



Introduce :

EasyTrace-AI™

Java MSA 환경을 위한 통합 APM 솔루션

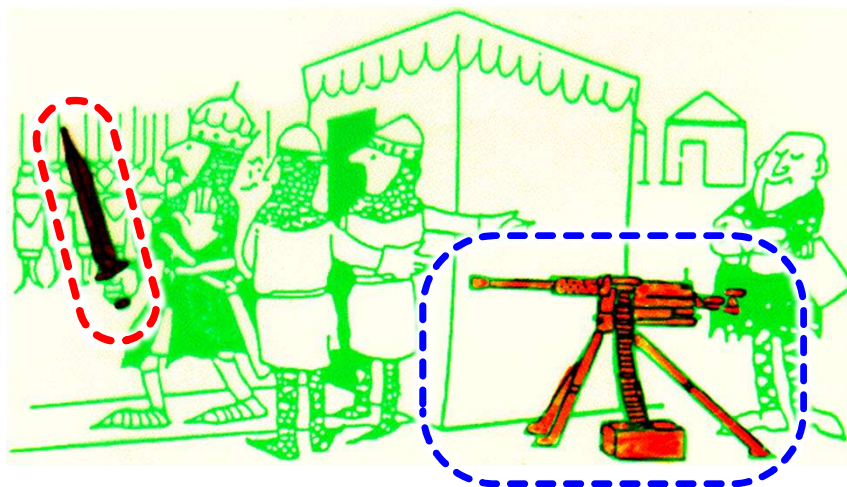
복잡한 Micro Service 환경,
이제 EasyTrace-AI™ 하나로 완벽하게 제어하십시오.
로그 수집부터 AI 기반 코드 수정 제안까지,
개발자를 위한 All-in-One 플랫폼입니다



귀사는 현재 어떤 비행기에 탑승하고 있습니까?



귀사는 어떤 무기를 갖고 전쟁터에 나서겠습니까?





MSA 시대, 개발자들이 마주한 도전

Micro Service 아키텍처(MSA)의 도입은 유연성을 주었지만, 운영과 디버깅의 복잡성이라는 새로운 장벽을 세웠습니다



복잡성 증가

수십, 수백 개의 서비스가 얹혀 있어
오류 발생 시 원인을 추적하기가 미로
찾기보다 어렵습니다



파편화된 데이터

기존 APM 도구는 분산된 로그와
데이터를 통합 관리하는 데 한계가
있어, 전체적인 맥락 파악이
불가능합니다



느린 문제 해결

문제 식별과 해결에 과도한 시간이
소요되며, 이는 비즈니스 민첩성을
저해하고 개발자의 피로도를 높입니다

EasyTrace-AI™가 제시하는 혁신적인 해법

EasyTrace-AI™는 단순한 모니터링 도구가 아닙니다.
Java 기반 MSA 환경에 특화되어, 개발자의 고민을 근본적으로
해결하는 파트너입니다

Java MSA 특화 설계

Java 생태계와 Micro Service 구조를 가장 잘 이해하는 최적화된
아키텍처를 제공합니다

All-in-One 통합 플랫폼

Log 수집, Tracing, Debugging, AI 분석을 단일 플랫폼에서 끊임
없이 제공합니다

운영/테스트 환경 혁신

배포 前 테스트 단계부터 실시간 운영 환경까지, 전체 Life-Cycle에
걸쳐 문제를 식별하고 해결합니다

완전한 어플리케이션 로그 수집 및 Full Text 검색

흩어진 로그를 찾아 헤매는 시간은 이제 끝났습니다. 모든 데이터를 한곳에 모으고 즉시 검색하십시오



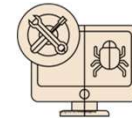
실시간 로그 수집

모든 어플리케이션 로그를 누락 없이
실시간으로 중앙 서버에 수집하여
데이터의 정합성을 보장합니다



통합 로그 및 Full Text 검색

강력한 검색 엔진을 통해 수십억 건의
로그 중 원하는 데이터를 즉시 찾아내어
디버깅 시간을 획기적으로 단축합니다.



연결 흐름 디버깅

복잡한 트랜잭션 흐름과 로그를 연결하여,
MSA 환경에서도 단일 어플리케이션처럼
명확한 디버깅 뷰를 제공합니다.

