AI Consulting Assistant Platform
기술 아키텍처 및 워크플로우 설계서

문서 버전: 1.2
작성일: 2025년 12월 2일
문서 유형: 기술 아키텍처 분석 보고서
작성: Brian Lee / R&D Center

목차

1. 플랫폼 개요

1.1 플랫폼 목적

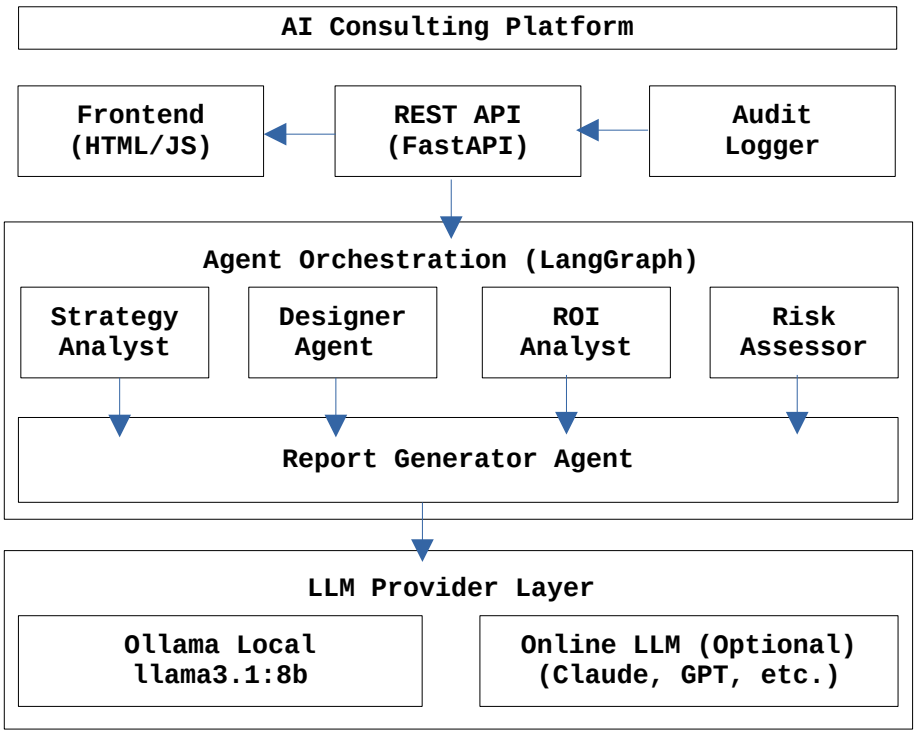
AI Consulting Assistant Platform은 기업의 AI 전환(AX: AI Transformation)을 지원하는 멀티 에이전트 기반 컨설팅 플랫폼이다. 인간 컨설턴트와 멀티 AI 에이전트가 역할 기반 협업을 수행하여 체계적인 AI 인프라 도입을 위한 5단계 접근법에 근거한 AI 컨설팅 업무의 전체적인 프레임워크를 자동화한 플랫폼이다.

이 플랫폼은 인공지능 인프라 도입 및 구축에 필요한 전문 AI 컨설턴트의 역활을 수행할 수 있도록 설계되었으며, AI 컨설팅 서비스 수요는 급증하고 있으나, 전문 인력이 매우 부족한 가운데, 플랫폼 도구를 활용하여 보다 정교한 컨설팅 결과물을 얻을 수 있도록 설계하였다.

1.2 핵심 기술 스택

계층	기술 스택	역할
Frontend	HTML5, CSS3, JavaScript	사용자 인터페이스
Backend	FastAPI (Python 3.12)	REST API 서버
LLM Engine	Ollama (llama3.1:8b)	로컬 LLM 추론
Embedding	nomic-embed-text	벡터 임베딩
Agent Framework	LangChain, LangGraph	에이전트 오케스트레이션
Multi-Agent	CrewAI, AutoGen	역할 기반 협업
Database	SQLite + aiosqlite	비동기 데이터 저장

1.3 아키텍처 다이어그램



2. 하이브리드 LLM 아키텍처

2.1 설계 철학

플랫폼은 보안 중심의 하이브리드 LLM 아키텍처를 채택한다. 이는 AI 컨설팅 의뢰 기업의 내부 보안을 강화하기 위한 방법론으로, 민감한 기업 데이터가 외부로 유출되지 않도록 설계되었다.

2.2 Ollama 기반 Local LLM

파일 위치: `src/core/llm_provider.py`

```
python
class LLMProvider:
    Ollama 기반 LLM 제공자

    def __init__(
        self,
        model: str = None,
        base_url: str = None,
        temperature: float = 0.7,
        top_p: float = 0.9,
    ):
        self.model = model or settings.OLLAMA_MODEL   llama3.1:8b
        self.base_url = base_url or settings.OLLAMA_BASE_URL   localhost:11434
```

핵심 구성 요소

구성 요소	설정 값	설명
Base URL	`http://localhost:11434`	Ollama 서버 엔드포인트
Model	`llama3.1:8b`	기본 추론 모델
Embedding Model	`nomic-embed-text`	벡터 임베딩 모델
Temperature	`0.7`	창의성/일관성 균형
Top-p	`0.9`	누적 확률 샘플링

2.3 하이브리드 전환 설계

플랫폼은 필요에 따라 온라인 LLM으로 전환 가능하도록 설계되어 있다:

```
python
class ConsultingLLMProvider(LLMProvider):
    AI 컨설팅 전용 LLM 제공자 (프롬프트 최적화)

    CONSULTING_SYSTEM_PROMPT = 당신은 기업 AI 전환(AX) 전문 컨설턴트입니다.
    [역할]
    - AI 인프라 도입 및 구축 전문 컨설턴트
    - AI 성숙도 진단 및 전략 수립 전문가
    - MLOps 및 AI 거버넌스 전문가
    - 산업별 AI 적용 사례 전문가
```

