MSAP.ai

AINP 소개

AI Native Platform

Kubernetes 는 학습장벽이 높은 기술

MSAP.ai

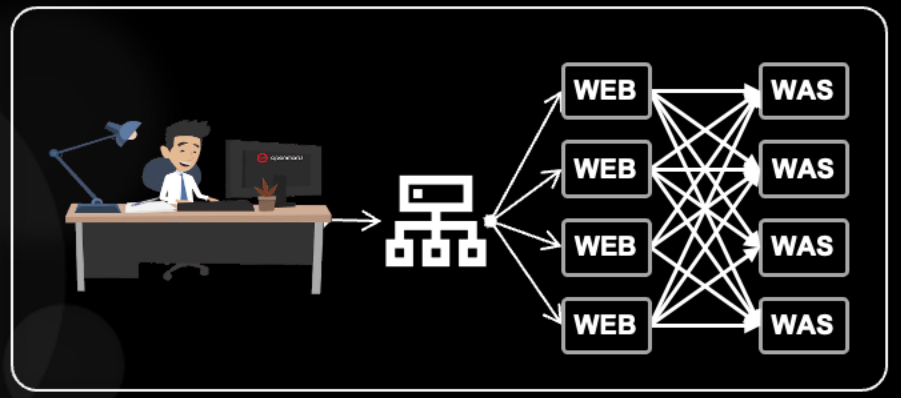
- 컨테이너를 유용하게 활용하기 위해서는 다양한 지식이 요구

시스템 운영의 현실

MSAP.ai

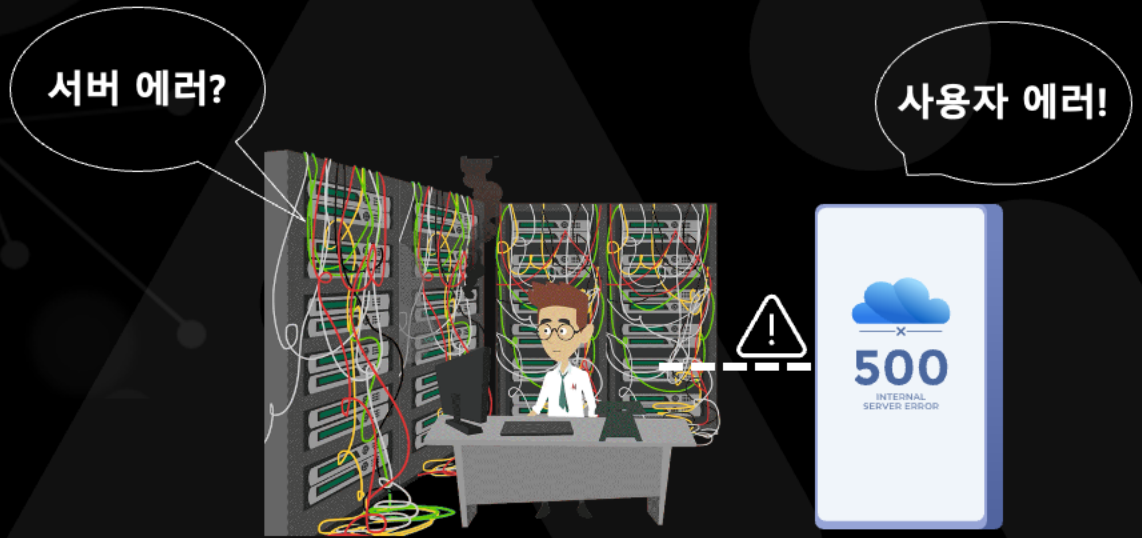
웹시스템의 요구사항과 특징

- Scale-out 형 인프라로 계층 형 아키텍처
- 가상화와 클라우드환경에 적합하며 인스턴스 개수가 많으며 노드 간 연결성이 높음
- 장애 민감하며 신속한 장애 복구와 재발 방지가 중요
- 설치되는 관련 소프트웨어가 많으며 환경 검증이 필요



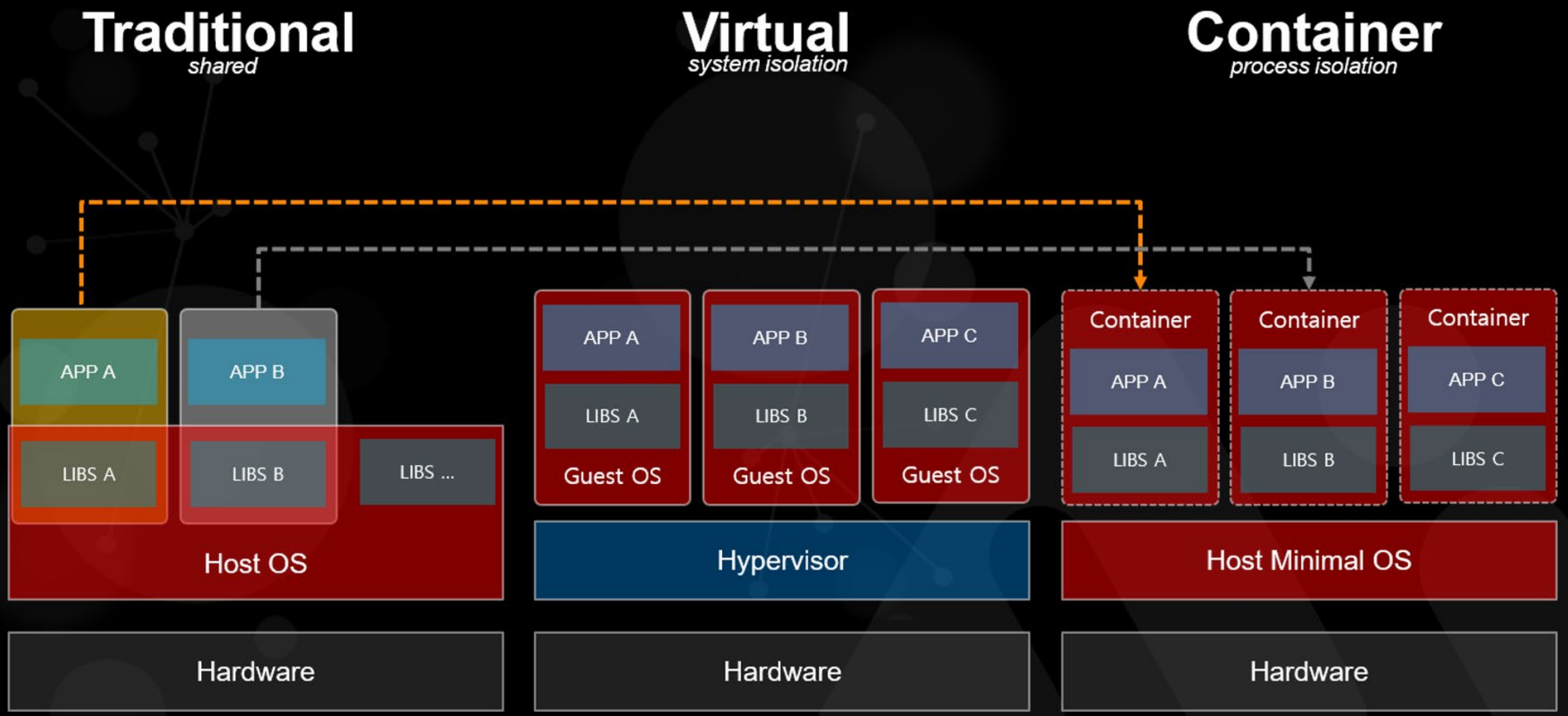
시스템 운영 이슈

- 시스템 환경의 불일치(Dev/Stage/Prod, 서버별)
- 긴 배포 시간
- 수 작업으로 인한 Human Errors
- IT Agility 부족으로 인한 운영팀 축소
  - 개발팀에서 직접 IT 인프라 운영
- 문제점의 발견과 조치에 많은 시간 소요



AINP (AI Native Platform) - Evolution

MSAP.ai



클라우드 네이티브 시대의 도전 과제

MSAP.ai

- "혁신을 위한 여정, 그러나 복잡성은 증가합니다."

기술의 파편화

마이크로서비스, 컨테이너, 쿠버네티스 등 다양한 기술 스택의 등 장으로 관리 포인트가 분산되고 복잡성이 기하급수적으로 증가합 니다.

운영의 어려움

분산된 멀티/하이브리드 클라우드 환경에서 일관된 정책을 적용 하고 장애를 추적하는 것은 매우 어렵습니다.

높은 총소유비용(TCO)

쿠버네티스 전문가 확보의 어려움, 가파른 학습 곡선, 지속적인 운 영 관리 비용은 기업에 큰 부담으로 작용합니다.

느려지는 개발 속도

개발자들이 인프라 문제에 시간을 쏟게 되어, 정작 중요한 비즈니 스 로직 개발에 집중하지 못합니다.

AINP (AI Native Platform)으로 이러한 도전들을 해결하세요.

클라우드 지원을 위한 애플리케이션 현대화

MSAP.ai

- Lift & Shift – 이동성은 애플리케이션 컨테이너화만으로는 부족
- 클라우드 네이티브 애플리케이션을 통한 클라우드 완벽 지원

기존 온-프레미스 사용자

새로운 클라우드 사용자

***Lift & Shift** Approach*

***Cloud Native** Approach*

컨테이너 전환 도구

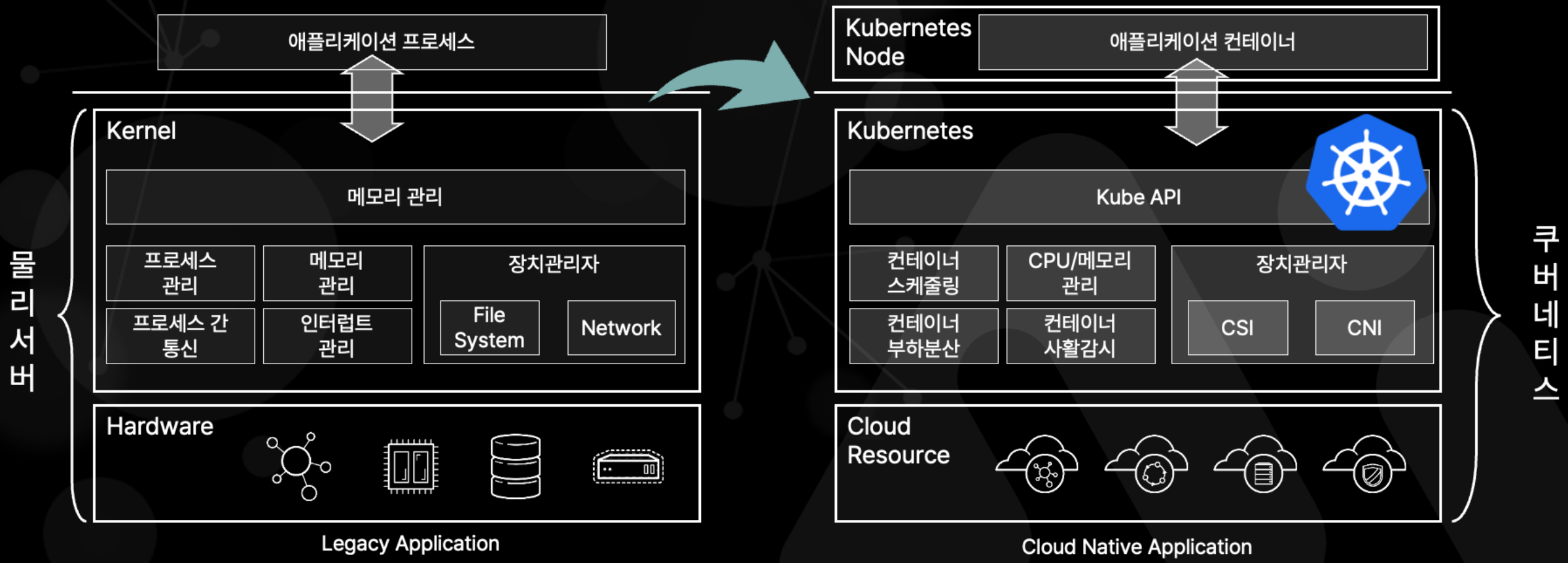
DevOps 구현 도구

Reliable ⟷ **Agile**

Kubernetes – Cloud Kernel (OS)

MSAP.ai

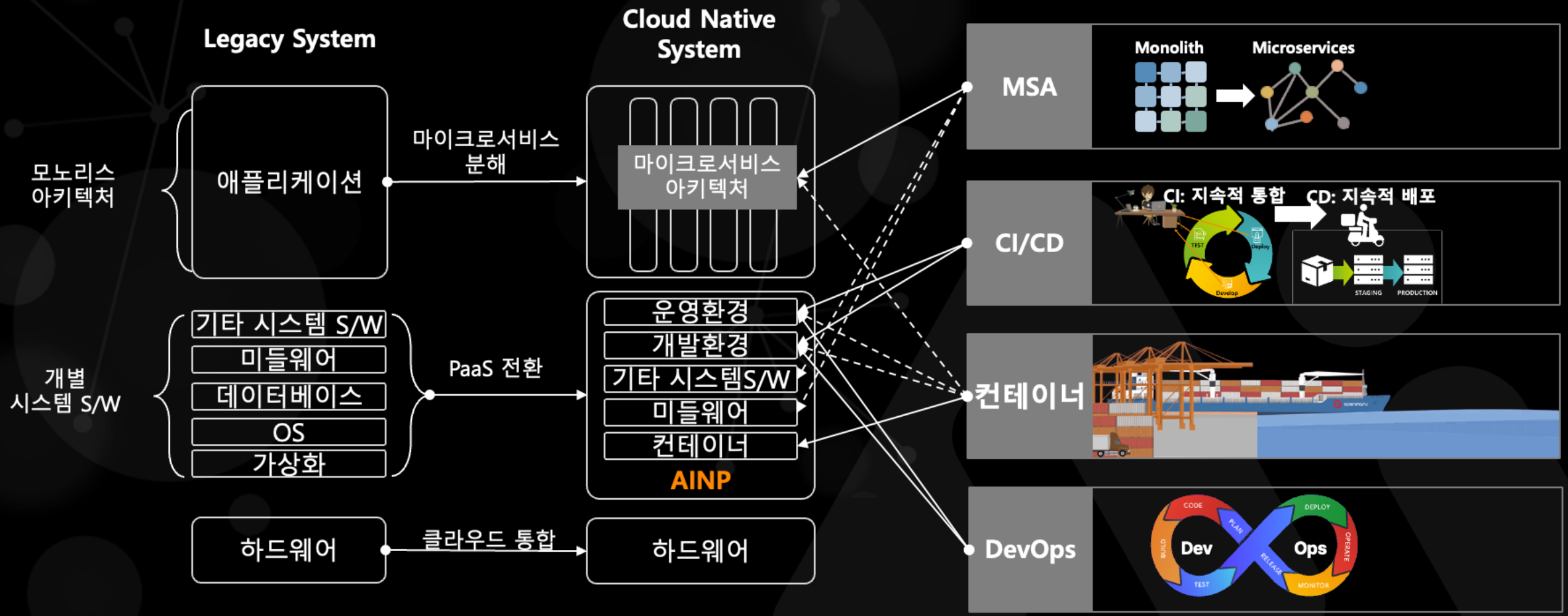
- 애플리케이션이 특정 서버에 의존하지 않고, 클라우드 환경 내에서 유연한 확장과 관리 지원
- Kubernetes API 로 자동 복구, 고가용성, 자동 스케일링과 같은 클라우드 네이티브 운영 지원



IT 운영환경도 클라우드 네이티브 기반 AINP로 전환

MSAP.ai

- 클라우드 네이티브는 "클라우드가 '클라우드 다울 수 있도록' 애플리케이션을 구축, 실행하는 방식"



MSAP.ai

AINP (AI Native Platform) 개요

AINP(AI Native Platform) 개요

MSAP.ai

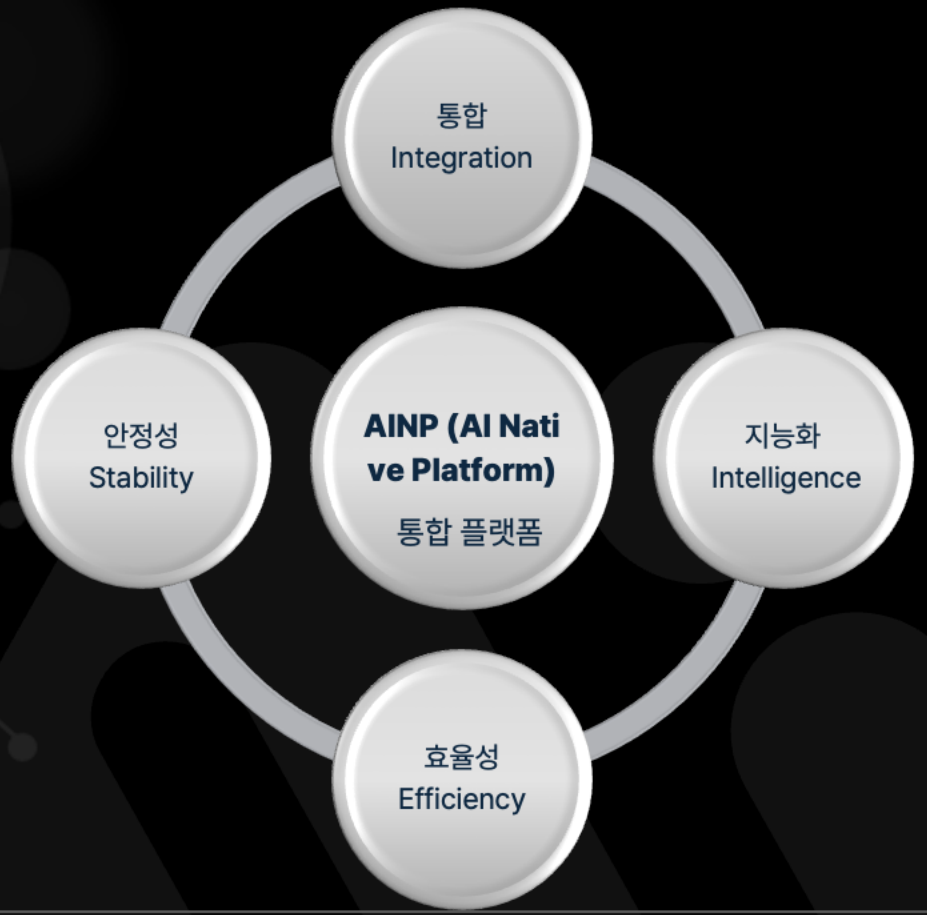
AINP이란?

AINP (AI Native Platform)은 쿠버네티스라는 강력한 엔진 위에 기업 환경에 필수적인
모든 요소를 완벽하게 통합하여 제공하는 '완성형 플랫폼(Well- architected Platform)'입니다.

제공 가치

- ☑ 복잡성을 해결하는 단 하나의 완성형 플랫폼
- ☑ 쿠버네티스 도입 효과 극대화
- ☑ TCO 및 운영 리스크 최소화
- ☑ 개발자와 운영자 모두의 생산성 향상

주요 특징



개발자 & 운영자 | 멀티/하이브리드 클라우드 | 완전히 통합된 운영

AINP (AI Native Platform) 핵심 구성요소

MSAP.ai

AINP
통합 플랫폼

Kubernetes
컨테이너 오케스트레이션 표준인 쿠버네티스를 기반으로 확장성, 안정성, 이식성을 보장

Session Clustering
분산 환경에서 세션 불일치 문 제를 해결하여 중단 없는 서비스 보장

APM
애플리케이션 성능 병목 구간을 실시간으로 추적하고 최적화

Observability
메트릭, 로그, 트레이스를 통합 한 전방위적 관찰 능력 제공

지능형 관리 (LLM)
AI를 활용한 예측 및 자동화된 운영으로 TCO 최소화

AINP (AI Native Platform)은 핵심 구성요소들을 유기적으로 결합하여 클라우드 네이티브 환경의 복잡성을 해결합니다.

AINP의 주요 특장점 6가지

MSAP.ai

지능형 자율 운영 지원	LLM 기반 AI 기술을 활용하여 시스템 상태를 예측하고 사전 대응을 통해 운영 안정성을 극대화
세션 클러스터링 내장	WAS 간 세션 공유 및 복제를 안정적으로 처리하여, 세션 유지가 중요한 업무 시스템에서 고가용성을 보장
Kubernetes 친화적 설계	쿠버네티스 엔진 위에서 작동하는 구조로, 클러스터 상태, 파드/컨테이너 동작, 리소스 사용량까지 정밀하게 모니터링
실시간 APM 기능	애플리케이션 트랜잭션, SQL, 외부 API 호출 등 전체 성능 흐름을 실시간으로 추적하고 병목을 분석
관찰 가능성 통합 플랫폼	로그, 메트릭, 이벤트, 트레이스를 통합 분석하여 운영 리스크를 빠르게 식별할 수 있도록 지원
운영 자동화 및 유연한 확장성	수천 개의 마이크로서비스가 운영되는 대규모 환경에서도 스케일 아웃·인 대응 및 자동 설정이 용이

Kubernetes 기반 인프라 관리

MSAP.ai

- AINP (AI Native Platform)은 컨테이너 오케스트레이션의 사실상 표준인 쿠버네티스를 기반으로 확장성, 안정성, 이식성을 보장합니다. 순수 쿠버네티스의 복잡성을 추상화하고, 엔터프라이즈 환경에 최적화된 기능을 제공합니다.

쿠버네티스 기반 인프라 특징

확장성
분산된 클러스터 관리 및 수평 확장 지원

안정성
자동 복구, 자동 스케일링, 서비스 중단 없이 업데이트

이식성
클라우드 제공업체에 종속되지 않고 모든 환경에서 실행

AINP (AI Native Platform)의 향상된 인프라 관리

추상화된 인프라 관리
쿠버네티스의 복잡성을 숨겨 사용자에게 단순한 인터페이스 제공

엔터프라이즈 최적화
기업 환경에 특화된 보안, 모니터링, 규정 준수 기능

완성형 플랫폼
애플리케이션의 전체 라이프사이클 관리에 필요한 모든 도구 통합

! AINP (AI Native Platform)은 쿠버네티스의 기능을 확장하고, 기업 환경에 맞게 최적화합니다.

AINP Pod Cluster – POD 간 세션 공유

- AINPSession Clustering은
  분산 환경에서 Stateful 애플리케이션의
  가장 큰 고민인 세션 불일치 문제를 해결합니다.

중단 없는 서비스 보장

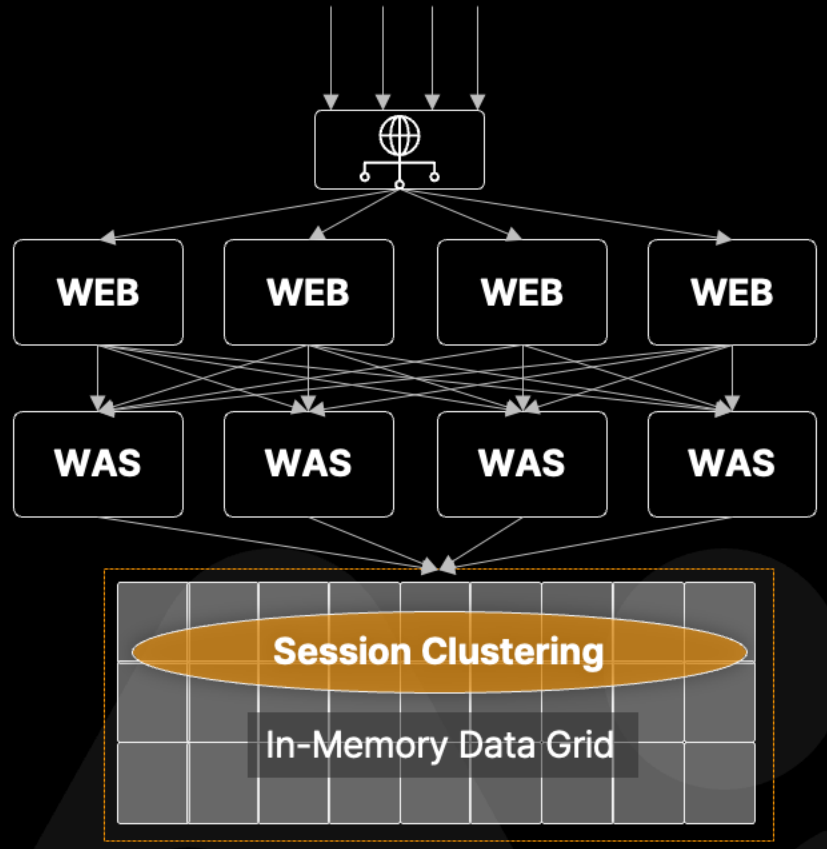
특정 Pod나 노드에 장애가 발생하더라도 사용자의 세션 정보를 안전 하게 유지합니다.

세션 정보의 안전한 유지

분산된 노드 간에 세션 상태를 복제하고 동기화하여 언제 든지 접근할 수 있습니다.

고성능 데이터 저장소

In Memory 기반 고성능 데이터 저장소를 활용하여 빠르게 세션 데이터 에 접근합니다.



세션 클러스터링의 작동 방식

AINPCluster은 분산된 여러 노드에 걸쳐 세션 데이터를 복제하고 동기화합니다. 따라서 사용자가 다른 서버로 이동하거나 특정 서버가 다운되어도, 세션 정보는 안전하게 유지되고 사용자는 중단 없이 서비스를 계속 받을 수 있습니다.

