BeSir Whitepaper Basic

BeSir 백서: 선언에서 실행으로 - AI Execution Layer를 통한 전사 AI 전략의 완성

1. 서론: 왜 엔터프라이즈 AI는 PoC의 늪에 빠지는가?

대부분의 대기업은 '전사 AI 전략'을 선언하며 AI 기반 업무 혁신을 약속하지만, 그 결과는 대부분 실망스럽습니다. AI 프로젝트는 매년 반복되지만 실제 운영 환경으로 확산되지 못하고, 전사 AI 전략은 PoC 목록을 나열한 'PoC 컬렉션'으로 남게 됩니다. AI는 여전히 문서 요약이나 FAQ 응답과 같은 보조적인 역할에 머물러 있습니다.

이러한 실패의 근본 원인은 AI 모델의 성능 부족이 아닙니다. 오늘날 기업들은 이미 충분히 강력한 LLM(대규모 언어 모델)을 활용할 수 있는 환경에 있습니다. 진짜 문제는 기술이 아닌 **구조**에 있습니다. 기존의 모든 엔터프라이즈 시스템은 '사람이 사용한다'는 대전제 아래 설계되었습니다. 사람이 화면을 보고, 맥락을 이해하며, 버튼을 클릭하여 결과를 해석하는 구조 위에서 AI는 외부 도구로 존재할 뿐, 시스템의 진정한 사용자가 될 수 없습니다. 본 백서는 이 구조적 문제의 본질을 분석하고, AI를 시스템의 '실행 주체'로 만드는 새로운 패러다임과 그 해법을 제시합니다.

현재 접근 방식의 한계 분석

현재 기업들이 채택하는 AI 도입 방식은 크게 세 가지로 요약할 수 있습니다. 그러나 각 방식은 부분적인 성과에 그치며 전사적 확산이라는 목표에는 도달하지 못하는 명백한 구조적 한계를 내포하고 있습니다.

- **챗봇 / Copilot: "똑똑한 상담원이지만 실행은 못하는 상황"** 자연어 질의응답, 문서 요약 등을 통해 일부 생산성 향상을 가져오지만, 핵심 업무 시스템과 분리된 '조언자' 역할에 그칩니다. AI는 무엇을 해야 할지 알려줄 수는 있지만, 시스템에 접속해 직접 업무를 실행할 권한이나 방법이 없습니다. 똑똑한 인턴에게 보고서 작성은 맡길 수 있지만, 실제 결재나 시스템 조작은 다시 담당자가 해야 하는 것과 같습니다.
- **BI / 리포트 자동화: "계기판일 뿐 운전자가 아닌 상황"** 자연어 기반으로 대시보드를 생성하거나 리포트를 자동화합니다. 자동차 계기판이 속도나 연료 상태는 알려주지만 직접 운전하지 못하는 것처럼, 이 방식의 AI는 정해진 데이터의 '결과'만 보여줍니다. '무엇(What)'이 일어났는지는 보여주지만, '왜(Why)'라는 질문에 답하기 위해 여러 시스템의 데이터를 넘나들며 맥락을 분석하고 새로운 가설을 검증하는 '운전자'의 역할은 수행하지 못합니다.

- **RPA / 자동화: "정해진 동작만 반복하는 자동 버튼"** 정형화된 시나리오에 따라 반복적인 작업을 자동화하지만, 업무의 '의미'를 이해하지 못하고 정해진 UI나 스크립트 경로만을 따릅니다. 엘리베이터 버튼처럼 정해진 조건에서는 완벽하게 작동하지만, 화면 구조가 조금만 바뀌거나 예외 상황이 발생하면 쉽게 멈춰 버립니다. AI의 진정한 가치인 '의미 기반 판단'이 필요한 복잡한 업무에는 적용할 수 없습니다.

문제의 본질 정의

위 세 가지 접근 방식의 실패는 모두 하나의 공통된 원인으로 귀결됩니다: **"AI가 시스템의 '실행 주체'가 아니다."** 이 구조적 문제 때문에 AI 도입은 다음과 같은 값비싼 실패 패턴을 반복하게 됩니다.

1. **모델 도입:** 강력한 최신 AI 모델을 도입합니다.
2. **PoC 성공:** 제한된 데이터와 시나리오 안에서 인상적인 결과를 보여주며 PoC는 성공으로 보고됩니다.
3. **확장 시도:** 성공적인 PoC를 실제 운영 환경으로 확장하려 합니다.
4. **기존 시스템과 충돌:** 수백 개의 테이블, 복잡한 권한 체계, 분산된 시스템 등 실제 운영 환경의 복잡성과 마주하며 AI는 직접 시스템을 다루지 못합니다.
5. **중단:** 보안, 권한, 책임, 감사 문제에 부딪히며 프로젝트는 파일럿 단계에서 중단됩니다.
6. **새로운 PoC 시작:** 문제는 해결되지 않은 채, 또 다른 기술로 새로운 PoC가 시작됩니다.

이 악순환을 끊기 위해서는 우리가 던져야 할 질문 자체를 바꿔야 합니다.

