2024.07.10

캡스톤 디자인 계획 발표

효율적인 자원관리를 위한 경량형 컨테이너 오케스트레이션 시스템(개발)

팀명 : 악동개발자
20211939 허유정
20217140 김창인

프로젝트 개요

1) 프로젝트 배경
2) 프로젝트 필요성
3) 프로젝트 목표

프로젝트 계획

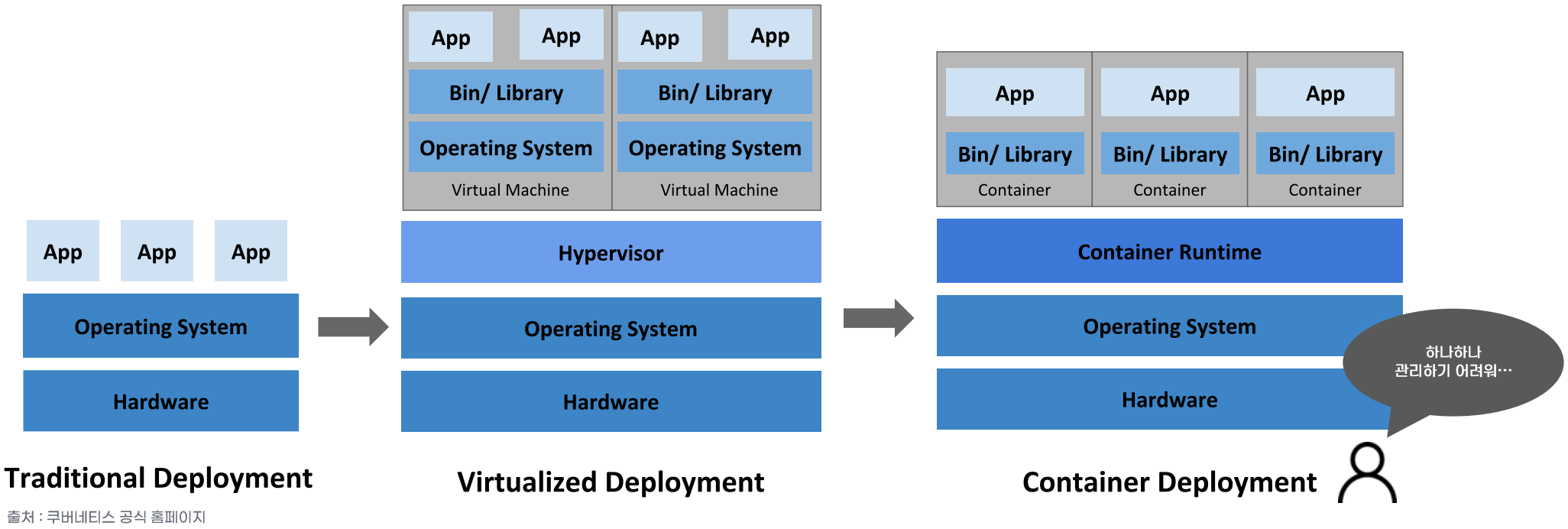
1) 진행상황
2) 요구사항 정의서
3) 시스템 아키텍처

추진전략 및 기대효과

1) 진행계획
2) 추진전략
3) 역할분담
4) 기대효과 및 활용방안

프로젝트 개요 - 프로젝트 배경

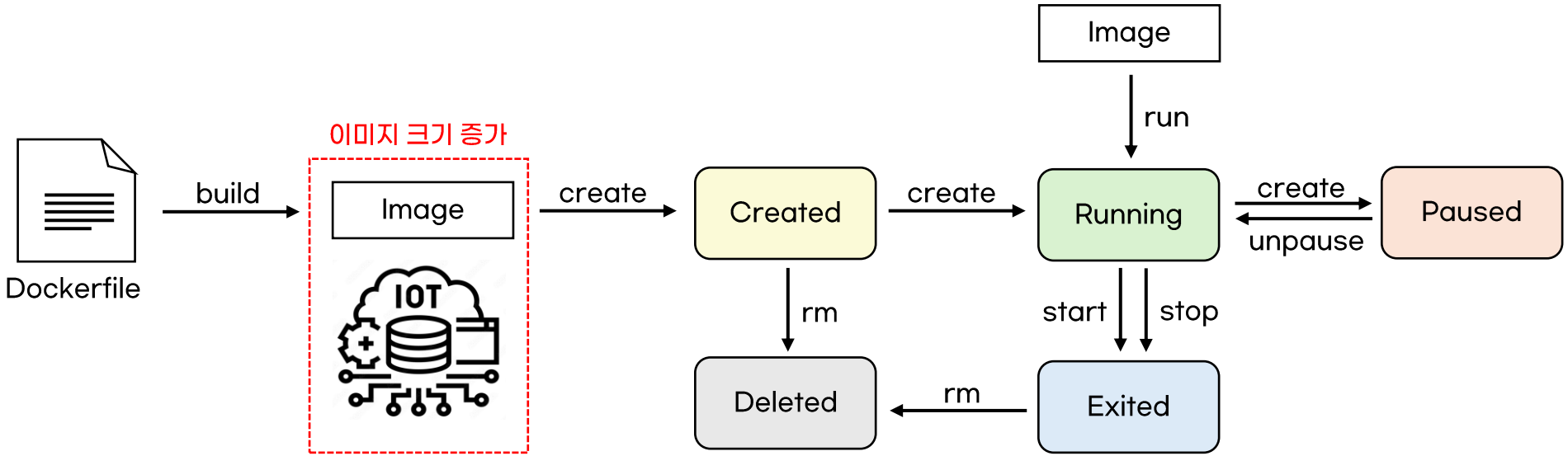
- 컨테이너 기술의 등장으로 애플리케이션의 빠른 배포와 확장, 일관된 환경 실행으로 서버 관리의 효율성 증대
- 컨테이너 관리 복잡성 증가로 효율적 운영을 위한 **컨테이너 오케스트레이션 도구 필요성** 대두



출처 : 쿠버네티스 공식 홈페이지

