다. 그다음 여러 세션에서 반복 패턴, 차이, 반대 사례, 접근 조건을 비교합니다.

9.2 Evidence card

```
Evidence ID: EV-__
Observed: _____________________
Direct quote or artifact: ______
Context/trigger: _____________
Source/method/sample/as-of: ____
Interpretation: _______________
Alternative explanation: ______
Limitation: ___________________
Owner/next check: _____________
```

9.3 강한 인사이트의 문법

> [어떤 사용자]는 [어떤 상황]에서 [목표]를 위해 [현재 행동]을 한다. [여러 증거]에서 [반복 패턴]이 보이지만 [반대 사례·한계]가 있어 [원인]은 아직 가설이다. 따라서 다음에는 [의사결정을 바꾸는 질문]을 확인한다.

10. 증상·문제·원인 가설·해결책 가설 분리하기

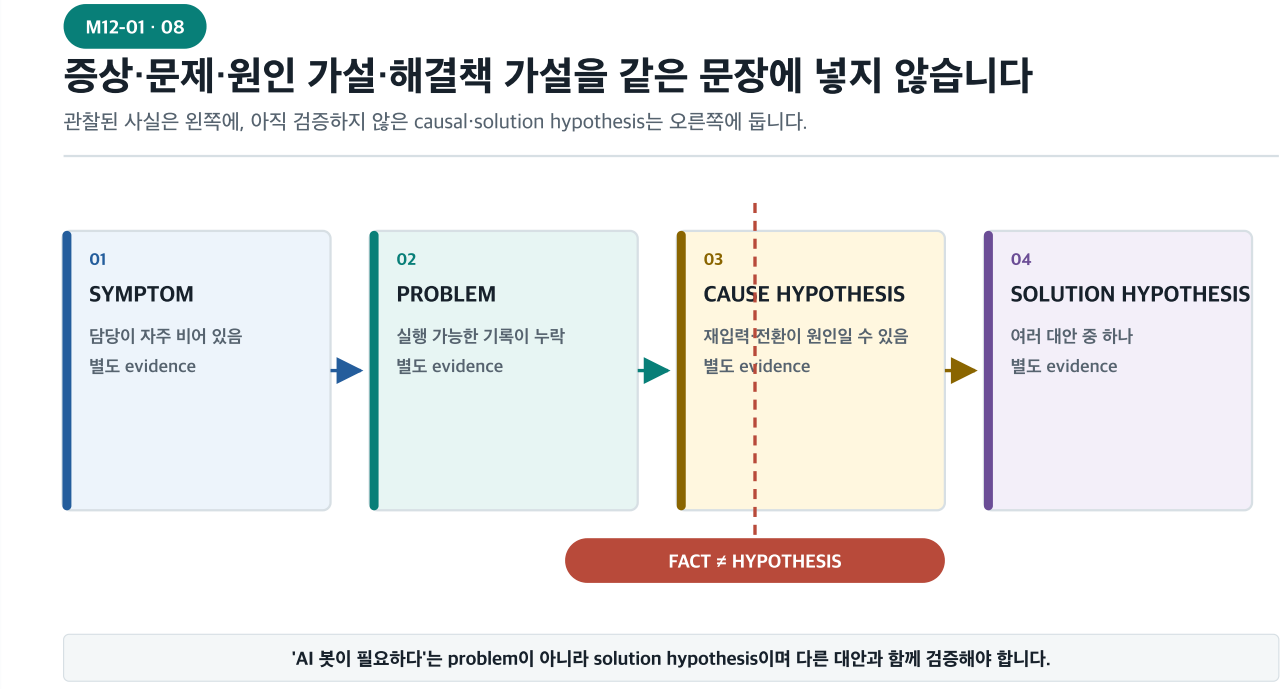


그림 8. 관찰된 현상에서 문제를 정의하되, 왜 생겼는지와 무엇으로 풀지는 별도 가설로 둡니다.

관찰된 사실은 왼쪽에, 아직 검증하지 않은 causal·solution hypothesis는 .

01

SYMPTOM

담당이 자주 비어 있음

별도 evidence

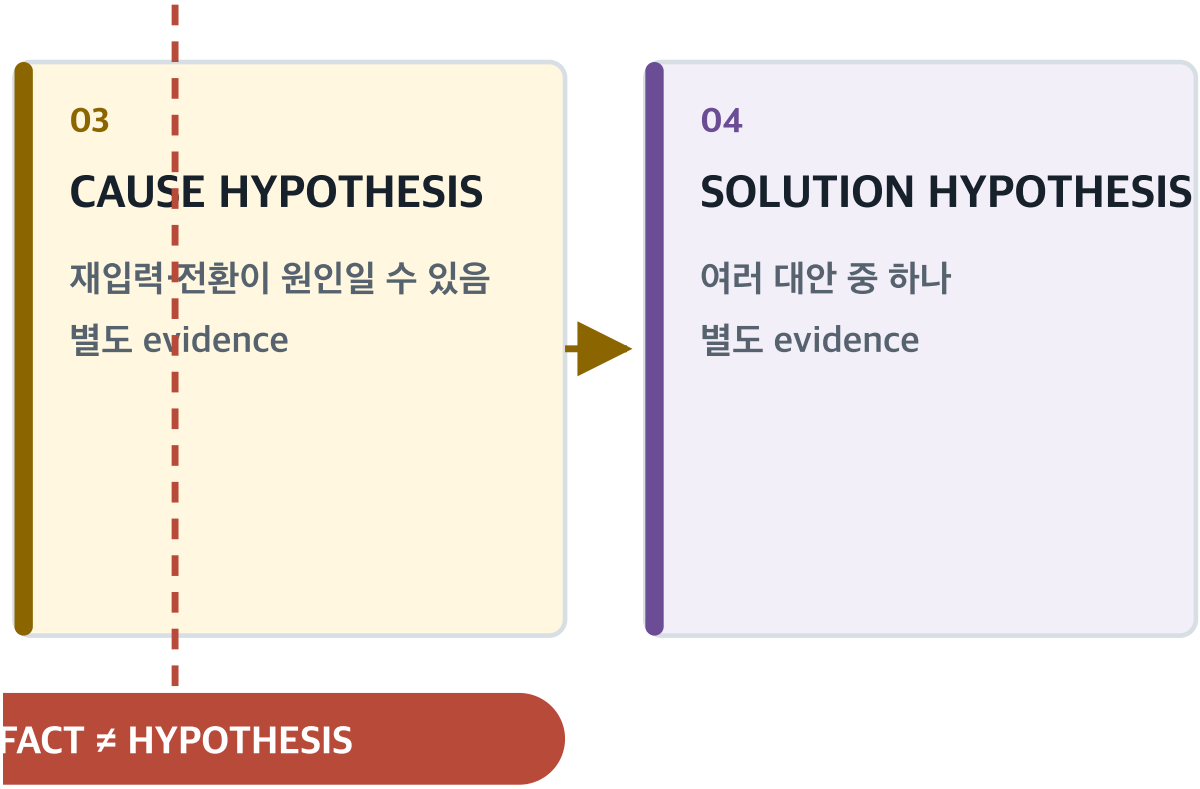
02

PROBLEM

실행 가능한 기록이 누락

별도 evidence

오른쪽에 둡니다.


03
CAUSE HYPOTHESIS
재입력 전환이 원인일 수 있음
별도 evidence
04
SOLUTION HYPOTHESIS
여러 대안 중 하나
별도 evidence
FACT ≠ HYPOTHESIS

10.1 그림을 읽는 질문

관찰된 현상에서 문제를 정의하되, 왜 생겼는지와 무엇으로 풀지는 별도 가설로 둡니다. 아래 표를 읽고 각 행에서 `evidence → decision` 연결을 말해 봅니다.

요소	합성 사례	확인할 것
Symptom	담당·기한 누락과 반복 재확인	Observed evidence
Problem	여러 채널 재입력 중 완전한 실행 기록을 만들기 어려움	Evidence-backed frame
Cause hypothesis	공유 확인 시점과 책임 규칙이 약함	검증 전 설명
Solution hypothesis	확인 흐름이나 도구가 개선할 수 있음	비교해야 할 후보

10.2 흔한 실패

- 한 사람의 강한 요청을 여러 사용자의 행동 증거처럼 씁니다.
- 맥락이나 분모를 지우고 인용·평균 숫자 하나만 남깁니다.
- 아직 확인하지 않은 원인과 해결책을 fact처럼 씁니다.
- 누가 어떤 증거를 언제 다시 확인할지 적지 않습니다.

10.3 3분 연습

1. 내 사례에서 이 그림의 빈칸 하나를 고릅니다.
2. 지금 아는 내용을 fact와 inference로 분리합니다.
3. 필요한 source·method·owner를 한 줄로 씁니다.
4. 답이 나오면 바뀌는 continue·reframe·stop 결정을 표시합니다.

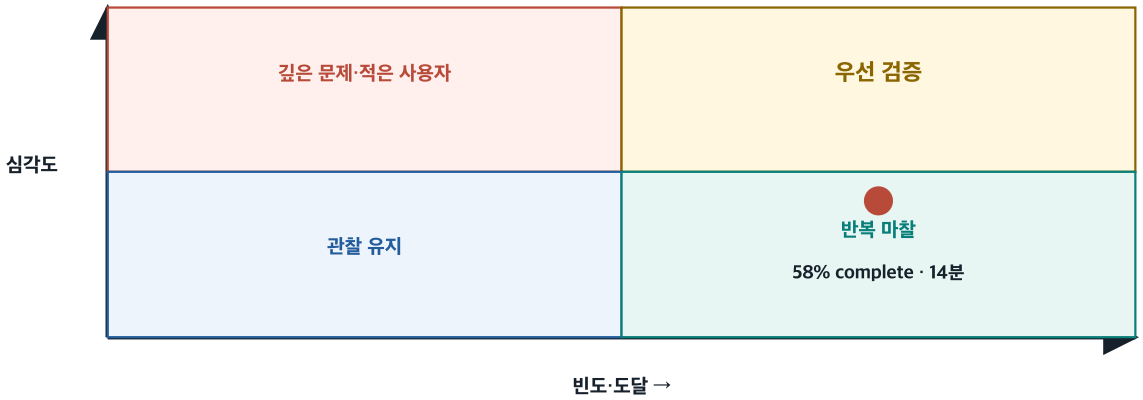
> 좋은 문장은 길어서가 아니라 source·상황·행동·결과·불확실성이 분리되어 다시 검토할 수 있어서 강합니다.

11. 문제 크기는 빈도·심각도·도달·baseline으로 보기

M12-01 · 09

문제의 크기는 빈도·심각도·도달·baseline으로 봅니다

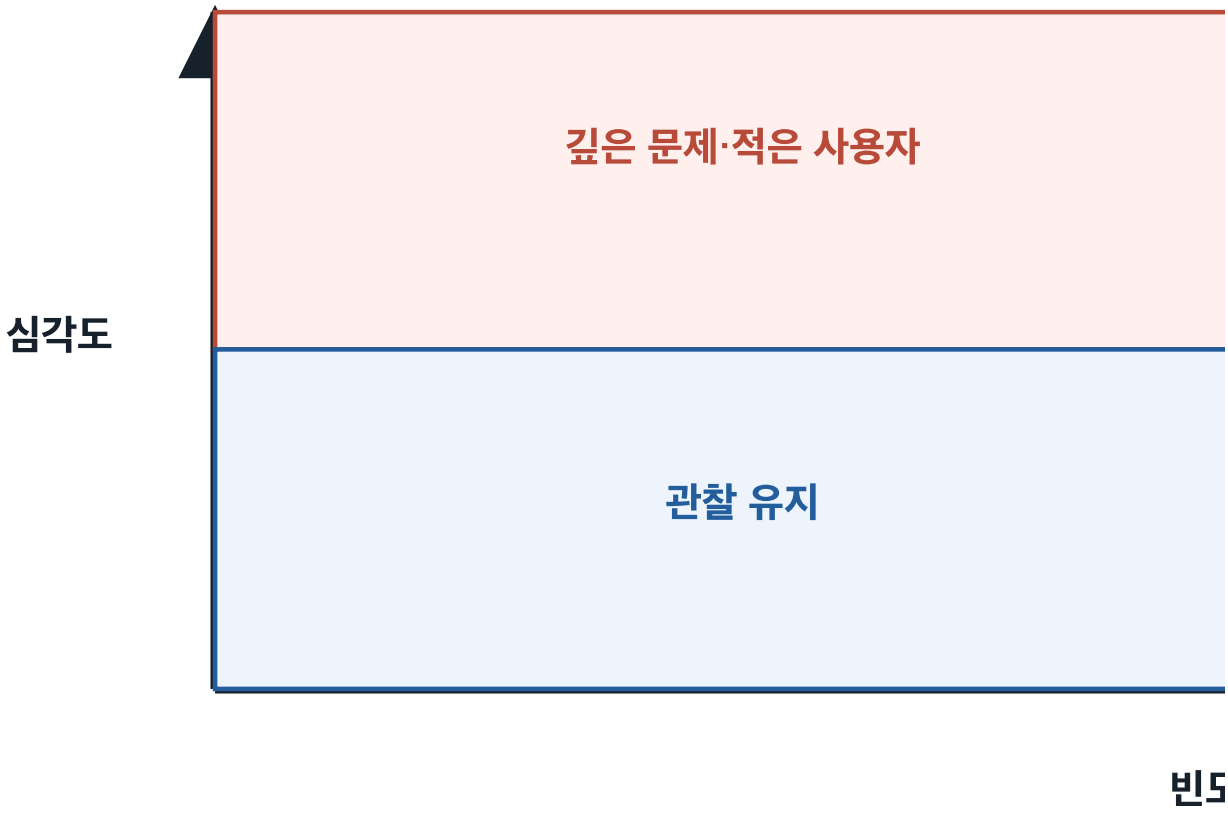
한 명의 강한 불만도 중요하지만 규모를 주장할 때는 분모·segment·기간·한계를 함께 둡니다.



