

# 클라우드 네이티브 이해와 업무 중심의 모니터링 방안

# 디지털플랫폼정부는 '클라우드 네이티브'로 구축하는 게 핵심이다



전자신문 etnews Conference allshowTV Eng

경제·금융 전자·모빌리티 통신·미디어·게임 소재·부품 SW·보안 산업·에너지·환경 플

## [기획]행안부, 클라우드 네이티브 확산 위한 지원사업 추진

발행일: 2022-06-29 16:00 지면: 2022-06-30 18면

클라우드 네이티브는 클라우드 성숙도 단계 중 최고 단계로, 클라우드의 기능과 장점을 최대한 활용해 애플리케이션을 구축·실행하는 것을 의미한다.

## 공공클라우드 전환 로드맵 손질...부처별 추진·2030년 완료

류태중 | 입력 2023. 5. 3. 15:32 | 수정 2023. 5. 4. 09:01

### 공공 클라우드 네이티브 전환 로드맵

<b>추진 배경</b>	<ul style="list-style-type: none"> <li>정부 재정 투자 방향 변화</li> <li>보안인증제 개편</li> <li>신기술 보편화 등</li> </ul>
<b>추진 환경</b>	<ul style="list-style-type: none"> <li>행안안전부 전환 사업 예산 축소</li> <li>정부 정책 '민간 클라우드 우선 이용' '클라우드 네이티브 우선 적용'으로 발전 등</li> </ul>
<b>추진 방향</b>	<ul style="list-style-type: none"> <li>범정부 정보자원 등록·관리시스템에 등록된 모든 시스템의 클라우드 네이티브 전환</li> </ul>
<b>전환 기간</b>	<ul style="list-style-type: none"> <li>2024년부터 2030년까지 7개년 추진</li> </ul>

## [테크&포커스] 정부·공공 시스템도 `클라우드 최첨단화`... `디플정`의 도전

입력: 2023-04-16 16:07 | 팽동현 기자

尹정부 핵심과제 청사진 공개  
민첩한 개발·유연한 확장 가능  
초거대 AI 등 디지털기술 도입  
홈택스 등 사이트 통합도 추진

디지털플랫폼정부 기대 효과(2026년)

<b>대국민 서비스 강화</b>	<b>민·관 성장 플랫폼 구축</b>
<ul style="list-style-type: none"> <li>공공서비스 1500여종 연계·통합</li> <li>해택 알리미 총 1021종 제공</li> <li>첨부서류 제로화로 연 2조원 절감</li> </ul>	<ul style="list-style-type: none"> <li>SaaS 기업 1만개 육성</li> <li>AI 유니콘 기업 5개 육성</li> <li>DPG 수출 연 20억달러 달성</li> </ul>
<b>정부·공공 시스템 혁신</b>	<b>사이버보안·개인정보보호</b>
<ul style="list-style-type: none"> <li>공공부문 종이 사용량 50% 감축</li> <li>대상 시스템 70% 클라우드 네이티브 전환</li> <li>광역·기초로 이원화된 지자체 시스템 통합</li> </ul>	<ul style="list-style-type: none"> <li>주요 분야 마이데이터 유통체계 구축</li> <li>제로트러스트 등 새로운 보안체계 도입</li> <li>신기술 공공 적용, 보안산업 경쟁력 강화</li> </ul>

모놀리식 아키텍처와 마이크로서비스 아키텍처(MSA) 차이

모놀리식 아키텍처 VS 마이크로서비스 아키텍처(MSA)

\*설명: 단일 구조인 모놀리식 아키텍처와 달리 MSA는 각 서비스가 병렬·분산돼 유연성과 가용성이 뛰어나다

디지털플랫폼정부는 초거대AI를 비롯한 디지털 기술을 적극 도입, 칸막이를 없애고 '원팀 정부'로 거듭나는 것을 목표로 한다. 이를 위한 공공 플랫폼을 단순 클라우드 전환이 아닌, **클라우드를 클라우드 답게 쓸 수 있도록 '클라우드 네이티브'로 구축하는 게 이번 정책의 핵심**이다. 공공SW(소프트웨어)사업 고질병을 극복할 해법을 제시하면서 각종 미래 IT산업 육성의 요람이 될 것으로 기대된다.



조달청 디지털서비스몰 등록 

오픈마루



디지털서비스몰에서 **오픈마루** 를 검색하세요.

Chapter I | 2023년 IT 기술은?

Chapter II | 진짜 클라우드 - 클라우드 네이티브

Chapter III | IT 인프라 운영의 변화

Chapter IV | 클라우드에 필요한 모니터링



조달청 디지털서비스몰 등록 

오픈마루



디지털서비스몰에서 **오픈마루** 를 검색하세요.

Chapter I | 2023년 IT 기술은?

Chapter II | 진짜 클라우드 - 클라우드 네이티브

Chapter III | IT 인프라 운영의 변화

Chapter IV | 클라우드에 필요한 모니터링

“**메르세데스 벤츠**는 이제  
자동차 기업이 아닌 소프트웨어  
기업이며 자동차는 궁극의  
웨어러블이다.

- 올라 케레니우스, 메르세데스 벤츠 그룹 대표

“우리는 클라우드 네이티브로  
기술을 재구축하면서  
조직 운영 방식도 바꿨다.

- 넷플릭스 클라우드 부문 부사장 이즈라일레브스키

“**골드만삭스**는 **정보기술(IT)** 회사다.

- 골드만삭스 로이드 블랭크퍼드 (CEO)

“인간 중심의 **디지털 전략**이  
미래 **리테일 산업**의  
**성공의 열쇠**가 될 것입니다.

- Kevin Johnson, CEO of Starbucks

## Development Process



WATERFALL



AGILE



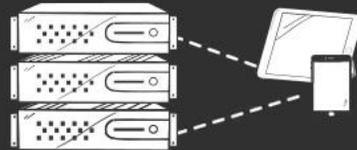
DEVOPS



## Application Architecture



MONOLITHIC



N-TIER



MICROSERVICES



## Deployment & Packaging



PHYSICAL SERVERS



VIRTUAL SERVERS



CONTAINERS



## Application Infrastructure



DATA CENTER



HOSTED



CLOUD



# SoR 에서 SoE 로 전환

## Systems of Record (SoR)

	Mainframe	Mini	PC
Time Frame	1960 - 1975	1975-1992	1992-2001
Data Types	Batch	Dept Process	Documents
First Movers	IBM, Unisys	DEC, Compaq	Microsoft, Dell, IBM

## Systems of Engagement (SoE)

	Internet	Mobile/Cloud	Connected
Time Frame	2001-2009	2010-2015	2016-
Data Types	Web Pages	User Interactions	IOT / AI
First Movers	Google, Microsoft	Facebook, Amazon, Apple	Airbnb, Uber



## 머신 중심에서 애플리케이션 중심 인프라로 변화

- 컨테이너화는 데이터 센터를 머신 중심에서 애플리케이션 중심으로 전환
  - 개발자와 운영팀에게 서버와 운영 환경에 대한 세부 사항을 추상화
  - 운영 중인 애플리케이션과 개발자에 미치는 영향을 최소화하면서 새로운 하드웨어 또는 운영 환경을 업그레이드하여 인프라팀에게 유연성을 제공
  - 서버의 CPU와 메모리 정보 뿐만 아니라 애플리케이션과 관련한 매트릭을 연결하여 오토 스케일링

### Machine Centric Infrastructure



### Application Centric Infrastructure





조달청 디지털서비스몰 등록 

오픈마루



디지털서비스몰에서 **오픈마루** 를 검색하세요.

Chapter I | 2023년 IT 기술은?

Chapter II | 진짜 클라우드 - 클라우드 네이티브

Chapter III | IT 인프라 운영의 변화

Chapter IV | 클라우드에 필요한 모니터링



조달청 디지털서비스몰 등록 

오픈마루



디지털서비스몰에서 **오픈마루** 를 검색하세요.

Chapter I

2023년 IT 기술은?

Chapter II

진짜 클라우드 - 클라우드 네이티브

- 클라우드는 클라우드 네이티브로 구현해야
- 컨테이너 기술과 가상화 기술
- 구글과 쿠버네티스
- CNCF 에서 클라우드 네이티브

Chapter III

IT 인프라 운영의 변화

Chapter IV

클라우드에 필요한 모니터링

# 클라우드 네이티브를 저해하는 요인

- 생각과 시스템을 **클라우드 네이티브** 하게 전환하지 못하면, 클라우드라도 개선이 없음



## 클라우드를 임대하여 사용하는 것일 뿐

- 가상머신과 스토리지를 임대하여 사용하고 있을 뿐, 기존의 인프라와 다르지 않음



## 클라우드 특징에 맞게 설계하고 운영 하지 않음

- 클라우드 특성을 이해하지 못하고 기존 인프라를 단순히 대체하여 설계
- 비용 부분에서만 정액제 클라우드로 전환하였으나, 벤더 종속성과 비용만 높아짐



## 인프라만 클라우드 일 뿐 조직은 그대로

- 기존의 개발팀과 운영팀이 수행하던 역할과 프로세스 그대로 운영



## 클라우드로 전환했으나 구인난과 고비용 구조로 더 큰 문제

- 클라우드를 이용하고 있음에도 불구하고 수작업 프로세스에서 벗어나지 못함
- 운영 인력 부족과 업무 효율성을 개선하지 못함

# Cloud Immigrant vs. Cloud Native



구분	Cloud Immigrant	Cloud Native
서비스 모델	가상화 기반 IaaS ( Infrastructure As A Service)	컨테이너 기반 PaaS ( Platform As A Service)
디자인	On Premise 에 구축된 시스템을 클라우드로 이전하여 운영	시작 단계부터 클라우드의 장점인 민첩성, 확장성 그리고 이동성을 최대한 활용할 수 있도록 설계
구현	특정 클라우드 벤더에 의존적인 설정이 있어 구축에 시간이 걸림	어떤 클라우드 환경에서도 빠르고 효율적으로 전환 ( Portability )
확장성	애플리케이션 업데이트가 수작업이기 때문에 장시간의 다운타임일 필요하고 Scale In/Out 이 어려움	컨테이너와 MSA 기반으로 서비스에 영향을 주지 않고, 업데이트가 필요한 서비스만 변경할 수 있으며, 서비스 단위의 Scale In/out 지원
비용	애플리케이션이 커질 수록 인프라 비용이 상승	인프라 부분의 종속성이 없어 비용이 저렴
유지보수	버전관리, 설치 그리고 구성관리가 수작업이며 복잡함	CI (Continuous Integration) / CD (Continuous Delivery )

# Native

- 네이티브(Native)의 사전적 의미는 '선천적인', '본래' 등이다.
- 클라우드 네이티브는 "클라우드가 '클라우드 다울 수 있도록' 애플리케이션을 구축, 실행하는 방식"

'어린이 또는 성인이 되어 언어를 배운 것이 아닌 태어나서 부터 특정 언어를 사용해 온 사람'

## 네이티브 스피커



'어린이 또는 성인이 되어 스마트폰을 접한 것이 아닌 유아기부터 스마트폰을 사용해 온 사람'

## 스마트폰 네이티브



'애플리케이션을 계획/설계할 때부터 클라우드 특징과 장점을 기반으로 개발/운영'

## 클라우드 네이티브



# 컨테이너는 클라우드 에서 Java 와 같이 벤더 종속성 해제

## 2000 년 - Java 를 통한 Vendor Lock-In 해제



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



# How does one build apps for the cloud?

 KVM  Google Cloud

 Xen Project

 aws

 Hyper-V

 Microsoft Azure

 Xen Project

 Alibaba Cloud

 Xen Project

 IBM Cloud

Hypervisor

Public Cloud



Virtual Machine

**Write once,  
run anywhere?**

# CNCF Cloud Native Definition v1.0

클라우드 네이티브 기술을 사용하는 조직은 현대적인 퍼블릭, 프라이빗, 그리고 하이브리드 클라우드와 같이 동적인 환경에서 확장성 있는 애플리케이션을 만들고 운영할 수 있다.

컨테이너, 서비스 메시, 마이크로서비스, 불변의 인프라스트럭처, 그리고 선언적 API가 전형적인 접근 방식에 해당한다.

이 기술은 회복성이 있고, 관리 편의성을 제공하며, 가시성을 갖는 느슨하게 결합된 시스템을 가능하게 한다.

견고한 자동화와 함께 사용하면, 엔지니어는 영향이 큰 변경을 최소한의 노력으로 자주, 예측 가능하게 수행할 수 있다.

Cloud Native Computing Foundation은 **벤더 중립적인 오픈소스 프로젝트 생태계**를 육성하고 유지함으로써 해당 패러다임 채택을 촉진한다.

우리 재단은 최신 기술 수준의 패턴을 대중화하여 이런 혁신을 누구나 접근 가능하도록 한다.