AINP APM-애플리케이션 성능 모니터링

MSAP.ai

- AINPAPM은 애플리케이션의 성능 병목 구간을 실시간으로 탐지하고 분석하는 강력한 도구입니다. 쿠버네티스 환경에서 애플리케이션의 성능 문제를 효과적으로 진단 하고 해결할 수 있습니다.

실시간 성능 모니터링

애플리케이션의 성능 지표를 실시간으로 수집하고 표시하여 병목 현상을 즉시 감지합니다.

병목 구간 탐지

코드 레벨에서 성능 병목 구간을 정확히 파악하고, 호출 트레이스를 통해 원인을 추적합니다.

성능 트렌드 분석

시간별, 날짜별 성능 데이터를 통해 트렌드를 분 석하고, 예측 가능한 성능 저하를 방지합니다.

소스 코드 맵핑

성능 데이터를 소스 코드로 맵핑하여 개발자가 문제를 직접 찾아 수정할 수 있습니다.

AINPAPM의 가치

- [x] 애플리케이션 성능 문제의 근본 원인을 빠르게 파악하고 해결
- [x] 쿠버네티스 도입 효과 극대화
- [x] TCO 및 운영 리스크 최소화
- [x] 개발자와 운영자 모두의 생산성 향상

AINP Observability – 포괄적 시스템 관측성

MSAP.ai

- 메트릭, 로그, 트레이스를 통합적으로 관리하여 시스템 전반의 관찰 가능성을 높이는 기능

메트릭

시스템 및 애플리케이션의 성능 지표를 실시간으로 모니터링하고 분석합니다. CPU 사용량, 메모리 소비, 요청 처리량 등 수량적 데이터를 통해 시스템의 상태를 정량화합니다

로그

애플리케이션의 동작 기록을 수집하고 검색 가능한 형식으로 저장합니다. 에러 메시지, 경 고, 정보성 로그 등으로부터 문제의 증상을 찾 아내고 진단합니다.

트레이스

분산된 서비스 호출 흐름을 추적합니다. 서비스 간 의존성 및 호출 지연을 통해 서비스 전체의 실행 흐름을 이해하고 최적화합니다.

통합 관찰성의 가치

통합 데이터로 문제의 근본 원인을 빠르게 파악합니다. 기존에 분리되어 있던 메트릭 , 로그, 트레이스를 하나의 컨텍스트로 통합합니다.

서비스 간의 의존성을 한눈에 보여주는 토폴로지 시각화로 복잡한 시스템 구조를 이해합니다.

컨테이너 환경 최적화와 실시간 성능 분석으로 애플리케이션의 성능을 지 속적으로 모니터링하고 개선합니다.

단순히 로그와 그래프를 모아 보는 수준을 넘어, 시스템의 모든 상태를 탐색 적으로 이해하는 능력을 제공합니다.

AINP CogentAI - VibeOps 지원

MSAP.ai

- AINP (AI Native Platform)의 지능형 관리 기능은 MSAP의 최신 AI 기술인 CogentAI를 기반으로 합니다.

주요 특징

고도화된 RAG (검색 증강 생성)

LLM의 약점인 환각(Hallucination) 현상을 극복합니다. 운영자의 질문에 대해 시스템의 실시간 데이터와 내부 기술 문서를 참조하여, 가장 정확하고 신뢰도 높은 답변을 생성합니다.

자동화된 지식베이스 구축

내부 장애 대응 매뉴얼, 기술 문서(PDF, Word 등)를 업로드하면 AI가 자동으로 학습하여 지식베이스를 구축합니다. 이를 통해 조직의 운영 노하우가 시스템에 축적됩니다.

RAG 기반 AI 분석 과정

1. **질문 이해**
   사용자의 질문을 분석하고 필요한 정보를 파악합니다.
2. **정보 검색**
   시스템 데이터베이스와 연결된 문서에서 관련 정보를 검색합니다.
3. **컨텍스트 생성**
   검색된 정보와 사용자의 질문을 바탕으로 AI가 이해할 수 있는 풍부 한 컨텍스트를 생성합니다.
4. **답변 생성**
   AI 모델이 최종 답변을 생성하고, 신뢰도 높은 결과를 제공합니다.

개발자를 위한 AINP (AI Native Platform)

MSAP.ai

- "인프라가 아닌, 코드에만 집중하세요."

Before AINP

- 복잡한 YAML 파일과 씨름으로 새로운 서비스 배포
- 인프라팀의 리소스 할당을 하염없이 기다림
- 배포 후 정상 작동 여부를 일일이 수동으로 확인

After AINP

- 웹 기반의 셀프서비스 포털에 로그인
- 미리 정의된 애플리케이션 템플릿을 선택하고 '배포' 버튼 클릭
- 배포 즉시 APM, 로그, 리소스 현황이 연동된 대시보드 확인

빠른 배포

클릭 몇 번으로 애플리케이션 배포, 기존에 걸렸 던 시간 단축

코드에만 집중

인프라 문제로 인한 개발 방해 없이 비즈니스로 직 개발에 집중

TTM 단축

Time to Market(TTM) 단축으로 경쟁 우위 확보

AINP (AI Native Platform): 개발자를 위한 완벽한 개발 환경

도입 효과 및 기대 성과

MSAP.ai

- AINP (AI Native Platform) 도입으로 기대되는 주요 효과와 변화

쿠버네티스 복잡성 해결

쿠버네티스 도입 및 운영의 복잡성을 해결합니다. 순수 쿠버네티스의 복잡성 을 추상화하고, 엔터프라이즈 환경에 최적화된 기능을 제공합니다.

개발&운영 생산성 향상

개발자와 운영자 모두의 생산성을 극대화합니다. 셀프서비스 포털을 통해 몇 번의 클릭만으로 필요한 애플리케이션을 신속하게 배포할 수 있습니다.

지능형 인프라 관리

AI 기반의 지능형 관리로 예측 가능하고 안정적인 서비스를 운영합니다. 지능 형 관리(MCP, CogentAI)로 인한 자동화된 운영 및 문제 예측/해결.

TCO 절감

기술 도입 및 운영에 따르는 총소유비용과 리스크를 최소화합니다. 쿠버네티스 전문가 확보의 어려움을 줄이고, 지속적인 운영 관리 비용을 감소시킵니다.

AINP (AI Native Platform)은 클라우드 네이티브의 진정한 가치를 실현하는 가장 확실한 방법입니다.

AINP (AI Native Platform) 이것만 기억해 주세요.

MSAP.ai

"AINP (AI Native Platform)은 클라우드 네이티브의 진정한 가치를 실현하는 가장 확실한 방법입니다."

AINP (AI Native Platform)의 핵심 가치

- ☑ 복잡성 해결: 쿠버네티스 도입 및 운영의 복잡성을 해결합니다.
- ☑ 생산성 향상: 개발자와 운영자 모두의 생산성을 극대화합니다.
- ☑ 안정적 운영: AI 기반의 지능형 관리로 예측 가능하고 안정적인 서비스를 운영합니다.
- ☑ TCO 절감: 기술 도입 및 운영에 따르는 총소유비용과 리스크를 최소화합니다.

차별화된 경쟁력

- ☑ 통합: 개발, 배포, 운영에 필요한 모든 도구와 기능을 하나로 통합합니다.
- ☑ 지능화: AI 기반의 지능형 관리로 운영을 자동화하고 예측 가능성을 높입니다.
- ☑ 효율성: 개발자와 운영자 모두의 생산성을 극대화하여 TCO를 절감합니다.

성공적인 클라우드 네이티브 전환을 위한 제안

AINP (AI Native Platform)과 함께 성공적인 클라우드 네이티브 전환을 시작하십시오.

검증된 플랫폼을 제안 합니다.

> AINP (AI Native Platform)은 쿠버네티스라는 강력한 엔진 위에 기업 환경에 필 수적인 모든 요소를 완벽하게 통합하여 제공하는 '완성형 플랫폼(Well- architected Platform)'입니다.

AINP를 애플리케이션 플랫폼으로 선택해야 하는 이유

MSAP.ai

통합 플랫폼

- 세션 관리 + APM + Observability를 하나의 플랫폼에서 제공

엔터프라이즈 적합성

- 대규모 서비스, 미션 크리티컬 시스템에 필수적인 고가용성 기능 내장

AI 기반 미래형 운영 지원

- LLM 기술로 단순 모니터링을 넘어, 예측과 판단을 지원

쿠버네티스 친화적 구조

- 클라우드 네이티브 환경에서의 확장성과 유연성을 보장

운영자 중심 설계

- IT 운영 인력의 반복 업무를 줄이고, 인사이트 중심의 운영을 가능

AINP (AI Native Platform)

지능형 쿠버네티스 통합 운영 플랫폼

Kubernetes

Session Clustering

APM

Observability

지능형 관리

클라우드 네이티브의 복잡성을 해결하고, 비즈니스 혁신을 가속화하는 가장 확실한 방법

MSAP.ai

AINP (AI Native Platform)란?

MSAP.ai

- AINP(AI Native Platform)기반 Kubernetes의 컨테이너 오케스트레이션 플랫폼으로, 효과적인 Kubernetes 클러스터 관리와 AINP특성 서비스의 원활한 통합을 제공합니다.

주요 역할

클러스터 리소스 관리

클러스터, 네임스페이스, 노드, 토폴로지, 워크로드, 네트워크, 저장소, 사용자 관리 등 클러스터 리소스의 전체적인 제어를 제공합니다.

AINP서비스 통합

Harbor Registry, GitLab, Jenkins, ArgoCD, Nexus, Observability, Cluster, APM 등 AINP생태계의 다양한 서비스와 자연스러운 상호작용을 가능합니다.

지능형 인프라 관리

AI 기반의 지능형 관리로 예측 가능하고 안정적인 서비스를 운영합니다. 지능 형 관리(MCP, CogentAI)로 인한 자동화된 운영 및 문제 예측/해결합니다.

DevOps 팀이 애플리케이션에만 집중

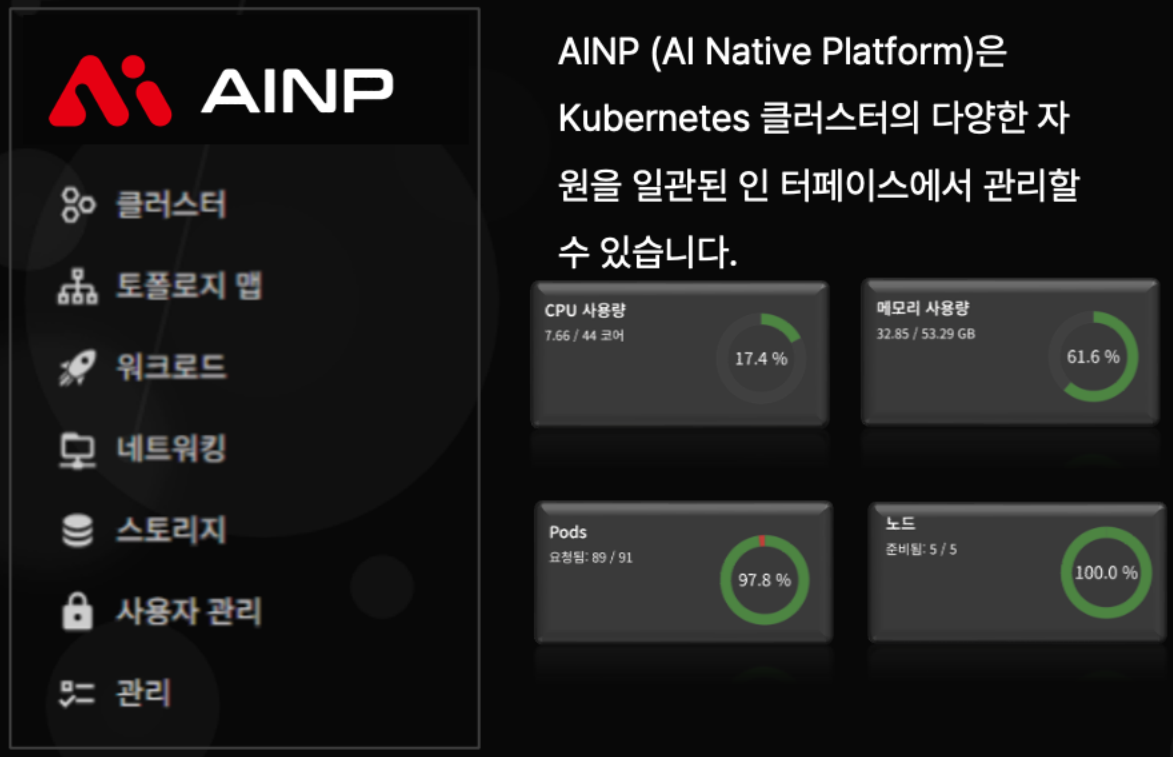
Kubernetes 대시보드 이상의 강력한 도구로, 운영자가 클러스터 자원 을 전체 제어&모니터링 하여 DevOps 팀이 애플리케이션 개발에만 집중 할 수 있습니다.

핵심 기능 개요

MSAP.ai

- AINP (AI Native Platform)은 두 가지 주요 기능으로 구성되어 있습니다. 이 두 기능은 Kubernetes 클러스터의 효과적인 관리와 AINP생태계의 서비스 통합을 가능하게 합니다.

Enterprise Kubernetes 자원 관리



AINP (AI Native Platform)은 Kubernetes 클러스터의 다양한 자원을 일관된 인 터페이스에서 관리할 수 있습니다.

Kubernetes 필수 서비스 통합

한 번의 클릭으로 Kubernetes 운영에 필수적인 다양한 서비스에 빠르게 접근할 수 있습니다.

- Keycloak SSO
- LDAP
- Harbor Registry
- Gitlab
- Jenkins
- ArgoCD
- Nexus
- Observability
- APM
- Session Cluster

Kubernetes 클러스터 관리 기능

MSAP.ai

- AINP (AI Native Platform)은 Kubernetes 클러스터의 다양한 자원을 통합적으로 관리하고 모니터링할 수 있도록 설계되었습니다. 클러스터 전반에 걸쳐 리소스, 워크로드, 네트워크 및 기타 구성 요소를 포괄적으로 제어할 수 있는 기능을 제공합니다.

리소스 관리

- 클러스터 및 네임스페이스 관리
- 노드 및 토폴로지
- 스토리지 관리
- 클러스터 업그레이드 및 롤백 자동화

워크로드 관리

- Pod 상태 및 배포 관리
- Deployment 및 ReplicaSet 제어
- StatefulSet 관리
- DaemonSet 관리

네트워크

- 네트워크 구성 및 정책 관리
- 자가 관리형 클러스터 설정
- 사용자 정의 리소스(CRD) 관리

보안 및 모니터링

- Role-Based Access Control(RBAC) 관리
- Secret 및 ConfigMap 관리
- 리소스 사용량 및 이벤트 실시간 모니터링
- 로그 수집 및 분석

네임스페이스 관리

MSAP.ai

Kubernetes 네임스페이스

네임스페이스는 Kubernetes에서 클러스터 리소스를 나누는 중요한 개념으로 AINP (AI Native Platform) 은 효율적으로 관리할 수 있도록 도구를 제공합니다.

네임스페이스 관리 기능

생성
필요한 네임스페이스를 쉽게 생성할 수 있습니다.

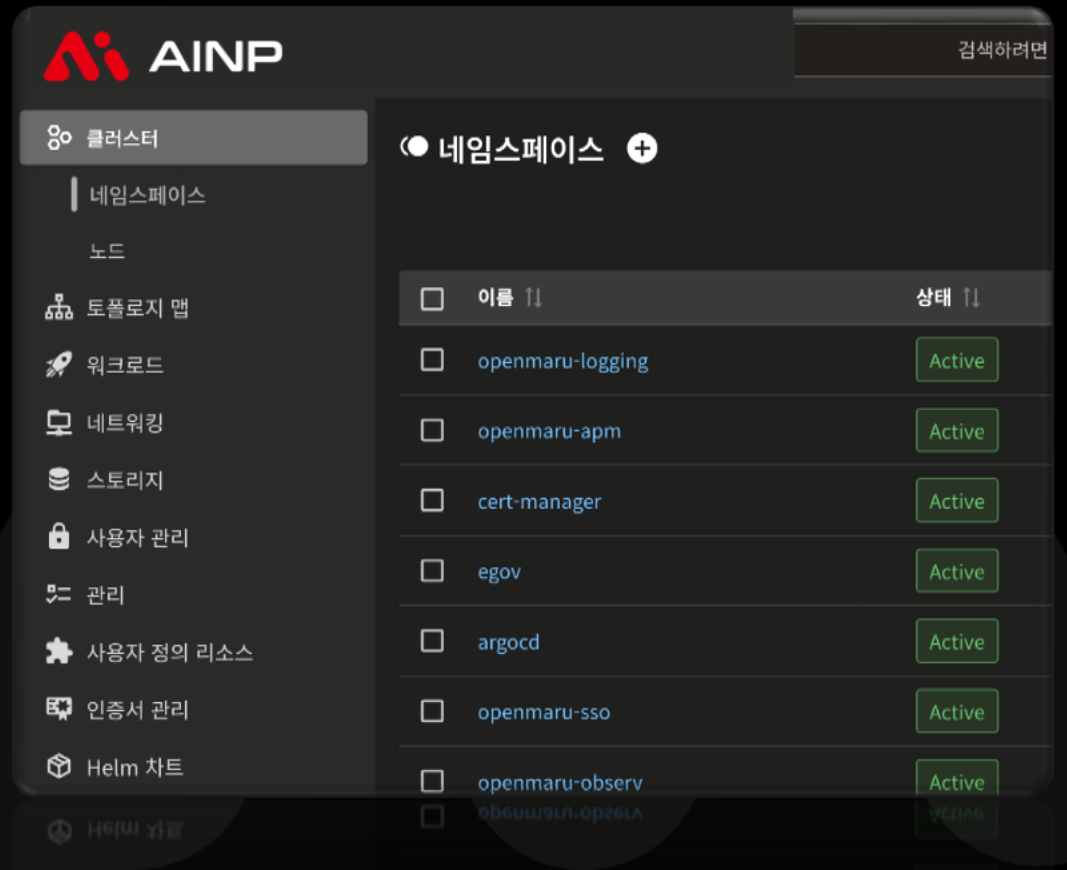
조회
클러스터의 모든 네임스페이스를 목록으로 확인할 수 있습니다.

수정
필요한 경우 네임스페이스의 설정을 수정할 수 있습니다.

삭제
불필요한 네임스페이스를 안전하게 삭제할 수 있습니다.

네임스페이스 리스트 뷰

AINP (AI Native Platform)은 네임스페이스의 리스트 뷰를 제공합니다. 이 뷰를 통해 사용자는 필터링 및 정렬 기능을 활용하여 필요한 정보를 빠르게 찾을 수 있습니다.



노드 관리

MSAP.ai

네임스페이스 관리 기능

노드 추가 및 삭제
클러스터에 새로운 노드 추가하거나 필요하지 않은 노드 삭제

노드 설정 수정
노드의 레이블, 테인트, 리소스 할당 등을 수정

노드 보안 관리
노드의 보안 설정 변경, 인증 및 권한 관리

노드 검색 및 필터링
특정 조건에 맞는 노드 검색, 정렬 및 그룹화

이벤트 알림
노드 상태 변경 및 중요한 이벤트에 대한 실시간 알림

노드 상태 모니터링

AINP (AI Native Platform)은 네임스페이스의 리스트 뷰를 제공합니다. 이 뷰를 통해 사용자는 필터링 및 정렬 기능을 활용하여 필요한 정보를 빠르게 찾을 수 있습니다.

이름	CPU	메모리	준비됨	Taints
oke-master-3	27%	44%	예	없음
oke-master-2	32%	64%	예	없음
oke-worker-1	9%	51%	예	없음
oke-worker-2	15%	81%	예	없음
oke-master-1	21%	62%	예	없음

클러스터 토폴로지

MSAP.ai

토폴로지 뷰 개요

AINP (AI Native Platform)의 토폴로지 뷰는 Kubernetes 클러스터의 구성 요소 간 관계를 시각적으로 표현합니다. 이 뷰를 통해 클러스터의 구조와 컴포넌트 간의 의존성을 쉽게 이해할 수 있습니다.

필터링

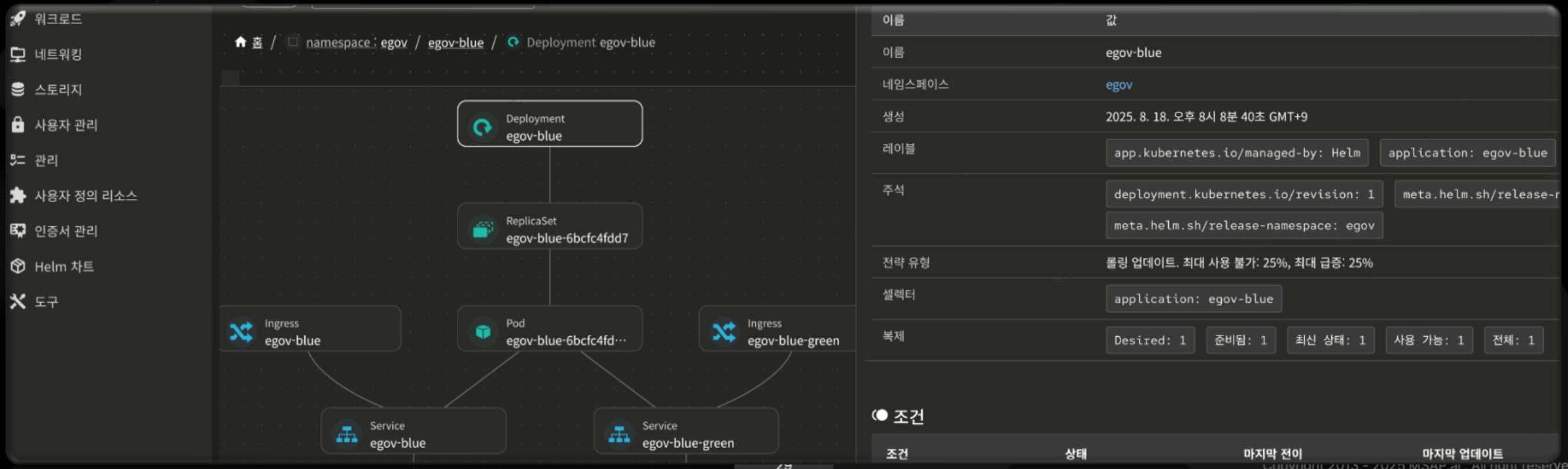
네임스페이스, 워크로드 유형, 상태 등을 기준으로 필터링 가능

그룹화

관련된 리소스들을 그룹으로 묶어서 보기 쉽게 구성

검색

특정 리소스를 빠르게 찾을 수 있는 검색 기능



워크로드 관리

MSAP.ai

- AINP (AI Native Platform)은 Kubernetes 워크로드 리소스를 포함한 클러스터 자원의 전체적인 관리를 제공합니다.

Pod 관리

- ✓ Pod 생성, 조회, 수정, 삭제
- ✓ Pod 상태 모니터링
- ✓ 로그 조회 및 스트리밍

Deployment 관리

- ✓ Deployment 생성, 조회, 수정, 삭제
- ✓ 배포 상태 및 히스토리 추적
- ✓ 롤링 업데이트 및 롤백

StatefulSet 관리

- ✓ StatefulSet 생성, 조회, 수정, 삭제
- ✓ 스테이트풀 상태 및 멤버십 관리
- ✓ 오더링 및 고유성 유지

DaemonSet 관리

- ✓ DaemonSet 생성, 조회, 수정, 삭제
- ✓ 데몬셋 상태 및 업데이트 관리
- ✓ 노드별 인스턴스 배포 및 모니터링

스토리지 관리

Kubernetes 스토리지 리소스 관리

AINP (AI Native Platform)은 Kubernetes 스토리지 리소스의 전체적인 관리를 제공합니다. 퍼시스턴트 볼륨(PV)과 스토리지 클래스 (StorageClass) 등을 통해 애플리케이션의 데이터 영속성을 효과적으로 관리할 수 있습니다.

퍼시스턴트 볼륨(PV)

- ✓ 클러스터의 물리적 저장소를 추상화
- ✓ PersistentVolumeClaim(PVC)과 바인딩
- ✓ 스토리지의 상태 및 용량 관리

스토리지 클래스(StorageClass)

- ✓ 동적 프로비저닝을 위한 스토리지 템플릿
- ✓ 지속성 및 복제 속성 정의
- ✓ 클레임에 대한 자동 바인딩 제공

스토리지 관리 작업

스토리지 클래스 생성 | 볼륨 스냅샷 | 볼륨 바인딩 관리 | 스토리지 클래스 수정 | 볼륨 확장 | 스토리지 리소스 삭제

네트워크 관리

MSAP.ai

- AINP (AI Native Platform)은 Kubernetes 네트워크 리소스의 전체적인 관리를 제공합니다. 서비스, 인그레스, 서비스 앤드포인트, 네트워크 정책 등을 포함한 클러스터의 네트워크 구성 요소를 효과적으로 관리하고 모니터링 할 수 있습니다.

서비스 관리

클러스터 내 서비스의 생성, 수정, 삭제 및 목록 조회를 제공합니다. 서비 스 타입별로(ClusterIP, NodePort, LoadBalancer, ExternalName) 다 양한 서비스 배포 방식을 지원합니다.

인그레스 컨트롤

인그레스 리소스의 생성, 수정, 삭제 및 목록 조회 기능을 제공합니다. 도 메인 이름 기반의 서비스 라우팅, TLS/SSL 종료, 그리고 서비스 의 가중치 기반 로드 밸런싱을 구성할 수 있습니다.