정성 증거는 왜를, 정량 baseline은 얼마나를 설명하며 둘 중 하나가 다른 하나를 대신하지 않습니다.

그림 9. 한 번의 강한 인용만으로 우선순위를 정하지 않고 반복 빈도, 피해 크기, 영향 집단, 시간·오류 baseline을 함께 봅니다.

한 명의 강한 불만도 중요하지만 규모를 주장할 때는 분모·segment·기간·한



[illegible]

[illegible]계를 함께 둡니다.

[illegible]·도달 →

11.1 그림을 읽는 질문

한 번의 강한 인용만으로 우선순위를 정하지 않고 반복 빈도, 피해 크기, 영향 집단, 시간·오류 baseline을 함께 봅니다. 아래 표를 읽고 각 행에서 `evidence → decision` 연결을 말해 봅니다.

요소	합성 사례	확인할 것
Frequency	주 3회 이상 회의 뒤 반복	기회 수와 기간 명시
Severity	누락 시 책임 혼선과 실행 지연	사용자·업무·안전 영향
Reach	관찰된 18명과 4개 사용자 group	전체 시장으로 일반화 금지
Baseline	완전 기록 58%, 재입력 중앙값 14분	분자·분모·출처·시점

11.2 흔한 실패

- 한 사람의 강한 요청을 여러 사용자의 행동 증거처럼 씁니다.
- 맥락이나 분모를 지우고 인용·평균 숫자 하나만 남깁니다.
- 아직 확인하지 않은 원인과 해결책을 fact처럼 씁니다.
- 누가 어떤 증거를 언제 다시 확인할지 적지 않습니다.

11.3 3분 연습

1. 내 사례에서 이 그림의 빈칸 하나를 고릅니다.
2. 지금 아는 내용을 fact와 inference로 분리합니다.
3. 필요한 source·method·owner를 한 줄로 씁니다.
4. 답이 나오면 바뀌는 continue·reframe·stop 결정을 표시합니다.

> 좋은 문장은 길어서가 아니라 source·상황·행동·결과·불확실성이 분리되어 다시 검토할 수 있어서 강합니다.

12. 해결책이 없는 문제 문장 쓰기

12.1 문장 공식

```
[사용자]는 [trigger와 context]에서 [job]을 하려 할 때,
[현재 대안과 barrier] 때문에 [관찰된 consequence]를 겪는다.
[source·as-of·limitation]에서 [frequency·severity·reach·baseline]이 보인다.
우리가 바라는 변화는 [desired outcome]이며,
[target·period·guardrail] 안에서 확인한다.
```

12.2 합성 사례의 문제 정의 후보

5~20명 팀의 운영 조정 담당자는 주 3회 이상 회의 직후 결정·담당·기한을 실행 가능한 기록으로 남기려 한다. 그러나 채팅·공유 표·개인 메모 사이를 전환하고 재입력하며 담당자를 다시 확인하는 현재 여정 때문에 누락과 지연이 반복된다. 2026-07 합성 자료 36세션에서는 완전 기록 baseline 58%, 재입력 중앙값 14분이었으며 실제 모집·통계적 일반화 근거는 아니다. 기대 결과는 4주 안에 완전 기록률 85%, 재입력 5분이며 오배정·privacy·접근성·회의 시간·비용 guardrail을 동시에 지킨다.

12.3 금지어 검토

문제 문장에 `AI 봇`, `챗봇`, `대시보드`, `앱을 만든다`, `자동화 솔루션`이 있으면 해결책 가설 칸으로 옮깁니다. 기술명이 항상 나쁜 것은 아니지만 problem statement의 주어가 되면 탐색을 고정하기 쉽습니다.

13. 기대 결과를 기능과 분리하기

13.1 Output과 outcome

Output	Desired outcome
회의 요약 화면	사용자가 실행 가능한 완전 기록을 남긴다

Output	Desired outcome
확인 알림 기능	담당자가 책임·기한을 정확히 확인한다
관리자 dashboard	팀이 누락을 빠르게 발견하고 복구한다
AI 분류 모델	잘못된 담당 지정 없이 정리 시간을 줄인다

13.2 세 역할의 결과

- 사용자 결과: 완전 기록률 증가, 재입력 시간 감소, 통제감 유지
- 수혜자 결과: 책임과 기한의 정확성, 진행 상태 가시성
- 제공자 결과: 지원·복구 부담이 증가하지 않음
- 사업 결과: 완전 기록당 비용과 위험이 허용 범위 안에 있음

13.3 결과 문장 검토

1. 기능명을 빼도 의미가 남는가?
2. 사용자의 행동이나 상태가 달라지는가?
3. 누가 언제 변화해야 하는가?
4. baseline과 target을 같은 정의로 잴 수 있는가?
5. 특정 집단의 악화를 숨기지 않는가?

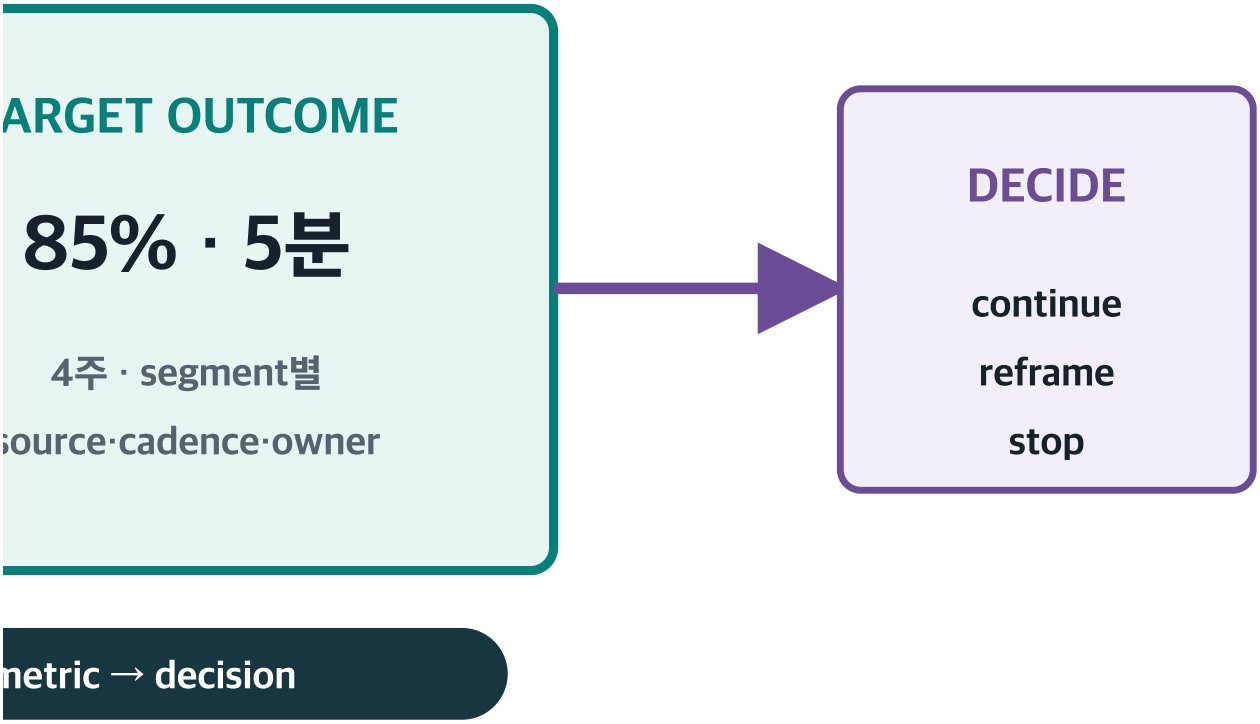
14. Baseline에서 target으로 가는 측정 계약

그림 10. 좋아지면 좋겠다는 문장을 baseline·target·기간·출처·주기·segment·owner가 있는 판정 가능한 계약으로 바꿉니다.

누가 얼마나 baseline에서 target으로

누가·무엇이·어느 방향으로·언제까지 바뀌며 어떤 source와 cadence로 판

정할지 씁니다.



metric → decision

14.1 그림을 읽는 질문

좋아지면 좋겠다는 문장을 baseline·target·기간·출처·주기·segment·owner가 있는 판정 가능한 계약으로 바꿉니다. 아래 표를 읽고 각 행에서 `evidence → decision` 연결을 말해 봅니다.

요소	합성 사례	확인할 것
Outcome	실행 가능한 완전 기록을 남긴다	기능명이 아닌 사용자 상태
Baseline	58% complete·14분	현재 정의와 출처
Target	4주 안에 85%·5분	방향·값·기간
Review	주별·group별 확인	continue·reframe·stop

14.2 흔한 실패

- 한 사람의 강한 요청을 여러 사용자의 행동 증거처럼 씁니다.
- 맥락이나 분모를 지우고 인용·평균 숫자 하나만 남깁니다.
- 아직 확인하지 않은 원인과 해결책을 fact처럼 씁니다.
- 누가 어떤 증거를 언제 다시 확인할지 적지 않습니다.

14.3 3분 연습

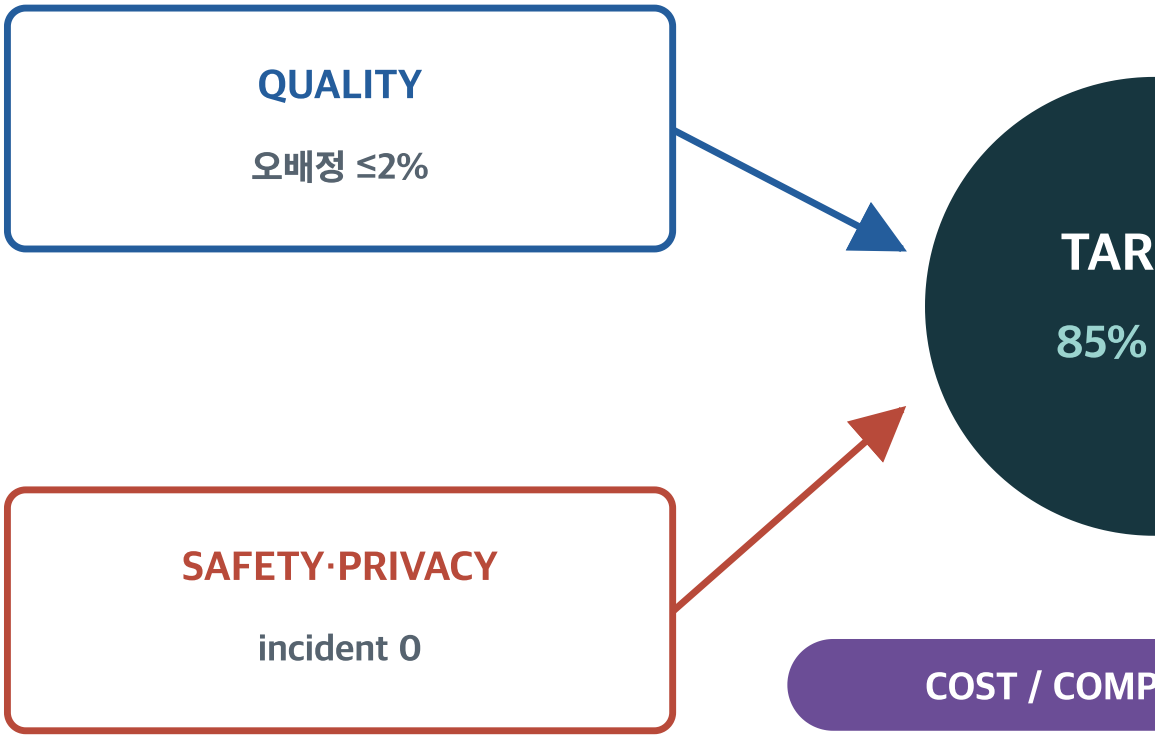
1. 내 사례에서 이 그림의 빈칸 하나를 고릅니다.
2. 지금 아는 내용을 fact와 inference로 분리합니다.
3. 필요한 source·method·owner를 한 줄로 씁니다.
4. 답이 나오면 바뀌는 continue·reframe·stop 결정을 표시합니다.