2.4 보안 이점

보안 요소	Local LLM	Online LLM
데이터 유출 위험	없음	존재
네트워크 의존성	없음	있음
응답 지연	낮음	네트워크 상황 의존
비용	초기 설치비용	사용량 기반
모델 커스터마이징	가능	제한적

3. 멀티 에이전트 프레임워크

3.1 프레임워크 통합 현황

플랫폼은 다음 4개의 멀티 에이전트 프레임워크를 통합 활용한다:

3.1.1 LangChain

역할: LLM 연동 및 체인 구성의 기반 레이어

```
python
from langchain_community.llms import Ollama
from langchain_community.embeddings import OllamaEmbeddings
from langchain_core.language_models.base import BaseLanguageModel
from langchain_core.messages import HumanMessage, AIMessage, SystemMessage
```

핵심 기능:
- Ollama LLM 인스턴스 관리

- 프롬프트 체이닝
- 메시지 히스토리 관리
- 임베딩 생성

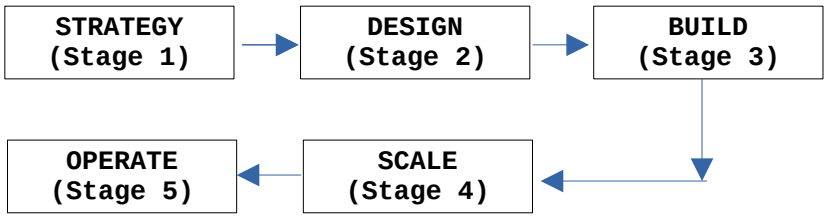
3.1.2 LangGraph

역할: 에이전트 워크플로우 상태 관리 및 오케스트레이션

파일 위치: `src/agents/agent_orchestrator.py`

```
python
class WorkflowState(TypedDict):
    워크플로우 상태
     project_id: str
     current_stage: ConsultingStage
     company_profile: Dict[str, Any]
     maturity_assessment: Optional[Dict[str, Any]]
     use_cases: List[Dict[str, Any]]
     scenarios: List[Dict[str, Any]]
     selected_scenario: Optional[str]
     roi_analysis: Optional[Dict[str, Any]]
     risk_assessment: Optional[Dict[str, Any]]
     reports: List[Dict[str, Any]]
     human_feedback: List[Dict[str, Any]]
     pending_approval: bool
     messages: List[Dict[str, Any]]
     errors: List[str]
```

상태 전이 다이어그램:



3.1.3 CrewAI

역할: 역할 기반 에이전트 협업 및 태스크 분배

플랫폼은 CrewAI의 패턴을 차용하여 5개의 전문 에이전트를 구성:

에이전트	역할 (Role)	담당 업무
Strategy Analyst	AI 전략 분석가	성숙도 진단, 기회 발굴, 로드맵 수립
Use Case Designer	Use Case 설계자	요건 정의, 아키텍처 설계, 거버넌스 수립
ROI Analyst	ROI 분석가	투자 효과 분석, TCO 분석, 시나리오 비교
Risk Assessor	리스크 평가 전문가	기술/조직/비즈니스/운영 리스크 평가
Report Generator	보고서 생성 전문가	Executive Summary, 전체 보고서 생성

3.1.4 AutoGen

역할: 자율적 에이전트 대화 및 피드백 루프

AutoGen 패턴은 인간-AI 협업 메커니즘에 적용:

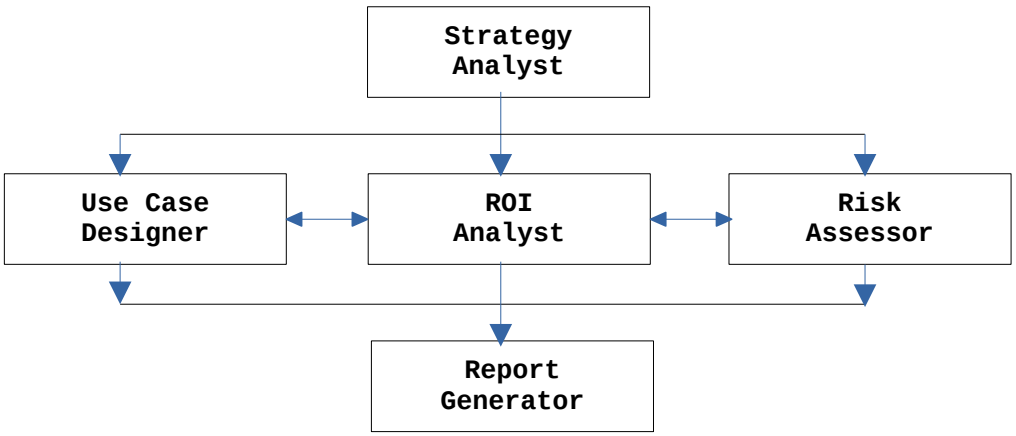
```
python
async def submit_human_feedback(
    self,
    project_id: str,
    feedback: HumanFeedback
) -> Dict[str, Any]:
    인간 전문가 피드백 제출
    피드백 저장
    project["human_feedback"].append(feedback.model_dump())

     AI 응답 생성 (AutoGen 패턴)
    ai_response = await self._process_human_feedback(project_id, feedback)
```

3.2 에이전트 간 연결 구조

파일 위치: `src/agents/agent_orchestrator.py:75-80`

```
python
def _connect_agents(self):
    에이전트 간 협업 연결
    for agent in self.agents.values():
        for other_agent in self.agents.values():
            if agent != other_agent:
                agent.connect_agent(other_agent)
```

에이전트 메시 네트워크:



