



AI를 활용한 구조 설계

- 고품질 소프트웨어의 효율적 개발



Yong Jin, Cho (drajin.cho@bosornd.com)

AI 시대, 구조 설계가 필요한가?



AI 도구의 발전

놀라운 속도로 즉시 실행 가능한 코드를 생성합니다.

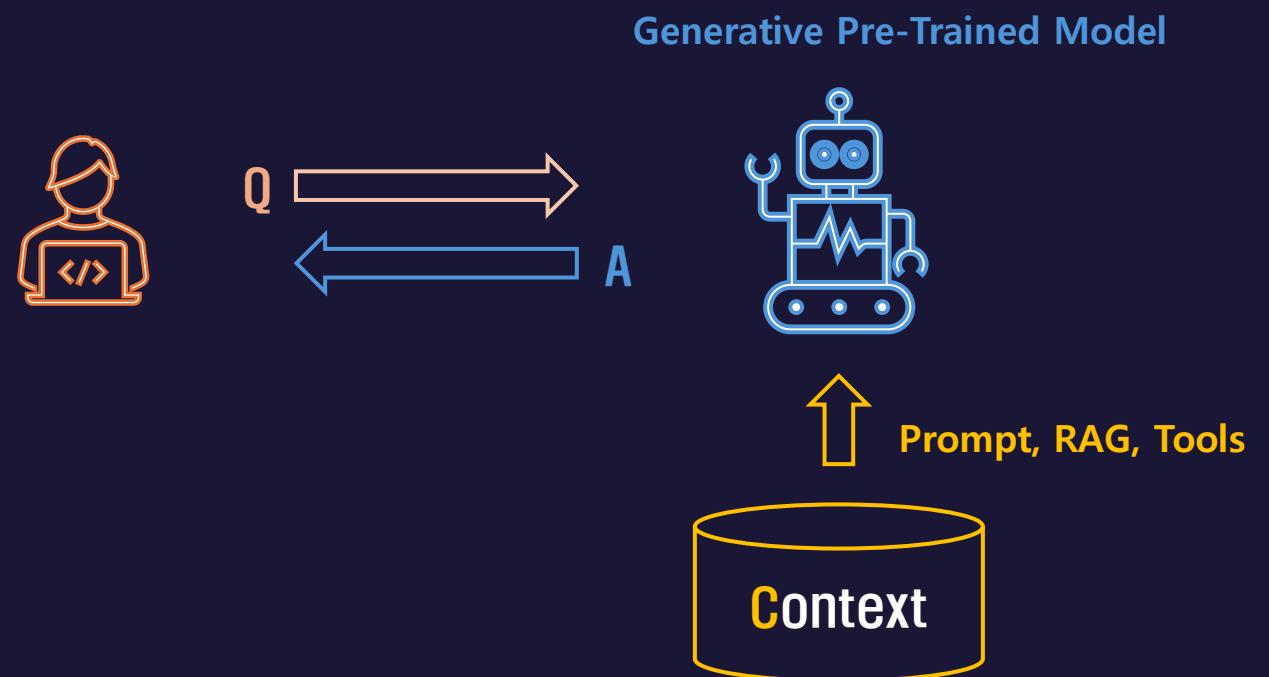


근본적인 질문

여전히 구조 설계 활동이 필요한가?



LLM은 기억하지 못한다.

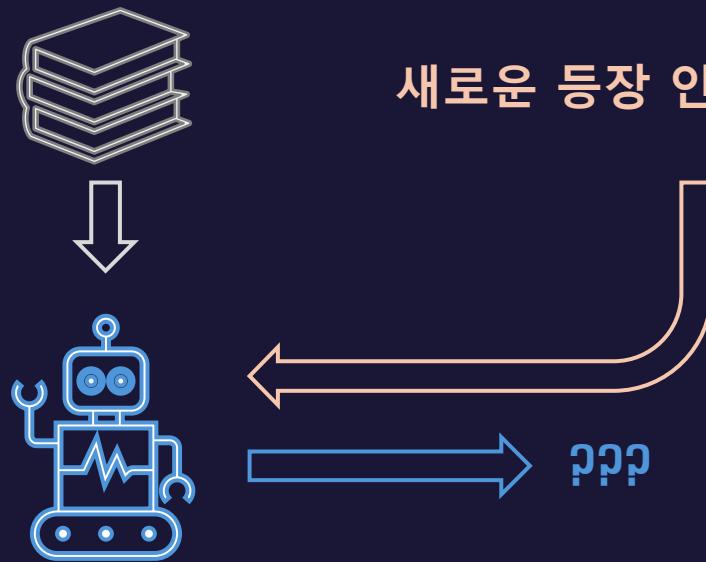


대규모 시스템을 다루기 위해서는
체계적인 Context Model이 필요하다.





소설 쓰기의 예



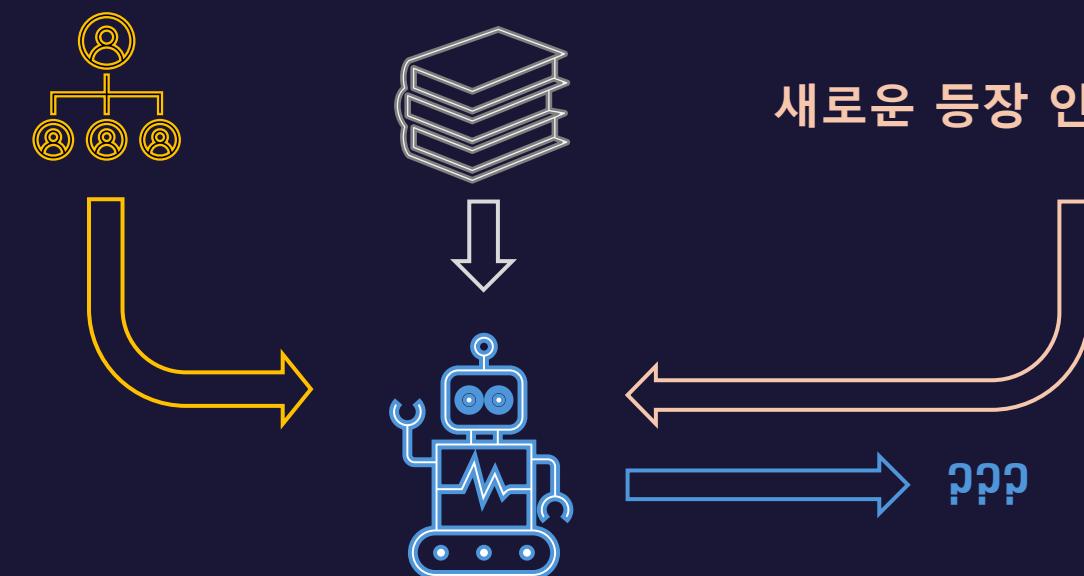
↪ 제공해야 하는 정보의 양 증가 → 비용 증가 💰

↪ 복잡도 증가 → 처리 오류 가능성 증가 💰

↪ 국소적 해결 → 기술 부채 증가 💰



소설 쓰기의 예

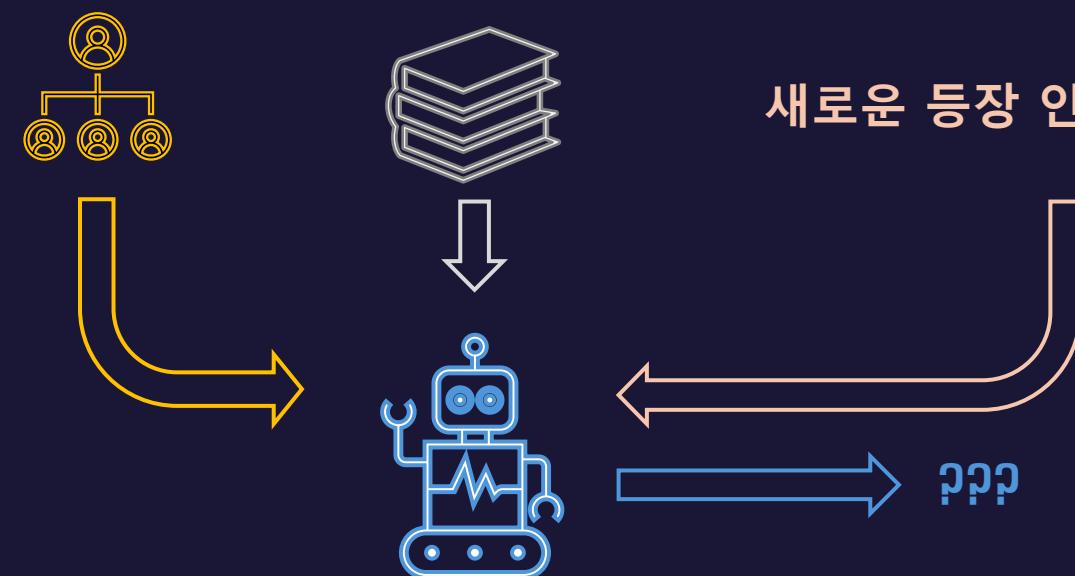


1. 등장인물 목록 - 등장인물, 관계, 역할
2. 시간적 구조 (플롯) - 사건의 순서와 흐름
3. 공간적 구조 - 배경과 장소 설정
4. 서술 구조 - 화자와 관점, 스타일 규칙
5. 주제와 모티프 - 중심 주제와 상징
6. 세계관 - 규칙과 법칙

체계적인 Context Model
→ 구조



소설 쓰기의 예



분석 용이

효율적 작업

정확한 수정

비용 절감

AI 코드 생성의 한계



국소적 해결책 제시

전체 시스템의 맥락을 고려하지 않는다.



구조적 일관성 부족

점진적으로 기술 부채가 누적된다.



품질 요구사항 충족 미흡

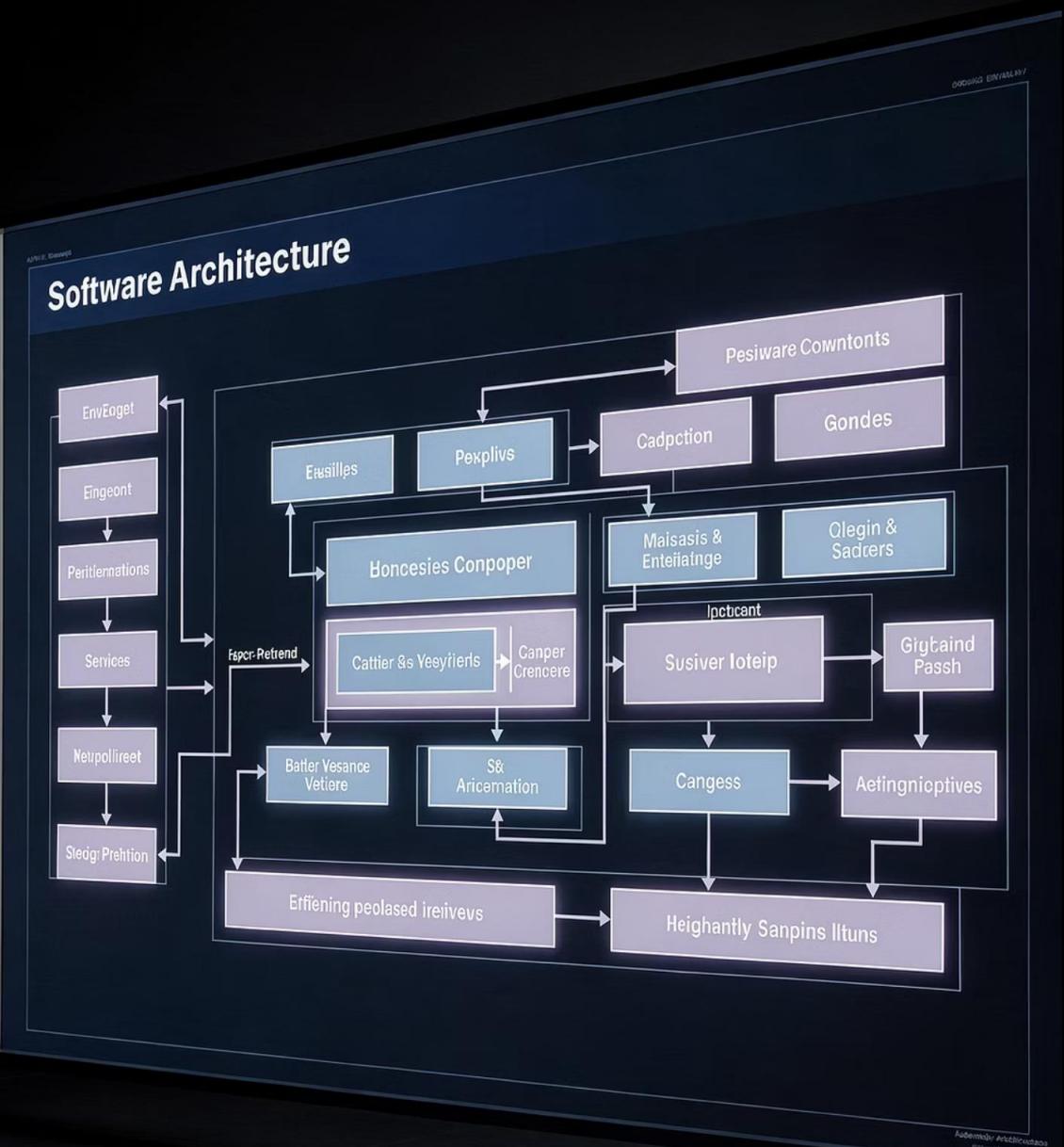
지속 가능성과 시장 경쟁력이 약화된다.



전체 코드 기반 컨텍스트

LLM 사용 비용이 시스템 규모에 따라 급격히 증가한다.





구조 설계가 필요한 이유



전체 시스템 맥락 제공

시스템 차원의 최적 해결책을 제시한다.



구조적 일관성 확보

설계 원칙 준수로 기술 부채를 최소화 한다.



품질 요구사항 반영

최적 개발로 비즈니스 경쟁력이 강화된다.

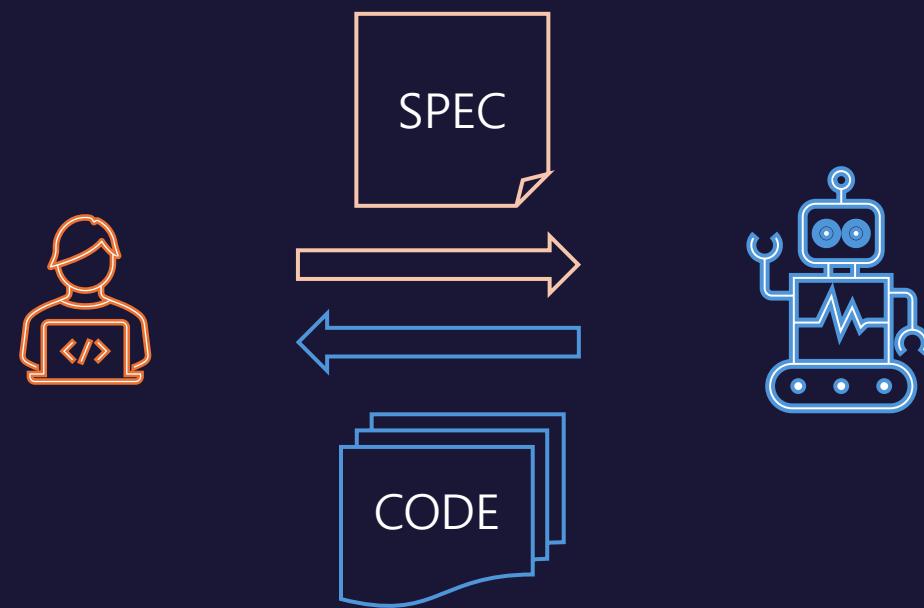


AI 코드 생성의 명확한 가이드 제공

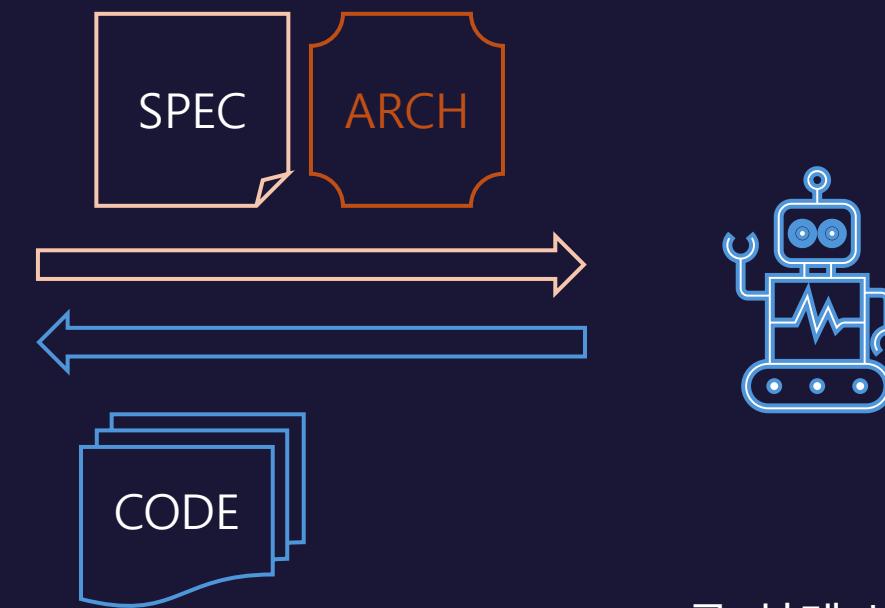
효율적 개발이 가능하다.