# Cloud Native Computing Foundation

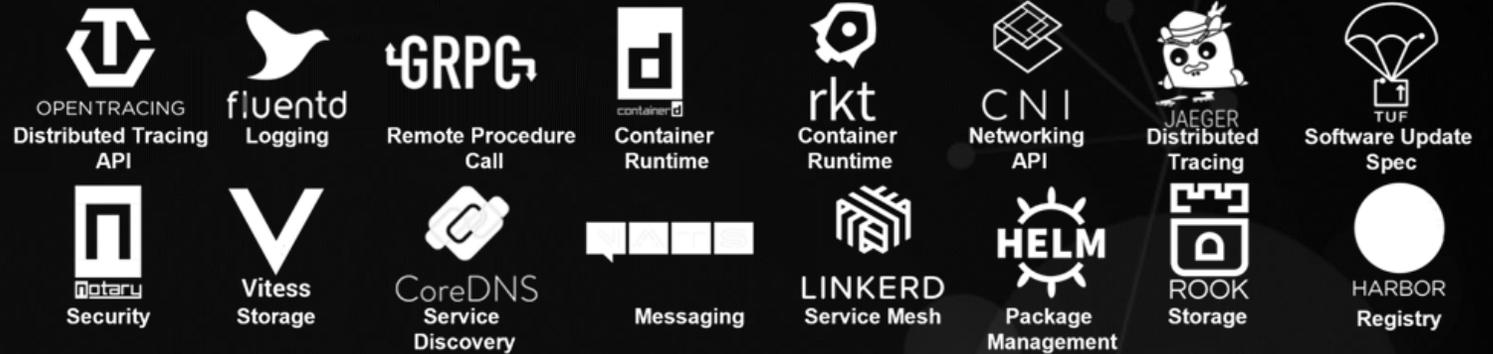


- Non-profit, part of the Linux Foundation; founded Dec 2015

## Graduated



## Incubating



- Platinum members:





조달청 디지털서비스몰 등록 

오픈마루



디지털서비스몰에서 **오픈마루** 를 검색하세요.

Chapter I

I

2023년 IT 기술은?

Chapter II

II

진짜 클라우드 - 클라우드 네이티브

- 클라우드는 클라우드 네이티브로 구현해야
- 컨테이너 기술과 가상화 기술
- 구글과 쿠버네티스
- CNCF 에서 클라우드 네이티브

Chapter III

III

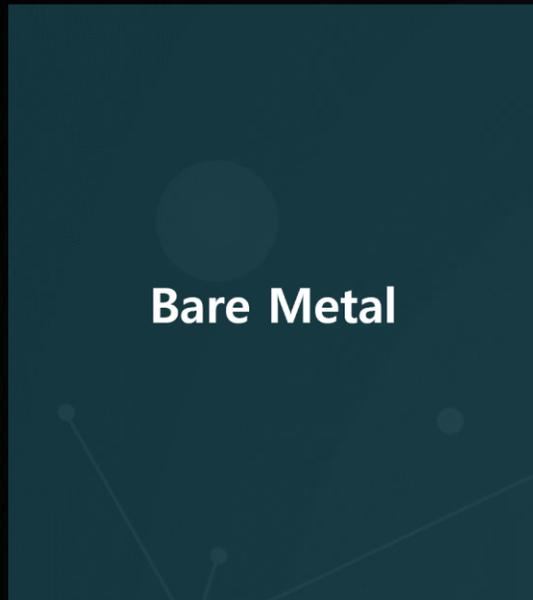
IT 인프라 운영의 변화

Chapter IV

IV

클라우드에 필요한 모니터링

# Evolution of Infrastructure Architectures



Bare Metal

Bare Metal



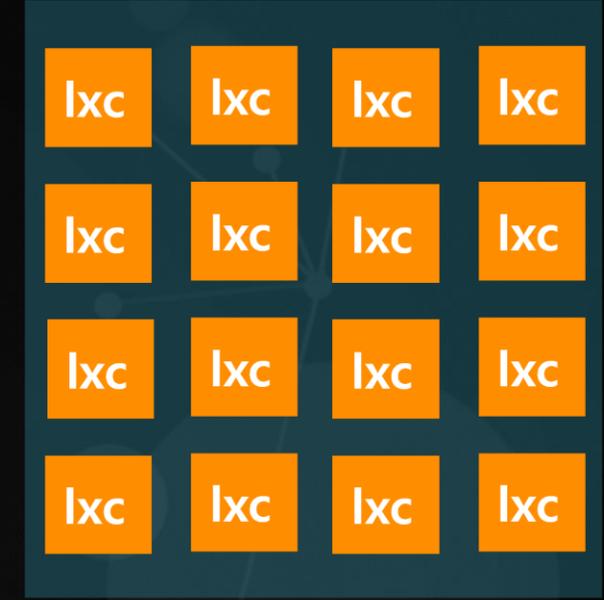
Virt

Virt

Virt

Virt

Virtualized



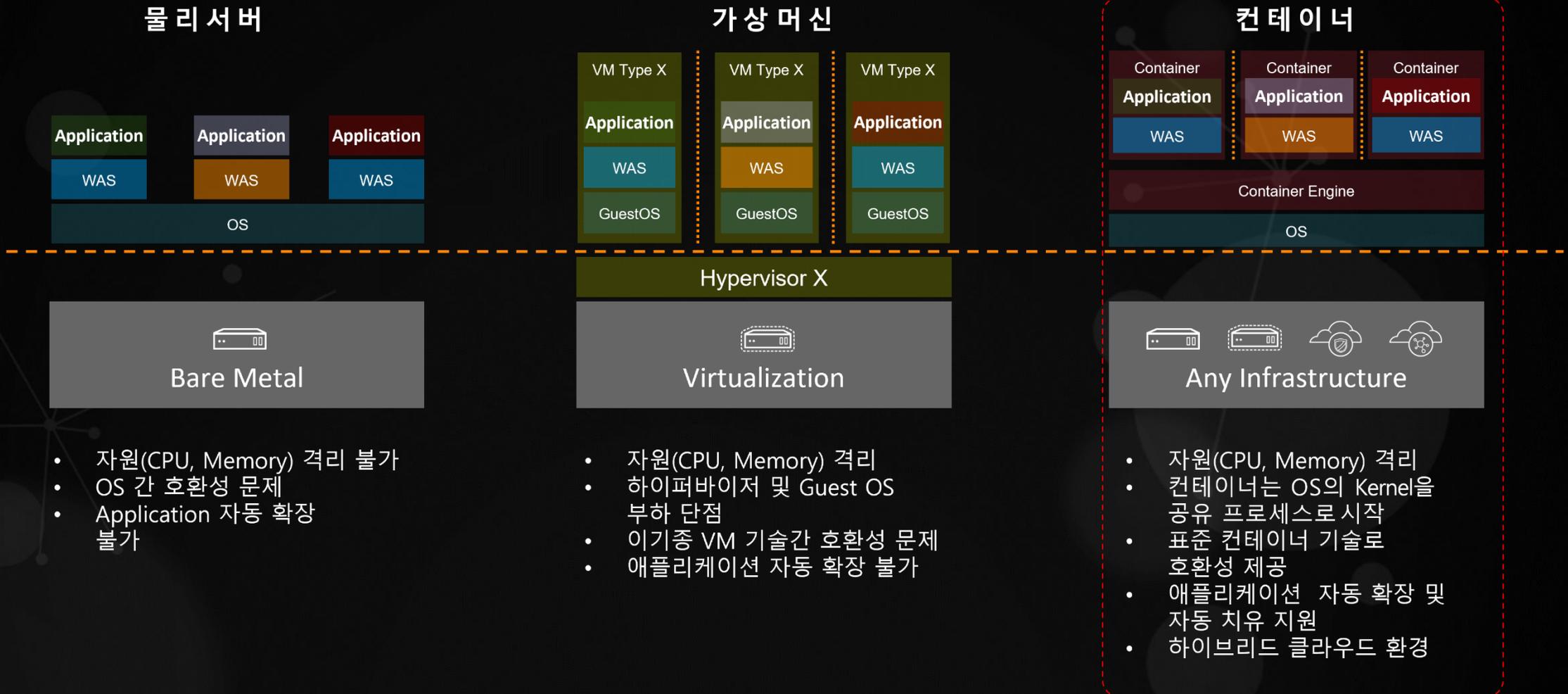
lxc

Containerized



# WHY CONTAINER ?

- 자원 효율성, 자원 격리, 호환성, Auto Scaling, DevOps, MSA, 관리 편의성



# 시작 시간 - Containers vs. VMs

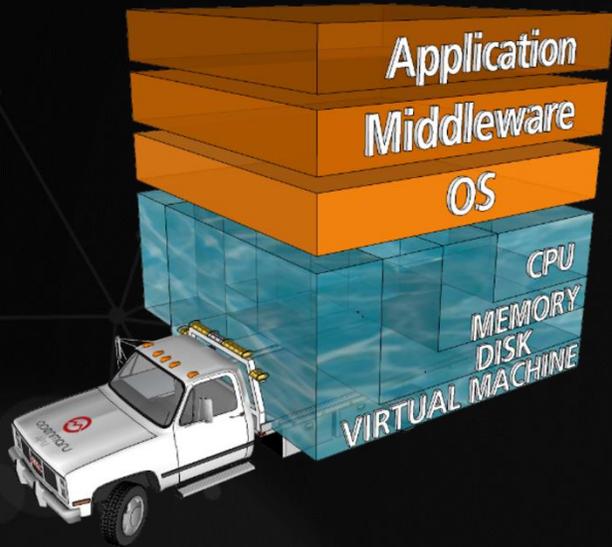
- 하드웨어 가상화는 CPU, 메모리, 하드 디스크 등의 하드웨어를 가상화하고 있기 때문에 하드웨어와 OS 부팅이 필요 (부팅에 분 단위 시간 소요)
- 컨테이너 형 가상화에서는 컨테이너 부팅 시 OS가 이미 시작된 상태 (프로세스 시작에 필요한 초 단위 시간 소요)



# 가상서버의 고질적인 문제점

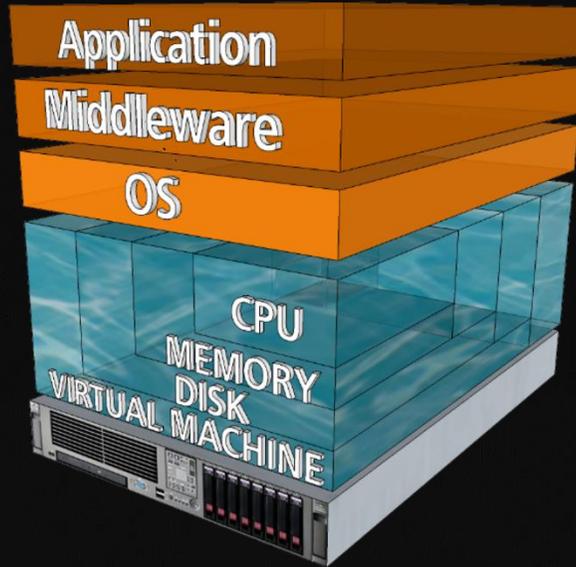
## 거대한 이미지 사이즈

- VM 을 템플릿 관리는 하지만  
사이즈가 커서 재사용성을  
높이기 어려움



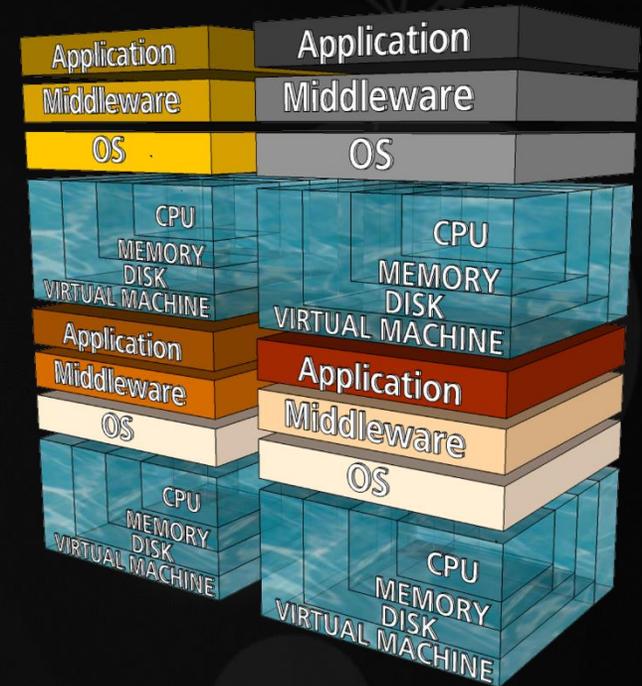
## 느린 시작 시간

- 부팅 시 Hypervisor – OS-  
미들웨어 – 애플리케이션까지  
실행 되어야함



## VM 간의 환경 불일치

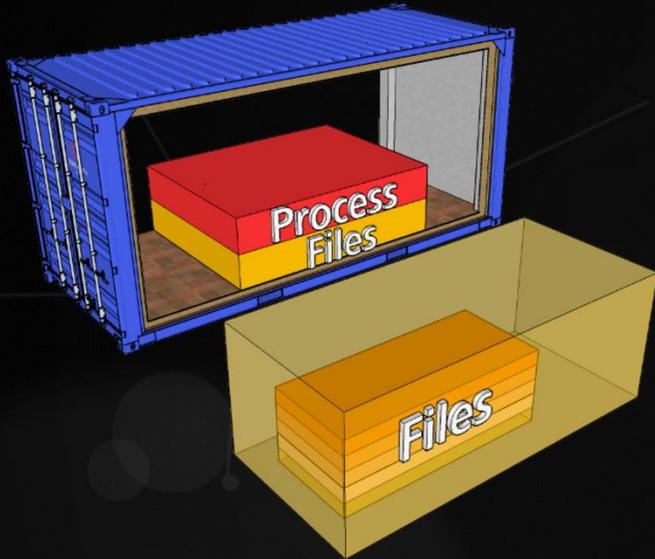
- VM 생성 후 개별로 변경 사항을  
관리하기 때문에 VM 간  
구성이나 환경이 불일치