질문의 전환 제시

지금까지 우리는 "어떤 AI 모델을 쓸 것인가?" 혹은 "AI를 어디에 붙일 것인가?"를 물어왔습니다. 하지만 이 질문들은 AI를 영원히 시스템 외부의 '애드온(Add-on)'으로 남게 할 뿐입니다. 이제 우리는 다음과 같이 질문해야 합니다.

"AI가 기존 시스템을 직접 이해하고 실행하게 하려면 어떤 구조가 필요한가?"

이 질문이야말로 전사 AI 전략을 '선언'에서 '실행'으로 전환시키는 핵심 열쇠입니다. 이 새로운 질문에 대한 해답이 바로 다음에 설명할 새로운 패러다임의 시작입니다.

---

2. 패러다임의 전환: AI를 '사용자'로 만드는 AI Execution Layer

기존의 '사람 중심 시스템 + AI 애드온' 방식은 AI를 영원히 시스템 외부의 보조 도구로 머물게 합니다. AI가 시스템의 내부 구조와 데이터의 업무적 의미를 이해하지 못하고, 권한과 책

임 모델이 부여되지 않은 상태에서는 아무리 모델 성능이 발전해도 전사적 확산은 불가능합니다. 따라서 단기적인 기능 추가가 아닌, AI가 시스템의 일원이 될 수 있도록 하는 근본적인 패러다임의 전환이 전략적으로 필수적입니다.

'Human-on-the-loop' 개념 설명

기존 시스템은 `Human-in-the-loop` 구조를 따릅니다. 이는 모든 업무 흐름의 중심에 항상 사람이 위치하며, AI는 사람이 요청한 정보를 제공하는 보조적인 역할을 수행하는 모델입니다. 이 구조에서는 최종 판단과 실행의 책임이 전적으로 사람에게 있습니다.

BeSir가 제안하는 새로운 패러다임은 `Human-on-the-loop` 입니다. 이 구조에서는 **AI가 기본 실행 주체가 되고, 사람은 감독자(Supervisor)가 됩니다.** AI는 데이터 조회, 분석, 실행 제안까지 스스로 수행하며, 사람은 AI의 실행 과정을 감독하고 예외 상황이나 최종 책임을 판단하는 역할에 집중합니다. 이를 통해 사람은 반복적인 분석 작업에서 벗어나 더 높은 수준의 의사결정에 집중할 수 있는 새로운 협업 모델이 완성됩니다.

AI Execution Layer 개념 정의

이러한 `Human-on-the-loop` 패러다임을 구현하기 위한 핵심적인 기술 구조가 바로 '**AI Execution Layer**'입니다. 이 레이어는 기존의 엔터프라이즈 시스템과 AI 모델 사이에 위치하는 중간 계층으로, AI가 마치 사람 사용자처럼 시스템을 이해하고, 정해진 규칙과 통제 하에 시스템을 실행할 수 있도록 만드는 역할을 수행합니다. AI Execution Layer는 다음 네 가지 핵심 기능을 책임집니다.

- **의미 이해 (Semantic Understanding):** 시스템의 데이터 구조(테이블, 컬럼)를 넘어, 데이터가 가지는 업무적 개념, 관계, 맥락을 AI가 이해할 수 있도록 **Ontology**를 통해 의미적으로 구조화합니다.
- **실행 가능성 추상화 (Executable Capability):** 조회, 계산, 데이터 변경, 승인 요청 등 실제 업무 행위를 AI가 호출하고 실행할 수 있는 표준화된 단위로 정의합니다. 이는 **Capability Graph**를 통해 구현됩니다.
- **실행 통제 (Governed Execution):** AI의 모든 실행은 사전에 정의된 권한, 승인 절차, 감사 추적의 규칙 안에서만 이루어지도록 통제합니다. 이는 전사 보안 및 거버넌스 정책을 준수하기 위한 필수 요소입니다.
- **사람과 AI의 협업 인터페이스:** AI가 실행 주체가 되고 사람이 감독자 역할을 수행하는 `Human-on-the-loop` 구조를 지원하는 워크스페이스를 제공합니다.

패러다임 전환 요약

기존 방식과 BeSir가 제안하는 새로운 방식의 차이점은 다음 표와 같이 명확하게 비교할 수 있습니다.

구분	기존 방식	BeSir 방식
AI 위치	외부 도구 (Add-on)	시스템 사용자 (User)
시스템 이해	없음 (개발자 의존)	Ontology 기반 (구조적 이해)
실행 주체	사람 중심 (Human-in-the-loop)	AI 중심 + 사람 승인 (Human-on-the-loop)
확산 방식	PoC 단위 (개별 프로젝트)	전사 표준 (구조 자산)
유지보수	반복적 재개발	구조 자산 재사용

AI Execution Layer라는 새로운 개념을 표준화된 플랫폼 형태로 구현한 것이 바로 BeSir입니다. 다음 장에서는 BeSir가 구체적으로 어떤 아키텍처를 통해 이 패러다임을 현실로 만드는지 자세히 살펴보겠습니다.

