

AIOps Auto Pilot

AI Agent 기반 AIOps 적용한
RCA & Remediation Automation System

2025 LLM Application 교육과정

AIOps Auto Pilot

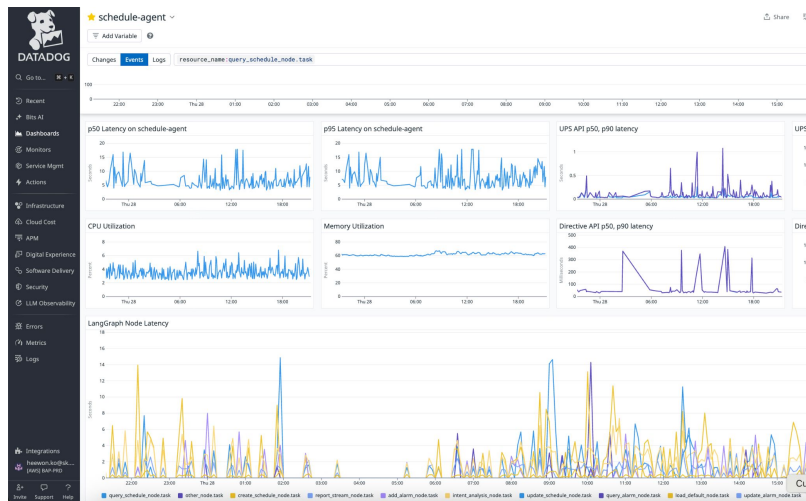
Pain Point : “기존 장애 조치 프로세스의 한계”

1. 장애 발생 시 수동으로 Log, Metric, Trace 등 다양한 플랫폼 (Datadog, Langfuse, AWS Console) 을 오가며 직접 분석
2. 원인 파악 및 대응까지 많은 시간 소요
3. 유사한 장애 발생 시에도 같은 프로세스를 반복하며 분석
4. 개인의 제한된 학습 경험과 직감에 의존하여 해결 방법 모색



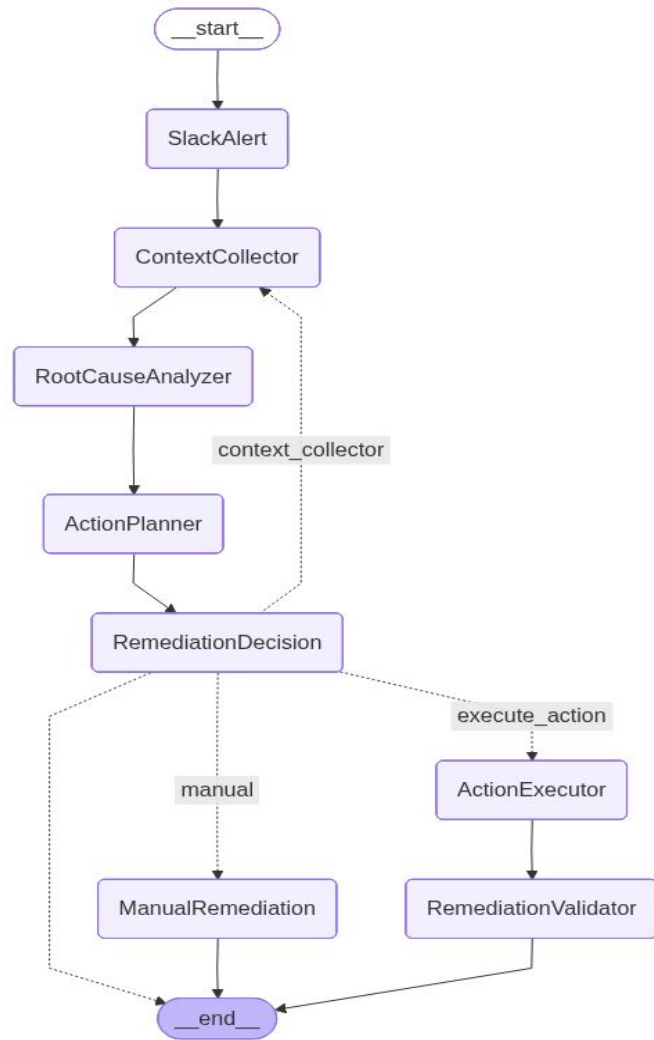
“데이터 통합 + LLM 기반 Root Cause 분석 및 Action Plan + User Feedback Loop”
을 통해 장애 대응 시간 단축

