
openmaru
APM
오픈마루가 직접 OpenShift 컨설팅부터 구축까지 수행한 고객

기존 환경에는 어떠한 문제가 있었나? (1/3)

openmaru

비 표준화 되어있는 빌드 배포 프로세스

- 서비스 담당자 개인 로컬 환경에서 빌드 후 배포되는 환경으로 서로 상이한 조건에서 빌드 및 배포
- 수동 빌드 배포 작업
- 개인적으로 애플리케이션을 빌드 후 FTP를 통해 서버에 업로드 및 WAS 재기동
- 기존 빌드 바이너리(.war)파일을 수작업으로 롤백하는 과정을 수행 중
- 정비 PM은 화요일 오전에 진행하며, 긴급 배포가 필요한 경우 수시 진행
- 관련하여 간혹 휴먼에러가 발생함
- 루틴한 작업에 대해서는 초기에 휴먼에러가 있었으나 현재는 많지않음
- ISMS 심사 필요
- 접근제어 솔루션을 도입할 예정이며, 이를 통해 인가 받은 사용자만 접근 가능하도록 계획중
- 종속성 라이브러리는 망분리 된 환경으로 인해 미러링 구성을 이미 사용 중

기존 환경에는 어떠한 문제가 있었나? (2/3)

openmaru

컨테이너 에코시스템 관리

- 컨테이너를 단일 빌드하여 별도 오케스트레이션 도구 없이 실행
- 컨테이너를 포트 포워딩을 이용한 서비스 노출 방식
- 컨테이너, MSA 기반 설계상의 애플리케이션 로그 수집에 대한 테스트 중
- 이슈가 발생하면 어떤 서비스에서 로그가 쌓였는지 찾는데 시간 소요
- 일부 서비스에 대해 ELK를 도입하여 운영하려고 시도하였으며, 로그 통합에 대한 필요성 있음
- 모니터링은 Spring boot 컨테이너의 경우 개별적으로 메모리 정도의 메트릭만 확인

기존 환경에는 어떠한 문제가 있었나? (3/3)

openmaru

서비스 및 확장성

- MSA 관점에서 서비스 노드간의 트래픽이 시각화 되면,
  추후 확장시 도움 될것으로 보여지나 Netflix OSS 기준으로 테스트 수준
- Service Mesh 기능에 대한 필요성
- 서버 확장을 위한 느린 프로세스
  - On-Premise 구조로 서버 확장을 위해서는 3주 정도 소요(하드웨어 도입기간 제외)
  - 최소 두 달 전부터 추이를 보고 준비 필요

기존 환경 VS 컨테이너 오케스트레이션 환경

대분류	소분류	기존 환경	컨테이너 오케스트레이션
빌드 배포	빌드 환경	로컬 환경에 따른 상이한 환경	완전 자동
	자동화	없음	완전 자동
	롤백	수작업 및 긴 시간	완전 자동
	휴먼 에러	발생 가능성 높음	발생 가능성 낮음
컨테이너	오케스트레이션	없음	있음
	로그통합	없음	있음
	애플리케이션 모니터링	OPENMARU APM 부분 활용	OPENMARU APM 전체 활용
MSA	트래픽 시각화	없음	있음
	서비스 메쉬	없음	있음
서버 확장	WAS Instance 확장	수작업 및 긴 기간 소요	자동 및 즉각 확장
	서버 확장	수작업 및 긴 기간 소요	단순 명령 2~3줄로 확장 가능

K8S가 아닌 OpenShift를 선택했는가?