---

3. BeSir 플랫폼 아키텍처: 설계, 실행, 통제의 3계층 구조

BeSir는 전사 AI 전략이 마주하는 현실적인 요구사항, 즉 기존 시스템의 변경 최소화, 전사 보안 정책의 완전한 준수, 통제 가능한 실행 보장 등을 충족시키기 위해 설계되었습니다. 이 아키텍처는 "AI는 무엇을 이해해야 하는가?", "무엇을 실행할 수 있는가?", 그리고 "어떻게 통제되는가?"라는 세 가지 핵심 질문에 명확하게 답하기 위한 3계층 구조로 이루어져 있습니다.

BeSir 한 문장 정의 재확인

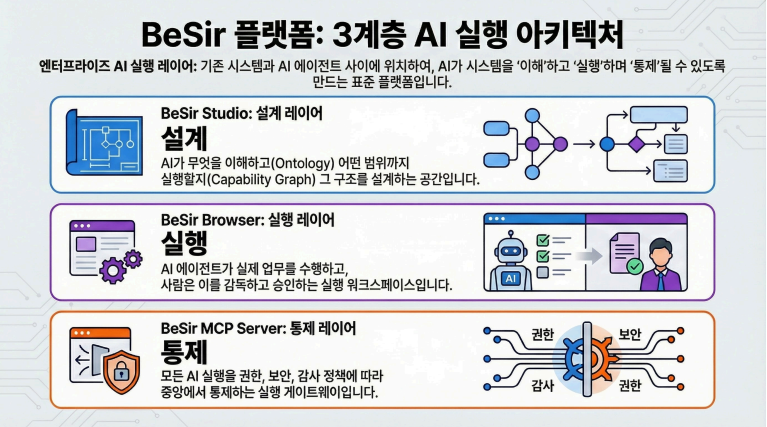
BeSir의 핵심 정체성을 다시 한번 명확히 정의하면 다음과 같습니다.

"BeSir는 기존 엔터프라이즈 시스템을 재개발 없이 AI Agent가 이해하고 실행할 수 있도록 만드는 Enterprise AI Execution Layer 플랫폼이다."

이 정의는 다음과 같은 중요한 전제를 포함합니다.

- **기존 시스템 유지:** ERP, PMS, Legacy System 등 기존 시스템은 그대로 유지됩니다.
- **기존 정책 존중:** 데이터 모델, API, 기존의 권한 체계 역시 존중됩니다.
- **단계적 권한 모델:** AI는 "조회 → 분석 → 제안 → 승인 기반 실행"의 단계를 따라, 데이터 변경과 같은 민감한 작업은 반드시 사람의 승인을 거칩니다.
- **책임 및 감사 보장:** 실행의 책임과 감사 가능성이 구조적으로 보장됩니다.

3계층 아키텍처 분석



BeSir는 AI 실행 구조의 설계, 실제 업무에서의 실행, 그리고 모든 과정을 통제하는 거버넌스의 역할을 명확히 분리한 세 개의 핵심 레이어로 구성됩니다.

- **설계 레이어 (Design Layer): BeSir Studio**
  - **역할:** AI가 시스템을 어떻게 이해하고, 어떤 범위까지 실행할 수 있는지를 정의하는 **설계 공간**입니다. 이곳에서 AI를 위한 의미 구조(Ontology)와 실행 구조(Capability Graph)가 만들어집니다.
  - **핵심:** 가장 중요한 설계 원칙은 **현업과 개발자의 역할 분리**입니다. 현업 전문가는 데이터의 '의미'를 정의하고, 개발자는 그 구조가 '안전하게 실행'되도록 책임집니다. 이를 통해 기술 지식이 없는 현업도 AI 설계에 직접 참여할 수 있습니다.
- **실행 레이어 (Execution Layer): BeSir Browser**
  - **역할:** 설계된 AI Agent를 사용자가 실제 업무에서 활용하는 **실행 인터페이스**입니다.
  - **핵심:** 단순한 챗봇 UI가 아닌, **대화 + 데이터 조인 + 인사이트 생성 중심의 워크스페이스**를 지향합니다. 사용자는 자연어 대화를 통해 여러 시스템에 흩어져 있는 방대한 데이터를 단번에 조회하고, 그 결과를 즉시 분석 리포트나 대시보드로 변환할 수 있습니다. 데이터 변경은 항상 사람의 승인을 기반으로 이루어집니다.
- **통제 레이어 (Governance Layer): BeSir Agent Server**
  - **역할:** AI Agent와 기존 시스템 사이에서 모든 실행 요청을 통제하는 **실행 게이트웨이**입니다.
  - **핵심: 접근 제어(RBAC), 실행 감사, 망분리 환경 대응** 등 엔터프라이즈의 엄격한 보안 및 거버넌스 요구사항을 책임집니다. 모든 AI의 실행은 반드시 Agent Server를 경유하므로, 권한 없는 접근이나 무단 실행이 원천적으로 차단되며 모든 활동은 로그로 기록됩니다.

이 세 가지 레이어는 유기적으로 결합하여 AI가 안전하고 투명하게, 그리고 통제된 방식으로 기업의 일을 수행하게 만듭니다. 그 중에서도 BeSir의 가장 근본적인 경쟁력은 모든 것의 시작점인 '설계'의 철학과 그 결과물인 핵심 자산에 있습니다. 다음 장에서는 이에 대해 더 깊이 알아보겠습니다.

