(주) 바벨브레인

언론사로부터 도서관까지,
AI가 업무를 실행하는 시스템

UnderDocs
지식 탐색 워크플로우

LibFlow
도서관 수서로부터 운영까지

PageOne
신문 편집로부터 인쇄까지

투자자 소개 자료 | 2025

우리가 처음 마주한 실제 현장

세 영역은 서로 다르지만, 동일한 구조적 문제를 안고 있다

도서관

수서 · 선정

결정 근거가 시스템에 남지 않아 담당자 교체 시 동일한 실수가 반복

#근거 유실

언론사

지면 편집

배치 기준이 개인 경험에 의존하고, 판단의 이유가 기록되지 않음

#암묵 판단

연구

문서 분석

비교·결론 과정이 재사용 불가, 매번 처음부터 근거를 재추적해야 함

#재사용 단절

공통 문제

**조직의 핵심 판단이 시스템 밖에서 이루어지고,
그 판단이 재사용 가능한 형태로 남지 않는다**

이 문제의 구조적 위험

단순한 비효율이 아니라, 조직 운영 리스크로 직결된다

01

추적 불가능한 판단

판단 근거가 기록되지 않으면 문제 발생 시 책임 추적이 불가능하고, 승인 절차와 인력만 늘어난다

#책임 공백

02

경험 의존의 함정

숙련자 개인에 의존한 업무는 인력 교체 시 품질이 급격히 흔들리며 훈련 비용이 반복 발생한다

#품질 단절

03

선형적 비용 증가

업무량 증가에 자동화가 아닌 인력 증원만이 유일한 대응이 되면서 비용 구조가 선형으로 커진다

#확장 한계

핵심 진단

조직이 커질수록 비용과 리스크가 동시에 증가하는 악순환

특히 공공·언론과 같이 오류 비용이 큰 기관에서 이 악순환은 더 강하게 작동한다

기존 시스템이 실패한 이유

기능이 부족해서가 아니라, 판단과 실행 사이에 구조가 없기 때문이다

RAG Q&A

설명은 가능하지만 실행은 불가

문서에서 답을 찾아 설명

조직 업무를 실제로 수행하지 못하고 결국 사람이 실행해야 한다

#지식 한정

ERP / LMS

절차는 관리하지만 판단은 부재

CRUD와 절차를 체계적으로 관리

왜 그렇게 결정해야 하는지에 해당하는 판단 로직을 담지 못한다

#판단 공백

자동화 스크립트

반복은 빠르지만 예외에 취약

반복 작업을 빠르게 처리

예외 상황과 정책 변화에 취약하고 신뢰 가능한 통제가 어렵다

#통제 한계

공통 실패 원인

설명(지식)과 실행(업무) 사이를 안전하게 연결하는 구조가 없다

AI 도입이 조언자 수준에 머무르거나, 자동화가 통제 불가능한 위험으로 인식되는 이유다

바벨브레인의 질문 전환

문제를 AI 모델 성능이 아니라, 업무 모델링으로 다시 정의하다

기존 접근

"더 좋은 답변을 생성할 수 있는가?"

AI 모델 성능에 집중 → 현장 적용 실패

vs

바벨브레인

"현장의 일을 어떤 단위로 쪼개야 시스템 안에서 재현 가능한가?"