프로젝트 개요 - 프로젝트 필요성

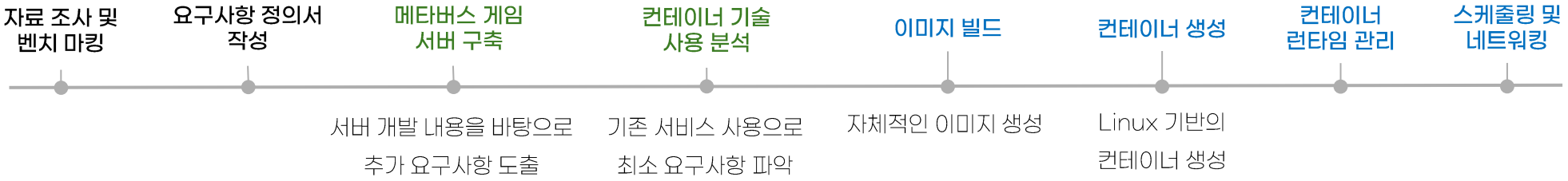
- IoT 기술의 확산에 따라 컨테이너 기술을 IoT 디바이스 융합 시켜 **효율적인 메모리**관리와 배포를 가능하도록 함
- 하지만 기존 컨테이너 기술의 경우 대규모 이미지를 저장하고 실행하는데 **제약사항**이 발생함
- 또한 기존 오케스트레이션 도구는 복잡성과 높은 학습 곡선으로 인해 **소규모 프로젝트나 조직에 부담**이 됨



< 이미지 기반의 컨테이너 생성 과정 >

프로젝트 개요 - 프로젝트 목표

- 최소 요구사항을 파악하여 소규모 프로젝트나 조직에 부담되지 않는 **경량화 된 시스템 개발**을 목표로 함
- 사용자 요구에 맞춘 특정 애플리케이션을 커스터마이징 할 수 있도록 하여 **효율적인 자원 관리**를 목표로 함
- 자원의 효율성, 자동화된 관리, 지속적인 모니터링을 위해 **DevOps 방법론**을 사용하여 더 나은 서비스 제공