# 컨테이너를 통한 가상서버 문제점 해결

## 작은 이미지 사이즈

- 컨테이너는 레이어 개념으로 이미지에 파일을 추가/삭제하여 관리함
- 레이어 사이즈를 최적화하여 이미지 사이즈를 최소화



## 빠른 시작 시간

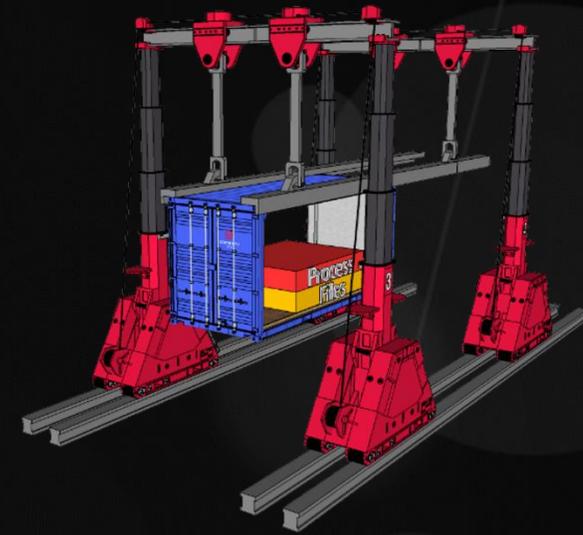
- 컨테이너는 분리된 프로세스 형식으로 OS 부팅이 필요 없기 때문에 부팅 시간을 최소화 할 수 있음



Container =  
Process

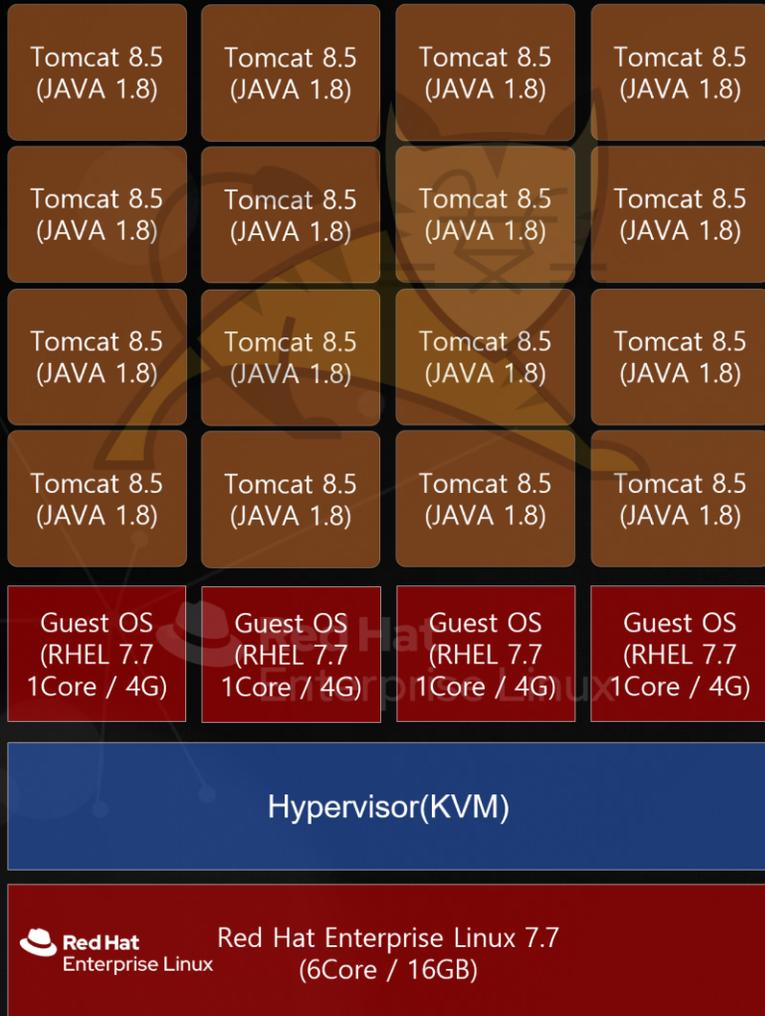
## 높은 이동성(Portability)

- 애플리케이션에 필요한 라이브러리나 의존 파일들을 이미지에 포함하기 때문에 환경에 의한 발행되는 문제가 거의 없음

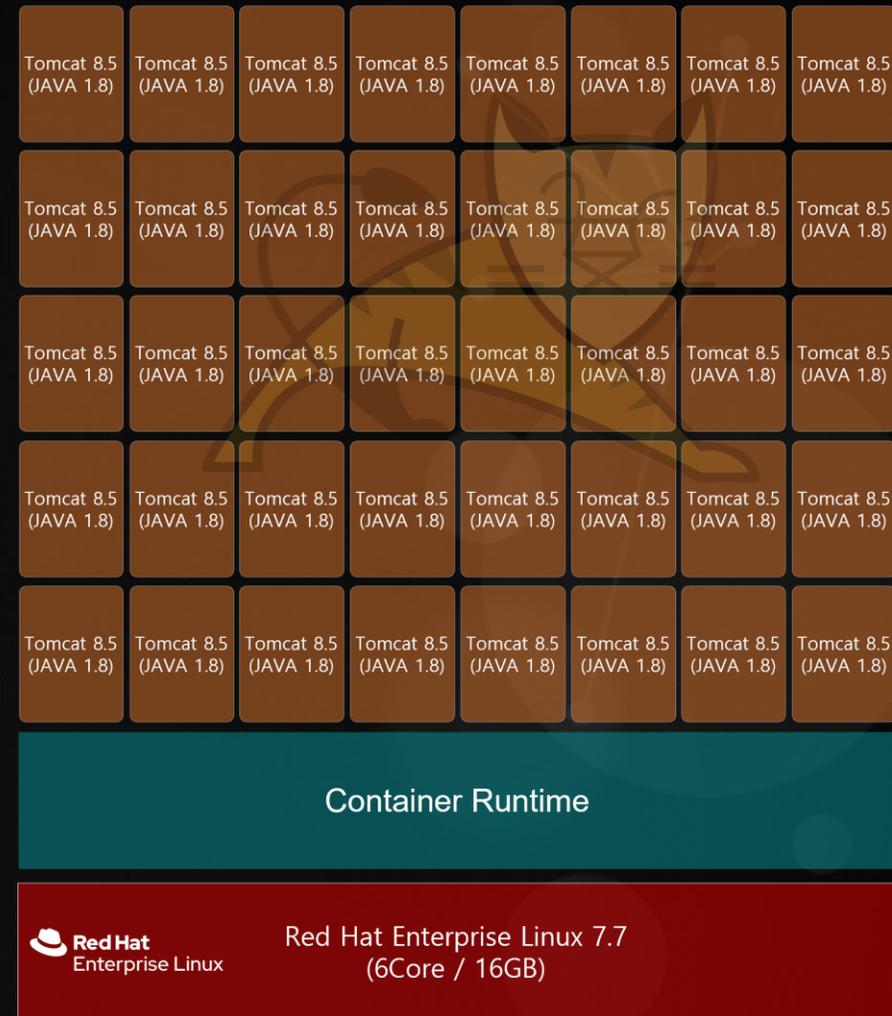


# 가상화와 컨테이너 집적도 비교

## 가상화 - 16개 Tomcat 인스턴스



## 컨테이너 - 40개 Tomcat 인스턴스





조달청 디지털서비스몰 등록 

오픈마루



디지털서비스몰에서 오픈마루 를 검색하세요.

Chapter I

I

2023년 IT 기술은?

