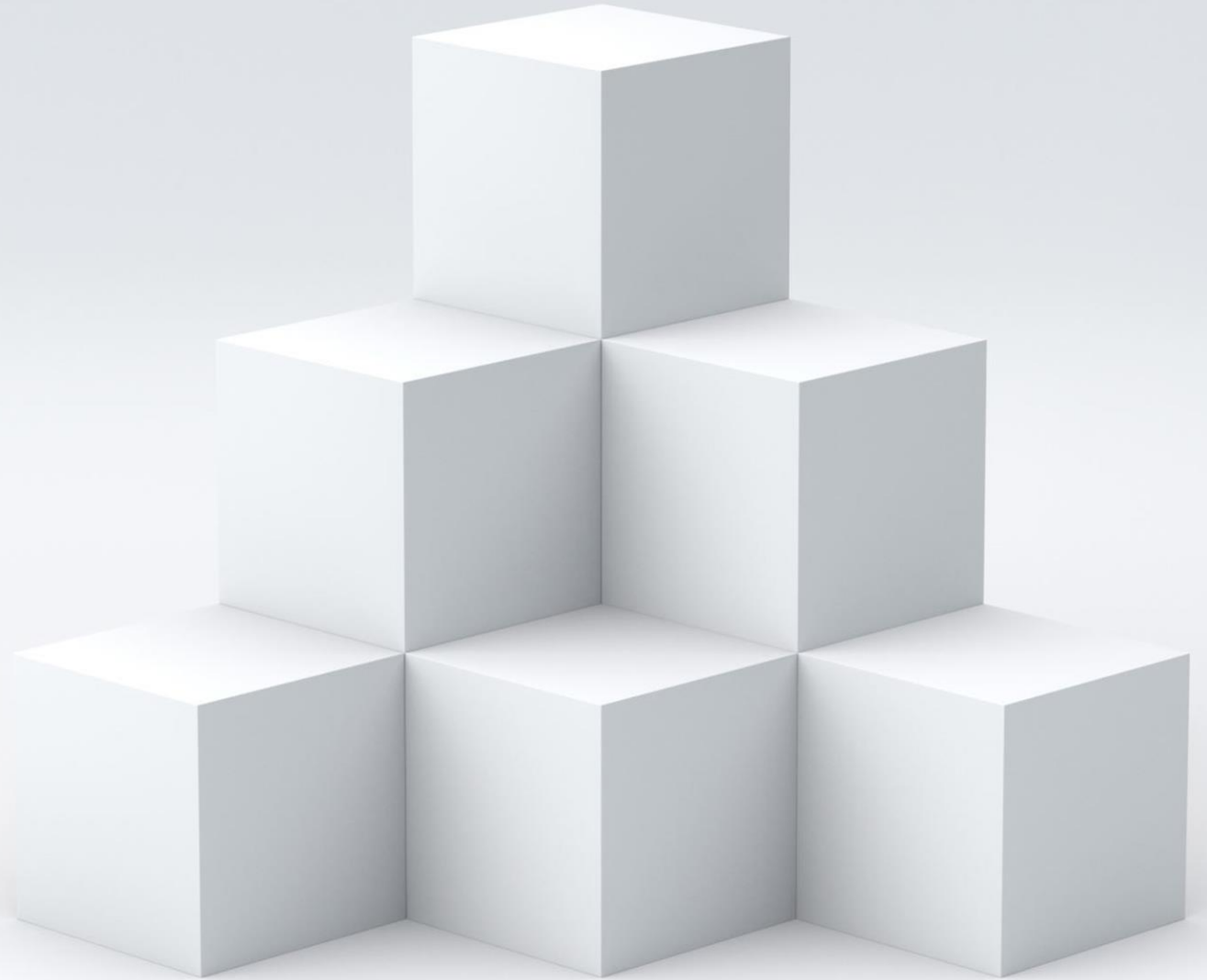


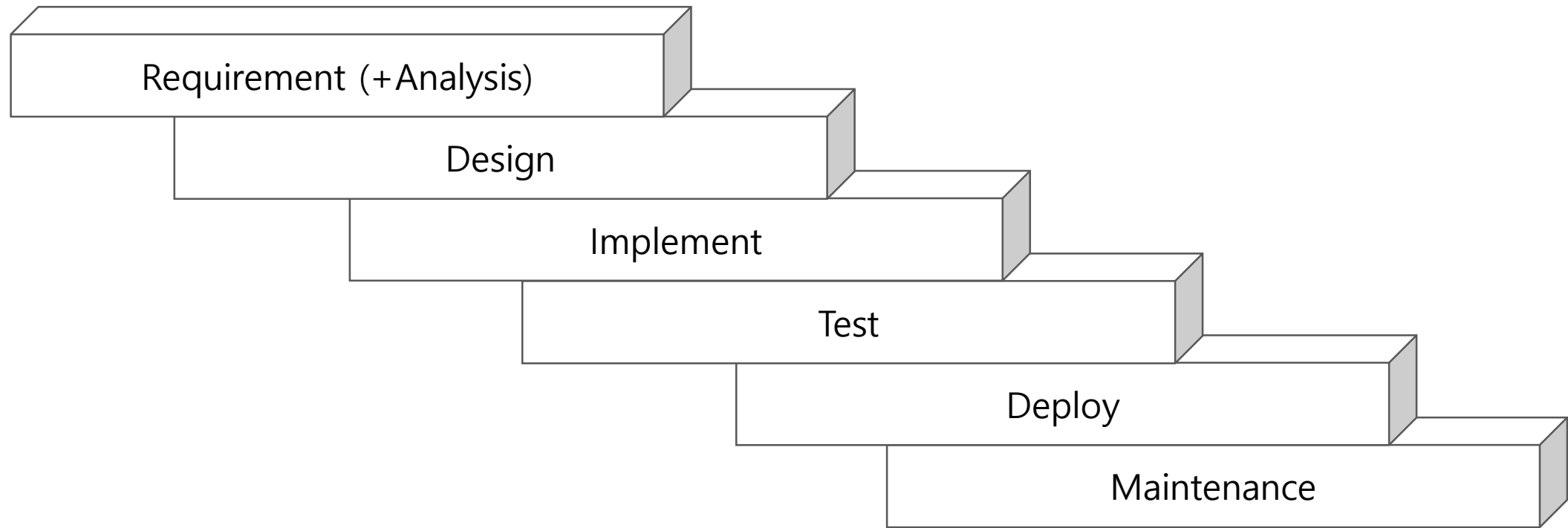
Software Engineering

Professor Kim Han-gyoo



Software Development methodology

1. Waterfall model



Waterfall Model에 따라 SW를 개발할 때의 갑갑한 문제들

- (1) High Risk + Uncertainty
- (2) 최종 단계에 가서야 동작하는 SW 가 생산됨
- (3) 테스트 단계에서 단점이 발견되어도 수정하기 어려움
- (4) Requirement들이 자주 변경되는 SW 개발에 적용하기 어려움
- (5) 복잡한 SW 개발에 부적절