---

4. BeSir의 핵심 IP: '코드'가 아닌 '구조'를 설계하다

대부분의 엔터프라이즈 AI 프로젝트가 실패하는 근본적인 이유는, 시스템의 업무적 '의미'와 '맥락'이 구현 코드 속에 갇혀 담당자가 바뀌거나 시스템이 변경되면 그대로 사라지기 때문입니다. 코드는 남지만, 왜 그렇게 만들었는지에 대한 지식은 유실됩니다. BeSir는 이 문제를 해결하기 위해 AI 시대의 핵심 자산은 구현 코드가 아니라, "**무엇을 이해하고(Ontology), 무엇을 실행할 수 있는지(CapGraph)**"를 명시적으로 정의한 '구조 자산'이라는 핵심 철학을 제시합니다.

Ontology의 개념과 가치

Ontology는 AI가 조직의 시스템을 '개념적 세계'로 이해하기 위한 의미 구조입니다. 이는 단순히 데이터를 어떻게 저장하는지를 기술하는 ERD(Entity-Relationship Diagram)와는 목적과 관점이 근본적으로 다릅니다. 이 둘의 차이는 다음과 같이 명확하게 정의할 수 있습니다.

ERD는 “어떻게 저장되는가”이고, Ontology는 “무엇을 의미하는가”이다.

ERD가 'SALES_TBL' 테이블과 'PROD_ID' 컬럼을 보여준다면, Ontology는 'SALES_TBL'이 '매출'이라는 개념을 나타내고 'PROD_ID'가 '제품'과 관계를 맺고 있음을 AI에게 알려줍니다. Ontology가 구축되면 AI는 "이번 분기 매출 감소 원인을 제품 관점에서 분석해줘"와 같은 복합적인 질문의 의도를 구조적으로 해석하고, 여러 테이블에 흩어진 데이터를 하나의 '개념'으로 통합하여 단순 질의응답을 넘어선 '**이해 기반 추론**'을 수행할 수 있게 됩니다.

CapGraph의 역할과 구조

Ontology가 AI의 '이해'를 담당한다면, CapGraph(Capability Graph)는 AI의 '행동'을 정의합니다. 실제 업무는 'A 다음 B 다음 C'와 같은 선형적 흐름이 아니라 조건에 따라 분기되고, 이전 단계의 결과에 따라 다음 행동이 달라집니다. CapGraph는 이러한 비선형적 업무 흐름을 그래프 구조로 표현합니다.

- **Node (노드):** 조회, 계산, 변경 요청 등 AI가 호출할 수 있는 개별 실행 단위(Capability)를 의미합니다.
- **Edge (엣지):** 노드 간의 조건, 흐름, 제약사항을 정의합니다.

이 구조를 통해 AI는 단순히 정해진 함수를 호출하는 것이 아니라, 주어진 상황에서 가능한 실행 경로들을 탐색하고 최적의 행동을 '선택하고 판단'할 수 있게 됩니다.

Ontology와 CapGraph의 결합 시너지

1장에서 언급한 "똑똑한 인턴에게 메뉴판만 준 상황"을 떠올려 보십시오. Ontology는 인턴에게 주방의 전체 설계도, 모든 재료의 용도, 그리고 셰프의 레시피를 주는 것과 같습니다. CapGraph는 실제로 요리를 할 수 있는 권한과 규칙을 부여하는 것과 같습니다. 이 두 가지가 없다면 AI는 메뉴판만 든 똑똑한 인턴에 머물지만, 이들이 결합될 때 비로소 유능한 셰프의 보조가 됩니다.

이 두 구조가 결합된 AI Agent는 다음과 같은 예측 가능한 방식으로 동작합니다.

1. **질문 의미 해석:** 사용자의 자연어 질문을 **Ontology**를 통해 업무적 개념으로 해석합니다.
2. **관련 데이터 범위 결정:** 해석된 개념과 연결된 데이터의 범위를 식별합니다.
3. **실행 가능 경로 탐색:** 해당 데이터를 다룰 수 있는 최적의 실행 경로를 **CapGraph**에서 탐색합니다.
4. **권한 기반 실행/제안:** 조직의 권한 및 승인 규칙에 따라 조회는 직접 실행하고, 데이터 변경 등은 사용자에게 승인을 요청하는 형태로 제안합니다.

이처럼 AI는 즉흥적으로 행동하는 것이 아니라, 사전에 명확하게 설계된 '구조' 안에서만 투명하고 예측 가능하게 움직입니다.

구조 자산의 전략적 가치

Ontology와 CapGraph로 대표되는 '구조 자산'은 일회성 프로젝트의 산출물이 아닙니다. 이는 조직 전체에서 재사용되고 확장될 수 있는 핵심 지적 재산입니다. 조직이 고유의 Ontology와 CapGraph를 구축하는 것은 AI를 위한 비즈니스 로직의 독점적인 '지도'를 만드는 것과 같습니다. 이 구조화된 지식은 쉽게 복제할 수 없는 강력한 경쟁 우위가 되며, 기술 종속이 아닌 축적된 가치에 기반한 건강한 '락인(Lock-in)' 효과를 창출합니다. 이 자산이 축적될수록 조직의 AI 실행 능력은 기하급수적으로 향상됩니다.