> 좋은 문장은 길어서가 아니라 source·상황·행동·결과·불확실성이 분리되어 다시 검토할 수 있어서 강합니다.

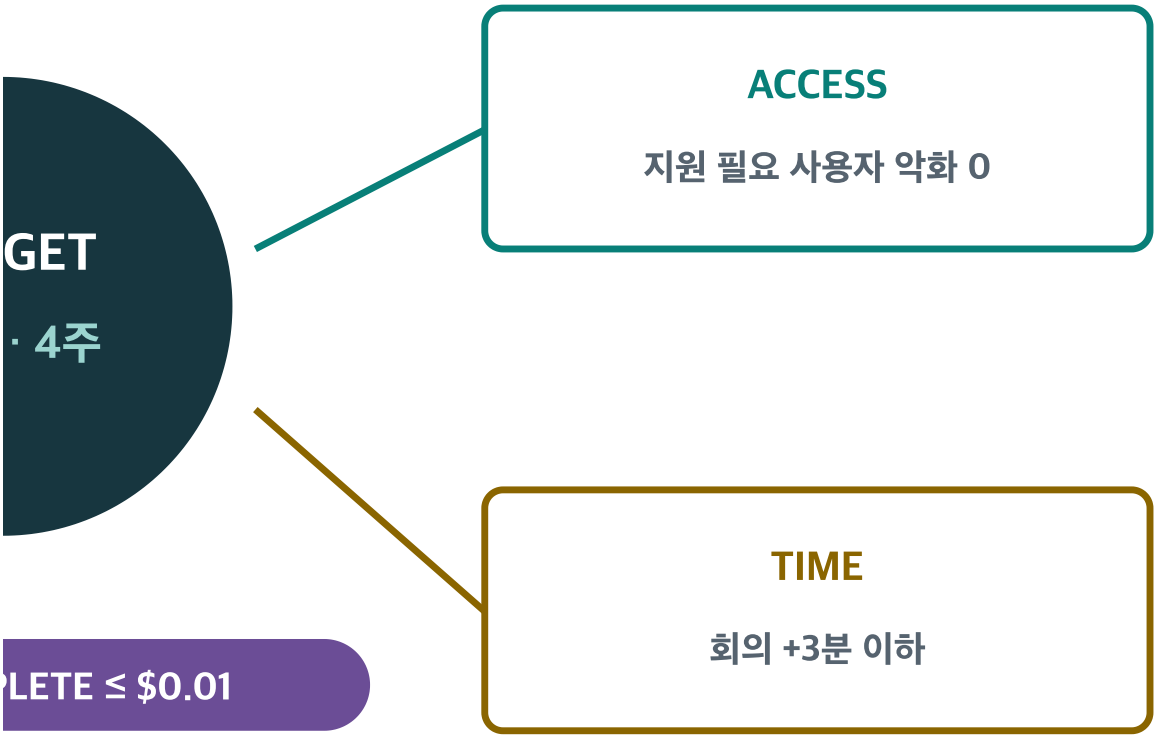
15. Target은 guardrail 안에서만 성공입니다

그림 11. 평균 결과가 좋아져도 품질·안전·privacy·접근성·시간·비용·신뢰성을 희생하면 성공으로 판정하지 않습니다.

Quality·safety·privacy·accessibility·time·cost가 평균 개선의 부작용과 트



특정 집단 악화를 막습니다.



15.1 그림을 읽는 질문

평균 결과가 좋아져도 품질·안전·privacy·접근성·시간·비용·신뢰성을 희생하면 성공으로 판정하지 않습니다. 아래 표를 읽고 각 행에서 `evidence → decision` 연결을 말해 봅니다.

요소	합성 사례	확인할 것
Quality	오배정률 2% 이하	빠르지만 틀린 결과 방지
Safety·privacy	민감정보 사고 0	수집·노출 경계
Access	지원 필요 사용자의 결과가 악화되지 않음	No group worse
Time	회의 연장 3분 이하	문제를 다른 단계로 이동 금지
Cost	완전 기록당 USD 0.01 이하	M11-04 제약 연결

15.2 흔한 실패

- 한 사람의 강한 요청을 여러 사용자의 행동 증거처럼 씁니다.
- 맥락이나 분모를 지우고 인용·평균 숫자 하나만 남깁니다.
- 아직 확인하지 않은 원인과 해결책을 fact처럼 씁니다.
- 누가 어떤 증거를 언제 다시 확인할지 적지 않습니다.

15.3 3분 연습

1. 내 사례에서 이 그림의 빈칸 하나를 고릅니다.
2. 지금 아는 내용을 fact와 inference로 분리합니다.
3. 필요한 source·method·owner를 한 줄로 씁니다.
4. 답이 나오면 바뀌는 continue·reframe·stop 결정을 표시합니다.

> 좋은 문장은 길어서가 아니라 source·상황·행동·결과·불확실성이 분리되어 다시 검토할 수 있어서 강합니다.

16. M11-04 비용·지연·license·reliability 제약 연결

문제 발견은 경제·운영 현실에서 분리되지 않습니다. 다만 비용 때문에 사용자·품질·안전 문제를 지우지 않고, trade-off와 승인 경계를 드러냅니다.

Handoff	합성 값	M12-01에서 하는 일
Cost per complete record	USD 0.01 이하	outcome guardrail에 연결
Quality floor	오배정률 2% 이하	빠르지만 틀린 성공 차단
Latency/time	회의 연장 3분 이하	부담의 단계 이동 방지
Privacy	민감정보 사고 0	데이터 최소화·접근 경계
Accessibility	지원 필요 사용자 no worse	segment별 결과 확인
Reliability	측정·복구 가능성 유지	운영 설계 제거 금지
License	사용권·데이터 조건 확인	실제 도구 선택 전 재검토

17. Fact·inference·assumption·unknown 구분하기

그림 12. 같은 보드에 있어도 네 상태는 서로 다른 행동을 요구합니다. Fact는 추적하고 inference는 대안을 비교하며 assumption과 unknown은 검증합니다.

Fact·inference·assumption·unknown

확인된 것과 해석한 것, 믿는 것과 모르는 것을 구분하면 가짜 확신과 끝없는

01

FACT

관찰·측정됨

source·as-of

owner·next

02

INFERENCE

pattern 해석

confidence·alternative

owner·next

= 조사 모두 줄어듭니다.

03

ASSUMPTION

참이라 기대

impact·uncertainty

owner·next

04

UNKNOWN

아직 모름

question·method

owner·next

17.1 그림을 읽는 질문

같은 보드에 있어도 네 상태는 서로 다른 행동을 요구합니다. Fact는 추적하고 inference는 대안을 비교하며 assumption과 unknown은 검증합니다. 아래 표를 읽고 각 행에서 `evidence → decision` 연결을 말해 봅니다.

요소	합성 사례	확인할 것
Fact	36개 합성 세션 중 완전 기록 baseline 58%	source·as-of
Inference	채널 전환이 누락에 기여할 가능성이 큼	confidence·alternative
Assumption	팀원이 확인 요청에 실제 응답할 것	impact·uncertainty
Unknown	어떤 확인 시점이 부담을 최소화하는가	question·method

17.2 흔한 실패

- 한 사람의 강한 요청을 여러 사용자의 행동 증거처럼 씁니다.
- 맥락이나 분모를 지우고 인용·평균 숫자 하나만 남깁니다.
- 아직 확인하지 않은 원인과 해결책을 fact처럼 씁니다.
- 누가 어떤 증거를 언제 다시 확인할지 적지 않습니다.

17.3 3분 연습

1. 내 사례에서 이 그림의 빈칸 하나를 고릅니다.
2. 지금 아는 내용을 fact와 inference로 분리합니다.
3. 필요한 source·method·owner를 한 줄로 씁니다.
4. 답이 나오면 바뀌는 continue·reframe·stop 결정을 표시합니다.

> 좋은 문장은 길어서가 아니라 source·상황·행동·결과·불확실성이 분리되어 다시 검토할 수 있어서 강합니다.

18. 가장 영향이 크고 불확실한 가정부터 검증하기

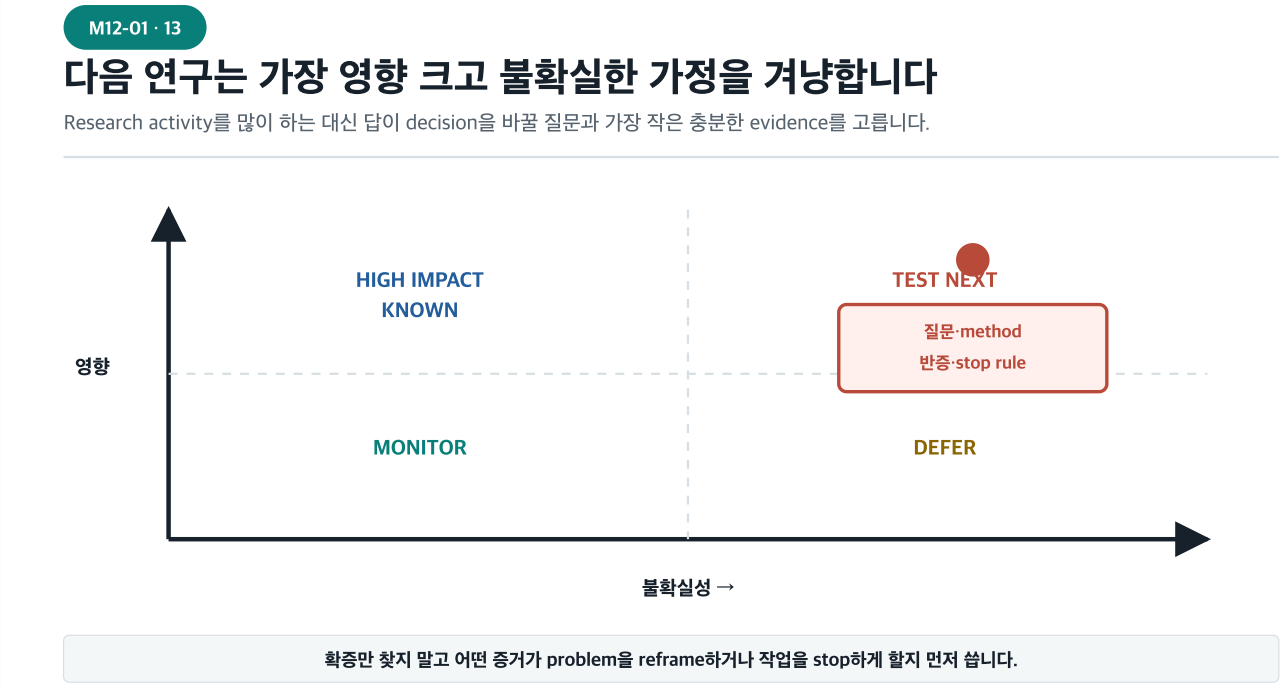
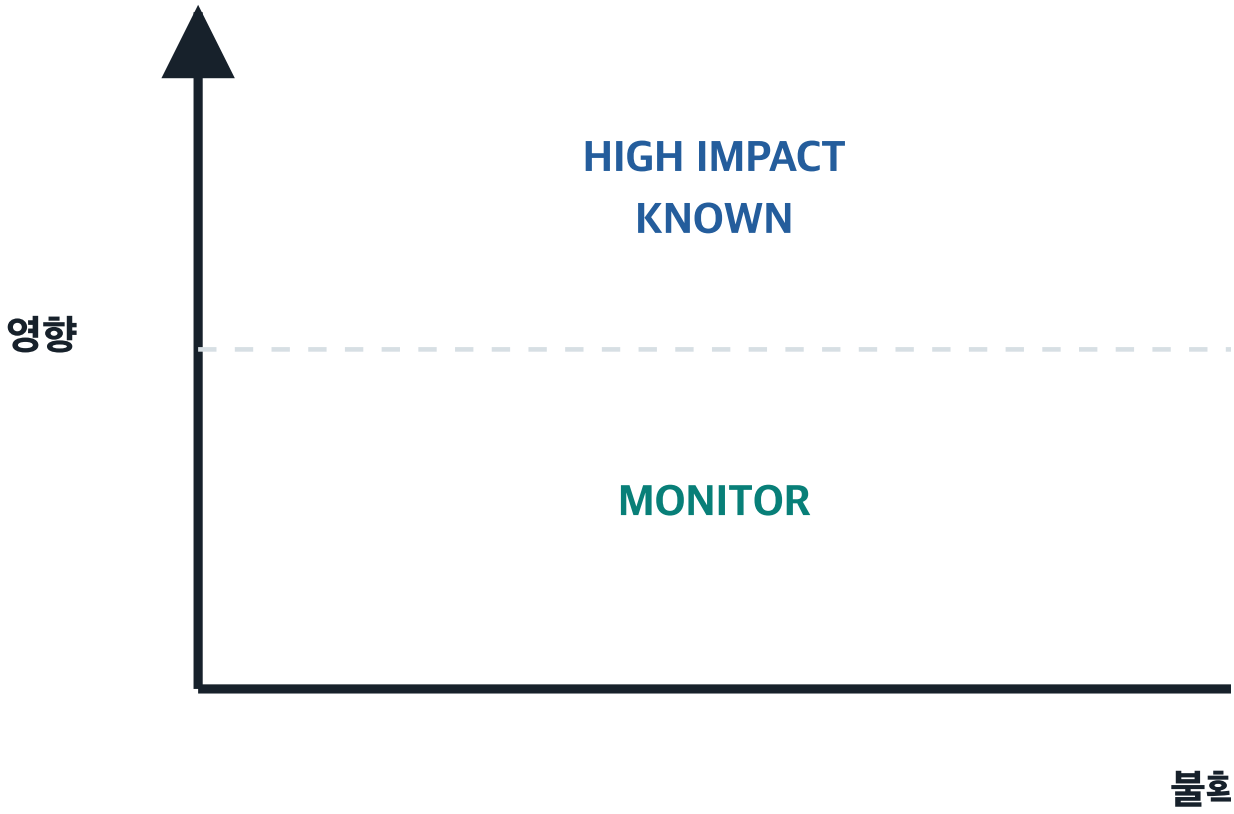
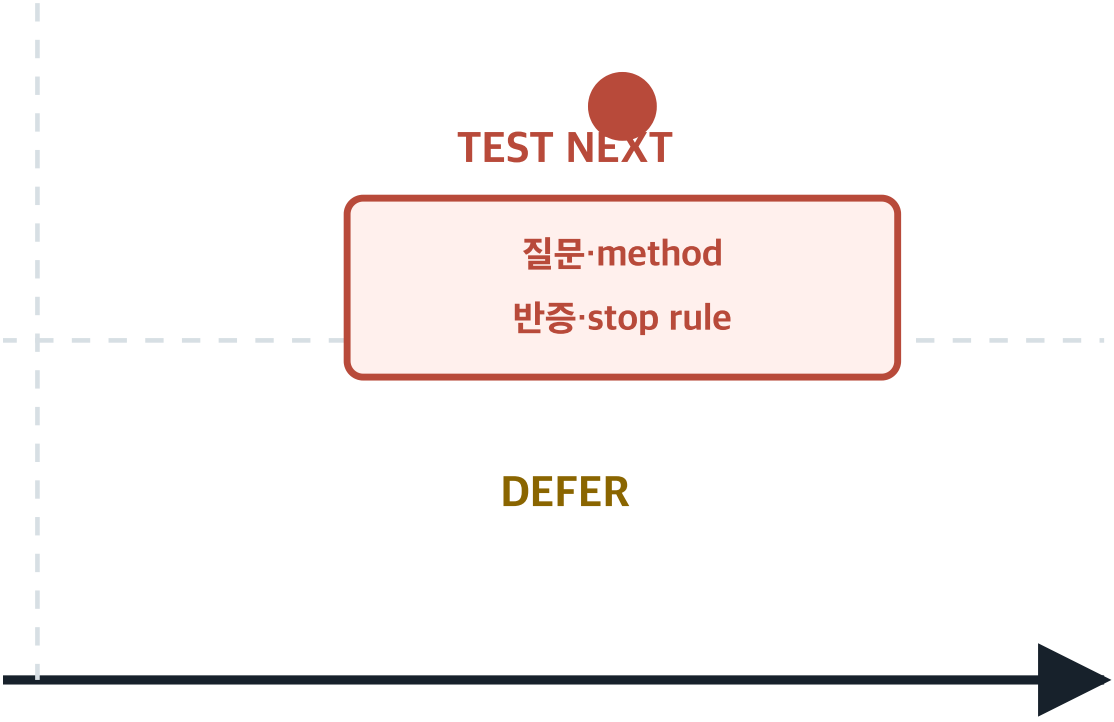