- (6) Object Oriented 프로젝트에 부적절

Agile Methodology ???

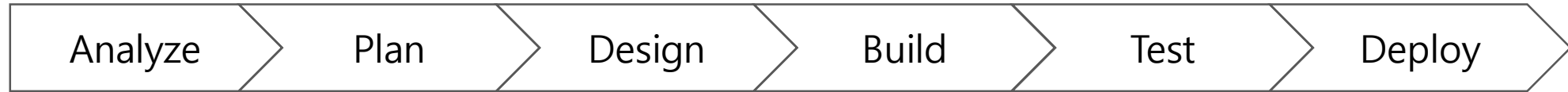
Waterfall 모델을 개선하려고 전체 개발 프로젝트를 여러 단계의 iteration으로 나눈 것

각 iteration 의 소요 시간은 동일해야 하며 대략 2~8주

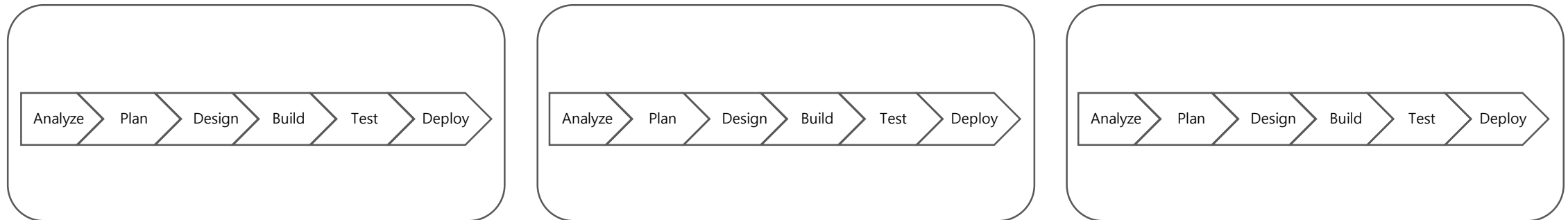
각 iteration 마다 정해진 기능이 동작하는 SW 생산됨

Software Development methodology

Waterfall model

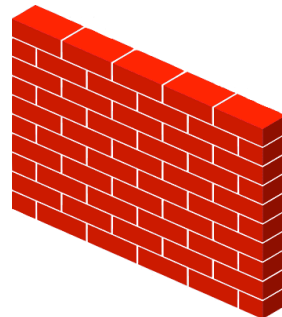


2. Agile model



(Development team)