openmaru

기대한 컨테이너 오케스트레이션의 기능

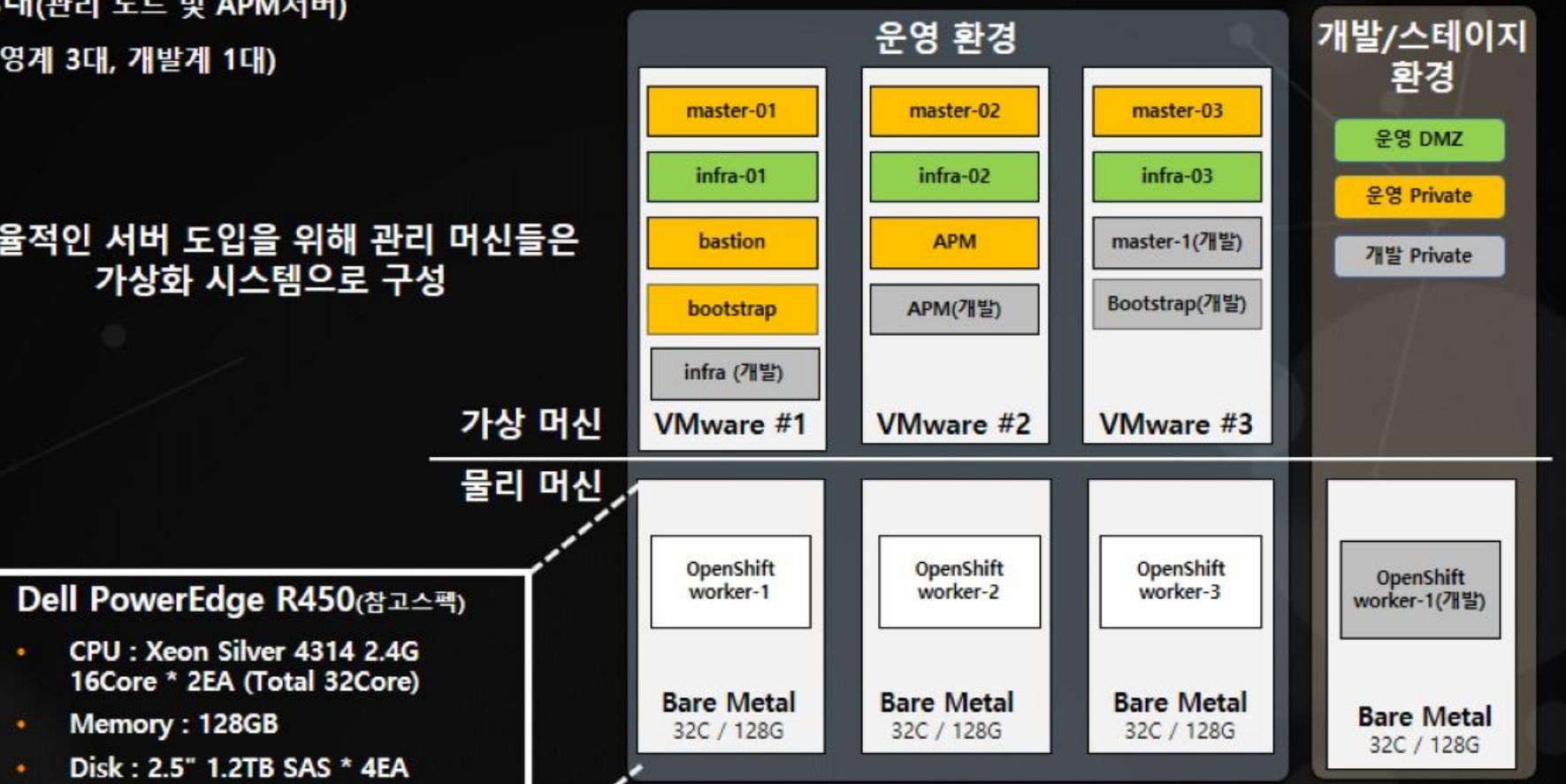
- 배포 관리 도구와의 통합
- 로깅 통합
- 상세한 모니터링과 지표에 따른 자동 확장 기능
- 서비스 메시 기능에 쉬운 적용
- 새로운 서비스 구성 및 대외 서비스와의 연계시 발생할 수 있는 부하 감당
- 서버 증설과 같은 요건이 발생하는 경우 쉬운 인프라 확장성
- 클라우드 기반 운영시 운영 인원 최소화 및 쉬운 관리 환경
- 망분리 요건에서의 구성 가능한 플랫폼
- 새로운 플랫폼 도입시 내재화 가능한 서비스 및 프로그램 여부