대상 사용자 : Software Engineer, DevOps Engineer, Project Manager




AI Ops Auto Pilot - 그래프 아키텍처

- **SlackAlert**: Slack 알림 발생에 대한 트리거 수신, 장애 발생 정보 (서비스 및 발생시각) 저장
- **ContextCollector**: 다양한 Event Source로부터 모니터링 데이터 수집 (로그, 메트릭, 트레이스)
- **RootCauseAnalyzer**: LLM (GPT 4o) 이용하여 수집한 데이터를 기반으로 에러 발생 원인을 분석
- **ActionPlanner**: LLM (GPT 4o) 이용하여 최대 3가지의 조치 액션과 필요한 Tool 리스트 (사전 정의됨) 생성
- **RemediationDecision**: 복구 조치 선택 (Human-in-the-loop)
 - 사용자가 Action Planner에서 제안한 3가지 조치 중 선택
 - 적합한 조치가 없을 경우, 재분석 단계로 돌아가거나 수동 조치도 가능
- **ActionExecutor**: RemediationDecision에서 사용자가 선택한 액션에 해당하는 도구 실행




Live Demo (with Gradio UI)

Live Demo

 **RCA Agent** 시연

LangGraph 기반 Root Cause Analysis 에이전트

이 데모는 시스템 장애 발생 시 자동으로 근본 원인을 분석하고 적절한 조치를 추천하는 RCA 에이전트를 시연합니다.

 **입력 정보**


서비스 이름


payment-service

에러 발생 시간


Black Friday 이벤트 중 결제 서비스 장애 시뮬레이션

2024-12-20 09:15:00


 **RCA 분석 시작**

 **분석 결과**

근본 원인 분석

 ****Root Cause Analysis 결과****

=====

 ****근본 원인****

- 근본 원인: 데이터베이스 연결 풀의 과도한 사용과 Redis 연결 문제로 인해 결제 서비스의 성능 저하 및 장애 발생.

- 분석 근거:

1. ****로그 및 트레이스 정보****: 로그에서 ``java.sql.SQLException: Connection is not available``와 트레이스에서 ``db_connection_timeout`` 에러가 발생하였으며, 이는 데이터베이스 연결 풀의 최대 용량 도달을 의미합니다.
``db_connection_pool_utilization``이 98%로 매우 높고, ``db_connection_timeouts``가 156건 발생한 점이 이를 뒷받침합니다.

2. ****Redis 연결 문제****: 로그에서 ``Redis connection timeout`` 에러가 발생하였고, 트레이스에서 ``redis_timeout`` 에러가 확인됩니다. 메트릭에서 ``redis_connection_errors``가 89건 발생하였고, ``redis_memory_usage_percent``가 97.1%로 매우 높은 상태입니다.

3. ****Stripe API Rate Limit****: 로그에서 ``Stripe API rate limit exceeded`` 에러가 발생하였고, 트레이스에서 ``rate_limit_exceeded`` 에러가 확인됩니다. 이는 외부 서비스 호출 빈도가 높아져 발생한 문제로, ``stripe_api_success_rate``가 52.3%로 낮은 점이 이를 뒷받침합니다.

- 영향 범위: 결제 서비스 전체에 영향을 미쳐, 결제 처리 지연 및 실패가 발생하였으며, 247건의 거래가 PENDING 상태로 남아 있고, 매 분당 \$2075의 수익 손실이 발생하고 있습니다.