4. 에이전트 오케스트레이션 시스템

4.1 ConsultingOrchestrator 클래스

파일 위치: `src/agents/agent_orchestrator.py`

Orchestrator는 전체 컨설팅 워크플로우를 관리하는 중앙 제어 컴포넌트이다:

```
python
class ConsultingOrchestrator:
    컨설팅 에이전트 오케스트레이터
```

멀티 에이전트 협업을 조율하고, 컨설팅 워크플로우를 관리합니다.
인간 전문가와 AI 에이전트 간의 협업을 지원합니다.

```
    def __init__(self):
        self.llm_provider = get_llm_provider()

        전문 에이전트 초기화
        self.agents: Dict[str, BaseConsultingAgent] = {
            "strategy": StrategyAnalystAgent(self.llm_provider),
            "designer": UseCaseDesignerAgent(self.llm_provider),
            "roi": ROIAnalystAgent(self.llm_provider),
            "risk": RiskAssessorAgent(self.llm_provider),
            "report": ReportGeneratorAgent(self.llm_provider)
        }
```

4.2 에이전트 메시지 프로토콜

파일 위치: `src/agents/base_agent.py`

```
python
class AgentMessage(BaseModel):
    에이전트 메시지
    id: str = Field(default_factory=lambda: str(uuid.uuid4()))
    sender: str            발신 에이전트 ID
    receiver: str          수신 에이전트 ID
    content: str           메시지 내용
    message_type: str      text, task, result, feedback, approval_request
    metadata: Dict[str, Any] = Field(default_factory=dict)
    timestamp: datetime = Field(default_factory=datetime.now)
```

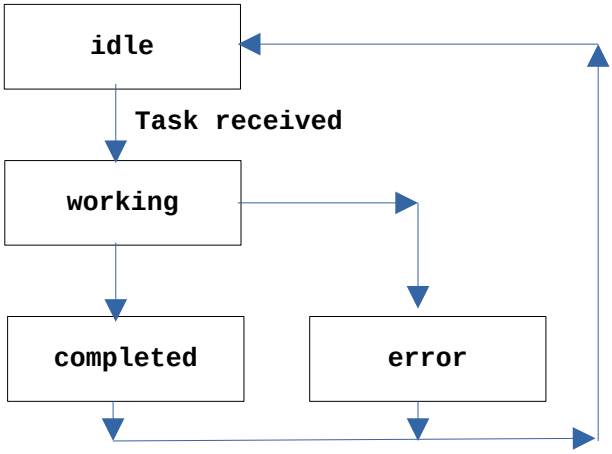
메시지 유형:

유형	설명	처리 방식
`text`	일반 텍스트 메시지	LLM 응답 생성
`task`	태스크 요청	`execute()` 메서드 호출
`result`	태스크 결과	결과 저장 및 전달
`feedback`	피드백 메시지	컨텍스트에 저장
`approval_request`	승인 요청	인간 검토 대기

4.3 에이전트 상태 관리

```
python
class AgentState(BaseModel):
    에이전트 상태
    agent_id: str
    agent_name: str
    status: str = "idle"   idle, working, waiting, completed, error
    current_task: Optional[str] = None
    progress: float = 0.0
    messages: List[AgentMessage] = Field(default_factory=list)
    context: Dict[str, Any] = Field(default_factory=dict)
    results: Dict[str, Any] = Field(default_factory=dict)
    errors: List[str] = Field(default_factory=list)
```

상태 전이:



5. 데이터 흐름 및 가중치 시스템

5.1 컨설턴트 입력 데이터 구조

파일 위치: `src/models/schemas.py`

5.1.1 기업 프로필 (CompanyProfile)

컨설턴트가 입력하는 기초자료의 핵심 구조:

```
python
class CompanyProfile(BaseModel):
    기업 프로필 (종합)
    id: Optional[str] = None
    name: str                                기업명
    industry: IndustryType                   산업 분류
    company_size: CompanySize                기업 규모
    business_description: str                사업 개요

    세부 리소스 (가중치 적용 대상)
    it_infrastructure: ITInfrastructure
    data_assets: DataAssets
    human_resources: HumanResources
    financial_resources: FinancialResources
    organizational_readiness: OrganizationalReadiness
```

5.1.2 세부 입력 항목 및 가중치 영향도

입력 영역	세부 항목	가중치 영향	영향받는 분석
IT Infrastructure	has_cloud	기술 점수 +0.5	성숙도,Use Case 적합도
	gpu_available	기술 점수 +0.5	성숙도,Use Case 적합도
	has_data_warehouse	기술 점수 +1.0	성숙도,데이터 파이프라인
	legacy_system_count	리스크 점수 영향	리스크 평가,합 복잡성
Data Assets	data_volume_tb	적합도 +10 (>1TB)	Use Case 적합도
	data_quality_score	기술 점수 +0.5 (≥3)	성숙도, 리스크
	has_data_governance	기술/프로세스점수 +1.0	성숙도, 거버넌스
Human Resources	data_scientist_count	조직 점수 +1.0 (>0)	성숙도, 적합도
	ml_engineer_count	조직 점수 +1.0 (>0)	성숙도, 실행력
	ai_experience_projects	조직 점수 +0.5 (>0)	성숙도, 리스크
Financial Resources	ai_investment_budget	시나리오 예산 기준	시나리오 생성, ROI
Organizational Readiness	xecutive_support	전략 점수 +1.0 (≥4)	성숙도, 변화 관리
	change_management_capability	리스크 감소	조직 리스크

