

MVP 기술 전문가와 만나보는

Microsoft Build 2022 After Party

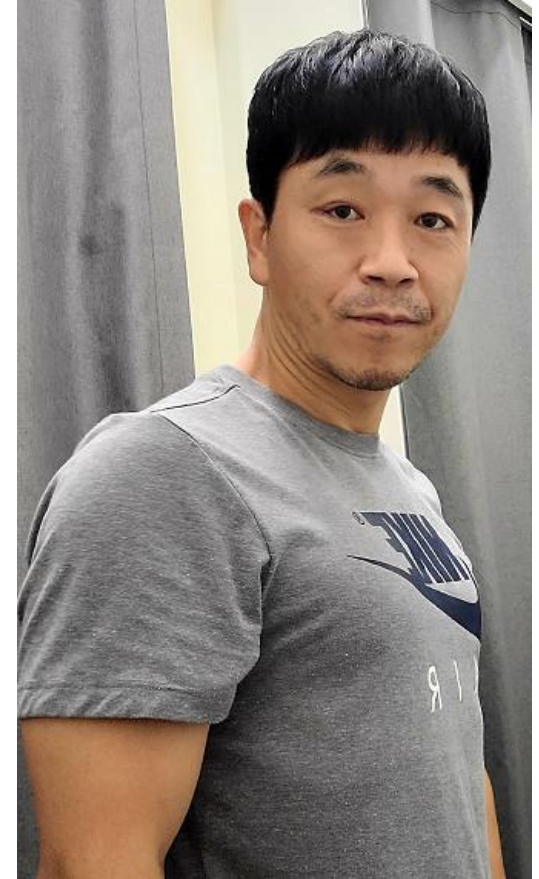
일시: 6월 25일 (토) 오후 1시-5시

장소: 한국마이크로소프트 11층



Cloud Native와 DevOps 그리고 Azure Container Apps를 활용한 App 현대화 이야기

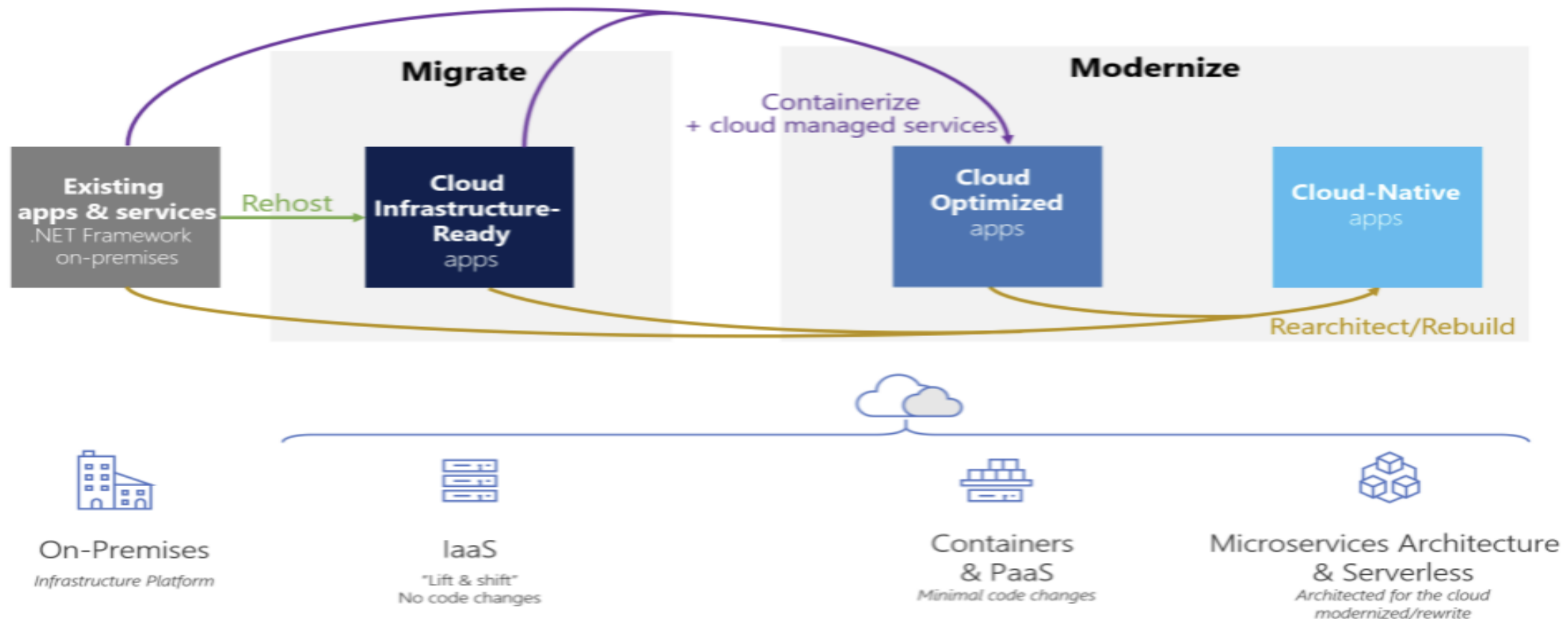
- 김영대 / 제로빅 / zerobig
- 베스핀글로벌 이사 / Azure SA팀 팀장
- 즐거운 Azure 생활 공동 운영자 (2020.1~)
- Azure Solutions Architect Expert (2019.10~)
- Azure DevOps Engineer Expert (2019.12~)
- 제로빅 블로그
- <https://zerobig-k8s.tistory.com/>
- 제로빅 이메일
- zerobig.kim@gmail.com / youngdae.kim@bespinglobal.com