AI를 활용한 구조 설계

- BEFORE -

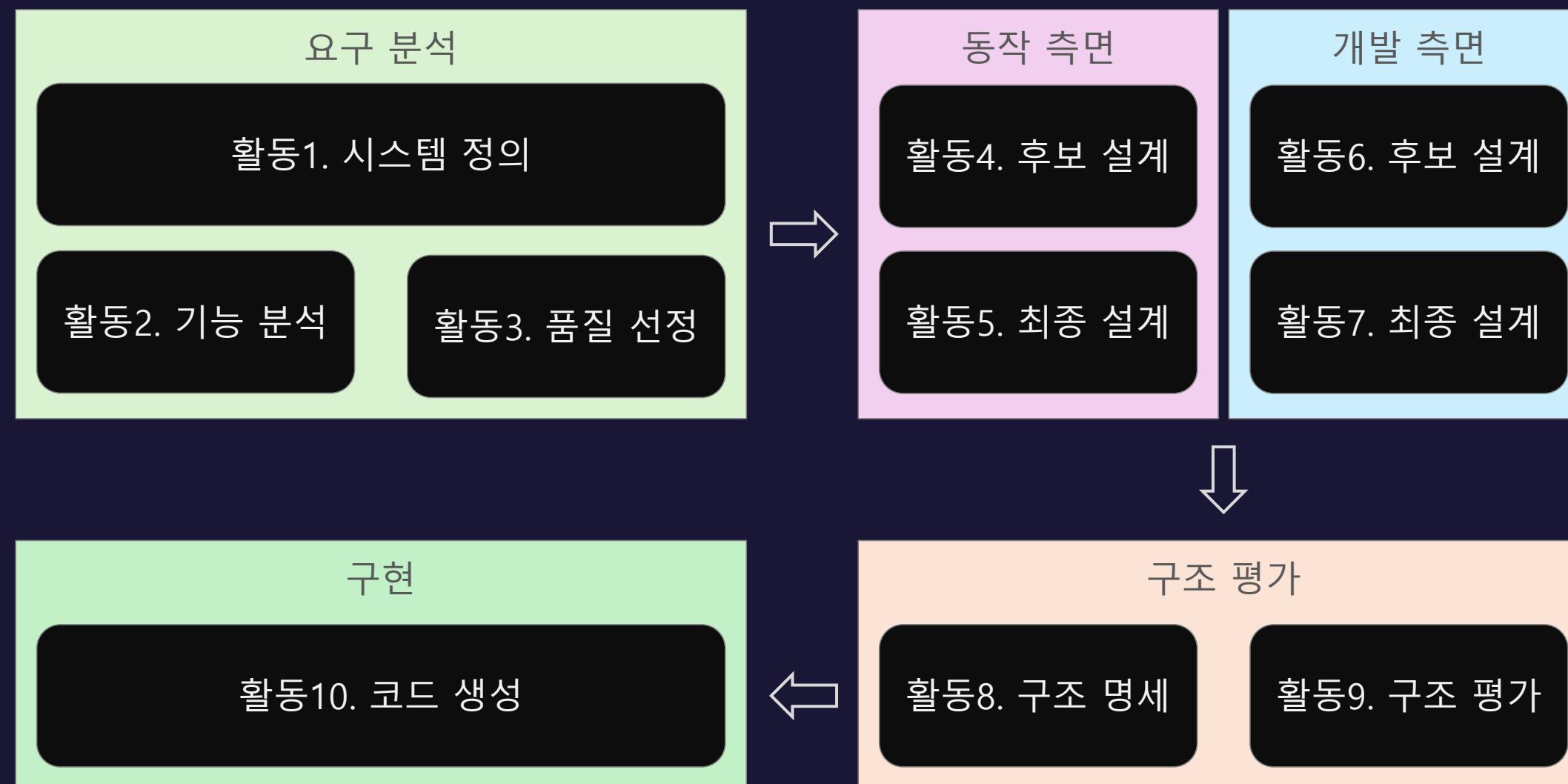


- AFTER -



ARCH를 설계/변경한다.
설계/변경된 ARCH에 근거해
CODE를 생성/변경한다.

구조 설계 프로세스 및 산출물



구조 설계 프로세스 및 산출물

활동	산출물
활동1. 시스템 정의	system.md, business.md
활동2. 기능 분석	usecases.md, usecase/UC-nnn.md, domain/model.md, domain/UC-nnn.md
활동3. 품질 요구사항 선정	qualities.md, quality/scenarios.md, quality/evaluations.md, quality/QS-nnn.md
활동4. 후보 구조 설계 [동작]	candidate/candidates.md, msa.md, solutions.md, QS-nnn.md, CA-nnn.md
활동5. 최종 구조 설계 [동작]	decision/decisions.md, decision/evaluations.md
활동6. 후보 구조 설계 [개발]	candidate/candidates.md, packages.md, frameworks.md, QS-nnn.md, CA-nnn.md
활동7. 최종 구조 설계 [개발]	decision/decisions.md, decision/evaluations.md
활동8. 구조 명세	architecture.md
활동9. 구조 평가	evaluation/decisions.md, evaluation/evaluation.md
활동10. 코드 생성	src/*

AI를 활용한 구조 설계 실습 환경



실습 자료: <https://github.com/bosornd/arch-with-ai>

참고 자료:
<https://github.com/YongJin-Cho/arch-with-ai>



활동1. 시스템 정의

- 시스템 범위 설정
- 비즈니스 드라이버 식별



Yong Jin, Cho (drajin.cho@bosornd.com)

활동1. 시스템 정의

- 구조 설계의 첫 단계
- 개발할 시스템의 목적, 범위, 경계, 제약사항 정의
- 비즈니스 목표와 드라이버 식별



활동1. 시스템 정의

agent	system-definer
command	define-system
task	<ul style="list-style-type: none">- 사용자 요구사항 수집 및 분석- 시스템 목적 및 범위 정의- 시스템 경계 설정- 제약사항 식별 및 문서화



system.md

agent	business-analyzer
command	analyze-business
task	<ul style="list-style-type: none">- system.md 분석 및 비즈니스 맥락 파악- 주요 이해관계자 분석- 비즈니스 목표 및 드라이버 식별



business.md

에이전트: system-definer

/agentk/define-system 서비스에서 발생하는 count를 통합 관리하는 시스템을 개발하는 과제를 정의해줘.



사용자 요구사항 수집 및 분석

- 시스템이 해결하려는 문제 또는 목적 파악
- 시스템의 주요 기능 영역 파악
- 시스템의 범위 (포함/제외) 파악
- 기술적/비기술적 제약사항 파악

시스템 정의 문서 작성

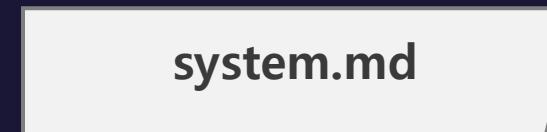
- 시스템 개요 (시스템 이름, 목적, 문제)
- 시스템 범위 (포함/제외 기능, 범위 설정 근거)
- 시스템 경계 (Mermaid 다이어그램)
- 제약사항 (기술적, 비기술적, 규제 및 규정)



system.md

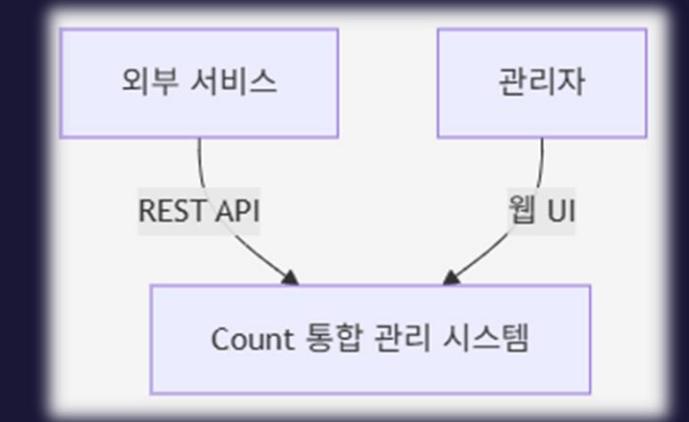
에이전트: system-definer

/agentk/define-system 서비스에서 발생하는 count를 통합 관리하는 시스템을 개발하는 과제를 정의해줘.



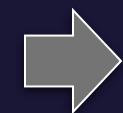
체크 포인트

1. 시스템 경계의 명확성
2. 시스템 범위의 명확성
3. 제약사항의 완전성



/agentk/define-system Kubernetes 환경에서 MSA로 개발하는 것을 제약사항으로 명시해줘.

에이전트: business-analyzer



/agentk/analyze-business 핵심 비즈니스 드라이버를 식별해줘.

비즈니스 정보 수집 및 분석

- 비즈니스 목표 (단기/중기/장기) 파악
- 비즈니스 드라이버 (경쟁력 요소 등) 파악
- 주요 이해관계자 및 역할 파악
- 비즈니스 가치 및 우선순위 파악

비즈니스 분석 문서 작성

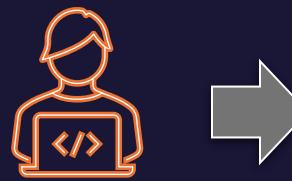
- 비즈니스 개요 (배경, 목적, 가치 제안)
- 비즈니스 목표 (단기/중기/장기, 우선순위)
- 비즈니스 드라이버 (경쟁력 요인)
- 이해관계자 (주요 이해관계자 관심사)



business.md

에이전트: business-analyzer

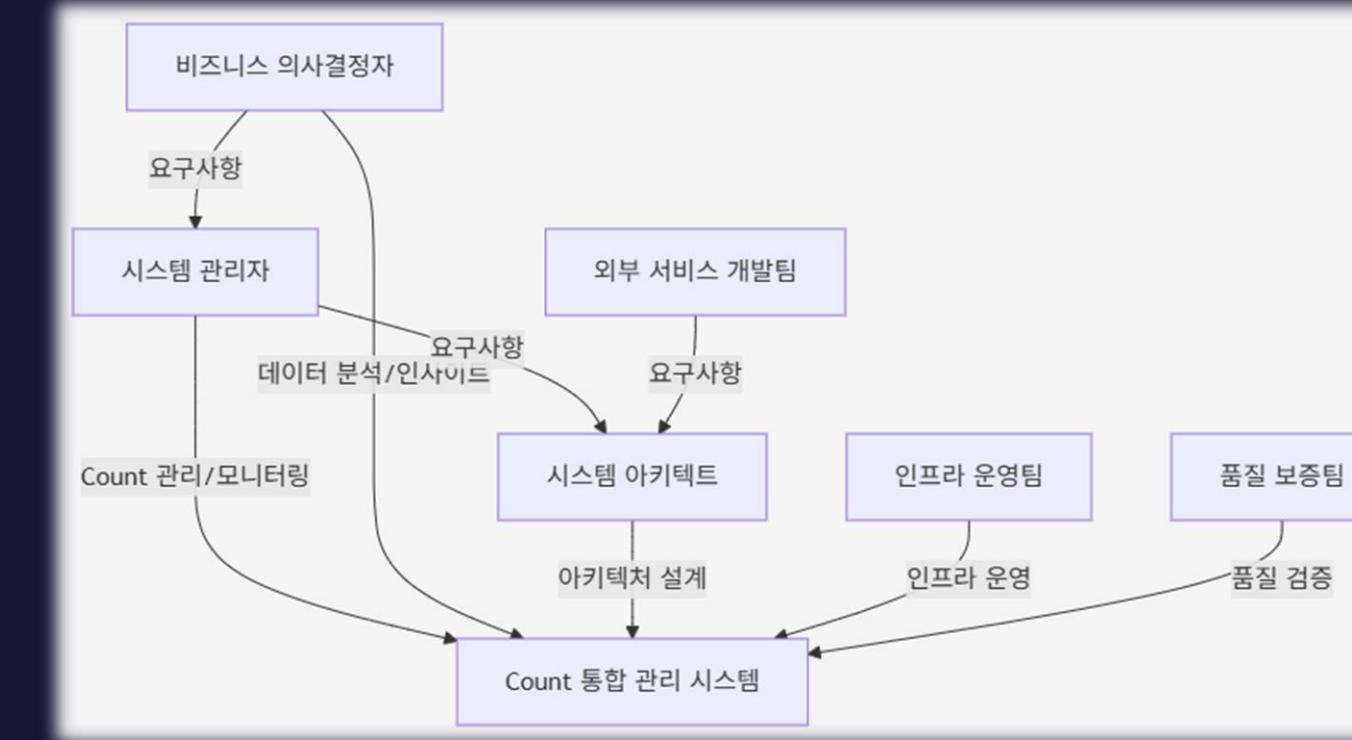
/agentk/analyze-business 핵심 비즈니스 드라이버를 식별해줘.



business.md

체크 포인트

1. 비즈니스 목표 및 드라이버의 명확성
2. 시스템 범위와 비즈니스 목표의 일관성



/agentk/analyze-business 현재는 대량의 Count 저장 요청에 대한 효율적인 관리가 가장 중요해.



