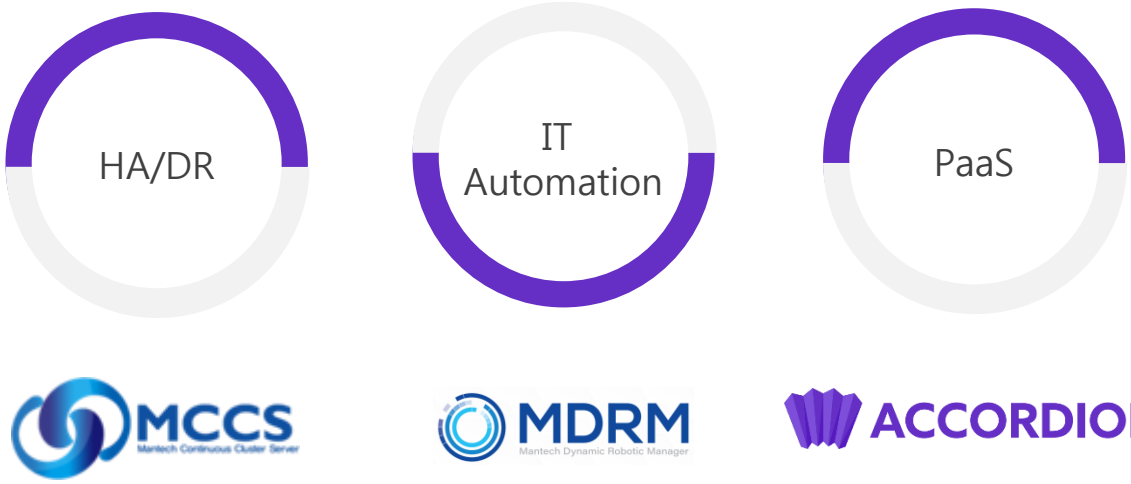
컨테이너는 왜 클라우드의 중심으로 떠올랐는가?

맨텍에 대하여...

맨텍은 무한 확장 가능하고 중단없는 IT인프라를 위한 솔루션을 개발하고 서비스를 제공하고 있는 전문 소프트웨어 기업입니다. 맨텍의 더 나은 솔루션을 통한 불만제로 서비스를 위한 노력은 이미 국내외 3,000여 레퍼런스를 통해서 확인이 가능합니다.

- ✓ 설립일: 1989
- ✓ 본사: 서울 성수동
- ✓ 임직원수: 98
- ✓ Phone : +82-2130-6900
- ✓ URL : www.mantech.co.kr