5.2 가중치 계산 알고리즘

5.2.1 AI 성숙도 점수 계산

파일 위치: `src/agents/consulting_agents.py:110-202`

```
python
def _calculate_maturity_scores(self, company: CompanyProfile) -> Dict[str, Any]:
    성숙도 점수 계산 (규칙 기반)
    scores = {}

    전략 및 비전 점수 (기본 2.0)
    strategy_score = 2.0
    if company.organizational_readiness.executive_support >= 4:
        strategy_score += 1.0
    if company.financial_resources.ai_investment_budget > 0:
        strategy_score += 0.5

    조직 및 역량 점수 (기본 1.0)
    hr = company.human_resources
    org_score = 1.0
    if hr.data_scientist_count > 0:
        org_score += 1.0
    if hr.ml_engineer_count > 0:
        org_score += 1.0
    if hr.ai_experience_projects > 0:
        org_score += 0.5
    if hr.training_budget_ratio > 0:
        org_score += 0.5

    데이터 및 기술 점수 (기본 1.0)
    data = company.data_assets
    infra = company.it_infrastructure
```

```
    tech_score = 1.0
    if data.data_volume_tb > 1:
        tech_score += 0.5
    if data.data_quality_score >= 3:
        tech_score += 0.5
    if data.has_data_governance:
        tech_score += 1.0
    if infra.has_cloud:
        tech_score += 0.5
    if infra.gpu_available:
        tech_score += 0.5
    if infra.has_data_warehouse or infra.has_data_lake:
        tech_score += 1.0
```

5.2.2 Use Case 적합도 점수 계산

파일 위치: `src/agents/consulting_agents.py:249-271`

```
python
def _calculate_fit_score(self, use_case: Dict, company: CompanyProfile) ->
float:
    Use Case 적합도 점수 계산
    score = 50.0   기본 점수

    데이터 가용성 가중치
    if company.data_assets.data_volume_tb > 1:
        score += 10
    if company.data_assets.data_quality_score >= 3:
        score += 10

    인프라 준비도 가중치
    if company.it_infrastructure.has_cloud:
        score += 10
    if company.it_infrastructure.gpu_available:
        score += 5

    인력 역량 가중치
    if company.human_resources.data_scientist_count > 0:
        score += 10
    if company.human_resources.ai_experience_projects > 0:
        score += 5

    return min(score, 100)
```

5.2.3 리스크 점수 계산

파일 위치: `src/agents/consulting_agents.py:713-834`

```
python
 리스크 점수 = 확률 × 영향도
risk_score = probability × impact

 예시: 데이터 품질 리스크
if data_quality < 3:
    risks.append({
        "name": "데이터 품질 부족",
        "probability": 4,
        "impact": 4,
```

```
        "risk_score": 16,   4 × 4
        "mitigation": "데이터 품질 개선 프로그램 선행 필요"
    })
```

5.3 시나리오별 가중치 파라미터

파일 위치: `src/agents/agent_orchestrator.py:288-310`

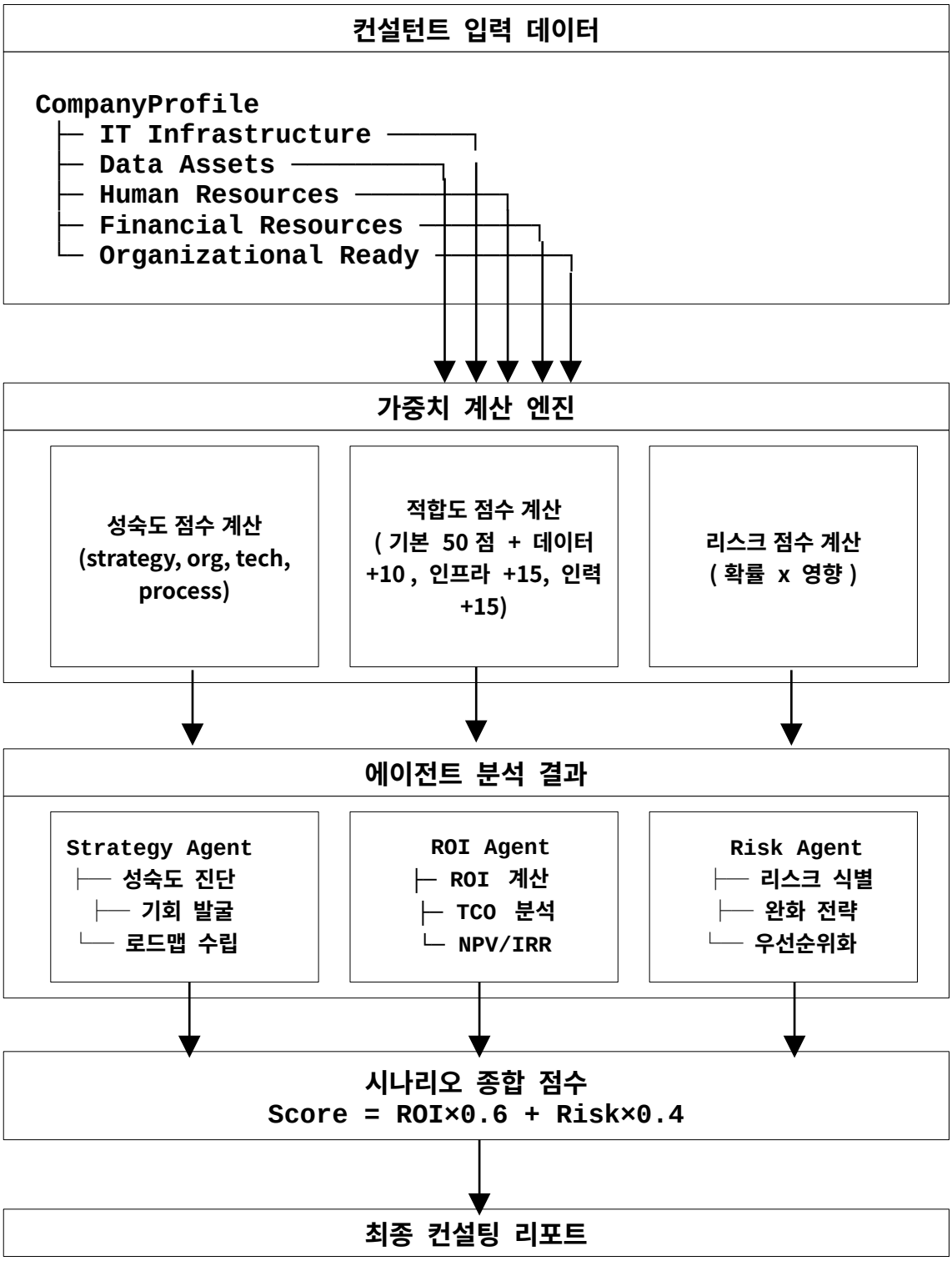
```
python
params_map = {
    "conservative": {
        "name": "보수적 시나리오",
        "budget_ratio": 0.6,       예산의 60%
        "risk_appetite": "low",
        "timeline_months": 18,
        "focus": "Quick Win 과제 집중"
    },
    "balanced": {
        "name": "균형 시나리오",
        "budget_ratio": 1.0,       예산의 100%
        "risk_appetite": "medium",
        "timeline_months": 24,
        "focus": "단기 성과와 중기 역량 구축 병행"
    },
    "aggressive": {
        "name": "적극적 시나리오",
        "budget_ratio": 1.5,       예산의 150%
        "risk_appetite": "high",
        "timeline_months": 36,
        "focus": "비즈니스 혁신 및 전사 확산"
    }
}
```