Log data



【 AI ASSISTANT 】

오류 근원 분석과 해결책 자동 제안

EasyTrace-AI™는 단순히 오류를 알려주는 것을 넘어,
어떻게 고쳐야 할지를 알려줍니다

실시간 오류 탐지

EasyTrace™ 【Agent가】 오류를 실시간 탐지하고, 해당 시점의 모든 로그를 완벽하게 기록합니다

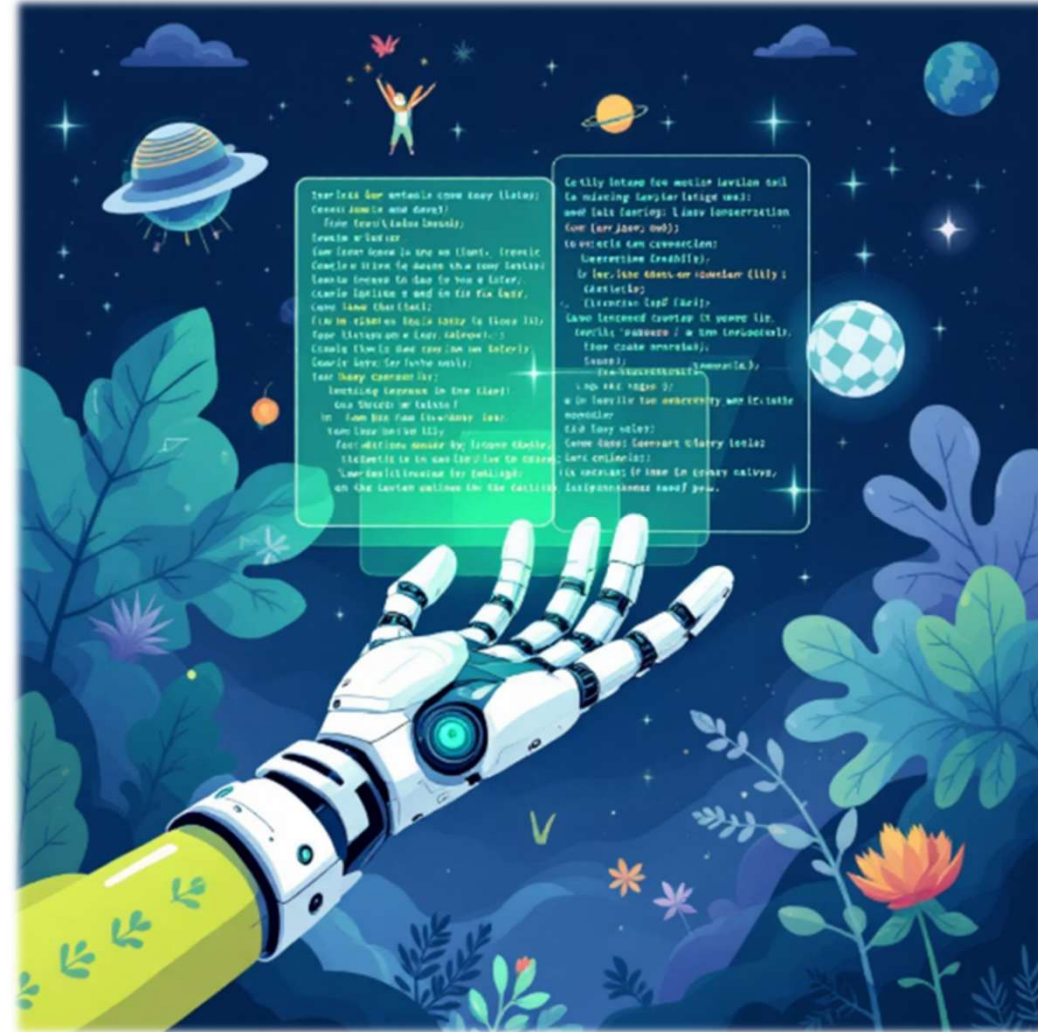
자동 원인 분석

EasyTrace™ 【AI Assistant】가 오류 로그와 Stack Trace 등을 분석하여, 오류가 발생한 근본 원인을 즉시 파악합니다

소스코드 변경 제안

단순 조언이 아닌, 수정이 필요한 구체적인 소스코드 변경案을 AI가 직접 제시합니다

➔ **개발자 부담 최소화** : 단순 반복적인 디버깅 업무를 AI에게 맡기고, 개발자는 비즈니스 로직 구현에 집중할 수 있습니다



【 AI ASSISTANT 】

코드 품질 리뷰 및 시큐어 코딩 가이드

더 안전하고, 더 깔끔한 코드를 작성하도록 돕는 지능형 가이드가 내장되어 있습니다



자동 Code Review

Commit된 코드의 품질 저하 요소를 실시간으로 탐지하고 리뷰하여, 기술적인 이슈가 쌓이는 것을 방지합니다



Secure Coding 제안

잠재적인 보안 취약점을 사전에 진단하고, 안전한 코딩 표준에 맞춘 구체적인 권고안을 제공합니다



Refactoring 제안

복잡도가 높거나 비효율적인 코드 로직을 발견하면, 유지보수성과 성능을 높일 수 있는 리팩토링 방향을 제시합니다

빠른 오류 식별과 해결 ▶ 운영/테스트 환경의 혁신

EasyTrace-AI™는 개발부터 운영까지, 서비스의 안정성을 지키는 든든한 방패입니다.