OpenShift 구성 신규 도입 서버

OpenShift 구성을 위한 신규도입서버 7대

- 신규 VMware 3대(관리 노드 및 APM서버)
- 물리서버 4대(운영계 3대, 개발계 1대)

효율적인 서버 도입을 위해 관리 머신들은
가상화 시스템으로 구성



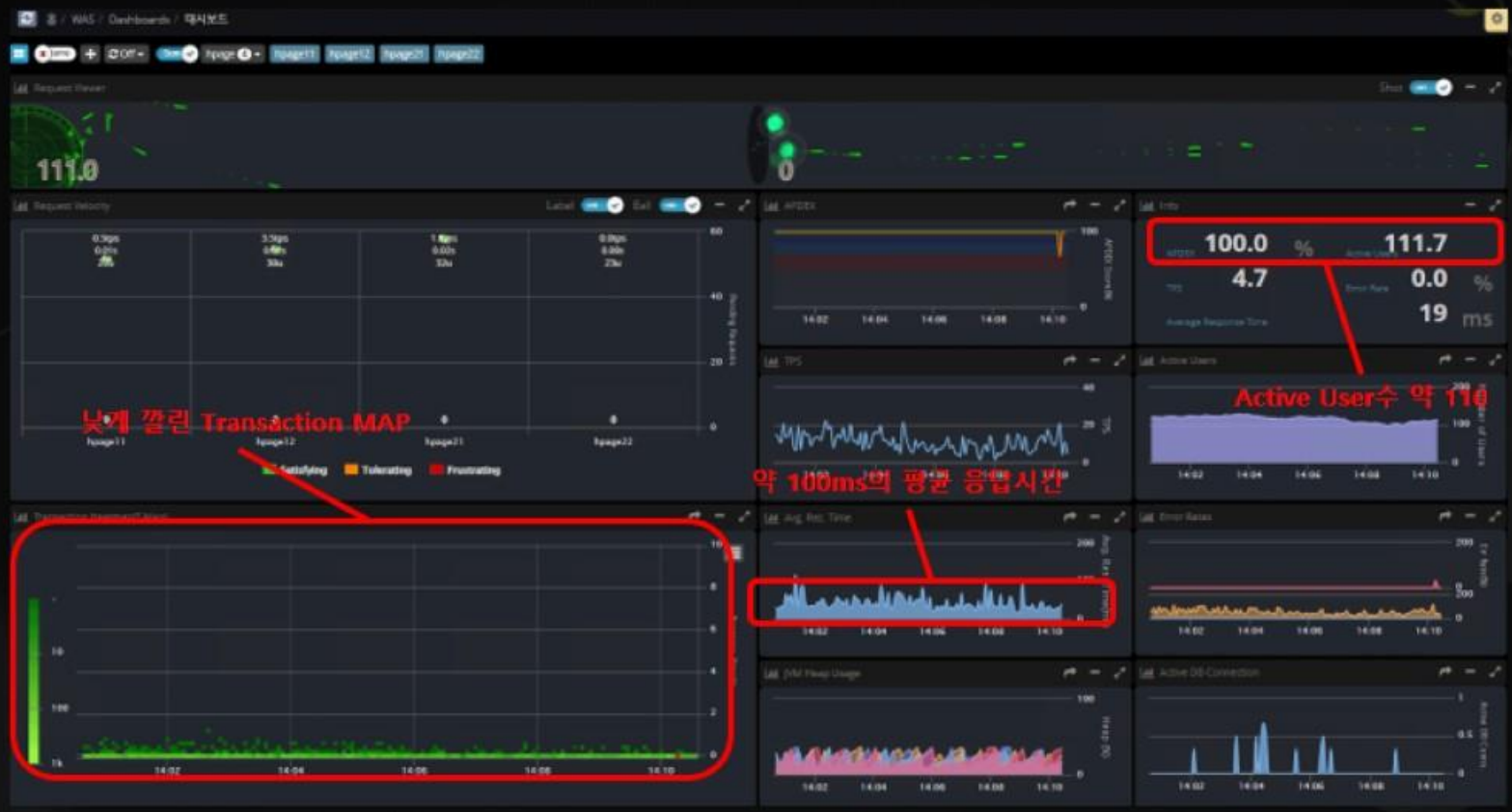
Dell PowerEdge R450(참고스펙)