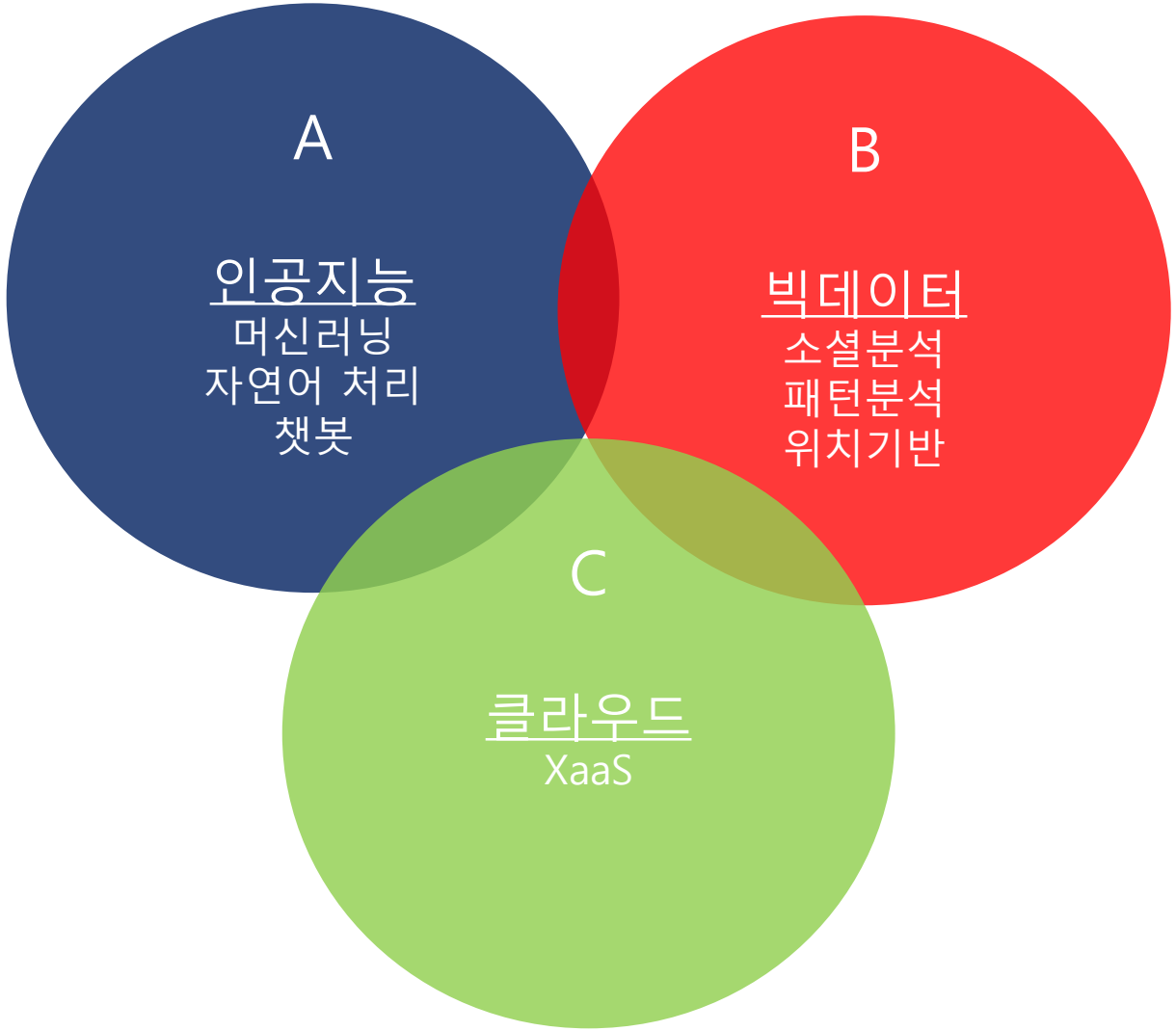


I

Why Container

ABC의 공통적인 트렌드

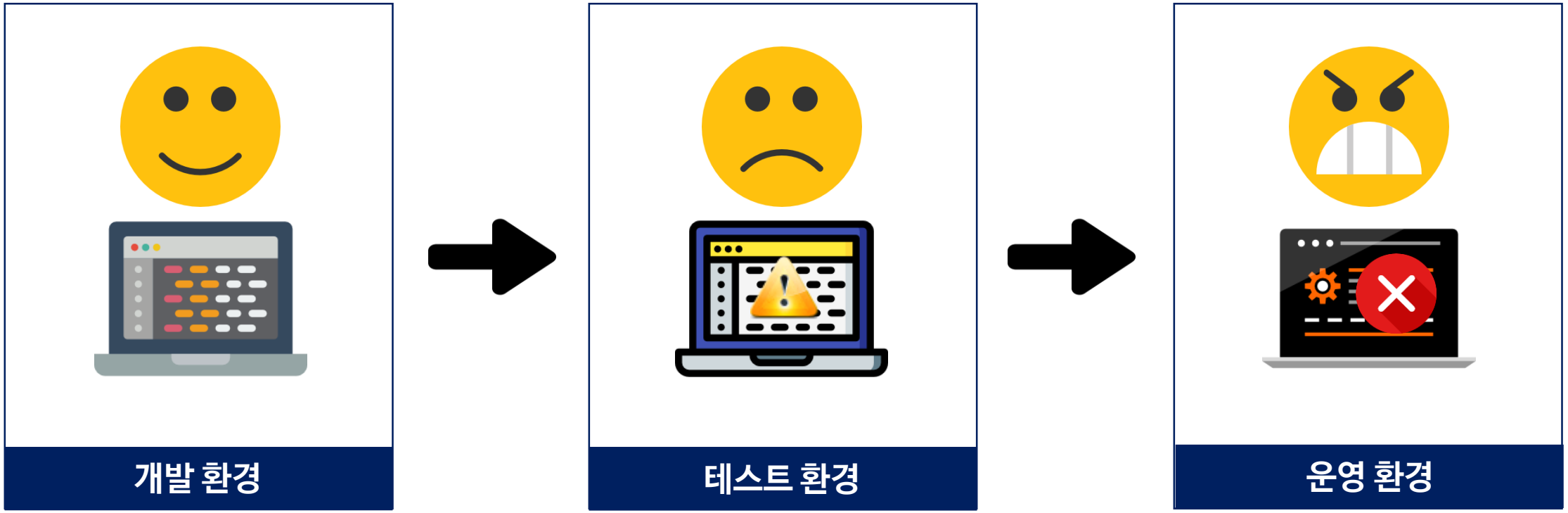
- 소프트웨어 정의
- 오픈소스
- 컨테이너 기반



늘 지속되어온 장벽

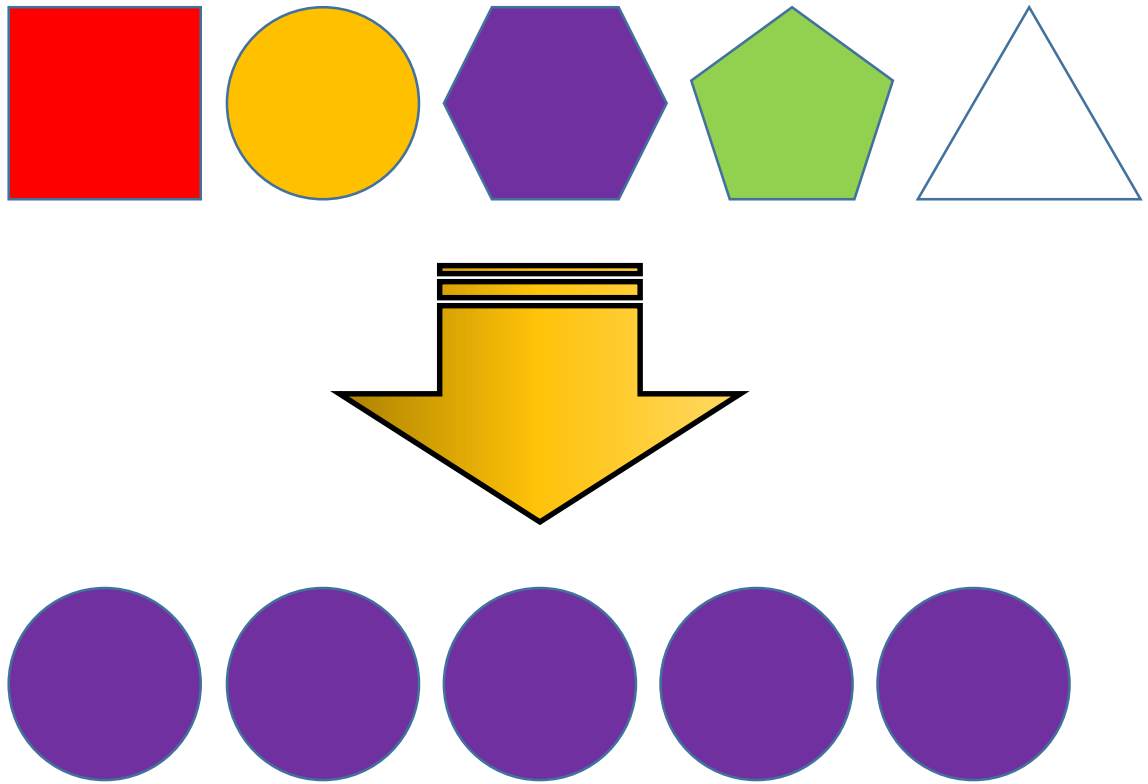
개발자의 고민 - 서로 다른 컴퓨팅 환경으로 인한 오류

'소프트웨어를 한 컴퓨팅 환경에서 다른 컴퓨팅 환경으로 이동하면서도 안정적으로 실행하는 방법이 없을까?'



운영자의 고민 - 여러 SW를 단일의 방식으로 관리할 방법은?