경량형 컨테이너 서비스

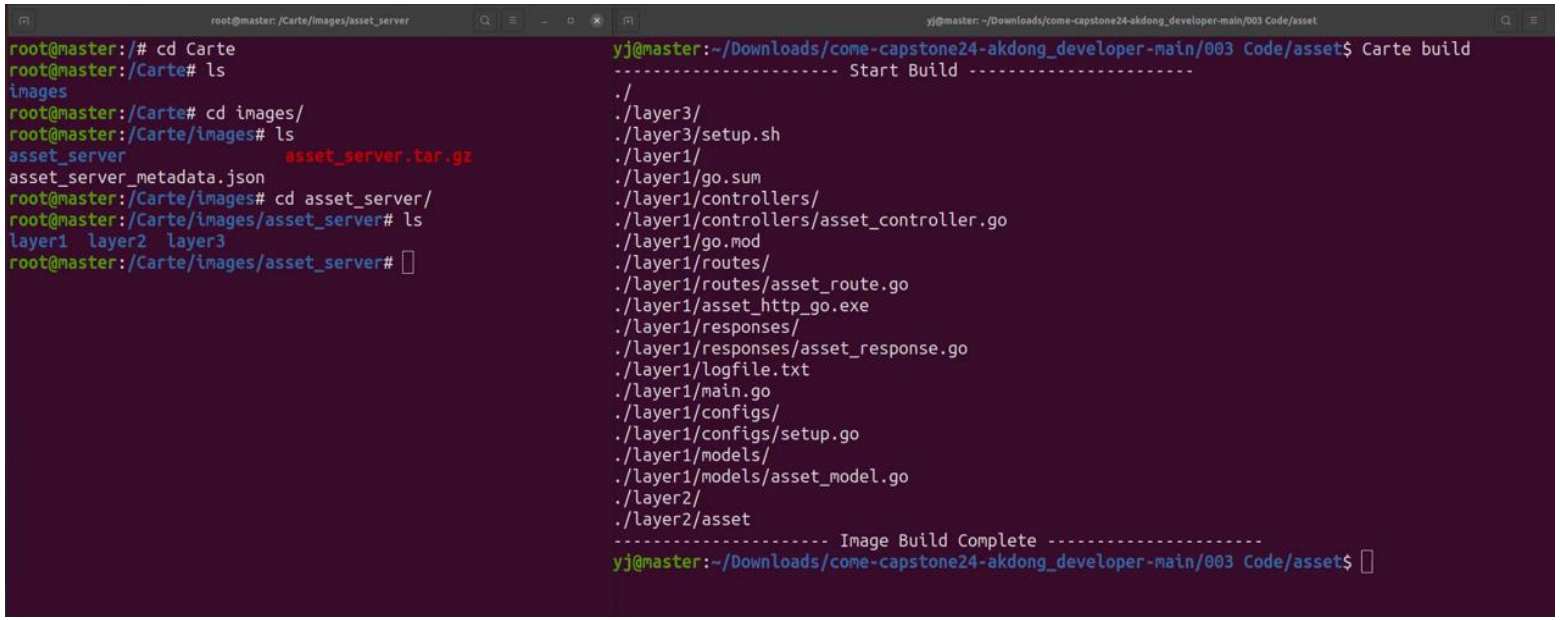
- 요구사항 분석 및 설계
- 이미지 빌드
- 컨테이너 런타임 관리
- 명령어 인터페이스

메타버스 게임 서버 개발

- 컨테이너 최소요건 파악
  (컨테이너 환경에서 일관성 유지)
- 지속적인 기능 테스트(안정성 유지)
- 지속적인 모니터링 및 로깅

part2 프로젝트 계획 - 진행상황

[캡스톤 진행 상황]