활동2. 기능적 요구사항 분석

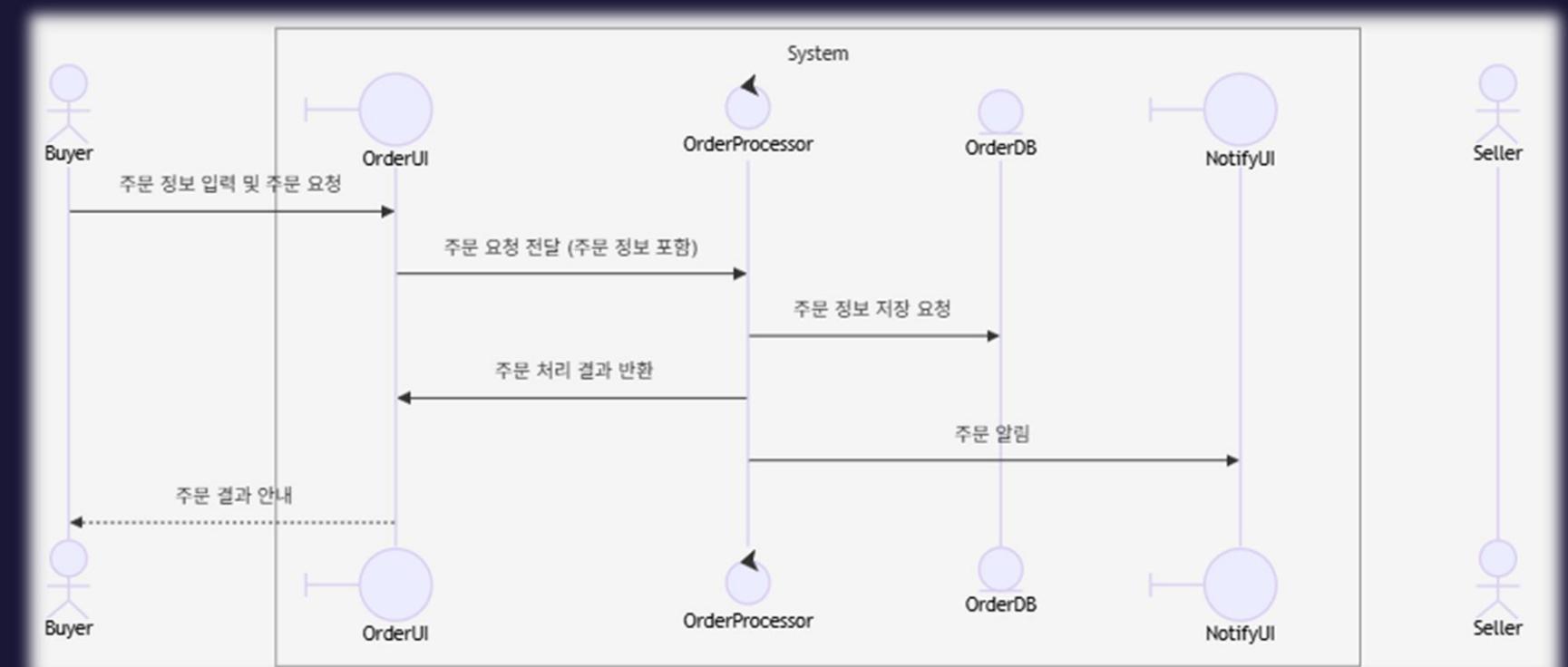
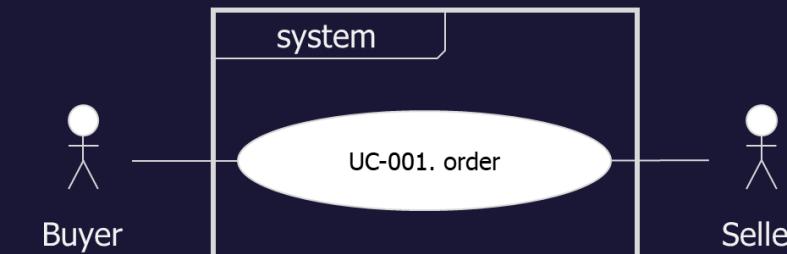
- 핵심 Use Case 식별
- Use Case 명세
- 도메인 모델 정립



Yong Jin, Cho (drajin.cho@bosornd.com)

활동2. 기능적 요구사항 분석

- Use Case 식별
- Use Case 명세
- Use Case 분석
- 도메인 모델 정립



활동2. 기능적 요구사항 분석

agent	usecase-extractor
command	extract-usecases
task	<ul style="list-style-type: none">- system.md와 business.md 분석- 시스템 범위에서 주요 기능 식별- Use Case 목록 작성 및 분류- usecases.md 문서 생성



usecases.md

agent	usecase-specifier
command	specify-usecase
task	<ul style="list-style-type: none">- usecases.md 분석- 각 Use Case별 상세 명세 작성- Actor, 사전/사후 조건, 시나리오- UC-nnn-{title}.md 문서 생성



UC-nnn-{title}.md

활동2. 기능적 요구사항 분석

agent	domain-modeler
command	design-model
task	<ul style="list-style-type: none">- usecase/UC-nnn-{title}.md 파일 분석- Use Case별 도메인 컴포넌트 식별- Use Case별 시스템 내부 동작 명세- domain/UC-nnn-{title}.md 문서 생성



UC-nnn-{title}.md

agent	domain-modeler
command	design-model
task	<ul style="list-style-type: none">- 컴포넌트 책임 분석- 통합 도메인 모델 작성 (중복 제거)- domain/model.md 문서 생성



model.md

에이전트: usecase-extractor



/agentk/extract-usecases 중요한 Use Case를 식별해줘.

Use Case 식별

- 시스템 범위에서 포함된 기능 영역 분석
- 각 기능 영역에서 주요 기능 식별
- 사용자 유형별로 필요한 기능 식별
- 핵심 ASR Use Case 선정

Use Case 목록 문서 작성

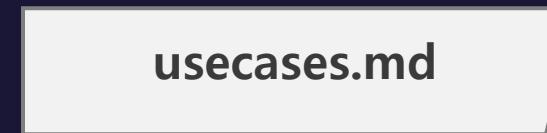
- Use Case 개요 (목적, 식별 기준)
- Use Case 목록 (ID, 제목, 설명, 주요 액터)
- Use Case 다이어그램 작성
- 핵심 ASR Use Case 목록



usecases.md

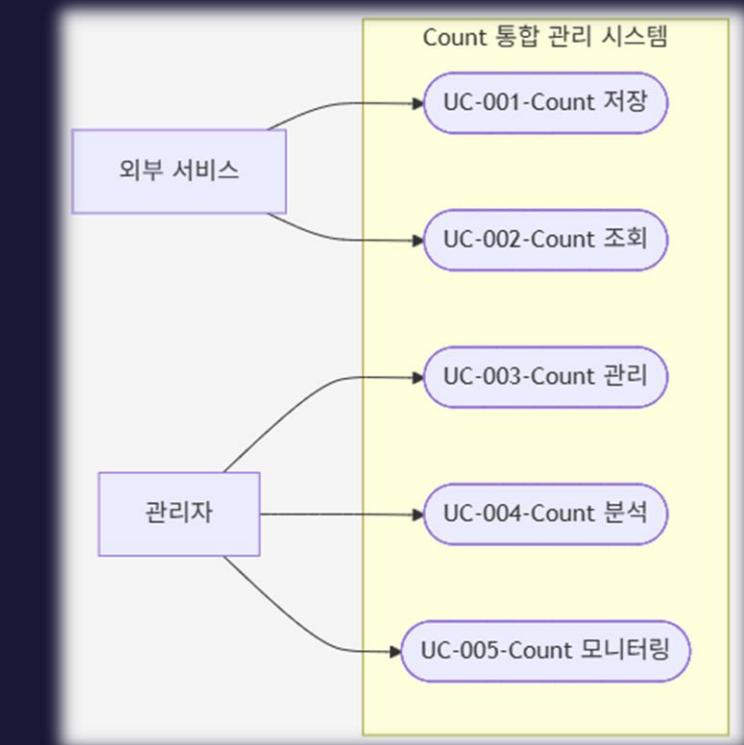
에이전트: usecase-extractor

/agentk/extract-usecases 중요한 Use Case를 식별해줘.



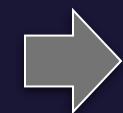
체크 포인트

1. Use Case 목록의 완전성
2. ASR 식별의 정확성



/agentk/extract-usecases 대시보드에 Count 값을 모니터링 하는 기능을 추가해줘.

에이전트: usecase-specifier



/agentk/specify-usecase UC-001을 명세해줘.

Use Case 상세 명세

- Use Case 기본 정보 확인 (ID, 제목, 설명)
- 액터 정의 (Primary Actor, Secondary Actor)
- 사전조건 및 사후조건 정의
- 주요 / 대안 / 예외 시나리오 정의



Use Case 상세 명세 문서 작성

- Use Case에 대해 UC-nnn-{title}.md 파일 작성



UC-nnn-{title}.md

에이전트: usecase-specifier

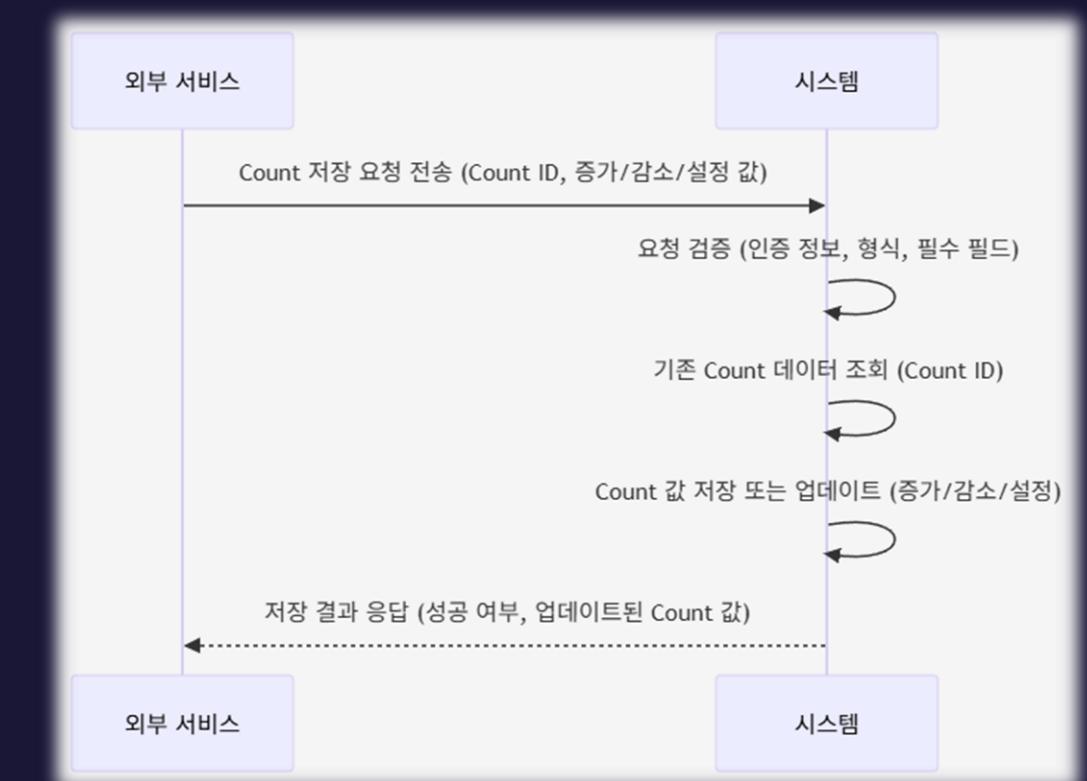
/agentk/specify-usecase UC-001을 명세해줘.



UC-nnn-{title}.md

체크 포인트

1. Use Case 명세의 완전성
2. 동작 시나리오의 구체성 / 명확성



/agentk/specify-usecase Count 분석을 위해 변경 이력을 저장하도록 수정해줘.

에이전트: domain-modeler

/agentk/design-domain Use Case를 분석해서 도메인 모델을 정립해줘.



Use Case 분석

- usecase/UC-nnn-{title}.md 파일 분석
- Use Case별 도메인 컴포넌트 식별
- Use Case별 시스템 내부 동작 명세
- domain/UC-nnn-{title}.md 문서 생성

도메인 모델 정립

- 컴포넌트 책임 분석
- 통합 도메인 모델 작성 (중복 제거)
- domain/model.md 문서 생성



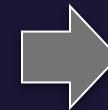
UC-nnn-{title}.md



domain.md

에이전트: domain-modeler

/agentk/design-domain Use Case를 분석해서 도메인 모델을 정립해줘.

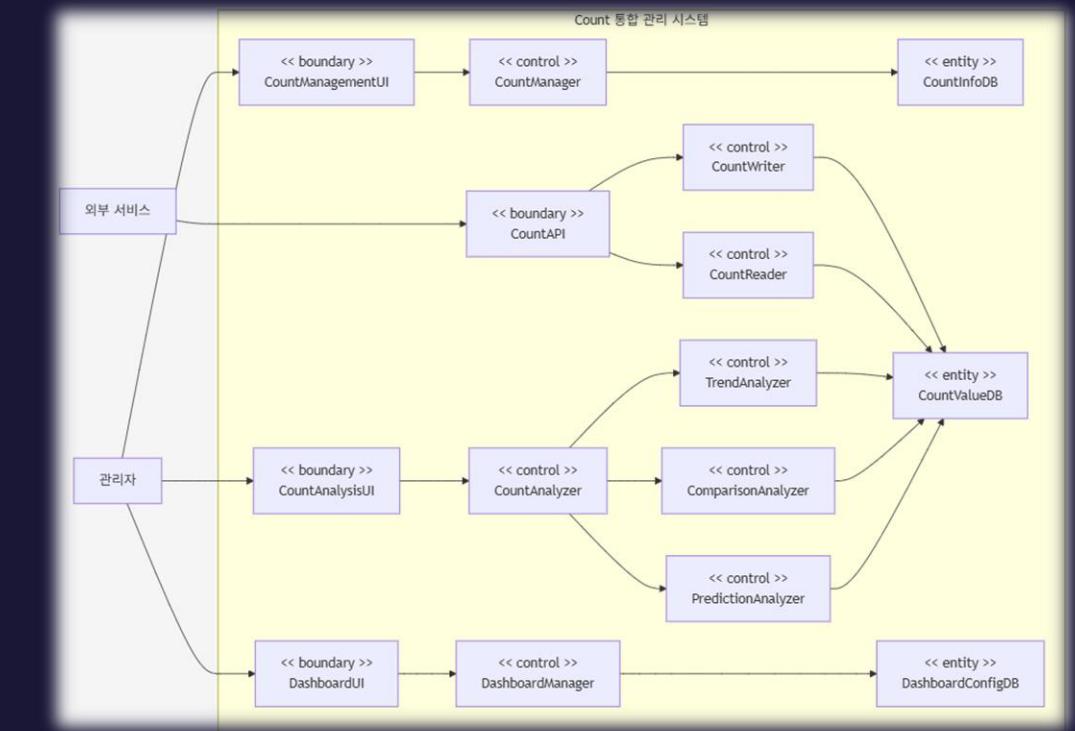


system.md

UC-nnn-{title}.md

체크 포인트

1. 도메인 모델의 완전성
2. 컴포넌트 분류의 명확성
3. 컴포넌트 역할/책임의 명확성



/agentk/design-domain 구체적인 Count 분석을 처리하는 컨트롤 컴포넌트를 식별해줘. 예) TrendAnalyzer



활동3. 품질 요구사항 선정

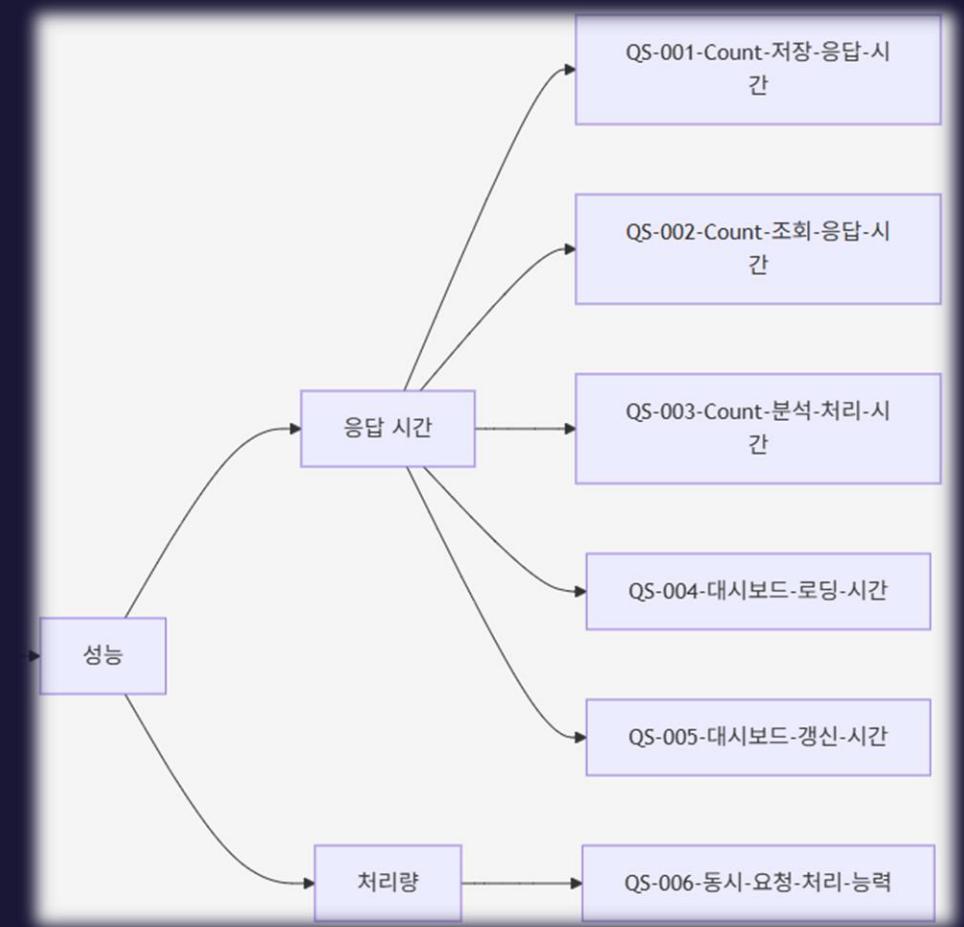
- 품질 시나리오 생성 및 명세
- 품질 시나리오 평가
- 품질 요구사항 선정