: modification of a software product is inevitable



(Operation team)

: wants stable software

왜 Agile 모델? -> waterfall 모델이 비효율적이라는 것을 발견한 다음에 개선해 보려는 시도 -> 그런데 -> 원래의 methodology가 잘못된 것이므로 개선에 한계

이유는? 이상적인 methodology를 찾지 못해서가 아니라 당시의 개발 기술을 기반으로 적용할 수 있는 개발 방법론을 갖춘 것에 불과

결론 : 구현 기술을 근거로 개발 방법론을 만드는 것은 절대 피해야 함

문제를 해결하려는 데, 구현 엔지니어가 구현 기술이 이러하기 때문에 이렇게 개발해야 한다는 주장을 듣고 결정하면? -> 망하는 첩경

-> 그렇다면? -> 구현자가 아니라 데이터 사용자가 이런 것이 필요하다는 "목표"를 가능한 구체적으로 결정하고!, 만일 엔지니어가 구현할 수 없다고 하면 잘라 버리고 다른 엔지니어를 고용해야 함

지금(2005~2015년)의 개발 방법론

CI/CD continuous integration / continuous delivery(deploy)

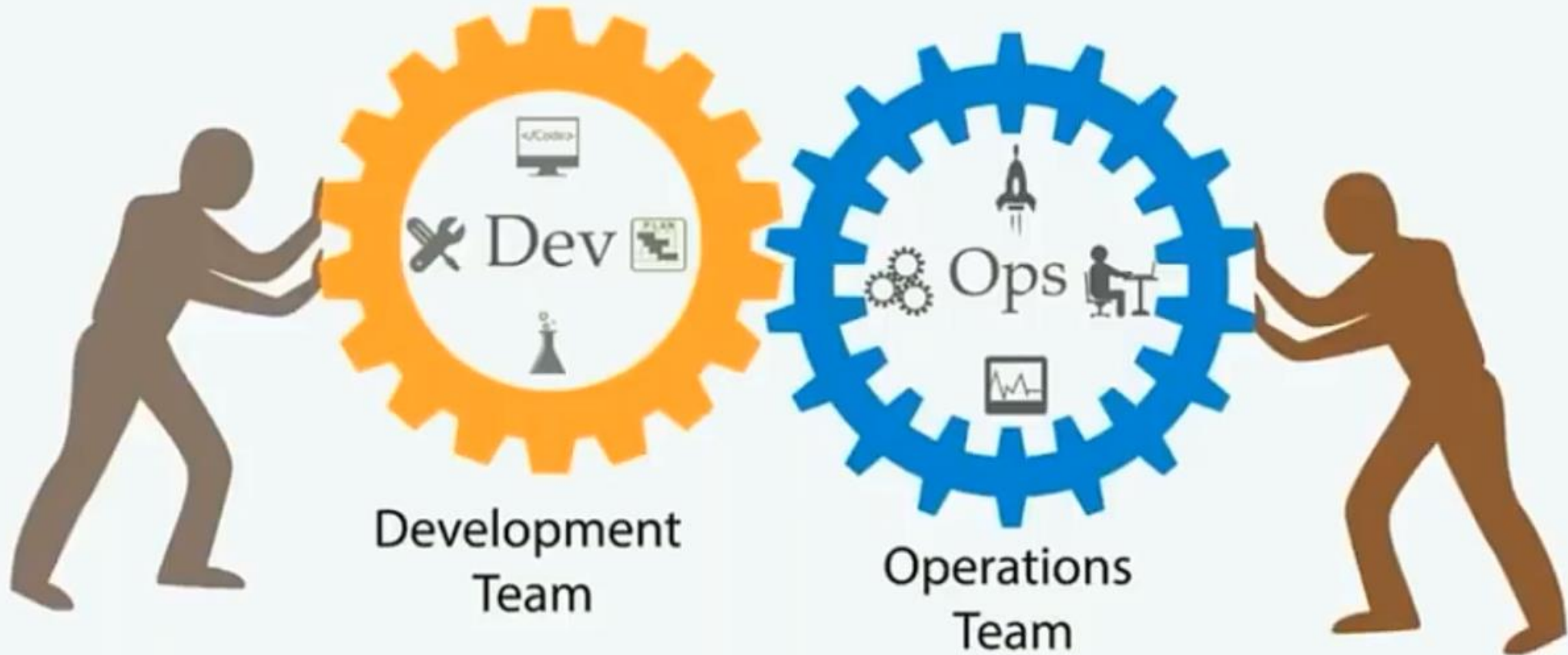
= 개발 팀 문화 ≠ 특정 technology

CI/CD 파이프라인은 더 자주 더 신뢰성 높은 수정된 코드를 deploy
할 수 있도록 DevOps 팀이 SW를 구현하는 방법론으로서 현재로서
는 가장 근사한 실제 개발 행위/팀 문화

OO SW design/implementation 기술 발전에 기인, 실제 복잡한 SW
를 개발한 경험자들의 누적된 경험에 의한 팀 문화적 산물

CI/CD 개발 문화가 성숙된 DevOps SW 개발 도구들의 발전으로 꽃
을 피우기 시작 -> DevOps 도구들을 사용하면 CI/CD가 강제적으로
수행됨