Chapter II

II

진짜 클라우드 - 클라우드 네이티브

- 클라우드는 클라우드 네이티브로 구현해야
- 컨테이너 기술과 가상화 기술
- 구글과 쿠버네티스
- CNCF 에서 클라우드 네이티브

Chapter III

III

IT 인프라 운영의 변화

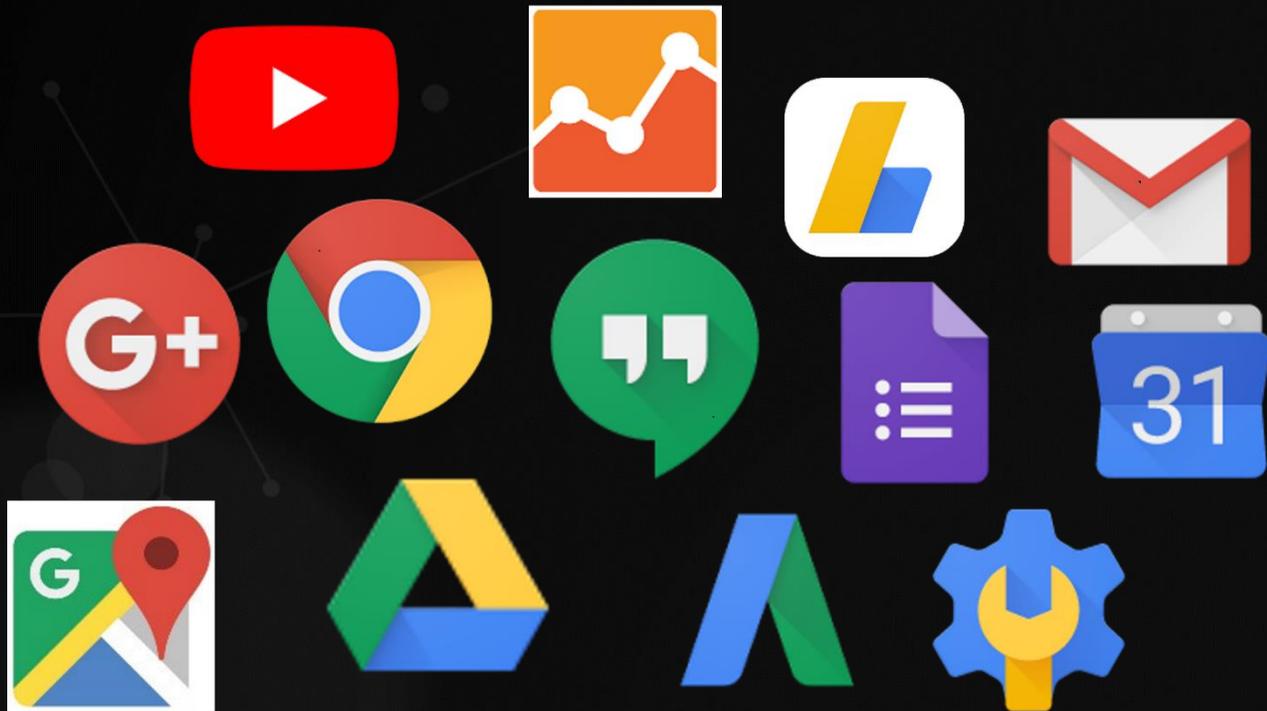
Chapter IV

IV

클라우드에 필요한 모니터링

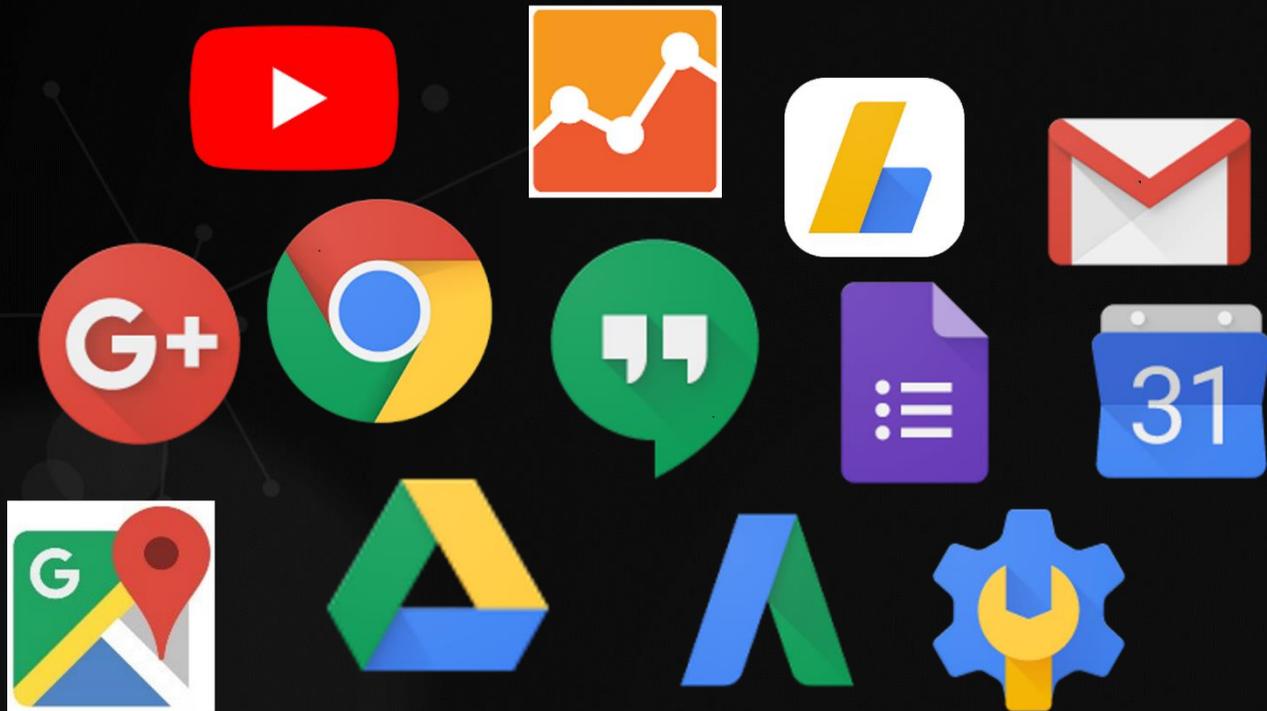
# Google 의 모든 서비스는 [ ] 에서 실행

- Gmail , 검색, 지도 ...
- MapReduce , GFS , Colossus ...
- Google Compute Engine 가상 머신도 [ ] 에서 실행!
- 매주 20 억개 이상의 [ ] 를 실행 중



# Google 의 모든 서비스는 **컨테이너** 에서 실행

- Gmail , 검색, 지도 ...
- MapReduce , GFS , Colossus ...
- Google Compute Engine 가상 머신도 **컨테이너** 에서 실행!
- 매주 20 억개 이상의 **컨테이너** 를 실행 중



# GOOGLE 과 컨테이너

- 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.



## About Kubernetes

- 쿠버네티스(K8s)는 컨테이너화된 애플리케이션을 자동으로 배포, 스케일링 및 관리해주는 오픈소스 소프트웨어
- 쿠버네티스", "쿠베르네티스", "K8s", "쿠베", "쿠버", "큐브"라고 부르며
- Go로 작성된 오픈 소스 , 오픈소스 S/W (Apache License 2.0) 라이선스
- 리눅스 재단 (Linux Foundation )산하 Cloud Native Computing Foundation (CNCF) 에서 관리
- 구글에서 개발하고 설계한 플랫폼으로서 사내에서 이용하던 컨테이너 클러스터 관리 도구인 "Borg"의 아이디어를 바탕으로 개발

"Kubernetes is open source-a contrast to Borg and Omega, which were developed as purely Google-internal systems. "

- Borg, Omega, and Kubernetes





“누군가가 나의 등잔의  
심지에서 불을 붙여가도  
내 등잔의 불은 여전히  
빛나고 있습니다.”

*미국의 정치가 토머스 제퍼슨*



조달청 디지털서비스몰 등록 

오픈마루



디지털서비스몰에서 오픈마루 를 검색하세요.

Chapter I

2023년 IT 기술은?

Chapter II

진짜 클라우드 - 클라우드 네이티브

- 클라우드는 클라우드 네이티브로 구현해야
- 컨테이너 기술과 가상화 기술
- 구글과 쿠버네티스
- CNCF 에서 클라우드 네이티브

Chapter III

IT 인프라 운영의 변화

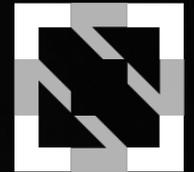
Chapter IV

클라우드에 필요한 모니터링

# Cloud Native 시대 2016 년 ~



- 2016 년 1 월에 정식 출범 한 Cloud Native Computing Foundation (이하 CNCF)는 진짜 클라우드 기술을 오픈소스로 해결하는 하는 것을 목표
- 애플리케이션을 실행하는 데 필요한 최적의 인프라 제공
  - 개발 한 것을 "즉시" "안정적으로" 제공 (비즈니스 우위)



CLOUD NATIVE  
COMPUTING FOUNDATION



# 공공 분야 - 클라우드 네이티브 정보시스템 구축사업 고려사항

- 클라우드 전환 및 도입 효과를 높이기 위해 단순한 기술 인프라 위주의 클라우드 도입보다
- 클라우드 환경에 최적화된 새로운 형태의 클라우드 네이티브 정보시스템 구축이 필요

클라우드 네이티브 정보시스템 구축을 위한  
**발주자 안내서**



행정안전부 **NIA 한국지능정보사회진흥원**

사업 추진 방향성과 사업 범위 작성 시 MSA, 컨테이너, 데브옵스 및 CI/CD 구성요소 관련 내용을 포함하여 클라우드 네이티브 사업임을 명시한다.

**[그림 5-2] 사업 추진 방향성 작성 예시**

**MSA, 컨테이너 구성요소를 포함하여 사업 추진 방향성 작성**

○ MSA 기반의 컨테이너 형태로 구현된 공간정보 서비스 기능(공간정보 표준 프레임워크)을 효율적으로 운영 관리하는 위한 개방형 공간정보 플랫폼 구축  
- 서비스 수요 증감에 따라 유연하게 컨테이너가 확장 및 축소가 가능한 운영관리 기능 및 컨테이너 동적 어부어 대응 상태 모니터링 기능 제공

[출처: 디지털 관영자원 통합관리시스템 재구축 및 운영 제안요청서, 한국관광공사]

**[그림 5-3] 사업 범위 작성 예시**

**MSA, 컨테이너, CI/CD 구성요소가 포함된 사업 범위 작성**

○ 다중화 기반 마이크로 서비스 구축  
- (마이크로 서비스 아키텍처 구축) 컨테이너 관리 기능, API 게이트웨이 관리 기능  
- (전자정부 표준프레임워크 적용) 실행환경 구성, 개발환경 구성, 운영환경 구성, 관리환경 구성  
- (인프라 구축) 인프라 가상화-자동화 구현, HW-SW 구축, 보안관리

○ (마이크로서비스) 잘 정의된 API를 통해 콘텐츠를 제공하는 콘텐츠 관리 기능 중심으로 시스템 구축

○ 마이크로서비스를 독립적으로 개발/배포/관리할 수 있는 프레임워크 제공  
\* 컨테이너 관리: 여러 대의 서버에서 여러 개의 컨테이너를 편리하게 관리하도록 서비스 메시, 컨테이너 오케스트레이션 등 자동화 기반의 컨테이너 배포 구현

○ 마이크로서비스 구현을 위한 가상화/자동화 환경을 제공  
\* 가상화: 물리적/논리적 서버 클러스터 구성을 통해 시스템 가용성 향상 및 가상 서버 복제 및 수평 확장을 통해 시스템 확장성 확보  
\* 자동화: 서비스 요청관리, 수요관리, 변경관리 등 사용자의 서비스 요청에 대한 해당 서비스를 사용자에게 제공하기 위해 다음과 같은 자동화된 서비스 제공관리 환경 구축

[출처: 디지털 관영자원 통합관리시스템 재구축 및 운영 제안요청서, 한국관광공사]

**[그림 5-5] 상세 요구사항 작성 예시**

요구사항 분류	클라우드 서비스 요구사항
요구사항 고유번호	CSR-003
요구사항 명칭	컨테이너 기반 서비스 메시 및 오케스트레이션 구현
요구사항 상세 설명	<ul style="list-style-type: none"> <li>여러 대의 서버에서 여러 개의 컨테이너를 편리하게 관리하도록 서비스 메시 기능 제공</li> <li>컨테이너를 적절한 서버에 배포하고 상태를 유지하기 위한 스케줄링</li> <li>여러 대의 서버를 1대의 서버처럼 관리하고, 가상 네트워크를 이용해 접근하기 위한 클러스터링</li> <li>컨테이너의 IP/포트정보를 서비스 레지스트리에 저장하며, 동적으로 변화하는 리소스의 위치를 API 게이트웨이가 검색하기 위한 서비스 디스커버리 기능 제공</li> <li>API 요청에 대한 최적의 경로를 지원하기 위한 다양한 API 라우팅 구현</li> <li>서비스 간 부하 분산을 위한 로드밸런싱</li> <li>오트스케일링 시 서버 수 지정, 서버의 사양 정의, 서버 실행 시간까지의 워밍업 시간 지정 등 트래픽 집중에 서버, 스토리지, 네트워크 등 인프라 자원의 자동 확장 및 축소를 자동화하여 서비스 상태에 따른 적정 서버 유지를 통해 유연한 서비스를 제공하도록 오트스케일링 지원</li> <li>특정 서비스에 오류가 발생하거나 실행 실패 시 신속하게 이전 버전으로 되돌아가도록 복구(Rollback) 기능 지원</li> <li>표준화된 로그 이벤트 수집 및 분석, 서비스 간 호출 추적 및 성능 관리 등 로깅 및 로그 분석 등</li> </ul>

[출처: 나라장터, 클라우드 네이티브 관련 제안요청서 참조]

요구사항 분류	클라우드 서비스 요구사항
요구사항 고유번호	CSR-004
요구사항 명칭	API 게이트웨이 관리 기능
요구사항 상세 설명	<ul style="list-style-type: none"> <li>API 호출을 위한 토큰 발급 및 인증, 엔드포인트별 API 호출 인증 및 인가, 접근 정책(예, 특정 클라이언트의 API 호출 불허에 의한 접근제한 기능) 등 API 인증 및 인가 처리</li> <li>동일 API를 클라이언트나 마이크로서비스에 따라 다른 엔드포인트를 통해 서비스를 제공하도록 API 라우팅을 지원하고, 동일 API를 여러 개의 클라이언트/마이크로서비스 별도 엔드포인트 제공</li> <li>로그, 인증 등 공통 기능을 중복 개발 또는 처리하지 않도록 요청과 응답의 표준화 및 공통 로직 처리</li> <li>동일 API를 HTTP, REST, XML, 웹 서비스 등 클라이언트와 마이크로서비스별로 상이한 프로토콜로 서비스하기 위한 프로토콜 변환 처리</li> <li>동기, 비동기 등 API를 호출하는 메시지 패턴을 변경할 수 있도록 메시지 호출 패턴 변환 기능 제공</li> <li>호출 횟수, 전송 용량, 네트워크 대역폭 등 서비스 레벨을 클라이언트나 마이크로 서비스별로 조정할 수 있도록 QoS(Quality of Service) 설정 기능 제공</li> <li>API 호출 패턴 분석, API 호출 실행 및 접근 상태 분석, 요청 IP/클라이언트/일시 등 API 호출에 대한 로깅 및 모니터링 등</li> </ul>

[출처: 나라장터, 클라우드 네이티브 관련 제안요청서 참조]

# 클라우드 네이티브 정의

- 클라우드 네이티브(Cloud Native)는 “클라우드의 장점을 최대한 활용하여 정보시스템을 구축 및 실행하는 환경”
- 2015년 최초로 클라우드 네이티브라는 용어를 사용한 리눅스 재단은 CNCF(Cloud Native Computing Foundation) 재단을 설립하여 클라우드 네이티브 관련 기술을 정의하고 오픈소스를 관리



Source - 클라우드 네이티브 정보시스템 구축을 위한 발주자 안내서

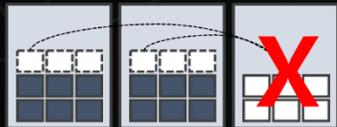
# Cloud Native Computing으로 전환 효과

- Cloud Native Computing 환경은 클라우드가 제공하는 민첩성, 가용성, 확장성의 장점을 어플리케이션/서비스의 개발, 운영, 관리에 적용하여 기존 컴퓨팅 환경을 최적화 함



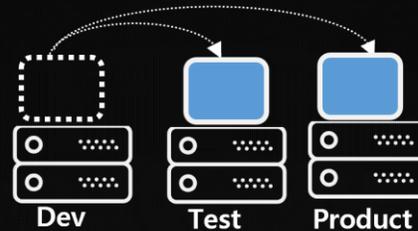
## On Demand Delivery

- 필요한 컴퓨팅 자원을 즉시 제공



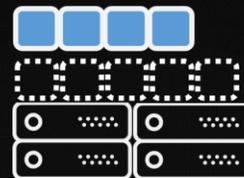
## Self Recovery

- 비정상 어플리케이션 재시작
- 노드의 장애 발생시 정상 서버 노드로 자동 재배포



## Consistency & Continuous

- 이미지 기반으로 구성, 배포 효율화
- 개발과 운영 환경의 일관성



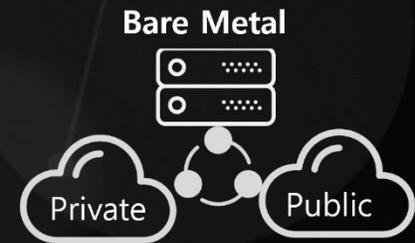
## Application Scaling

- VM 단위가 아닌 어플리케이션 단위의 오토스케일링



## Rolling Update

- 업그레이드 또는 패치 시 다운 타임은 제로 또는 최소화



## Portable

- 멀티/하이브리드 클라우드 기반 어플리케이션/서비스 운영



조달청 디지털서비스몰 등록 

오픈마루



디지털서비스몰에서 **오픈마루** 를 검색하세요.