업무 모델링 우선 → 필요한 형태로 LLM 연결

예시 — 수서 업무의 분해

검색 → 복본 확인 → 정책 검토 → 예산 검토 → 등록 → 선정 처리

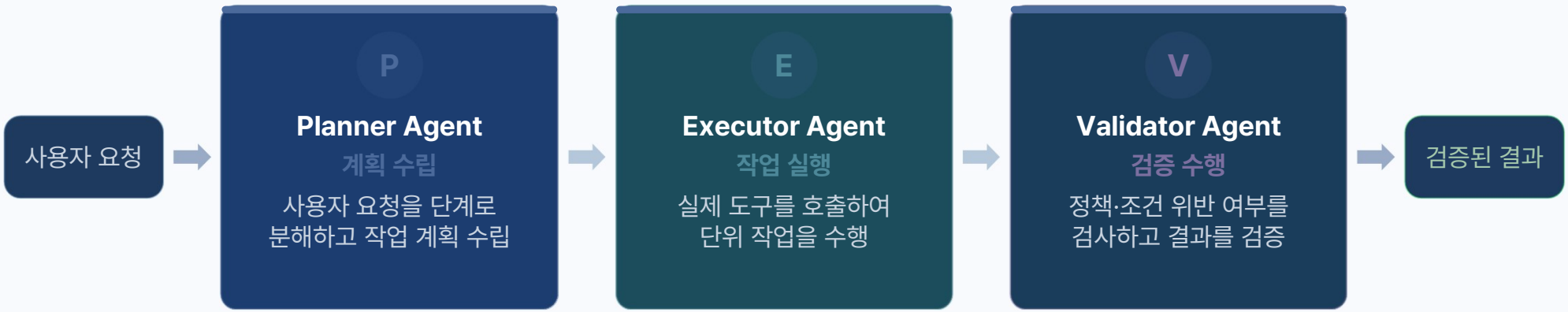
설계 원칙

**업무를 분해하고, 각 단계의 입력/출력과 책임을 고정한 뒤,
이후에 필요한 형태로 LLM을 연결한다**

에이전트 구조의 실제 정의

자율적으로 결정하는 인공지능이 아니라, **책임을 가진 실행 단위**

단일 책임 | 입출력 명확 | 실패 추적 가능



블랙박스가 아니라,

각 단계가 투명하게 드러나고 통제되는 실행 체계

조직 환경에서 AI를 신뢰할 수 있는 이유

UnderDocs: 구현된 구조

연구자의 판단 과정을 6단계로 고정한 지식 탐색 파이프라인

01 Query 분석
질문을 검색 가능한 형태로 분해

02 멀티 문서 검색
여러 소스에서 관련 내용 수집

03 교차 비교
서로 다른 문서의 주장과 근거를 비교

04 근거 검증
충돌 및 약한 근거를 식별하고 필터링

05 결과 합성
검증된 정보를 하나의 결과로 정리

06 시각 자료 포함
관련 이미지, 도표 등 자동 첨부

운영 인프라

실시간 스트리밍
SSE/WebSocket 기반으로 장시간 탐색 작업의 중간 결과를 실시간 전달

SSE · WebSocket · Streaming

멀티 LLM 전환
Gemini, OpenAI, Ollama 등을 유연하게 전환하여 비용, 보안, 성능 요구에 대응

Gemini · OpenAI · Ollama

운영 방식이 바뀌어도 **시스템 구조 자체는 유지된다**

UnderDocs 사용 흐름

연구 현장에 맞춤 4단계 파이프라인 — 업로드부터 산출물까지

01

업로드

- 문서 업로드 및 컬렉션 관리
- PDF, 논문, 보고서 포맷 지원
- 문서 그룹핑으로 주제별 정리

#문서 수집

02

분석

- 핵심 주장과 근거 자동 추출
- 문서별 구조화된 요약 생성
- 근거 강도 평가 및 분류

#자동 추출

03

통합 비교

- 다중 문서 교차 비교 분석
- 상충하는 주장 및 결과 식별
- 근거 약한 부분 표시·배제

#교차 검증

04

산출물 생성

- Q&A 형태 질의응답 지원
- 리포트·슬라이드 자동 생성
- 구조화된 결론 및 시각자료

#결과 산출

SYSTEM 문서 읽기 → 비교 → 정리 → 산출물 제작

HUMAN 판단과 최종 결론 검토에 집중

설계 결과

사람은 판단과 검토에 집중하고, 반복 보조 업무는 시스템이 맡는다

LibFlow: 수서 업무의 실제 분해

담당자의 사고 순서를 에이전트 실행 단계로 고정한 운영 시스템



지원 인프라

RAG 기반 정책 해석
조직 규정과 매뉴얼을 지식베이스로 연결

실행 로그 및 감사 추적
모든 실행 결과를 로그로 기록하여 책임 추적

설계 원칙 | **담당자의 사고 순서를 고정하고, 정책과 예산을 구조적으로 검증한다**

LibFlow 실행 예시

한 줄 명령이 곧 실행 가능한 업무 요청이 된다

U 사용자 요청

"인공지능 도서 5권을 검색해서 복본 없는 것만 긴급 신청하고 선정까지 처리해줘"