네트워크 정책

네트워크 정책의 생성, 수정, 삭제 및 목록 조회를 제공합니다. 파드 간 통 신을 제어하고, 특정 IP 주소 또는 CIDR 블록에 대한 접근을 허용 하거나 차단하는 등의 네트워크 보안 규칙을 정의할 수 있습니다.

서비스 앤드포인트

서비스 앤드포인트의 목록 조회 및 세부 정보 확인 기능을 제공합니 다. 이 는 서비스가 실제로 연결된 백엔드 파드들의 집합을 나타냅니 다. 서비스의 가용성 및 연결 상태를 진단하는 데 유용합니다.

AINP (AI Native Platform)은 Kubernetes 네트워크 리소스의 선언적 관리 방식을 따르며, 필요한 경우 자동으로 클러스터의 네트워크 구성 요소를 생성, 수정 또는 삭제합니다.

사용자 관리

MSAP.ai

- AINP (AI Native Platform)은 클러스터 리소스 관리의 일부로, 사용자 계정 및 권한 관리 기능을 제공합니다. 이 기능을 통해 클러스터에 대한 접근을 제 어하고, 사용자의 역할에 따른 권한을 설정할 수 있습니다.

사용자 역할 관리

사용자를 생성하고 수정하며 삭제할 수 있습니다. 또한, 클러스터 및 네임스페이스 수준의 역할을 할당하여 적절한 권한을 설정합 니다.

그룹 및 정책

사용자를 그룹으로 묶고, 그룹별로 일관된 권한 정책을 적용할 수 있습니다. 이 기능은 조직의 계층적 구조에 따라 권한을 관리하는 데 유용합니다.

API 토큰 관리

사용자를 위한 API 토큰을 생성하고 관리할 수 있습니다. 이 토큰 들은 프로그래밍 방식으로 클러스터에 액세스하거나, 외부 애플리케이션과의 통합에 사용될 수 있습니다.

활동 로그 및 감사

사용자의 활동을 로깅하고 감사할 수 있습니다. 이 기능은 보안 목 적으로도 사용되며, 누가 언제 어떤 작업을 수행했는지 추적하는 데 도움이 됩니다.

커스텀 리소스(CRD) 관리

MSAP.ai

커스텀 리소스 정의

Kubernetes의 Custom Resource Definition (CRD)는 클러스터에 새 종류의 리소스를 추가할 수 있게 해줍니다. AINP (AI Native Platform)은 이 러한 커스텀 리소스의 생성, 조회, 수정, 삭제 및 필터링 기능을 제공합니다.

관리 기능

생성 및 수정

사용자는 AINP (AI Native Platform)을 통해 새로운 CRD를 생성하거나 기존 CRD를 수정할 수 있습니다. 이는 Kubernetes API를 직접 호출하지 않고도 사용자 인터페이스를 통해 쉽게 수행할 수 있습니다.

검색 및 필터링

제공된 검색 및 필터링 기능을 통해 사용자는 특정 CRD를 빠르게 찾을 수 있습니다. 이는 클러스터에 많은 CRD가 존재하는 경우에 특히 유용 합니다.

삭제 기능

더 이상 필요하지 않은 CRD를 안전하게 삭제할 수 있습니다. 이 기능은 클러스터 리소스의 깨끗한 관리와 유지보수에 도움이 됩니다.

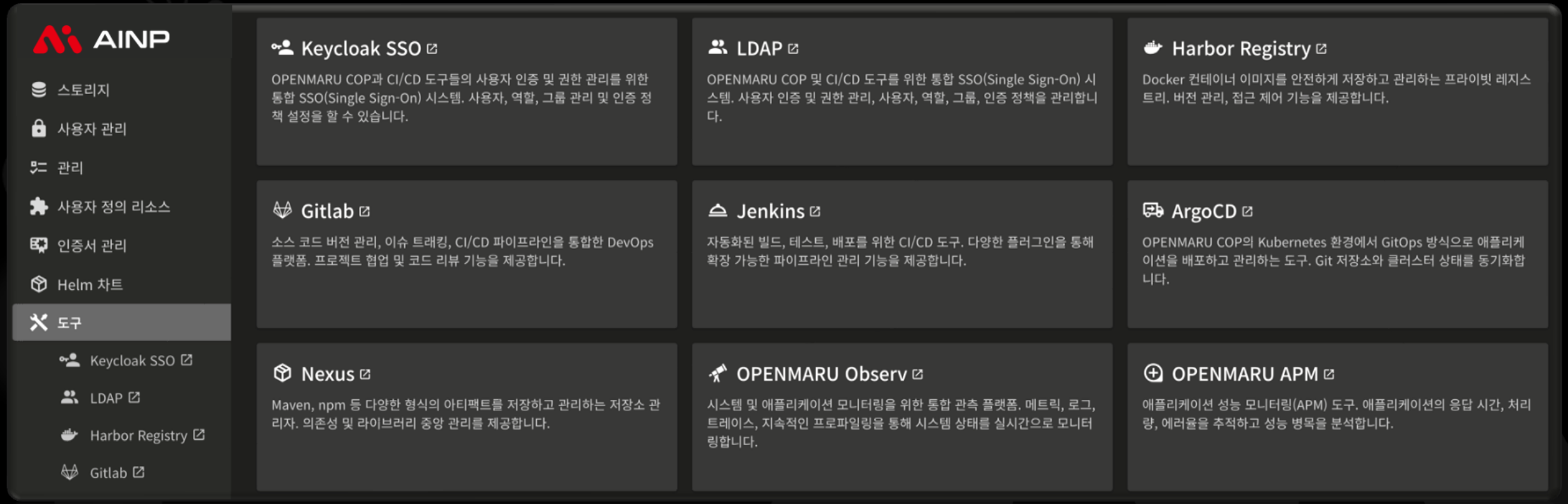
목록 및 정렬

생성된 모든 CRD의 목록을 확인할 수 있으며, 이를 이름, 생성 날짜, 상태 등으로 정렬할 수 있습니다. 이는 CRD의 효과적인 관리와 모니터 링에 도움이 됩니다.

AINP서비스 통합

MSAP.ai

- AINP (AI Native Platform)은 사이드바 메뉴를 통해 클러스터 관리 기능 외에도 다양한 AINP서비스에 원활하게 접근할 수 있는 통합 인터페이스 를 제공합니다. 이를 통해 한 플랫폼에서 여러 서비스를 일관된 방식으로 관리할 수 있습니다.



Harbor Registry 연동

MSAP.ai

- AINP (AI Native Platform)은 사이드바 메뉴를 통해 Harbor Registry와의 원활한 연동을 제공합니다. 이 통합은 컨테이너 이미지 저장소에 대한 접근 및 관리를 간편하게 만들어줍니다.

쿠버네티스 기반 인프라 특징

사이트바 메뉴 접근

AINP (AI Native Platform) 사이드바에서 Harbor Registry에 접근 가능

이미지 검색

검색 기능을 통해 Harbor Registry의 이미지 검색

자동 동기화

클러스터와 Harbor Registry 간의 자동 동기화

AINP (AI Native Platform)의 향상된 인프라 관리

효율적인 이미지 관리

단일 인터페이스에서 이미지 전체 생명주기를 관리

보안 향상

통합된 보안 정책 및 접근 제어

워크로드 통합

클러스터 워크로드와의 자연스러운 연동

GitLab 연동

MSAP.ai

소스 코드 관리 도구 통합

AINP (AI Native Platform)은 GitLab과의 자연스러운 연동을 제공하여, 개발 프로세스의 시작점에서 Kubernetes 클러스터 관리까지 일관된 인터페이스로 접근 가능합니다.

사이드바 메뉴 통합

AINP (AI Native Platform)의 사이드바 메뉴를 통해 GitLab에 바로 접근할 수 있 습니다. 클릭 한 번으로 개발 코드 저장소에 즉시 접속 가능합니다.

보안 및 접근 제어

AINP (AI Native Platform)의 인증 시스템은 GitLab과 연동되어, 동일한 인증 정 보로 두 플랫폼 모두 접근 가능합니다. 권한 관리도 일관되게 이루어집니 다.

개발-배포 파이프라인

GitLab의 저장소와 Kubernetes 클러스터를 연동하면, 코드 커밋부터 배포까지의 전체 개발 생명주기를 관리할 수 있습니다.

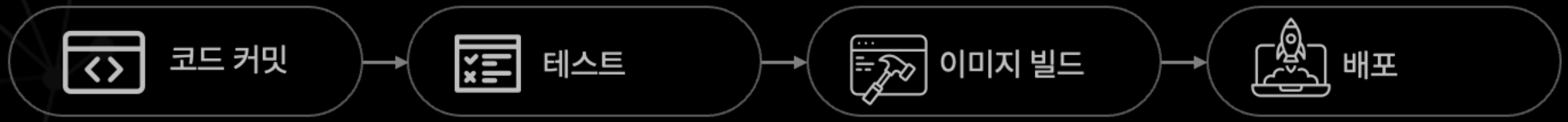
프로젝트 통합

AINP (AI Native Platform)에서 실행 중인 애플리케이션과 GitLab의 프로젝트를 연동하면, 코드 변경 내용을 실시간으로 추적하고, 배포된 애플리케이션 의 소스 코드를 즉시 확인할 수 있습니다.

Jenkins 연동

Jenkins와의 통합

Jenkins는 AINP (AI Native Platform) 사이드바 메뉴를 통해 쉽게 접근할 수 있는 CI/CD 도구입니다. AINP (AI Native Platform)과의 연동을 통해 자동화 파 이프라인을 구축하고, 애플리케이션 배포 흐름를 자동화할 수 있습니다.



Jenkins 와 통합

- ✓ AINP (AI Native Platform)의 사이드바 메뉴를 통해 Jenkins에 바로 접근
- ✓ 클러스터 리소스와 Jenkins 작업 사이의 통합
- ✓ 일관된 인터페이스로 두 플랫폼 간 작업 전환

AINP서비스 통합

- ✓ 개발-배포 파이프라인의 일관성 유지
- ✓ 클러스터 관리와 CI/CD 프로세스의 자연스러운 결합
- ✓ AINP생태계의 다른 서비스들과의 원활한 통합

ArgoCD 연동 – AINP (AI Native Platform) 통합의 가치

MSAP.ai

GitOps 도구 ArgoCD

ArgoCD는 선언형 GitOps 를 통해 애플리케이션 배포를 자동화하는 오픈소스 도구입니다. AINP (AI Native Platform)은 ArgoCD와의 원활한 연동 을 제공하여 개발 및 운영 흐름을 스트레스리스하게 연결합니다.

AINP (AI Native Platform) 통합

AINP (AI Native Platform)의 사이드 메뉴에서 ArgoCD에 손쉽게 접근할 수 있으며, 개발자가 코드를 커밋하면 자동으로 배포되는 CI/CD 파이프라인을 설정할 수 있습니다. 이로써 애플리케이션 운영 환경이 한층 단순화되고 안정화됩니다.

실시간 동기화

ArgoCD는 Git 저장소의 변경 사항을 실시간으로 감지해 Kubernetes 클러스터와 자동 동기화합니다. 이를 통해 인프라의 '선언된 상태'와 '실제 동작 상태'가 항상 일치하도록 유지할 수 있습니다.

ArgoCD 와 통합을 통한 주요 이점

- 코드 저장소에서 직접 애플리케이션을 배포 및 관리
- 변경 이력 추적과 롤백을 통한 안정성 확보
- 간편한 애플리케이션 롤아웃과 스케일링
- 권한 기반 접근 제어로 보안성 강화
- 클러스터 간 애플리케이션 동기화 지원
- 애플리케이션 상태를 실시간으로 모니터링

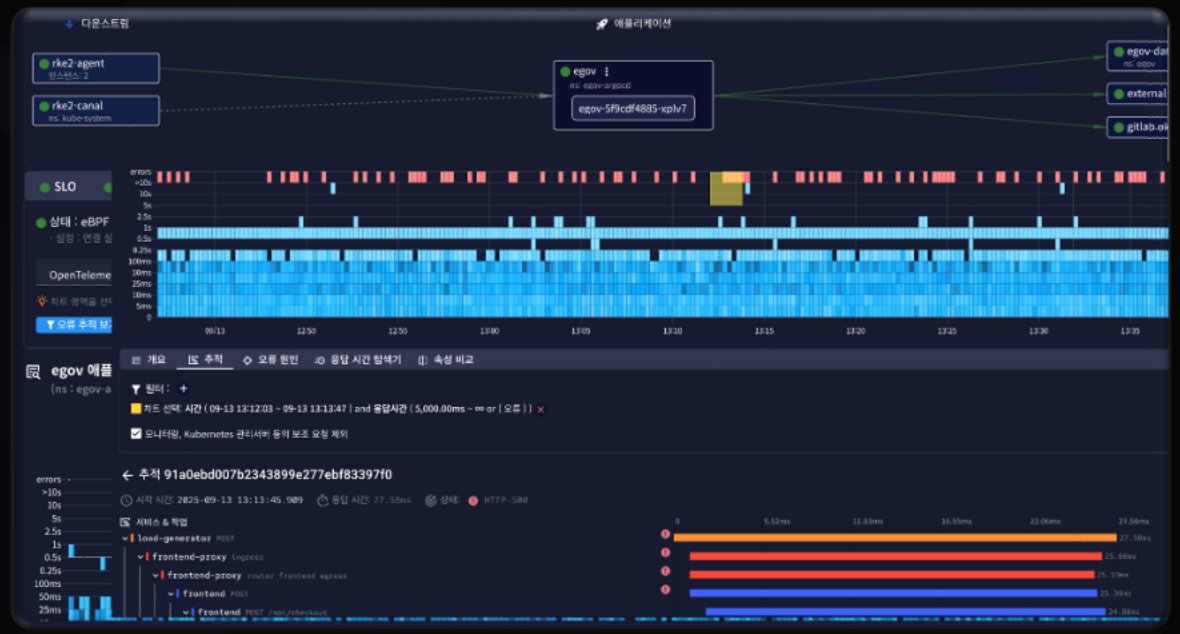
모니터링과 옵저버빌리티

MSAP.ai

- AINP (AI Native Platform)은 사이드바 메뉴를 통해 AINP생태계의 모니터링 도구에 접근할 수 있습니다. 이 통합은 클러스터 상태 및 애플리케이션 성능을 실시간으로 모니터링하고 분석하는 데 도움이 됩니다.

AINPObservability

클러스터 및 애플리케이션의 실시간 상태를 모니터링하고, 시스템 성능 을 차트 및 대시보드로 시각화합니다.



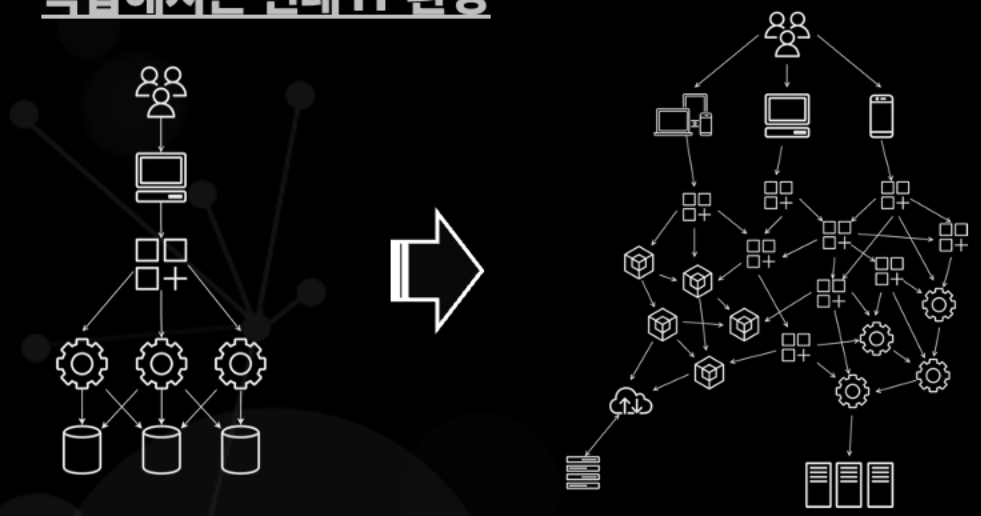
AINPAPM

애플리케이션 성능을 관리하고, 트랜잭션 추적 및 애플리케이션 레벨의 진단 도구를 제공합니다.

서론: 복잡해지는 IT 환경과 VibeOps의 필요성

MSAP.ai

복잡해지는 현대 IT 환경



클라우드 네이티브 아키텍처

다양한 클라우드 서비스와 API로 인한 복잡성 증가

마이크로서비스(MSA)

수십, 수백 개의 서비스가 유기적으로 연결되어 운영 복잡성 가중

쿠버네티스

컨테이너 오케스트레이션을 통한 인프라 확장성 확보와 함께 운영 난이도 상승

모니터링 데이터 관리 어려움

방대한 데이터 관리의 어려움

로그와 메트릭, 이벤트 데이터가 폭증하며 단순 수집·보관만으로는 가치를 발휘하기 어렵습니다.

주요 도전

시스템 안정성을 유지하고 장애나 이상 징후 발생 시 빠르게 대응하는 것이 IT 운영팀의 핵심 과제가 되었습니다.

VibeOps의 등장

2016년 가트너가 AIOps를 제시하며 IT 운영 패러다임에 변화를 예고했듯, **VibeOps는 그 한계를 넘어선 진화된 모델**입니다.AIOps가 자동화된 데이터 분석에 초점을 맞췄다면, VibeOps는 운영자의 의도와 시스템의 맥락을 결합해 **더 빠르고 더 지능적인 의사결정을 가능**하게 합니다. 이는 단순한 운영 효율화를 넘어, IT 운영 방식을 근본적으로 혁신하는 새로운 접근법입니다.

VibeOps는 **복잡해진 IT 환경 속에서 운영팀이 안정성을 확보하고 문제 대응 속도를 높이는 동시에, 운영 경험 자체를 지능적으로 진화시키는 차세대 운영 모델**이라 할 수 있습니다.

VibeOps란 무엇인가?

MSAP.ai

- VibeOps는 기존 AIOps에 LLM의 언어 이해와 생성 능력을 결합하여, IT 운영 데이터를 맥락적으로 분석하고 문제 해결까지 지원하는 차세대 인공지능 기반 운영 체계입니다.

맥락 이해

IT 운영 데이터를 단순 수치로만 보는 것이 아니라, 인간의 언어로 설명하고 의미를 해석합니다. 로그 메시지, 장애 보고서, 운영 문서 등을 자유롭게 이해할 수 있습니다.

자연어 소통

대화형 인터페이스(챗봇)를 통해 운영자는 복잡한 쿼리 없이도 자연스러운 질문만으로 원하는 분석 결과를 즉시 받아볼 수 있습니다.

인과관계 추론

방대한 학습 데이터를 기반으로 '원인–결과'의 연결고리를 도출합니다. 이를 통해 문제 해결 시간을 획기적으로 단축시킵니다.

예시: 기존 AIOps vs VibeOps

기존 AIOps:

"결제 서비스에서 에러율 5 % 증가"

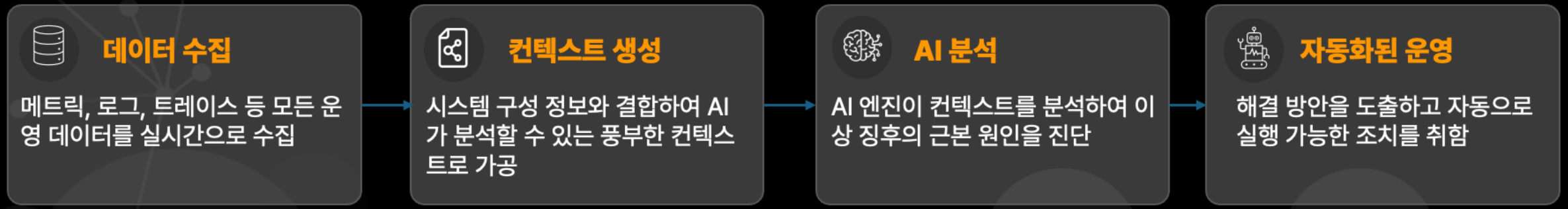
VibeOps :

"30분 전 '결제 서비스'의 새로운 버전이 배포되었습니다. 이후 데이터베이스 연결 시간 초과(timeout) 관련 에러 로그가 급증하기 시작했습니다. 최근 변경된 코드 중 데이터베이스 커넥션 풀 설정이 의심되며, 이전 버전으로 롤백하거나 관련 설정 값을 확인하는 것을 권장합니다."

LLM(Large Language Model) 을 통한 지능형 관리

MSAP.ai

- LLM을 통해 시스템의 상태 데이터(메트릭, 로그 등)와 구성 정보(모델)를 이해하고 분석 합니다. AINP (AI Native Platform)은 이 프로토콜을 통해 쿠버네티스 운영에 지능을 더합니다.

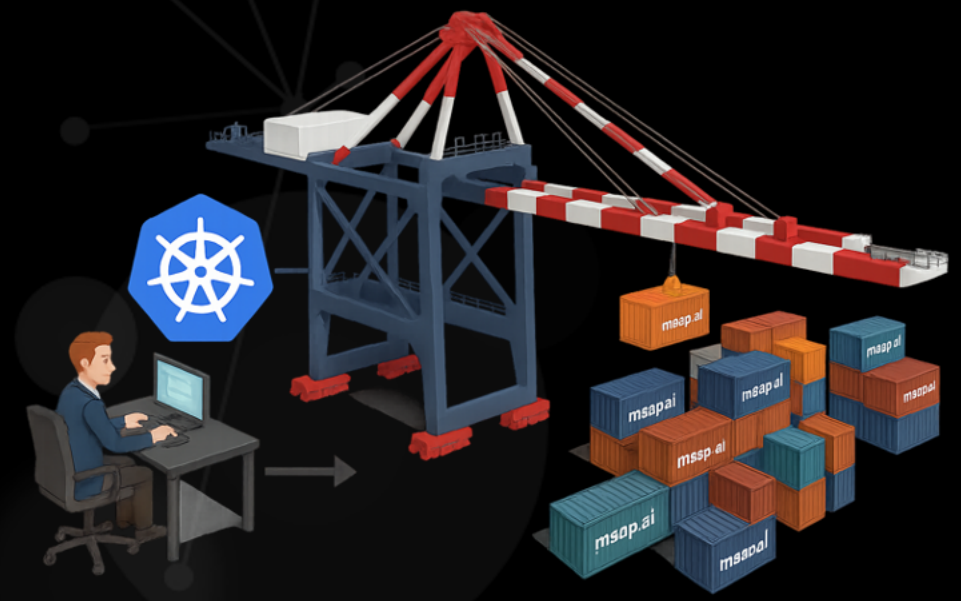


★단순 자동화를 넘어 지능형 운영(VibeOps)으로의 진화

- 정해진 규칙에 따른 자동화(Automation)를 넘어, AI가 스스로 상황을 판 단합니다.
- 시스템의 복잡한 상태를 종합적으로 분석하여 최적의 해결책을 제안합니다.
- 반복적인 문제에 대해 학습하여 지속적인 개선을 통해 운영 효율성을 향상 합니다.
- 예측 가능한 운영으로 서비스의 안정성(SRE)을 크게 강화합니다.

Cloud Native Platform

Kubernetes 특징과 컨테이너 과제에 대한 해결책



Kubernetes 특징

- 여러 컨테이너 노드를 중앙에서 통합관리
- Pod 라이프 사이클 모니링, 시작 / 정지
- Pod 네트워크 설정
- Pod 스토리지 설정
- 부하에 따른 자동 확장/축소

컨테이너의 과제	Kubernetes 의한 해결
컨테이너 배포	• 설정에 따라 컨테이너를 자동 배치, 배포 • Pod 단위로 자동 IP 주소 부여 및 연결 경로 설정 • 영구 저장소 연결
자동 장애 복구	• 자동 복구에 의한 가용성 확보 • 설정에 따라 POD 복제를 일정하게 유지 • 노드 장애와 지정 수량에 맞게 자동으로 복구
자동 부하	• 업무 절정의 스케일 조작과 자원 관리의 부하 경감 • 접속량을 모니터링하여 부하에 따라 자동으로 추가 배치

Development Process

WATERFALL

AGILE

DEVOPS

Application Architecture

MONOLITHIC

N-TIER

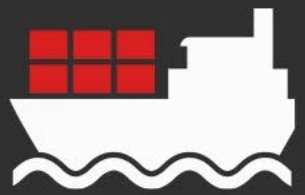
MICROSERVICES

Deployment & Packaging

PHYSICAL SERVERS

VIRTUAL SERVERS

CONTAINERS



Application Infrastructure

DATA CENTER

HOSTED

CLOUD

Container Market Research

MSAP.ai

2022 년까지 글로벌 조직의 **75 %** 이상이 프로덕션 환경에서 **컨테이너화** 된 애플리케이션을 실행할 것으로 예상합니다.