그림 13. 연구 활동이 재미있는지가 아니라 답이 의사결정을 바꾸는지, 가장 작은 증거가 무엇인지로 다음 검증을 고릅니다.

Research activity를 많이 하는 대신 답이 decision을 바꿀 질문과 가장 작



은 충분한 evidence를 고릅니다.



18.1 그림을 읽는 질문

연구 활동이 재미있는지가 아니라 답이 의사결정을 바꾸는지, 가장 작은 증거가 무엇인지로 다음 검증을 고릅니다. 아래 표를 읽고 각 행에서 `evidence → decision` 연결을 말해 봅니다.

요소	합성 사례	확인할 것
High impact·high uncertainty	즉시 검증	틀리면 문제나 결과를 reframe
High impact·known	지속 관찰	guardrail로 관리
Low impact·high uncertainty	보류	우선순위 낮춤
Stop rule	반증 기준 충족 시 중단	확증 편향 제한

18.2 흔한 실패

- 한 사람의 강한 요청을 여러 사용자의 행동 증거처럼 씁니다.
- 맥락이나 분모를 지우고 인용·평균 숫자 하나만 남깁니다.
- 아직 확인하지 않은 원인과 해결책을 fact처럼 씁니다.
- 누가 어떤 증거를 언제 다시 확인할지 적지 않습니다.

18.3 3분 연습

1. 내 사례에서 이 그림의 빈칸 하나를 고릅니다.
2. 지금 아는 내용을 fact와 inference로 분리합니다.
3. 필요한 source·method·owner를 한 줄로 씁니다.
4. 답이 나오면 바뀌는 continue·reframe·stop 결정을 표시합니다.

> 좋은 문장은 길어서가 아니라 source·상황·행동·결과·불확실성이 분리되어 다시 검토할 수 있어서 강합니다.

19. 열두 control로 문제 정의를 점검하기

CTRL-01 · 사용자·수혜자·이해관계자 구분

Primary user·beneficiary·service provider·stakeholder를 분리한다.

- 최소 evidence: `User-role map`
- 확인: source·as-of·owner·limitation이 있는가?
- 실패 신호: 요청이나 가정을 관찰된 fact처럼 썼는가?
- 교정: 반대 사례와 다음 검증을 한 줄 추가합니다.
- 사용자 보호: 이 control이 빠질 때 불리해지는 역할·집단을 적습니다.
- 의사결정 보호: 이 control이 어떤 잘못된 continue 결정을 막는지 적습니다.
- 갱신 조건: 새 evidence가 나오면 어떤 필드와 version을 바꿀지 적습니다.
- Handoff: M12-02가 참조할 evidence ID를 남깁니다.

CTRL-02 · 상황·trigger·빈도·채널

문제가 발생하는 구체 상황과 시작 사건·반복·채널을 기록한다.

- 최소 evidence: `Context and trigger record`
- 확인: source·as-of·owner·limitation이 있는가?
- 실패 신호: 요청이나 가정을 관찰된 fact처럼 썼는가?
- 교정: 반대 사례와 다음 검증을 한 줄 추가합니다.
- 사용자 보호: 이 control이 빠질 때 불리해지는 역할·집단을 적습니다.
- 의사결정 보호: 이 control이 어떤 잘못된 continue 결정을 막는지 적습니다.
- 갱신 조건: 새 evidence가 나오면 어떤 필드와 version을 바꿀지 적습니다.
- Handoff: M12-02가 참조할 evidence ID를 남깁니다.

CTRL-03 · 현재 대안·workaround·journey

지금 하는 일과 전환·재입력·지원·실패 비용을 end-to-end로 본다.

- 최소 evidence: `Current journey map`
- 확인: source·as-of·owner·limitation이 있는가?
- 실패 신호: 요청이나 가정을 관찰된 fact처럼 썼는가?
- 교정: 반대 사례와 다음 검증을 한 줄 추가합니다.
- 사용자 보호: 이 control이 빠질 때 불리해지는 역할·집단을 적습니다.

- 의사결정 보호: 이 control이 어떤 잘못된 continue 결정을 막는지 적습니다.
- 갱신 조건: 새 evidence가 나오면 어떤 필드와 version을 바꿀지 적습니다.
- Handoff: M12-02가 참조할 evidence ID를 남깁니다.

CTRL-04 · 증거 출처·방법·동의·한계

관찰·인용·측정의 source·method·sample·as-of·consent·limitation을 남긴다.

- 최소 evidence: `Evidence ledger`
- 확인: source·as-of·owner·limitation이 있는가?
- 실패 신호: 요청이나 가정을 관찰된 fact처럼 썼는가?
- 교정: 반대 사례와 다음 검증을 한 줄 추가합니다.
- 사용자 보호: 이 control이 빠질 때 불리해지는 역할·집단을 적습니다.
- 의사결정 보호: 이 control이 어떤 잘못된 continue 결정을 막는지 적습니다.
- 갱신 조건: 새 evidence가 나오면 어떤 필드와 version을 바꿀지 적습니다.
- Handoff: M12-02가 참조할 evidence ID를 남깁니다.

CTRL-05 · 빈도·심각도·도달·영향

문제가 얼마나 자주 누구에게 어떤 시간·오류·포기·비용을 만드는지 baseline으로 센다.

- 최소 evidence: `Problem magnitude baseline`
- 확인: source·as-of·owner·limitation이 있는가?
- 실패 신호: 요청이나 가정을 관찰된 fact처럼 썼는가?
- 교정: 반대 사례와 다음 검증을 한 줄 추가합니다.
- 사용자 보호: 이 control이 빠질 때 불리해지는 역할·집단을 적습니다.
- 의사결정 보호: 이 control이 어떤 잘못된 continue 결정을 막는지 적습니다.
- 갱신 조건: 새 evidence가 나오면 어떤 필드와 version을 바꿀지 적습니다.
- Handoff: M12-02가 참조할 evidence ID를 남깁니다.

CTRL-06 · 증상·문제·원인·해결책 분리

관찰 사실과 원인 가설·해결책 가설을 다른 칸에 둔다.

- 최소 evidence: `Problem hypothesis map`
- 확인: source·as-of·owner·limitation이 있는가?
- 실패 신호: 요청이나 가정을 관찰된 fact처럼 썼는가?
- 교정: 반대 사례와 다음 검증을 한 줄 추가합니다.

- 사용자 보호: 이 control이 빠질 때 불리해지는 역할·집단을 적습니다.
- 의사결정 보호: 이 control이 어떤 잘못된 continue 결정을 막는지 적습니다.
- 갱신 조건: 새 evidence가 나오면 어떤 필드와 version을 바꿀지 적습니다.
- Handoff: M12-02가 참조할 evidence ID를 남깁니다.

CTRL-07 · 사용자 기대 결과

사용자가 하려는 일과 얻어야 할 변화를 기능명 없이 쓴다.

- 최소 evidence: `Desired outcome statement`
- 확인: source·as-of·owner·limitation이 있는가?
- 실패 신호: 요청이나 가정을 관찰된 fact처럼 썼는가?
- 교정: 반대 사례와 다음 검증을 한 줄 추가합니다.
- 사용자 보호: 이 control이 빠질 때 불리해지는 역할·집단을 적습니다.
- 의사결정 보호: 이 control이 어떤 잘못된 continue 결정을 막는지 적습니다.
- 갱신 조건: 새 evidence가 나오면 어떤 필드와 version을 바꿀지 적습니다.
- Handoff: M12-02가 참조할 evidence ID를 남깁니다.

CTRL-08 · Baseline·target·기간·측정

현재값·목표 방향·target·time horizon·measurement source를 연결한다.

- 최소 evidence: `Outcome metric contract`
- 확인: source·as-of·owner·limitation이 있는가?
- 실패 신호: 요청이나 가정을 관찰된 fact처럼 썼는가?
- 교정: 반대 사례와 다음 검증을 한 줄 추가합니다.
- 사용자 보호: 이 control이 빠질 때 불리해지는 역할·집단을 적습니다.
- 의사결정 보호: 이 control이 어떤 잘못된 continue 결정을 막는지 적습니다.
- 갱신 조건: 새 evidence가 나오면 어떤 필드와 version을 바꿀지 적습니다.
- Handoff: M12-02가 참조할 evidence ID를 남깁니다.

CTRL-09 · 품질·안전·privacy·접근성 guardrail

평균 개선이 특정 사용자나 품질·권리를 희생하지 못하게 한다.

- 최소 evidence: `Guardrail register`
- 확인: source·as-of·owner·limitation이 있는가?
- 실패 신호: 요청이나 가정을 관찰된 fact처럼 썼는가?

- 교정: 반대 사례와 다음 검증을 한 줄 추가합니다.
- 사용자 보호: 이 control이 빠질 때 불리해지는 역할·집단을 적습니다.
- 의사결정 보호: 이 control이 어떤 잘못된 continue 결정을 막는지 적습니다.
- 갱신 조건: 새 evidence가 나오면 어떤 필드와 version을 바꿀지 적습니다.
- Handoff: M12-02가 참조할 evidence ID를 남깁니다.

CTRL-10 · 사업·비용·지연·license·reliability 제약

M11-04의 unit cost·quality floor·latency·commercial·license·reliability를 넘겨받는다.