이미지 크기 비교 사진

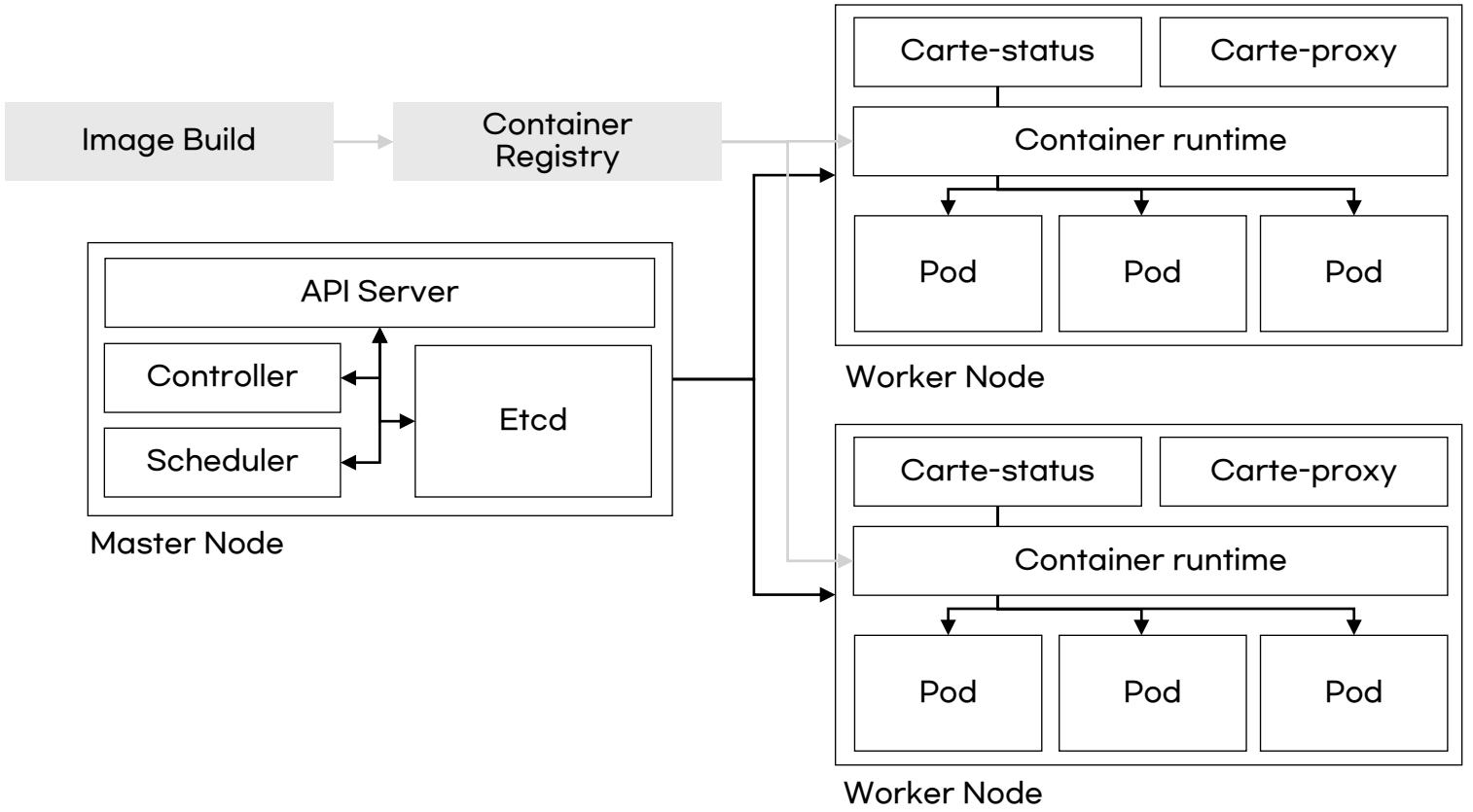
- 메타버스 게임 서버를 기존 컨테이너에 접목시켜 서비스 개발에 필요한 최소 요건 파악
- 이미지 생성을 위한 명령어 단순화와 불필요한 파일 제거로 이미지 크기 축소화 진행
- 여러 레이어로 구성된 이미지 생성을 통해 수정되는 요구사항에 대한 신속한 대응이 가능하도록 함

프로젝트 계획 - 요구사항 정의서

[요구사항 정의서]

요구사항 명	요구사항 설명	요구사항 ID
경량화 된 이미지 빌드	애플리케이션을 컨테이너 이미지로 빌드하는 기능	REQ-001
컨테이너 런타임 엔진	실행 중인 컨테이너에 대한 기본적인 관리 기능	REQ-002
스케일링	확장성을 위한 오케스트레이션 기능으로 컨테이너 인스턴스 조정 자동화	REQ-003
네트워킹	컨테이너 간 통신 및 외부 네트워크와의 연결을 관리할 수 있는 기능 제공	REQ-004
CLI	사용자가 명령줄에서 컨테이너 관리 기능을 사용할 수 있는 CLI 제공	REQ-005
테스트	메타버스 게임 서버를 기반으로 한 지속적인 테스트 진행	REQ-006