— Gartner

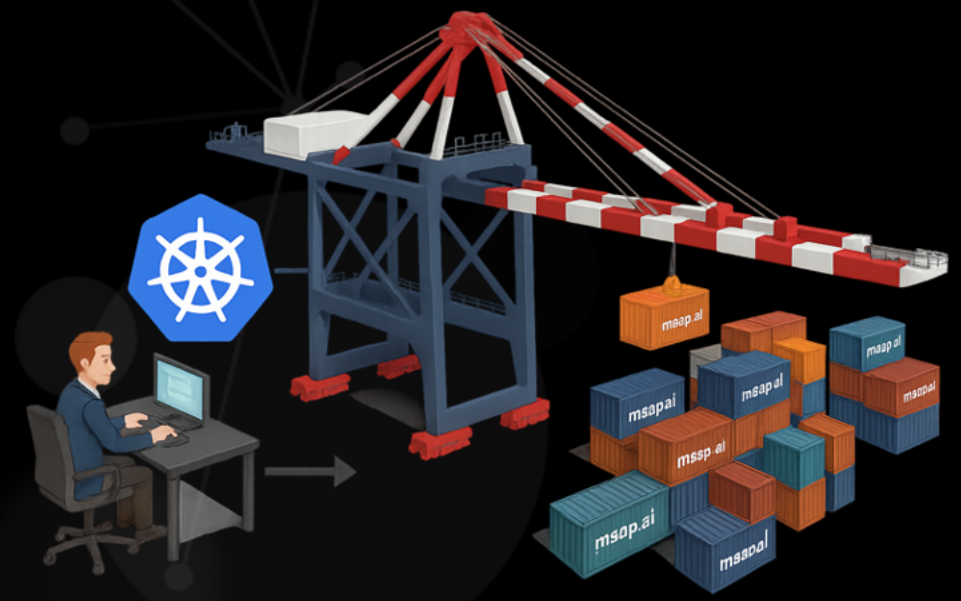
컨테이너가 가상머신을 대체하기 시작했습니다. Kubernetes 와 마이크로 서비스의 사용이 가속화되면서 **클라우드 네이티브가 프로덕션에 적용**되고 있습니다.

— 451 Research

컨테이너 채택은 프라이빗 과 퍼블릭 클라우드 인프라 간의 격차를 해소하여 **하이브리드 클라우드 아키텍처를 재정의** 할 것입니다.

— Forbes

Kubernetes 특징과 컨테이너 과제에 대한 해결책



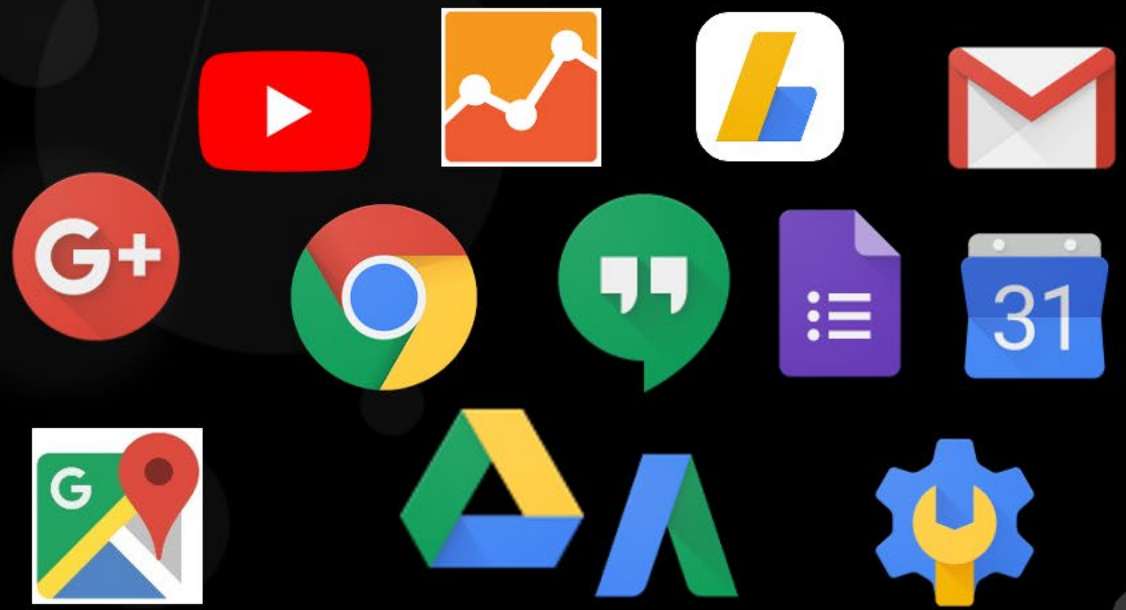
Kubernetes 특징

- 여러 컨테이너 노드를 중앙에서 통합관리
- Pod 라이프 사이클 모니링, 시작 / 정지
- Pod 네트워크 설정
- Pod 스토리지 설정
- 부하에 따른 자동 확장/축소

컨테이너의 과제	Kubernetes 의한 해결
컨테이너 배포	• 설정에 따라 컨테이너를 자동 배치, 배포 • Pod 단위로 자동 IP 주소 부여 및 연결 경로 설정 • 영구 저장소 연결
자동 장애 복구	• 자동 복구에 의한 가용성 확보 • 설정에 따라 POD 복제를 일정하게 유지 • 노드 장애와 지정 수량에 맞게 자동으로 복구
자동 부하	• 업무 절정의 스케일 조작과 자원 관리의 부하 경감 • 접속량을 모니터링하여 부하에 따라 자동으로 추가 배치

Google 는 모두 컨테이너에서 실행

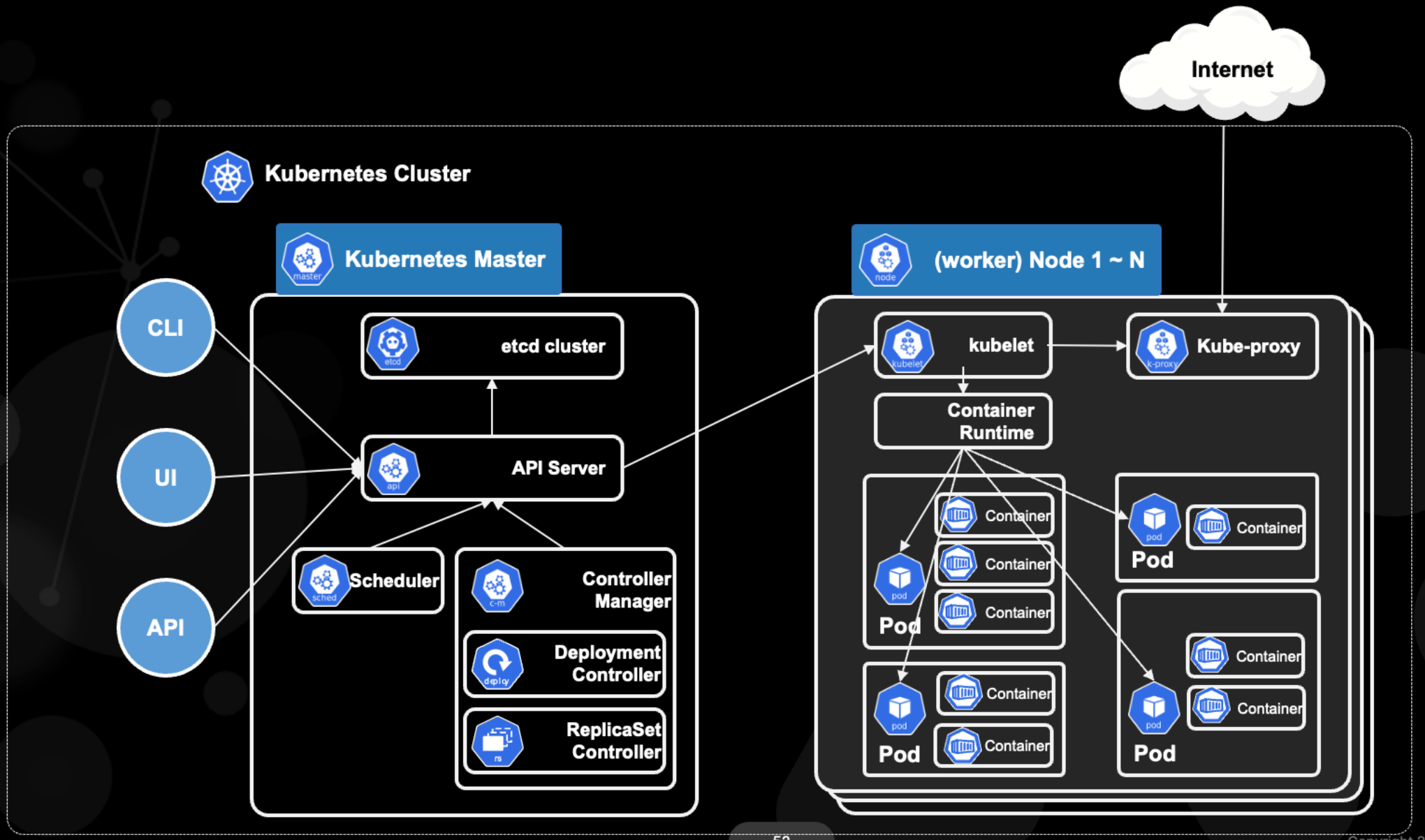
- Gmail , 검색, 지도 ...
- MapReduce , GFS , Colossus ...
- Google Compute Engine 가상 머신도 컨테이너에서 실행!
- 매주 20 억개 이상의 컨테이너를 실행 중



MSAP.ai

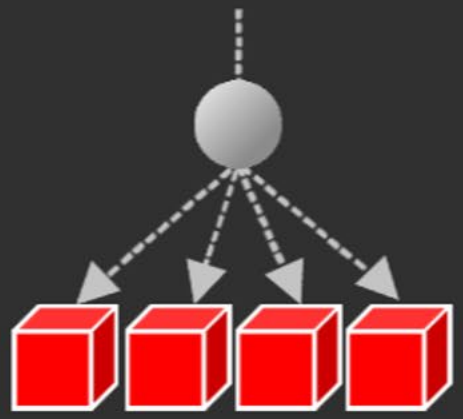
AINP (AI Native Platform) & Kubernetes

Kubernetes Architecture

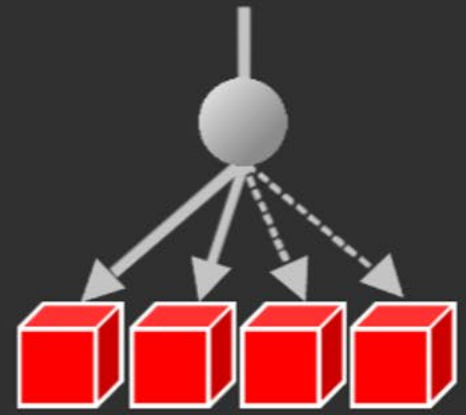


Auto Scale Out / In

Load Balancer

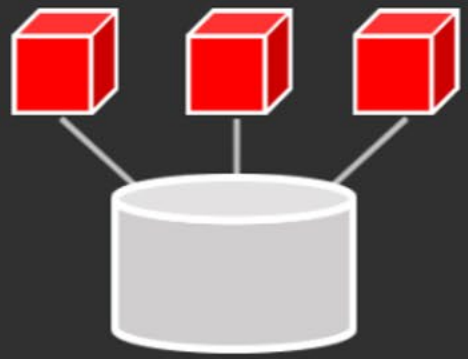


Auto Rolling Update

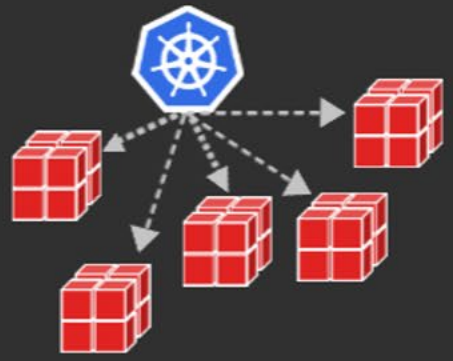


Auto Healing

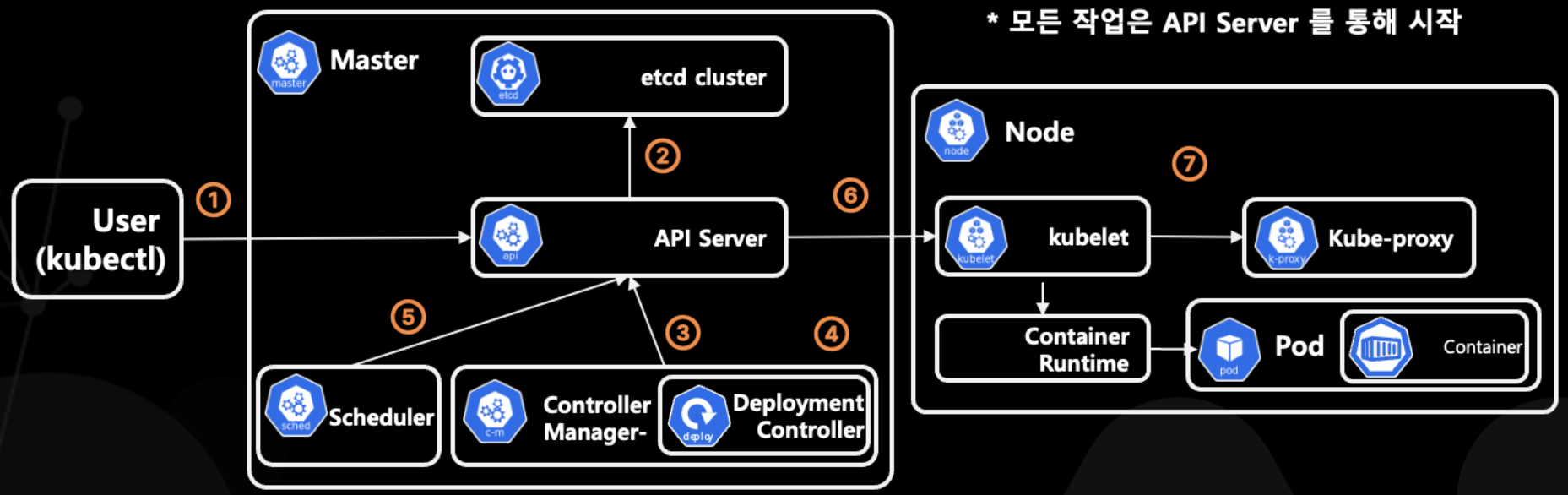
Persistence Volume



Container Orchestration



Kubernetes Architecture



① 사용자는 "kubectl"를 사용하여 컨테이너 배치에 필요한 정보 (컨테이너 이미지 개수 등)를 전달

② API Server는 받은 내용을 etcd cluster (DB)에 저장

③ Controller는 자원의 변화를 감지하면 변경 내용에 상태가 되도록 실행

④ 배포 일 경우 Deployment Controller는 새로운 Pod 정보를 API Server를 통해 DB (etcd)에 저장

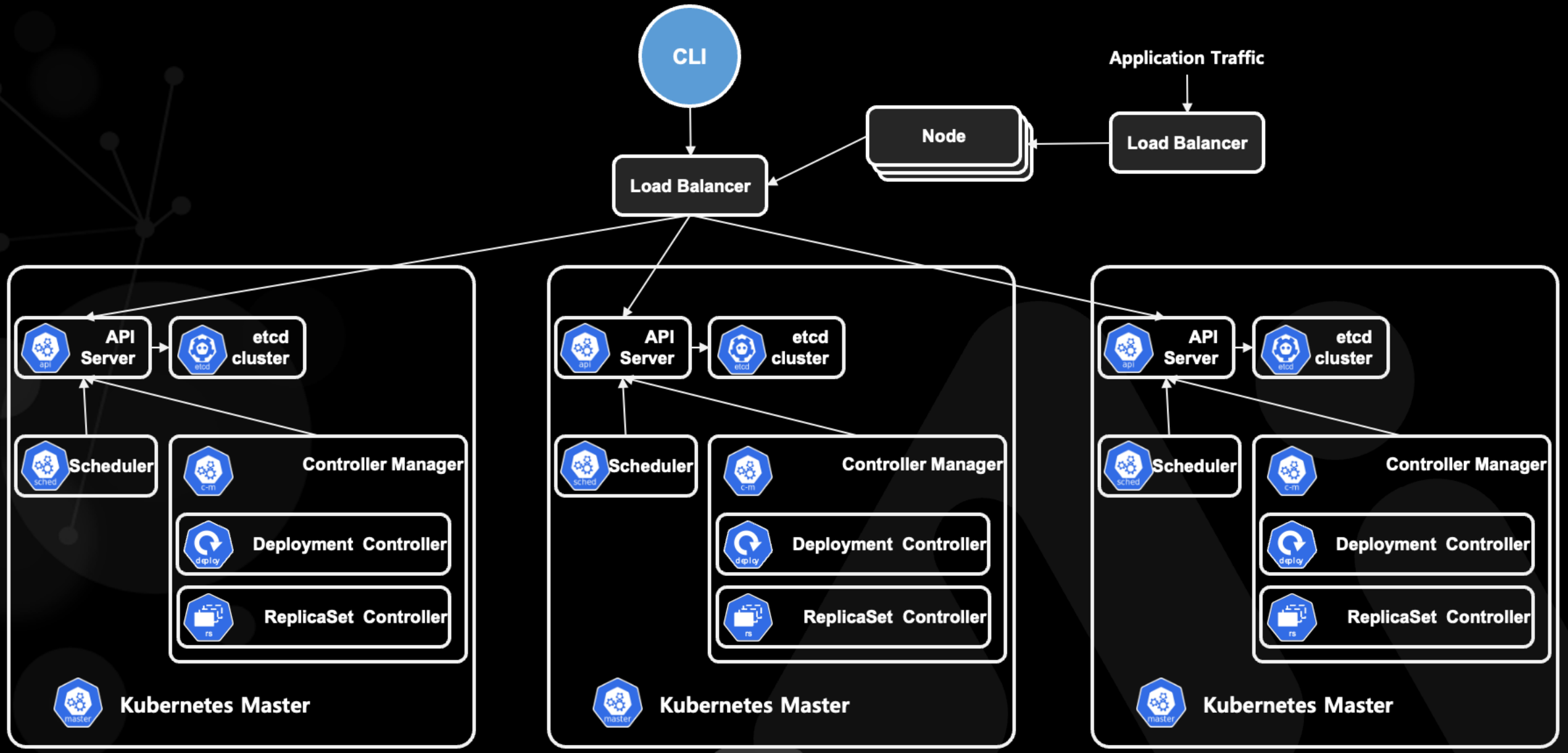
⑤ Scheduler는 정보에 따라 해당 Node 결정

⑥ Kublet 는 Container Runtime 를 사용하여 Pod를 생성

⑦ Kube-Proxy는 클러스터의 내부 또는 외부에서 Pod에 대한 연결을 포워딩

Kubernetes Master 고가용 구성

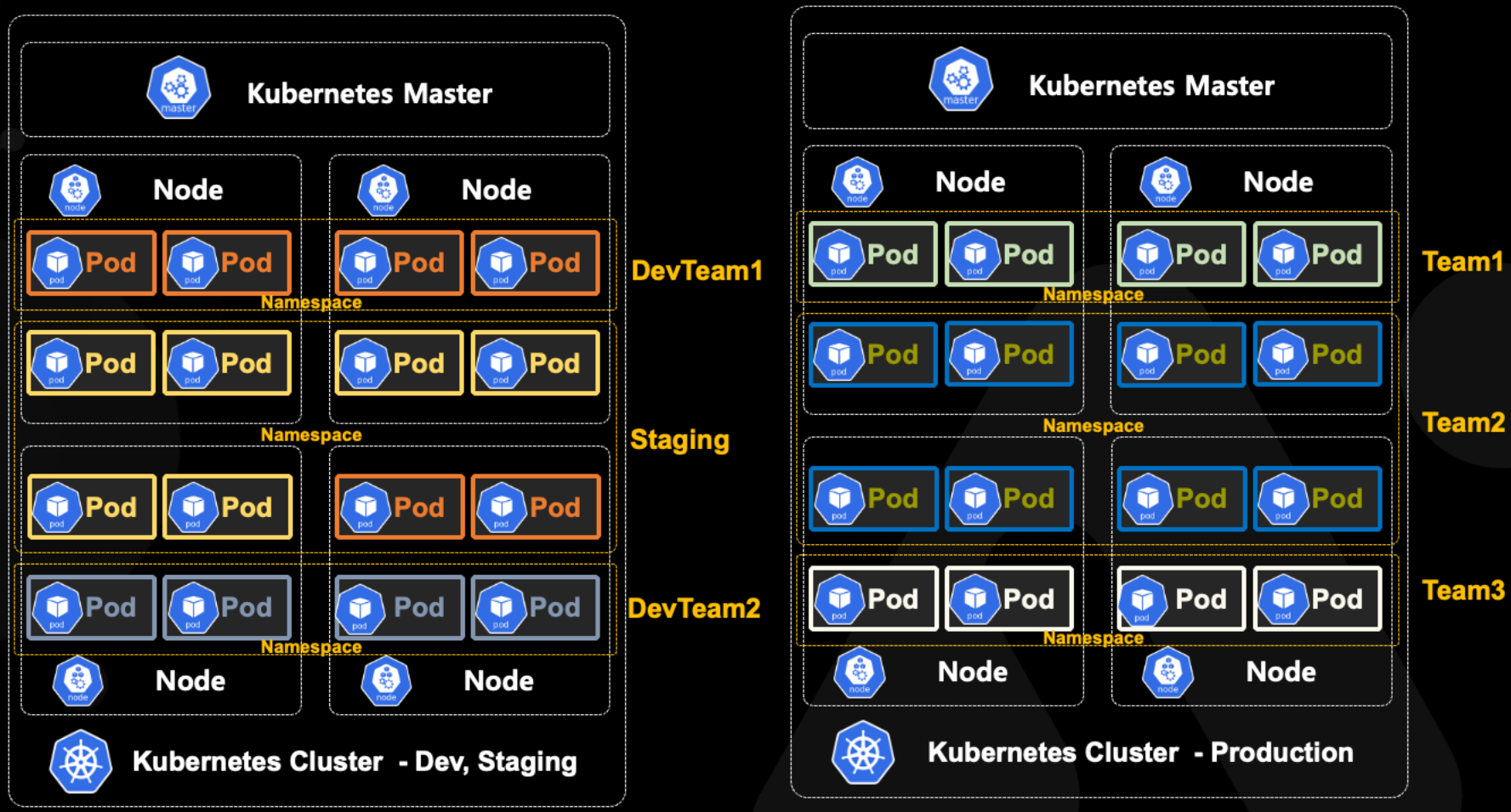
MSAP.ai



Kubernetes – 업무별 구분

MSAP.ai

- Kubernetes Cluster 단위



Google & Kubernetes

GOOGLE 과 컨테이너

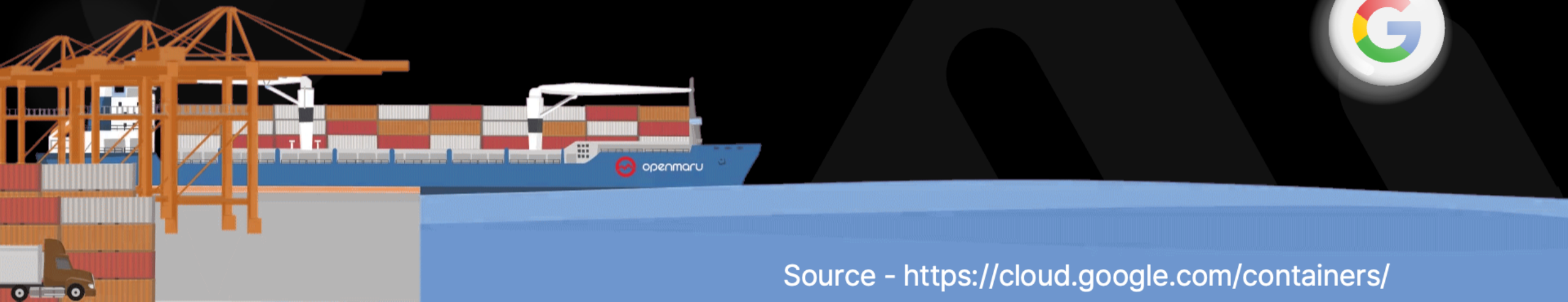
MSAP.ai

- Google의 업무 방식

Gmail에서 YouTube, 검색에 이르기까지 Google의 모든 제품은 컨테이너에서 실행됩니다.