- 단일 형상으로의 표준화 필요
- VM은 하이퍼바이저간, 물리, 가상, 클라우드간 호환성의 해결을 하지 못함
- HW와 운영체제의 종속성을 벗어날 방안이 필요



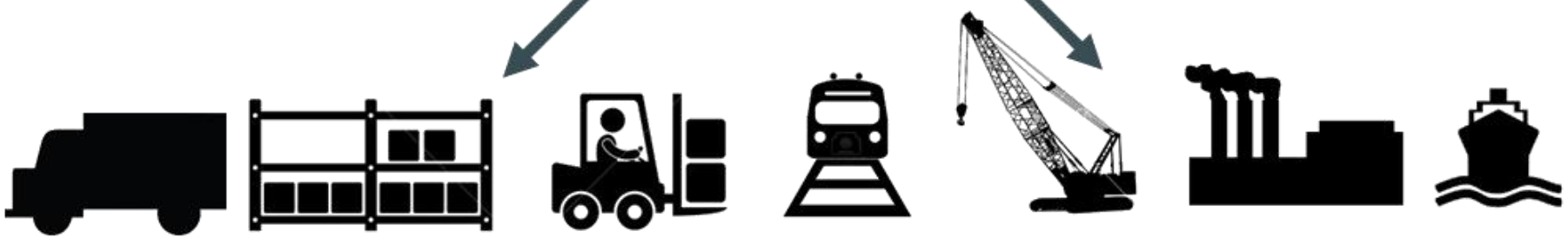
물류 수송 표준 규격화의 혁명

다양한 형태의 상품들

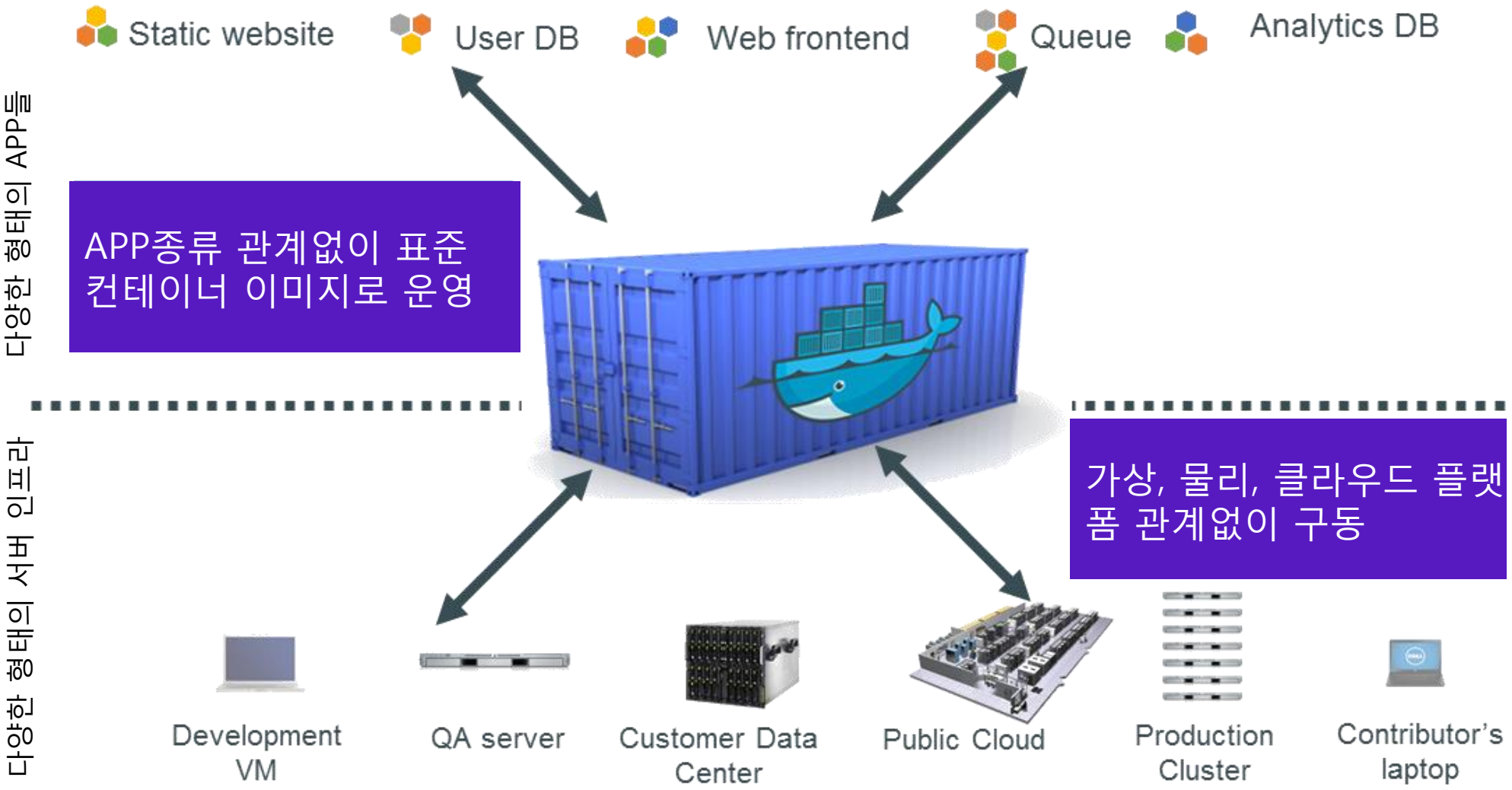
대상 품목에 관계없이
표준 규격화된 화물 관리

운송수단 관계없이 동일한
적재와 하역 방식

다양한 형태의 운송수단들

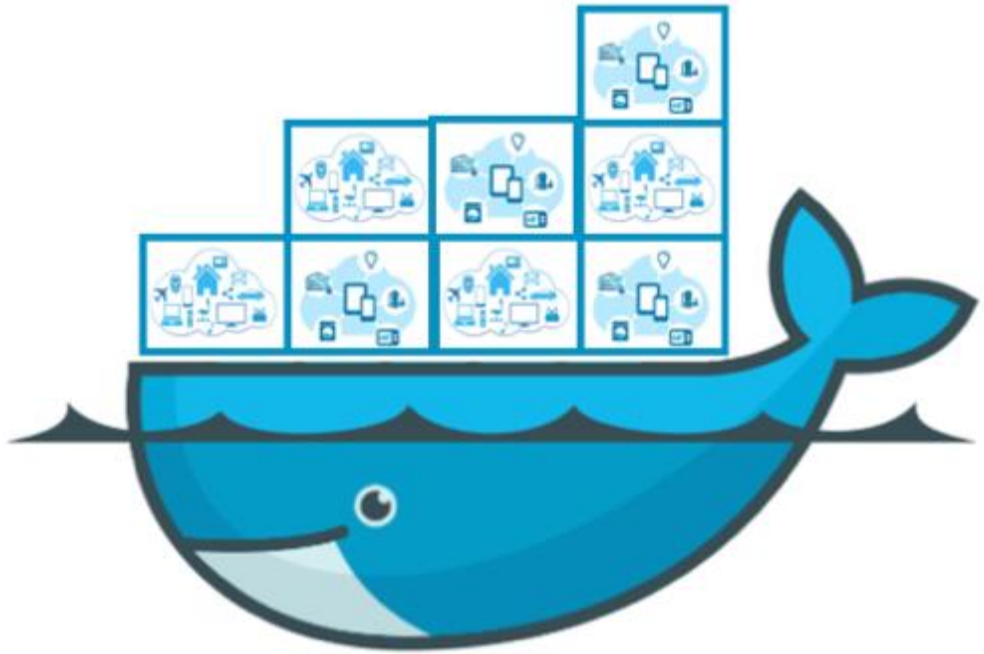


도커 컨테이너를 통한 애플리케이션의 단일 규격화



컨테이너의 장점

- 이식성 甲
- 싸이즈 甲
- 속도 甲
- 호환성 甲

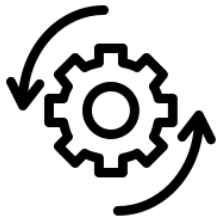


컨테이너 사용이 갈수록 늘어가는 이유

환경 일관성

- 개발과 운영 환경의 격차로 인한 오류 최소화
- 지속적인 최적화된 애플리케이션 실행 환경 제공

효율적인 운영



- 리소스 효율성 향상