이렇게 설계된 구조 자산을 기반으로 AI가 실제로 어떻게 신뢰성 있게 작업을 수행하고 엄격하게 통제되는지, 그 구체적인 실행 메커니즘을 다음 장에서 자세히 살펴보겠습니다.

---

5. 신뢰할 수 있는 AI 실행: 추적, 분석, 그리고 거버넌스

AI가 아무리 똑똑하게 분석하고 제안하더라도, 그 실행 과정이 '블랙박스'로 남아 신뢰할 수 없다면 실제 운영 환경에 도입될 수 없습니다. 엔터프라이즈 환경에서는 AI의 모든 행동이

왜 그렇게 판단했는지 설명 가능해야 하며(Explainability), 모든 실행 과정이 추적되고 감사(Audit)될 수 있어야 합니다. BeSir는 이러한 기업의 엄격한 요구사항을 '추적 가능한 실행 구조'와 '중앙 집중식 거버넌스'를 통해 구조적으로 해결합니다.

T-PTC 기반의 추적 가능한 실행 구조

BeSir Browser의 기술적 핵심은 **T-PTC (Traceable Programmatic Tool Calling)**입니다. 이는 일반적인 LLM의 Tool Calling 기능이 가진 한계, 즉 실행 과정이 불투명하고 추적이 어렵다는 문제를 해결하기 위해 고안된 구조입니다. T-PTC는 AI의 역할과 실제 데이터 처리의 역할을 명확하게 분리합니다.

- **AI의 역할:** 사용자의 의도를 파악하고, 어떤 Tool을 어떤 순서로 호출할지 결정하는 **'판단과 오케스트레이션'**을 담당합니다.
- **Tool의 역할:** 명시적인 입력값을 받아 실제 데이터를 조회하고 계산하는 **'코드 기반의 실행'**을 담당합니다.

이 구조는 두 가지 핵심적인 비즈니스 가치를 제공합니다. 첫째, AI의 모든 실행 경로는 '어떤 Tool이, 어떤 파라미터로, 언제 호출되었는지' 코드 레벨에서 100% 추적 가능해져 **설명 가능성**과 **결과의 신뢰성**을 확보합니다. 둘째, 대규모 원본 데이터를 LLM으로 전송하는 대신, Tool 내부에서 처리하고 요약된 결과만 AI에게 전달함으로써 LLM의 토큰 한계와 비용 문제를 구조적으로 해결합니다. 실제로 POSCO DX PoC에서는 이 구조를 통해 **대규모 분석 시 입력 토큰을 98~99% 절감**하며 대규모 엔터프라이즈 데이터 처리에 대한 확장성과 효율성을 증명했습니다.

자연어 기반 동적 대시보드 생성 기능

BeSir Browser는 현업 사용자의 데이터 접근성을 극대화하기 위해 자연어 기반 대시보드 자동 생성 기능을 제공합니다. 기존에는 BI 전문가에게 의뢰하여 며칠씩 걸리던 대시보드 제작을, 이제는 사용자가 자연어로 요청하는 즉시 AI가 동적으로 생성해줍니다.

그 과정은 다음과 같습니다.

1. 사용자가 "이번 분기 지역별 매출 추이와 재고 회전율이 낮은 품목을 함께 보여줘"라고 요청합니다.
2. AI는 Ontology를 통해 '매출', '재고 회전율' 등의 의미를 해석합니다.
3. 필요한 데이터를 조회하고 가공하는 Tool들을 순차적으로 호출합니다.
4. 결과 데이터를 바탕으로 최적의 쿼리, 차트 유형, 레이아웃까지 포함된 완전한 대시보드를 동적으로 생성하여 사용자에게 제시합니다.

이 기능은 BI 전문가에 대한 의존도를 획기적으로 낮추고, 현업 담당자가 데이터에 기반하여 스스로 질문하고 답을 찾는 데이터 중심 의사결정 문화를 확산시키는 강력한 도구가 됩니다.

엔터프라이즈 거버넌스 체계

BeSir Agent Server는 AI 실행의 중앙 통제 게이트웨이로서, 전사 AI 거버넌스를 책임지는 핵심 역할을 수행합니다. 모든 AI의 실행 요청은 반드시 Agent Server를 거치며, 다음과 같은 세 가지 핵심 영역을 통해 통제됩니다.

- **접근 제어 (RBAC - Role-Based Access Control):** 사용자, AI Agent, 그리고 개별 Tool 별로 역할을 기반으로 한 세분화된 권한 제어를 수행합니다. 이를 통해 AI는 허용된 범위 내에서만 데이터를 조회하고 기능을 실행할 수 있습니다.
- **승인 기반 실행:** BeSir는 '조회(Read)는 자동으로, 데이터 변경(Create/Update/Delete)은 반드시 승인 절차를 거친다'는 명확한 원칙을 따릅니다. AI가 데이터 변경을 제안하면 Agent Server는 자동으로 승인 워크플로우를 생성하여 책임자의 승인을 받도록 강제합니다. AI는 스스로 판단할 수는 있지만, 스스로 결정할 수는 없습니다.
- **감사 로그:** AI의 모든 실행에 대해 `누가(실행 주체)`, `언제(시간)`, `무엇을(Tool과 파라미터)`, `어떻게(결과)` 했는지 상세한 로그를 기록합니다. 이 로그는 사후 감사 및 문제 발생 시 책임 추적을 위한 명확한 근거를 제공하여 운영의 신뢰성을 보장합니다.