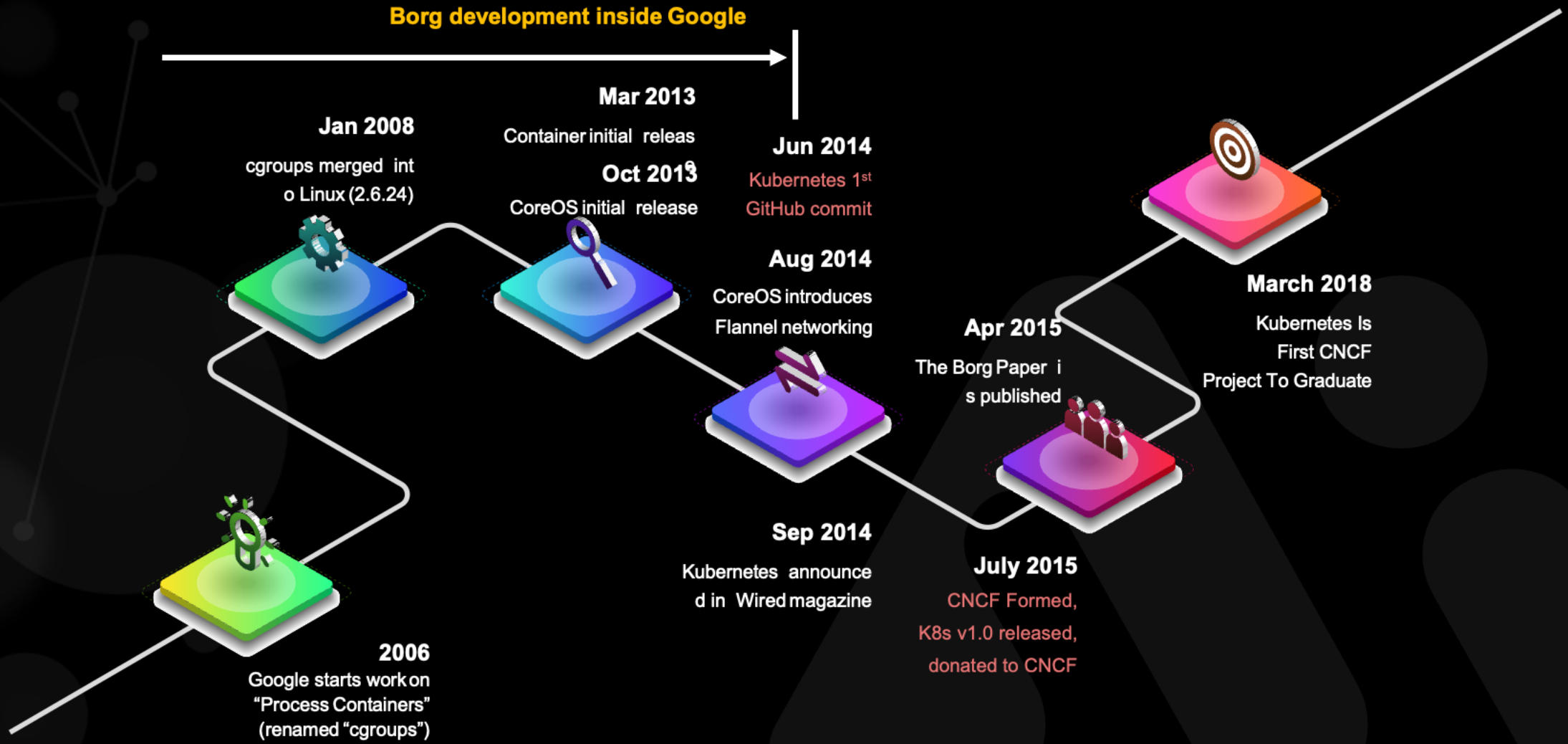
개발팀은 컨테이너화를 통해 더욱 신속하게 움직이고, 효율적으로 소프트웨어를 배포하며 전례 없는 수준의 확장성을 확보할 수 있게 되었습니다. Google은 매주 수십억 개가 넘는 컨테이너를 생성합니다. 지난 10여 년간 프로덕션 환경에서 컨테이너화된 워크로드를 실행하는 방법에 관해 많은 경험을 쌓으면서 Google은 커뮤니티에 계속 이 지식을 공유해 왔습니다.

초창기에 cgroup 기능을 Linux 커널에 제공한 것부터 내부 도구의 설계 소스를 Kubernetes 프로젝트로 공개한 것까지 공유의 사례는 다양합니다. 그리고 이 전문 지식을 Google Cloud Platform으로 구현하여 개발자와 크고 작은 규모의 회사가 최신의 컨테이너 혁신 기술을 쉽게 활용할 수 있도록 하였습니다.



Source - https://cloud.google.com/containers/

Kubernetes History



Kubernetes는 어디서나 사용할 수 있습니다.

MSAP.ai

AINP (AI Native Platform) 이식성

Kubernetes 도입에 따른 인프라, 개발 환경, 조직 및 보안 체계의 주요 변화

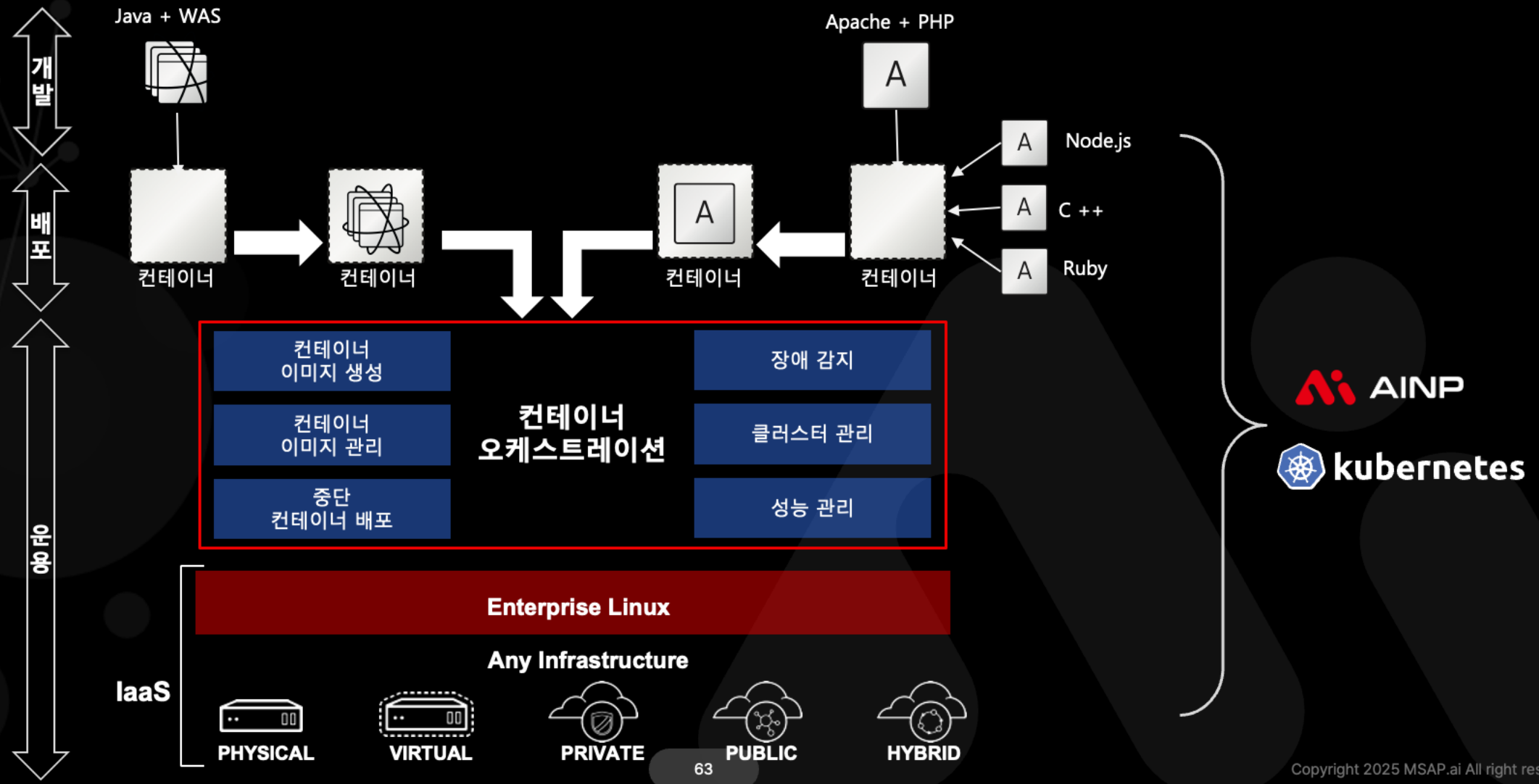
MSAP.ai

- 인프라 및 운영 방식 변화
  - Kubernetes 기반 네트워크 구조 재설계
  - 스토리지 동적 프로비저닝 대응
  - 롤링 업데이트 및 카나리 배포 방식 적용
  - 모니터링 및 관찰 체계 전환 (Observability)
  - 선언적 운영 및 GitOps 기반 배포 체계 도입
  - 로그 수집 및 집계 방식 재구성
  - Kubernetes 리소스 중심 문제 해결 방식 적용
- 개발 및 기술 환경 변화
  - 애플리케이션의 컨테이너화 전환
  - 설정 자동화 및 YAML 기반 정의
  - 다양한 운영 자동화 기능 적용 (단순 운영 부재 아님)
- 조직 및 보안 체계 변화
  - RBAC, NetworkPolicy, PodSecurity 중심 보안 체계 설계
  - DevOps 및 플랫폼 팀 중심의 협업 구조 전환
  - Kubernetes 기반 CI/CD 파이프라인 재구축
  - Liveness/Readiness Probe 기반 장애 대응 전략 수립

AINP (Container Orchestration Platform: 컨테이너 오케스트레이션 플랫폼)

MSAP.ai

- 모든 컨테이너는 오케스트레이션 환경에서 자동으로 배포 및 운영

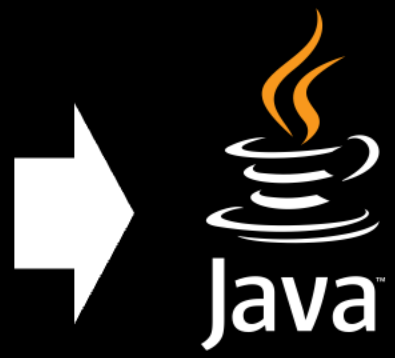


Vendor Lock-in Issues

MSAP.ai

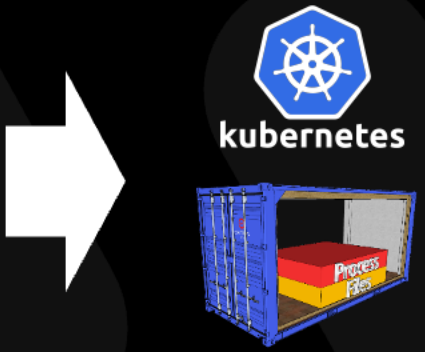
2000 년 – Java 를 통한 Vendor Lock-In 해제

OS	HP-UX	AIX	Solaris	Linux	Windows
CPU	Intanium	Power	Sparc	X86	X86
Vendor	HP	IBM	Oracle	The Linux Foundation	Microsoft



2020 년 – 컨테이너와 Kubernetes 를 통한 Vendor Lock-In 해제

Hypervisor	Xen Project	KVM	Hyper-V	Xen Project	Xen Project
Public Cloud	aws	Google Cloud	Microsoft Azure	Oracle	IBM Cloud



How does one build apps for the cloud?

MSAP.ai

Virtual Machine

Write once, run anywhere?

Unix Server Battle , There Are No Winners

MSAP.ai

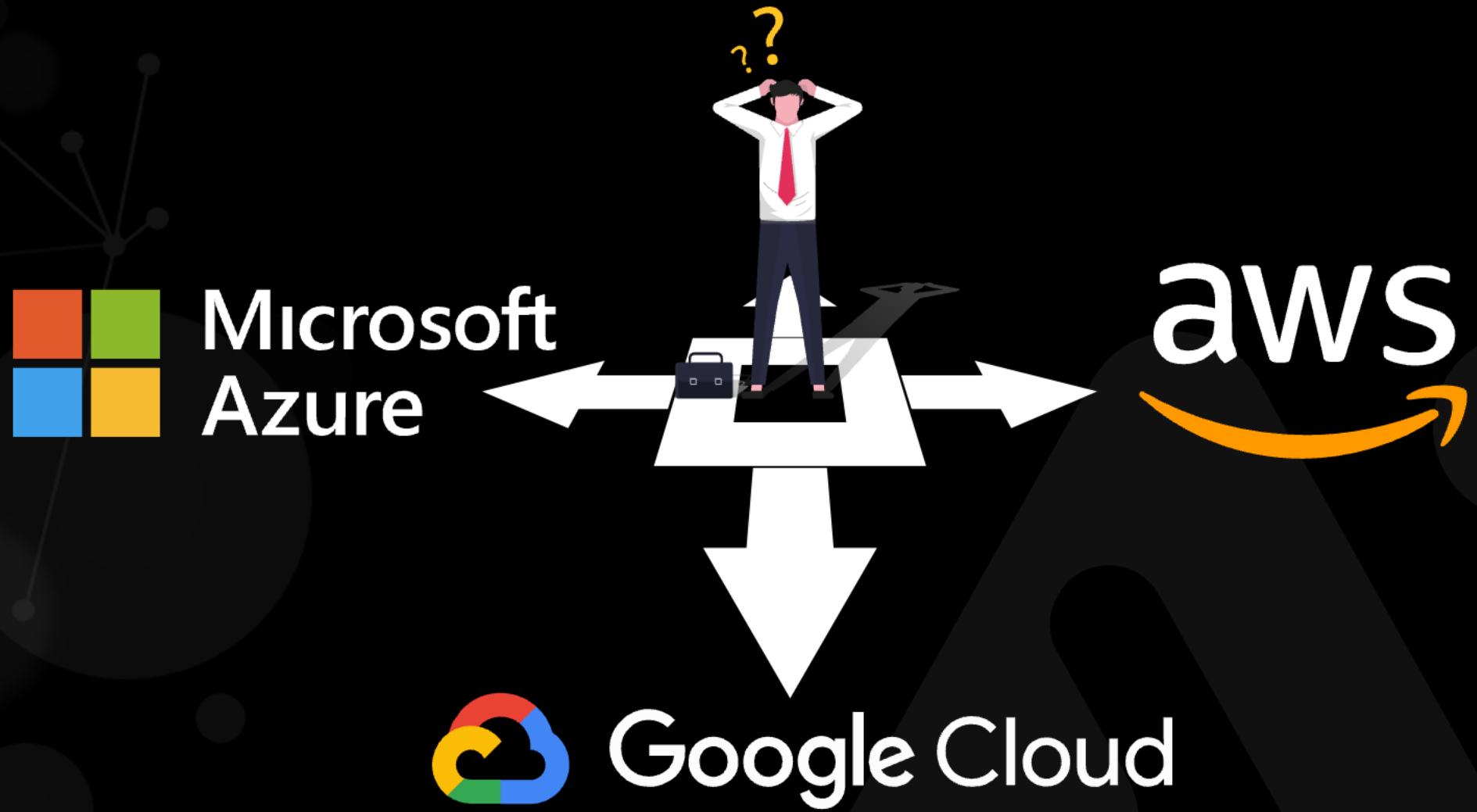
Unix

Vs

Linux™

AWS Vs Azure Vs GCP – The Cloud Platform of Your Choice?

MSAP.ai



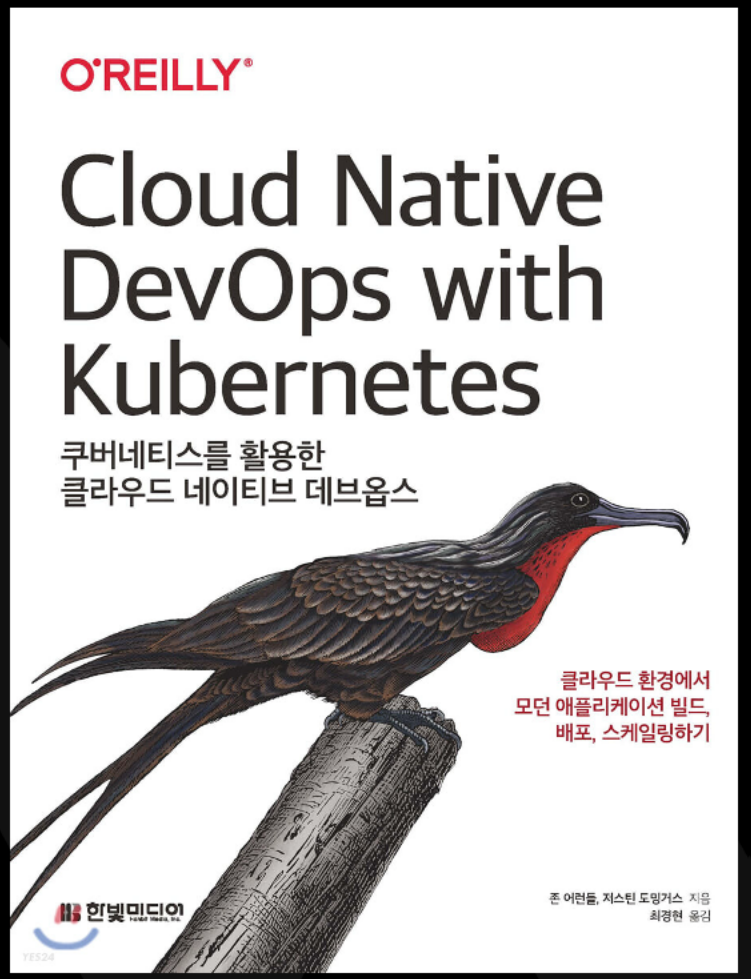
개발팀을 위한 AINP (AI Native Platform)

클라우드 네이티브 애플리케이션

MSAP.ai

- 클라우드 환경에서 현대적인 애플리케이션 빌드, 배포, 스케일링 하기

배포 자동화	사람이 수동으로 처리하지 않고, 소프트웨어가 애플리케이션의 배포와 운영을 자동으로 수행함.
이식성	디스크나 머신과 같은 물리적 리소스에 대한 상세한 지식 없이도, 애플리케이션을 클러스터 간 또는 서로 다른 클라우드 환경 간에 손쉽게 이전할 수 있음.
분산화	하나의 거대한 모놀리식 애플리케이션 대신, 서로 독립적이면서도 협력하는 여러 마이크로서비스로 구성되어 분산된 방식으로 운영됨.
확장성	분산된 구조의 애플리케이션은 중복성 확보와 그레이스풀 디그레이션(graceful degradation)을 통해 높은 가용성을 제공함.
무중단 서비스	스케일 아웃을 통해 가용성을 확보하고, 롤링 업데이트 방식으로 애플리케이션을 중단 없이 업그레이드할 수 있음.
관측 가능성	분산 시스템의 복잡성으로 인해 프로파일링과 디버깅이 어려워지므로, 시스템 내부 상태를 파악할 수 있는 강력한 관측 가능성(Observability)이 필수적임.



The Twelve-Factor App 이란 무엇인가요?

MSAP.ai

I. Codebase (코드베이스)	모든 서비스는 단일한 코드 저장소에서 관리되어야 하며, 배포 환경마다 별도의 복제본을 사용합니다.
II. Dependencies (의존성)	라이브러리나 패키지 등 외부 의존성은 명확히 선언하고, 격리된 환경에서 관리해야 합니다.
III. Config (설정)	설정은 코드에서 분리되어야 하며, 환경 변수 등 외부에서 관리되어야 합니다.
IV. Backing services (백엔드 서비스)	DB, 메시지 브로커, 캐시 등 외부 리소스든 상관없이 URI 등으로 추상화하여 접근합니다.
V. Build, release, run (빌드, 릴리즈, 실행)	애플리케이션의 앱은 빌드(build), 릴리즈(release), 실행(run)의 세 단계를 명확히 분리해야 합니다.
VI. Processes (프로세스)	애플리케이션은 무상태(stateless) 프로세스로 실행되어야 하며, 상태는 외부에 저장해야 합니다.
VII. Port binding (포트 바인딩)	애플리케이션은 자체적으로 HTTP 서버를 실행하고, 외부에 포트를 통해 서비스를 제공합니다.
VIII. Concurrency (동시성)	애플리케이션은 수평 확장을 위해 여러 개의 프로세스(혹은 컨테이너)를 병렬로 실행할 수 있어야 합니다.
IX. Disposability (폐기 가능성)	애플리케이션은 빠르게 시작하고, 정상적으로 종료되며, 종료 시에도 상태를 정리하고 복구 가능해야 합니다.
X. Dev/prod parity (개발/운영 환경 일치)	개발, 스테이징, 운영 환경 간의 차이를 최소화하여 “내 환경에서는 잘 되는데?” 문제를 방지합니다.
XI. Logs (로그)	애플리케이션은 로그를 파일에 저장하지 않고 표준 출력으로 기록하며, 로그 수집은 외부 시스템이 담당합니다.
XII. Admin processes (관리 프로세스)	DB 마이그레이션이나 임시 디버깅 작업 등은 앱과 같은 실행 환경에서 실행되어야 합니다.

Cloud Native Application 도전!

MSAP.ai

- 신속한 개발과 클라우드 확장성 확보를 위한 클라우드 네이티브 애플리케이션 개발은 필수
  - ➢ MSA , Container , Kubernetes, CI/CD 등

spring

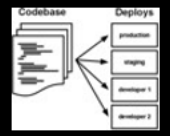
ORACLE WEBLOGIC

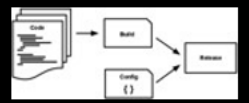
JBoss by Red Hat

개발 리더

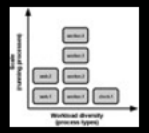
아키텍트

기존 개발에 필요한 기술





12-Factor applications



Independent, autonomous, stateless, processes

OpenJDK kafka

Microservices redis

CouchDB mongoDB

Application as composition of polyglot services

etcd

Cloud Environment

NETFLIX

Configuration sources, service references

CD.FOUNDATION Spinnaker

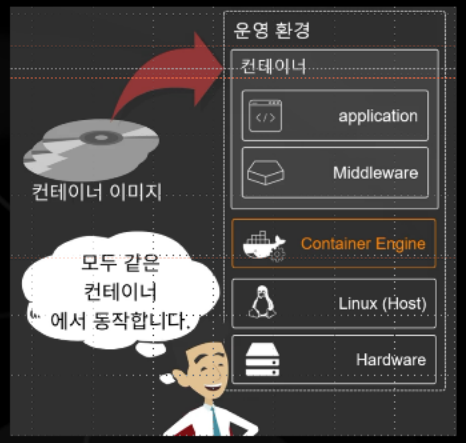
DevOps

JENKINSX

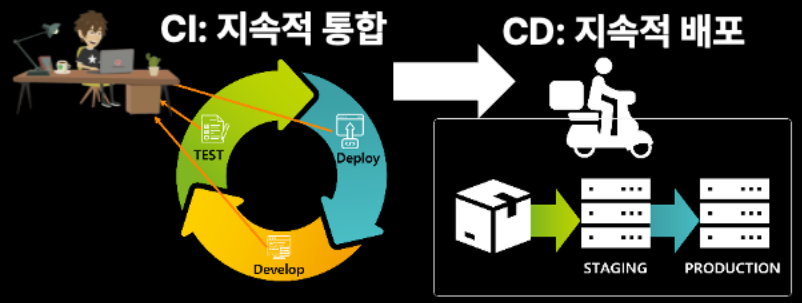
Build, package, deploy, observe

클라우드 네이티브 애플리케이션 기술들

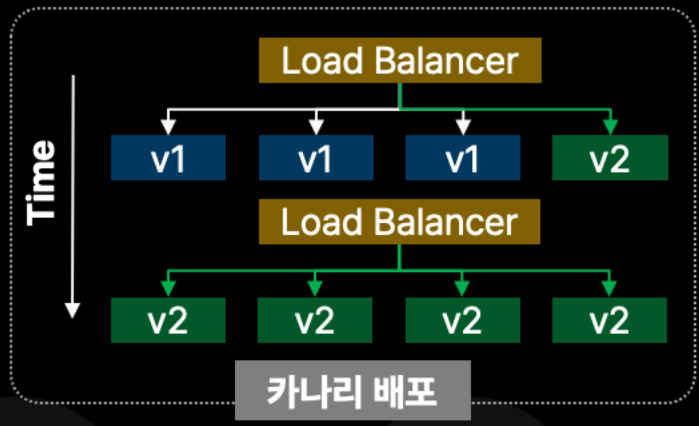
개발자 측면의 컨테이너 기반 환경의 장점



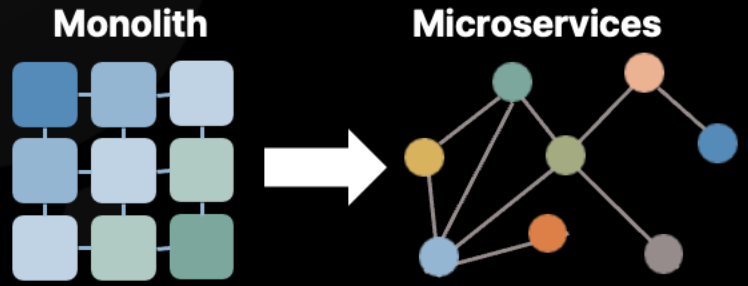
컨테이너 기반의 빠른 개발환경



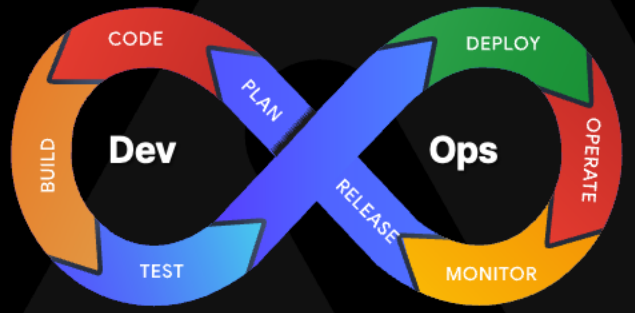
신속한 개발과 편리한 배포



서비스 무중단 배포



MSA개발에 적합한 환경



DevOps기반의 민첩한 개발

컨테이너는 애플리케이션 배포하면 운용에"일반적인 방법"를 제공

MSAP.ai

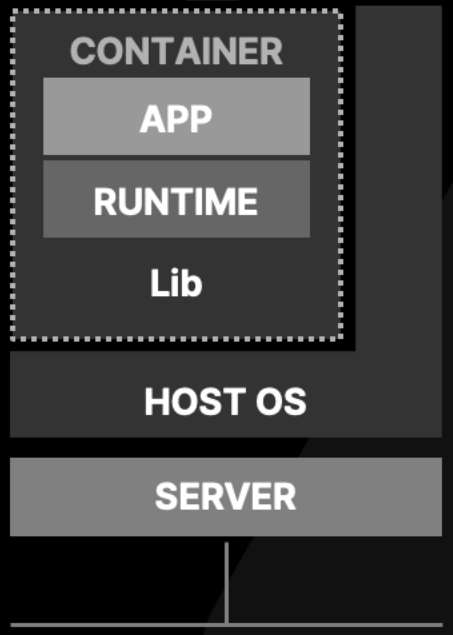
- 컨테이너는 애플리케이션 배포하면 운용에"일반적인 방법"를 제공

기존 새 여러 유형의
애플리케이션을
컨테이너에 패키지 수
호스트OS 전개

Java + JBoss

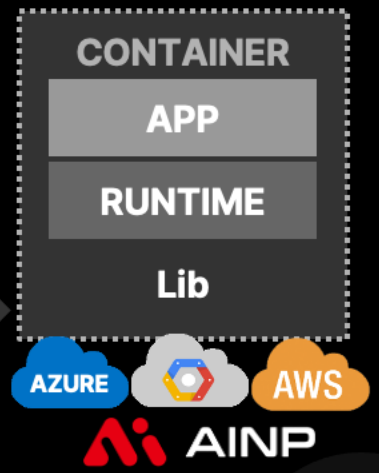
PHP, C ++ , Ruby

Database



이동성

높은 호환성



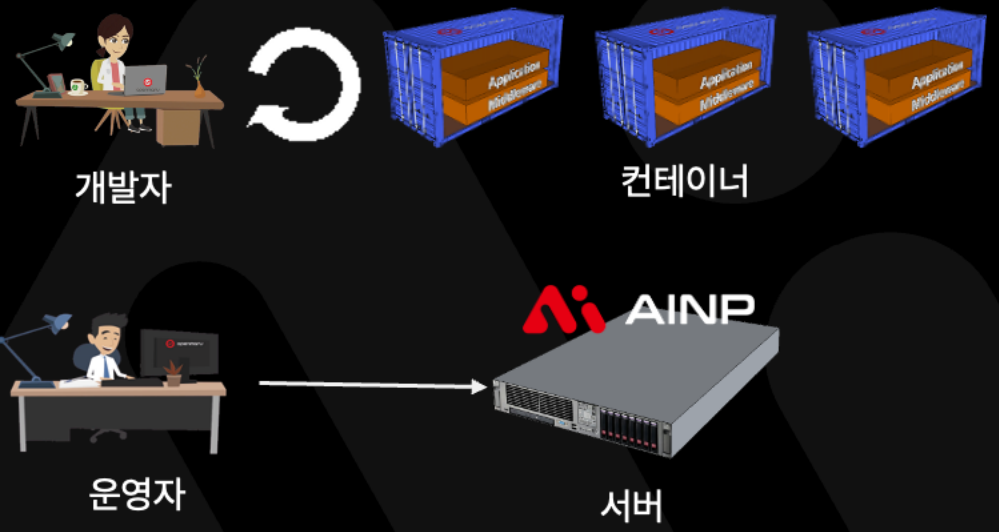
작업 관리자는 애플리케이션
종류에 관계없이 " 일반적인 방법 " 로 관리 할
수 있다. 애플리케이션 작기 때문에 배치도 고속