- 최소 evidence: `Constraint handoff`
- 확인: source·as-of·owner·limitation이 있는가?
- 실패 신호: 요청이나 가정을 관찰된 fact처럼 썼는가?
- 교정: 반대 사례와 다음 검증을 한 줄 추가합니다.
- 사용자 보호: 이 control이 빠질 때 불리해지는 역할·집단을 적습니다.
- 의사결정 보호: 이 control이 어떤 잘못된 continue 결정을 막는지 적습니다.
- 갱신 조건: 새 evidence가 나오면 어떤 필드와 version을 바꿀지 적습니다.
- Handoff: M12-02가 참조할 evidence ID를 남깁니다.

CTRL-11 · 가정·불확실성·다음 검증

Fact·inference·assumption·unknown을 구분하고 가장 위험한 질문부터 검증한다.

- 최소 evidence: `Assumption and test register`
- 확인: source·as-of·owner·limitation이 있는가?
- 실패 신호: 요청이나 가정을 관찰된 fact처럼 썼는가?
- 교정: 반대 사례와 다음 검증을 한 줄 추가합니다.
- 사용자 보호: 이 control이 빠질 때 불리해지는 역할·집단을 적습니다.
- 의사결정 보호: 이 control이 어떤 잘못된 continue 결정을 막는지 적습니다.
- 갱신 조건: 새 evidence가 나오면 어떤 필드와 version을 바꿀지 적습니다.
- Handoff: M12-02가 참조할 evidence ID를 남깁니다.

CTRL-12 · Version·owner·Gate·handoff

문제 정의의 source·as-of·owner·decision·residual risk와 M12-02 handoff를 남긴다.

- 최소 evidence: `Problem Definition Gate`
- 확인: source·as-of·owner·limitation이 있는가?

- 실패 신호: 요청이나 가정을 관찰된 fact처럼 썼는가?
- 교정: 반대 사례와 다음 검증을 한 줄 추가합니다.
- 사용자 보호: 이 control이 빠질 때 불리해지는 역할·집단을 적습니다.
- 의사결정 보호: 이 control이 어떤 잘못된 continue 결정을 막는지 적습니다.
- 갱신 조건: 새 evidence가 나오면 어떤 필드와 version을 바꿀지 적습니다.
- Handoff: M12-02가 참조할 evidence ID를 남깁니다.

20. 24개 scenario를 네 lane으로 검증하기

각 scenario는 기능 테스트가 아니라 문제 정의 계약의 빈칸을 찾는 검토 질문입니다. 합성 후보는 24/24, critical 100%, lane·control coverage 100%, solution term 0, guardrail 5개 이상, open high-risk assumption 2개 이하를 만족해야 합니다.

Lane A · 사용자·상황

요청자가 아니라 실제·잠재 사용자의 역할·상황·현재 대안과 접근 조건을 확인합니다.

읽는 순서	학습자 질문	기록
1. Source	무엇을 직접 보거나 측정했나	Evidence ID
2. Contract	어떤 필드가 있어야 판정 가능한가	Expected field
3. Protection	빠지면 누가 어떤 피해를 받나	Risk·guardrail
4. Decision	답이 어떤 결정을 바꾸나	continue·reframe·stop

SC-01 · Primary user와 stakeholder 요청자 구분

- Lane: `user-context` · Risk: `proxy-user-bias` · Priority: `P1`
- Control: `CTRL-01` · Trust zone: `user-reality`
- Evidence path: `role-map → primary-user-record`
- Expected contract:
  - [ ] `code` = `PRIMARY_USER_IDENTIFIED`
  - [ ] `primary_user_present` = `True`
  - [ ] `stakeholder_separate` = `True`
  - [ ] `actual_or_likely_user` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-02 · User·beneficiary·provider·supporter 역할과 결과

- Lane: `user-context` · Risk: `proxy-user-bias` · Priority: `P1`
- Control: `CTRL-01` · Trust zone: `user-reality`
- Evidence path: `service-actors → role-outcome-map`
- Expected contract:
  - ☐ `code` = `USER_ROLES_MAPPED`
  - ☐ `beneficiary_present` = `True`
  - ☐ `provider_present` = `True`
  - ☐ `supporter_present` = `True`
  - ☐ `role_conflicts_visible` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-03 · 상황·trigger·빈도·시간 압박

- Lane: `user-context` · Risk: `context-collapse` · Priority: `P1`
- Control: `CTRL-02` · Trust zone: `user-reality`
- Evidence path: `context-observation → trigger-record`
- Expected contract:
  - ☐ `code` = `CONTEXT_TRIGGER_DEFINED`
  - ☐ `situation_specific` = `True`
  - ☐ `trigger_present` = `True`
  - ☐ `frequency_present` = `True`
  - ☐ `time_pressure_present` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-04 · 현재 journey의 online·offline·support channel

- Lane: `user-context` · Risk: `context-collapse` · Priority: `P1`
- Control: `CTRL-02, CTRL-03` · Trust zone: `user-reality`
- Evidence path: `journey-observation → current-journey-map`
- Expected contract:
  - ☐ `code` = `CURRENT_JOURNEY_MAPPED`
  - ☐ `end_to_end` = `True`
  - ☐ `online_offline_present` = `True`
  - ☐ `support_steps_present` = `True`
  - ☐ `handoffs_present` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-05 · 현재 대안·workaround·전환·재입력 비용

- Lane: `user-context` · Risk: `context-collapse` · Priority: `P1`
- Control: `CTRL-03` · Trust zone: `user-reality`
- Evidence path: `current-work → alternative-map`
- Expected contract:
  - ☐ `code` = `CURRENT_ALTERNATIVES_EVIDENCED`
  - ☐ `alternatives_count` = `4`
  - ☐ `switching_visible` = `True`
  - ☐ `reentry_minutes_present` = `True`
  - ☐ `workaround_reason_present` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-06 · 장애·낮은 digital skill·지원 필요 사용자 포함

- Lane: `user-context` · Risk: `constraint-exclusion` · Priority: `P0`
- Control: `CTRL-01, CTRL-02, CTRL-09` · Trust zone: `user-reality`
- Evidence path: `inclusive-recruitment → access-context-map`
- Expected contract:
  - ☐ `code` = `INCLUSIVE_CONTEXT_COVERED`
  - ☐ `disabled_users_in_scope` = `True`
  - ☐ `low_digital_skill_in_scope` = `True`
  - ☐ `support_need_in_scope` = `True`
  - ☐ `access_barriers_present` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

Lane B · 문제 증거

관찰·인용·측정·반대 사례를 추적하고 문제·원인·해결책 가설을 분리합니다.

읽는 순서	학습자 질문	기록
1. Source	무엇을 직접 보거나 측정했나	Evidence ID
2. Contract	어떤 필드가 있어야 판정 가능한가	Expected field
3. Protection	빠지면 누가 어떤 피해를 받나	Risk·guardrail
4. Decision	답이 어떤 결정을 바꾸나	continue·reframe·stop

SC-07 · 관찰·직접 인용·해석·의견 분리

- Lane: `problem-evidence` · Risk: `evidence-weakness` · Priority: `P1`
- Control: `CTRL-04` · Trust zone: `research-evidence`
- Evidence path: `research-notes → evidence-ledger`
- Expected contract:
  - ☐ `code` = `EVIDENCE_TYPES_SEPARATED`
  - ☐ `observation_separate` = `True`
  - ☐ `quote_separate` = `True`
  - ☐ `interpretation_labeled` = `True`
  - ☐ `stakeholder_opinion_labeled` = `True`
- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-08 · Source·method·sample·as-of·동의·한계

- Lane: `problem-evidence` · Risk: `evidence-weakness` · Priority: `P1`
- Control: `CTRL-04` · Trust zone: `research-evidence`
- Evidence path: `research-rounds → provenance-record`
- Expected contract:
  - ☐ `code` = `EVIDENCE_PROVENANCE_COMPLETE`
  - ☐ `source_present` = `True`
  - ☐ `method_present` = `True`
  - ☐ `sample_present` = `True`
  - ☐ `consent_boundary_present` = `True`
  - ☐ `limitations_present` = `True`
- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-09 · 빈도·심각도·도달·시간·오류 baseline

- Lane: `problem-evidence` · Risk: `evidence-weakness` · Priority: `P1`
- Control: `CTRL-05` · Trust zone: `research-evidence`
- Evidence path: `mixed-evidence → magnitude-baseline`
- Expected contract:
  - [ ] `code` = `PROBLEM_MAGNITUDE_BASELINED`
  - [ ] `frequency_present` = `True`
  - [ ] `severity_present` = `True`
  - [ ] `reach_present` = `True`
  - [ ] `time_or_error_present` = `True`
  - [ ] `denominator_present` = `True`
- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-10 · Contradictory evidence·negative case·non-user

- Lane: `problem-evidence` · Risk: `evidence-weakness` · Priority: `P1`
- Control: `CTRL-04, CTRL-05` · Trust zone: `research-evidence`
- Evidence path: `contradiction-log → evidence-confidence`
- Expected contract:
  - [ ] `code` = `CONTRADICTORY_EVIDENCE_RETAINED`
  - [ ] `negative_cases_present` = `True`
  - [ ] `nonusers_present` = `True`
  - [ ] `contradictions_not_deleted` = `True`
  - [ ] `confidence_labeled` = `True`
- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-11 · 증상에서 원인 가설로 가되 사실처럼 쓰지 않기

- Lane: `problem-evidence` · Risk: `solution-lock-in` · Priority: `P1`
- Control: `CTRL-06` · Trust zone: `problem-model`
- Evidence path: `observed-symptoms → cause-hypotheses`
- Expected contract:
  - ☐ `code` = `CAUSE_HYPOTHESIS_LABELED`
  - ☐ `symptom_present` = `True`
  - ☐ `cause_as_hypothesis` = `True`
  - ☐ `alternative_causes_present` = `True`
  - ☐ `causal_claim_avoided` = `True`
- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-12 · Problem·cause·solution hypothesis 분리

- Lane: `problem-evidence` · Risk: `solution-lock-in` · Priority: `P0`
- Control: `CTRL-06` · Trust zone: `problem-model`
- Evidence path: `evidence-synthesis → problem-model`
- Expected contract:
  - ☐ `code` = `PROBLEM_SOLUTION_SEPARATED`
  - ☐ `problem_statement_present` = `True`
  - ☐ `cause_separate` = `True`
  - ☐ `solution_separate` = `True`
  - ☐ `solution_terms_in_problem` = `0`
- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

Lane C · 기대 결과

기능명이 없는 사용자 변화와 baseline·target·guardrail·경제 운영 제약을 연결합니다.

읽는 순서	학습자 질문	기록
1. Source	무엇을 직접 보거나 측정했나	Evidence ID
2. Contract	어떤 필드가 있어야 판정 가능한가	Expected field
3. Protection	빠지면 누가 어떤 피해를 받나	Risk·guardrail
4. Decision	답이 어떤 결정을 바꾸나	continue·reframe·stop

SC-13 · 사용자가 하려는 일과 desired outcome

- Lane: `outcome-measure` · Risk: `outcome-ambiguity` · Priority: `P1`
- Control: `CTRL-07` · Trust zone: `outcome-contract`
- Evidence path: `user-language → desired-outcome`
- Expected contract:
  - ☐ `code` = `DESIRED_OUTCOME_DEFINED`
  - ☐ `job_present` = `True`
  - ☐ `user_change_present` = `True`
  - ☐ `user_words_used` = `True`
  - ☐ `feature_name_absent` = `True`
- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-14 · Baseline·target·direction·time horizon

- Lane: `outcome-measure` · Risk: `outcome-ambiguity` · Priority: `P1`
- Control: `CTRL-08` · Trust zone: `outcome-contract`
- Evidence path: `problem-baseline → outcome-target`
- Expected contract:
  - ☐ `code` = `OUTCOME_TARGET_CONTRACTED`

    - [ ] `baseline_rate` = `0.58`
    - [ ] `target_rate` = `0.85`
    - [ ] `direction` = `increase`
    - [ ] `time_horizon_weeks` = `4`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-15 · Leading·outcome measure·source·cadence·segment

- Lane: `outcome-measure` · Risk: `outcome-ambiguity` · Priority: `P1`
- Control: `CTRL-08` · Trust zone: `outcome-contract`
- Evidence path: `measurement-plan → outcome-dashboard-contract`
- Expected contract:
    - [ ] `code` = `MEASUREMENT_PLAN_DEFINED`
    - [ ] `leading_measure_present` = `True`
    - [ ] `outcome_measure_present` = `True`
    - [ ] `source_present` = `True`
    - [ ] `cadence_present` = `True`
    - [ ] `segment_present` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-16 · Quality·safety·privacy·accessibility guardrail

- Lane: `outcome-measure` · Risk: `constraint-exclusion` · Priority: `P0`
- Control: `CTRL-09` · Trust zone: `outcome-contract`
- Evidence path: `risk-register → guardrail-contract`
- Expected contract:

    - [ ] `code` = `USER_GUARDRAILS_DEFINED`
    - [ ] `quality_present` = `True`
    - [ ] `safety_present` = `True`
    - [ ] `privacy_present` = `True`
    - [ ] `accessibility_present` = `True`
    - [ ] `no_group_worse` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-17 · 사용자 결과·제공자 부담·사업 결과의 긴장

- Lane: `outcome-measure` · Risk: `outcome-ambiguity` · Priority: `P1`
- Control: `CTRL-07, CTRL-10` · Trust zone: `outcome-contract`
- Evidence path: `outcome-map → value-tension-record`
- Expected contract:
    - [ ] `code` = `OUTCOME_TENSIONS_VISIBLE`
    - [ ] `user_outcome_present` = `True`
    - [ ] `provider_outcome_present` = `True`
    - [ ] `business_outcome_present` = `True`
    - [ ] `tradeoff_owner_present` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-18 · M11-04 비용·품질·지연·license·reliability handoff

- Lane: `outcome-measure` · Risk: `constraint-exclusion` · Priority: `P0`
- Control: `CTRL-10` · Trust zone: `outcome-contract`
- Evidence path: `m11-04-handoff → constraint-register`

- Expected contract:
  - ☐ `code` = `ECONOMIC_OPERATIONAL_CONSTRAINTS_LINKED`
  - ☐ `target_cost_per_unit_present` = `True`
  - ☐ `quality_floor_present` = `True`
  - ☐ `latency_objective_present` = `True`
  - ☐ `license_constraint_present` = `True`
  - ☐ `reliability_floor_present` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

Lane D · 가정·결정

지식 상태를 분리하고 가장 위험한 질문·반증·중단·Gate·handoff를 연결합니다.