Yong Jin, Cho (drajin.cho@bosornd.com)

활동3. 품질 요구사항 선정

- 품질 시나리오 생성
- 품질 시나리오 평가
- 품질 시나리오 선정
 - NFR (Non-Functional Requirement)
 - QA (Quality Attribute)



활동3. 품질 요구사항 선정

agent	quality-elicitor
command	elicit-scenarios
task	<ul style="list-style-type: none">- usecases.md 분석- Utility Tree 구성 (품질 속성 계층 구조)- 품질 시나리오 생성- quality/scenarios.md 문서 생성



scenarios.md

agent	quality-specifier
command	specify-scenario
task	<ul style="list-style-type: none">- quality/scenarios.md 분석- 각 품질 시나리오별 상세 명세 작성- 측정 방법 정의- quality/QS-nnn-{title}.md 문서 생성



QS-nnn-{title}.md

활동3. 품질 요구사항 선정

agent	quality-evaluator
command	evaluate-scenarios
task	<ul style="list-style-type: none">- quality/QS-nnn.md 분석- 각 품질 시나리오별 중요도/난이도 평가- 비즈니스 영향도 분석- quality/evaluations.md 문서 생성



evaluations.md

agent	quality-selector
command	select-scenarios
task	<ul style="list-style-type: none">- quality/evaluations.md 분석- 주요 품질 요구사항 선정- 우선순위 설정- qualities.md 문서 생성



qualities.md

에이전트: quality-elicitor



/agentk/elicit-scenarios 품질 시나리오를 생성해줘.

품질 시나리오 생성

- system.md, business.md, usecases.md 분석
- 품질 시나리오 분류
- Utility Tree 구성
- 품질 시나리오 생성



품질 시나리오 목록 문서 작성

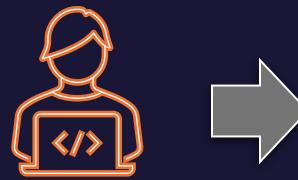
- 품질 시나리오 개요 (목적, 생성 기준)
- Utility Tree 다이어그램
- 품질 시나리오 목록 (ID, 제목, 설명, 측정)
- quality/scenarios.md 문서 생성



scenarios.md

에이전트: quality-elicitor

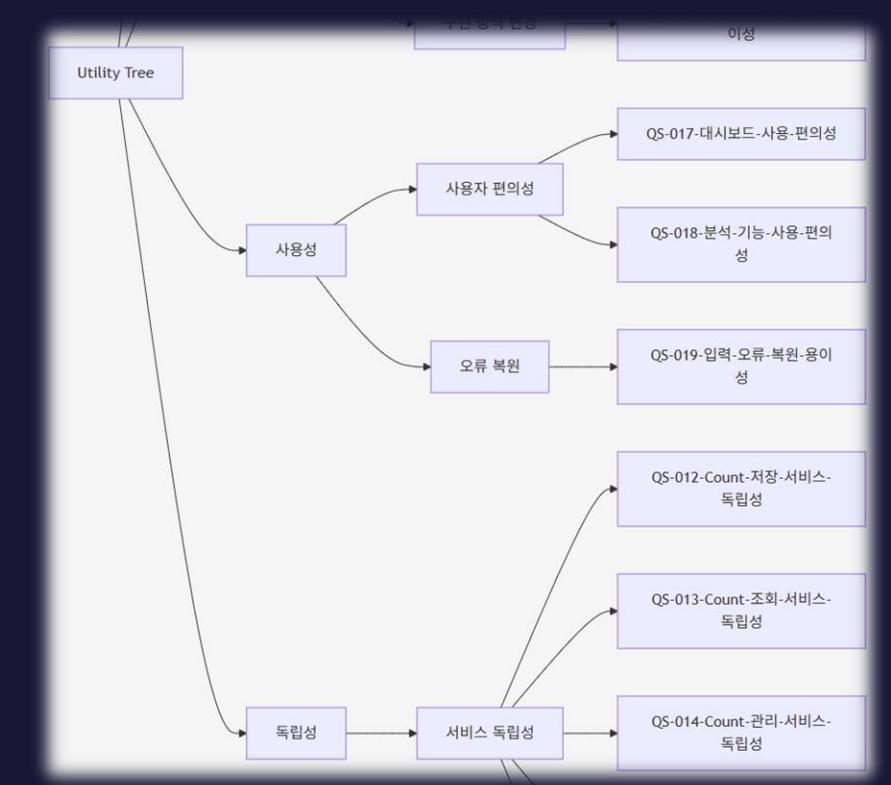
/agentk/elicit-scenarios 품질 시나리오를 생성해줘.



usecases.md

체크 포인트

1. 품질 시나리오 목록의 완전성
2. 품질 시나리오의 측정 가능성



/agentk/elicit-scenarios 서비스 독립성에 관한 품질 시나리오를 추가해줘.

에이전트: quality-specifier

/agentk/specify-scenario QS-001의 상세 명세를 작성해줘.



품질 시나리오 상세 명세

- 품질 시나리오의 기본 정보 확인
- 측정 환경 및 시나리오 구체화
- 측정 방법 구체화
- 품질 시나리오 상세 명세



품질 시나리오 상세 명세 문서 작성

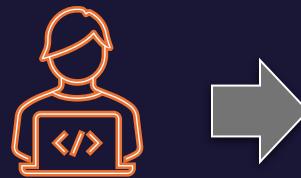
- 품질 시나리오에 대해 QS-nnn-{title}.md 파일 작성



QS-nnn-{title}.md

에이전트: quality-specifier

/agentk/specify-scenario QS-001의 상세 명세를 작성해줘.



QS-nnn-{title}.md

체크 포인트

1. 품질 시나리오 명세의 완전성
2. 품질 시나리오의 구체성 / 명확성
3. 품질 시나리오의 측정 가능성

Quality Scenario ID

QS-001

제목

Count 저장 응답 시간

설명

외부 서비스가 Count 값을 저장하거나 증가/감소시킬 때의 응답 시간

품질 속성

성능 (응답 시간)

환경

시스템 상태

- 시스템이 정상 가동 중
- Count 통합 관리 시스템이 Kubernetes 환경에서 MSA로 배포되어 있음
- 데이터베이스가 정상 동작 중

/agentk/specify-scenario QS-001의 명세에서 변경된 값이 저장되는 부분은 제거해줘.

에이전트: quality-evaluator

/agentk/evaluate-scenarios 품질 시나리오를 평가해줘.



품질 시나리오 평가

- 품질 시나리오 분석
- 품질 시나리오 중요도/난이도 평가
- 비즈니스 영향도 및 우선순위 분석
- 품질 시나리오 평가

품질 시나리오 평가 문서 작성

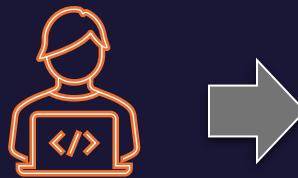
- 품질 시나리오 평가 개요 (목적, 평가 기준)
- 품질 시나리오 평가 결과 작성
- 비즈니스 영향도 및 우선순위 분석 결과 작성
- quality/evaluations.md 문서 작성



evaluations.md

에이전트: quality-evaluator

/agentk/evaluate-scenarios 품질 시나리오를 평가해줘.



evaluations.md

체크 포인트

1. 품질 시나리오 평가의 정확성

QS-001-Count-저장-응답-시간

- 중요도: 매우 높음
 - 평가 근거: 외부 서비스 개발팀의 핵심 관심사로, Count 저장 기능은 시스템의 기본 기능이며, 응답 시간은 외부 서비스의 성능에 직접적인 영향을 미침. 마이크로서비스 환경에서 빠른 응답 시간은 경쟁력의 핵심 요소임.
- 난이도: 중간
 - 평가 근거: 기본적인 CRUD 작업으로 표준적인 데이터베이스 작업과 API 처리로 구현 가능. Kubernetes/MSA 환경에서 일반적인 성능 최적화 기법 적용 가능.
- 비즈니스 영향도: 매우 높음
 - 평가 근거: Count 데이터 일관성 보장 및 기본 CRUD 기능 제공은 최우선 비즈니스 목표임. 외부 서비스들이 Count 데이터를 효율적으로 활용할 수 있는 인프라 제공은 비즈니스 가치 제안의 핵심임.

/agentk/evaluate-scenarios 사용성에 관한 품질의 중요성은 그다지 높지 않은 것 같아.

에이전트: quality-selector



/agentk/select-scenarios 품질 요구사항을 선정해줘.

품질 요구사항 선정

- 품질 시나리오 평가 결과 분석
- 비기능적 요구사항 선정 (허용치)
- 품질 속성 선정 (우선순위)
- 품질 요구사항 요약

품질 요구사항 문서 작성

- 품질 요구사항 개요 (목적, 선정 기준)
- 비기능적 요구사항 목록 작성
- 품질 속성 목록 작성
- qualities.md 문서 작성



qualities.md

에이전트: quality-selector

/agentk/select-scenarios 품질 요구사항을 선정해줘.



qualities.md

체크 포인트

1. NFR의 허용치 적절성
2. QA의 우선순위 적절성
3. 비즈니스 드라이버와의 일치성

NFR 요약

- 총 4개의 비기능적 요구사항 선정
- 모든 NFR을 만족해야 과제 성공 및 시장 진입 가능
- NFR 목록:
 - NFR-001: Count 저장 응답 시간 < 100ms
 - NFR-002: Count 조회 응답 시간 < 100ms
 - NFR-003: 동시 요청 처리량 >= 1000 RPS
 - NFR-004: 대시보드 로딩 시간 < 3초

QA 요약

- 총 12개의 품질 속성 선정
- 우선순위에 따라 의사결정 수행
- 트레이드오프 발생 시 우선순위 높은 QA를 우선
- 최우선 QA (우선순위 1-4):
 - QA-001: Count 저장 응답 시간 최소화
 - QA-002: Count 조회 응답 시간 최소화
 - QA-003: Count 저장 서비스 독립성 최대화
 - QA-004: Count 조회 서비스 독립성 최대화

/agentk/select-scenarios 사용 편의성 관련 QS-017, QS-018은 품질 속성에서 제거해줘.



활동4. 후보 구조 설계 [동작]

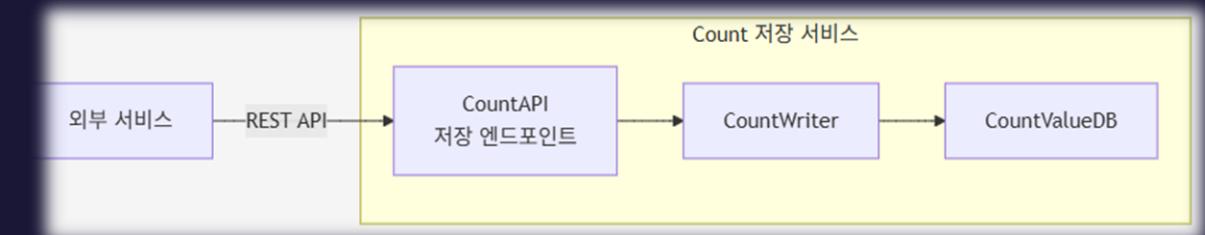
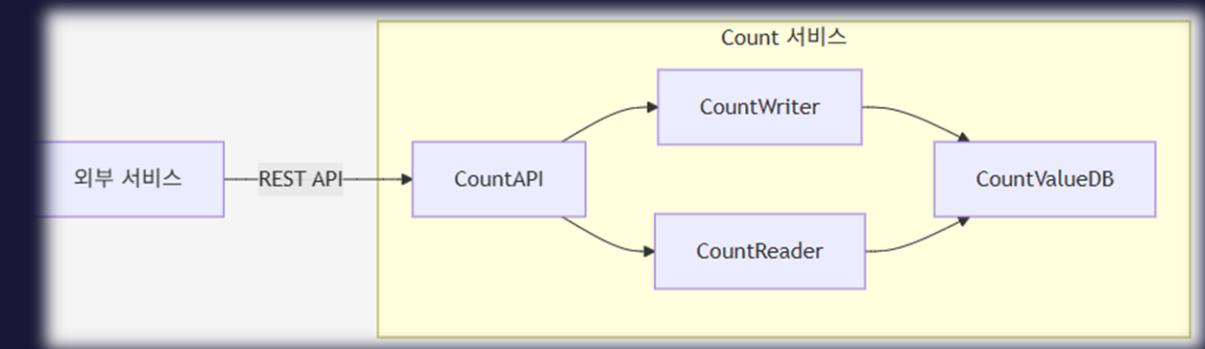
- 동작 측면의 후보 구조 설계

Y Yong Jin, Cho (drajin.cho@bosornd.com)

활동4. 후보 구조 설계 [동작]

▪ 동작 측면의 후보 구조 설계

- 배치 관점: 마이크로 서비스
 - 품질 관점: 성능, 사용성, 가용성 등
 - 기술 솔루션 관점: 데이터베이스 등
- ⋮



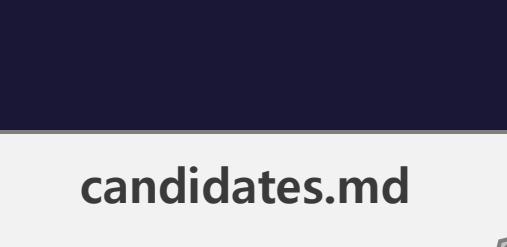
활동4. 후보 구조 설계 [동작]

agent	msa-architect
command	design-msa
task	<ul style="list-style-type: none">- 서비스 분할/통합 문제 식별- 서비스 경계 설계 후보 구조 제시- 서비스 통합 방식 설계 후보 구조 제시- 후보 구조의 장단점 분석

agent	solution-architect
command	select-solutions
task	<ul style="list-style-type: none">- 기술 솔루션 선택 문제 식별- 기술 솔루션 후보 검토 및 비교- 기술 솔루션 선택 후보 구조 제시- 후보 구조의 장단점 분석



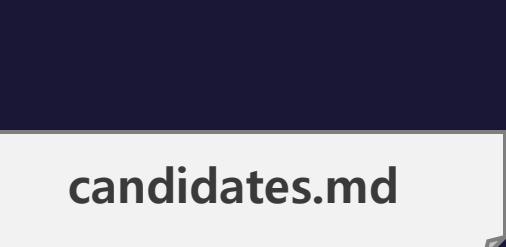
msa.md



candidates.md



solutions.md



candidates.md

활동4. 후보 구조 설계 [동작]

agent	performance-architect
command	design-performance
task	<ul style="list-style-type: none">- 성능 시나리오 분석 및 문제 식별- 성능 패턴 및 전술 적용 검토- 성능 최적화 후보 구조 제시- 후보 구조의 장단점 분석 <p>...</p>



QS-nnn-{title}.md

candidates.md

에이전트: msa-architect



/agentk/design-msa msa 관점에서 후보 구조를 설계해줘.