Java시스템 구축 및 운영 단계 전반에 걸쳐 사용할 수 있는 「어플리케이션 결함 및 장애 해결에 특화된 툴」입니다.

조기 문제 발견

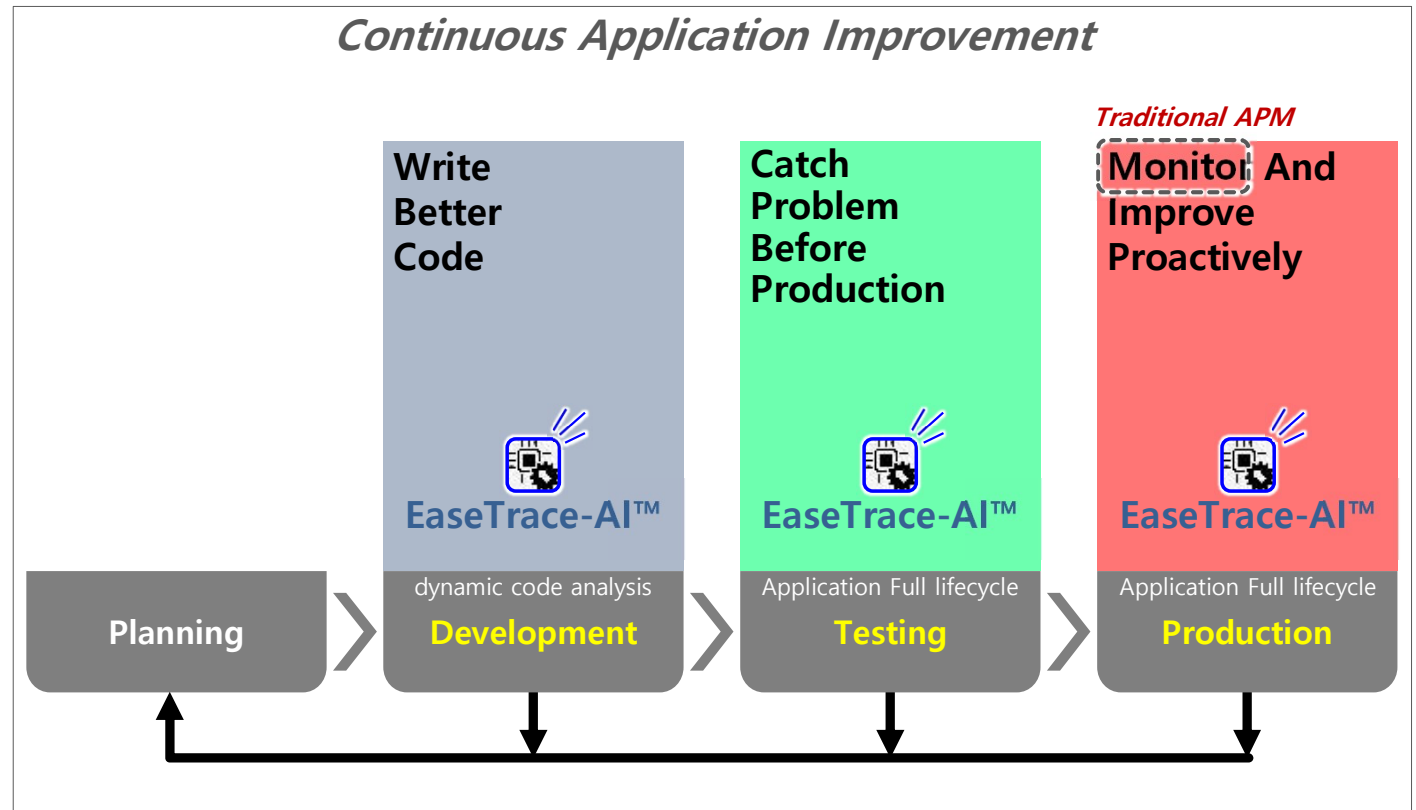
Shift Left : 테스트 단계부터 코드 품질과 보안 문제를 EasyTrace-AI™가 분석하여, 배포 전 잠재적 결함을 제거합니다

운영 중 장애 최소화

실시간 모니터링과 즉각적인 원인 분석으로 장애 발생 시 평균 복구 시간을 극적으로 단축시킵니다

서비스 안정성 극대화

지속적인 품질 관리와 빠른 대응 체계를 통해 최종 사용자에게 중단 없는 서비스를 제공합니다



EasyTrace-AI™ 도입 사례 ▶ Java MSA 기업의 변화

실제 현장에서 증명된 EasyTrace의 압도적인 성능과 효율성을 확인하세요

70%

오류 탐지 시간 단축 (A사)

복잡한 결제 서비스 장애 원인 분석 시간을 획기적으로 줄여 서비스 가용성을 확보했습니다

40%

개발 생산성 향상 (A사)

반복적인 디버깅 업무 감소로 개발팀이 신규 기능 개발에 더 집중할 수 있게 되었습니다

85%

보안 취약점 감소 (B사)

AI 기반 상시 코드 리뷰 시스템 도입 후, 배포 전 보안 결함 발견율이 급격히 개선되었습니다

결과적으로 운영 비용은 절감되고, 안정적인 서비스 제공으로 고객 만족도는 상승했습니다

경쟁 솔루션과의 차이점

왜 EasyTrace-AI™여야 할까요?