BeSir가 제공하는 이러한 독창적인 구조와 실행 방식이 시장의 다른 도구들과 어떻게 근본적으로 다른지, 다음 장에서 명확히 비교 분석해 보겠습니다.

---

6. 경쟁 환경 분석: 왜 내부 운영 도구(Retool)로는 부족한가?

BeSir는 종종 Retool이나 Appsmith와 같은 내부 운영 도구 빌더(Internal Tool Builder)와 비교되곤 합니다. 세 제품 모두 기존 시스템의 데이터를 활용해 새로운 업무 인터페이스를 만든다는 점에서 유사해 보이기 때문입니다. 하지만 이는 문제 정의의 출발점과 지향하는 목표가 근본적으로 다릅니다. 이 섹션의 목적은 제품의 우열을 가리는 것이 아니라, 각 도구가 해결하려는 문제의 '레이어'가 어떻게 다른지를 명확히 설명하는 데 있습니다.

핵심 차이점 비교 분석

Retool/Appsmith와 BeSir의 근본적인 차이점은 다음 5가지 관점에서 명확하게 드러납니다.

구분	Retool / Appsmith	BeSir
주 사용자	**사람 (Human)**	**AI Agent**
설계 철학	사람이 쓰기 편한 UI 생산성 향상	AI가 실행하기 위한 구조 설계
시스템 이해 방식	암묵적 (개발자의 머릿속에 존재)	명시적 (Ontology로 구조화된 자산)

실행 모델	UI 이벤트 기반 (버튼 클릭 → API 호출)	T-PTC + Agent Server 기반 (AI 판단 → 통제 → 실행)
전사 AI 전략 관점	보조 기능 (Add-on)	**전략의 중심 (Core Engine)**

Retool의 핵심 가치는 **사람**이 사용하기 불편한 내부 시스템의 UI를 빠르고 효율적으로 만드는 데 있습니다. 시스템의 의미 구조는 개발자의 머릿속에 암묵적으로 존재할 뿐, 구조화된 자산으로 남지 않습니다.

반면, BeSir의 핵심 사용자는 **AI Agent**입니다. BeSir의 설계 철학은 AI가 시스템을 '이해'하고 '실행'할 수 있는 구조를 만드는 데 집중합니다. Ontology를 통해 시스템의 의미를 명시적으로 자산화하고, T-PTC와 Agent Server를 통해 AI의 모든 실행을 통제하고 감사합니다. 이는 전사 AI 전략의 '보조 기능'이 아닌, '핵심 실행 엔진'으로서의 역할을 전제로 합니다.

레이어의 차이 설명

위 분석을 바탕으로, BeSir는 Retool/Appsmith의 대체재가 아님을 명확히 할 수 있습니다. 두 솔루션은 서로 다른 레이어의 문제를 해결합니다.

- **Retool / Appsmith:** `Human UI Layer` (사람을 위한 사용자 인터페이스 계층)
- **BeSir:** `AI Execution Layer` (AI를 위한 실행 계층)

Retool이 사람이 더 편하게 일할 수 있는 '화면'을 만든다면, BeSir는 AI가 안전하게 일할 수 있는 '구조'를 만듭니다. 이 두 레이어는 경쟁 관계가 아니라, 각자의 역할에 따라 공존할 수 있는 구조입니다. 그러나 전사 AI 전략 관점에서 AI가 시스템의 실행 주체가 되어야 한다는 목표를 가졌다면, Human UI Layer만으로는 근본적인 한계에 부딪힐 수밖에 없습니다.

이러한 구조적 차이가 실제 기업 환경에서 어떤 구체적인 성공 사례로 이어졌는지, 실제 PoC 결과를 통해 다음 장에서 증명해 보이겠습니다.

---

7. 성공적인 도입 사례: 이론을 현실로 증명하다

본 백서에서 제시된 BeSir의 구조와 철학은 단순한 이론이 아닙니다. 이미 국내 유수의 대기업들이 가진 복잡하고 거대한 실제 운영 환경에서 그 효과와 가능성을 성공적으로 검증했습니다. 서로 다른 산업과 시스템을 가진 두 기업의 PoC 사례는 BeSir의 AI Execution Layer 구조가 가진 재사용성과 강력한 효과성을 명확히 입증합니다.

사례 1: POSCO DX - PMS 기반 AI 경영체계 전환 PoC 분석

- **문제 정의:** 포스코DX의 PMS(Project Management System)는 약 475개의 테이블과 복잡한 데이터 구조를 가진 초대형 핵심 시스템입니다. 기존 AI/BI 접근 방식으로는

질의 범위가 조금만 넓어져도 LLM의 토큰 한계를 초과하거나, Text-to-SQL 방식의 보안 및 안정성 리스크로 인해 실제 경영 분석에 활용하기 어려웠습니다.