Chapter I | 2023년 IT 기술은?

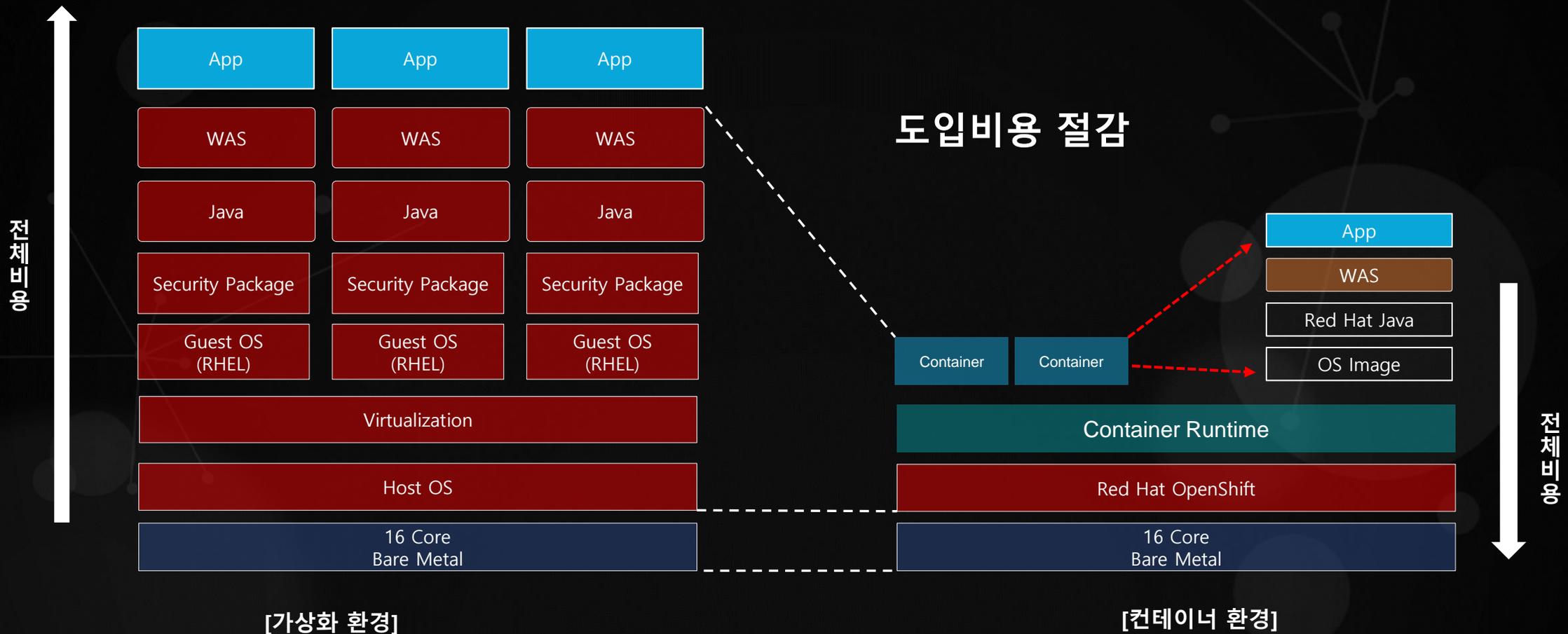
Chapter II | 진짜 클라우드 - 클라우드 네이티브

Chapter III | IT 인프라 운영의 변화

Chapter IV | 클라우드에 필요한 모니터링

# 가상화 VS 컨테이너 비교 - 비용적인 측면

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



# 아직도 WAS BMT 나 POC를 하시나요?

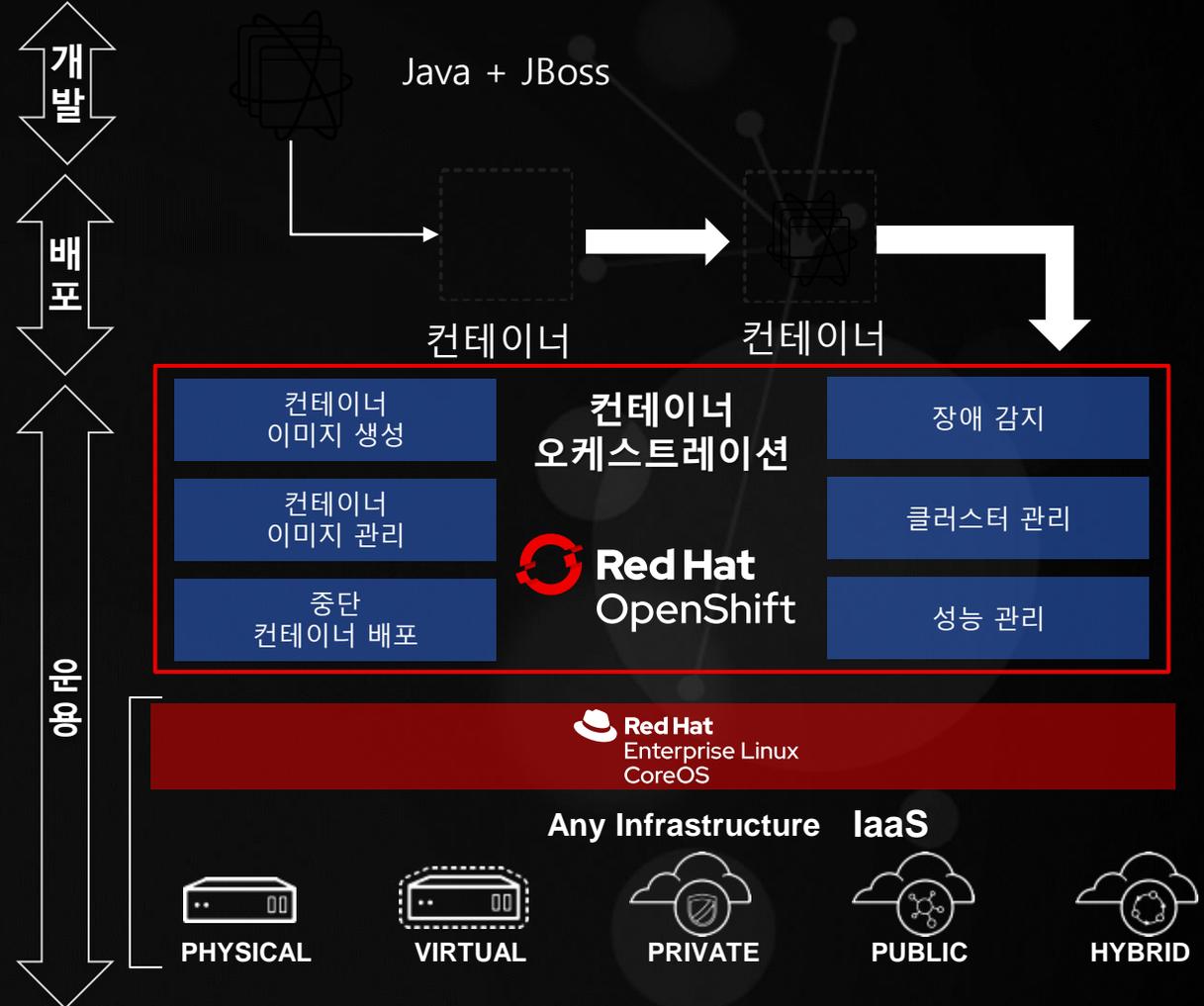
- WAS 의 가용성/확장성/성능/신뢰성 등의 기능은 플랫폼으로 이전
- 더 가볍고 더 빠르고, 자동화에 친화적인 WAS 로 전환

## WAS 제품 BMT



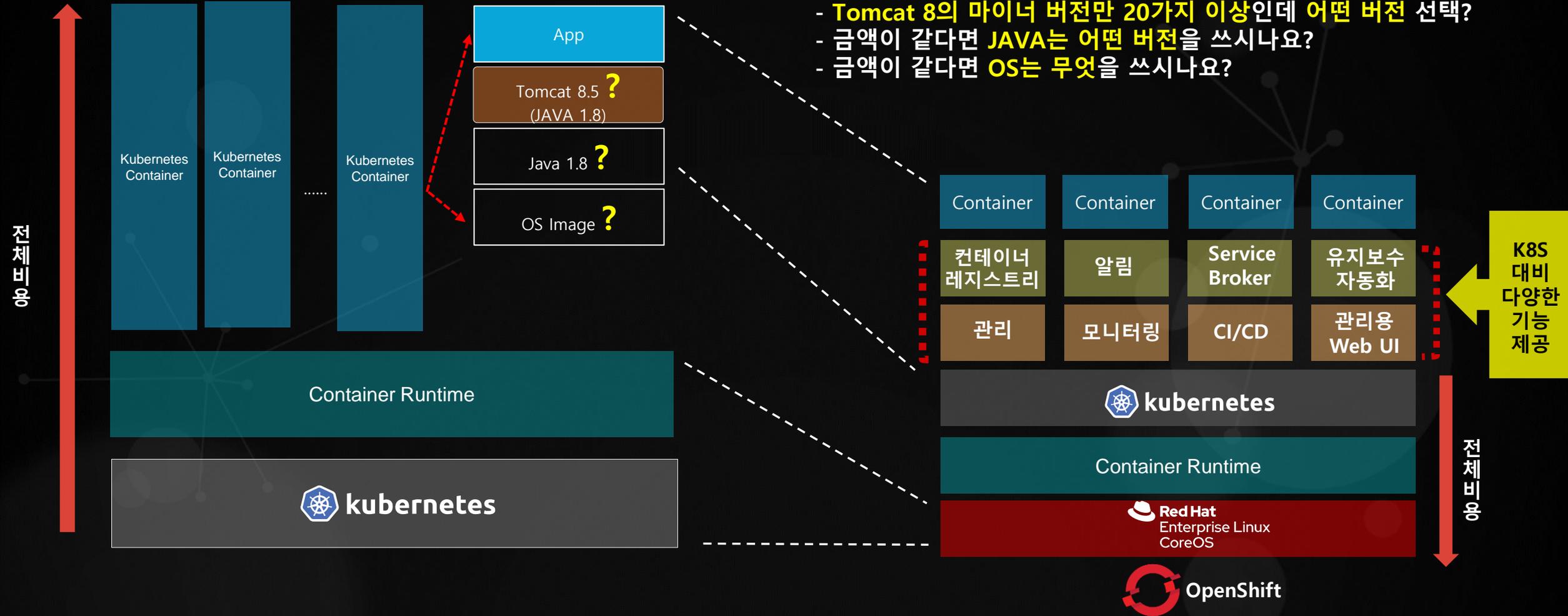
## RASP 에 의한 WAS평가

- 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



# 운영 환경에 부족한 Kubernetes vs 완벽한 OpenShift

Openshift는 Tomcat/JAVA/OS 이외에도 많은 오픈소스 유지보수 제품들이 포함되어 있습니다.





조달청 디지털서비스몰 등록 

오픈마루



디지털서비스몰에서 **오픈마루** 를 검색하세요.