애플리케이션 배포 주기 단축

MSAP.ai

- 개발팀과 운영팀에서 지속적으로 논의하면서 배포

- 개발팀에서 자동으로 애플리케이션 배포



개발 환경

컨테이너

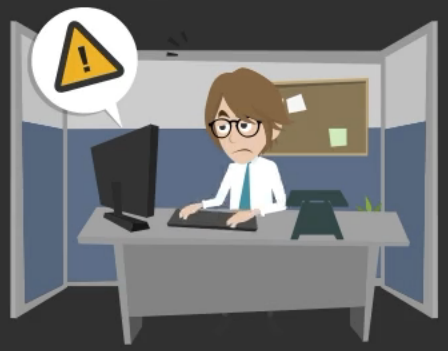
application

Middleware

Container Engine

Linux (Host)

Hardware



테스트 환경

컨테이너

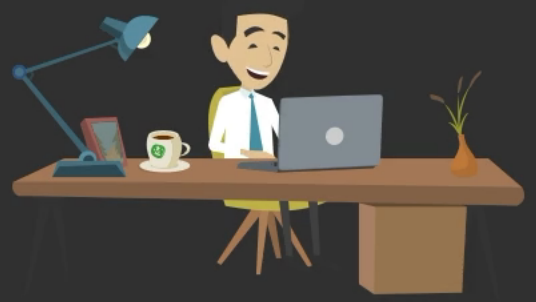
application

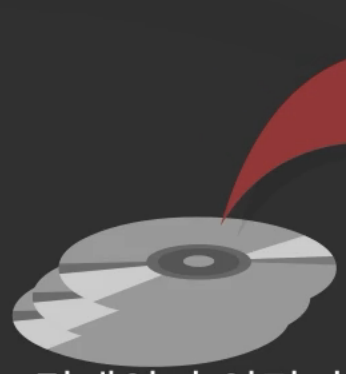
Middleware

Container Engine

Linux (Host)

Hardware





컨테이너 이미지

운영 환경

컨테이너

application

Middleware

Container Engine

Linux (Host)

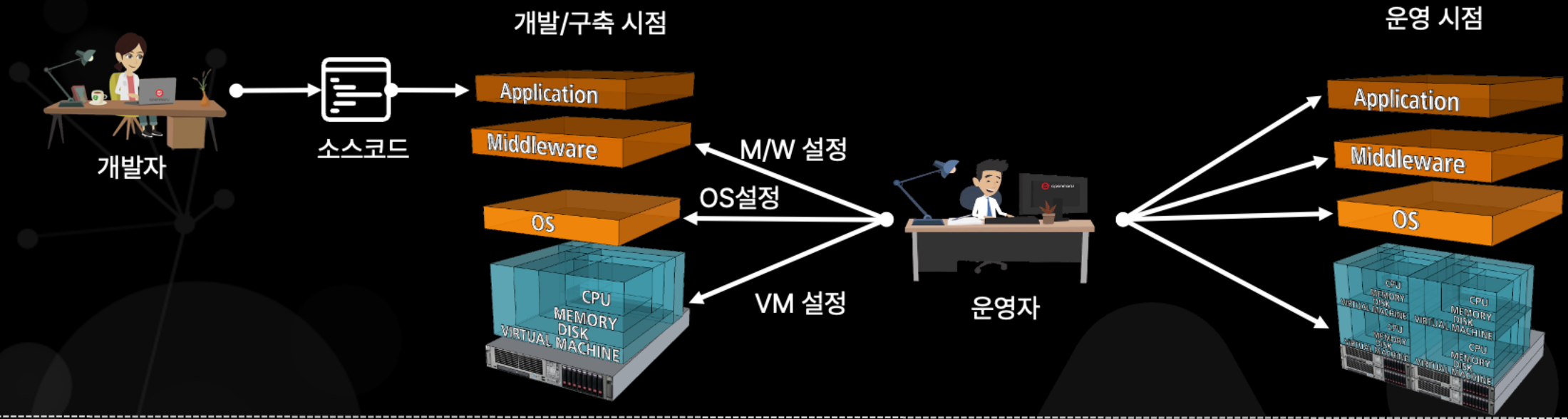
Hardware

모두 같은
컨테이너
에서 동작합니다.

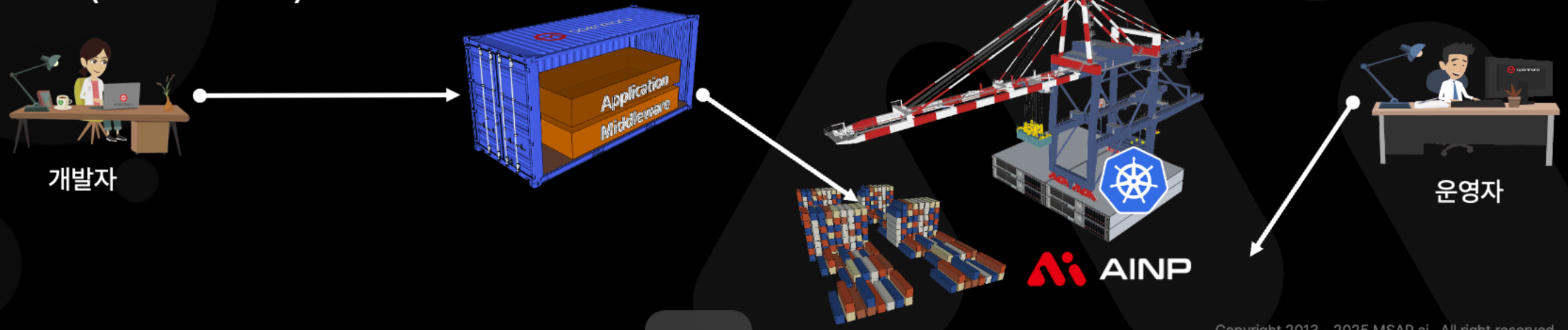
개발과 운영의 변화

MSAP.ai

- 가상화 (Virtualization) 에서 개발 및 운영



- 컨테이너 (Kubernetes)에서 개발과 운영의 변화



개발 환경이 늘어나는 요인 = 버전 × 용도 × 작업자

MSAP.ai

- 버전에 따라 필요한 개발 환경
  - 예: v1.0.0 개발, v1.1.0 개발, v1.2.0 개발, ...
  - 버전이 바뀌면 기본적으로 서버 코드도 변경
  - master 브랜치 하나로 운영하면서 미래 기능까지 모두 포함되면 결국 매우 복잡해짐

- 용도에 따라 필요한 개발환경
  - 예: 개발 환경, QA 환경, LQA 환경, 전용 테스트 환경 등
  - 개발 환경과 QA 환경은 분리되어야 함
  - 최소한 버전 1개당 2개 이상의 환경이 필요
  - 그 외의 환경은 목적에 따라 추가로 계속 증가

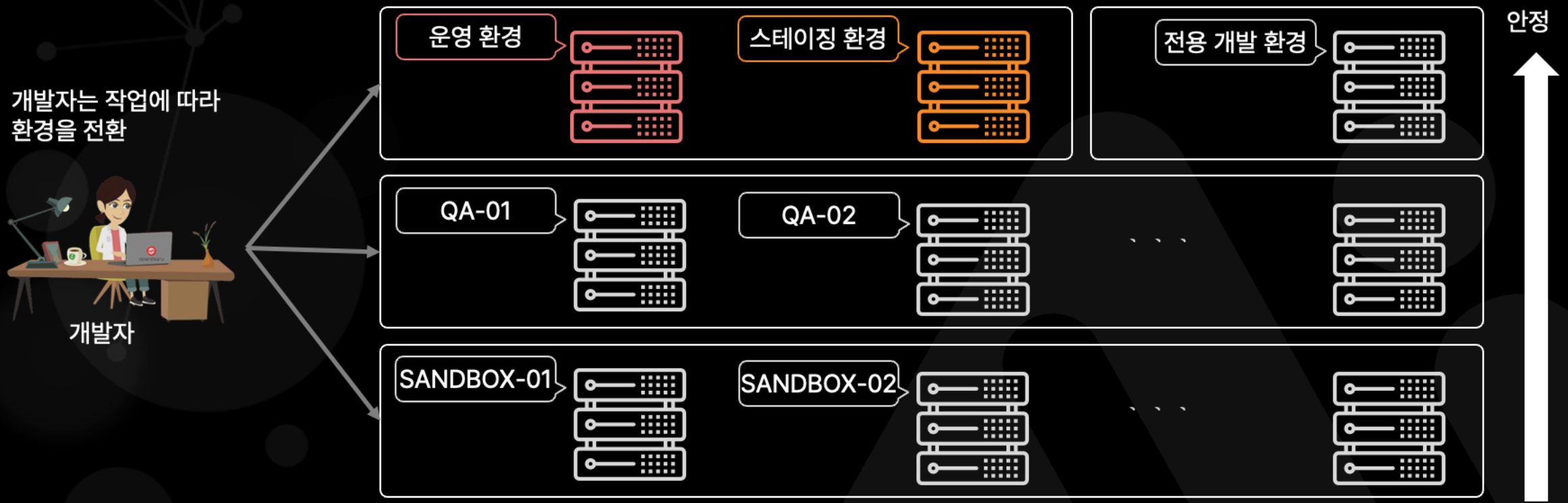
- "작업자마다 서버를 따로 나눠야 하는가?"
  - 전체 환경에 영향을 주는 작업을 하는 경우 - 예: 전체 업그레이드 환경 , 특수한 검증 작업 등
  - 동작 확인을 위해 시간 변경이 필요한 경우
    - 서버 시간을 직접 변경할 수 없다면
    - 사용자별로 시간을 조작할 수 있는 기능이 필요한 경우
  - 작업자마다 master 브랜치를 편집할 수 있는 환경이 필요한 경우
    - 이 작업을 동시에 여러 사람이 하게 되면 서버 환경이 기하급수적으로 늘어나며 폭발적으로 증가

🚨 주의! 작업자 환경이 늘어나면 서버 수가 폭증할 수 있음

대규모 시스템의 개발을 운용을 위한 개발 환경

MSAP.ai

- 기본적으로 개발 환경은 여러 서버가 필요하며, 개발팀이 커질 수록 많이 필요
- 운영/스테이징/개발 및 샌드박스 환경 구축



컨테이너 기반의 빠른 표준 개발 환경 구축

MSAP.ai

개발환경	테스트환경	운영환경
개발머신요청	테스트머신요청	운영머신요청
OS 설치 구성	OS 설치 구성	OS 설치 구성
미들웨어 설치구성	미들웨어 설치구성	미들웨어 설치구성
빌드스크립트 작성	빌드스크립트 작성	빌드스크립트 작성
개발 배포	테스트 배포	운영배포
코딩	테스트	배포확인

기존환경에서 개발환경 구축 절차

개발환경	테스트환경	운영환경
프로젝트생성	프로젝트생성	프로젝트생성
이미지 선택		
빌드스크립트 작성		
자동 배포	자동 배포	자동 배포
코딩	테스트	배포확인

컨테이너 환경의 개발환경 구축 절차

개발자의 핵심업무에 집중

요구사항 변경에 즉시 대응가능한 DevOps 개발 환경

MSAP.ai

AS-IS 개발 체계

"확정되지 않은 요구사항 + Waterfall 개발"

이슈

업무 및 기능요건 미정의

개발기간 중 다수의 요구사항 발생

개발시 예상치 못한 이슈 발생

현상

추가 요건 수용 한계

설계 → 개발 → 테스트 ⇢ 운영 배포

개발환경 ≠ 운영환경

이슈

- 명확한 업무 정의 없이 개발이 진행되어 혼란 발생
- 개발 환경과 운영 환경의 불일치로 인해 비효율 증가

DevOps 개발 체계

"요구사항 명확화 + DevOps 방식 개발"

요구사항 사전 정비

업무 프로세스 개선 → 업무 흐름 정의 → 기준 및 정책수립 → 기능요건 구체화

DevOps 순환 개발

설계 → 개발 → 테스트 → 실시간 → 운영 배포

컨테이너 인프라 기반 DevOps 툴체인 지원

CI = CD

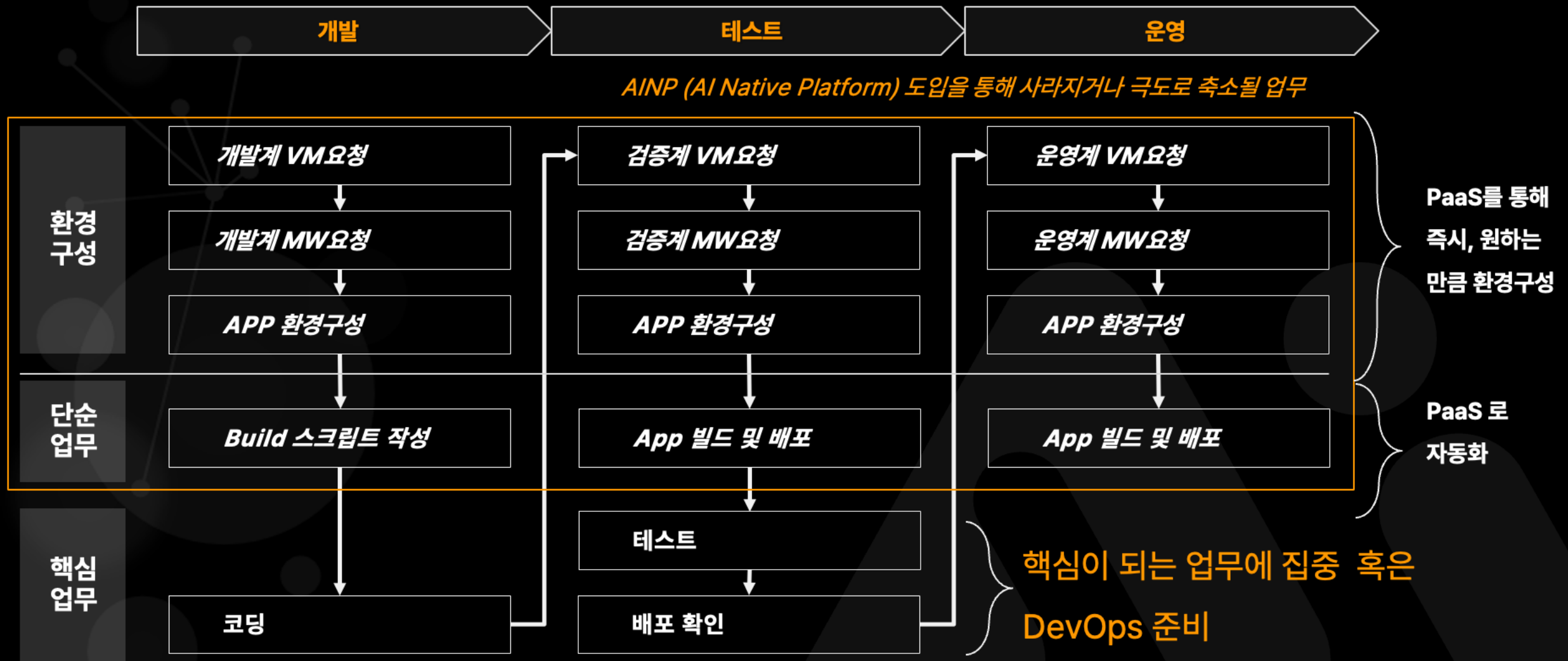
CD FOUNDATION

효과

- 개발 전 요구사항을 명확히 정의하여 불확실성 최소화
- DevOps 기반으로 변경사항에 빠르게 대응 가능

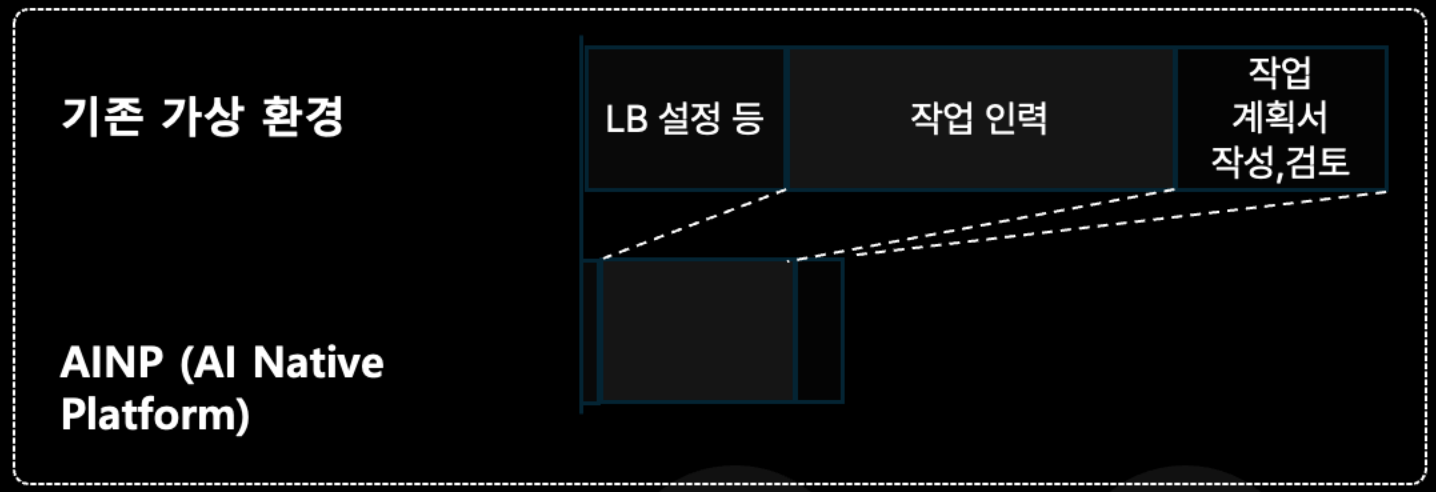
복잡한 환경 구성에서 핵심 업무 집중으로 – AINP (AI Native Platform) 활용 방안

MSAP.ai

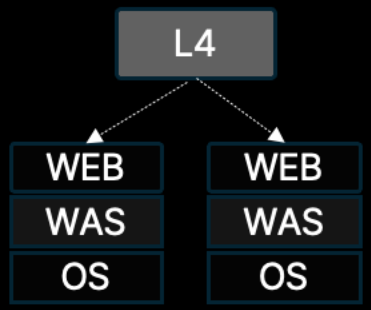


애플리케이션 배포 비용

- 애플리케이션 배포 비용이 약 1/3
- 야간 작업 폐지/축소



Before



- LB 설정/서버마다 수동 배포
- 수작업으로 인한 휴먼 에러
- 장시간 배포/ 서비스 정지
- 작업계획서 작성 및 리뷰

After

- 자동 배포
- 단 시간/무정지서비스

비즈니스 가치 창출을 위해 코드 개발에 집중하고,

비효율적인 운영은 피하는 것

AI Native Platform

AINP - Cost Effectiveness

KUBERNETES 를 제대로 운영하기 어려운 이유는?

MSAP.ai

Kubernetes 사용자 중 **75 %** 는

구축과 운영의 복잡성으로 도입하기 어렵다고 함

INSTALL
• Templating
• Validation
• OS Setup

DEPLOY
• Identity & Security Access
• App Monitoring & Alerts
• Storage & Persistence
• Egress, Ingress & Integration
• Host Container Images
• Build / Deploy Methodology

HARDEN
• Platform Monitoring & Alerts
• Metering & Chargeback
• Platform Security Hardening
• Image Hardening
• Security Certifications
• Network Policy
• Disaster Recovery
• Resource Segmentation

OPERATE
• OS Upgrade 및 Patch
• Platform Upgrade 및 Patch
• Image Upgrade 및 Patch
• App Upgrade 및 Patch
• Security Patches
• Continuous Security Scanning
• Multi-environment Rollout
• Enterprise Container Registry
• Cluster & App Elasticity
• Monitor, Alert, Remediate
• Log Aggregation

Source : The New Stack, The State of the Kubernetes Ecosystem, August 2017

아직도 WAS BMT 나 POC를 하시나요?