- 빠른 부팅 시간
- 애플리케이션의 신속한 확장 및 축소

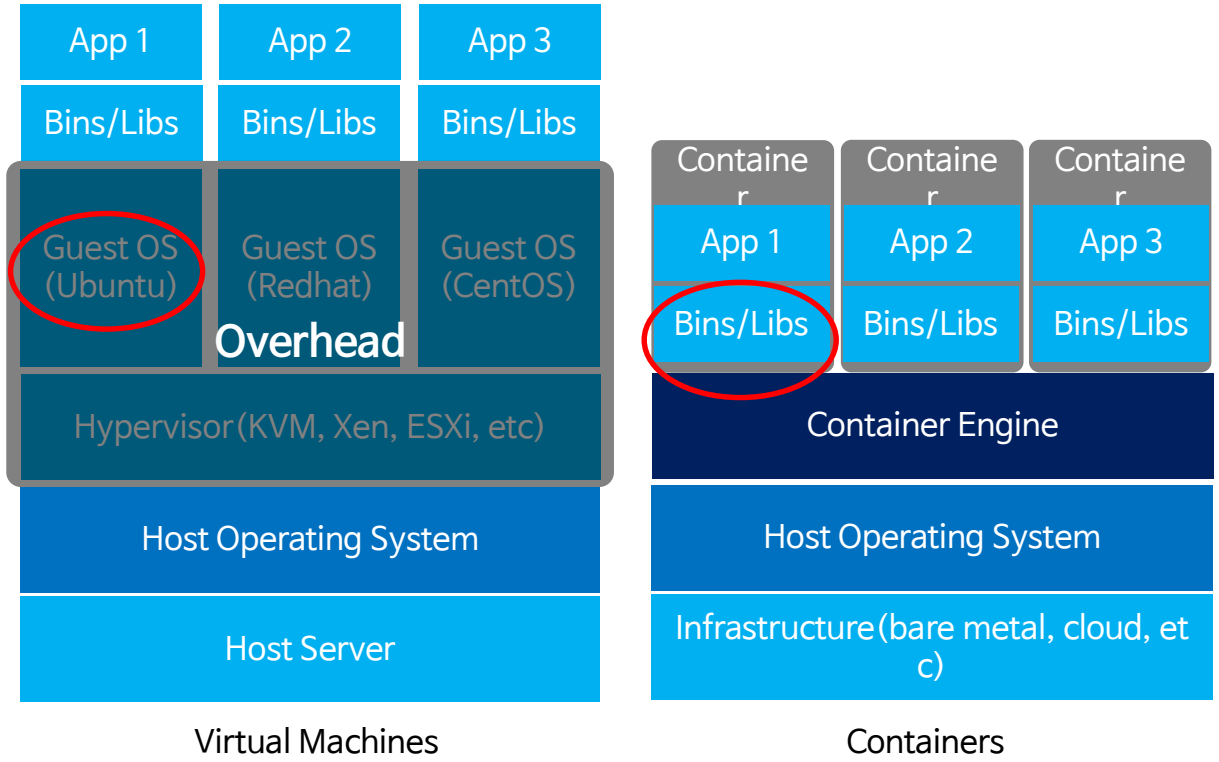
개발자 생산성

- 여러 OS 버전과 환경변수에 대한 추가 개발 불필요
- QA기간의 대폭 단축으로 인한 배포 일정 감소

하이브리드

- 이기종 물리, 가상, 클라우드간 마이그레이션 불필요
- 이기종 인프라간 배포 및 확장이 가능하여 하이브리드 클라우드 전략에 최적

가상 서버와 비슷하면서도 완전 다름



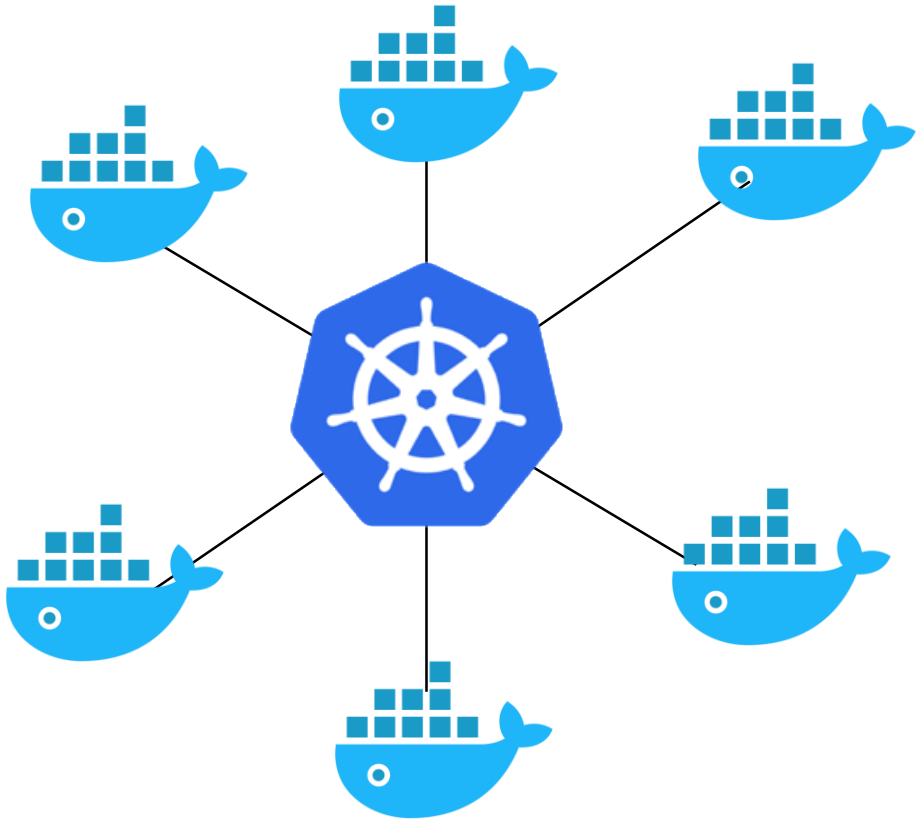
- 앱 간 간섭 받지 않고 독립적인 환경으로 격리되어 운영
- 컨테이너는 OS가 없음
- VM은 Host OS커널에 비종속적인데 반해 컨테이너는 Host OS 커널에 종속적

VM vs Container

	가상 서버	컨테이너
가상화	서버 가상화 VM 간 격리	호스트 자원 공유 네임 스페이스를 통해 앱 간 격리
이미지화	OS와 가상 디바이스 등도 포함되어 사이즈가 큼	필요한 앱과 bin/lib만 포함되어 **사이즈가 매우 작음**