Chapter I | 2023년 IT 기술은?

Chapter II | 진짜 클라우드 - 클라우드 네이티브

Chapter III | IT 인프라 운영의 변화

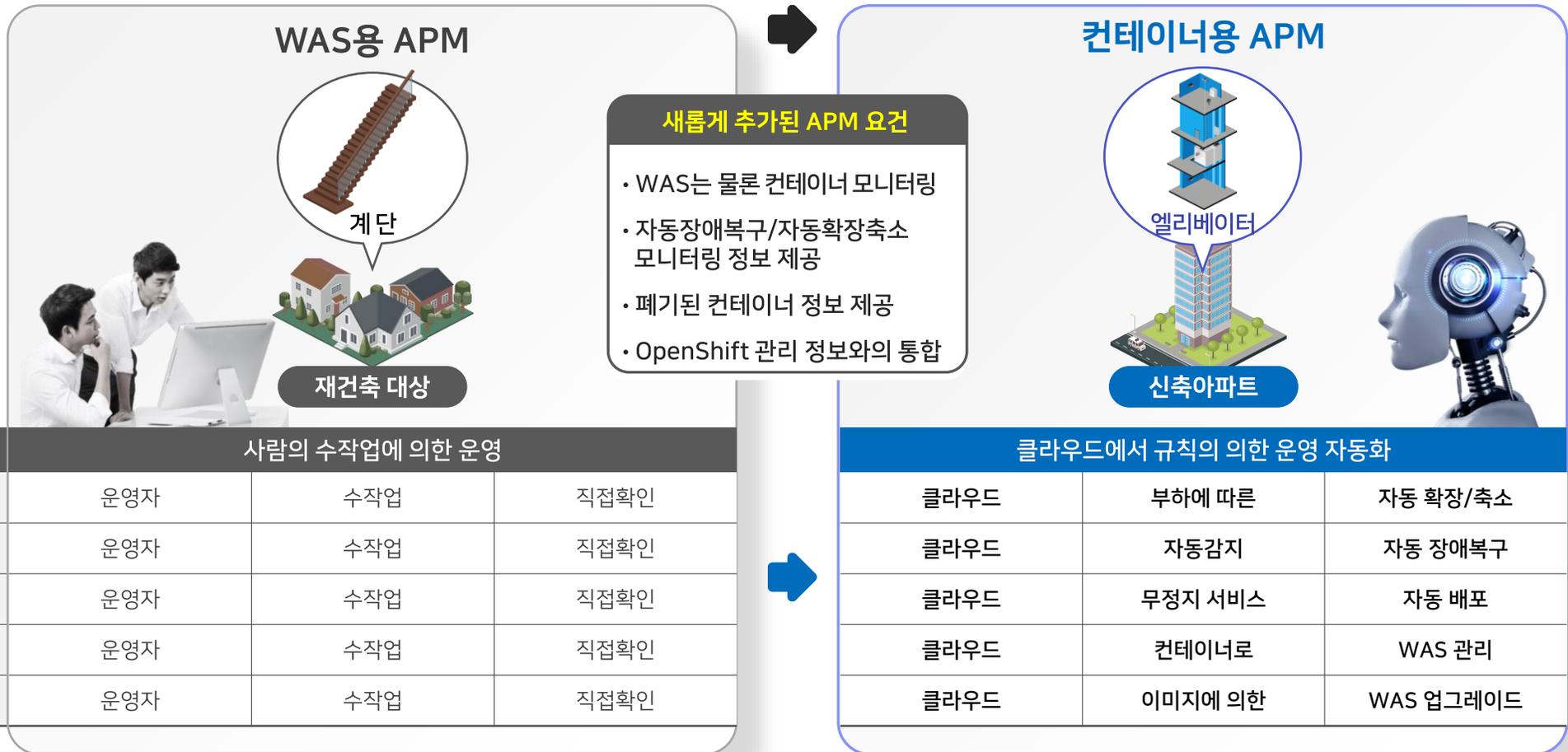
Chapter IV | 클라우드에 필요한 모니터링



# 클라우드 전환에 따른 APM 요구사항의 변화

## 클라우드에서는 APM 모니터링 대상이 WAS에서 컨테이너로 변경

- 컨테이너 단위로 WAS에 대한 확장/축소, 장애 복구, 업그레이드, 패치 작업하여 WAS와 함께 컨테이너를 모니터링해야 합니다.
- 기존 물리서버나 가상서버와는 달리 컨테이너는 휘발성으로 상태를 가지고 있지 않습니다.





# 머신러닝 기반 장애 경고 알람

## OPENMARU APM은 머신러닝 기반으로 임계치에 도달하기 전에 이상 징후를 사전에 예측



고려사항

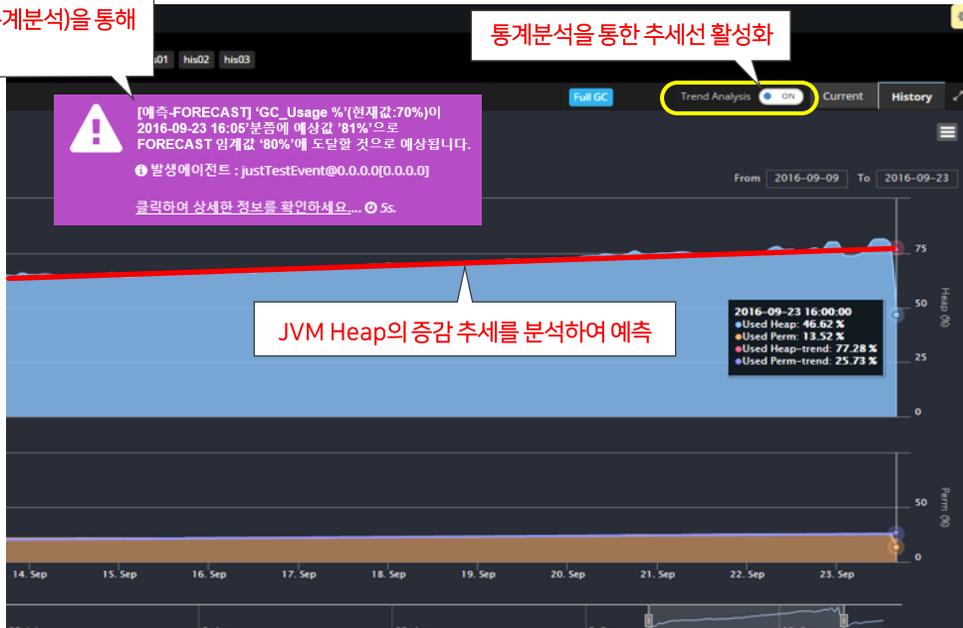
- ⊙ 사용자가 다양한 성능과 장애 정보들의 임계치나 추이 파악이 어려움
- ⊙ APM에서 제공하는 모든 정보를 확인할 수 선제적 대응과 장애와 성능관리가 어려움



방안

- 실시간 머신러닝(통계분석)을 통해 앞으로 몇 분 후에 관리자가 설정한 임계 값에 도달할 것이라는 예측 이벤트를 통보합니다.
- APM은 현재 상태 모니터링 및 추세분석을 통해 임계값 도달 시점을 예측하여 이벤트를 통보 한다.

머신러닝(통계분석)을 통해 예측 알림



Java VM의 Heap 사용량 증가 추이를 분석하여 GC 임계값 초과 시점에 대한 정보 제공

### ☑ 장애 경고 알람 대상 항목

항목	이벤트 발생 항목
WAS	<ul style="list-style-type: none"> <li>JVM Heap Usage</li> <li>APDEX</li> <li>Error Rate</li> <li>JVM Perm Usage</li> <li>Database Response Time</li> <li>Pending Transactions</li> <li>GC Usage</li> <li>Database Connection Pool Usage</li> </ul>
WEB	<ul style="list-style-type: none"> <li>WEB Traffic</li> <li>Worker Usage</li> </ul>
System	<ul style="list-style-type: none"> <li>Memory Usage</li> <li>Disk Usage</li> <li>Memory Swap Usage</li> <li>CPU Usage</li> <li>Network Packet Error Rate</li> </ul>
Cubrid	<ul style="list-style-type: none"> <li>CAS Usage</li> </ul>

# 클라우드 장애관리와 성능상의 특화기능



➡ APM은 모니터링을 넘어 클라우드 환경에서 장애에 대한 원인규명과 분석을 할 수 있는 기능이 있어야 합니다.



고려사항



고려사항

- ⊙ 장애 발생 시 엔지니어가 도착하기 전에 장애 분석이 필요함
- ⊙ APM 사용시 APM 에서 제공하는 정보가 궁금할 때 즉각 문의 할 수 있어야 함



방안

- 애플리케이션 성능 모니터링 및 진단, 장애 원인 분석을 통해
- 서비스를 최적의 상태로 운영할 수 있는 모니터링 솔루션이다.

### Java 스레드 덤프 분석기

#	Lin.	Type	Name	State	URL	Durati...	CPU Tl.	Class	At
1	696	Active Thread	ap-0.0.0.0-8009...	RUNNABLE			1,309	0.0	java...
2	871	Active Thread	ap-0.0.0.0-8009...	RUNNABLE			123,355	0.0	java...
3	936	Active Thread	ap-0.0.0.0-8009...	RUNNABLE			156,825	10.0	java...
4	1,556	Active Thread	ap-0.0.0.0-8009...	RUNNABLE			183,887	0.0	java...

Lock을 추적가능, URL 정보표시

### Java 메모리 누수 분석기

#	Num	Classname	Bytes [%]	Bytes	Instances
1	1	char[]	35.25%	35.2 MB	258,535
2	2	byte[]	8.62%	8.6 MB	10,388
3	3	java.lang.String	6.11%	6.1 MB	253,859
4	4	java.util.HashMap\$Node	5.55%	5.5 MB	173,073
5	5	java.util.jar.JarFile\$JarFileEntry	5.06%	5.0 MB	52,595

Java 메모리를 점유한 객체 분석/비교

### 네트워크 상태 분석기

#	Proto	Recv.	Send-Q	Local Address	Foreign Address	State	PID/Program name
1	tcp	0	0	127.0.0.1:9999	0.0.0.0:*	LISTEN	509/java
2	tcp	0	0	10.6.3.224:5455	0.0.0.0:*	LISTEN	509/java
3	tcp	0	0	10.6.3.224:7600	0.0.0.0:*	LISTEN	509/java
4	tcp	0	0	0.0.0.0:8080	0.0.0.0:*	LISTEN	509/java

Java 프로세스,시스템이 사용중인 네트워크 분석

### 오픈파일 분석기

#	Cam.	PID	User	FD	Type	Device	Size/Off	Node	Name
1	java	509	jboss	cwd	DIR	253,28	85	4199509	/home/jboss
2	java	509	jboss	txt	DIR	253,28	261	1027	/
3	java	509	jboss	rd	REG	253,28	7376	6295581	/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.1...
4	java	509	jboss	mem	REG	253,28	411090	8420366	/opt/esp/standalone/tmp/dfu/deployme...
5	java	509	jboss	mem	REG	253,28	24956	4372527	/opt/esp/standalone/tmp/dfu/deployme...
6	java	509	jboss	mem	REG	253,28	2666695	90124	/opt/esp/standalone/tmp/dfu/deployme...
7	java	509	jboss	mem	REG	253,28	220526	29437985	/opt/esp/standalone/tmp/dfu/deployme...

Java 프로세스가 오픈한 파일 분석

### 시스템 프로세스 분석기

750 Total / 749 Sleeping, 1 Running, 0 Zombie, 0 Stopped

CPU Usage: User: 10.0%, System: 0.0%, Nice: 0.0%, Wait: 0.0%, Idle: 90.0%

#	PID	User	Start Time	Mem	RSS	Mem	Shared	State	CPU Time	CPU
1	115,263	root	2019-01-14 17:53...	4.2 GB	254.5 MB	10.5 MB	0	Sleeping	05:37:11.2	23.1%
2	78,365	?	2019-01-18 18:32...	6.3 GB	404.8 MB	20.1 MB	0	Sleeping	00:02:20.0	21.5%

시스템의 프로세스 CPU, 메모리 사용량 분석/비교

### 데이터 추세 분석

Datasource Pool Count  
higher values represent higher usages

Number of Pools

13:42 13:43 13:44 13:45 13:46

과거 데이터의 증감 추세를 분석하는 기능



# 사용자 관점의 주기적인 서비스 품질점검 서비스

## ➡ 서비스 접속이 안된다고 불편이 접수되는데, 내부에서는 인지하지 못하는 상황을 겪어보진 않으셨나요?





고려사항

- ⊙ 인터넷망 사용자의 접속이 문제가 있는 경우 메신저, SMS, 이메일 등으로 상황전파가 되어야함
- ⊙ 사용자 관점의 SLA 를 일간/주간/월간 등의 주기로 보고서를 생성해서 전달해야 합니다.