MSAP.ai

- WAS 의 가용성/확장성/성능/신뢰성 등의 기능은 플랫폼으로 이전
- 더 가볍고 더 빠르고, 자동화에 친화적인 WAS 로 전환

WAS 제품 BMT

ORACLE WEBLOGIC

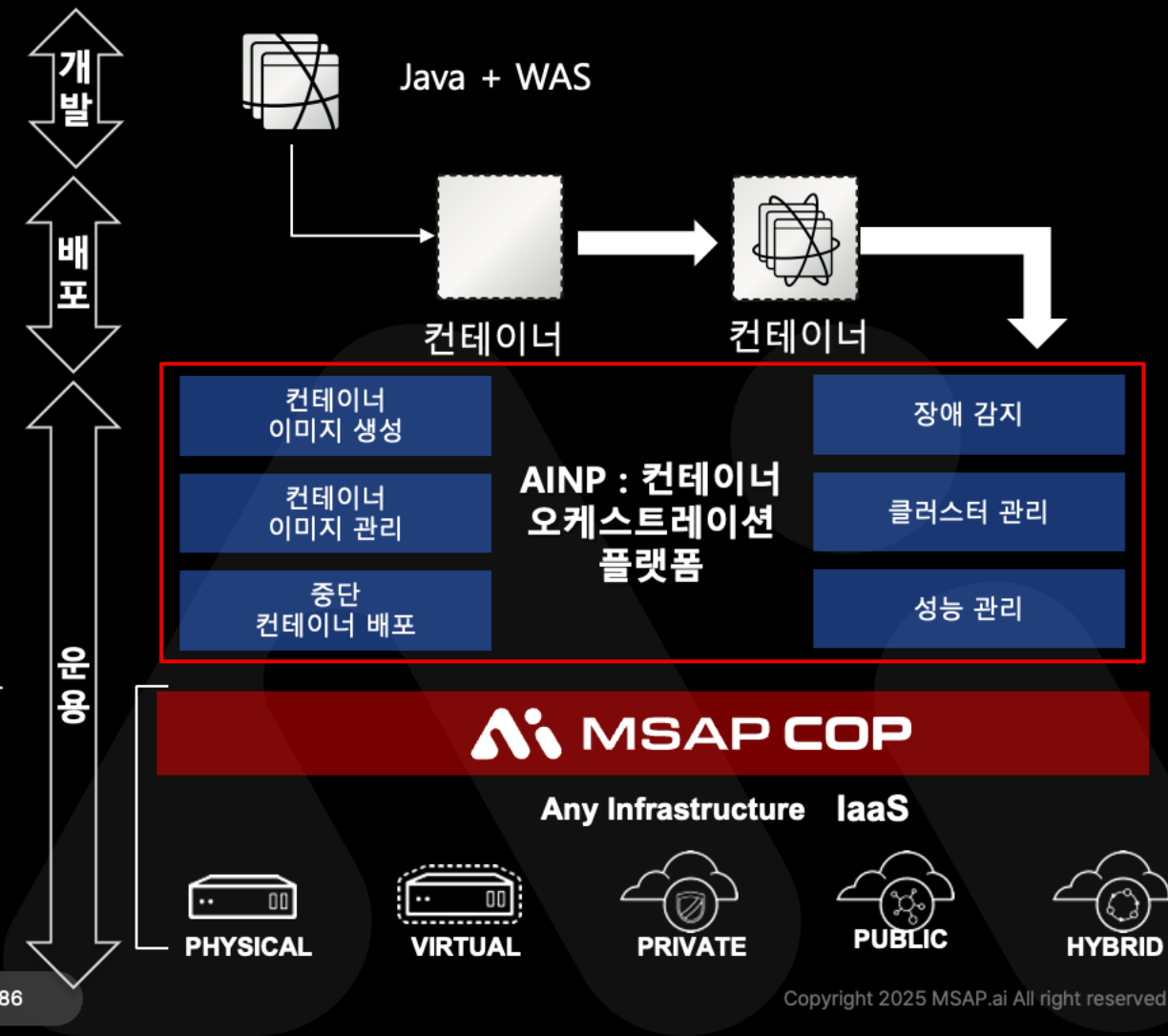
JBoss by Red Hat

WebSphere

Tmax Web Application Server JEUS

RASP 에 의한 WAS평가

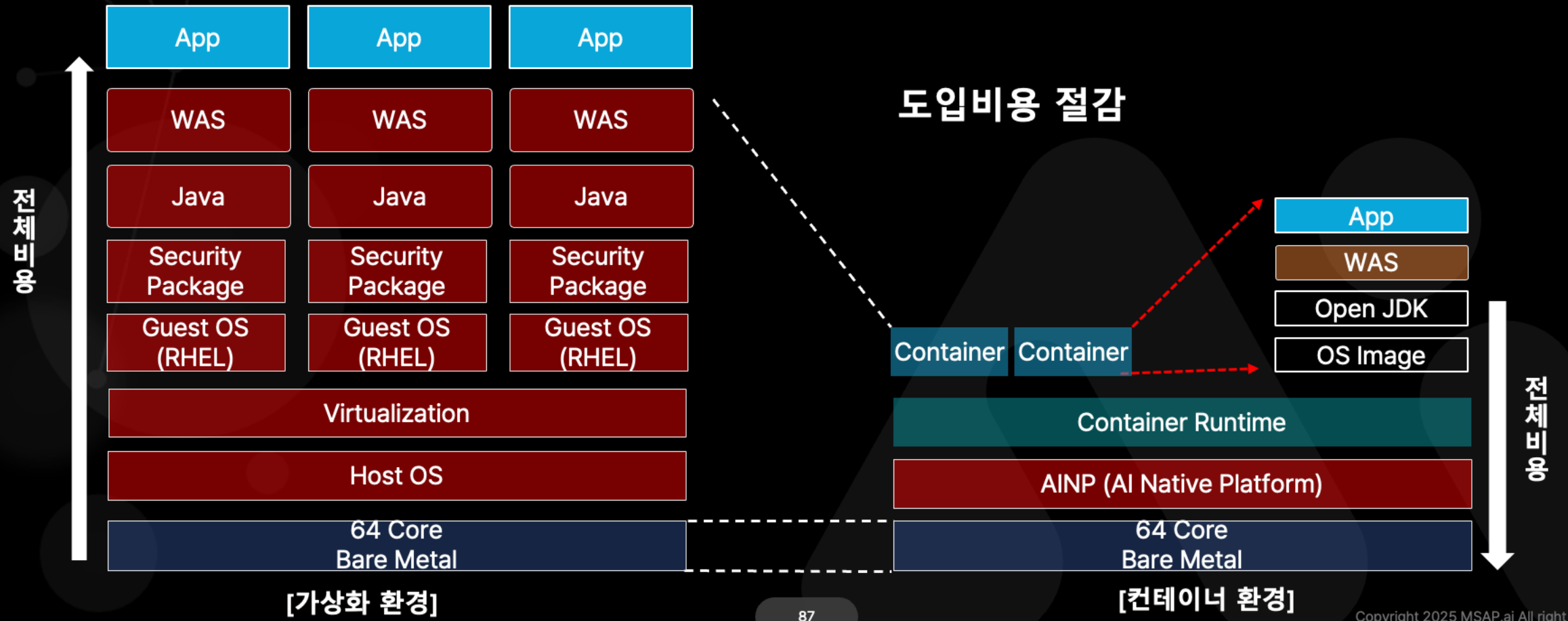
- 신뢰성 (Reliability):
  - 과부하 테스트
- 가용성 (Availability):
  - 일부 장애 발생시 전체 시스템 영향 최소화
- 확장성 (Scalability):
  - 자원 추가에 따른 선형적인 성능 개선
- 성능 (Performance):
  - TPS, 응답시간 등 평가
- 보안 기능 (Security)
  - (RBAC 등)
- WAS 도구 평가 (Manageability)
  - Admin Console



가상화 VS 컨테이너 비교 – 비용적인 측면

MSAP.ai

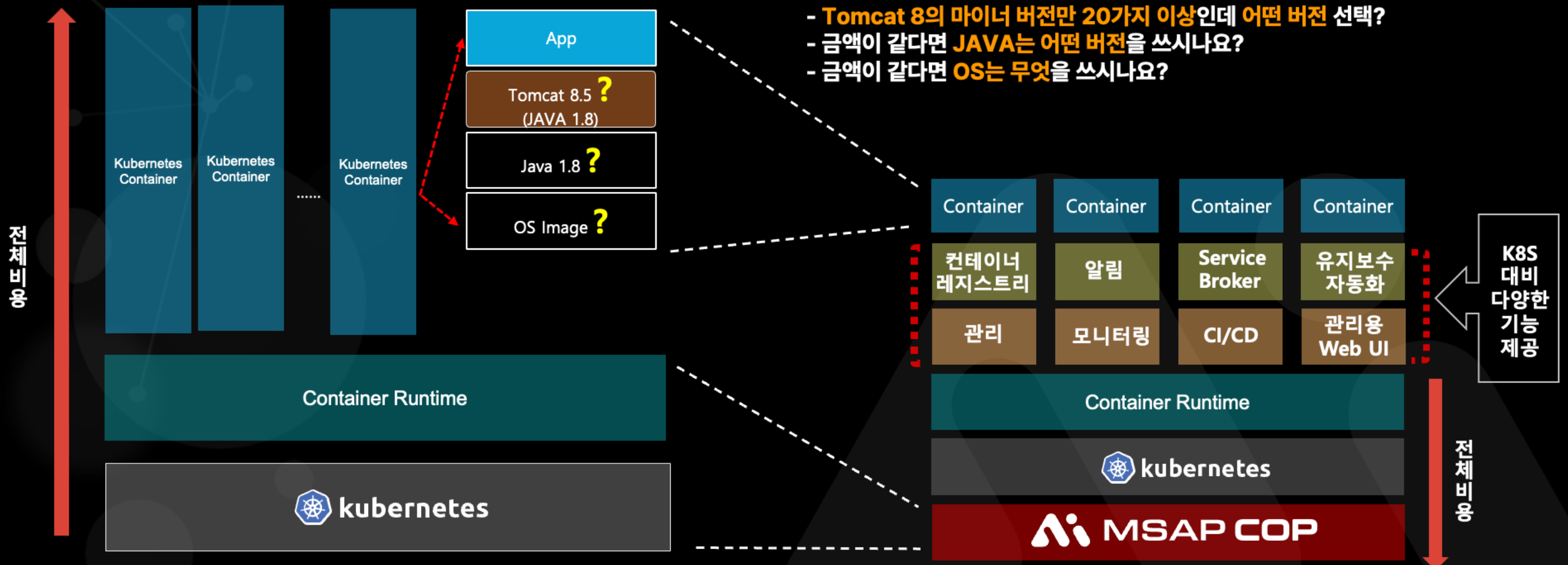
- 가상화 대비 Guest OS 유지보수, 라이선스, 관리비용 제거
- 서버 접근제어를 비롯한 보안 솔루션 제거



운영 환경에 부족한 Kubernetes vs 통합된 MSAP.ai

MSAP.ai

- MSAP.ai 에는 Tomcat/JAVA/OS 이외에도 APM , Observability 등 유지보수 제품들이 포함



AINP 기반의 클라우드 네이티브 혜택

MSAP.ai

- AINP (AI Native Platform) 을 적용하여 민첩성 높은 서비스를 제공

작업 공수 절감

기존 수동으로 해왔 던 작업을 자동화하여 작업 공정 및 납기 단축

운영 품질 향상

관리자의 개입을 최소로 자동화하여 작업 품질을 균일하게 유지

시스템 운영의 표준화 촉진

- 자동화 및 버전 관리 함으로써 시스템 운영 정책 및 업무 표준화

작업 통제 강화

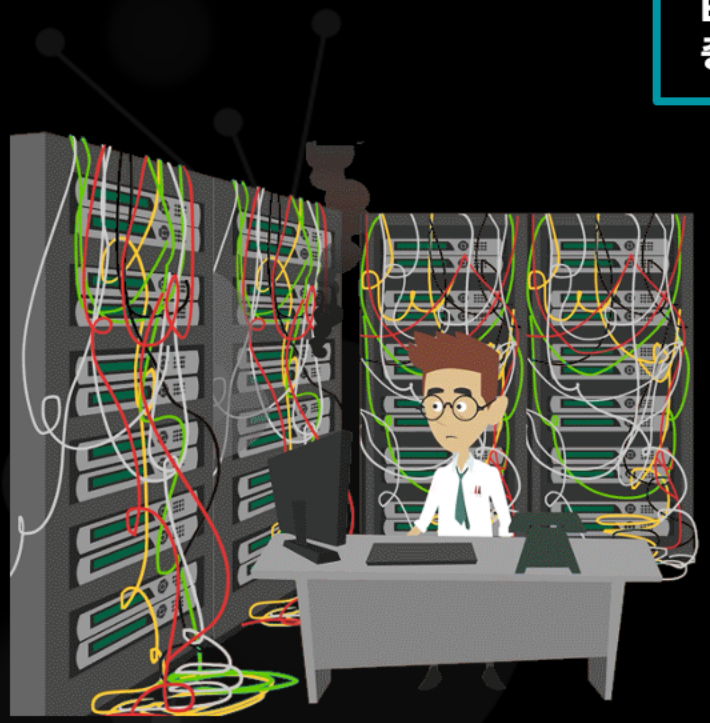
작업 작업을 자동화함으로써 내부 통제 및 보안 측면에서의 효과를 기대

AI Native Platform

운영자를 위한 AINP (AI Native Platform)

시스템 비대화로 작업 폭증과 인력 부족 어떻게 할까요?

MSAP.ai



장애의 65 %는 Human Error이며,시스템 복잡도와 난이도 증가

시스템 운용 업무의 45 %는 정기적으로 수행해야하는 반복 작업

운영 효율화를 통한 비용 절감의 요구

시스템의 대규모 화

높은 수준의 엔지니어 부족

지속적인 통합 요구

동일한 작업 반복

운영 품질 향상

운영 비용 (TCO) 절감 요구

업무 확대와 관련 데이터양의 비약적인 증가

가상화, 클라우드 등 다양한 운영 환경의 증가와 관리 효율화 요구

운영 품질에 대한 지속적인 향상 요청

개발 환경 제공 – 기존 시스템별 vs. Kubernetes

MSAP.ai

- 시스템 마다 절차와 관리도구가 다르고 수작업에 의한 운영
- PaaS를 통한 개발팀 스스로 필요한 개발 환경 구축

업무 별로 분산된 개발 환경

컨테이너를 통한 표준화

개발 환경 구성/관리가 어려움	필요한 만큼 개발팀 스스로 구성
시스템 마다 다른 구성과 지속적인 변경 관리	컨테이너기반 표준화된 운영 환경과 관리
배포는 다운 타임으로 인해 야간 수작업	필요에 따라 서비스 정지 없이 배포

개발 환경 자동화

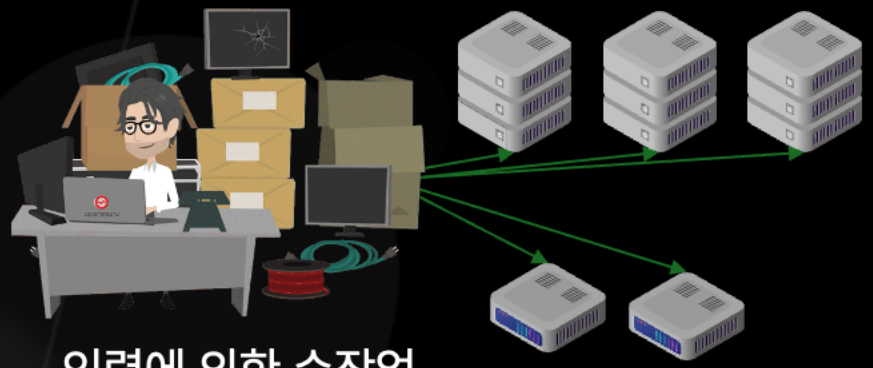
표준화와 자동화된 개발 환경

인프라 운영 – 기존 수작업 vs. Kubernetes 자동화 비교

MSAP.ai

- IT 인프라의 대규모화, 고도화에 따라 IT 장비에 대한 환경설정 및 정보 취합이 복잡하고 어려움
- 작업 계획시간과 현장 작업 시간의 증가와 휴먼 에러의 증가

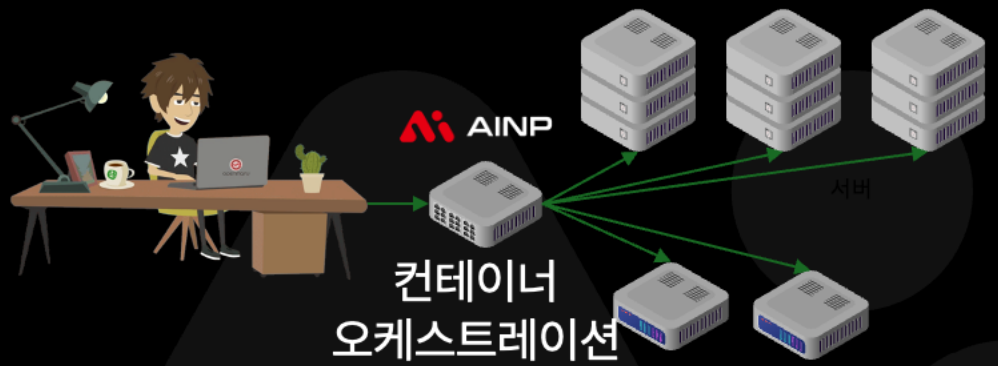
기존 시스템별 수작업



인력에 의한 수작업

- ✓ 시스템 운영을 위한 관리 작업 증가
- ✓ 현장 작업 시간 증가
- ✓ Human Error 와 품질의 차이

무정지 서비스/ 완전 자율 운영



컨테이너
오케스트레이션

- ✓ 운영 기술 표준화를 통한 준비 시간 및 작업 시간 감소
- ✓ 시스템 일괄 설정 작업 시간 단축
- ✓ 시스템을 통한 작업으로 Human Error감소

빠르고 정확한 개발 환경

MSAP.ai

시스템 별로 개발환경을 수작업으로 구축

수작업 계획서

수작업 계획서

수작업 계획서

수작업 계획서

WebLogic 12

Oracle DB 11

개발 서버 1

JBoss 6

MariaDB

개발 서버 2

Tomcat8

PostgreSQL

개발 서버 3

통합 개발환경으로 다양한 S/W 제공

AINP

Tomcat 8 Container

WebLogic 12 Container

JBoss 6.4 Container

Redis Container

Kafka Container

Apache httpd Container

Oracle 8 Container

MariaDB Container

PostgreSQL Container

Container Runtime

개발환경

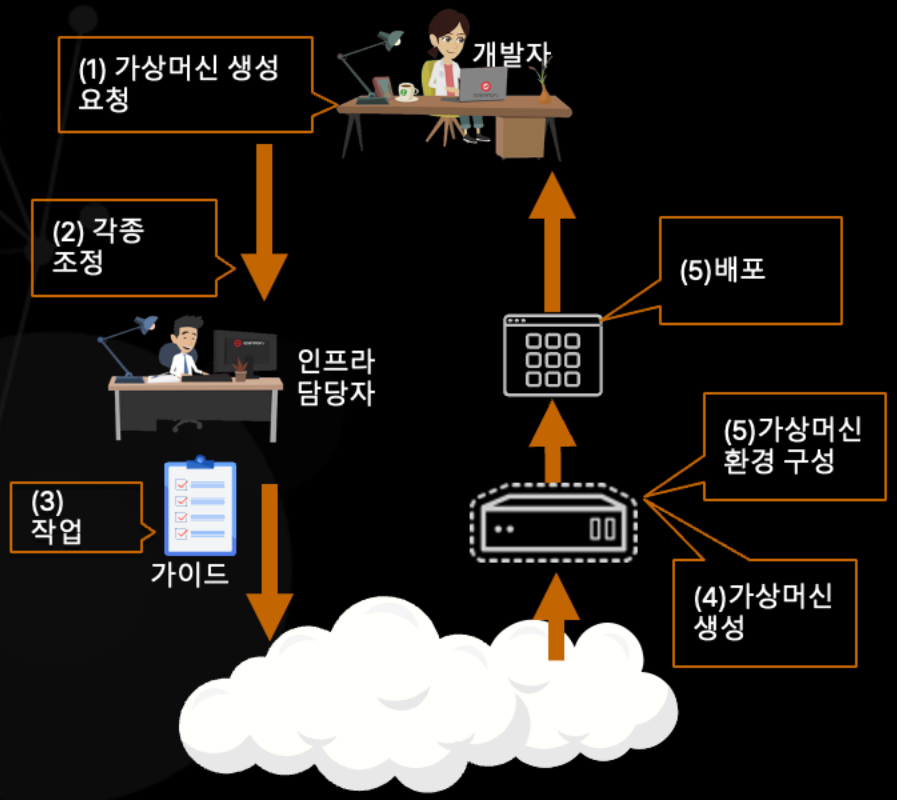
컨테이너 이미지

컨테이너 레지스트리

서버 추가 및 배포 – 기존 수작업 vs. Kubernetes 자동화 비교

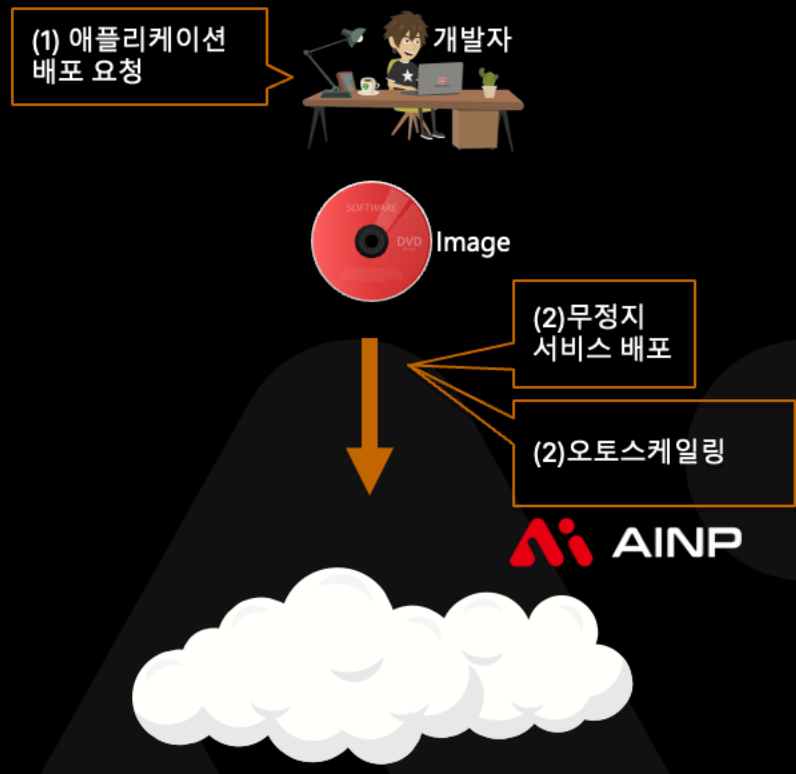
MSAP.ai

수작업에 의한 서버 구성 및 배포



- ✓ 사람에 의한 조정과 작업
- ✓ Human Error 와 품질의 차이

컨테이너 이미지 기반 무정지 서비스/자동 배포



- ✓ 테스트까지도 자동화
- ✓ 일정한 품질 유지

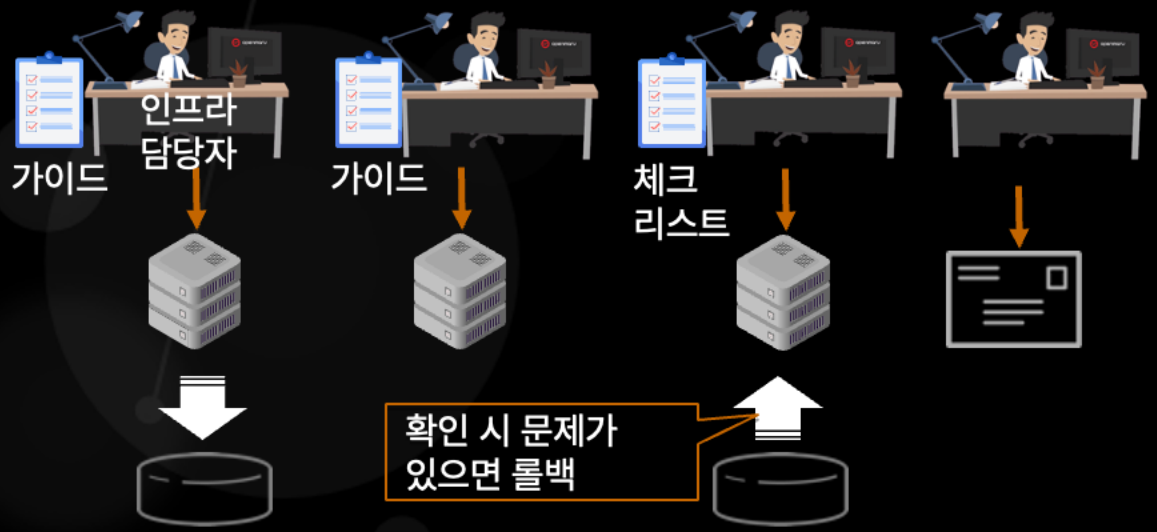
운영환경 패치 – 기존 수작업 vs. Kubernetes 자동화 비교

MSAP.ai

수작업으로 인한 야간 작업

업무 시간 외 작업

21:00	23:00	3:00	5:00
사전 백업	패치 적용	동작 확인	작업완료 메일



- ✓ 작업하는 동안 상주 지원하며 시간 소요
- ✓ 오류 확인 및 롤백에 대한 위험요소
- ✓ 품질의 차이

무정지 서비스로 업무 시간 중 패치

업무 시간 중

10:00	10:20	10:30	10:40
이미지 배포	롤링 업그레이드	동작 확인	작업완료 메일

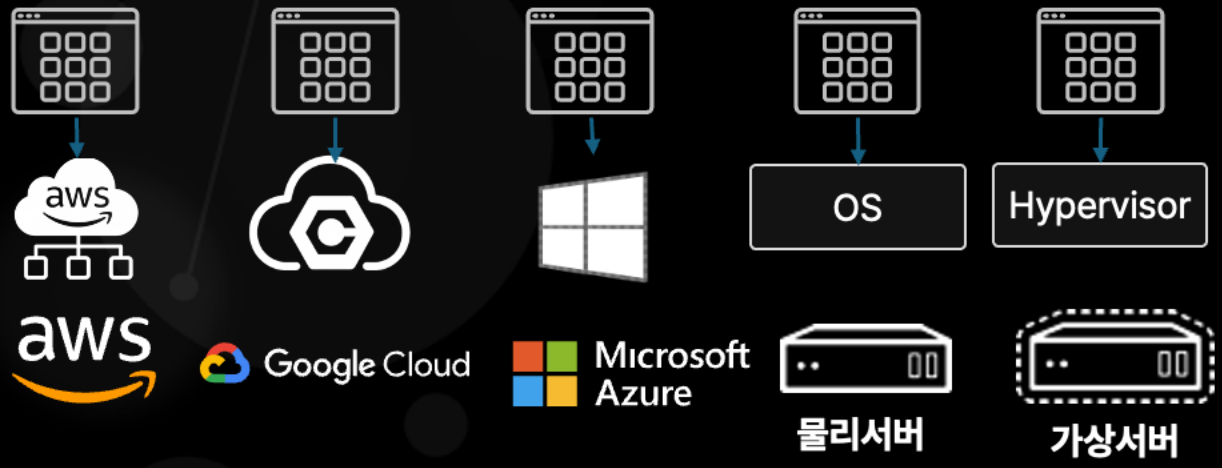
- ✓ 비상주 지원과 유비보수 시간 단축
- ✓ Human Error 차단
- ✓ 일정한 품질 유지