플랫폼간 이식성	동일 하이퍼바이저 간 only	**이기종 물리, 가상, 클라우드 간**
이기종 OS 사용	가능	불가능 (Host OS와 동일한 OS 계열이어야 됨)
부팅시간	수십초~수분	수초~수십초
개발 환경 구축	수작업으로 앱 설치 및 삭제 필요 보통 하루 이상 소요	환경 구축의 수작업 요소 제거 **수 분~수 십분 소요**
직접도	물리 서버당 4~6 VM	물리 서버당 20~30 컨테이너
앱 확장성	Low (수작업 요소 많음)	High (완전 자동화 가능)
OS Cost	VM 당	Host 당
베어메탈 대비 성능	50~80%	**98%**
자원 할당	CPU, Memory, GPU, 네트워크, Storage 등	CPU, Memory, GPU, 네트워크, Storage 등

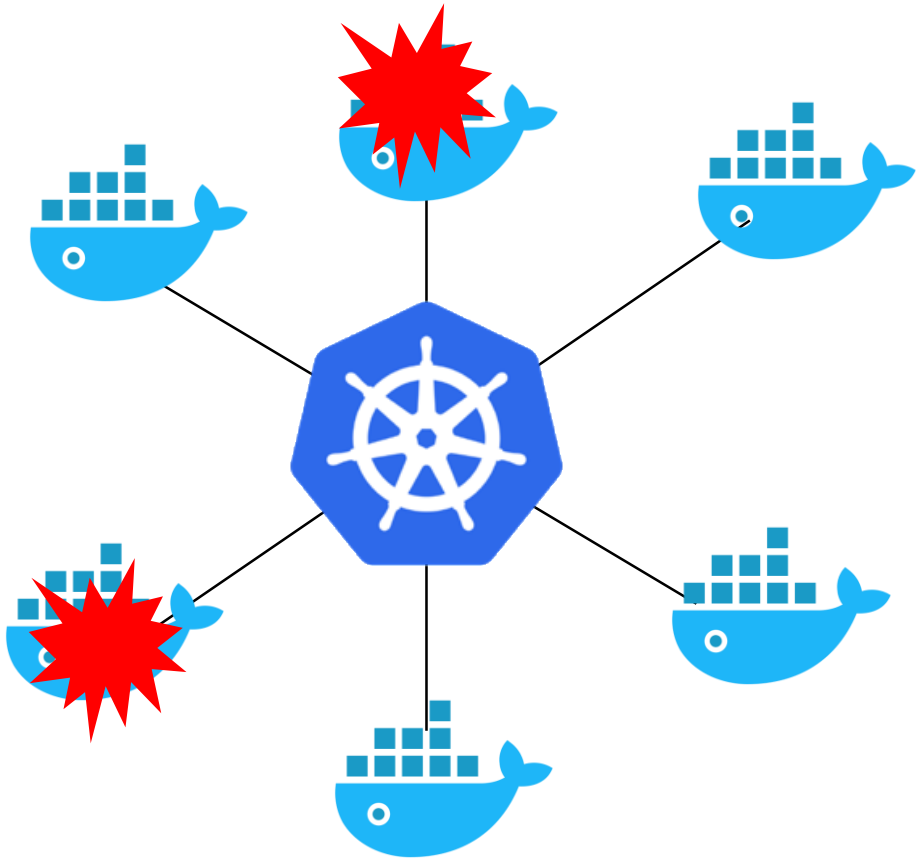
컨테이너의 특징 1 – 분산형

- 분산된 노드에 다수의 컨테이너가 **집합체**를 이루어 운영
- 이기종 플랫폼 및 클라우드 간 **분산 클러스터링** 가능
- 하이브리드 및 멀티 클라우드 환경 구축에 최적