- 사용자 입장에서 주기적인 서비스 이상유무를 판단합니다. (예를 들어, 서버는 이상이 없는데 네트워크 장비 등의 문제로 서버 접속이 안되는 경우)
- 서비스별로 URL 기준으로 정의하며, 요청 성공율, 가용성, 평균응답시간, 호출 횟수 등의 서비스 정상 유무를 확인 합니다.

브라우저 엔진 활용하여 주기적으로 체크



### ☑ 헬스체크 기능 활용

#### 헬스체크 기능

- 서비스에 대한 주기적인 생사 확인
- 평균 응답 시간 추이 확인
- 특정 기능/페이지 상태 점검
- 재 배포 후 다수의 페이지에 대한 일괄 점검
- 사용자 관점의 서비스 상태 확인
- 인프라 구간별 서비스 상태 확인
- 복합적인 점검 수행과 오류에 대한 사용자 알림
- 스크립트를 이용하여 프로세스/파일/자원 사용량 등을 주기적으로 모니터링



# 멀티 애플리케이션 운영 환경에서 클라우드 모니터링

## 여러 애플리케이션을 동시 운영 시 자원을 효율적으로 활용할 수 있을까요?



고려사항



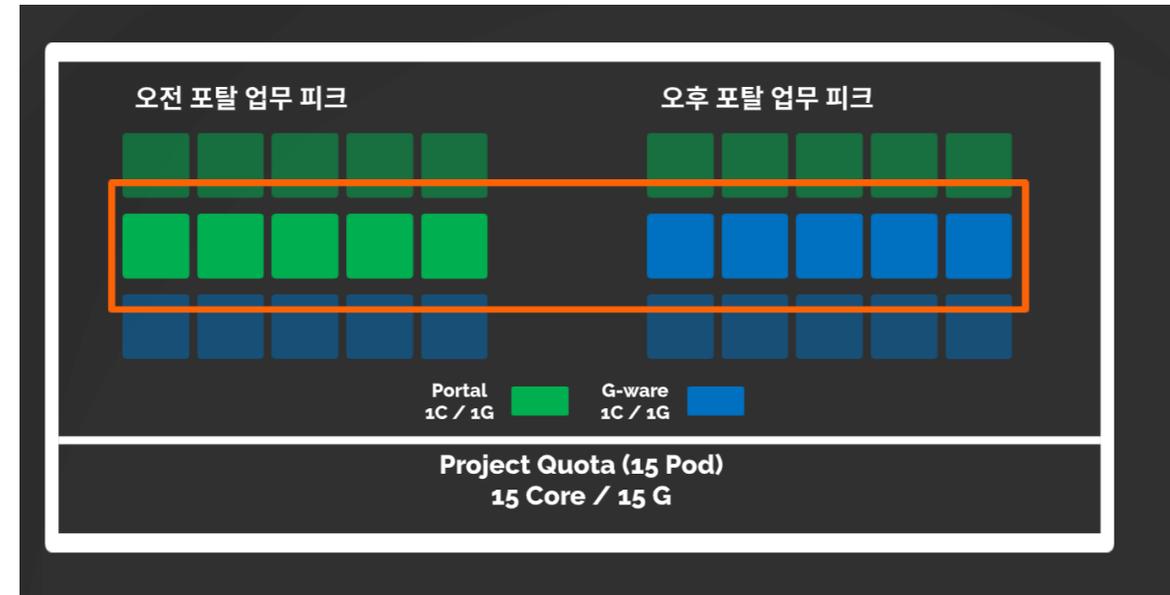
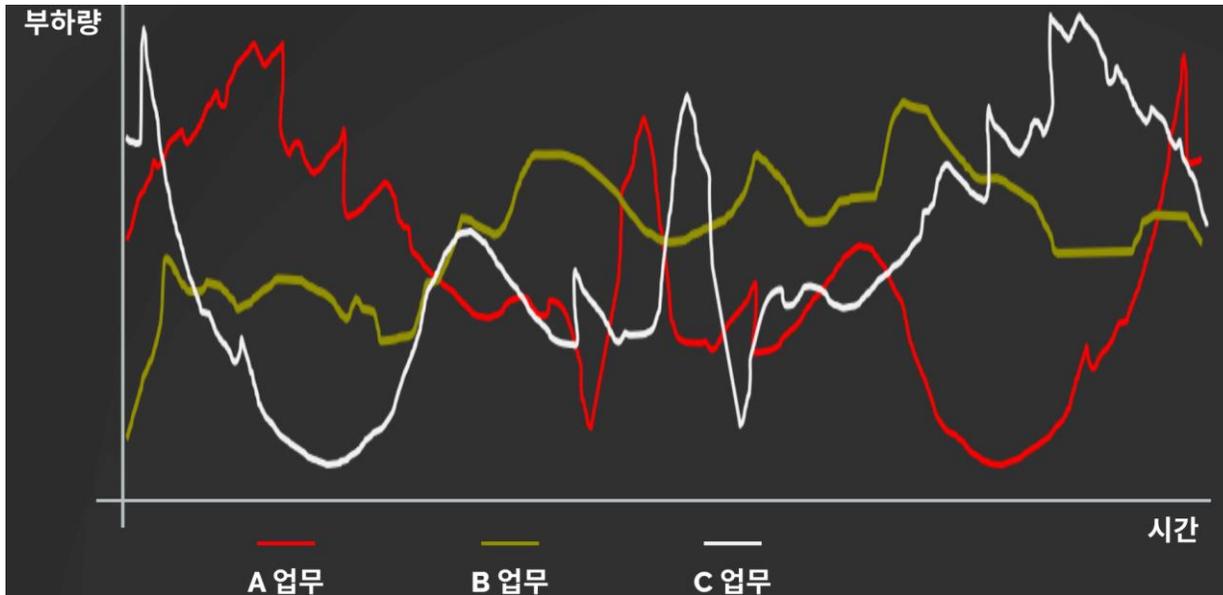
고려사항

- ⊙ Peak 시간 대가 다른 여러 애플리케이션을 운영 시 기존의 시스템 구성과 사이징은 적합하지 않음
- ⊙ 관리자가 매일 시스템사항을 모니터링하고 그에 맞게 수작업으로 자원을 조정하기 어려움
- ⊙ 업무 별로 Peak 시 사용량 기준으로 사이징 하여 구축 비용에 대한 상당한 부담



방안

- 관리자의 개입없이도 시스템 부하량에 맞추어 자원일
- 클라우드 서비스 환경에서도 서비스 품질에 중요한 요소인 응답시간과 TPS 를 보장
- 자동확장에 의한 자원 효율성 확보를 통한 시스템 구축 비용 20% 이상 절감



# T-MI : 인스턴스 개수 한정된 환경(기존 나라통계)



기존에는 가상화 환경으로 부하 상황에서 리소스를 효율적으로 활용할 수 없음

Request Viewer

Request Velocity

Label	Throughput	Response Time	Users	Status
gro..4hjkz	0.5tps	0.00s	1u	Satisfying
gro..89j7t	0.0tps	0.00s	1u	Satisfying
gro..cqd92	0.5tps	0.01s	1u	Satisfying
gro..lbh58	0.0tps	0.00s	1u	Satisfying
gro..xrsxh	0.0tps	0.00s	1u	Satisfying
por..2xfkl	11.5tps	1.05s	158u	Satisfying
por..5ghkq	10.7tps	1.08s	161u	Satisfying
por..f4nm9	12.5tps	1.07s	156u	Satisfying
por..mlxwj	0.0tps	0.00s	158u	Satisfying
por..qr4t2	10.5tps	1.06s	140u	Satisfying

APDEX: 100.0 % (Active Users: 721.2), TPS: 50.0 (Error Rate: 0.0 %), Average Response Time: 756ms

Transaction Heatmap(T-Map)

autoscaling.jmx (C:\Users\qhdhm\Downloads\apache-jmeter-5.2.1\bin\autoscaling.jmx) - Apache JMeter (5.2.1)

jp@gc - Composite Graph

jp@gc - Active Threads Over Time > Portal - Concurrency Thread Group-ThreadStarter (x10)

jp@gc - Response Times Over Time > Overall Response Times

# T-02 : 전국사업체조사 Peak일 때 자동부하분산 환경 부하테스트



## 부하에 따른 컨테이너 자동확장으로 응답시간 보장과 TPS 증가

0. auto-scaling | groupw-10-4hjkz | groupw-10-89j7t | groupw-10-cqd92 | groupw-10-lbh58 | groupw-10-xrsxh | portal-19-2xfkl | portal-19-5ghkq | portal-19-f4nm9 | portal-19-mlxwj | portal-19-qr4t2

Request Viewer: 400.0 (Left), 36 (Right)

Request Velocity

Container	TPS	RT	U	Health
gro..4hjkz	0.0tps	0.00s	1u	0
gro..89j7t	0.0tps	0.00s	1u	0
gro..cqd92	0.0tps	0.00s	1u	0
gro..lbh58	0.5tps	0.01s	1u	0
gro..xrsxh	0.0tps	0.00s	1u	0
por..2xfkl	7.5tps	1.02s	66u	9
por..5ghkq	6.0tps	1.02s	81u	7
por..f4nm9	8.0tps	1.02s	95u	6
por..mlxwj	7.0tps	1.05s	87u	8
por..qr4t2	7.0tps	0.96s	60u	7

APDEX: 100.0 % (Active Users: 566), TPS: 29.3 (Error Rate: 0), Average Response Time: 6ms

Transaction Heatmap(T-Map): Response Time (s) vs Time

```

192.168.23.160:22 - root@bastion:~ - Xshell 6
ssh://root@192.168.23.160:22
root@bastion:~
root@bastion:~
root@node1:~
root@master1:~
root@cobbler:/var/named/dy...
Every 1.0s: oc get hpa ; oc get po -l deploymentconfig=portal-demo ; oc get po -l deploymentconfig=group... Mon Nov 16 16:51
NAME          REFERENCE          TARGETS  MINPODS  MAXPODS  REPLICAS  AGE
groupware-demo DeploymentConfig/groupware-demo 0%/60%   5         10        5          2m38s
portal-demo   DeploymentConfig/portal-demo     78%/60%  5         10        5          2m45s
NAME          READY   STATUS    RESTARTS  AGE
portal-demo-19-2xfkl 1/1     Running  0         38m
portal-demo-19-5ghkq 1/1     Running  0         37m
portal-demo-19-ckm7  0/1     Running  0         15s
portal-demo-19-f4nm9 1/1     Running  0         37m
portal-demo-19-mlxwj 1/1     Running  0         38m
portal-demo-19-qr4t2 1/1     Running  0         38m
portal-demo-19-vvsk2 0/1     Running  0         15s
NAME          READY   STATUS    RESTARTS  AGE
groupware-demo-10-4hjkz 1/1     Running  0         38m
groupware-demo-10-89j7t 1/1     Running  0         37m
groupware-demo-10-cqd92 1/1     Running  0         38m
groupware-demo-10-lbh58 1/1     Running  0         38m
groupware-demo-10-xrsxh 1/1     Running  0         37m
    
```

# T-03 : 초중고 사교육비조사 Peak일 때 자동부하분산 환경 부하테스트



## 부하에 따른 컨테이너 자동확장으로 응답시간 보장과 TPS 증가

홈 / WAS / Dashboards / 대시보드

0.auto-scaling groupw-10-4hjkz groupw-10-89j7t groupw-10-cqd92 groupw-10-lbh58 groupw-10-xrsxh portal-19-2xfkl portal-19-5ghkq portal-19-f4nm9 portal-19-mlxwj portal-19-qr4t2

Request Viewer

Request Velocity

Container	TPS	Latency	Uptime	Status
gro..4hjkz	9.0tps	1.07s	10u	Satisfying
gro..89j7t	7.0tps	1.08s	9u	Satisfying
gro..cqd92	8.5tps	1.04s	93u	Satisfying
gro..lbh58	8.0tps	1.20s	100u	Satisfying
gro..xrsxh	6.5tps	1.02s	97u	Satisfying
por..2xfkl	0.0tps	0.00s	1u	Satisfying
por..5ghkq	0.0tps	0.00s	1u	Satisfying
por..f4nm9	0.5tps	0.00s	1u	Satisfying
por..mlxwj	0.5tps	0.00s	1u	Satisfying
por..qr4t2	0.0tps	0.00s	1u	Satisfying

APDEX

TPS

Transaction Heatmap(T-Map)

192.168.23.160:22 - root@bastion:~ - Xshell 6

```

ssh://root@192.168.23.160:22
root@bastion:~
root@bastion:~
root@node1:~
root@master1:~
root@cobbler:/var/named/dy...
Every 1.0s: oc get hpa ; oc get po -l deploymentconfig=portal-demo ; oc get po -l deploymentconfig=group... Mon Nov 16 17:07:03 2015
NAME REFERENCE TARGETS MINPODS MAXPODS REPLICAS AGE
groupware-demo DeploymentConfig/groupware-demo 93%/60% 5 10 5 18m
portal-demo DeploymentConfig/portal-demo 0%/60% 5 10 5 3m33s
NAME READY STATUS RESTARTS AGE
portal-demo-19-2xfkl 1/1 Running 0 54m
portal-demo-19-5ghkq 1/1 Running 0 52m
portal-demo-19-f4nm9 1/1 Running 0 53m
portal-demo-19-mlxwj 1/1 Running 0 54m
portal-demo-19-qr4t2 1/1 Running 0 54m
NAME READY STATUS RESTARTS AGE
groupware-demo-10-4hjkz 1/1 Running 0 54m
groupware-demo-10-89j7t 1/1 Running 0 52m
groupware-demo-10-cqd92 1/1 Running 0 54m
groupware-demo-10-jz9bh 0/1 Running 0 10s
groupware-demo-10-jzj6z 0/1 Running 0 10s
    
```

[정보-INFO] WAS가 시작되어 에이전트가 연결되었습니다.  
 발생에이전트 : groupw-10-zh4w9@groupware-demo-10-zh4w9[10.131.0.206]  
 클릭하여 상세한 정보를 확인하세요.... 5s.