- **BeSir 적용 방식:** 기존 PMS 시스템은 전혀 변경하지 않은 상태에서, DDL(데이터 정의 언어)을 기반으로 Ontology를 자동 생성하고, 실제 경영진의 주요 질의(CoQ)를 반영하여 의미 구조를 정교화했습니다. 특히 T-PTC 구조를 적용하여 LLM이 원본 데이터를 직접 수신하지 않고 요약된 결과만 받도록 설계함으로써 토큰 한계와 보안 문제를 구조적으로 해결했습니다.
- **핵심 성과:**
  - **대규모 분석 시 입력 토큰 98~99% 절감** 및 안정적인 대규모 질의 처리
  - 주요 경영 분석 **업무 소요시간 70~90% 단축**
  - API 개발 및 변경 공수의 60~70% 자동화
  - **연간 약 4.5억~6억 원 규모의 실질적인 비용 절감 효과** 검증
- **전략적 의미:** 이 PoC는 단순히 '데이터 분석'을 넘어 **'AI 기반 경영체계'로의 전환**이라는 더 큰 비전을 현실화했습니다. 기존 시스템을 변경하거나 보안 정책을 훼손하지 않고도, AI가 복잡한 경영 환경을 이해하고 실행할 수 있는 구조적 준비가 완료되었음을 증명했다는 점에서 큰 전략적 의미를 가집니다.

사례 2: KMG - AI 기반 판매·재고 운영 체계 PoC 분석

- **문제 정의:** 다수의 매장을 운영하는 리테일 기업 KMG는 실시간 재고 파악이 어렵고, 대부분의 분석을 엑셀 수작업에 의존하며, 재고 발주를 담당자의 경험과 감에 의존하는 전형적인 데이터 운영의 한계를 겪고 있었습니다.
- **BeSir 적용 방식:** DDL조차 없는 환경에서 판매 트랜잭션 데이터를 역설계하여 Ontology를 구성하고, 매장 실무 기준의 주요 질의(CoQ)를 반영하여 현업 주도로 AI 실행 구조를 구축했습니다. 개발자의 개입을 최소화하고 현업 담당자가 직접 자연어로 원하는 분석을 수행할 수 있는 환경을 만드는 데 집중했습니다.
- **핵심 성과:**
  - **실시간 판매/재고 통합 분석** 체계 확보
  - 자연어 질의만으로 **5분 내에 동적 대시보드 생성**
  - 수작업 집계 및 보고 **업무 90% 이상 감소**
  - 매장 매니저 단위의 데이터 기반 의사결정 문화 정착 가능성 확인
- **전략적 의미:** 이 PoC는 **중앙 집중식 보고 체계에서 벗어나 현장 중심의 민주화된 의사결정 체계**로의 전환 가능성을 보여주었습니다. 개발자에 대한 의존 없이 현업 주도형 데이

터 활용이 가능함을 증명했으며, 이는 BeSir의 구조가 대기업뿐 아니라 중견기업까지 동일한 원리로 확장될 수 있음을 보여주었다는 점에서 중요한 의미를 갖습니다.

사례 종합 결론

두 사례는 산업, 시스템 규모, 데이터 환경이 전혀 달랐지만, 공통적으로 다음과 같은 사실을 증명했습니다. "문제는 AI 모델의 성능이나 데이터의 양이 아니라 **구조의 문제**이며, BeSir의 AI Execution Layer(Ontology + CapGraph + MCP) 구조는 **재사용 가능하고 PoC를 실제 운영으로 연결할 수 있다**."

이 검증된 성공 사례를 바탕으로, 기업이 실제로 BeSir를 어떻게 단계적으로 도입하고 전사 표준으로 확산시킬 수 있는지 구체적인 로드맵을 다음 장에서 제시하겠습니다.

---

8. 단계적 확산 전략: PoC에서 전사 표준으로

성공적인 전사 AI 전략은 '처음부터 완벽한 전사 도입'을 목표로 하지 않습니다. 오히려 '실패하지 않는 확산 경로'를 신중하게 설계하는 것에서 시작됩니다. BeSir는 기존 시스템 유지, 보안 정책 준수, 단계적 검증 등 대기업의 현실적인 제약과 요구사항을 전제로, 리스크를 최소화하고 성공 확률을 극대화하는 4단계 확산 시나리오를 제안합니다.

4단계 도입 시나리오

각 단계는 명확한 목적과 성공 기준을 가지며, 이전 단계의 성공이 다음 단계로 나아가는 기반이 됩니다.

- **Phase 1: Foundation (기반 구축)**
  - **목적:** PoC를 위한 기술 실험이 아닌, 전사 AI 실행을 위한 공통 '구조'를 확보하고 검증하는 단계입니다.
  - **주요 활동:** 핵심 시스템 1~2개를 선정하여 Ontology를 자동 생성 및 검토하고, 기본 실행 범위(CapGraph)를 정의하며, 통제를 위한 Agent Server를 배포합니다.
  - **성공 기준: "AI가 우리 시스템을 안전하게 이해하고 조회할 수 있는가?"**
- **Phase 2: Pilot Agent (파일럿 실행)**
  - **목적:** 구축된 기반 위에서 실제 업무 시나리오를 해결하는 AI Agent를 구현하고, PoC를 넘어 실제 운영 환경에서의 가능성을 검증합니다.
  - **주요 활동:** 경영 분석, 재고 분석 등 1~2개의 핵심 업무를 선정하여 BeSir Studio로 Agent를 설계하고, 현업 사용자가 BeSir Browser에서 직접 테스트하며 피드백을 수집합니다.