STEP 1 검색
AI 도서 5권 탐색
Search API

STEP 2 복본 확인
소장 여부 교차 확인
OPAC DB

STEP 3 정책/예산
규정 및 잔여 예산 검증
RAG + Budget

STEP 4 신청 등록
통과 도서 자동 등록
ERP System

STEP 5 선정 처리
선정 결정 및 로그 기록
Audit Log

실행 로그 미리보기

```
09:31:02  [STEP 1]  검색 실행 — 키워드: 인공지능 / 결과: 5권 탐색 완료  ✔ PASS
09:31:04  [STEP 2]  복본 확인 — 2권 소장 확인, 3권 미소장 확인  ✔ 3권 통과
09:31:05  [STEP 3]  정책 검토 — 학과 우선순위 충족, 잔여 예산 850,000원 확인  ✔ PASS
09:31:06  [STEP 4]  신청 등록 — 3권 ERP 시스템 등록 완료 (긴급)  ✔ DONE
09:31:07  [STEP 5]  선정 처리 — 3권 선정 완료, 감사 로그 기록 완료  ✔ DONE
---
RESULT  총 5권 검색 → 3권 미소장 → 3권 정책/예산 통과 → 3권 긴급 신청 → 3권 선정 완료
```

핵심 원칙 | **자동화가 아닌 통제 가능한 실행 — 결정 과정이 숨겨지지 않는 구조**

PageOne: 편집 시스템의 물리적 제약 구현

신문 편집·조판의 물리적 규칙을 웹에서 재현하는 운영 시스템

6단 × 30행 신문 그리드 구조



기사를 Zone에 드래그앤드롭으로 배치
다단 흐름으로 컬럼 자동 채움

고정 레이아웃 구조

6단×30행 그리드 기반 Zone 시스템
머리·어깨·배꼽기사 등 편집 규칙 적용

다단 조판 엔진

기사 본문의 다단 흐름 자동 처리
이미지 배치·측션 규칙 자동 조정

인쇄 품질 출력

300 DPI · CMYK 변환 · PDF 생성 ~14초
인쇄소 표준 충족 웹 기반 출력

기존 InDesign 등 고가 데스크톱 SW → **PageOne** 웹 기반 · 조판 규칙 내장

AI가 기사를 쓰는 단계가 아니라, 가장 많은 판단이 개입되는 '마지막 출력 단계'를 시스템화

세 시스템이 하나의 스택을 이룬다는 증거

공통 설계 원칙 기반의 반복 가능한 운영 모델

Planner 자연어 → 단계 분해 → **Executor** 도구 호출 → 실행 → **Validator** 정책·규정 검증

	Planner	Executor	Validator
UnderDocs	**질문 분해** 사용자 질문을 탐색 가능한 하위 질문으로 변환	**문서 탐색·추출** 벡터 검색, 청크 추출, 근거 문단 식별	**근거 검증** 출처 추적성 확인, 할루시네이션 필터링
LibFlow	**업무 분해** 자연어 명령을 실행 가능한 단계로 분리	**도구 호출·실행** 검색 API, 복본 확인, 신청 등록 등 순차 처리	**정책 준수 검증** 예산 한도, 선정 기준, 기관 규정 자동 확인
PageOne	**편집 계획** 기사·이미지 배치를 Zone 기반으로 구성	**조판·배치 실행** 다단 흐름, 측션 배치, 이미지 크롭 처리	**물리적 제약 검증** 그리드 초과, DPI 기준, CMYK 변환 규칙 확인

핵심 패턴 세 시스템 모두 Planner → Executor → Validator 구조를 공유하며, 도메인만 다르고 운영 모델은 동일하다

단일 제품의 기능이 아니라, 서로 다른 기관 업무에서 재사용 가능한 운영 모델의 증거

왜 '언론사·도서관'인가

가장 먼저 AI 실행·검증 구조가 효과를 내는 환경

AI 적용을 논할 때
가장 먼저 떠오르는 산업은 아니다.