우리는 개발자가 필요로 하는 「진짜 기능」에 집중합니다



완벽한 통합 플랫폼

별도의 로깅 서버나 분석 도구
연동 없이, 로깅-트레이싱-AI 분석이
하나의 흐름으로 연결된
유일한 솔루션입니다



Java MSA 최적화

범용 솔루션이 놓치는 Java 런타임의
특성과 MSA 통신 패턴을 정밀하게
분석하여 깊이 있는 인사이트를
제공합니다



Actionable AI

단순히 "에러가 났다"고 말하는 것이
아니라, "이 코드를 이렇게 고치라"고
제안하는 실행 가능한 AI 솔루션입니다

Future with EasyTrace-AI™

개발 생산성과 서비스 안정성, 지금 바로 극대화하세요!

EasyTrace-AI™는 복잡한 MSA 환경에서 개발자가
길을 잃지 않도록 돕는 나침반입니다.

AI와 통합된 혁신적인 APM으로 개발과 운영의
새로운 표준을 경험해보십시오

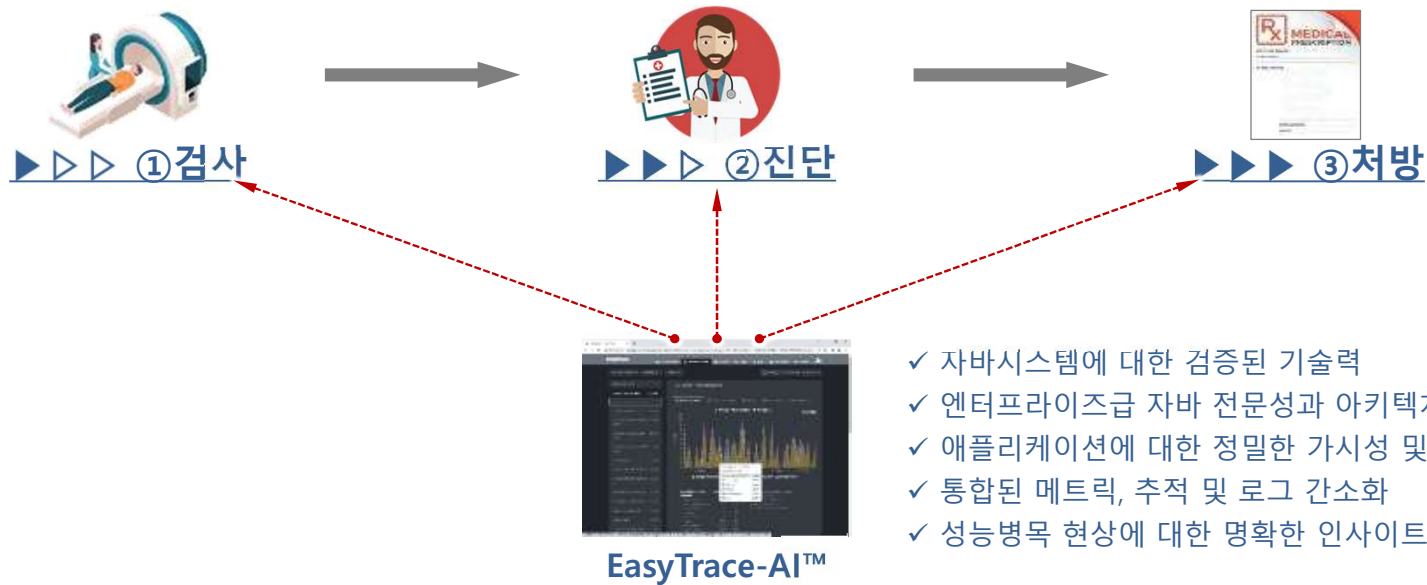
❏ 시작할 준비가 되셨나요?
데모 신청 및 도입 문의를 통해 귀사의 개발 환경을 혁신하세요.