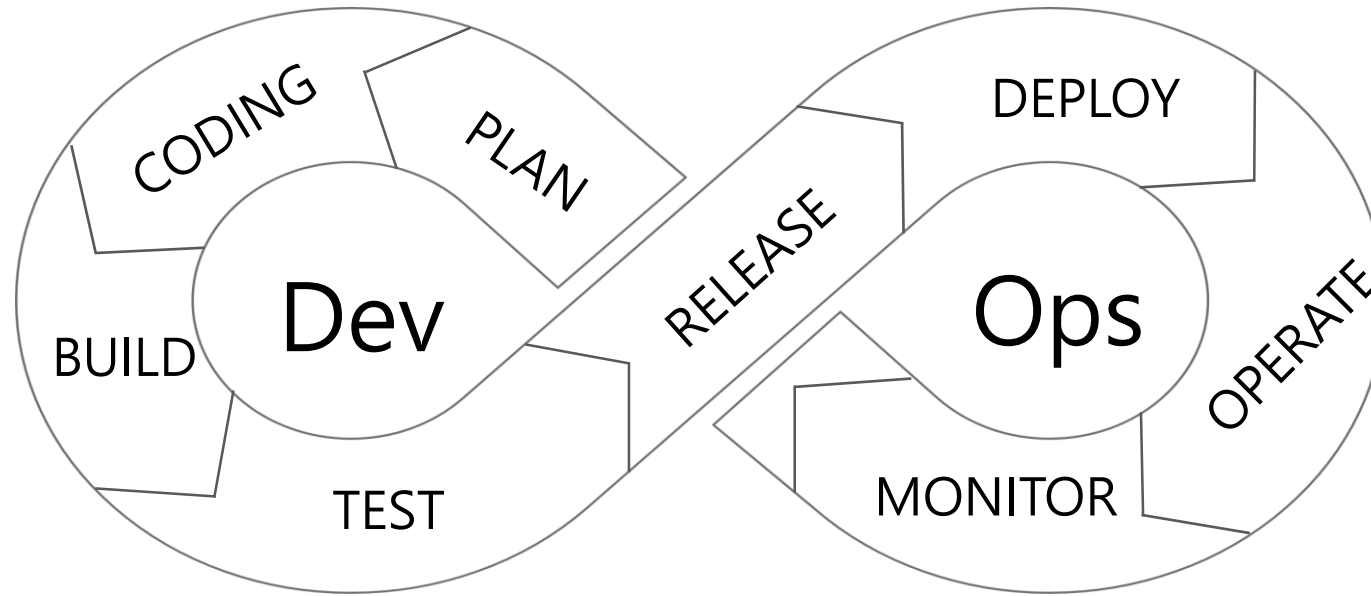
Solution is DevOps



특정 기술을 지칭하는 것이 아니라 개발 방법론

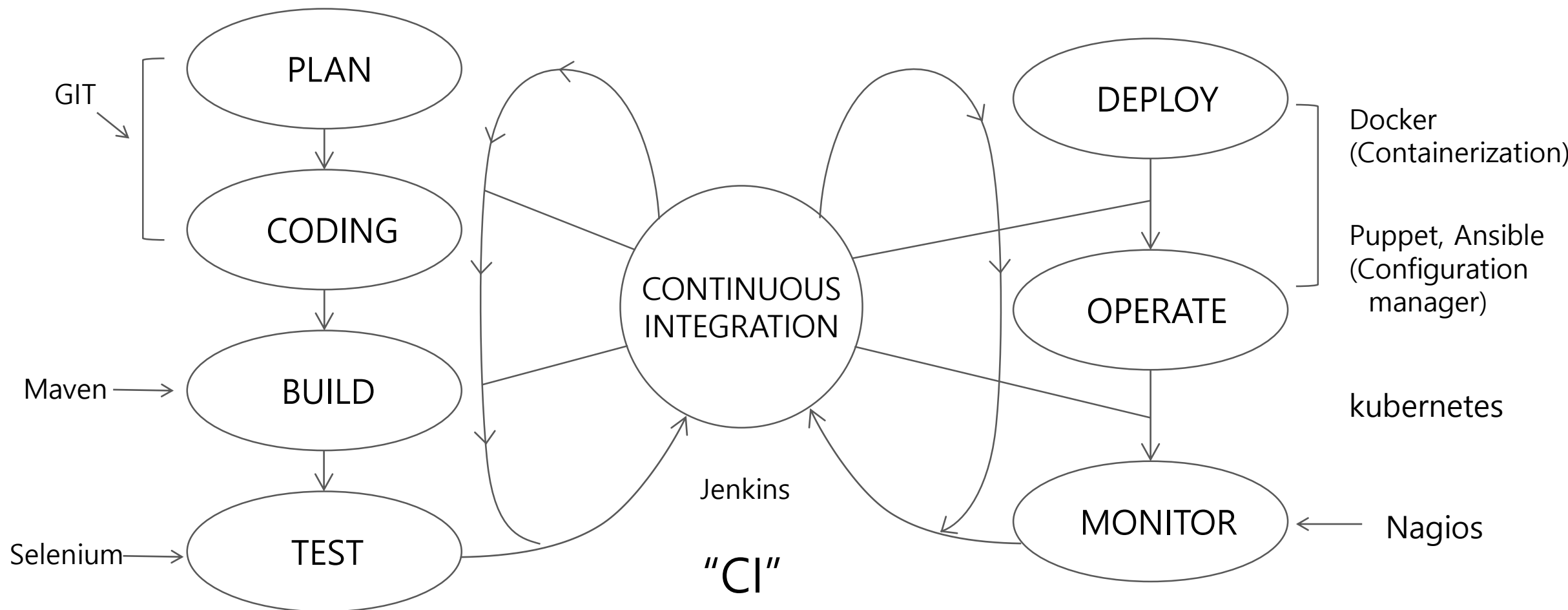


Developers



Operators
















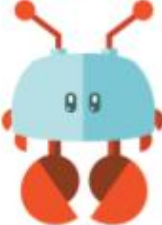















DevOps = CI / CD Methodology
realized by a set of automation tools



How DevOps works?

- (1) Continuous Development
 - (a) Version control – Git, SVN
 - (b) Automatic builder/packager to executables – Maven, Ant, Gradle
- (2) Continuous Testing – Selenium
- (3) Continuous Deployment – 실행 파일을 특정 서버 (테스팅 서버, production 서버)에 배포
 - (a) Containerization (vs VM) – Docker + Docker Swarm, kubernetes
 - (b) Configuration management – Puppet, Ansible
- (4) Continuous Monitoring – 네트워크 모니터링과 배포된 소프트웨어의 버그 및 사용자로부터의 feedback 수집 – Nagios
- (5) Continuous Integration – 위 모든 SW lifecycle의 도구들을 연합하고 연속적이고 자동적으로 각 작업이 수행되도록 함, 모든 위 도구들과 연동
 - Jenkins – 300개 이상의 plug-ins
 - cf) Kubernetes – Docker container orchestration manager (vs docker swarm)

DevOps Enabler Tools v2 (Caution!!!! : Consider only after DevOps mindset is established)

Infra-as-code	CI/CD	Test Automation	Container	Orchestration	Deployment	Measurement	ChatOps
 ANSIBLE	 Jenkins	 Se	 docker	 kubernetes	 DEPLOY  Octopus	 New Relic	 HU-BOT
 puppet	 shippable	 Cucumber	 Rocket	 MESOS	 vamp	 elasticsearch.	 LITA
 CHEF™	 Bamboo	 appium	 unik	 MARATHON	 DBmaestro DevOps for Database	 logstash	 COG v0.5
 SALTSTACK	 TeamCity	 APACHE JMeter™	 docker SWARM	 Elastic Beanstalk	 sumologic	 Kibana	 kloia
						 DATADOG	