IaaS 종속형 운영 vs. 컨테이너 기반 Kubernetes 자동화 비교

MSAP.ai

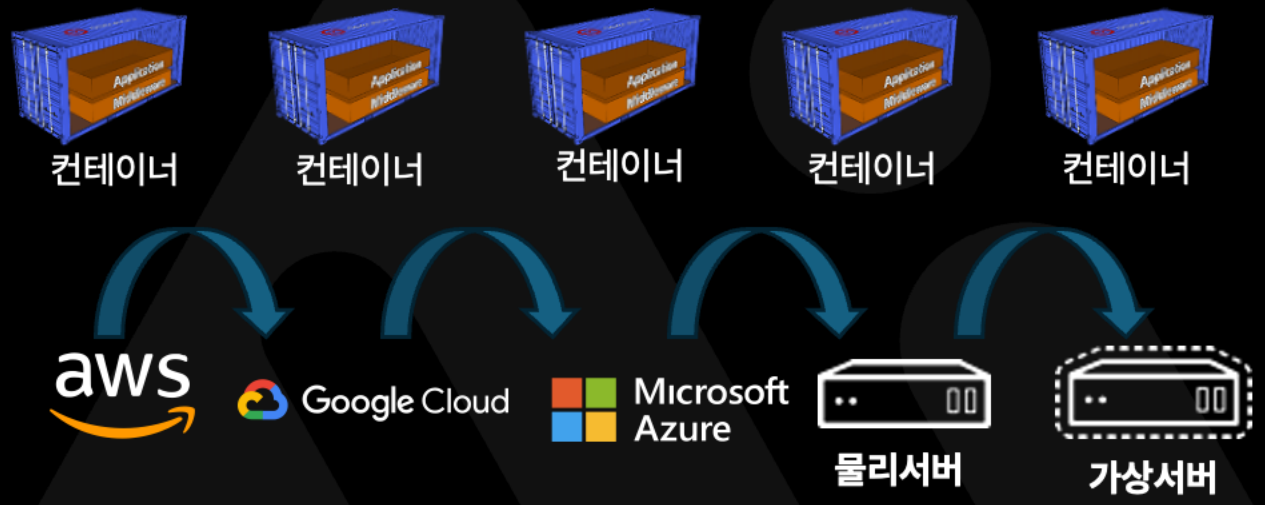
클라우드 환경에 의존적인 애플리케이션

- 클라우드 별, OS/하드웨어별 환경에 맞춰 수작업 배포 필요
- 각 클라우드(AWS, GCP, Azure 등) 및 인프라(OS, Hypervisor 등)에 따라 애플리케이션 구성이 달라짐



컨테이너 기반으로 모든 인프라 환경에서 동작

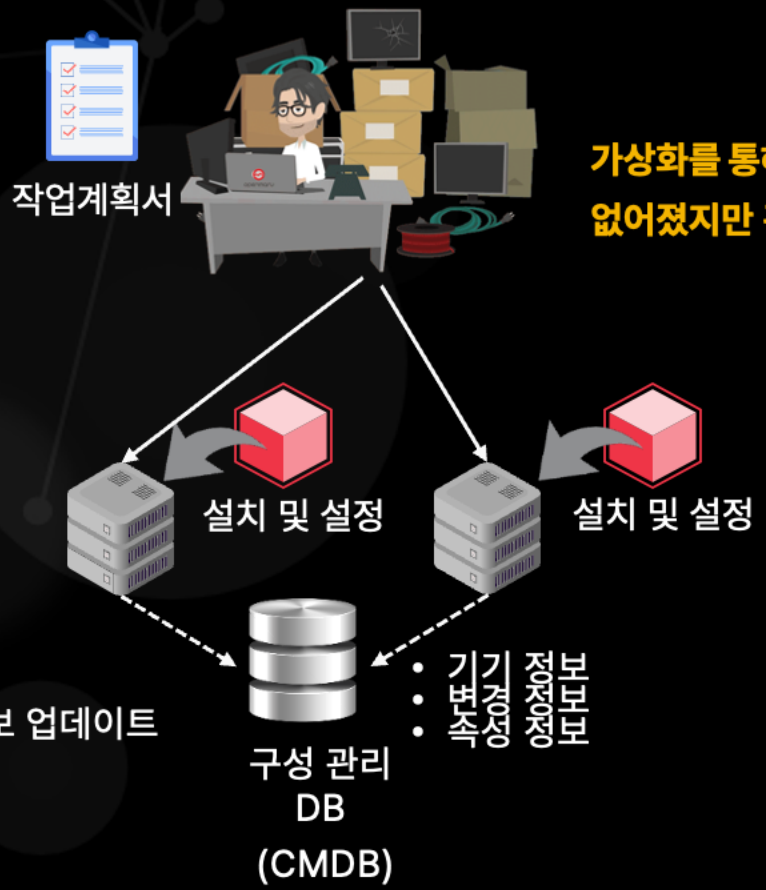
- 컨테이너 기반으로 동일한 패키지 형식으로 어떤 환경에서도 실행 가능
- 클라우드 및 인프라 무관하게 동일한 방식으로 배포 가능



IaaS 기반의 인프라 구성 관리 이슈

MSAP.ai

- On Premise IaaS 기반 인프라
  - 엔지니어가 복잡한 인프라 구축
  - 구축 후 만료될 때까지 장기적 운영
  - 미들웨어 클러스터 구축 ,수백 대의 OS 초기 설치 및 설정 등



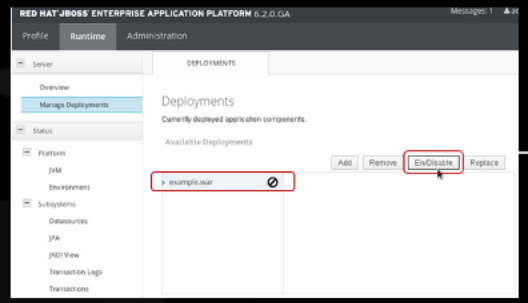
가상화를 통해 물리적 제약이 없어졌지만 관리 복잡성 증가

- IaaS 기반 클라우드 전환하면서
  - 인프라 자원의 라이프 사이클이 짧아짐
  - 구축과 운영 비용이 급증

인프라 운영 - WAS 관리 방법의 변화

MSAP.ai

- WAS 컨테이너 이미지는 관리 콘솔을 생략하고 배포
- 더 이상 개별 WAS 인스턴스에 대한 관리가 무의미 함



WAS Admin Console

- WAS 수동 시작/정지
- WAS 수동 장애 복구

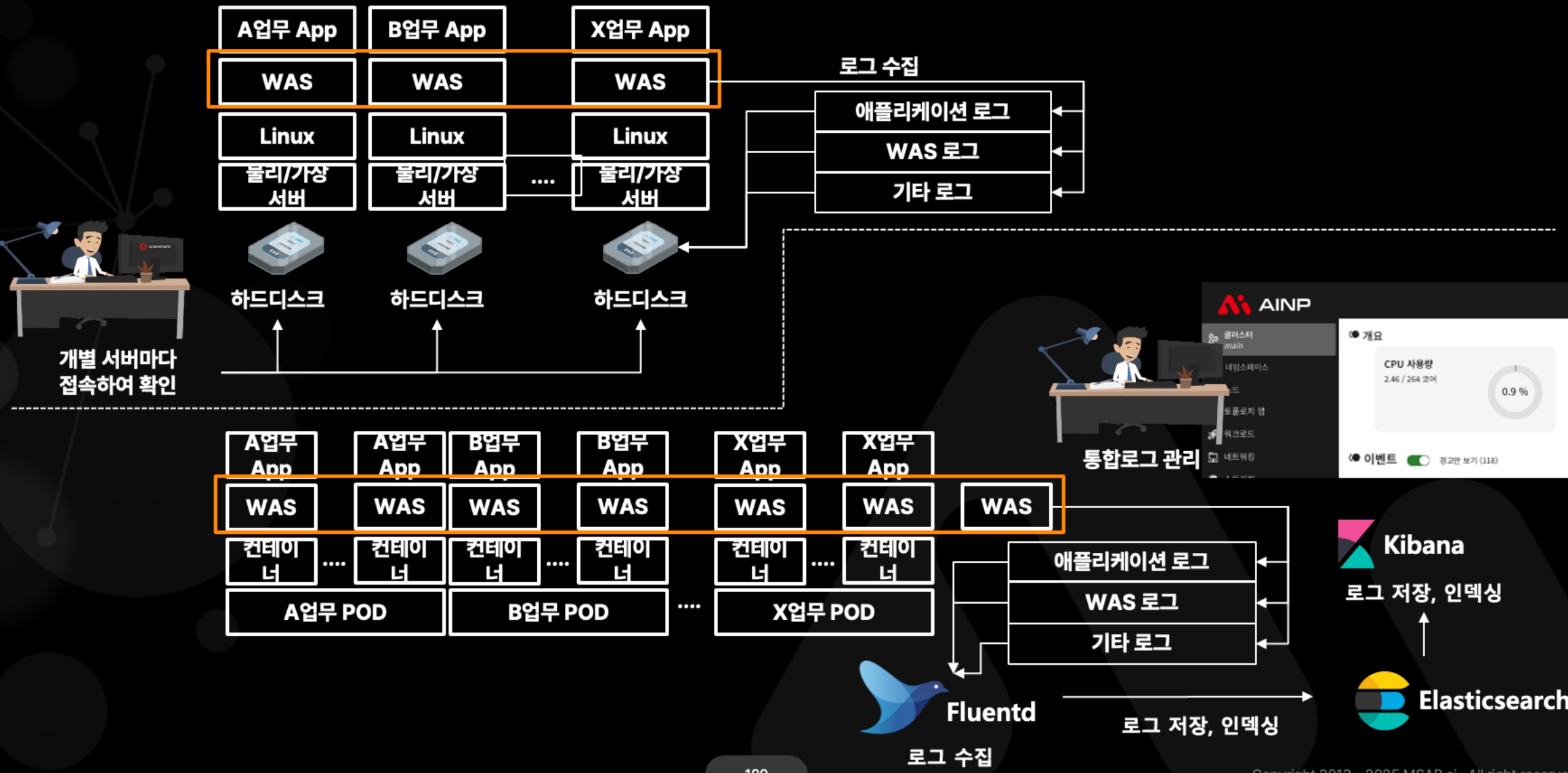
WAS 관리 콘솔 - WAS 관리

MSAP.ai 관리 콘솔 - 컨테이너(WAS 포함)

AINP (AI Native Platform) Console

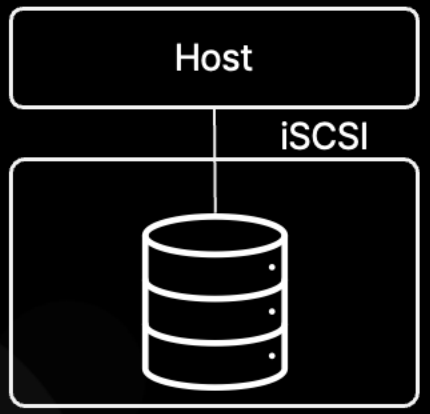
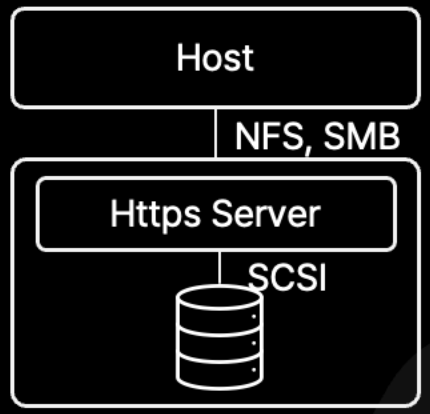
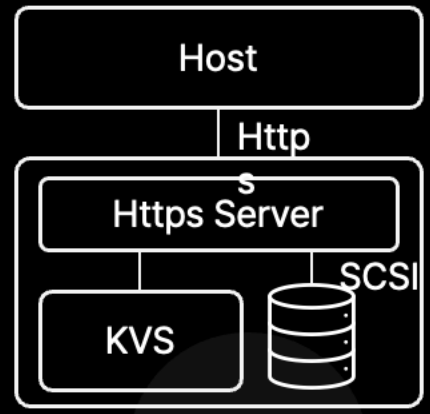
- 컨테이너 자동확장/축소
- 컨테이너 자동장애 복구
- 컨테이너 서비스체크
- 컨테이너 모니터링

로그 관리 방법의 변화



스토리지 타입

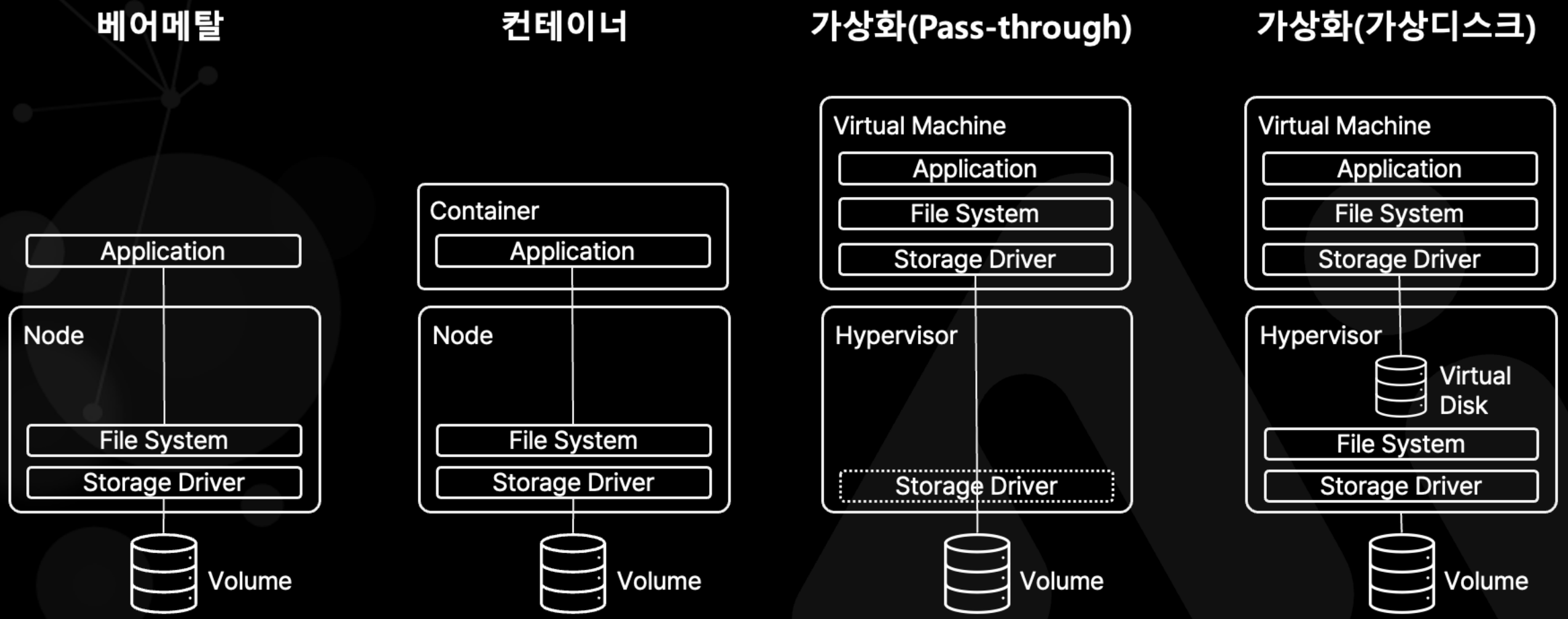
MSAP.ai

분류	블록 스토리지	파일 스토리지	오브젝트 스토리지
CaaS			
특징	• 내장 드라이브와 동일한 원시 장치 • 파일 시스템은 자유롭게 선택	• 네트워크 드라이브 • 파일 시스템은 스토리지임 해, 변경 불가	• 객체 단위로 액세스 • 파일 시스템 독립적 • 대량 데이터 저장
데이터 전송 프로토콜	• iSCSI, FC, NVMe	• NFS,SMB	• HTTPS(HTTP)
마운트 경로	/dev/sda	\\192.168.0.1\share\hoge	https://aaa.org/storage/hoge
성능*	• High	• Middle	• Low
주요 용도	• DB, OS(Boot Disk)	• 파일 공유	• 사진, 동영상 저장

베어메탈 vs. 컨테이너 vs. 가상화 Volume 연결 방식 비교

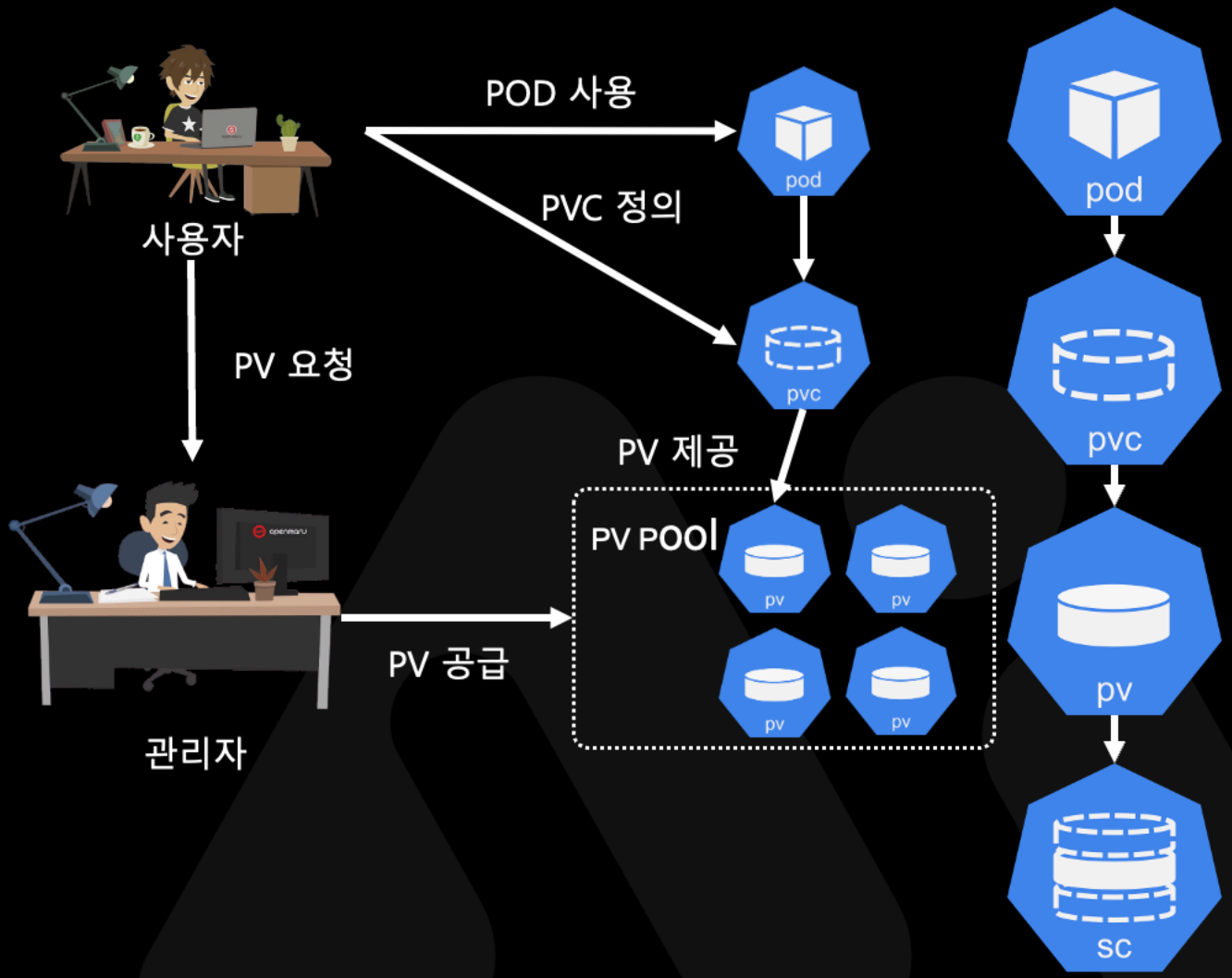
MSAP.ai

- VM 상의 컨테이너를 운영하는 경우 컨테이너 가상화와 VM가상화의 옥상옥 구조



스토리지 모델

- 벤더 독립 모델
  - PersistentVolumeClaim
  - PersistentVolume
  - StorageClass
- 관리자와 사용자의 역할을 고려한 모델
- Volume 을 컨테이너에 연결
- 주로 블록 스토리지/파일 스토리지 대상



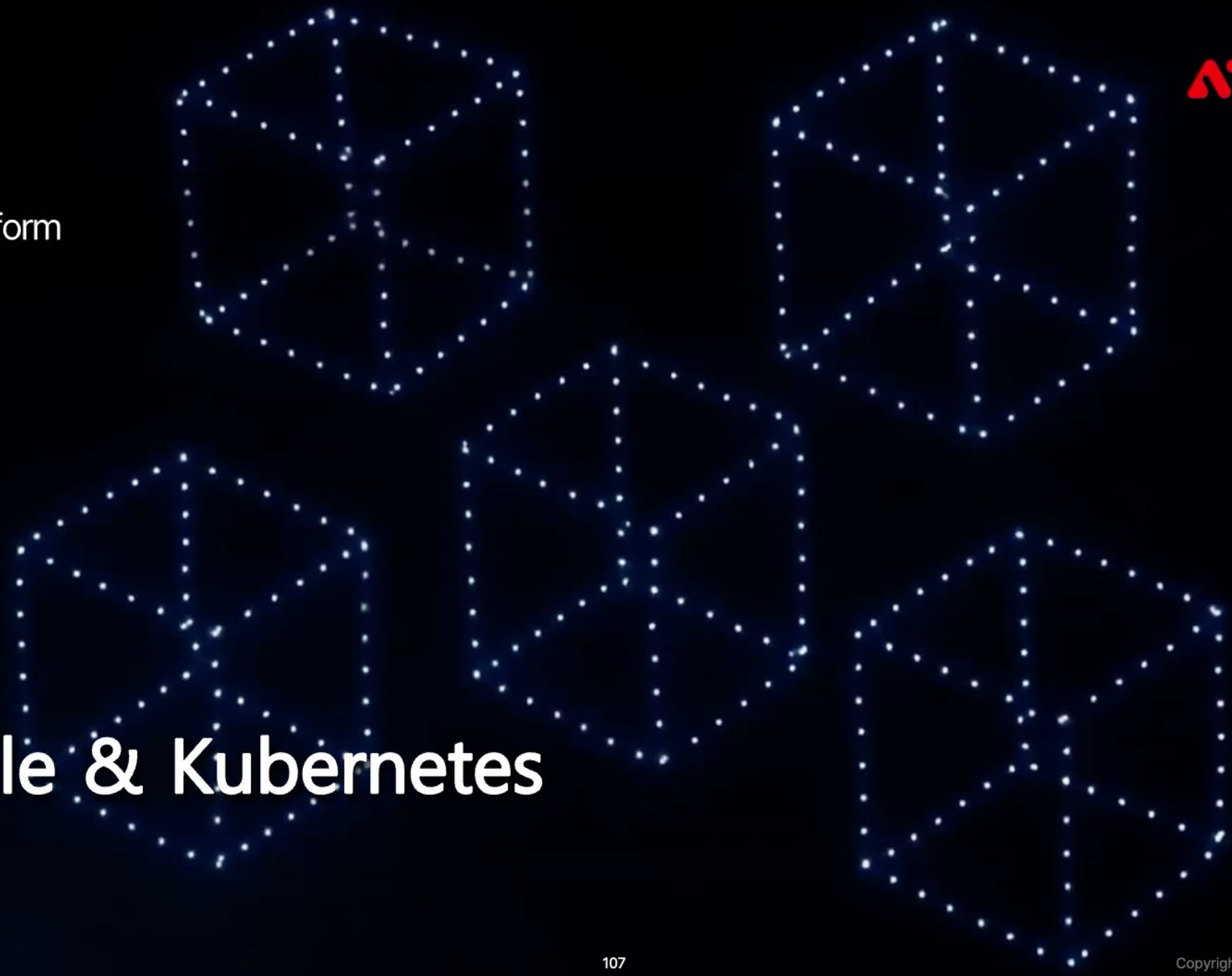
AINP (AI Native Platform) 교육 프로그램

MSAP.ai

AINP 환경에서 애플리케이션 개발 가이드 제공

MSAP.ai

- 컨테이너 어플리케이션의 개발이 어렵다
  - ➢ 구축이나 평가의 노하우로부터 가이드를 작성
  - ➢ 어플리케이션의 개발 방법을 가이드화
    - CI/CD 방법 가이드화
    - PaaS 를 이용해 개발 환경, 테스트 환경, staging 환경, 운영 환경 구축



Google & Kubernetes

MSAP.ai

AINP (AI Native Platform) 이란?

MSAP.ai

AI Native Platform

인프라 운영의 변화

AINP (AI Native Platform) Products

openmaru

openmaru iAP

애플리케이션을 위한 지능형 미들웨어 플랫폼

Includes :

- AINPCogentAI
- AINPCluster
- AINPAPM
- AINPApache/Tomcat

AINP

지능형 컨테이너 오케스트레이션 플랫폼

Adds :

- AINPObservability
  - Log, Metric, Trace, Profiling
- AINP (AI Native Platform)
  - Kubernetes Platform
  - DevOps
  - CI/CD

MSAP.ai

AIaaS를 포함한 MSA 와 클라우드 네이티브 애플리케이션을 위한 플랫폼

Adds :

- AIaaS
  - RAG, LLM, AI Framework
- MSA Accelerator
  - DDD Designer, Application Designer, MSA Framework

클라우드 네이티브에 최적화된 MSA 플랫폼을 원하신다면?

https://www.msap.ai

E. hello@msap.ai

MSAP.ai