- CPU : Xeon Silver 4314 2.4G 16Core * 2EA (Total 32Core)
- Memory : 128GB
- Disk : 2.5" 1.2TB SAS * 4EA
- NIC : X710 10G 2port Card * 2EA

AS-IS 환경 성능

확장/축소가 자동화 되지 않은 환경의 성능

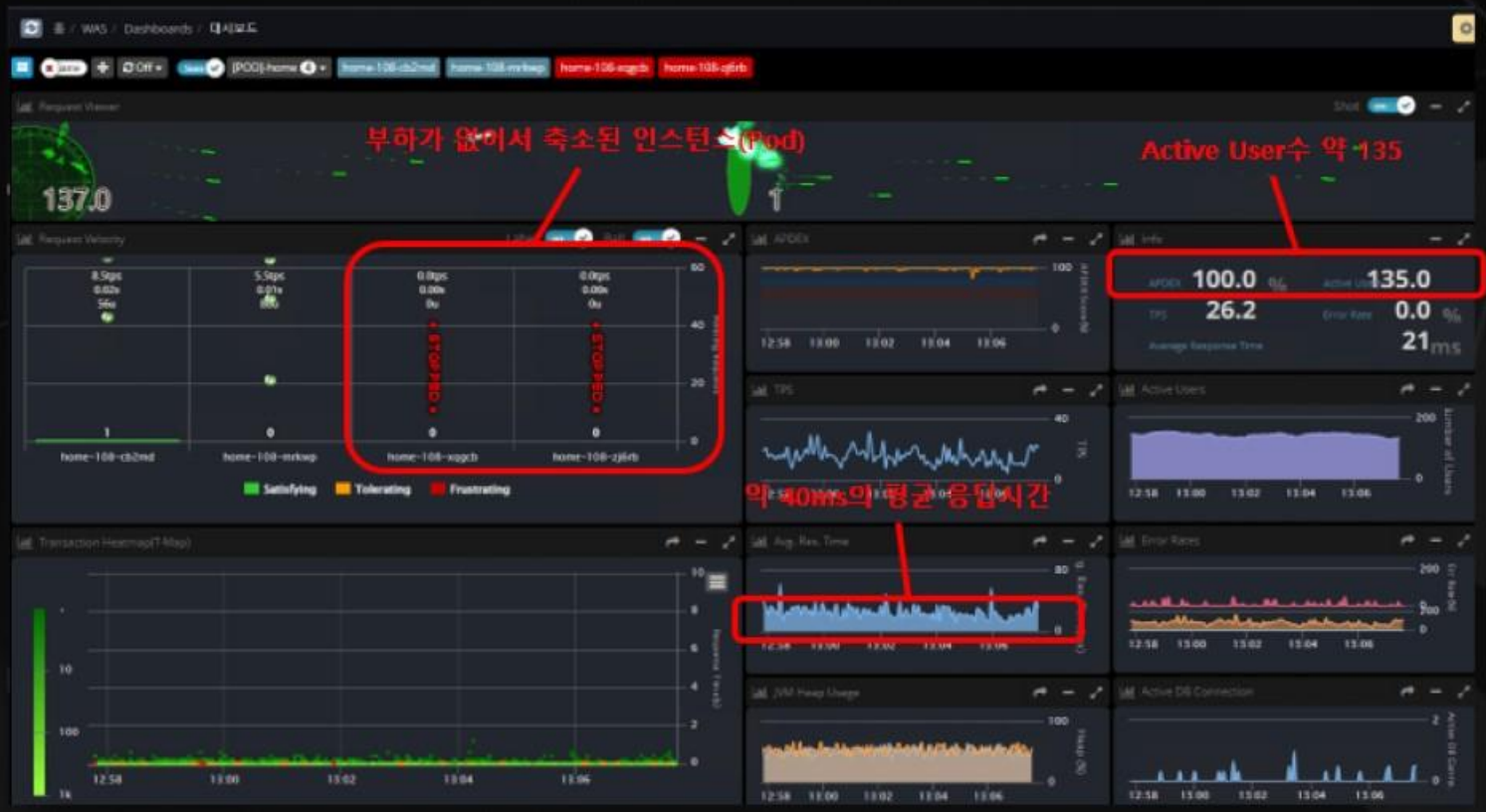
- 기존 환경 홈페이지에 대한 성능 분석
- 부하가 없는 상황이지만 가상화 환경임에 따라 유연하게 인스턴스를 축소할 수 없음
- 필요하지 않은 서버를 기동함에 따라 불필요한 리소스 낭비 발생



