Jenkins 배치 통제 체계 구현

분산 운영 중이던 Jenkins 환경을 중앙 집중화하고, 배치 작업 전 과정에 내부 통제를 적용한 운영 체계를 구축

Jenkins Python ITGC Compliance

프로젝트 배경

- 당시 Jenkins는 부서별로 독립 운영되며, 모니터링·권한 제어·증적 관리 등 금융 서비스에 필수적인 IT 통제 기능이 미비해 관리가 어려웠습니다.
- 특히 작업 이력이나 실패 기록을 체계적으로 관리할 수 있는 프로세스가 부재하여, 승인 누락·권한 오남용·장애 원인 추적 등에서 리스크가 존재했습니다.

역할 및 수행 업무

배치 운영 총괄

- 정보보호·내부통제 요구사항을 반영해 등록–승인–실행–이력 관리 전 과정을 체계화
- Jenkins 환경을 개선하며 안정적 운영과 자동화 기반의 관리 구조 확립

배치 환경 중앙화 및 협업 체계 구축

- 부서별로 분산·독립 운영되던 Jenkins 환경을 중앙 집중 관리 체계로 전환
- 여러 부서와 협업해 요구사항을 반영하고, 공용 정책·운영 가이드를 수립
- 개발자 및 타 부서 요청자에게 표준화된 절차와 가이드 제공

승인 프로세스 기반 배치 작업 관리

- 절차를 자동화하여 배치 작업 등록·변경·실패 시 통제 일관성 확보
- 자동 검증 및 리포트 체계를 통해 운영 효율과 정확성 향상

운영 증적 확보 강화

- 작업 실행 로그를 실시간 수집해 성공/실패 여부를 즉시 확인
- 실행·실패 시 자동 리포트 발송 체계를 구현

- 배치 실행 현황 및 오류 이력을 감사 증적으로 활용

권한 관리 및 통제 체계 운영

- 직무 분리 원칙에 따라 작업 등록·수정·실행 권한을 차등 부여
- 승인·변경 이력 기반의 통제 구조를 통해 보안성과 책임성을 강화

감사 및 내부통제 대응 지원

- 감사 자료 제공 및 피드백 반영
- 증적 이력 보고서 작성 및 감사 대응 로직 검토·리스크 제거
- 감독규정 개정 시 통제 완화·보완 방안을 검토하여 제도 변화에 선제 대응

서비스 연속성 확보 및 리스크 관리

- 망분리 강화 환경에서도 서비스 중단 없이 안정적 운영 지속
- 자원 부족·스토리지 장애 등 긴급 이슈에 즉각 대응하여 장애 영향 최소화
- DR 체계 구축 및 복구 훈련을 통해 재해 발생 시 신속한 복구 대응 체계 확립

자동화 기반 운영 효율화

- 배치 요청·승인·결과 알림을 웍스봇으로 자동화하여 업무 속도 향상
- Jenkins 노드 관리 및 상태 모니터링 자동화를 통해 운영 부담 경감

프로젝트 성과 및 기여

직무 분리 실현 및 통제 리스크 차단

- 개발자는 요청·코드 작성, 운영자는 승인·실행 관리로 역할을 명확히 분리
- 승인 누락·권한 오남용 등 핵심 통제 리스크를 구조적으로 예방

감사 대응 역량 강화

- 감사 대응용 증적 체계를 데이터 기반으로 완성
- 요청 시 즉시 제출 가능한 이력·로그 체계 구축으로 감사 대응 속도 향상

서비스 연속성 확보

- 망분리 정책 강화, NAS 장애 등 환경 변화에도 사전 리스크 검토 및 자동화 대응 수행
- 운영 중단 없이 배치 서비스 연속성을 보장

운영 효율성 향상

- Job 라벨링 및 노드 추가 자동화로 운영자 개입 최소화
- 장애 인지 및 복구 대응 속도를 실시간 수준으로 개선

통제 기반 운영 문화 정착

- 단순 운영을 넘어 '감사 대응 가능한 운영 체계'로 전환
- 정책 요구를 넘어 협업 중심의 실행 가능한 프로세스를 설계