5.4 시나리오 종합 점수 계산

파일 위치: `src/agents/agent_orchestrator.py:378-388`

```
python
def _calculate_scenario_score(
    self,
    roi_result: Dict,
    risk_result: Dict
) -> float:
    시나리오 종합 점수 계산
     ROI 점수 (60% 가중치)
    roi_score = min(roi_result.get("metrics", {}).get("roi_percent", 0) / 10,
10)

    리스크 점수 (40% 가중치) - 낮을수록 좋음
    risk_score = 10 - risk_result.get("total_risk_score", 5)

    종합 점수 = ROI × 0.6 + Risk × 0.4
    return round((roi_score  0.6 + risk_score  0.4), 2)
```

5.5 데이터 흐름 다이어그램



6. 컨설팅 워크플로우 프로세스

6.1 5단계 컨설팅 프레임워크

파일 위치: `config/settings.py`

```
python
CONSULTING_FRAMEWORK = {
    "stages": [
        {
            "id": 1,
            "name": "AI 비전 및 전략 수립",
            "name_en": "Strategy",
            "activities": ["AI 성숙도 진단", "기회 발굴", "전략 및 로드맵 수립"],
            "outputs": ["AI 비전 선언문", "AI 성숙도 진단 보고서",
                        "AI 활용 사례 후보 목록", "AI 전략 로드맵"]
        },
        {
            "id": 2,
            "name": "Use Case 및 설계 정의",
            "name_en": "Define & Design",
            "activities": ["상세 요건 정의", "기술 및 아키텍처 설계",
                           "거버넌스 및 윤리 체계 수립"]
        },
        {
            "id": 3,
            "name": "플랫폼 및 솔루션 구축",
            "name_en": "Build & Implement",
            "activities": ["PoC 수행", "AI 플랫폼 구축", "솔루션 개발 및 통합"]
        },
        {
            "id": 4,
            "name": "파일럿 및 확산",
            "name_en": "Scale & Deploy",
            "activities": ["파일럿 운영", "변화 관리", "전사 확산"]
        },
        {
            "id": 5,
            "name": "운영, 모니터링 및 개선",
            "name_en": "Operate & Optimize",
            "activities": ["운영 및 모니터링", "피드백 루프 및 지속적 개선",
                           "거버넌스 관리"]
        }
    ]
}
```

6.2 전체 워크플로우 실행 프로세스

파일 위치: `src/agents/agent_orchestrator.py:595-649`

```
python
async def run_full_consultation(
    self,
    project_id: str,
    auto_approve: bool = False
) -> Dict[str, Any]:
```

전체 컨설팅 워크플로우 실행

1단계: 전략 수립
```
maturity = await self.run_maturity_assessment(project_id)
opportunities = await self.identify_opportunities(project_id)
roadmap = await self.create_roadmap(project_id)
```