TO-BE 환경 성능

확장/축소의 자동화 환경의 성능

- 컨테이너 환경 홈페이지에 대한 성능 분석
- 부하가 없는 상황에 유연하게 인스턴스를 축소
- 비교적 더 많은 Active User에도 더 빠른 평균 응답시간 측정



성능 변화에 대한 요약

기존 환경과 컨테이너 환경의 성능 비교

- 15시 ~ 16시 경 가장 많은 부하시간 기준
- 더 많은 사용자, 부하이지만 더 빨라진 평균응답시간
- 대외서비스인 홈페이지, 모바일 애플리케이션에 대한 비교

분류	홈페이지	모바일
Active User	약 230명	약 1200명
T-MAP	3~6초 다수 식별	3~6초 식별
평균 응답시간	45ms	37ms
TPS	110	114

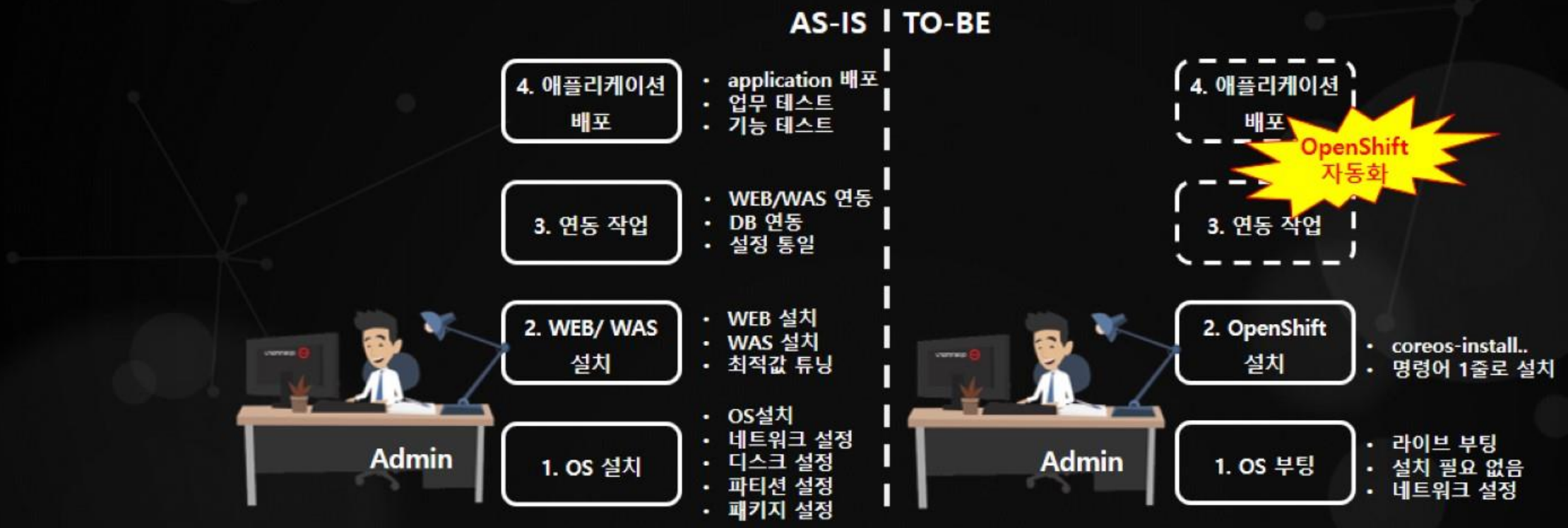
분류	홈페이지	모바일
Active User	약 730명	약 1500명
T-MAP	2초 이내	2초 이내
평균 응답시간	30ms	21ms
TPS	210	230