그러나 바벨브레인이
이 영역을 선택한 이유는
구조적으로 명확하다.

두 산업 모두 자동화 실패 시
오류 비용이 크기 때문에,
조직은 통제 가능한 방식으로만
AI를 도입하려 한다.

단순 생성 AI가 아닌,
규칙 준수·단계적 실행·검증 가능한
시스템의 가치가 더 크다.

→ 바벨브레인의 에이전트 기반 실행·검증 구조의 최적 환경

1. **대규모 텍스트 자산 보유**
   언론사의 기사 아카이브, 도서관의 장서·문헌 DB
2. **정책·관행·규칙에 의한 강한 제약**
   편집 규정, 수서 정책, 예산 기준, 선정 규칙 등
3. **판단 결과에 대한 높은 책임**
   보도 정확성, 도서 선정의 공공성과 투명성 요구
4. **검증 가능한 실행이 핵심 가치**
   "좋은 답변" 보다 "규칙 준수 + 단계적 실행 + 결과 기록"
5. **통제 가능한 AI 도입 요구**
   오류 비용이 크므로 조직이 수용하는 방식만 가능

바벨브레인의 실행·검증 구조가 가장 먼저 효과를 내는 산업적 조건

시장의 구조적 특성

레거시가 깊고, 운영이 보수적이며, 단기간 구조 변경이 어려운 시장

1

레거시의 깊이

편집 시스템·도서관 정보시스템은
10년 이상 축적된 업무 흐름 위에 구축

기능 추가는 가능하지만,
핵심 구조 변경은
매우 큰 비용을 수반한다.

구조적 관성

2

AI 단순 삽입의 리스크

AI를 기존 시스템에 단순 삽입하면
워크플로우 붕괴 또는
책임 구조 흔들림 발생

도입 자체가 거부될 가능성이 높아
"추가"가 아닌 "확장" 방식이 필요

도입 실패 리스크

3

바벨브레인의 접근

레거시를 이해하고
그 위에 안전하게 확장하는 방식

시스템 내부에서
단계적 실행과 검증이 가능하도록
설계하는 것을 핵심으로 한다.

안전한 확장 전략

이 시장에서는 "AI를 추가하는 방식"이 아니라 **"레거시를 이해하고 안전하게 확장하는 방식"**이 요구된다

바벨브레인은 처음부터 이 전제를 받아들이고, **시스템 내부에서 단계적 실행과 검증이 가능하도록 설계**

진입장벽 (Barrier)

모델 성능이 아닌, 업무 구조 이해와 연동 경험이 만드는 구조적 장벽

BARRIER

범용 AI 스타트업이 "LLM만으로" 빠르게 진입하기 힘든 구조적 장벽

이 영역의 진입장벽은
모델 성능이 아니라,
업무 구조 이해와
연동 경험에 있다.