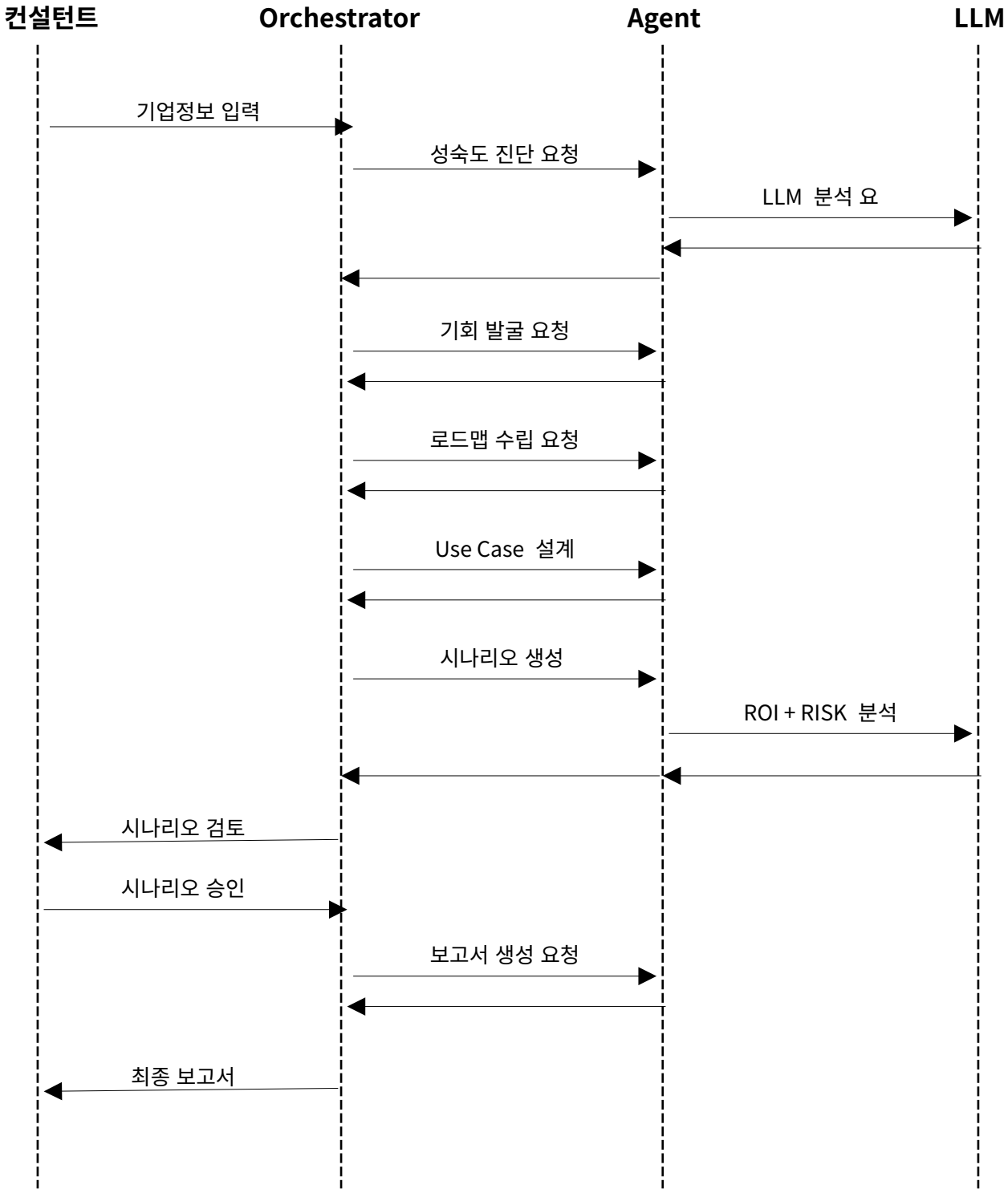
2단계: Use Case 설계 (상위 3개)
```
for i in range(min(3, len(project["use_cases"]))):
    design = await self.design_use_case(project_id, i)
```

시나리오 생성
```
scenarios = await self.generate_scenarios(project_id)
```

자동 승인 모드
```
if auto_approve and scenarios:
    best_scenario = max(scenarios, key=lambda x: x.get("overall_score", 0))
    approval = await self.approve_scenario()
```

보고서 생성
```
report = await self.generate_report(project_id, "executive_summary")
```

6.3 워크플로우 시퀀스 다이어그램



7. 최종 컨설팅 리포트 생성 메커니즘

7.1 Report Generator Agent

파일 위치: `src/agents/consulting_agents.py:873-1031`

```
python
class ReportGeneratorAgent(BaseConsultingAgent):
    보고서 생성 에이전트

    def get_system_prompt(self) -> str:
        return 당신은 AI 컨설팅 보고서 작성 전문가입니다.

        [전문 영역]
        1. 경영진용 Executive Summary 작성
        2. 기술 보고서 작성
        3. 전략 제안서 작성
        4. 실행 로드맵 문서화

        [작성 원칙]
        - 명확하고 간결한 문장
        - 데이터 기반의 객관적 서술
        - 핵심 메시지의 명확한 전달
        - 전문적이면서도 이해하기 쉬운 표현
```

7.2 리포트 데이터 수집

파일 위치: `src/agents/agent_orchestrator.py:503-550`

```
python
async def generate_report(
    self,
    project_id: str,
    report_type: str = "executive_summary"
) -> Dict[str, Any]:
    컨설팅 보고서 생성

    보고서 데이터 수집 (모든 에이전트 결과 종합)
    report_data = {
        "overview": {
            "project_id": project_id,
            "company": project["company_profile"].get("name", "Unknown"),
            "industry": project["company_profile"].get("industry", "Unknown")
        },
        "assessment": project["maturity_assessment"],       Strategy Agent
        "use_cases": project["use_cases"],                  Designer Agent
        "scenarios": project["scenarios"],                  ROI + Risk Agents
        "selected_scenario": self._get_selected_scenario(project),
        "recommendations": self._compile_recommendations(project),
        "expected_benefits": self._compile_benefits(project),
        "roadmap": self._compile_roadmap(project)
    }
```

7.3 보고서 구조

파일 위치: `src/services/report_generator.py`

7.3.1 Executive Summary 구조

```
python
def generate_executive_summary(self, project_data: Dict) -> Dict:
    report = {
        "title": f"{company_name} AI 전환 컨설팅 Executive Summary",
        "sections": [
            {"title": "1. 프로젝트 개요", "content": },
            {"title": "2. AI 성숙도 진단 결과", "content": },
            {"title": "3. 시나리오 분석", "content": },
            {"title": "4. 권고사항 및 Next Steps", "content": }
        ]
    }
```