읽는 순서	학습자 질문	기록
1. Source	무엇을 직접 보거나 측정했나	Evidence ID
2. Contract	어떤 필드가 있어야 판정 가능한가	Expected field
3. Protection	빠지면 누가 어떤 피해를 받나	Risk·guardrail
4. Decision	답이 어떤 결정을 바꾸나	continue·reframe·stop

SC-19 · Fact·inference·assumption·unknown 구분

- Lane: `assumption-decision` · Risk: `evidence-weakness` · Priority: `P1`
- Control: `CTRL-11` · Trust zone: `decision-gate`
- Evidence path: `synthesis-board → assumption-register`
- Expected contract:
  - ☐ `code` = `KNOWLEDGE_STATES_SEPARATED`
  - ☐ `fact_present` = `True`
  - ☐ `inference_labeled` = `True`
  - ☐ `assumption_labeled` = `True`

- [ ] `unknown_present` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-20 · 영향×불확실성으로 가장 위험한 가정 우선

- Lane: `assumption-decision` · Risk: `evidence-weakness` · Priority: `P1`
- Control: `CTRL-11` · Trust zone: `decision-gate`
- Evidence path: `assumption-register → risk-priority`
- Expected contract:
  - [ ] `code` = `HIGHEST_RISK_ASSUMPTION_PRIORITIZED`
  - [ ] `impact_scored` = `True`
  - [ ] `uncertainty_scored` = `True`
  - [ ] `owner_present` = `True`
  - [ ] `top_assumption_count` = `2`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-21 · Research question·method·decision 연결

- Lane: `assumption-decision` · Risk: `evidence-weakness` · Priority: `P1`
- Control: `CTRL-11` · Trust zone: `decision-gate`
- Evidence path: `priority-unknown → next-test-plan`
- Expected contract:
  - [ ] `code` = `NEXT_TEST_DECISION_LINKED`
  - [ ] `open_question_present` = `True`
  - [ ] `method_fit_present` = `True`
  - [ ] `decision_changed_if_answered` = `True`

- [ ] `least_evidence_needed_present` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-22 · 반증·중단·reframe 기준

- Lane: `assumption-decision` · Risk: `solution-lock-in` · Priority: `P1`
- Control: `CTRL-11, CTRL-12` · Trust zone: `decision-gate`
- Evidence path: `test-plan → stop-reframe-rule`
- Expected contract:
  - [ ] `code` = `INVALIDATION_CRITERIA_DEFINED`
  - [ ] `disconfirming_evidence_present` = `True`
  - [ ] `stop_rule_present` = `True`
  - [ ] `reframe_rule_present` = `True`
  - [ ] `decision_owner_present` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-23 · Version·source·as-of·owner·residual risk

- Lane: `assumption-decision` · Risk: `outcome-ambiguity` · Priority: `P1`
- Control: `CTRL-12` · Trust zone: `decision-gate`
- Evidence path: `problem-definition → decision-record`
- Expected contract:
  - [ ] `code` = `PROBLEM_DEFINITION_VERSIONED`
  - [ ] `version_present` = `True`
  - [ ] `source_links_present` = `True`
  - [ ] `as_of_present` = `True`

    - [ ] `owner_present` = `True`
    - [ ] `residual_risk_present` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

SC-24 · Anti-solution 문제 정의·Problem Gate·M12-02 handoff

- Lane: `assumption-decision` · Risk: `solution-lock-in` · Priority: `P0`
- Control: `CTRL-01, CTRL-07, CTRL-12` · Trust zone: `decision-gate`
- Evidence path: `problem-evidence-bundle → problem-definition-gate`
- Expected contract:
    - [ ] `code` = `PROBLEM_READY`
    - [ ] `scenario_passed` = `24`
    - [ ] `critical_pass_rate` = `1.0`
    - [ ] `solution_terms_in_problem` = `0`
    - [ ] `outcome_contract_complete` = `True`
    - [ ] `m12_02_handoff_present` = `True`
    - [ ] `residual_risk_approved` = `True`

- 확인할 것: source·method·sample·as-of·owner·limitation을 원래 자료로 추적할 수 있는가?
- 실패하면: fact와 가설을 다시 나누고 가장 작은 다음 증거를 정합니다.
- 학습 기록: 통과 이유와 아직 남은 한계를 각각 한 문장으로 씁니다.
- 보호 질문: 이 contract가 빠지면 가장 불리해질 사용자·역할은 누구인가?
- 결정 질문: expected가 틀리면 continue·reframe·stop 중 무엇이 바뀌는가?

21. Problem Definition Gate는 네 lane의 증거를 동시에 봅니다

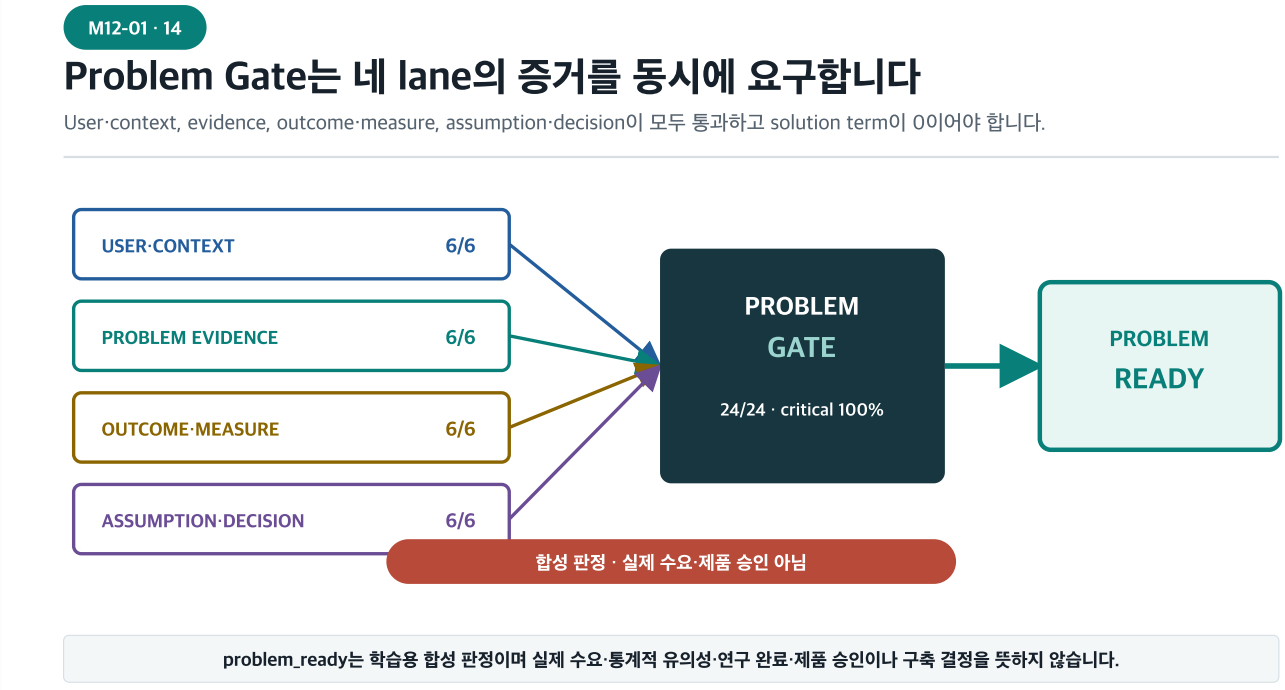
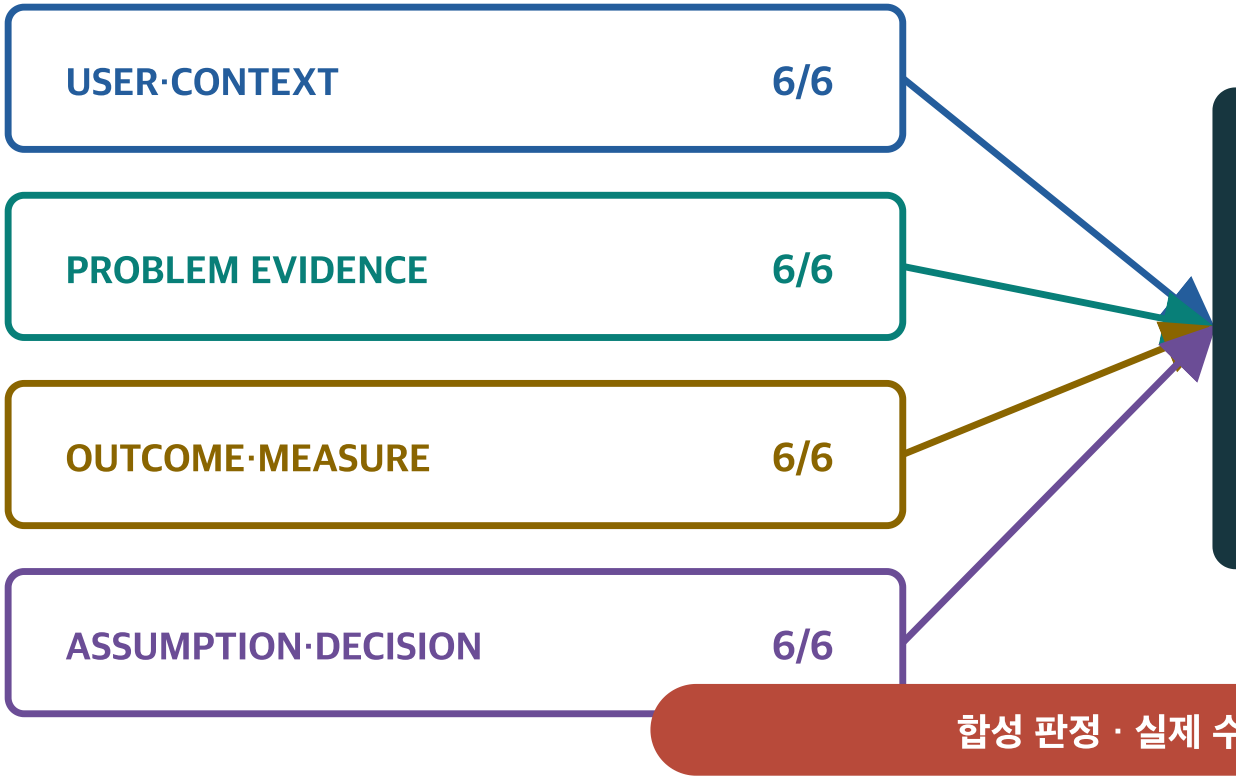


그림 14. 사용자·상황, 문제 증거, 기대 결과, 가정·결정 lane이 모두 6/6이고 critical 100%일 때만 problem_ready 후보가 됩니다.