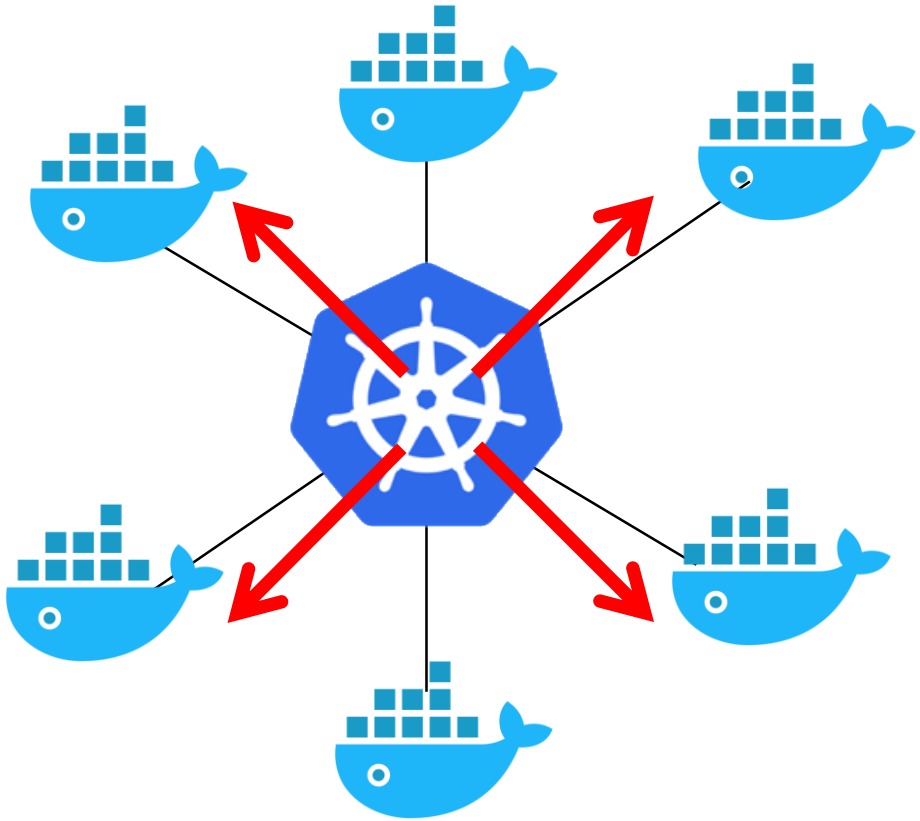
컨테이너의 특징 2 – 무중단

- 특정 노드 장애 시 다른 노드의 컨테이너에서 연속적인 서비스 제공
- 컨테이너 장애 시 **자동 복구**
- DC/OS를 통한 롤링업그레이드 연동 시 **무중단 애플리케이션 배포**



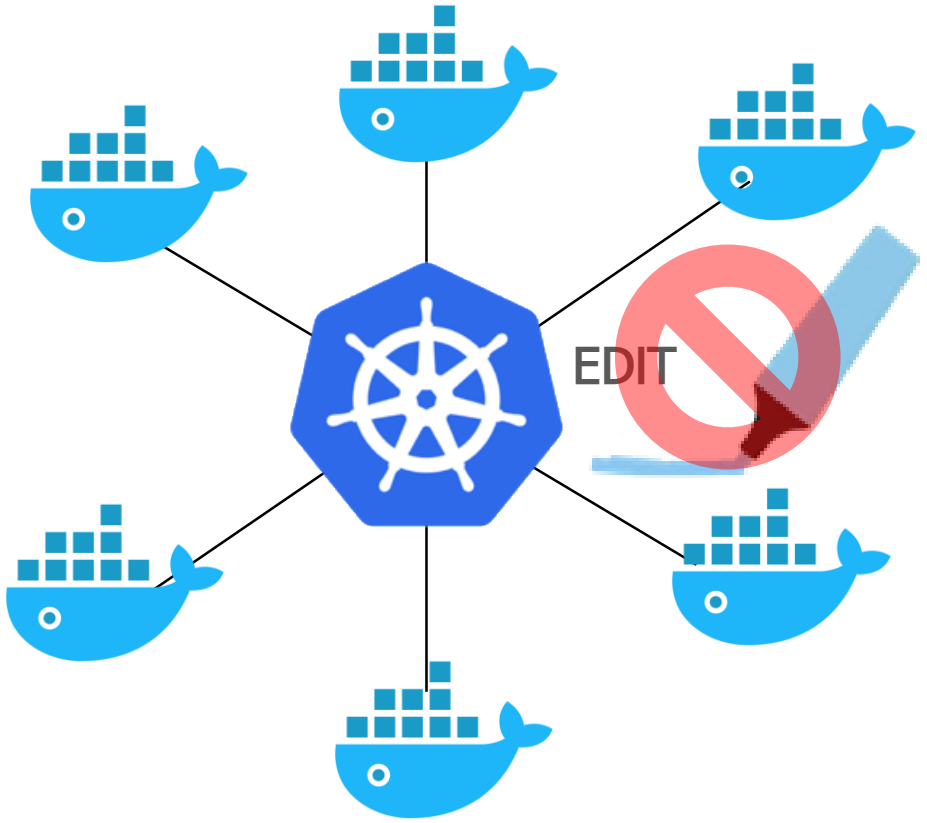
컨테이너의 특징 3 – 확장성

- 성능 요구 시 플랫폼의 종류와 노드 간 거리 상관없이 확장 가능
- 쿠버네티스와 같은 DC/OS를 통해 **자동 확장/축소** 구현
- 확장을 위한 앱 및 네트워크 등의 **수정 필요 없음**



컨테이너의 특징 4 – 불변성

- 이미 생성된 컨테이너 내부의 애플리케이션의 수정 불가
- 수정이 필요한 경우 새로운 컨테이너를 생성하여 배포
- 호불호가 갈림 (투명한 버전 관리 vs 관리 어려움) → CI/CD를 통한 자동화된 버전관리로 단점 극복

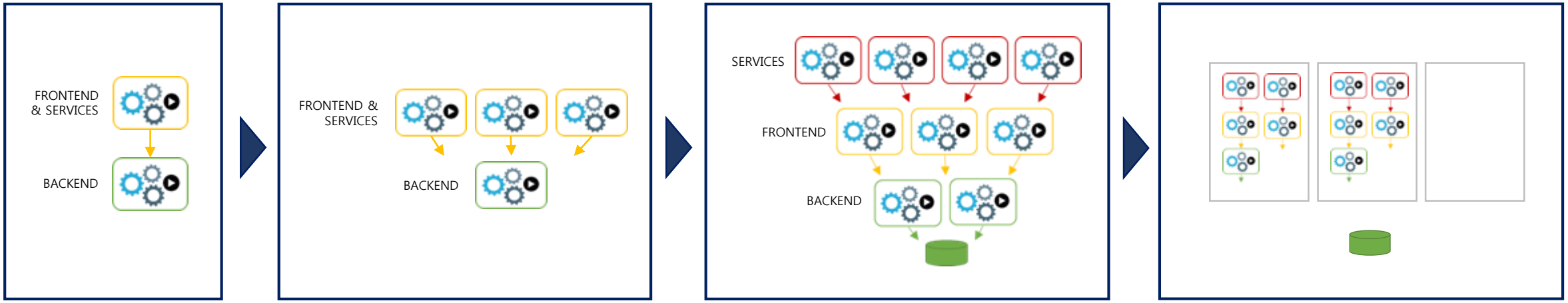


Ⅱ

컨테이너 통합 관리의 필요성

컨테이너 오케스트레이션(DC/OS)은 왜 필요한가?