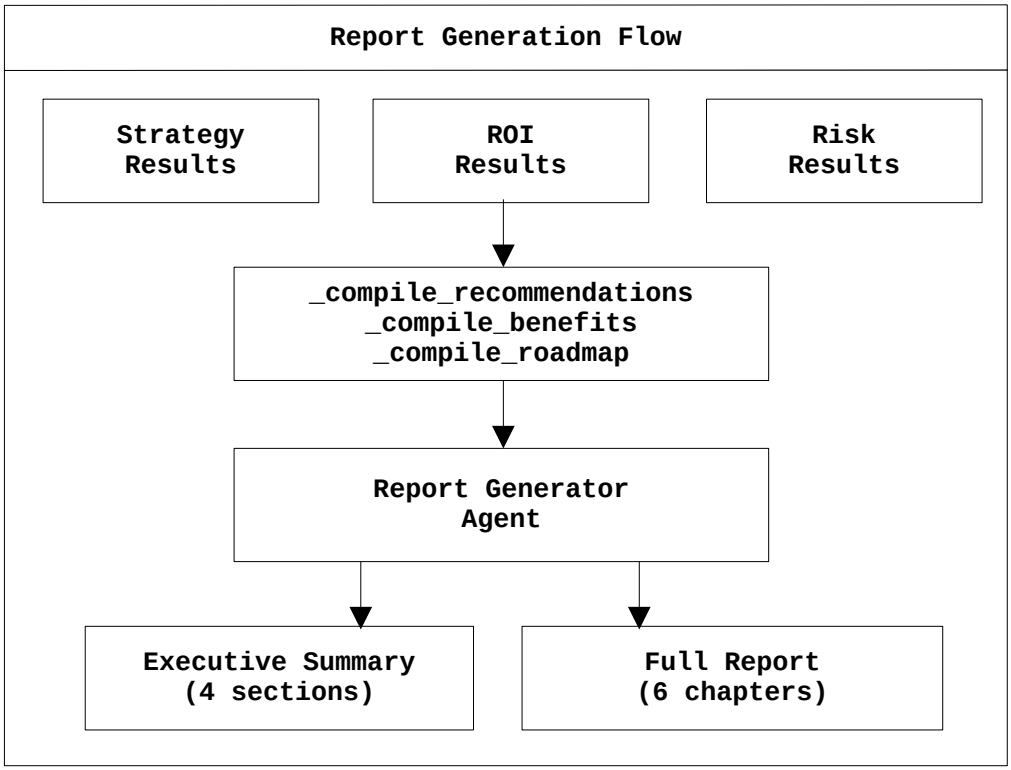
7.3.2 전체 보고서 구조

```
python
def generate_full_report(self, project_data: Dict) -> Dict:
    report = {
        "chapters": [
            {"number": 1, "title": "개요",
             "sections": ["배경 및 목적", "프로젝트 범위", "추진 경과"]},
            {"number": 2, "title": "현황 분석",
             "sections": ["AI 성숙도 진단", "IT 인프라 분석",
                          "조직 역량 분석", "데이터 자산 분석"]},
            {"number": 3, "title": "AI 전략 수립",
             "sections": ["AI 비전", "핵심 Use Case", "우선순위 결정"]},
            {"number": 4, "title": "실행 계획",
             "sections": ["시나리오 분석", "로드맵", "투자 계획"]},
            {"number": 5, "title": "기대 효과",
             "sections": ["정량적 효과", "정성적 효과", "ROI 분석"]},
            {"number": 6, "title": "리스크 관리",
             "sections": ["리스크 식별", "완화 전략"]}
        ]
    }
```

7.4 입력 데이터가 최종 리포트에 미치는 영향

입력 데이터	영향받는 리포트 섹션	영향 방식
Company Profile	1장 개요	기업 정보, 산업 분류 표시
IT Infrastructure	2장 현황 분석	인프라 현황 테이블 생성
Data Assets	2장 현황 분석	데이터 자산 분석 섹션
Human Resources	2장 현황 분석	조직 역량 분석 섹션
Maturity Assessment	2장 현황 분석	성숙도 점수 테이블, 레이더 차트
Use Cases	3장 AI 전략	Use Case 목록, 우선순위 매트릭스
Scenarios	4장 실행 계획	시나리오 비교표, 투자 계획
ROI Analysis	5장 기대 효과	ROI 지표 테이블, 비용-효과 분석
Risk Assessment	6장 리스크 관리	리스크 매트릭스, 완화 전략

7.5 리포트 생성 플로우



8. 보안 및 거버넌스 통합

8.1 감사 로깅 시스템

파일 위치: `src/security/audit_logger.py`

```
python
class AuditEventType(Enum):
    SYSTEM_START = "system_start"
    SYSTEM_STOP = "system_stop"
    PROJECT_CREATED = "project_created"
    ANALYSIS_COMPLETED = "analysis_completed"
    REPORT_GENERATED = "report_generated"
    USER_ACTION = "user_action"

class SeverityLevel(Enum):
    INFO = "info"
    WARNING = "warning"
    ERROR = "error"
    CRITICAL = "critical"
```