응답시간 기준
약 30% 빨라짐

서버 확장 운영관리 비교

인프라 팀(확장성)

- 기존 환경을 확장하기 위해서는 WEB/WAS 설치 및 연동 모든 구간 수작업 필요
- 기존 환경과 동일한 설정을 수작업으로 유지해야함
- 컨테이너 환경은 커맨드 기준 2~3줄의 Worker Node 추가만 하면 작업 완료

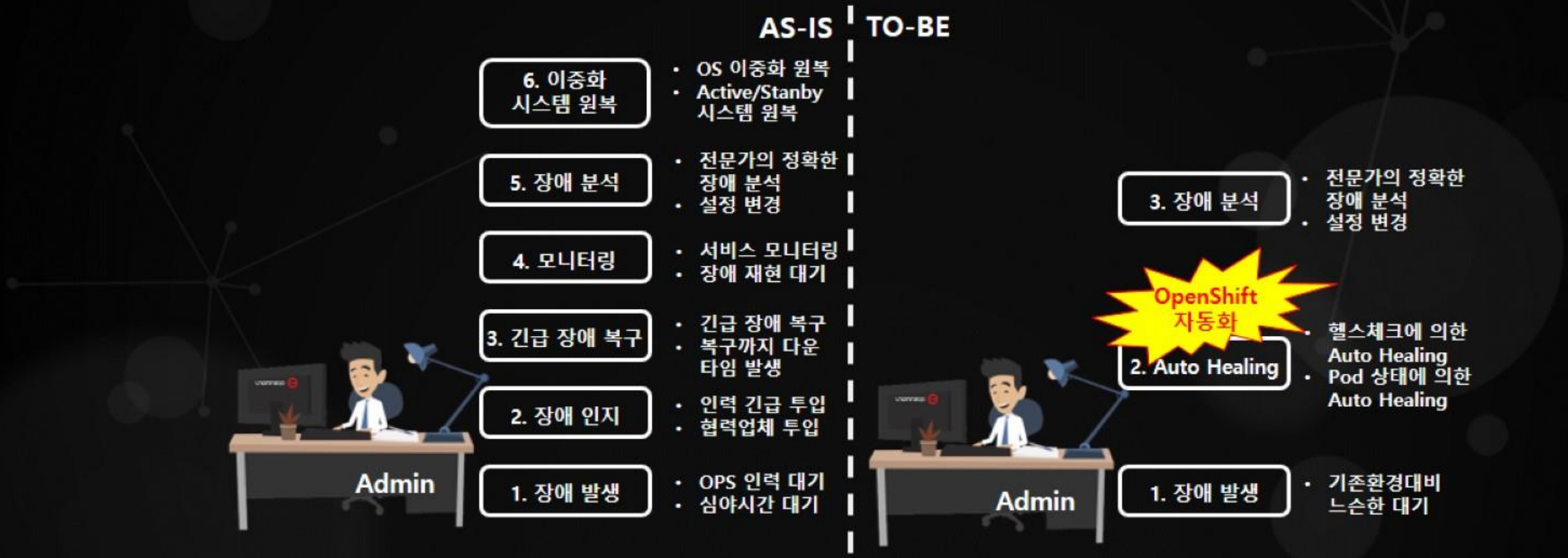


장애 상황 운영관리 비교

openmaru

인프라 팀(장애상황)