1

복잡한 업무 규칙과 예외

규칙이 많고 예외가 다양해
단순 자동화로는 운영 불가능

업무 복잡성

2

정책·규정의 시스템 내재화

정책·규정을 시스템 로직으로 내재화 필요
문서만 보고 구현하기 어려운 영역

도메인 지식

3

레거시 시스템 연동 경험

데이터 구조·업무 흐름·예외 처리 경험 없이는
제품이 현장에 들어가기 어렵다

현장 경험

바벨브레인은 **이 장벽 안쪽에서 이미 시스템을 구축하고 운영 중**

서울시스템·퓨쳐누리 협업

실사용 환경에서의 연동·검증을 전제로 한 협업 구조

서울시스템 주식회사
언론사 편집 시스템 개발

연동

바벨브레인
에이전트 기반 실행·검증 시스템

연동

주식회사 퓨쳐누리
도서관 정보시스템 개발·배포

협업의 세 가지 의미

협업 형태

용역 기반 사업 수행
실제 시스템 연동을 전제로
기능 구현 및 검증 진행

용역 수행

학습 효과

현업 데이터 구조와
업무 예외 상황을 학습하고
에이전트·도구 체계 정제

현장 학습

투자자 관점

연구실 데모가 아닌
운영 환경에서 검증 진행 중
→ 실행 단계 진입 신호

검증 진행

"연구실 데모"가 아니라 **"운영 환경에서 검증이 진행 중"**이라는 신호

IR에서 '용역 매출'의 해석

용역은 코드 납품이 아니라, 재사용 가능한 운영 모델을 축적하는 경로다

흔한 오해

"SI 회사 아닌가?" — 인력 투입형, 확장성 없는 매출

바벨브레인의 실제

요구사항 획득 + 제품 현실화 경로

- 기관별 업무 구조·정책·예외 케이스 확보
- 레거시 시스템 연동을 통한 제품 현실화
- **업무 분해 → 에이전트 도구화 → 모델 축적**
- 장기적으로 용역 비중 ↓ 시스템 비중 ↑

용역 → 제품 전환 흐름

STEP 1 용역 수행 — 요구사항 획득
기관별 업무 구조, 정책, 데이터 모델, 예외 케이스 확보
레거시 시스템 연동 경험 축적

STEP 2 일반화 — 에이전트 도구화
업무를 분해하고 제품화 가능한 형태로 일반화
Planner·Executor·Validator 구조에 반영

STEP 3 제품 전환 — 시스템 매출 확대
재사용 가능한 운영 모델로 축적 → 제품 독립
용역 비중 ↓ 시스템·라이선스 비중 ↑

**핵심: 용역을 통해 코드만 납품하는 것이 아니라,
업무를 분해하고 에이전트 도구화하여 재사용 가능한 운영 모델로 축적한다**

용역은 시장 진입을 위한 요구사항 획득 경로이자, 제품을 현실화하는 통로다

기술 유출 우려와 대응

협업 기반 용역에서 발생 가능한 기술 유출 리스크에 대한 구조적 방어 논리

투자자 우려

"협업 과정에서 기술이 유출되는 것 아닌가? 용역 산출물이 곧 핵심 기술 아닌가?"