[Back-up]

EasyTrace-AI™ 활용 프로세스

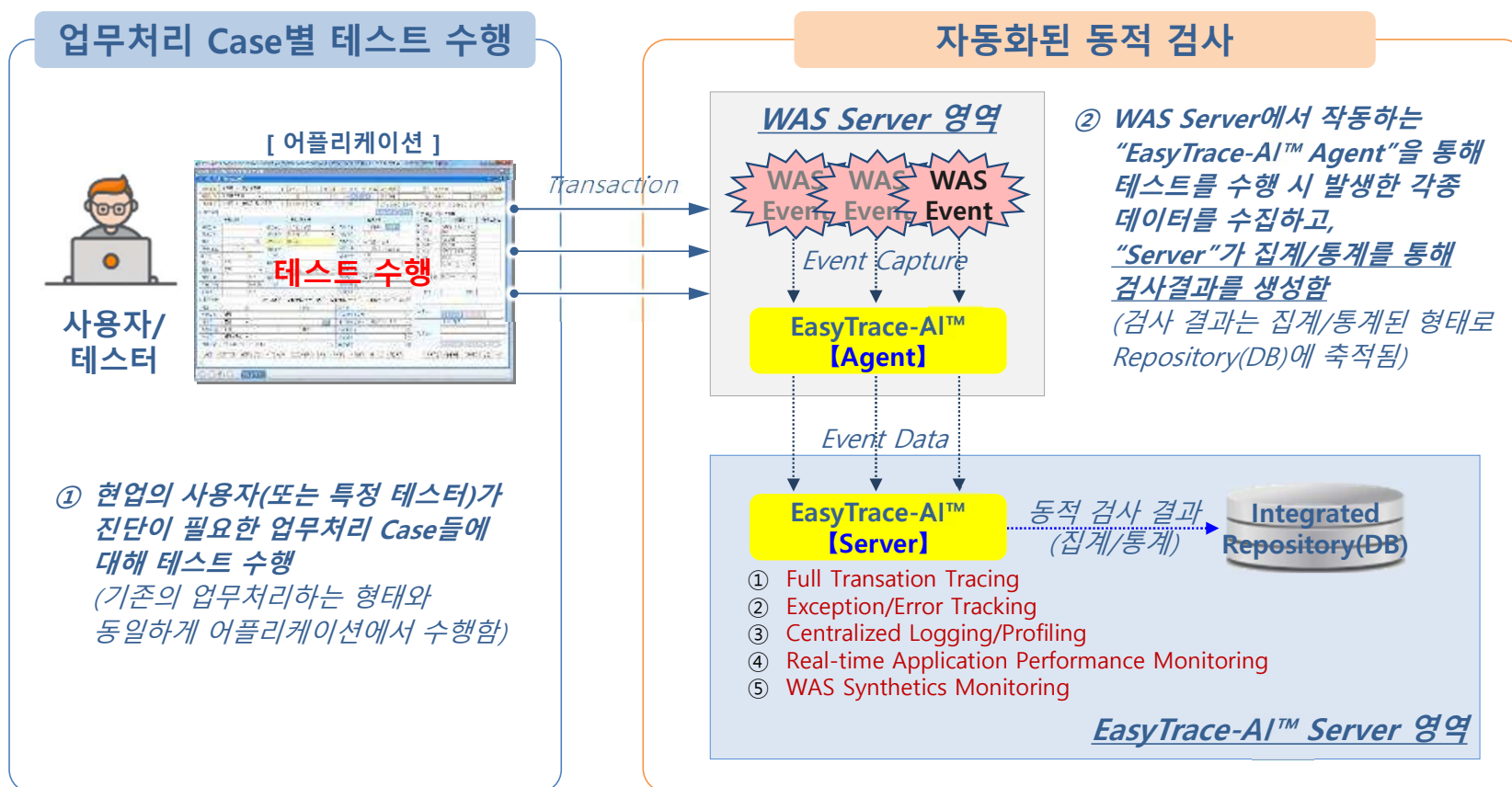
결함에 대해 "검사→진단→처방"의 프로세스로 EasyTrace-AI™를 활용할 수 있습니다



[Back-up]