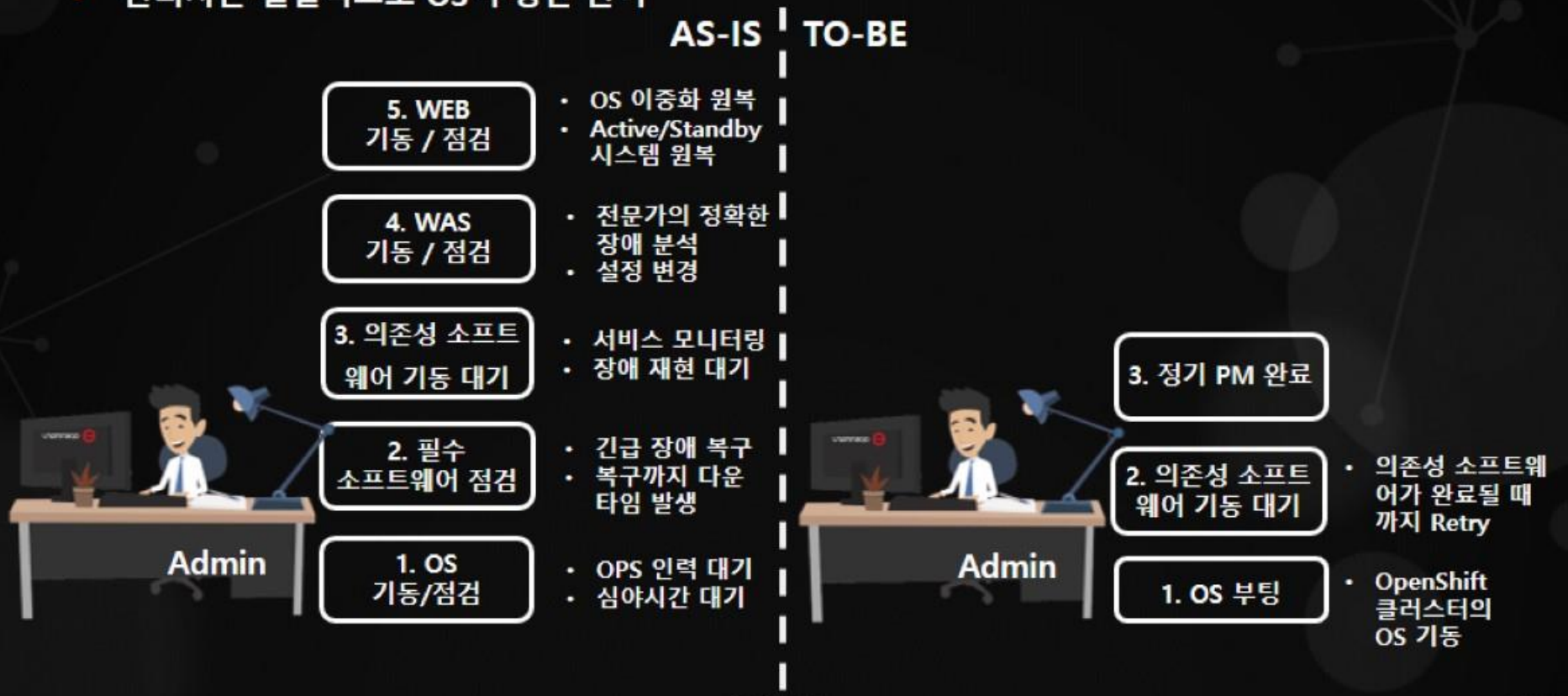
- 기존 환경은 장애 발생을 대비한 인력들이 상시 대기해야 함
- 야간 장애 발생시 **복구까지 긴 시간 소요, 서비스 다운 타임** 발생
- 컨테이너 환경은 Auto Healing으로 인해 장애 발생시 바로 자동 복구



정기 PM 운영관리 비교

인프라 팀(정기 PM)