애플리케이션이 처음에는 한 개의 컨테이너로부터 시작하지만 곧 많은 애플리케이션과 인스턴스로 늘어나게 되면서 여러 머신에서 컨테이너를 운영하게 됩니다. 컨테이너 오케스트레이션은 여러 개의 컨테이너를 편리하게 관리해주는 작업으로 여러 개의 컨테이너로 하나의 서비스를 구성할 수 있게 됩니다.



간단 애플리케이션 | 스케일 아웃 | 더 많은 컨테이너를 사용
더 많은 로드 처리 | 여러 대의 머신에서 분산 처리

자, 이제 컨테이너 오케스트레이션 시스템이 필요한 시점입니다.

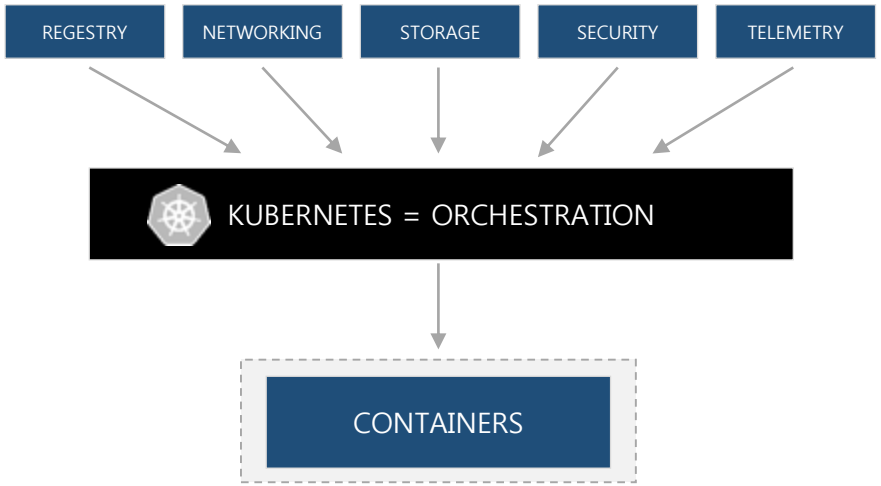
컨테이너 오케스트레이션(DC/OS)는 왜 필요한가?

Without DC/OS

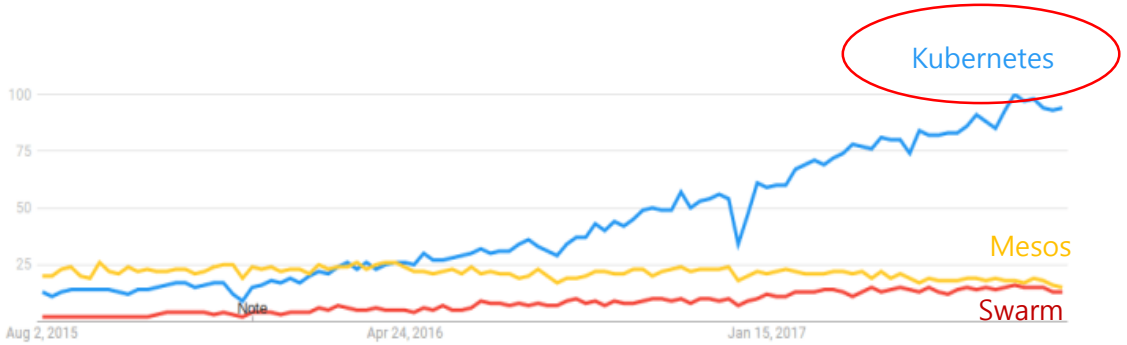
With DC/OS

오케스트레이션 천하를 통일한 쿠버네티스

여러 오픈 소스에 의해서 컨테이너 오케스트레이션 플랫폼이 혼재된 상황이었는데 최근 들어서 쿠버네티스(Kubernetes)가 사실상의 표준으로 자리잡기 시작했습니다. AWS, Azure, IBM, 구글 등은 모두 컨테이너 제공 플랫폼의 기반으로 쿠버네티스를 활용하고 있으며, on-premise 솔루션 업체들도 쿠버네티스를 지원하고 있습니다.

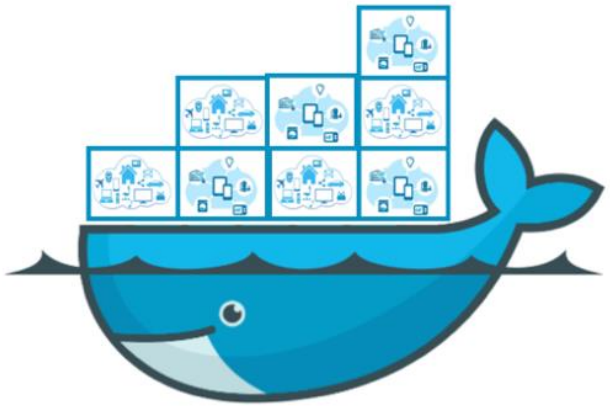


< 오케스트레이션의 최강자 '쿠버네티스' >



< Google Trends: Levels of interest in Kubernetes>

쿠버네티스와 컨테이너만 있으면 PaaS를 금방 구축할 수 있는가?