EasyTrace-AI™ 활용 프로세스 ▶ ① 검사

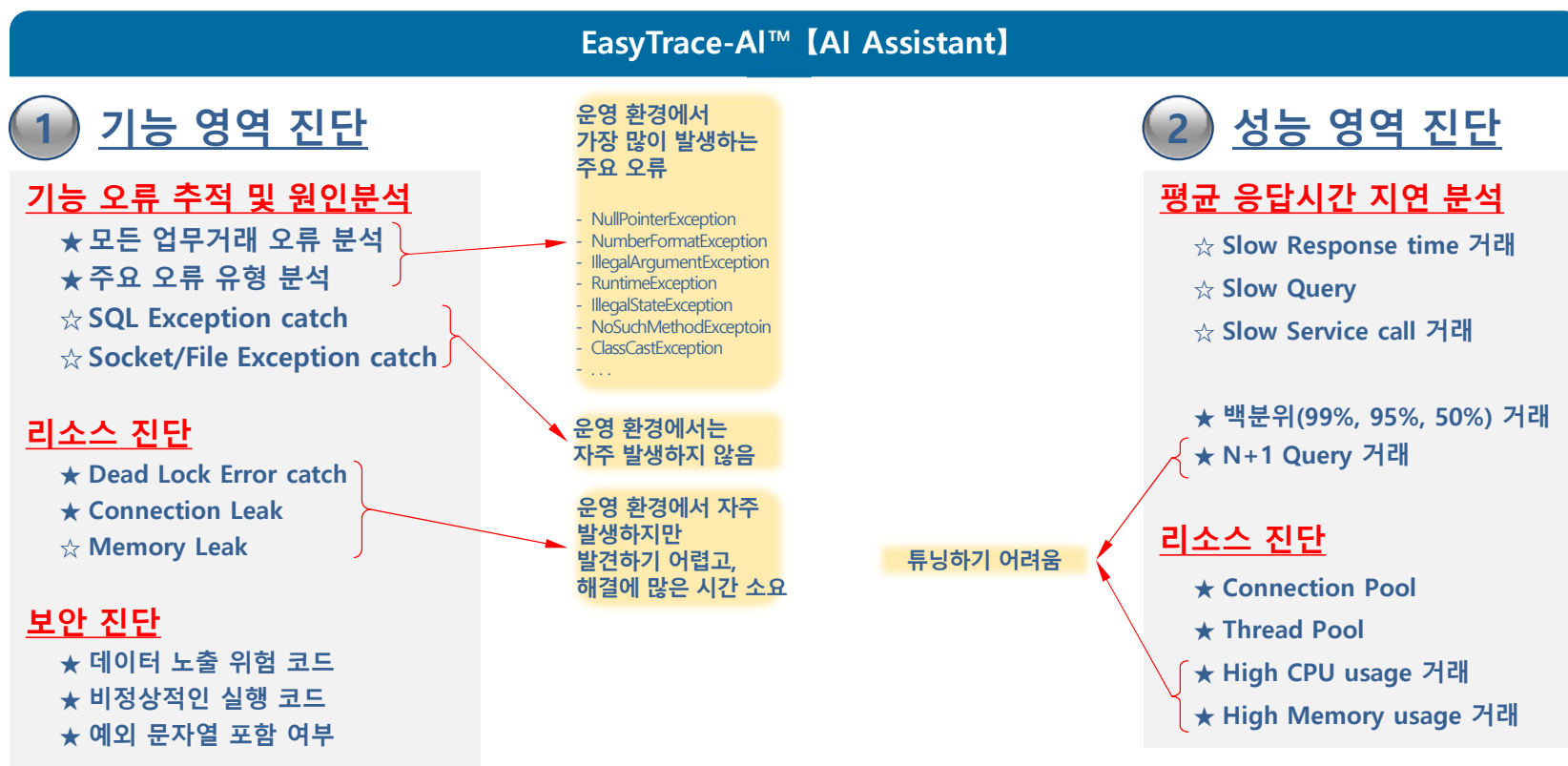
우선 현업 등의 사용자가 품질진단을 위해 선정된 업무처리 Case를 테스트하면 EasyTrace-AI™【Agent】가 실시간으로 데이터를 수집하여 자동 검사를 진행합니다



[Back-up]

EasyTrace-AI™ 활용 프로세스 ▶ ② 진단

탐지/기록된 검사 결과를 참조하여 EasyTrace-AI™의 【AI Assistant】가 설정된 지표에 따라 1차적으로 '기능'과 '성능' 영역에 대해 동적 자동 진단을 수행합니다



☆ : 현재 국내 주요 APM 솔루션들이 제공하는 정보 : APM에서는 해당 정보들만 제공

[Back-up]

EasyTrace-AI™ 활용 프로세스 ▶ ③ 처방

진단에 의해 도출된 장애/이슈/결함 등에 대해 건별(Case-by-Case) 해결 가이드 즉, 개선 또는 코드 수정이 즉시 가능한 수준으로 “처방전”을 제공합니다

AI Assistant

##Source:({org.mybatis.jpetest.service.OrderService}) ##Error Message: Caused by: org.hsqldb.HsqlException: user lacks privilege or object not found: INSERT_ORDER ##Exception Stack: net.sourceforge.jtdm.jdbc.support.CQLExceptionTranslator.translate(CQLExceptionTranslator.java:100)

Source Code

Error Analyzer

주요 수정안

제목: OrderMapper 를 테이블 기반 INSERT 구문으로 교체
우선순위: Critical
설명:
현재 해킹이 저장 프로시저 호출 형태이므로 HSQLDB (기본적으로 프로시저를 지원하지 않음) 에서는 실행 실패합니다.
테이블에 대한 일반 INSERT 문으로 변경하면 권한 문제와 객체 미존재 문제를 한 번에 해결할 수 있습니다.