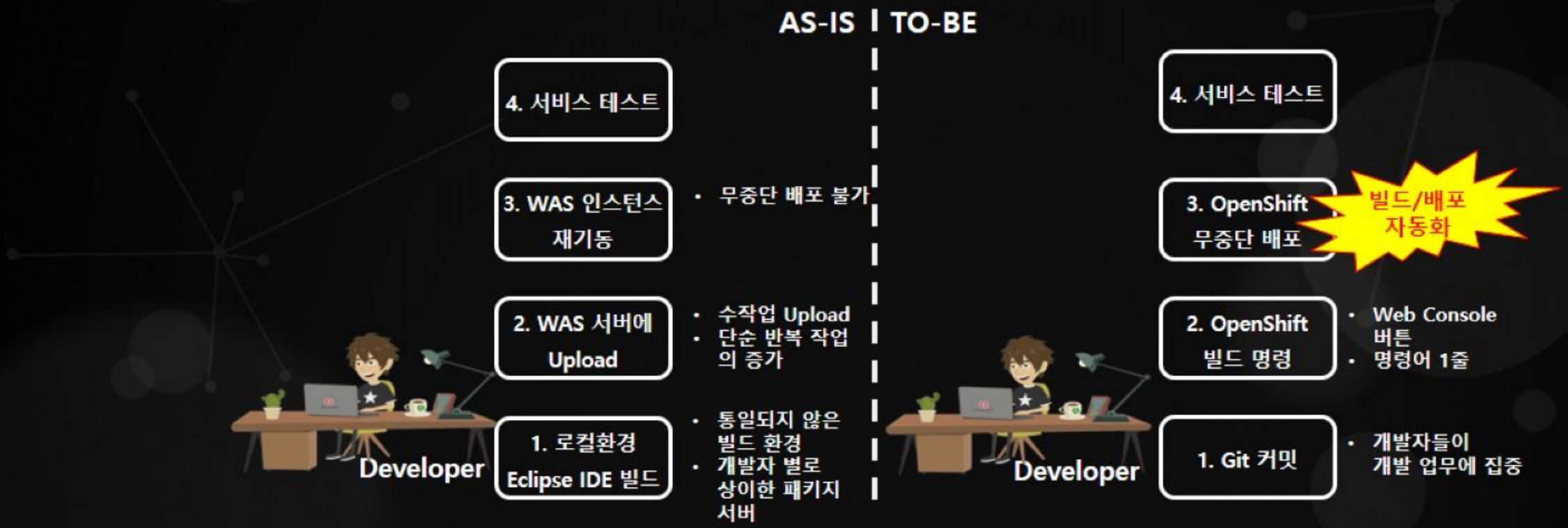
- 기존 환경은 소프트웨어에 따라 기동 순서가 있음
- 의존성 소프트웨어(ex: DataBase)가 정상 기동 될 때 까지 관리자는 작업 대기
- 컨테이너 환경은 의존성 소프트웨어로 인해 기동 실패하더라도 성공 시 까지 Retry
- 관리자는 실질적으로 OS 부팅만 관여



빌드/배포 운영관리 비교

개발 팀(빌드/배포)

- 기존 환경은 애플리케이션을 빌드, 서버에 업로드, 재기동 모두 수작업으로 진행
- 무중단 배포가 불가능하였으며 배포작업시 휴먼 에러 발생가능성이 높음
- 컨테이너 환경은 빌드 명령 혹은 Console 버튼으로 빌드, 배포까지 완전 자동화
- OpenShift에 대한 전문 지식이 없어도 빌드 배포 가능



애플리케이션 Rollback 운영관리 비교

openmaru

개발 팀(Rollback)

- 애플리케이션 롤백 또한 자동화 되지 않은 수작업으로 진행
- 기존 환경의 빌드 배포와 같은 방식
- 컨테이너 환경은 애플리케이션 배포 실패 시 자동으로 정상 버전으로 롤백
- 관리자의 개입으로 명령어 한 줄로 특정 버전의 애플리케이션 버전으로 롤백 가능

제품 / 서비스에 관한 문의

- 콜 센터 : 02-469-5426 (휴대폰 : 010-2243-3394)
- 전자 메일 : sales@opennaru.com