    - **성공 기준: "AI가 실제 업무 질문에 답하고, 데이터 변경을 안전하게 제안할 수 있는가?"**
- **Phase 3: Department Expansion (부서 확산)**
    - **목적:** 파일럿 성공 이후, 부서별로 난립하는 개별 PoC를 방지하고, Phase 1에서 구축한 구조 자산의 '재사용' 효과를 본격적으로 검증합니다.
    - **주요 활동:** 기존에 구축된 공통 Ontology를 기반으로, 다른 부서의 특화된 요구사항(CapGraph)만 추가하여 새로운 Agent를 신속하게 확장합니다.
    - **성공 기준: "새로운 Agent가 처음부터 개발하는 것이 아니라 '빠르게' 만들어지는가?"**
- **Phase 4: Enterprise Standardization (전사 표준화)**
    - **목적:** AI를 일회성 프로젝트가 아닌, 조직의 지속 가능한 '운영 인프라'로 완전히 전환합니다.
    - **주요 활동:** Ontology와 CapGraph를 전사 표준 자산으로 관리하고, AI 실행 정책과 감사 기준을 고도화하며, AI CoE(Center of Excellence)를 통해 Agent 포트폴리오를 체계적으로 관리합니다.
    - **성공 기준: "AI가 특정 부서의 도구가 아닌, 조직의 기본 실행 옵션이 되었는가?"**

도입 전후 변화 요약

BeSir의 단계적 도입을 통해 기업의 AI 활용 방식은 다음과 같이 근본적으로 변화합니다.

항목	도입 전	도입 후
AI 활용	부서별 PoC 난립	전사 표준 실행
시스템 연계	개별 포인트-투-포인트 구현	Execution Layer 기반 표준 연계
개발 비용	반복 증가	구조 재사용으로 감소
운영 리스크	높음 (통제 부재, 블랙박스)	통제 가능 (중앙 거버넌스)
AI 역할	보조 도구 (Add-on)	**핵심 실행 주체 (Core Engine)**

지금까지 논의된 모든 내용을 종합하여, BeSir가 제안하는 전사 AI의 미래와 핵심 메시지를 다음 결론에서 전달하고자 합니다.

---

9. 결론: AI 실행의 시작

본 백서는 "왜 엔터프라이즈 AI는 PoC에서 멈추는가?"라는 질문에서 출발했습니다. 우리는 그 원인이 AI 모델의 성능이나 데이터의 양이 아닌, AI가 직접 이해하고 사용할 수 없는 '사람 중심의 시스템 구조'에 있음을 확인했습니다. AI를 기존 시스템 위에 단순 기능으로 '붙이는' 방식으로는 전사 AI 전략은 결코 완성될 수 없습니다.

이러한 구조적 문제를 해결하기 위한 유일한 해답은 바로 **'AI Execution Layer'**를 구축하는 것입니다. BeSir는 이 새로운 패러다임을 표준화된 플랫폼 형태로 제공하는 솔루션입니다. BeSir는 Ontology와 CapGraph를 통해 AI에게 시스템을 '이해'하고 '실행'할 수 있는 구조를 제공하고, Agent Server를 통해 모든 실행을 '통제'합니다. 이를 통해 BeSir는 AI를 단발성 `프로젝트`에서 지속 가능한 `인프라`로, 불확실한 `실험`에서 신뢰할 수 있는 `운영`으로, 그리고 일회성 `이벤트`에서 조직의 `지속 가능한 구조`로 전환시키는 핵심적인 역할을 수행합니다.

전사 AI 전략을 위한 최종 체크리스트

현재 전사 AI 전략을 수립 중이거나 재정비하고 있는 조직이라면, 다음 질문들을 스스로에게 던져보아야 합니다.

- AI가 우리 시스템의 데이터 구조가 아닌 '업무적 의미'를 이해하고 있는가?
- AI의 모든 실행 과정은 예외 없이 추적되고 감사될 수 있는가?
- AI는 조직의 승인과 권한 체계 안에서만 통제되어 움직이는가?
- 새로운 AI 요구사항이 생길 때마다 처음부터 다시 개발해야 하는가?
- AI 도입이 늘어날수록 관리 불가능한 리스크가 커지는가, 아니면 재사용 가능한 조직의 자산이 되는가?

이 질문들에 명확하게 '그렇다'고 답할 수 없다면, 현재의 전략은 다시 PoC의 늪으로 돌아갈 가능성이 높습니다.

최종 메시지

전사 AI 전략의 진정한 성공은 "AI를 도입했는가"라는 사실 자체로 평가되어서는 안 됩니다. 성공의 유일한 척도는 **"AI가 실제로 조직의 일을 하고 있는가"**입니다. AI가 실제 시스템에 접근하여 데이터를 분석하고, 의미 있는 제안을 하며, 승인 하에 업무를 실행할 때 비로소 전략은 완성됩니다.

BeSir는 AI 도입의 복잡한 여정의 끝이 아닙니다. BeSir는 진정한 의미의 **'AI 실행의 시작'**을 가능하게 하는 첫 번째 열쇠입니다.