원본 코드 (가정되는 매퍼 정의)

<!-- OrderMapper.xml (오류가 있는 버전) -->
<mapper namespace="org.mybatis.jpetest.mapper.OrderMapper">

 <!-- 잘못된 CALLABLE 선언 -->
 <insert id="INSERT_ORDER" statementType="CALLABLE">
 {call INSERT_ORDER(#{orderId}, #{userId}, #{orderDate}, #{totalPrice})}
 </insert>

 <!-- 다른 매퍼들 -->
</mapper>

수정된 코드

<!-- OrderMapper.xml (수정된 버전) -->
<mapper namespace="org.mybatis.jpetest.mapper.OrderMapper">

 <!-- 테이블 INSERT 구문으로 교체 -->
 <insert id="insertOrder" parameterType="org.mybatis.jpetest.domain.Order">
 INSERT INTO ORDERS
 (ORDER_ID, USER_ID, ORDER_DATE, TOTAL_PRICE, STATUS_ID)
 VALUES
 (#{orderId}, #{userId}, #{orderDate}, #{totalPrice}, #{statusId})
 </insert>

 <!-- 주문 상태 기록 (테이블 기반) -->
 <insert id="insertOrderStatus" parameterType="org.mybatis.jpetest.domain.Order">
 INSERT INTO ORDER_STATUS
 (ORDER_ID, STATUS_ID, CREATED_DATE, UPDATED_DATE)
 VALUES
 (#{orderId}, #{statusId}, CURRENT_TIMESTAMP, CURRENT_TIMESTAMP)
 </insert>
</mapper>

※ '주요 수정안' 및 '대안 수정안' 까지 제시

대안 수정안

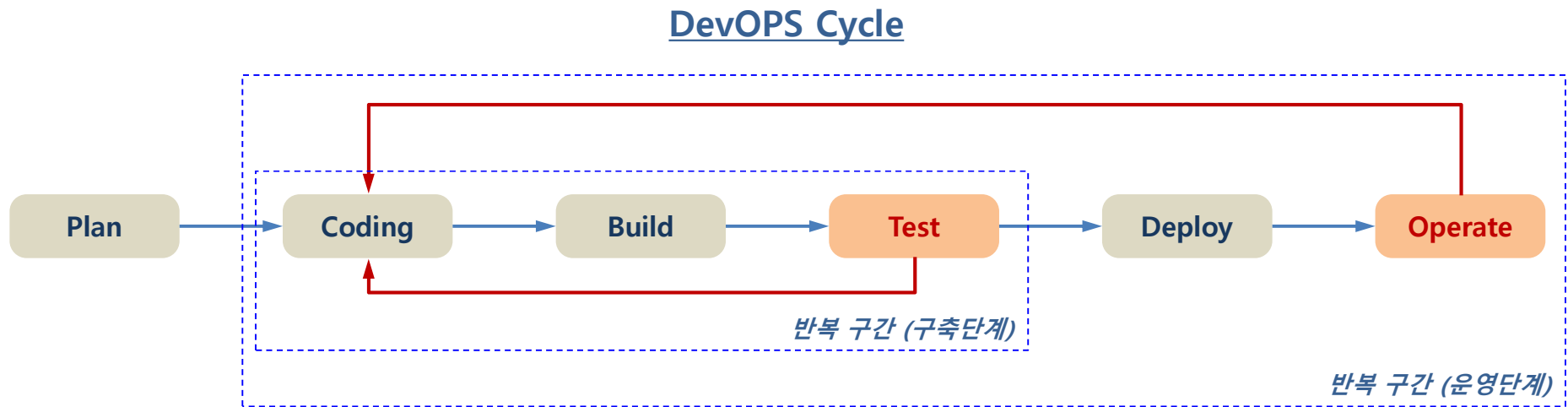
제목: HSQLDB 에 맞는 저장 프로시저 생성 및 권한 부여
우선순위: Medium
설명:
만약 기존 비즈니스 로직이 저장 프로시저를 사용하도록 설계되었다면, 프로시저를 HSQLDB 에 생성하고 현재 DB 사용자에게 INSERT 권한을 부여할 수 있습니다.
그러나 HSQLDB는 경량형이며 프로시저 지원이 제한적이므로, 대부분의 경우 배치 방식(테이블 기반 삽입) 으로 바꾸는 것이 표준적이고 유지보수에 유리합니다.