후보 구조 설계

- 서비스 분할 및 통합 문제 식별
- 설계 패턴 및 전술 적용 검토
- 후보 구조 설계
- 설계 과정 가시화 (마인드 맵)

후보 구조 목록 문서 작성

- 후보 구조 목록 작성 (ID, 제목, 분류 등)
- 상충/종속 관계 명시
- candidate/candidates.md 문서 간신

msa.md

candidates.md

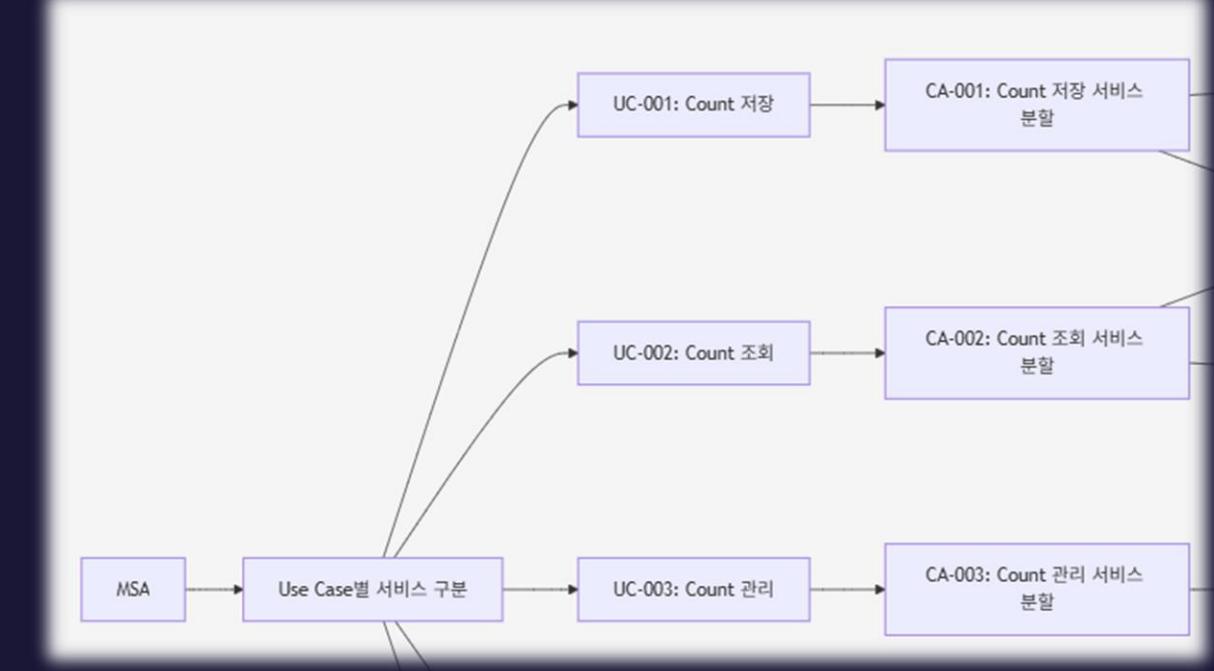
에이전트: msa-architect

/agentk/design-msa msa 관점에서 후보 구조를 설계해줘.



체크 포인트

1. 후보 구조 설계의 완전성
2. 후보 구조 분석의 정확성



/agentk/design-msa Count 값의 저장/조회를 처리하는 서비스를 통합하는 후보 구조를 검토해줘.

에이전트: performance-architect

/agentk/design-performance QS-001을 분석하고 후보 구조를 설계해줘.



후보 구조 설계

- 성능 시나리오 문제 식별
- 설계 패턴 및 전술 적용 검토
- 후보 구조 설계
- 설계 과정 가시화 (마인드 맵)

후보 구조 목록 문서 작성

- 후보 구조 목록 작성 (ID, 제목, 분류 등)
- 상충/종속 관계 명시
- candidate/candidates.md 문서 간신



QS-nnn-{title}.md



candidates.md

에이전트: performance-architect

/agentk/design-performance QS-001을 분석하고 후보 구조를 설계해줘.

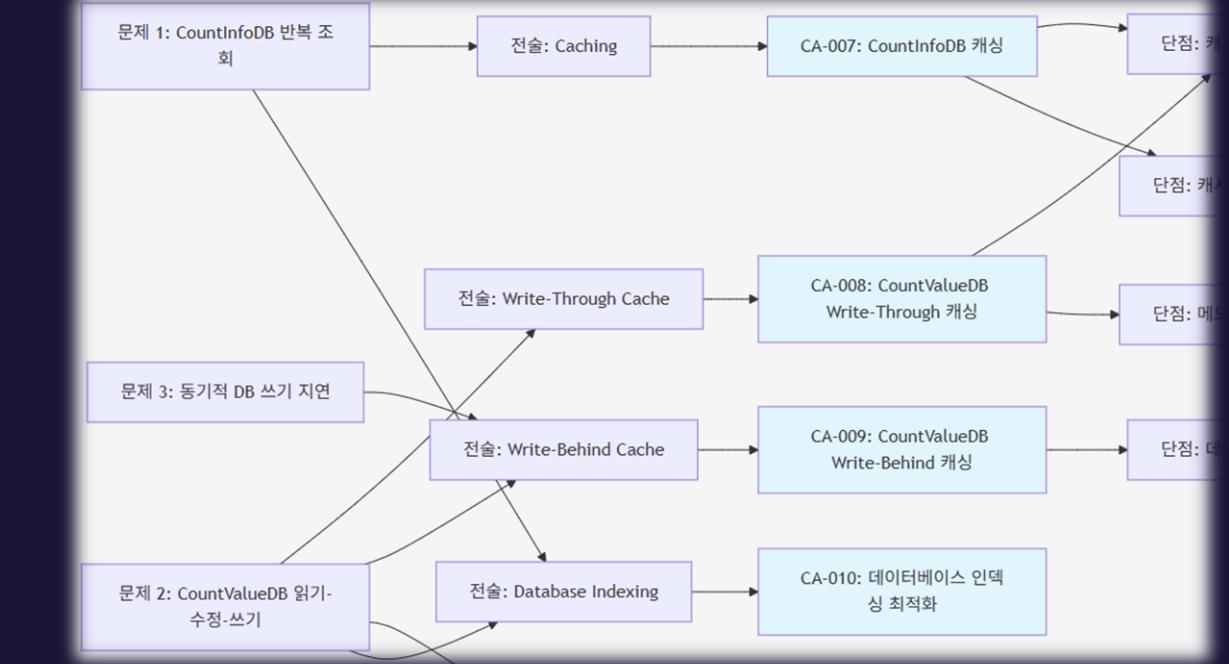


QS-nnn-{title}.md

candidates.md

체크 포인트

1. 후보 구조 설계의 완전성
2. 후보 구조 분석의 정확성



/agentk/design-performance 어떤 DB 인덱싱이 필요한지를 구체적으로 검토하고 후보 구조를 설계해줘.

에이전트: solution-architect

/agentk/select-solutions 후보 구조를 분석하고 채택 가능한 솔루션 후보 구조를 설계해줘.



후보 구조 설계

- 기술 솔루션 선택 문제 식별
- 기술 솔루션 후보 검토 및 비교
- 기술 솔루션 선택 후보 구조 제시
- 설계 과정 가시화 (마인드 맵)

후보 구조 목록 문서 작성

- 후보 구조 목록 작성 (ID, 제목, 분류 등)
- 상충/종속 관계 명시
- candidate/candidates.md 문서 간신



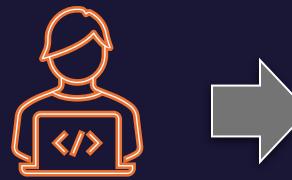
solutions.md



candidates.md

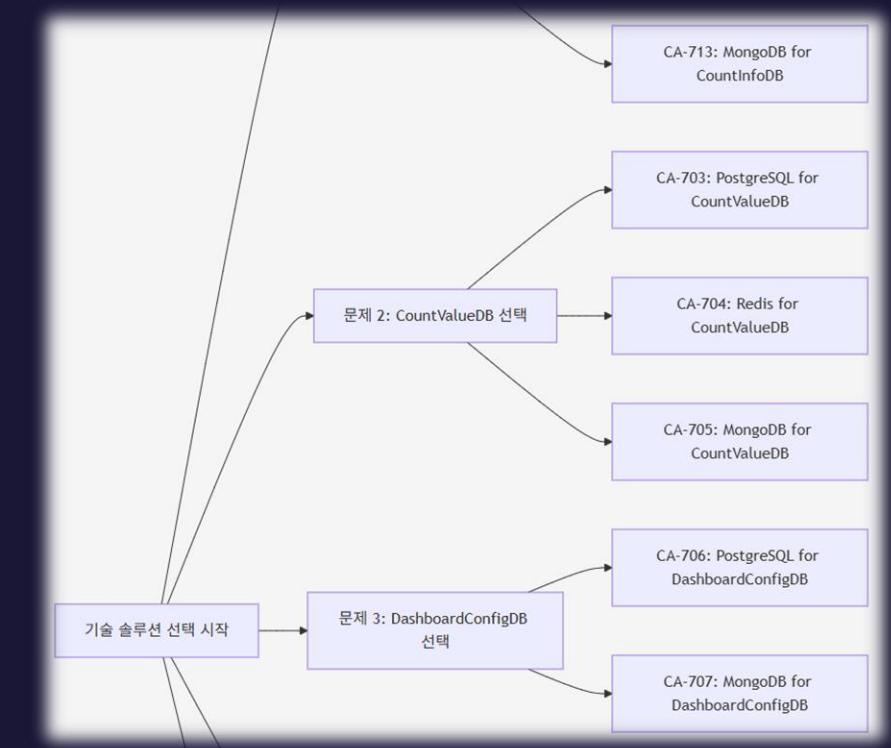
에이전트: solution-architect

/agentk/select-solutions 후보 구조를 분석하고 채택 가능한 솔루션 후보 구조를 설계해줘.



체크 포인트

1. 후보 구조 설계의 완전성
2. 후보 구조 분석의 정확성



/agentk/select-solutions 데이터 일관성을 유지하기 위한 CDC 솔루션에 대해서도 검토해줘.



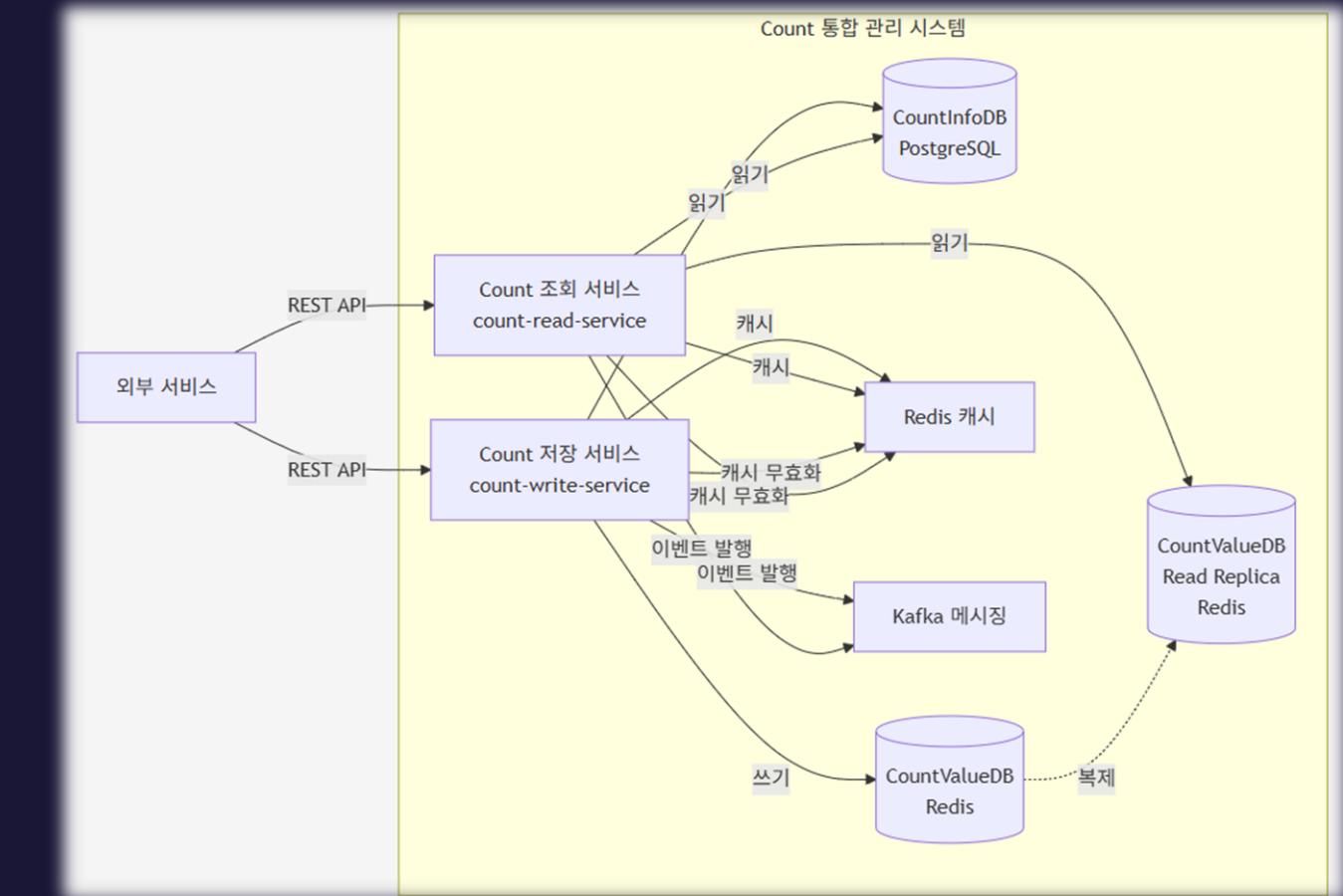
활동5. 최종 구조 설계 [동작]

- 동작 측면의 후보 구조 평가
- 동작 측면의 최종 구조 설계

Y Yong Jin, Cho (drajin.cho@bosornd.com)

활동5. 최종 구조 설계 [동작]

- 동작 측면의 후보 구조 평가
- 평가 기반 채택
- 최종 구조로의 통합 (배치 구조)



활동5. 최종 구조 설계 [동작]

agent	candidate-evaluator
command	evaluate-candidates
task	<ul style="list-style-type: none">- 후보 구조별 품질 시나리오 영향도 평가- 충돌하는 후보 구조 상호 비교 평가- 부정적 영향의 완화 전략 수립- 평가 결과를 근거로 채택/기각 결정



evaluations.md

decisions.md

agent	system-architect
command	integrate-deployment
task	<ul style="list-style-type: none">- 채택된 후보 구조 분석 및 통합- 배치 구조 설계- 주요 Use Case 동작 확인- 동작 측면의 품질 요구사항 확인



deployment.md

에이전트: candidate-evaluator

/agentk/evaluate-candidates 동작 측면의 후보 구조를 평가하고 채택 여부를 결정해줘.



후보 구조 평가

- 후보 구조별 품질 시나리오 영향도 평가
- 충돌하는 후보 구조 상호 비교 평가
- 부정적 영향의 완화 전략 수립
- 후보 구조 평가 결과 문서 작성

후보 구조 채택

- 평가 결과를 근거로 채택/기각 결정
- 채택된 후보 구조 목록 문서 작성



evaluations.md



decisions.md

에이전트: candidate-evaluator

/agentk/evaluate-candidates 동작 측면의 후보 구조를 평가하고 채택 여부를 결정해줘.



evaluations.md

decisions.md

체크 포인트

1. 후보 구조 평가의 완전성
2. 후보 구조 채택/기각의 적절성

CA-001 (Count 저장 서비스 분할) vs CA-006 (Count 저장/조회 서비스 통합)

품질 시나리오	CA-001	CA-006	비교
QS-001 (Count 저장 응답 시간)	(+)	(+)	동등 (네트워크 오버헤드 vs 단순화)
QS-002 (Count 조회 응답 시간)	(0)	(+)	CA-006 우세 (통합으로 인한 단순화)
QS-012 (Count 저장 서비스 독립성)	(++)	(--)	CA-001 매우 우세 (서비스 분할로 확장성)
QS-013 (Count 조회 서비스 독립성)	(0)	(--)	CA-001 우세 (CA-002와 함께 채택)
QS-006 (동시 요청 처리 능력)	(+)	(0)	CA-001 우세 (독립적 확장 가능)

트레이드오프 분석:

- CA-001: 서비스 독립성(++)과 확장성(+)을 크게 향상시키지만, 네트워크 오버헤드로 인한 비용이 발생합니다.
- CA-006: 단순성과 개발 생산성을 향상시키지만, 서비스 독립성(--)을 크게 저하시킵니다.