바벨브레인의 답변

핵심 자산은 '코드'가 아니라 '업무 분해 방식 · 에이전트 실행 · 검증 패턴'에 있습니다

구조적 방어 논리

① 산출물은 기관별 맞춤형

프로젝트 산출물은 기관 요구에 따라 매번 달라지며, 특정 코드가 그대로 재사용되지 않음

② 공통 설계는 내부 코어 분리

작업 단위 분해 · 도구 체계 · 검증 흐름 등 공통 구조는 납품 범위 밖, 내부 자산으로 관리

③ 재현에는 동일한 경험 필수

동일 결과 재현에는 동일한 운영 경험과 구조 설계 능력이 필수

"코드 일부를 본다"와 "시스템을 동일하게 재구축한다"는 완전히 다른 차원입니다

핵심 구조를 지속적으로 내재화 · 재사용하여 경쟁력을 유지합니다

오픈소스 sLLM 기반 전략

특정 벤더에 종속되지 않는 모델 전환 아키텍처와 기관 시장 대응 전략

핵심 설계 원칙

운영 로직은 유지하고, 모델만 교체 가능한 구조

구조적 이점

① 멀티 모델 전환 구조

OpenAI · Gemini · Claude 등 상용 모델과 Ollama 기반 로컬 LLM을 유연하게 전환

② 보안 · 폐쇄망 · 데이터 반출 금지 대응

기관 시장의 반복적 보안 요구에 구조적으로 대응 — 단순 비용 절감이 아닌 시장 적합성 확보

③ 장기 운영 안정성

정책·가격 변화로 특정 모델 사용이 어려워져도 핵심 로직(업무 분해·도구 실행·검증 흐름)은 유지

④ 모델 경쟁 심화 = 구조적 가치 증가

LLM 시장 경쟁이 심해질수록 모델 비종속 아키텍처를 가진 바벨브레인의 전략적 가치 증가

모델은 교체 가능한 부품, 바벨브레인의 실행 구조는 교체 불가능한 자산

벤더 비종속 구조로 기관 보안 요구 대응과 장기 운영 안정성을 동시에 확보합니다

경쟁력의 위치

동일한 설계 패턴이 서로 다른 도메인에서 반복 작동합니다

UnderDocs

지식 탐색 · 근거 검증

연구 문서 기반 질의에 대해 에이전트 파이프라인이 관련 근거를 탐색·검증·종합하여 답변 생성

LibFlow

도서관 업무 자동화

도서관 업무를 단계적 도구 호출과 정책 검증으로 구현하여 사서의 운영 판단을 시스템화

PageOne

신문 편집 운영 시스템

신문 편집 규칙과 출력 표준을 웹에서 재현하는 운영 시스템으로 편집 판단을 구조화

공통 설계 패턴

요청 분해 → 도구 실행 → 결과 검증

도메인은 다르지만 구조는 동일 — 이것이 반복 가능한 경쟁력의 근거

바벨브레인은 단일 앱이 아닌, 기관 업무를 시스템화하는 운영 추상화 레이어를 축적하고 있습니다

투자자가 걱정하는 리스크

리스크를 숨기지 않고 구조적으로 인정하며, 대응 전략을 함께 제시합니다

리스크 요인	구조적 대응
느린 도입 결정 속도 보수적 기관 시장(언론사·도서관)의 의사결정 구조가 느리고, 예산 편성과 조달 절차에 의존	**운영 시스템으로 자리잡기 전략** 도입 후 운영 시스템으로 정착하면, 장기 계약과 낮은 이탈률로 안정적 매출 축적
긴 영업 사이클 기관별 검토·시범운영·본도입까지 수개월 이상 소요되며, 단기 매출 예측이 어려움	**축적형 매출 구조** 한번 도입된 기관은 운영 의존도가 높아 이탈이 어렵고, 연간 계약 갱신으로 매출 누적
레거시 시스템 연동 범위 차이 기관마다 기존 시스템이 달라 연동 범위와 난이도가 상이하며, 초기 확장 속도 제한 가능	**모듈형 연동 설계** 공통 운영 구조 위에 기관별 연동 모듈을 분리 설계하여 확장 시 연동 비용을 최소화

폭발적 성장이 아닌, 운영 시스템으로 축적되는 시장을 전제로 전략을 설계합니다

현재 위치

과장 없이 현재 상태를 확정적으로 정의합니다

보유 시스템

UnderDocs 구현 완료
지식 탐색 · 근거 검증 에이전트 시스템

LibFlow 구현 완료
도서관 업무 자동화 운영 시스템

PageOne 구현 완료
신문 편집 운영 시스템

현장 운영 검증

서울시스템 — 협업 용역 수행 중
실제 운영 환경에서 기능 구현 · 검증 진행

퓨쳐누리 — 협업 용역 수행 중
현장 검증을 통한 제품화 강화

진행 단계

아이디어 단계	개념 검증	★ 운영 검증	시장 확장 (초기)

▲ 현재 위치

바벨브레인을 한 문장으로 정의하면

기관 업무를 실제로 다루는 운영 소프트웨어를 만들고,
현장 검증을 통해 제품화를 강화해가는 회사

아이디어 단계의 AI 스타트업이 아닌, 운영 검증을 통해 제품화를 강화하는 단계

결론

바벨브레인이 만들어온 것, 지금 서 있는 곳, 그리고 앞으로의 방향

검증된 시스템

3개 제품 구현 완료

UnderDocs · LibFlow · PageOne
현장 운영을 통해 실증된
AI 운영 소프트웨어

현장 운영

실제 기관과 협업 수행

서울시스템 · 퓨쳐누리
현장 검증을 통해
제품화를 강화하는 구조

설계 원칙

업무 분해 → 도구 실행 → 검증

요청을 분해하고 책임을 고정한 뒤
LLM을 연결하는
운영 추상화 레이어

바벨브레인을 한 문장으로 정의하면

기관 업무를 실제로 다루는 운영 소프트웨어를 만들고,
현장 검증을 통해 제품화를 강화해가는 회사

아이디어가 아닌
운영 검증

단일 앱이 아닌
운영 추상화 레이어

폭발적 성장이 아닌
축적되는 시장

지금이 바벨브레인과 함께할 가장 좋은 타이밍입니다

운영 시스템으로 축적되는 시장, 그 출발점에 서 있습니다