-- HSQLDB 환경에서 저장 프로시저 생성
CREATE PROCEDURE INSERT_ORDER (
 IN p_order_id INT,
 IN p_user_id VARCHAR(50),
 IN p_order_date DATE,
 IN p_total_price DECIMAL(10,2)
)
BEGIN
 INSERT INTO ORDERS (ORDER_ID, USER_ID, ORDER_DATE, TOTAL_PRICE)
 VALUES (p_order_id, p_user_id, p_order_date, p_total_price);
END;

[Back-up]

DevOPS Cycle과의 결합

EasyTrace-AI™는 DevOPS와 완벽하게 결합되어, Full Life-Cycle을 완성합니다



Test와 Operate 시 발생하는 오류/결함에 대해,

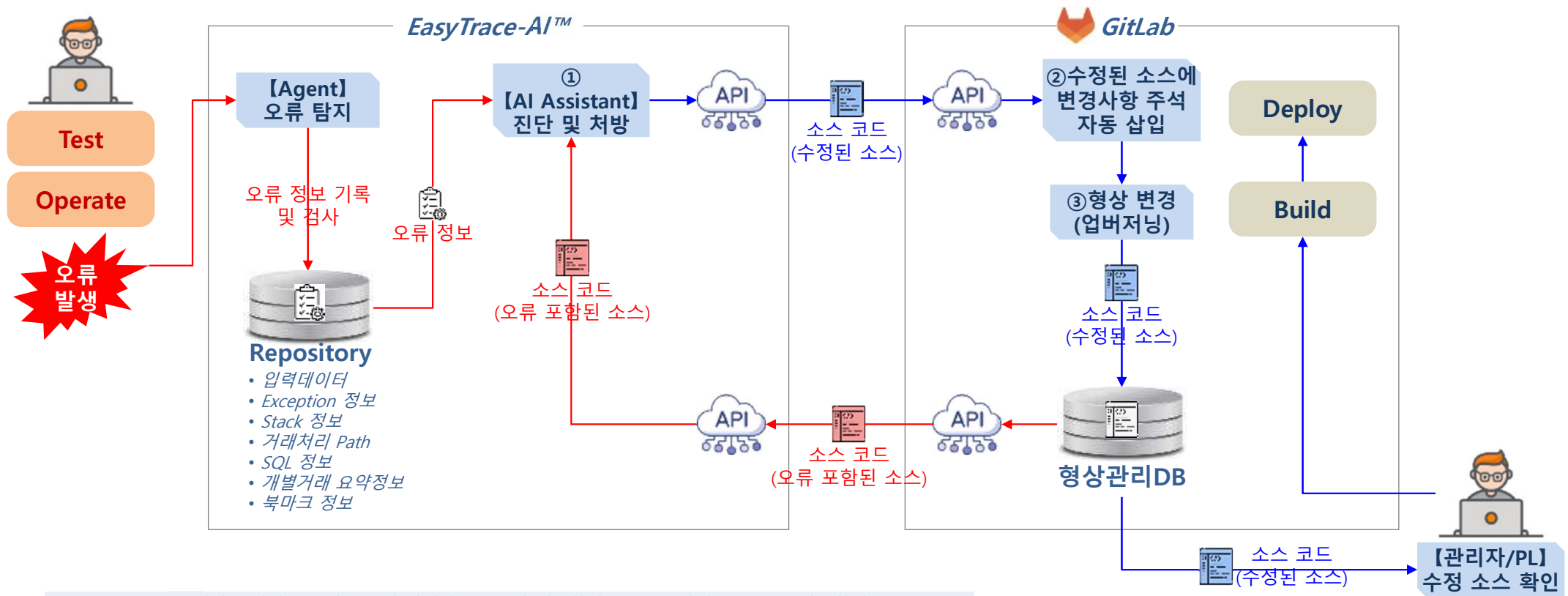
- EasyTrace-AI™에서 검사/진단을 통해 원인을 분석하고, AI Assitant가 제시하는 수정된 소스가
- GitLab을 통해 형상이 변경처리 되고, 관리자/PL 확인 후 Build되는

전체 과정을 자동화할 수 있습니다

[Back-up]

DevOPS Cycle과의 결합

Tester와 관리자 역할만이 Manual로 진행되고, EasyTrace-AI™와 GitLab 영역은 전체 과정이 완전 자동화되어 운영될 수 있습니다



- ① EasyTrace-AI™에서 수집된 오류정보 뿐만 아니라 GitLab과 연동하여 소스코드까지 Input으로 사용하면 EasyTrace-AI™의 AI Assistant가 실 소스를 대상으로 오류 Fix한 코드를 제안해 줌
- ② 수정된 소스코드를 EasyTrace-AI™로부터 받으면, 이전 소스와 비교 분석하여 GitLab이 자동으로 주석을 삽입하고,
- ③ 형상관리 반영(버전 업)하여, 관리자가 Build를 위해 확인하기 직전 단계까지를 EasyTrace-AI™와 연계하여 자동화 함