/agentk/evaluate-candidates CountInfoDB는 스키마 유연성이 중요해. MongoDB를 사용하도록 수정해줘.

에이전트: system-architect

/agentk/integrate-deployment 채택된 후보 구조를 근거로 동작 측면의 최종 구조를 설계해줘.



최종 구조 설계

- 채택된 후보 구조 분석
- 배치 구조 설계
- 주요 Use Case 동작 확인
- 동작 측면의 품질 요구사항 확인

최종 구조 문서 작성

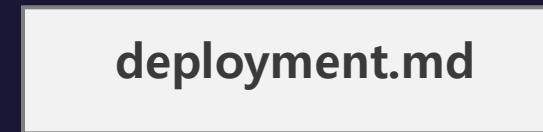
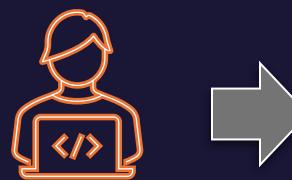
- 최종 배치 구조 개요 (목적, 설계 근거 등) 작성
- 전체 배치 구조도 작성
- 배치 목록 및 내부 구성 작성
- 주요 동작, 품질 요구사항 확인 결과 작성



deployment.md

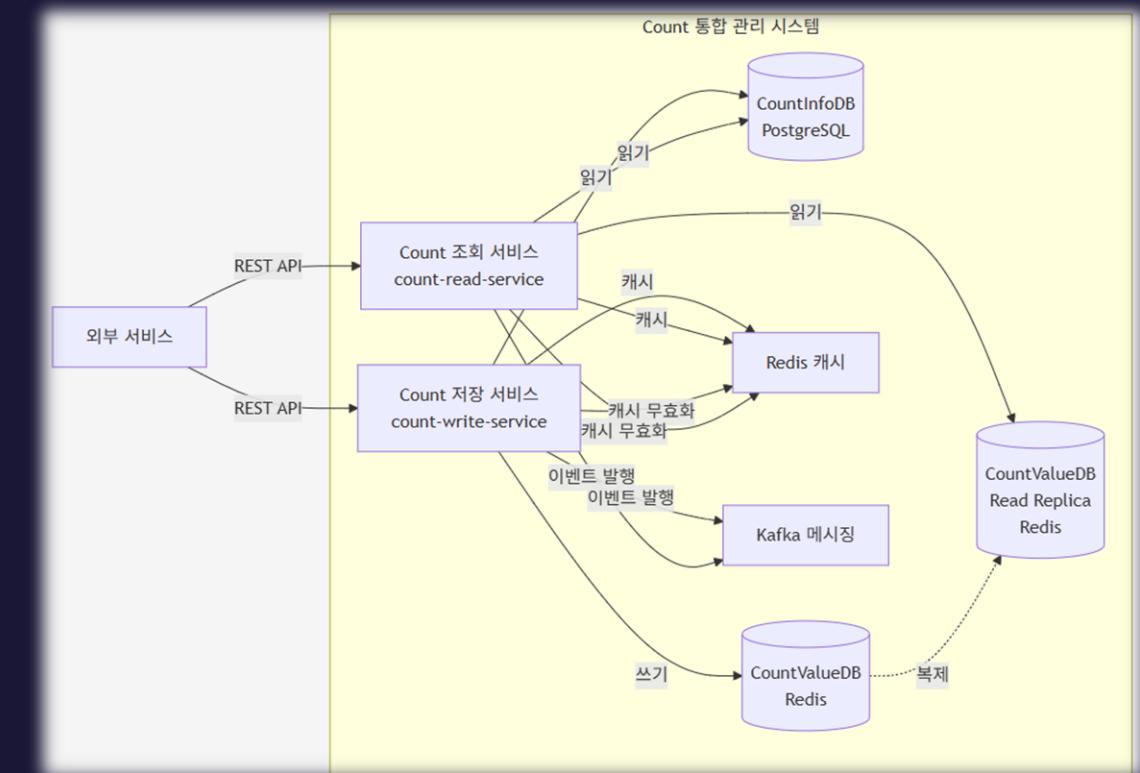
에이전트: system-architect

/agentk/integrate-deployment 채택된 후보 구조를 근거로 동작 측면의 최종 구조를 설계해줘.



체크 포인트

1. 채택된 후보 구조와의 일관성
2. 최종 구조 설계의 완전성



/agentk/integrate-deployment 채택된 후보 구조 CA-021가 반영되지 않은 듯.



활동6. 후보 구조 설계 [개발]

- 개발 측면의 후보 구조 설계

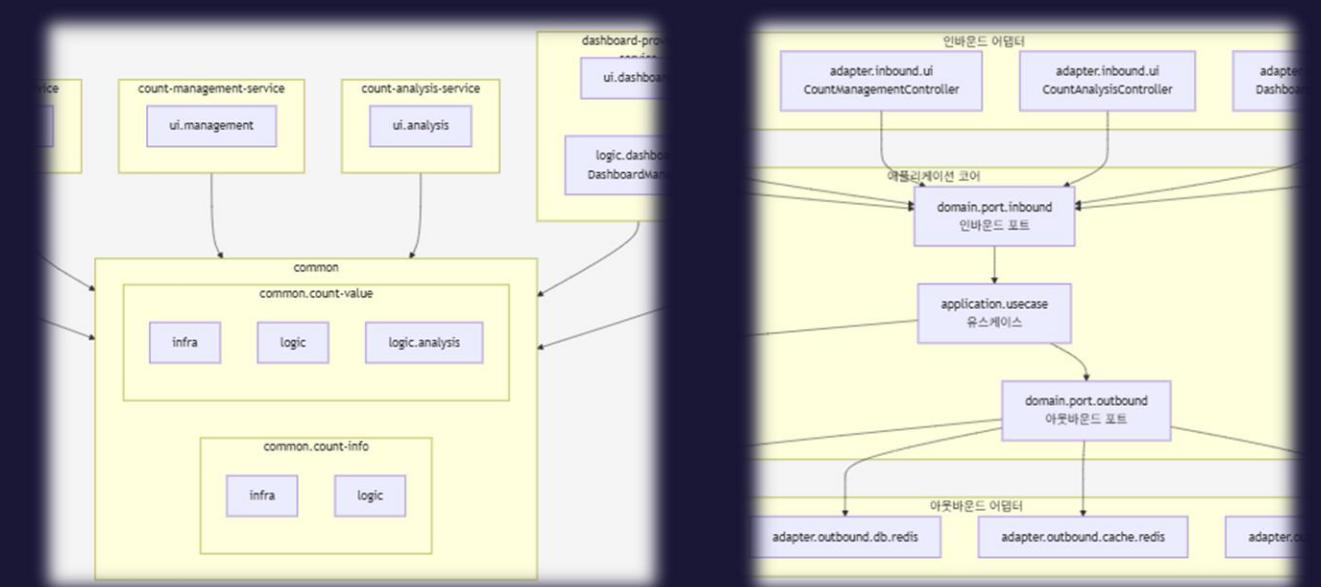
Y Yong Jin, Cho (drajin.cho@bosornd.com)

활동6. 후보 구조 설계 [개발]

▪ 개발 측면의 후보 구조 설계

- 패키지 관점: 패키지 / 레이어
- 품질 관점: 변경 / 유지보수 용이성 등
- 개발 프레임워크 관점: 앱 프레임워크 등

⋮



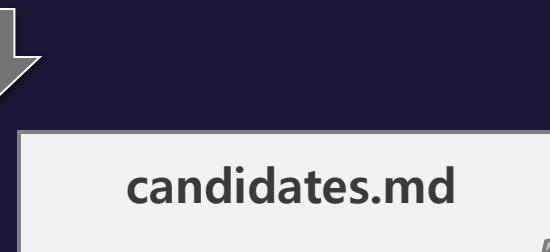
활동6. 후보 구조 설계 [개발]

agent	package-architect
command	design-packages
task	<ul style="list-style-type: none">- 모듈의 관심사 분석- 패키지 구성 후보 구조 설계- 패키지 간 의존성 후보 구조 설계- 후보 구조의 장단점 분석

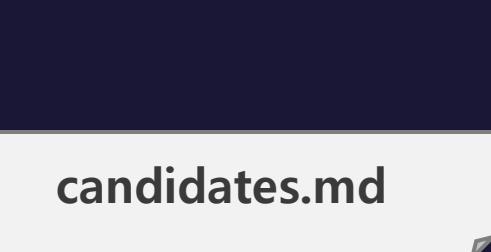
agent	framework-architect
command	select-frameworks
task	<ul style="list-style-type: none">- 개발 프레임워크 선택 문제 식별- 개발 프레임워크 후보 검토 및 비교- 개발 프레임워크 선택 후보 구조 제시- 후보 구조의 장단점 분석



packages.md



frameworks.md



활동6. 후보 구조 설계 [개발]

agent	modifiability-architect
command	design-modifiability
task	<ul style="list-style-type: none">- 변경 용이성 시나리오 분석 및 문제 식별- 변경 용이성 패턴 및 전술 적용 검토- 변경 용이성 향상 후보 구조 제시- 후보 구조의 장단점 분석 <p>...</p>



QS-nnn-{title}.md

candidates.md

에이전트: package-architect



/agentk/design-packages 패키지/레이어 구성에 관한 후보 구조를 설계해줘.

후보 구조 설계

- 개발 모듈 식별 및 관심사 분석
- 설계 패턴 및 전술 적용 검토
- 후보 구조 설계
- 설계 과정 가시화 (마인드 맵)

후보 구조 목록 문서 작성

- 후보 구조 목록 작성 (ID, 제목, 분류 등)
- 상충/종속 관계 명시
- candidate/candidates.md 문서 간신



packages.md



candidates.md

에이전트: package-architect

/agentk/design-packages 패키지/레이어 구성에 관한 후보 구조를 설계해줘.

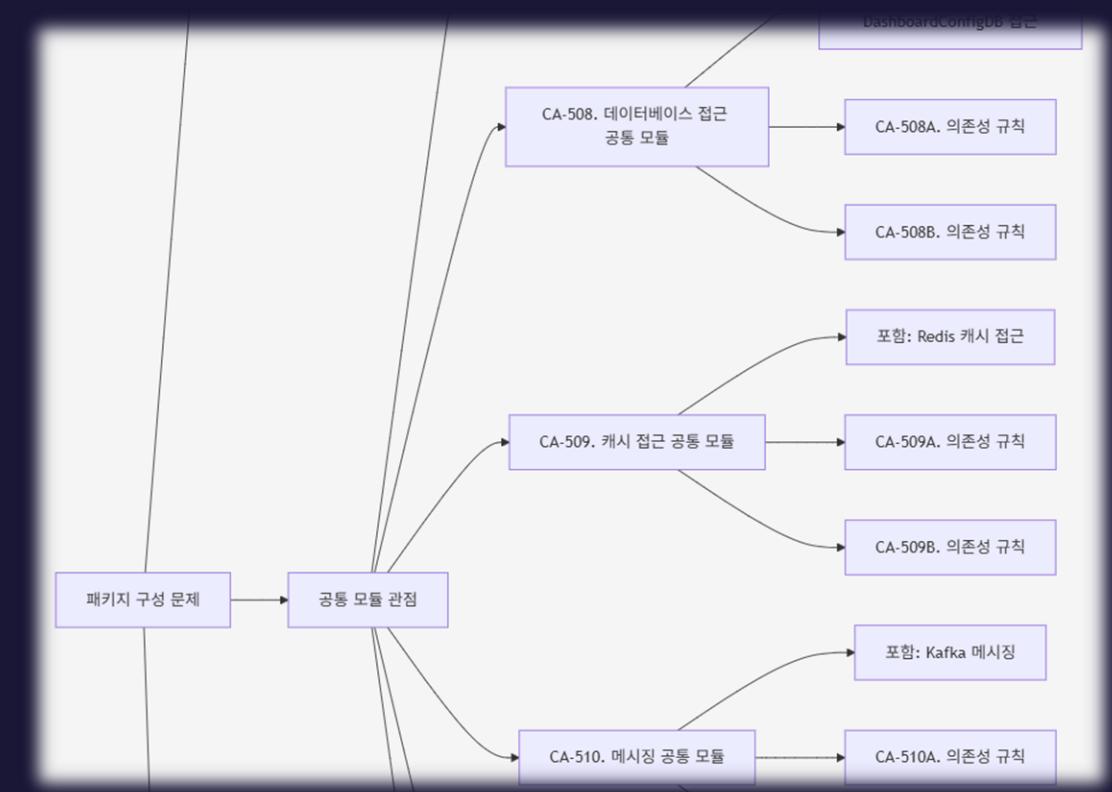


packages.md

candidates.md

체크 포인트

1. 후보 구조 설계의 완전성
2. 후보 구조 분석의 정확성



/agentk/design-packages DB와 Logic을 공통 패키지로 하고, 서비스별 패키지가 의존하는 구조를 설계해줘.

에이전트: modifiability-architect

/agentk/design-modifiability QS-008을 분석하고 후보 구조를 설계해줘.



후보 구조 설계

- 변경 용이성 시나리오 문제 식별
- 설계 패턴 및 전술 적용 검토
- 후보 구조 설계
- 설계 과정 가시화 (마인드 맵)

후보 구조 목록 문서 작성

- 후보 구조 목록 작성 (ID, 제목, 분류 등)
- 상충/종속 관계 명시
- candidate/candidates.md 문서 간신



QS-nnn-{title}.md



candidates.md

에이전트: modifiability-architect

/agentk/design-modifiability QS-008을 분석하고 후보 구조를 설계해줘.

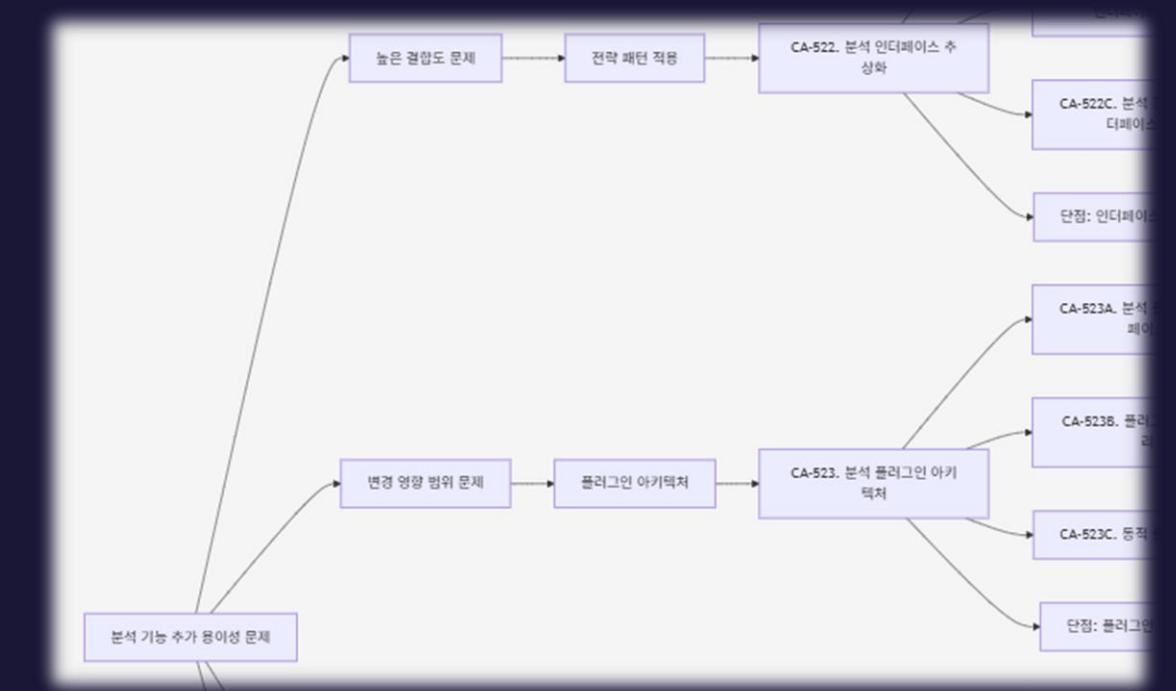


QS-nnn-{title}.md

candidates.md

체크 포인트

1. 후보 구조 설계의 완전성
2. 후보 구조 분석의 정확성



/agentk/design-modifiability 분석 모듈의 추상화 / 구현 패키지를 명확히 구분하는 후보 구조를 추가해줘.