federation

Ingress

앱 카타로그

멀티 테넌시

컨테이너 저장소

영구 볼륨 구축

형상관리

CNI

컨테이너 스케줄러

Logging

모니터링

CI/CD

배포 파이프라인

Alerting

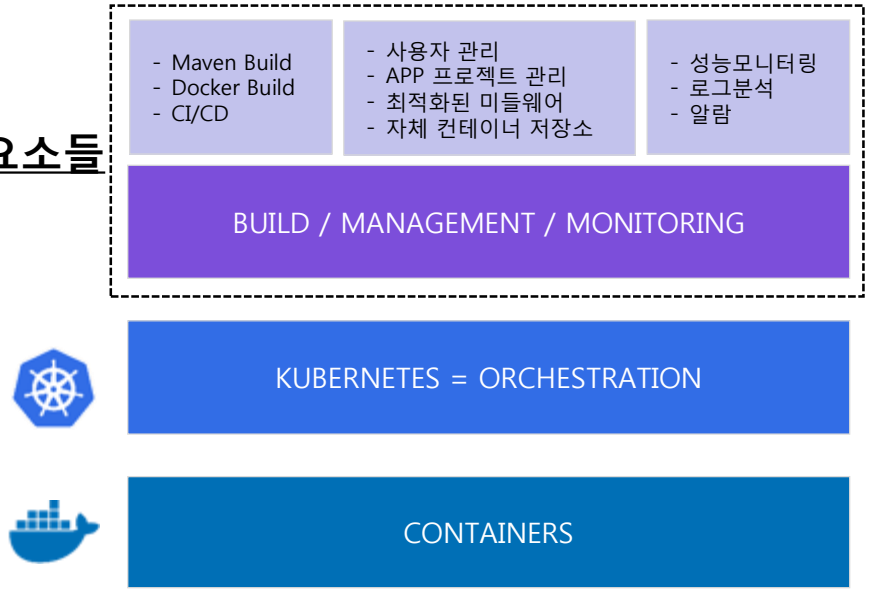
쿠버네티스를 쉽게 사용하기 위한 필요한 요소들

쿠버네티스가 이렇게 사실상의 표준으로 자리를 잡아가고 있지만, 쿠버네티스 기반 애플리케이션은 쓰기 편할지 몰라도 쿠버네티스 자체는 사용하기 쉽지 않다는 문제가 있습니다. 기업들은 사용하기 까다롭고 복잡한 쿠버네티스를 쉽게 사용하고 싶어합니다. 이를 위해 컨테이너 통합 관리 솔루션들이 하나 둘 씩 등장해 기업들이 쿠버네티스를 단순화하여 사용할 수 있도록 하는데 도움을 주고 있습니다.

쿠버네티스 클러스터를 블록체인에 집어 넣으려 오픈 스택(OpenStack) 배치를 쏟아 붓고 있는가? 걱정할 필요 없다. 다른 대부분의 사람들은 아직 쿠버네티스를 파악하려 시도하는 단계이다. 도커(Docker)이후 가장 '핫'한 기술이기는 하지만, 대부분 메인스트림 기업들에게 쿠버네티스는 여전히 '흑마술'이다. 구글의 엔지니어링 '신'들이 만든 쿠버네티스를 보통 엔지니어들이 이해하려면, 학습해야 할 '추상화'가 많기 때문이다.

- Matt Asay | InfoWorld / 2018.05.23

< 컨테이너 통합 관리 솔루션이 필요한 이유>



< 컨테이너 통합 관리 솔루션 스택>

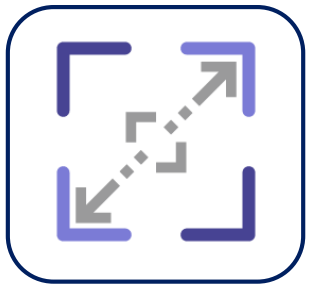
프로젝트 아코디언

- ✓ 서비스 중단 없이 요구사항에 대한 즉시 배포 및 확장 가능
- ✓ 미들웨어, 가상 인프라 및 관리 툴이 통합된
- ✓ 쿠버네티스와 도커 기반의 **컨테이너 플랫폼**
- ✓ www.accordions.co.kr 에서 다운로드 가능

무엇을 제공해 주는가?

1. 애플리케이션 배포 관리

- 번들된 Tomcat과 wildfly를 One-Click 으로 쉽고 빠르게 설치
- 여러 대의 WAS 서버를 클러스터 구성 할 때 네트워크 설정, 용량 프로비전, 부하분산을 자동으로 구성
- 번들된 WAS 이외의 다양한 애플리케이션을 컨테이너화하여 아코디언을 통해 서비스



2. 자동 확장 및 운영

- 자동 및 수동 확장을 통해 갑자기 폭증하는 사용자 요청 빠르게 처리
- 자동화된 컨테이너 복제 및 복구를 통해 중단 없는 서비스 구축

3. 모니터링

- 시스템(CPU, Memory, Disk, Network) 모니터링과 APM (Application Performance Management), 로그 검색 서비스, 알람 (Email, Slack) 서비스를 통해 사전 대응적 문제해결 및 안정적인 서비스 운영

4. 빌드 및 통합 관리

- 개발된 애플리케이션을 CI/CD를 통해 매우 쉽게 빌드/배포 자동화
- 롤링업그레이를 통해 배포 시에도 중단 없는 서비스 제공
- 이전 버전으로의 롤백 필요시 원 클릭으로 수초 내 롤백

결론적으로 극대화된 생산성 향상이 컨테이너 급부상의 이유

전통적 환경

1. 서버 하드웨어 구성
2. 스토리지 구성
3. 네트워크 구성
4. OS 설치와 계정 생성
5. 프레임워크와 미들웨어 구축
6. 성능 모니터링, HA 구축
7. 테스팅 툴 구축
8. Application 로직 개발 및 구축

9. 서비스 확장 시, 1~8항 반복

Months

가상화 / IaaS

1. 가상 머신 생성
2. 프레임워크와 미들웨어 구축
3. 성능 모니터링, HA 구축
4. 테스팅 툴 구축
5. Application 로직 개발 및 구축

6. 1~5항에 대한 템플릿 생성
7. 서비스 확장 시, 템플릿 복사
8. 각 가상 머신(VM) 별 네트워크 수정, 클러스터 도메인 구성

Days

컨테이너 / PaaS

1. 개발 자원 할당
2. Application 로직 개발 및 구축

3. 서비스 확장 시,
   자동 혹은 원 클릭 확장

Minutes

정리를 하자면...

- 물리, 가상, 클라우드가 공존하는 인프라의 구성은 시대의 흐름
- 인프라의 종류와 OS의 종속성에서 벗어날 수 있는 유일한 방법은 컨테이너뿐
- 컨테이너로의 전환 전략과 솔루션 제시는 필수

Thank You