프로젝트 계획 - 시스템 아키텍처



- Master Node는 전반적인 노드의 상태를 제어하고 수행하며 이벤트를 감지함
- Worker Node는 Carte-status를 통해 마스터 노드의 명령을 받아 컨테이너화 된 애플리케이션을 동작하고 유지시킴

part3 추진전략 및 기대효과 - 진행 계획

구분	1월	2월	3월	4월	5월	6월	7월	8월	9월	10월	11월
계획 및 서비스 벤치마킹											
컨테이너 기술 관련 학습 및 언어 학습											
메타버스 게임 서버 구축											
추가 요구사항 추가											
컨테이너 이미지 빌드											
컨테이너 런타임 관리											
스케줄링 관리											
네트워킹 관리											
CLI											
최종 테스트											

part3 추진전략 및 기대효과 - 추진 전략

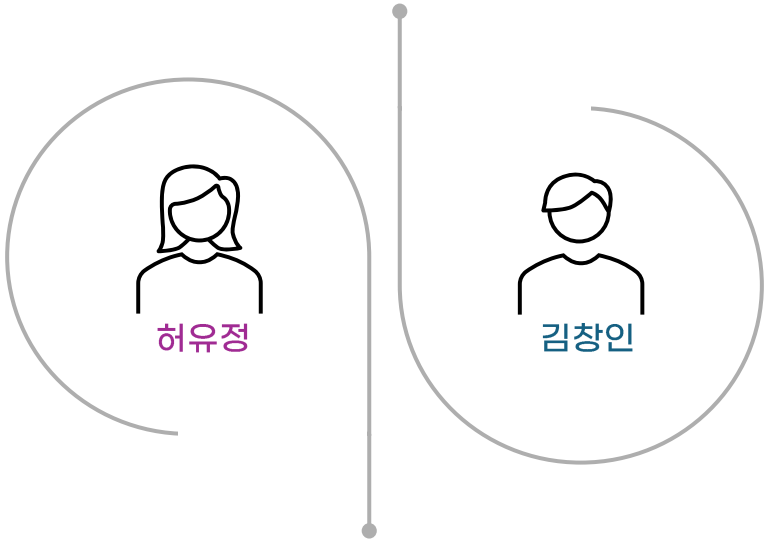
- 주기적인 스프린트 계획과 리뷰를 통한 진행 상황 모니터링
- 소프트웨어 아키텍처 설계를 통한 시스템 이해도 확립
- 애자일 개발 방법론 채택으로 프로젝트 유연성 확립

- 소스 코드와 변경 사항의 효과적인 추적 및 관리 가능
- 풀 리퀘스트 생성으로 변경 사항 검토 및 통합
- 개발자들 간 소스 코드 공유 및 병합 간편화

- 계획 공유를 통한 일정 관리
- 문서 공유의 불편함 최소화
- 멘토와의 의사소통 강화

part3 추진전략 및 기대효과 - 역할 분담

- 이미지 레이어 계층 분류
- 이미지 생성 및 빌드
- 기능에 맞는 CLI 구현
- 컨테이너 스케줄링 관리



- 이미지 레이어 계층 분류
- 컨테이너 생성 및 관리
- 설치 스크립트 정의
- 컨테이너 네트워킹 관리

part3 추진전략 및 기대효과 - 기대효과 및 활용방안

- 소규모 프로젝트나 조직에 부담되지 않는 경량화 된 서비스 개발로 효율적인 메모리 관리와, 생산성을 향상시킬 것으로 기대함
- 특허 및 실용신안 출원을 통해 개발된 기술의 지적 재산권을 확보함으로써, 기술적 리더십을 확립하고 장기적인 사업화 기반 마련 가능
- 컨테이너 기술 교육 자료로 활용하여 실습 중심의 학습의 학생들에게 인프라 관리 기술을 가르치는데 유용한 도구가 될 수 있을 것으로 기대함

참고문헌

[1]P. Sharma, et al., “Containers and virtual machines at scale: A comparative study,” in Proc. 17th Int. Middleware Conf. ACM, Trento, Italy, Nov. 2016

[2]최원석, 정혜진, 나연묵. (2019). 컨테이너 기반 클러스터 환경에서 효율적인 자원관리를 위한 오케스트레이션 방법. 한국차세대컴퓨팅학회 논문지, 15(2), 71-78.

[3]박영기, 양현식, 김영한. (2019). 컨테이너 네트워킹 기술의 성능비교. 한국통신학회논문지, 44(1), 158-170, 10.7840/kics.2019.44.1.158

[4]RedHat.What is Kubernetes?Retrievedfrom https://www.redhat.com/ko/topics/containers/what

[5]kubernetes:https://github.com/kubernetes/kubernetes

2024.03.13

감사합니다