에이전트: framework-architect

/agentk/select-frameworks 후보 구조를 분석하고 개발을 위한 프레임워크 후보 구조를 설계해줘.



후보 구조 설계

- 기술 솔루션 선택 문제 식별
- 기술 솔루션 후보 검토 및 비교
- 기술 솔루션 선택 후보 구조 제시
- 설계 과정 가시화 (마인드 맵)

후보 구조 목록 문서 작성

- 후보 구조 목록 작성 (ID, 제목, 분류 등)
- 상충/종속 관계 명시
- candidate/candidates.md 문서 간신



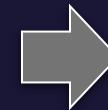
frameworks.md



candidates.md

에이전트: framework-architect

/agentk/select-frameworks 후보 구조를 분석하고 개발을 위한 프레임워크 후보 구조를 설계해줘.

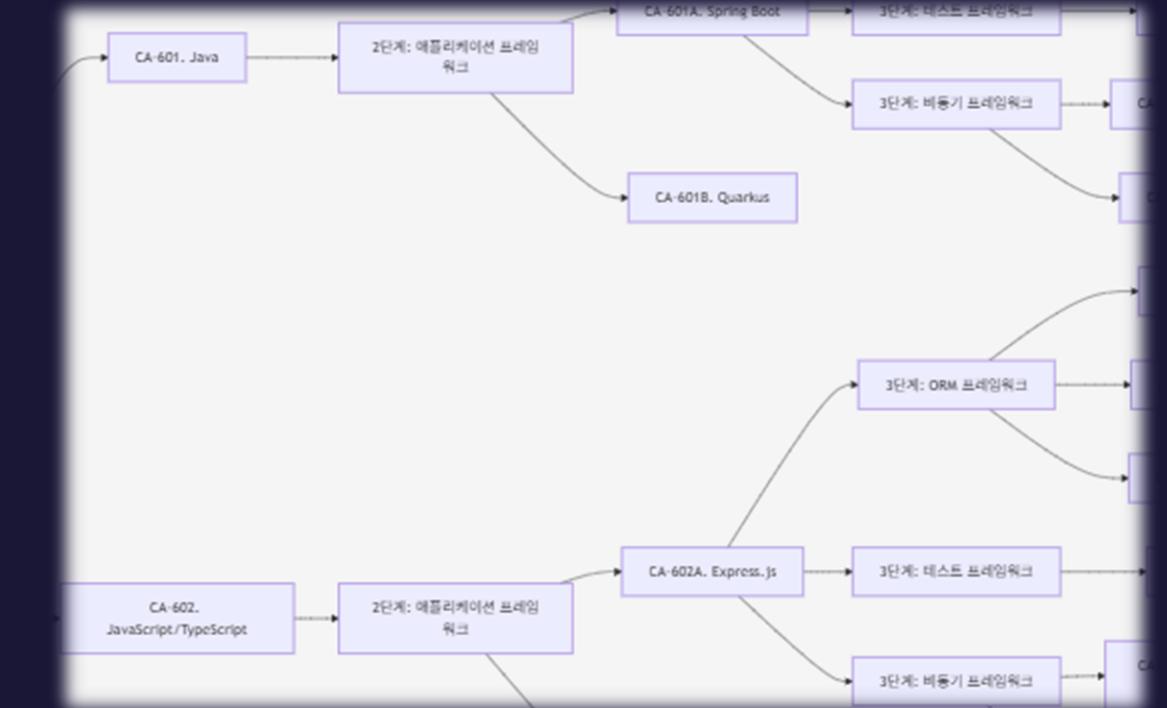


frameworks.md

candidates.md

체크 포인트

1. 후보 구조 설계의 완전성
2. 후보 구조 분석의 정확성



/agentk/select-frameworks Express.js 대신에 Nest.js를 사용하는 것은 어떨까? 후보 구조에 반영해줘.



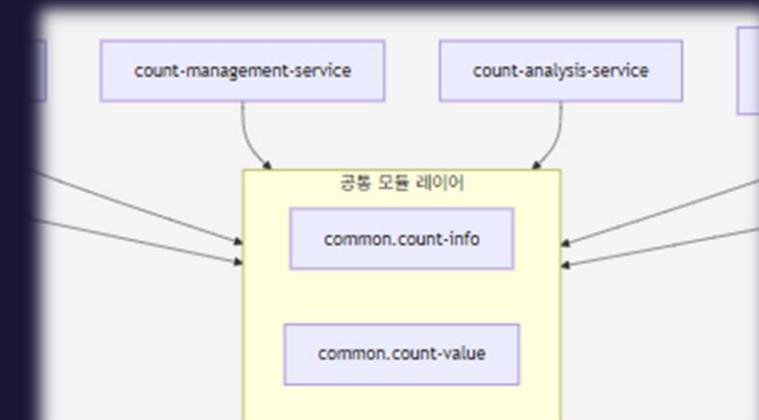
활동7. 최종 구조 설계 [개발]

- 개발 측면의 후보 구조 평가
- 개발 측면의 최종 구조 설계

Y Yong Jin, Cho (drajin.cho@bosornd.com)

활동7. 최종 구조 설계 [개발]

- 개발 측면의 후보 구조 평가
- 평가 기반 채택
- 최종 구조로의 통합 (모듈 구조)



```
count-management-system/
  └── common/
    ├── count-info/
    │   ├── infra/
    │   └── logic/
    └── count-value/
        ├── infra/
        └── logic/
            └── analysis/
  └── count-write-service/
      └── api/
  └── count-read-service/
      └── api/
```

CA-519A, CA-519B: 도메인별 공통 모듈
CA-519A: CountInfo 도메인 공통 모듈
infra 레이어
logic 레이어

CA-519B: CountValue 도메인 공통 모듈
infra 레이어
logic 레이어
analysis 로직

CA-519C: Count 저장 서비스
API 레이어

CA-519D: Count 조회 서비스
API 레이어

활동7. 최종 구조 설계 [개발]

agent	candidate-evaluator
command	evaluate-candidates
task	<ul style="list-style-type: none">- 후보 구조별 품질 시나리오 영향도 평가- 충돌하는 후보 구조 상호 비교 평가- 부정적 영향의 완화 전략 수립- 평가 결과를 근거로 채택/기각 결정



evaluations.md

decisions.md

agent	module-architect
command	integrate-module
task	<ul style="list-style-type: none">- 채택된 후보 구조 분석 및 통합- 모듈 구조 설계- 프로젝트 폴더 구조 설계- 개발 측면의 품질 요구사항 확인



module.md

에이전트: candidate-evaluator

/agentk/evaluate-candidates 개발 측면의 후보 구조를 평가하고 채택 여부를 결정해줘.



후보 구조 평가

- 후보 구조별 품질 시나리오 영향도 평가
- 충돌하는 후보 구조 상호 비교 평가
- 부정적 영향의 완화 전략 수립
- 후보 구조 평가 결과 문서 작성

후보 구조 채택

- 평가 결과를 근거로 채택/기각 결정
- 채택된 후보 구조 목록 문서 작성



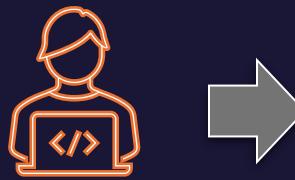
evaluations.md



decisions.md

에이전트: candidate-evaluator

/agentk/evaluate-candidates 개발 측면의 후보 구조를 평가하고 채택 여부를 결정해줘.



evaluations.md

decisions.md

체크 포인트

1. 후보 구조 평가의 완전성
2. 후보 구조 채택/기각의 적절성

CA-602A (Express.js) vs CA-602B (NestJS)

품질 시나리오	CA-602A	CA-602B	비교
QS-007 (API 인터페이스 변경 용이성)	(+)	(++)	CA-602B 우세 (모듈 시스템, 의존성 주입)
QS-008 (UI 변경 용이성)	(+)	(++)	CA-602B 우세 (모듈 시스템, 컴포넌트 분리)
QS-009 (분석 기능 추가 용이성)	(+)	(++)	CA-602B 우세 (모듈 시스템, Provider 패턴)
QS-010 (대시보드 기능 추가 용이성)	(+)	(++)	CA-602B 우세 (모듈 시스템, Provider 패턴)
QS-011 (데이터베이스 변경 용이성)	(+)	(++)	CA-602B 우세 (의존성 주입, 리포지토리 패턴 지원)
개발 생산성	(++)	(+)	CA-602A 우세 (경량 프레임워크, 빠른 시작)
구조화	(-)	(++)	CA-602B 매우 우세 (Spring Boot와 유사한 구조)
학습 곡선	(++)	(+)	CA-602A 우세 (간단한 구조)

트레이드오프 분석:

- CA-602A: 개발 생산성(++)과 학습 곡선(++)이 우수하지만, 구조화(-)와 변경 용이성 향상이 제한적
- CA-602B: 모든 변경 용이성(++)을 크게 향상시키며, 구조화(++)가 매우 우수. 사용자 선호사항(구조적 접근)과 일치

/agentk/evaluate-candidates 향후 확장을 위해 NestJS를 채택하도록 수정해줘.

에이전트: module-architect

/agentk/integrate-module 채택된 후보 구조를 근거로 개발 측면의 최종 구조를 설계해줘.



최종 구조 설계

- 채택된 후보 구조 분석
- 모듈 구조 설계 (레이어 / 패키지)
- 프로젝트 폴더 구조 설계
- 개발 측면의 품질 요구사항 확인

최종 구조 문서 작성

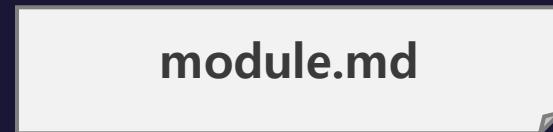
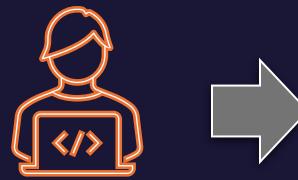
- 최종 모듈 구조 개요 (목적, 설계 근거 등) 작성
- 전체 모듈 구조도 / 폴더 구조 작성
- 레이어 / 패키지 목록 및 내부 구성 작성
- 개발 측면 품질 요구사항 확인 결과 작성



module.md

에이전트: module-architect

/agentk/integrate-module 채택된 후보 구조를 근거로 개발 측면의 최종 구조를 설계해줘.



체크 포인트

1. 채택된 후보 구조와의 일관성
2. 최종 구조 설계의 완전성

```
count-management-system/
  -- common/
    -- count-info/
      -- infra/
        -- repository/
          -- entity/
        -- logic/
          -- service/
    -- count-value/
      -- infra/
        -- repository/
          -- entity/
        -- logic/
          -- service/
            -- analysis/
              -- strategy/
              -- analyzer/
  -- count-write-service/
    -- api/
      -- write/
        -- controller/
        -- adapter/
        -- dto/
        -- version/
  -- count-read-service/
    -- api/
```

공통 모듈 레이어
CA-519A: CountInfo 도메인 공통 모듈
infra 레이어
Repository 인터페이스 및 구현체
Entity 모델 (TypeORM)
logic 레이어
Service 인터페이스 및 구현체

CA-519B: CountValue 도메인 공통 모듈
infra 레이어
Repository 인터페이스 및 구현체
Entity 모델 (Redis)
logic 레이어
Service 인터페이스 및 구현체
CA-522: 분석 로직 레이어
AnalysisStrategy 인터페이스
Analyzer 구현체

CA-519C: Count 저장 서비스
API 레이어
api.write 패키지
REST API Controller
CA-537: API 어댑터
DTO (Data Transfer Object)
CA-541: API 버전 관리

CA-519D: Count 조회 서비스
API 레이어

/agentk/integrate-module Nest.js 사용을 명확히 명세해줘.



활동8. 구조 명세

- 구조 명세서 작성

Y Yong Jin, Cho (drajin.cho@bosornd.com)

활동8. 구조 명세

- 구조 명세서 작성
- 추가 설계 활동 (X), 설계 산출물 통합
- 설계 결과를 중심으로 설명

[목차]

- | | |
|---------------|-----------|
| 1. 시스템 개요 | 3. 시스템 구조 |
| 2. 요구사항 | 4. 모듈 사양 |
| 2.1 기능적 요구사항 | |
| 2.2 비기능적 요구사항 | |
| 2.3 품질 속성 | |

Appendix

- | | |
|---------------|-------------|
| A. 도메인 모델 | D. 후보 구조 |
| B. 품질 시나리오 | E. 후보 구조 평가 |
| C. 품질 시나리오 분석 | F. 최종 구조 설계 |

활동8. 구조 명세

agent	architecture-specifier
command	specify-architecture
task	<ul style="list-style-type: none">- 시스템 개요 작성- 요구사항 요약- 최종 구조 명세 (배치 구조, 모듈 구조)- 설계 결과 설명



architecture.md

agent	architecture-specifier
command	specify-architecture
task	<ul style="list-style-type: none">- 부록 작성- 설계 중간 산출물 요약- 설계 과정 설명



architecture.md

에이전트: architecture-specifier

/agentk/specify-architecture 설계된 구조의 통합 명세서를 작성해줘.



구조 명세

- 시스템 개요
- 요구사항 (기능, NFR, QA, 제약 사항)
- 동작 측면의 시스템 구조 (배치)
- 개발 측면의 시스템 구조 (모듈)

구조 명세 (부록)

- 도메인 모델
- 품질 시나리오 생성 및 평가
- 후보 구조 설계 및 평가
- 최종 구조 통합



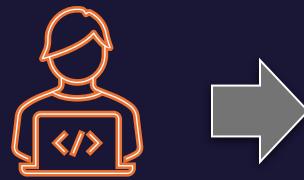
architecture.md



architecture.md

에이전트: architecture-specifier

/agentk/specify-architecture 설계된 구조의 통합 명세서를 작성해줘.



architecture.md

체크 포인트

1. 구조 설계서의 완전성
2. 구조 설계서의 이해 용이성

[목차]

- | | |
|-----------|-----------|
| 1. 시스템 개요 | 3. 시스템 구조 |
| 2. 요구사항 | 4. 모듈 사양 |
- 2.1 기능적 요구사항
2.2 비기능적 요구사항
2.3 품질 속성

Appendix

- | | |
|---------------|-------------|
| A. 도메인 모델 | D. 후보 구조 |
| B. 품질 시나리오 | E. 후보 구조 평가 |
| C. 품질 시나리오 분석 | F. 최종 구조 설계 |

/agentk/specify-architecture 각각 서비스가 제공하는 API를 구체적으로 명세해줘.