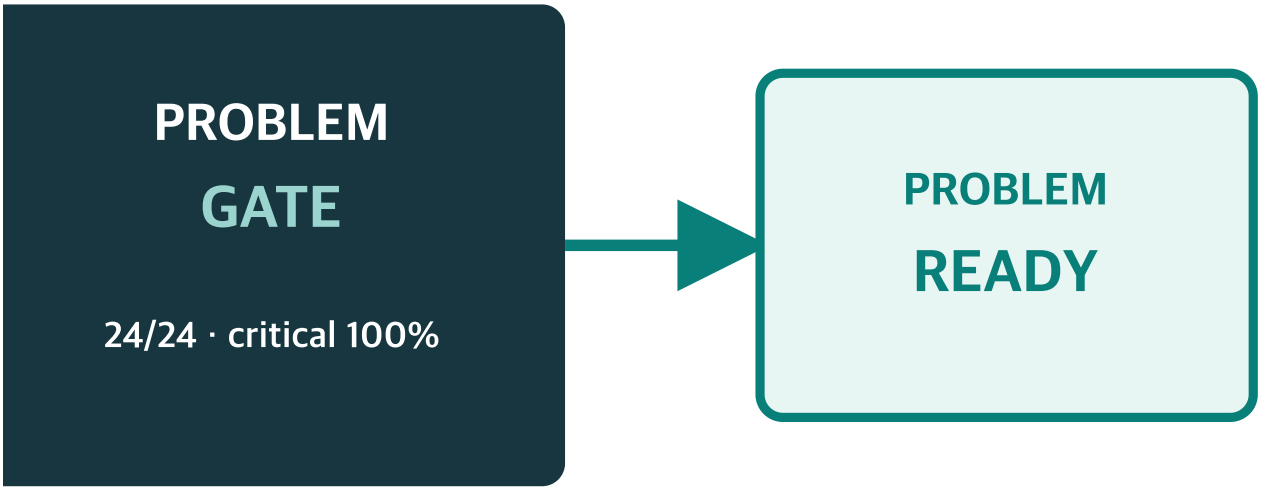
Problem Gate로 내부 구성·제품 정

User·context, evidence, outcome·measure, assumption·decision이 모



[illegible]

두 통과하고 solution term이 0이어야 합니다.



[illegible]요·제품 승인 아님

21.1 그림을 읽는 질문

사용자·상황, 문제 증거, 기대 결과, 가정·결정 lane이 모두 6/6이고 critical 100%일 때만 problem_ready 후보가 됩니다. 아래 표를 읽고 각 행에서 `evidence → decision` 연결을 말해 봅니다.

요소	합성 사례	확인할 것
User-context	SC-01~06	역할·상황·현재 대안·포용
Problem-evidence	SC-07~12	출처·규모·반대 사례·가설 분리
Outcome-measure	SC-13~18	outcome·target·guardrail·제약
Assumption-decision	SC-19~24	불확실성·다음 검증·Gate·handoff

21.2 흔한 실패

- 한 사람의 강한 요청을 여러 사용자의 행동 증거처럼 씁니다.
- 맥락이나 분모를 지우고 인용·평균 숫자 하나만 남깁니다.
- 아직 확인하지 않은 원인과 해결책을 fact처럼 씁니다.
- 누가 어떤 증거를 언제 다시 확인할지 적지 않습니다.

21.3 3분 연습

1. 내 사례에서 이 그림의 빈칸 하나를 고릅니다.
2. 지금 아는 내용을 fact와 inference로 분리합니다.
3. 필요한 source·method·owner를 한 줄로 씁니다.
4. 답이 나오면 바뀌는 continue·reframe·stop 결정을 표시합니다.

> 좋은 문장은 길어서가 아니라 source·상황·행동·결과·불확실성이 분리되어 다시 검토할 수 있어서 강합니다.

22. 문제 정의 스튜디오 실습

22.1 데스크톱 전체 화면

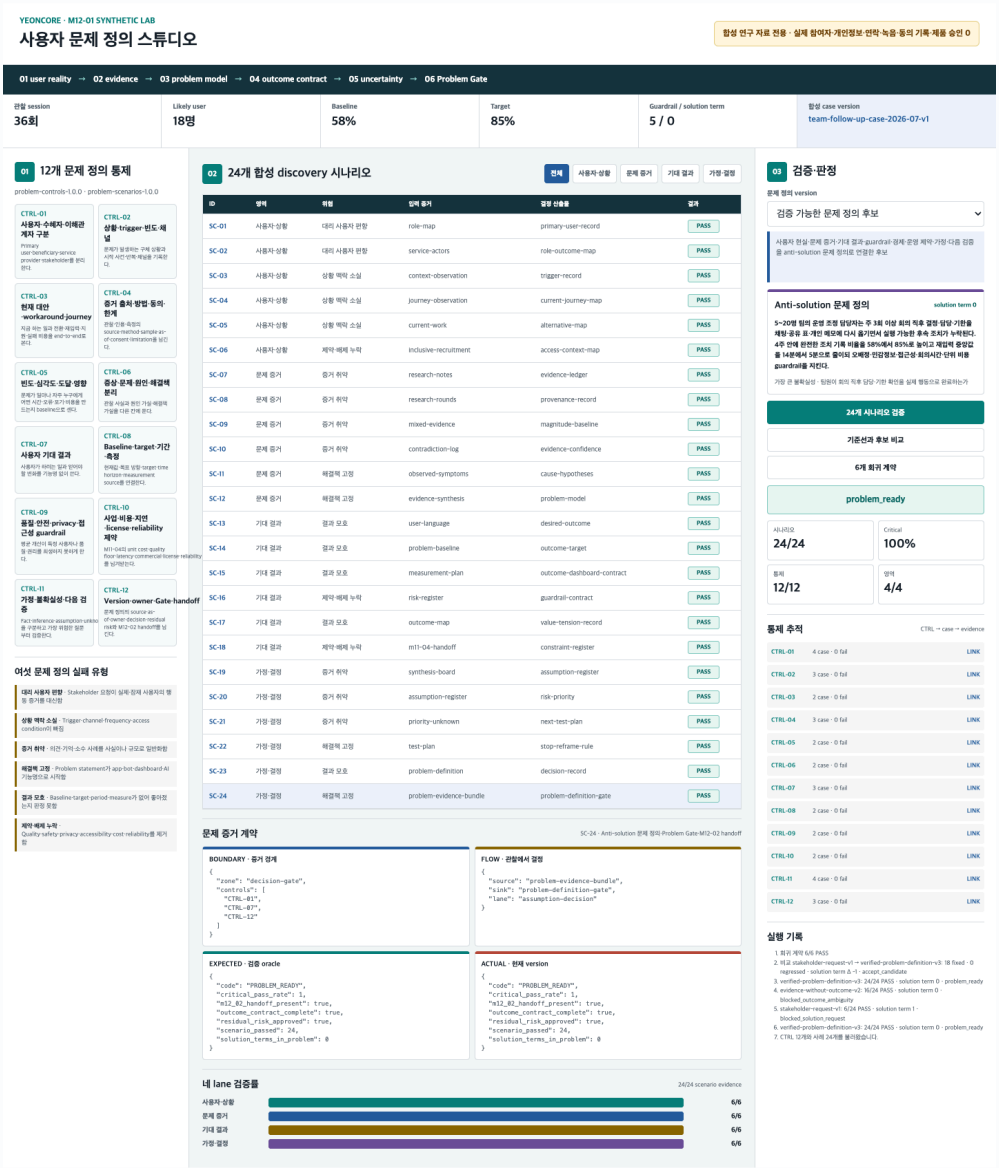


그림 15. 왼쪽에서 version과 lane을 고르고, 가운데 24개 scenario, 오른쪽 evidence·decision을 함께 읽습니다. 숫자 점수만 보지 말고 실패한 contract가 어떤 사용자 피해나 오판을 만드는지 확인합니다.

22.2 모바일 긴 화면

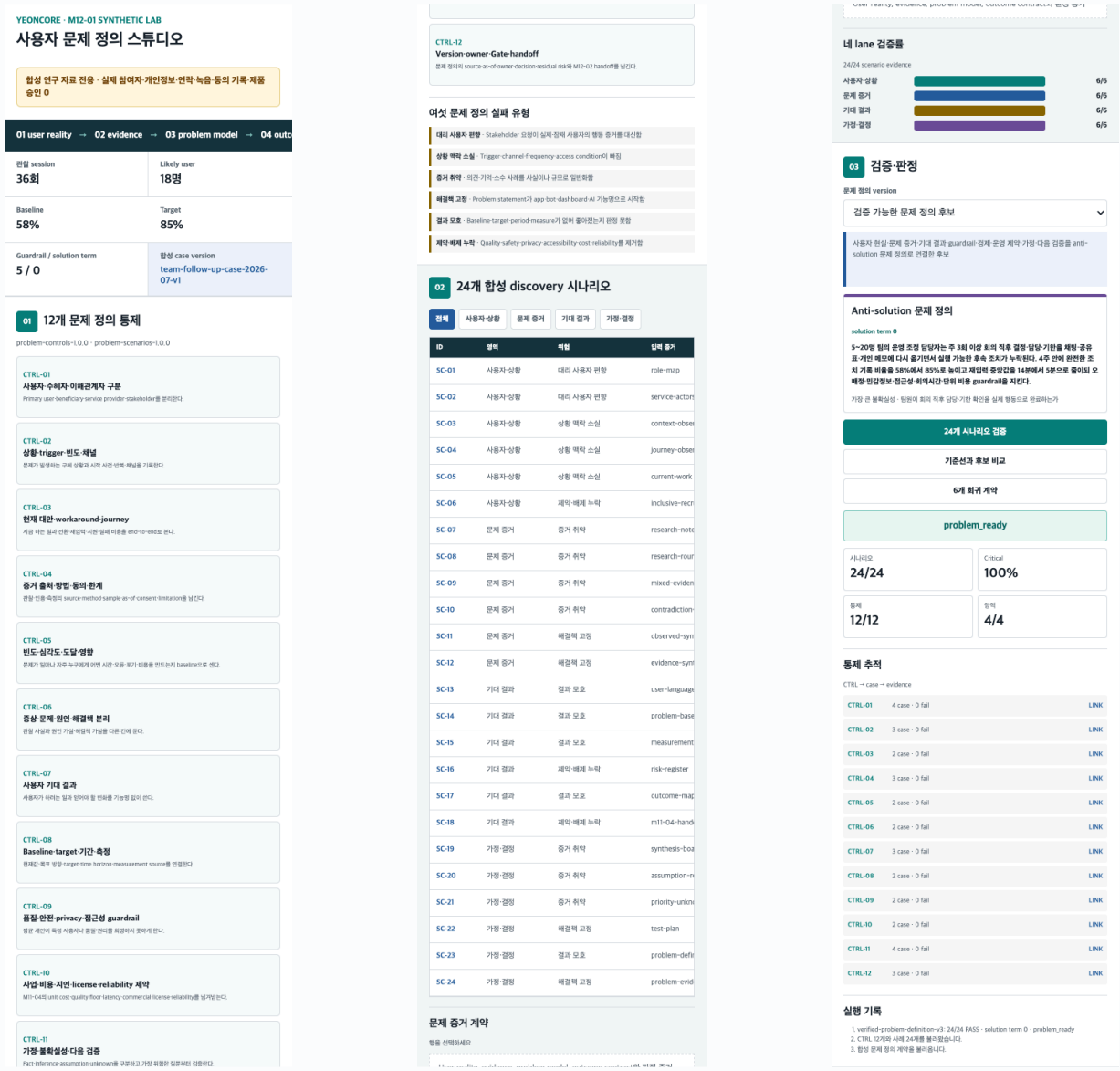


그림 16. 모바일에서는 version 요약, scenario 표, 상세 evidence를 세 구간으로 읽습니다. 표는 내부 가로 스크롤을 사용하며 문서 자체는 화면 너비를 넘지 않습니다.

22.3 세 version에서 배울 것

Version	무엇이 보이나	다음 행동
stakeholder-request-v1	기능 요청은 있으나 18개 contract 실패	사용자·상황·현재 대안부터 탐색
evidence-without-outcome-v2	증거는 있으나 결과·guardrail·Gate 8개 실패	baseline·target·제약·반증 기준 작성
verified-problem-definition-v3	24개 scenario 통과 후보	실제 권한자가 residual risk 검토

`problem_ready`는 이 합성 계약의 자동 판정입니다. 실제 수요 증명, 통계적 유의성, 연구 완료, 법률·정책 승인, 제품 구축 결정을 의미하지 않습니다.

23. 실제 서비스에 적용하는 순서

23.1 0일차 · 기존 증거 모으기

지원 문의, 업무 기록, analytics, 이전 연구, 정책, M11-04 비용·운영 제약을 모으고 출처·시점·담당을 붙입니다. 이미 아는 것과 모르는 것을 구분합니다.

23.2 1주차 · 사용자와 상황 탐색

Actual·likely user, non-user, 접근 필요 사용자를 포함할 모집과 연구 질문을 정합니다. 이해관계자 요청은 가설로 등록합니다.

23.3 2주차 · 현재 여정과 증거 합성

Before·during·after·support·recovery를 그립니다. 관찰·인용·service data·반대 사례를 evidence ledger에 연결합니다.

23.4 3주차 · 문제·결과 계약

문제·원인·해결책 가설을 분리하고 desired outcome, baseline, target, period, segment, guardrail을 씁니다.

23.5 4주차 · 가장 위험한 가정 검증

Impact×uncertainty가 큰 1~2개 질문에 대해 최소 증거와 stop·reframe rule을 정합니다. 새 증거가 나오면 version을 올립니다.

23.6 Gate · 다음 단계 전달

24개 scenario를 검토하고 미통과 항목이 있으면 기능 목록을 늘리지 않고 evidence contract를 보완합니다. 통과 후보는 M12-02에 넘깁니다.