[정보-INFO] WAS가 시작되어 에이전트가 연결되었습니다.  
 발생에이전트 : groupw-10-jz9bh@groupware-demo-10-jz9bh[10.131.0.205]  
 클릭하여 상세한 정보를 확인하세요.... 5s.

[정보-INFO] WAS가 시작되어 에이전트가 연결되었습니다.  
 발생에이전트 : groupw-10-jzj6z@groupware-demo-10-jzj6z[10.131.0.207]  
 클릭하여 상세한 정보를 확인하세요.... 5s.

# DEMO - 부하 테스트 결과 비교

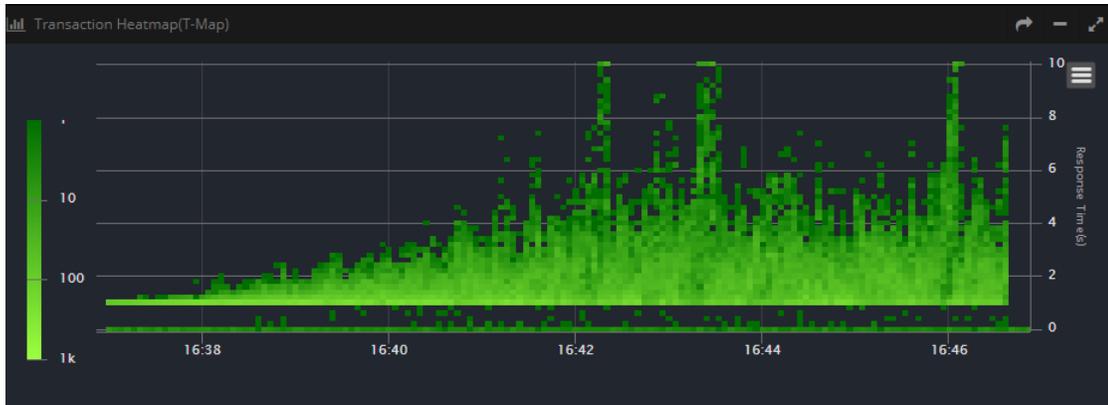


## ➡ Peak 시점이 다른 통계조사에 대해 별도 개입없이 자동자원할당이 가능함을 확인

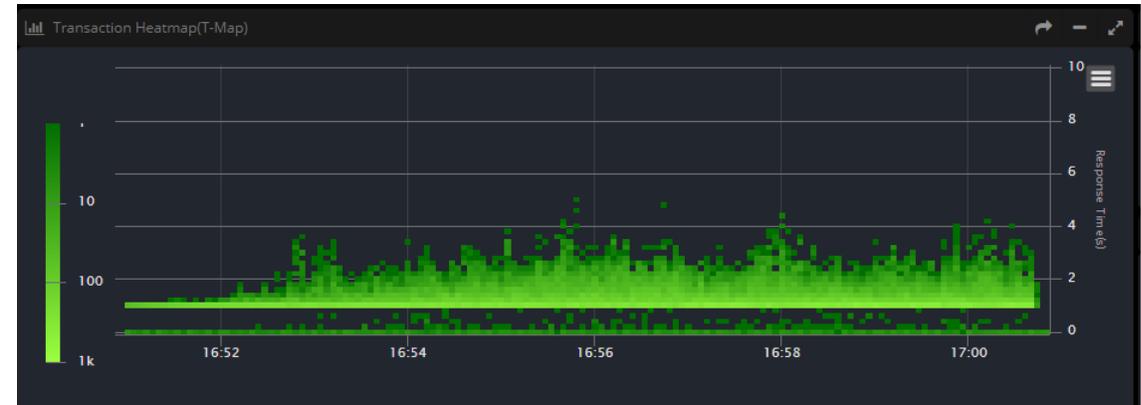
- 기존환경과 비교하여 자동자원할당이 되는 환경이 **1.7 배** 많은 양을 처리하며, **평균응답시간이 2.5 배** 빠름

테스트 케이스	테스트 내용	시뮬레이션 테스트 환경	처리량	처리량 비교	평균응답시간	응답시간비교	최소응답시간	최대응답시간	TPS
T-01	자동자원할당이 안 되는 환경에서 부하테스트	기존 나라통계 환경	53,582	100%	2,374	100%	1,010	19,979	88
T-02	Peak 시점이 다른 자동 자원할당이 되는 환경에서 부하테스트	전국사업체 조사 Peak 일 때 환경	93,869	169%	919	258%	10	4,589	155
T-03	Peak 시점이 다른 자동 자원할당이 되는 환경에서 부하테스트	초중고 사교육비 조사 Peak일 때 환경	91,394	164%	971	244%	10	5,134	151

T-01 자동자원할당 안되는 기존 환경 응답시간분포



T-02/03 자동자원할당이 되는 환경 응답시간분포



# 행정안전부 - 국가 80 여개 중앙부처 그룹웨어(온-나라) 시스템



## 한국지역정보개발원 온-나라 클라우드 문서 2.0 사업(2018년) - Red Hat PaaS & OPENMARU APM



프로젝트 명	온-나라 클라우드 문서 2.0
얼마나 중요한 업무인가?	정부부처 20군데와 지방자치단체 약 80군데 등 100만 공무원이 사용하는 문서 결재 시스템
규모는 어떻게 되는가?	Worker Node 80대 규모
오픈마루 역할은?	PaaS 전문인력 상주와 OPENMARU APM으로 성능관리를 담당하여 안정화 지원

OPENMARU APM은 국내 최초, 최다 컨테이너 & 클라우드 환경을 지원

### 법정부 기반 업무관리시스템으로 효율적인 행정 구현

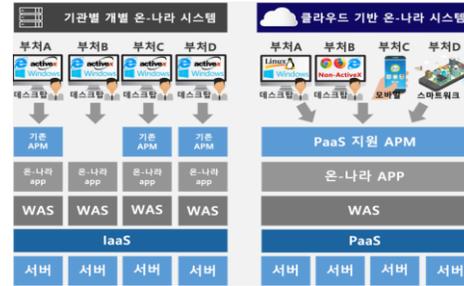
- 행안부, 26개 기관 대상으로 법정부 기반 온-나라시스템 고도화 착수 -

- 행정안전부(장관 김부겸)는 17일 정부세종컨벤션센터(세종시 소재)에서 중앙부처 및 지자체 온-나라시스템\* 담당자 및 관계자 등 70여명이 참석한 가운데 '클라우드(인터넷 기반 정보 통신 자원 통합·공유 서비스) 기반 온-나라시스템 고도화 사업' 착수보고회를 개최하였다.
  - \* 온-나라 시스템 : 행정기관의 업무에 대한 문서 작성·전도·결재·등록·공유·공개 등 문서처리의 모든 과정을 기록·관리하는 전자결재시스템
- 이번 고도화 사업은 국무조정실, 금융위원회 등 26개 기관을 대상으로 각 기관별로 보고서 및 문서를 저장·보관하는 기존방식에서 통합저장소(클라우드)에서 공동기간·결재가 가능하도록 하는 사업으로 2019년까지 전 중앙부처에 확산할 계획이다.

행정안전부 보도 자료  
법정부 기반 온-나라 시스템 고도화 착수



클라우드 전환 시 노후 장비 교체 대비 2.6배의 비용 효과 발행



클라우드 온-나라 문서 2.0 개념도



온나라 클라우드 APM 성능 테스트

# 롯데카드 - LP (Life Platform)



↳ 롯데카드의 계정/정보/채널계를 모두 Digitalization 인프라 클라우드 플랫폼을 구현했습니다.

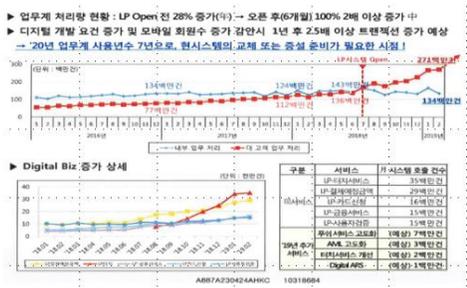


프로젝트 명	롯데카드 채널계 클라우드 구축
얼마나 중요한 업무인가?	금융권 최초로 채널계 시스템 도입으로 대고객 접점 서비스
규모는 어떻게 되는가?	채널계 전체 클라우드 도입
오픈마루 역할은?	자동 확장, 축소되는 클라우드 컨테이너를 국내 최초로 OPENMARU APM으로 안정화 지원 및 성능, 튜닝

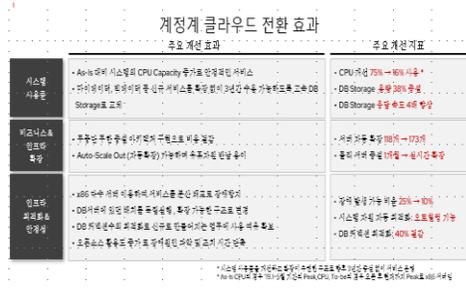
## ✔ 롯데카드의 클라우드 환경 도입 효과



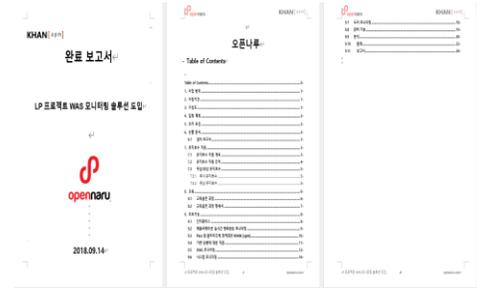
롯데카드, '레드햇 포럼 서울 2018' 최고 디지털 전환 상 수상



채널계 클라우드 도입 효과



계정계 클라우드 전환 효과

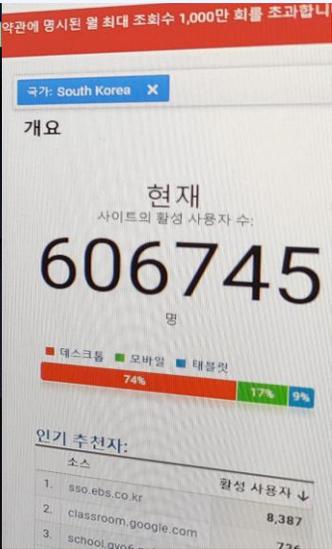


롯데카드 LP 프로젝트 OPENMARU APM 도입 완료 보고서

# 한국교육방송공사 - 코로나 대응을 위한 온라인 클래스 시스템



➡ EBS 온라인 클래스에 도입되어 동시접속자 130만명, VM 600 대, WAS 인스턴스 1200 여 개 모니터링



프로젝트 명	EBS 온라인 클래스
얼마나 중요한 업무인가?	코로나로 인하여 초,중,고 300백만명의 학생이 접속하는 온라인 클래스 구축 업무
규모는 어떻게 되는가?	WEB/WAS 600대 / 인스턴스 1,200개 규모
오픈마루 역할은?	1주일 내 600대 머신에 1,200개 인스턴스 구축 및 APM으로 오픈지원, 성능 튜닝 및 안정화 지원

✔ OPENMARU APM으로 안정화를 실현한 실제 EBS 방송 화면



교육 차관에게 OPENMARU APM으로 보고하는 화면



600여대의 서버의 총 1,228개 WAS 모니터링



실제 인스턴스 1,200개 그룹대시보드 모니터링 화면-2



실시간 Active User를 확인할 수 있는 모니터링 화면



openmaru