활동9. 구조 평가

- 구조 설계의 충실도 및 완전성 평가
- 품질 요구사항 만족도 평가

Y Yong Jin, Cho (drajin.cho@bosornd.com)

활동9. 구조 평가

- 구조적 의사결정 식별
- 후보 구조와의 관계 분석
- 품질 요구사항 만족도 평가

- NFR 충족 여부: 충족 (4/4)
 - 충족: NFR-001, NFR-002, NFR-003, NFR-004
 - 모든 NFR의 허용치를 만족하는 구조적 의사결정이 채택됨
- QA 만족도:
 - 우선순위 1-4: 충분
 - QA-001 (우선순위 1): 충분 - 여러 최적화 전술로 응답 시간 최소화
 - QA-002 (우선순위 2): 충분 - Read Replica 및 캐싱으로 응답 시간 최소화
 - QA-003 (우선순위 3): 충분 - 서비스 분할 및 Database per Service로 독립성 최대화
 - QA-004 (우선순위 4): 충분 - 서비스 분할 및 Database per Service로 독립성 최대화
 - 우선순위 5-7: 충분
 - QA-005 (우선순위 5): 충분 - 도메인별 공통 모듈 및 인터페이스 추상화로 분석 기능 추가 용이성 최대화
 - QA-006 (우선순위 6): 충분 - 서비스 분할로 대시보드 기능 추가 용이성 최대화
 - QA-007 (우선순위 7): 충분 - UI와 API 계층 분리로 UI 변경 용이성 최대화
 - 우선순위 8-12: 충분
 - 서비스 독립성 관련 QA는 서비스 분할로 충분히 만족
 - 대시보드 갱신 시간 최소화는 이벤트 기반 통신 및 SSE로 충분히 만족

활동9. 구조 평가

agent	architecture-analyzer
command	analyze-architecture
task	<ul style="list-style-type: none">- architecture.md 분석- 명세된 구조적 의사결정 식별- 구조적 의사결정 목록 작성- evaluation/decisions.md 문서 생성



decisions.md

agent	architecture-evaluator
command	evaluate-architecture
task	<ul style="list-style-type: none">- 채택된 후보 구조와의 비교 분석- 구조 설계 활동의 충실도 및 완전성 평가- 품질 요구사항 만족도 평가- evaluation/evaluation.md 문서 생성



evaluation.md

에이전트: architecture-analyzer

/agentk/analyze-architecture 명세된 구조를 분석하고 구조적 의사결정을 식별해줘.



구조적 의사결정 식별

- architecture.md 파일 읽기 및 분석
- 동작(배치) 측면의 구조적 의사결정 식별
- 개발(모듈) 측면의 구조적 의사결정 식별
- 품질에 미치는 영향 분석

구조적 의사결정 목록 작성

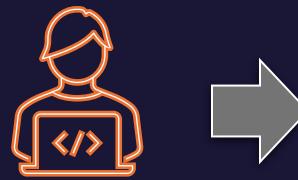
- 구조적 의사결정 개요 (목적, 분석 범위 등)
- 구조적 의사결정 목록 (ID, 제목, 내용) 작성
- 대안, 품질에 미치는 영향 작성
- evaluation/decisions.md 문서 생성



decisions.md

에이전트: architecture-analyzer

/agentk/analyze-architecture 명세된 구조를 분석하고 구조적 의사결정을 식별해줘.



decisions.md

체크 포인트

1. 구조적 의사결정 식별의 완전성

AD-001: Count 저장/조회 서비스 분할

- 의사결정 내용: Count 저장 기능과 조회 기능을 별도의 마이크로서비스(count-write-service, count-read-service)로 분할하여 독립적으로 배포 및 스케일링 가능하도록 설계
- 관련 구조: 3장 3.1 컴포넌트 명세의 Count 저장 서비스 및 Count 조회 서비스
- 대안:
 - 대안 1: Count 저장/조회 서비스 통합 - 단일 서비스로 통합하여 단순한 구조 유지
- 영향받는 품질 요구사항:
 - QA-001 (Count 저장 응답 시간 최소화): (+) - 서비스 분할로 인한 네트워크 오버헤드가 있지만, 독립적 성능 최적화로 전체적으로 개선
 - QA-002 (Count 조회 응답 시간 최소화): (++) - 독립적 최적화 및 읽기 전용 복제본 활용으로 조회 성능 크게 향상
 - QA-003 (Count 저장 서비스 독립성 최대화): (++) - 서비스 분할로 독립적인 배포, 스케일링, 운영 가능하여 독립성 크게 향상
 - QA-004 (Count 조회 서비스 독립성 최대화): (++) - 서비스 분할로 독립적인 배포, 스케일링, 운영 가능하여 독립성 크게 향상
 - NFR-003 (동시 요청 처리 능력): (++) - 독립적 확장 가능하여 처리량 크게 향상

/agentk/analyze-architecture RedisDB를 CountValueDB의 캐시로 사용하는 것도 확인하고 추가해줘.

에이전트: architecture-evaluator

/agentk/evaluate-architecture 명세된 구조를 평가해줘.



구조 평가

- 채택된 후보 구조와의 비교 분석
- 구조 설계 활동의 충실도 및 완전성 평가
- 품질 요구사항 만족도 평가
- 위험 요소 식별 및 완화 전략 제시

구조 평가 문서 작성

- 구조 평가 개요 (목적, 평가 기준 등) 작성
- 구조 설계 활동의 충실도 및 완전성 평가 작성
- 품질 요구사항 만족도 평가 작성
- evaluation/evaluation.md 문서 작성



evaluation.md

에이전트: architecture-evaluator

/agentk/evaluate-architecture 명세된 구조를 평가해줘.



module.md

체크 포인트

1. 구조 평가의 완전성
2. 품질 요구사항 만족도 평가의 적절성
3. 평가 보고서의 명확성

- **NFR 충족 여부:** 충족 (4/4)
 - 충족: NFR-001, NFR-002, NFR-003, NFR-004
 - 모든 NFR의 허용치를 만족하는 구조적 의사결정이 채택됨
- **QA 만족도:**
 - 우선순위 1-4: 충분
 - QA-001 (우선순위 1): 충분 - 여러 최적화 전술로 응답 시간 최소화
 - QA-002 (우선순위 2): 충분 - Read Replica 및 캐싱으로 응답 시간 최소화
 - QA-003 (우선순위 3): 충분 - 서비스 분할 및 Database per Service로 독립성 최대화
 - QA-004 (우선순위 4): 충분 - 서비스 분할 및 Database per Service로 독립성 최대화
 - 우선순위 5-7: 충분
 - QA-005 (우선순위 5): 충분 - 도메인별 공통 모듈 및 인터페이스 추상화로 분석 기능 추가 용이성 최대화
 - QA-006 (우선순위 6): 충분 - 서비스 분할로 대시보드 기능 추가 용이성 최대화
 - QA-007 (우선순위 7): 충분 - UI와 API 계층 분리로 UI 변경 용이성 최대화
 - 우선순위 8-12: 충분
 - 서비스 독립성 관련 QA는 서비스 분할로 충분히 만족
 - 대시보드 갱신 시간 최소화는 이벤트 기반 통신 및 SSE로 충분히 만족

/agentk/evaluate-architecture 추가된 구조적 의사결정 AD-025를 포함해서 평가를 보완해줘.



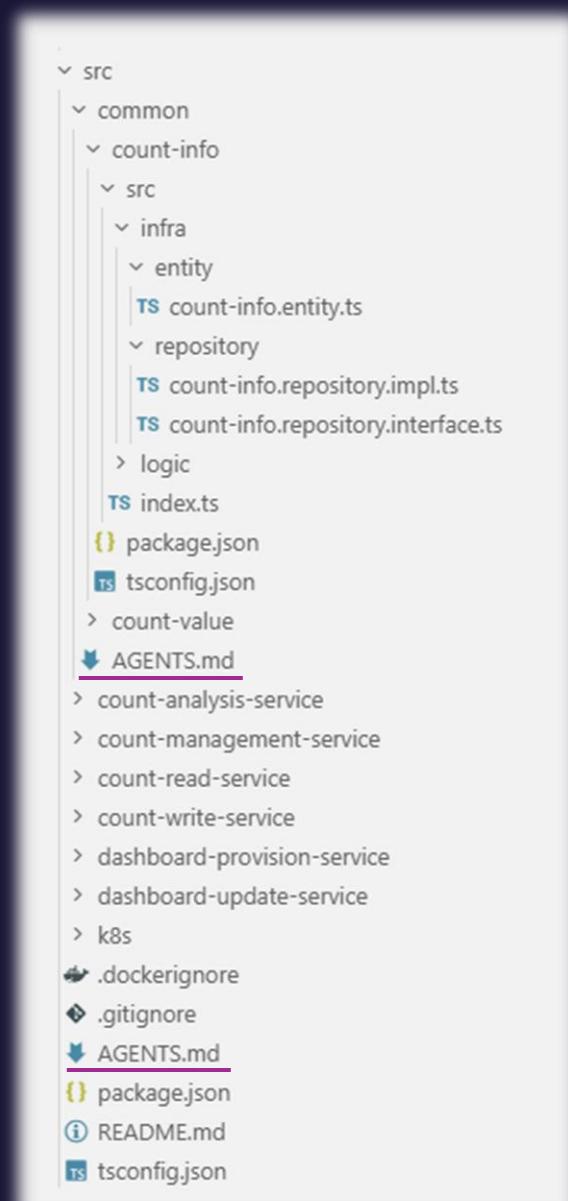
활동10. 코드 생성

- 코드 생성

Y Yong Jin, Cho (drajin.cho@bosornd.com)

활동10. 코드 생성

- 프로젝트 폴더 구조 생성
- AGENTS.md 파일 생성
- 코드 생성



```
import { CountInfoEntity } from '../entity/count-info.entity';

export interface ICountInfoRepository {
    findById(id: string): Promise<CountInfoEntity | null>;
    findAll(): Promise<CountInfoEntity[]>;
    save(entity: CountInfoEntity): Promise<CountInfoEntity>;
    update(id: string, entity: Partial<CountInfoEntity>): Promise<void>;
    delete(id: string): Promise<void>;
    exists(id: string): Promise<boolean>;
}
```

활동10. 코드 생성

agent	work-assigner
command	assign-worker
task	<ul style="list-style-type: none">- 설계 문서 분석- 소스 코드 디렉토리 구조 생성- 전체 프로젝트 레벨 AGENTS.md 생성- 서브 디렉토리별 AGENTS.md 생성



AGENTS.md

agent	code-generator
command	generate-code
task	<ul style="list-style-type: none">- 모듈별 클래스 및 인터페이스 생성- 의존성 관계에 따른 코드 구조 생성- 비즈니스 로직 구현- 빌드 및 배포 설정 파일 생성



source code

에이전트: work-assigner

/agentk/assign-worker 설계된 구조를 기반으로 소스 구조와 AGENTS.md 파일을 생성해줘.



전체 AGENTS.md 생성

- 설계 문서 분석
- 폴더 구조 생성
- 전체 프로젝트 구조 및 공통 규칙 식별
- AGENTS.md 명세

서브 AGENTS.md 생성

- 모듈별 개발 항목 분석
- 모듈별 내부 구조, 코드 생성 규칙 정의
- 상위 AGENTS.md 내용과 중복 제거
- AGENTS.md 명세



AGENTS.md

에이전트: work-assigner

/agentk/assign-worker 설계된 구조를 기반으로 소스 구조와 AGENTS.md 파일을 생성해줘.



AGENTS.md

체크 포인트

1. 폴더 구조 생성의 정확성
2. AGENTS.md 파일의 완전성
3. 역할 분리의 명확성

Count 통합 관리 시스템 에이전트 명세 (전체)

개요

이 문서는 `count/src` 디렉토리의 전체 시스템을 관리하는 에이전트의 역할과 책임을 정의합니다. 이 에이전트는 전체 프로젝트의 구조를 조정하고, 서비스 간 의존성을 관리하며, 통합 빌드 및 테스트를 담당합니다.

역할과 책임

주요 역할

- 전체 프로젝트 구조 관리 및 조정
- 워크스페이스(npm workspaces) 관리
- 서비스 간 의존성 관리
- 공통 모듈 버전 관리
- 빌드 및 테스트 통합 관리
- 배포 파이프라인 조정

/agentk/assign-worker count-info 모듈을 관리하는 AGENTS.md를 작성해줘.

에이전트: code-generator

/agentk/generate-code 설계된 구조를 기반으로 코드를 생성해줘.



코드 생성

- 레이어 구조에 따라 패키지 및 클래스 생성
- 의존성 관계에 따른 코드 구조 생성
- 비즈니스 로직 구현
- 테스트 코드 작성

설정 파일 생성

- 빌드 설정 파일 생성
- 애플리케이션 설정 파일 생성
- 배포 설정 생성



source code

에이전트: code-generator

/agentk/generate-code 설계된 구조를 기반으로 코드를 생성해줘.



source code

체크 포인트

1. 코드 생성의 완전성
2. 비즈니스 로직 구현의 정확성
3. 코드 품질

```
import { CountInfoEntity } from '../entity/count-info.entity';

export interface ICountInfoRepository {
  findById(id: string): Promise<CountInfoEntity | null>;
  findAll(): Promise<CountInfoEntity[]>;
  save(entity: CountInfoEntity): Promise<CountInfoEntity>;
  update(id: string, entity: Partial<CountInfoEntity>): Promise<void>;
  delete(id: string): Promise<void>;
  exists(id: string): Promise<boolean>;
}
```

/agentk/generate-code 전체 서비스의 docker image를 생성하는 스크립트를 추가해줘.

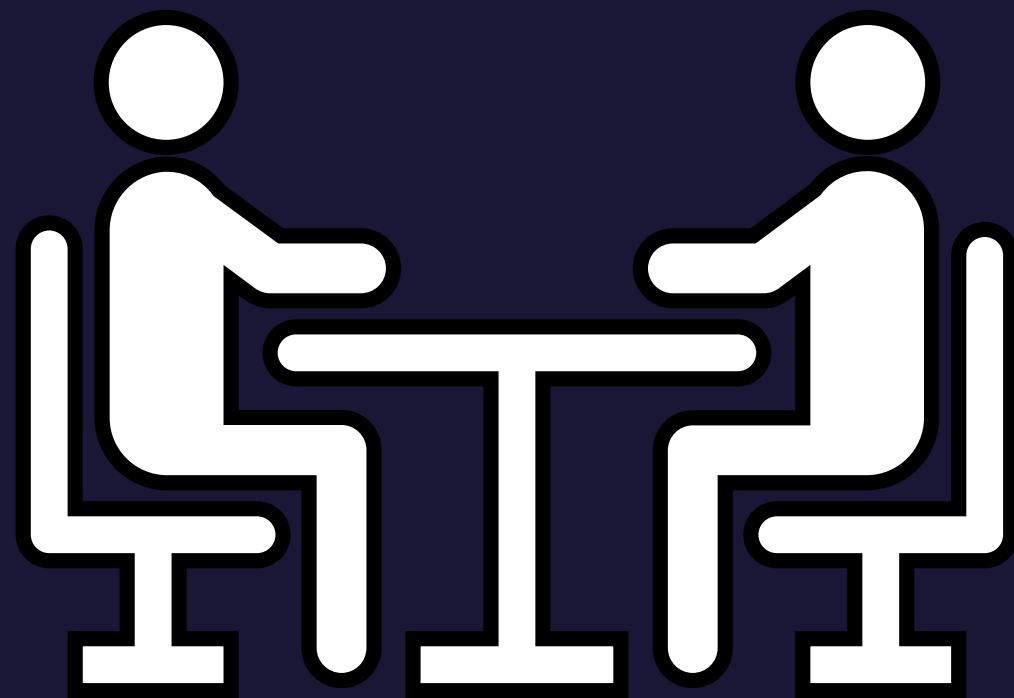


토론

Y Yong Jin, Cho (drajin.cho@bosornd.com)

토론

- AA 양성과정내 활용 방안
- AI 시대, 구조 설계 활동의 방향





요약

Y Yong Jin, Cho (drajin.cho@bosornd.com)

구조 설계, AI 시대 소프트웨어 개발의 필수 요소

▪ AI와의 효과적인 협업을 위한 필수 요소

- LLM은 기억하지 못하므로, 명시적이고 체계적인 구조 명세가 필요합니다.
- 구조를 매개로 하면 AI가 일관성 있게 작업할 수 있습니다.
- 구조 설계를 통해 고품질 소프트웨어를 만드는 것이 비즈니스 경쟁력의 핵심입니다.