Live Demo (with Gradio UI)

🎯 추천 액션

추천 조치 목록

설명: Stripe API 호출 빈도를 조정하여 Rule Limit 문제 해결

위험도: 중간

예상 시간: 5분

실행할 도구:

- reduce_traffic: {'service': 'payment-service', 'percentage': 30}
- restart_all_services: {'service': 'stripe-integration'}
- gradual_traffic_restore: {'service': 'payment-service', 'step': 5}

** 📁 추가 옵션:**

** manual. 수동 처리**

설명: 시스템 관리자가 직접 문제를 해결합니다

위험도: 사용자 판단

예상 시간: 사용자 판단

** re_analyze. 재분석**

설명: 시스템 상태를 다시 분석합니다

위험도: 없음

예상 시간: 3-5분

👉 위 액션 중 하나를 선택하여 실행하세요.

실행할 액션 선택

1, 2, 3: 추천 액션 / manual: 수동 처리 / re_analyze: 재분석

☐ 1

☐ 2

☒ 3

☐ manual

☐ re_analyze

⚡ 액션 실행

Live Demo (with Gradio UI)

실행 결과

액션 실행 결과

⚡ **액션 실행 결과**

🔴 **실행된 액션:** ECS 서비스 재시작 및 상태 검증
설명: ECS 서비스의 상태를 점검하고 필요 시 재시작하여 서비스의 안정성을 확보합니다.

🔍 **도구 실행 결과**

✅ check_ecs_health: 성공
- service: payment-service
- status: RUNNING
- running_count: 3
- desired_count: 3
- healthy_tasks: 2
- health_status: degraded

✅ restart_ecs_task: 성공
- service: payment-service
- action: restart_completed
- new_task_count: 3
- status: success

실행 히스토리

지금까지의 실행 기록

📅 **실행 히스토리**

1. 2025-08-29T09:49:43.242209
선택한 액션: 2
최종 상태: failed
실행 결과: 1/3 성공

2. 2025-08-29T09:50:10.622347

AI Ops Auto Pilot - 주요 기술

- **LangGraph** 를 활용한 State 기반 workflow 구성
- Event Data 수집을 위한 **외부 API (AWS API)** 호출
- LLM 기반 RCA 요약 Report, Action Plan 생성
 - LLM: (**Open AI - GPT 4o**) 모델 사용
 - **Tool Calling**: LLM 이 현재 수행 가능한 Tool 중에 적합한 조합까지 포함한 Action List 생성
- **Human-in-the-loop** 적용
 - langgraph interrupt 기능 및 조건부 라우팅을 사용하여 사용자 개입을 통한 승인 시스템 구축
 - 안전한 자동화
- **Gradio UI**: Web 기반 Demo 인터페이스 구성
- **LangGraph Studio**: 개발 및 디버깅에 이용

향후 계획 및 개선 방향

- (1) RCA 고도화
 - 과거 사례 학습을 통한 Feedback Loop (RAG) 구성으로 지속적 개선
 - 이전 사례와 post-action report 을 벡터 임베딩하여 유사한 사례 및 권장 조치 검색
 - 배포 로그 및 위키 등 다양한 event source 추가를 통해 다각도로 원인 분석
- (2) 현업 서비스 모니터링 파이프라인과의 통합
 - Datadog service error metric > threshold => Slack 알림 및 Agent 실행 => RCA & Remediation
 - Cloudwatch, Datadog, Open Telemetry 등 Log, Trace, Metric 실제 데이터 통합
- (3) Action Executor 파이프라인 고도화
 - Manual Remediation 이 필요한 경우 Report 와 함께 Jira 티켓 생성, 에러 원인에 맞는 담당자 escalation
- (4) Tool Calling 고도화
 - 다양한 지표 및 로그에 대한 RCA 및 Action 데이터 기반 최적화된 Tool Calling 구성
 - 실제 조치 전 Sandbox 기반의 Dry-run 을 통한 영향 범위 사전 시뮬레이션으로 안전성 강화