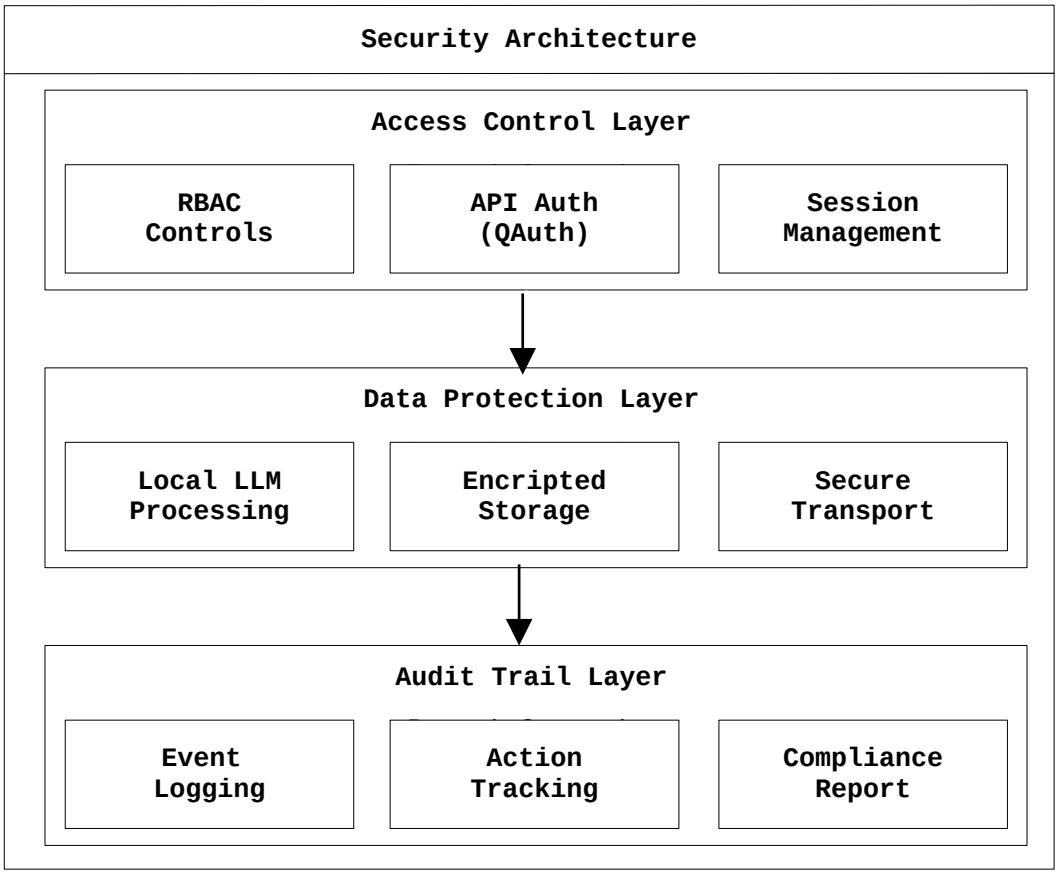
8.2 MLOps 보안 및 거버넌스 표준

파일 위치: `config/settings.py`

```
python
MLOPS_STANDARDS = {
    "sections": [
```

```
        {
            "id": "security_governance",
            "name": "5.7 보안 및 거버넌스 통합",
            "areas": [
                {
                    "name": "접근 통제 (Access Control)",
                    "standard": "RBAC 구현, 최소 권한 원칙",
                    "tools": ["Keycloak", "AWS IAM", "Azure AD"]
                },
                {
                    "name": "파이프라인 감사 추적",
                    "standard": "모든 작업 단계 자동 기록",
                    "tools": ["MLflow Tracking", "Apache Atlas"]
                },
                {
                    "name": "보안 스캐닝",
                    "standard": "코드 및 컨테이너 보안 취약점 스캐닝",
                    "tools": ["Snyk", "Trivy", "SonarQube"]
                }
            ]
        }
    ]
}
```

8.3 데이터 보호 아키텍처



부록: API 엔드포인트 목록

프로젝트 관리 API

Method	Endpoint	설명
POST	`/api/v1/projects`	새 프로젝트 생성
GET	`/api/v1/projects/{id}`	프로젝트 상세 조회
GET	`/api/v1/projects/{id}/status`	프로젝트 상태 조회

전략 수립 API (1단계)

Method	Endpoint	설명
POST	`/api/v1/projects/{id}/maturity-assessment`	AI 성숙도 진단
POST	`/api/v1/projects/{id}/opportunities`	기회 발굴
POST	`/api/v1/projects/{id}/roadmap`	로드맵 수립

Use Case 설계 API (2단계)

Method	Endpoint	설명
POST	`/api/v1/projects/{id}/use-cases/{idx}/design`	Use Case 상세 설계

시나리오 분석 API

Method	Endpoint	설명
POST	`/api/v1/projects/{id}/scenarios`	시나리오 생성
GET	`/api/v1/projects/{id}/scenarios`	시나리오 목록 조회
POST	`/api/v1/projects/{id}/scenarios/{sid}/approve`	시나리오 승인

협업 API

Method	Endpoint	설명
POST	`/api/v1/projects/{id}/feedback`	피드백 제출
GET	`/api/v1/projects/{id}/feedback`	피드백 이력 조회

보고서 API

Method	Endpoint	설명
POST	`/api/v1/projects/{id}/reports`	보고서 생성
GET	`/api/v1/projects/{id}/reports`	보고서 목록 조회

전체 워크플로우 API

Method	Endpoint	설명
POST	`/api/v1/projects/{id}/run-full-consultation`	전체 컨설팅 실행

결론

AI Consulting Assistant Platform은 하이브리드 LLM 아키텍처와 멀티 에이전트 협업 시스템을 통해 기업 AI 전환 컨설팅을 체계적으로 지원한다.

핵심 특징:
1. 보안 중심 설계: Ollama 기반 Local LLM으로 민감한 기업 데이터 보호
2. 역할 기반 협업: 5개 전문 에이전트의 분업과 협업
3. 가중치 기반 분석: 컨설턴트 입력 데이터의 정량적 평가
4. 자동화된 워크플로우: 5단계 컨설팅 프레임워크 자동 실행
5. 종합 리포트 생성: 모든 분석 결과를 통합한 전문 보고서 자동 생성

이 플랫폼은 인간 컨설턴트의 전문성과 AI의 분석 능력을 결합하여 고품질 AI 컨설팅 서비스를 제공하는 것을 목표로 한다. 인공지능 시스템, 플랫폼 도입을 희망하는 기업을 대상으로 전체적인 컨설팅과 AI 인프라스트럭쳐의 설계, 구현, 실제 업무에 적용까지 구축하는 전체 과정을 수행하는데 있어서 필요한 전문 컨설턴트가 부족한 상황에서도 컨설팅 업무를 수행할 수 있도록 하기 위해 설계하였다.

문서 끝