24. 흔한 실패와 교정

실패	왜 생기나	교정
대표 한 명을 전체 사용자로 봄	접근이 쉬운 사람만 만남	actual·likely·non-user·underserved 분리
persona 이름만 있음	상황과 실제 행동이 없음	trigger·journey·alternative 기록
인용을 증명으로 과대 해석	강한 문장이 기억에 남음	방법·sample·반대 사례·한계 표시
숫자만 있음	분모와 정의가 사라짐	metric contract와 segment 추가
원인을 fact로 씀	설명을 빨리 닫고 싶음	cause hypothesis와 대안 원인 분리
AI 기능이 문제 문장의 주어	solution request에서 시작	기술명을 solution hypothesis로 이동
target만 있음	baseline 수집이 번거로움	현재값·같은 정의·기간 연결
평균만 개선	취약 group이 숨음	no-group-worse guardrail과 segment
조사 활동이 목적	결정과 질문이 분리	answer → continue·reframe·stop 연결
problem_ready를 승인으로 오해	자동 점수에 권한을 부여	residual risk와 실제 승인자 명시

25. M12-02로 넘기는 handoff

다음 매뉴얼은 **기능·비기능 요구사항과 완료 기준 쓰기**입니다. M12-01에서는 상세 요구사항을 미리 만들지 않고 아래 근거 묶음을 넘깁니다.

1. Primary·secondary user, beneficiary, provider, stakeholder 역할
2. Trigger·frequency·channel·access condition이 있는 핵심 상황
3. Current journey·alternative·workaround·support·recovery
4. Source·method·sample·as-of·consent·limitation evidence ledger
5. Anti-solution problem statement와 problem boundary
6. Desired user outcome과 역할별 outcome tension
7. Baseline·target·period·source·cadence·segment·owner
8. Quality·safety·privacy·accessibility·time·cost·reliability guardrail

9. M11-04 cost·latency·license·reliability constraint
10. Fact·inference·assumption·unknown과 top 1~2 next test
11. Stop·reframe rule, residual risk, decision owner
12. Explicit exclusion과 source link·version·change log

25.1 Requirement trace의 시작

```
user evidence
→ problem statement
→ desired outcome
→ guardrail / constraint
→ M12-02 functional·non-functional requirement
→ acceptance criterion
```

---

26. 셀프 테스트 30

질문 1

문제 정의가 기능 목록과 다른 가장 중요한 이유는 무엇인가요?

질문 2

Primary user와 stakeholder를 왜 분리해야 하나요?

질문 3

Likely user를 연구에 포함하는 이유는 무엇인가요?

질문 4

좋은 상황 문장에 필요한 다섯 요소를 쓰세요.

질문 5

현재 대안을 반드시 조사해야 하는 이유는 무엇인가요?

질문 6

Workaround는 왜 단순한 불편의 증거가 아닌가요?

질문 7

관찰과 직접 인용과 해석은 어떻게 다르나요?

질문 8

증거 ledger의 최소 필드 여섯 개를 쓰세요.

질문 9

반대 사례를 삭제하면 어떤 문제가 생기나요?

질문 10

표본 36세션이 전체 시장을 증명하나요?

질문 11

빈도·심각도·도달을 함께 보는 이유는 무엇인가요?

질문 12

증상·문제·원인 가설·해결책 가설을 구분해 예를 드세요.

질문 13

문제 문장에 'AI 봇'을 넣지 않는 이유는 무엇인가요?

질문 14

Desired outcome과 output의 차이는 무엇인가요?

질문 15

Baseline 없는 target이 약한 이유는 무엇인가요?

질문 16

Outcome contract의 최소 요소를 쓰세요.

질문 17

Leading indicator와 outcome measure의 역할 차이는 무엇인가요?

질문 18

평균 성공률이 좋아져도 Gate를 막아야 하는 예를 드세요.

질문 19

Guardrail 다섯 종류를 쓰세요.

질문 20

M11-04에서 넘겨받을 제약은 무엇인가요?

질문 21

Fact와 inference의 차이는 무엇인가요?

질문 22

Assumption과 unknown을 왜 분리하나요?

질문 23

다음 연구 우선순위를 무엇으로 정하나요?

질문 24

Research question이 좋은지 판정하는 한 가지 기준은 무엇인가요?

질문 25

Disconfirming evidence를 미리 쓰는 이유는 무엇인가요?

질문 26

`problem_ready`가 의미하지 않는 것 세 가지를 쓰세요.

질문 27

24개 scenario에서 critical pass 100%를 요구하는 이유는 무엇인가요?

질문 28

문제 정의 version을 올려야 하는 때는 언제인가요?

질문 29

M12-02에 넘기는 최소 묶음을 쓰세요.

질문 30

자신의 문제 정의를 60초에 설명하는 순서를 쓰세요.

---

27. 모범 답안

답 1

문제 정의는 사용자·상황·현재 대안·증거·기대 결과·불확실성을 연결하며 해결책 선택을 열어 둡니다. 기능 목록은 이미 solution space의 선택입니다.

답 2

요청·예산 권한을 가진 stakeholder와 실제 작업을 수행하는 사용자의 목표·부담·성공 기준이 다를 수 있기 때문입니다.

답 3

아직 서비스 사용자가 아니어도 제안된 서비스의 실제 사용자가 될 사람의 현재 행동과 접근 장벽을 배울 수 있기 때문입니다.

답 4

Who, trigger, context/channel, trying to do, barrier/consequence입니다. 빈도와 시간 압박까지 있으면 더 재현하기 쉽습니다.

답 5

사용자가 이미 지불하는 시간·전환·위험과 새 서비스가 이겨야 할 기준을 알 수 있기 때문입니다.

답 6

사용자가 중요하다고 느끼는 목표와 기존 과정의 부족함, 동시에 새 해결책이 바꿔야 할 습관을 보여 주기 때문입니다.

답 7

관찰은 본 행동, 직접 인용은 실제 발화, 해석은 연구자가 붙인 의미입니다. 세 칸을 분리해야 해석이 사실로 둔갑하지 않습니다.

답 8

Source, method, sample, as-of, owner, limitation입니다. 사람 연구라면 consent scope와 보관 경계도 필요합니다.

답 9

패턴의 적용 경계와 대안 설명을 잃어 과도한 일반화와 확신을 만들 수 있습니다.

답 10

아닙니다. 합성 사례의 36세션은 학습용이며 실제 연구에서도 표본·모집·방법의 한계를 함께 설명해야 합니다.

답 11

자주 일어나지만 가벼운 문제와 드물지만 치명적인 문제, 일부에게만 집중되는 배제를 구분하기 위해서입니다.

답 12

증상은 후속 조치 누락, 문제는 여러 채널 재입력 중 완전한 기록을 만들기 어려움, 원인 가설은 책임 확인 단계 부재, 해결책 가설은 확인 흐름 제공입니다.

답 13

AI 봇은 가능한 해결책 하나이며 문제 자체가 아닙니다. 넣으면 대안 탐색과 반증이 어려워집니다.

답 14

Output은 팀이 만든 기능·문서이고 desired outcome은 사용자의 성공·시간·안전·통제에 나타난 변화입니다.

답 15

출발점을 모르므로 변화량·난이도·측정 일관성과 실제 개선 여부를 판단할 수 없습니다.

답 16

Baseline, direction, target, time horizon, measure definition, source, cadence, segment, owner입니다.

답 17

Leading indicator는 결과 전에 방향을 빠르게 알리고 outcome measure는 사용자의 최종 상태 변화를 판정합니다.

답 18

장애 사용자의 성공률이 악화되거나 민감정보 사고가 생기거나 오배정이 허용 상한을 넘는 경우입니다.

답 19

품질, 안전·privacy, 접근성·형평, 시간·지연, 비용·신뢰성·license 중 사례에 맞는 최소 다섯 경계를 둡니다.

답 20

결과 단위당 비용, 품질 floor, latency objective, license 조건, reliability floor와 그 source·as-of·owner입니다.

답 21

Fact는 출처가 있는 관찰·측정이고 inference는 여러 fact를 연결한 해석입니다. inference에는 대안 설명과 confidence가 필요합니다.

답 22

Assumption은 현재 결정에서 참이라고 사용 중인 믿음이고 unknown은 아직 답과 확인이 필요한 질문이기 때문입니다.

답 23

틀렸을 때의 영향과 현재 불확실성을 함께 보고, 둘 다 큰 가정부터 최소 증거로 검증합니다.

답 24

답이 continue·reframe·stop 중 적어도 하나의 결정을 실제로 바꾸는지 확인합니다.

답 25

좋아하는 가정을 확인하는 쪽으로만 자료를 해석하는 편향을 줄이고 중단·재정의 기준을 선명하게 하기 위해서입니다.

답 26

실제 수요 증명, 통계적 일반화, 연구 완료, 법률·정책 승인, 제품 구축 결정 중 세 가지 이상입니다.

답 27

포용·문제/해결책 분리·guardrail·경제 운영 제약·최종 handoff 같은 핵심 경계는 평균 점수로 상쇄할 수 없기 때문입니다.

답 28

새 증거가 사용자·상황·문제·baseline·target·guardrail·가정·결정 중 하나를 실질적으로 바꿀 때입니다.

답 29

사용자·상황, current journey, anti-solution problem statement, desired outcome, baseline·target·guardrail, 제약, 가정·open question, 제외 범위, source·owner입니다.

답 30

누가·언제 → 지금 무엇을 함 → 어떤 증거가 있음 → 어떤 문제가 생김 → 어떤 결과를 기대함 → 무엇을 넘으면 안 됨 → 가장 큰 불확실성과 다음 검증 순서입니다.

---

28. 최종 제출 Checklist

- [ ] 실제·잠재 사용자와 요청자를 분리했다.
- [ ] 사용자·수혜자·제공자·지원자·이해관계자의 결과를 구분했다.
- [ ] 상황·trigger·frequency·channel·access condition을 썼다.
- [ ] Current journey의 before·during·after·support·recovery를 봤다.
- [ ] 현재 대안·workaround·재입력·전환 비용을 기록했다.
- [ ] Evidence마다 source·method·sample·as-of·owner·limitation이 있다.
- [ ] 실제 연구라면 consent·privacy·retention·access 경계를 확인했다.
- [ ] 관찰·인용·해석·의견·반대 사례를 분리했다.
- [ ] 빈도·심각도·도달·분모·baseline을 함께 봤다.
- [ ] 증상·문제·원인 가설·해결책 가설을 다른 칸에 뒀다.
- [ ] Problem statement에 solution term이 없다.
- [ ] Desired outcome은 기능이 아닌 사용자 변화다.
- [ ] Baseline·target·기간·source·cadence·segment·owner가 있다.
- [ ] 품질·안전·privacy·접근성·시간·비용 guardrail이 있다.
- [ ] M11-04 비용·지연·license·reliability 제약을 연결했다.

- [ ] Fact·inference·assumption·unknown을 구분했다.
- [ ] 가장 영향이 크고 불확실한 가정 1~2개를 골랐다.
- [ ] Research question·method·minimum evidence·decision을 연결했다.
- [ ] Disconfirming evidence·stop·reframe rule이 있다.
- [ ] 24개 scenario와 critical 항목을 모두 검토했다.
- [ ] `problem_ready`의 의미와 비의미를 설명할 수 있다.
- [ ] Version·source·as-of·owner·residual risk가 있다.
- [ ] M12-02 handoff와 제외 범위를 작성했다.

29. 요약

문제 정의는 기능을 고르는 회의가 아닙니다. 실제·잠재 사용자의 구체 상황과 현재 대안을 보고, 추적 가능한 증거로 문제와 영향을 설명하고, 사용자의 기대 결과를 baseline·target·guardrail로 계약하는 일입니다.

좋은 문제 정의는 자신감이 넘치는 문서가 아니라 **무엇을 알고 무엇을 추론했고 무엇을 가정하며 무엇을 다음에 확인할지**가 보이는 문서입니다. 그래서 새 증거가 나오면 version이 올라가고, 필요하면 reframe하거나 stop할 수 있습니다.

M12-02에서는 이 근거를 기능·비기능 요구사항과 완료 기준으로 변환합니다. Problem statement와 outcome에 연결되지 않는 요구사항은 먼저 존재 이유를 다시 묻습니다.

30. 공식 참고 자료

- ISO 9241-210:2019 Human-centred design for interactive systems
- Design Council Framework for Innovation
- Design Council History of the Double Diamond
- GOV.UK User research in discovery
- GOV.UK Start by learning user needs
- GOV.UK Find user research participants
- GOV.UK Getting informed consent for user research
- GOV.UK Plan user research for your service
- GOV.UK Capturing research questions
- GOV.UK Analyse a research session
- GOV.UK Using data to improve your service

- GOV.UK Define what success looks like and publish performance data
- Google Research HEART framework paper
- W3C WAI Involving Users in Evaluating Web Accessibility

> 공식 자료는 개념과 연구·측정 원칙의 기준입니다. 실제 연구, 개인정보 처리, 접근성, 법률·윤리·정책, 제품 투자 판단은 적용 시점의 조직 절차와 권한 있는 담당자의 검토를 따릅니다.