Cloud Native – App Modernization

- 고객은 24시간 내내 즉각적인 피드백을 제시하며 혁신적인 제품과 서비스를 원함
- 이러한 시장 환경 및 고객 정서에 유연하고 신속한 대응을 위해서는 애플리케이션의 현대화가 필수적임
- 클라우드 및 컨테이너와 Kubernetes 기술을 통해 App Modernization(애플리케이션 현대화)을 가속화 할 수 있음
- 궁극적으로 Cloud-Native 환경에서 App Modernization을 이루기 위해서는 클라우드로 마이그레이션/혁신이 필요함
- App Modernization 및 마이그레이션 전략은 처해진 환경에 따라 또는 요구 사항 및 우선 순위에 따라 단계적으로 실현이 가능함

The Journey to the Cloud



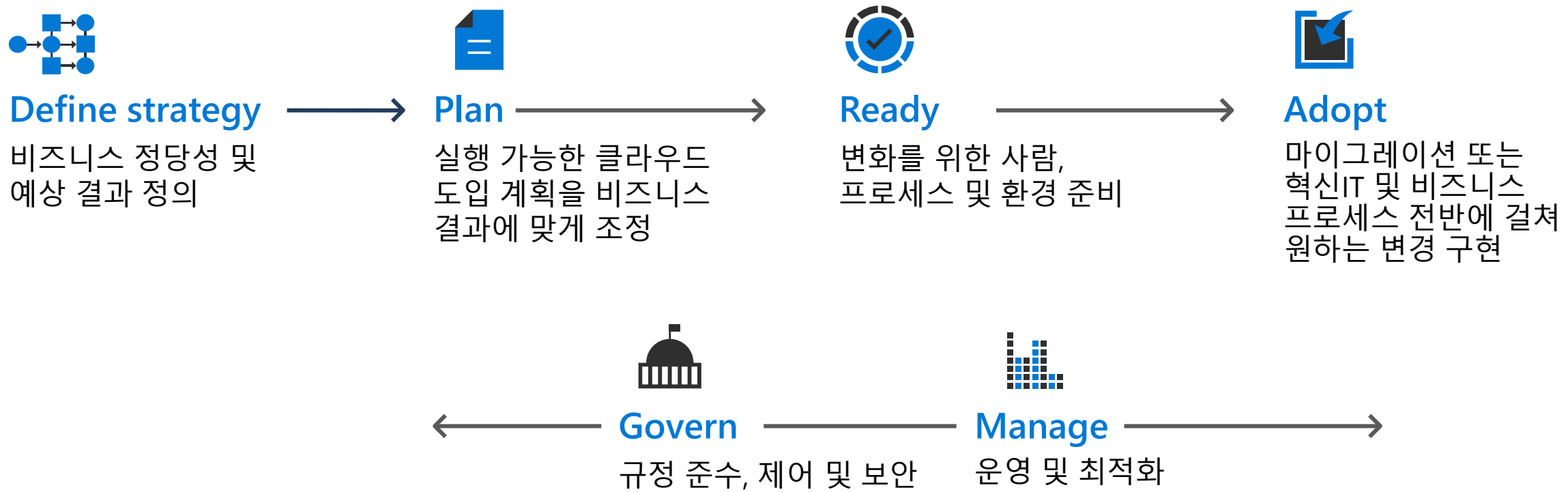
Cloud Native – App Modernization (계속)

Cloud로 Migration하기 위한 전략은 대표적으로 Rehost, Refactor, Rearchitect, Rebuild로 구분됨

패턴	내용	사용 시기	구현 기술 스택
Rehost	코드 변경 없이 애플리케이션을 있는 그대로 신속하게 클라우드로 마이그레이션. (Life and shift 라고도 함)	<ul style="list-style-type: none">• 애플리케이션 수정없이 신속하게 클라우드로 이동하는 경우• Azure IaaS 확장성을 활용할 수 있도록 디자인된 경우• 애플리케이션이 비즈니스에 중요하나 애플리케이션 기능을 즉시 변경할 필요가 없는 경우	
Refactor (현대화)	추가 클라우드 공급자 서비스를 통해 애플리케이션을 리팩터링하여 비용, 안정성 및 성능을 최적화 (Repackaging 이라고도 함)	<ul style="list-style-type: none">• 앱을 쉽게 다시 패키징하여 Azure에서 작동할 수 있는 경우• Azure에서 제공하는 혁신적인 DevOps 방법을 적용하려는 경우 또는 워크로드에 대한 컨테이너 전략을 사용하여 DevOps를 고려하는 경우• 리팩터링의 경우, 기존 코드 베이스의 이식성과 사용 가능한 개발 기술을 고려해야 함	Azure App Service 또는 AKS Azure SQL Managed Instance, Azure Database for MySQL, Azure Database for PostgreSQL 및 Azure Cosmos DB
Rearchitect (현대화)	마이그레이션을 위한 아키텍처 변경은 애플리케이션 코드 베이스를 수정하거나 확장하고 클라우드 플랫폼에 맞게 최적화 하여 확장성을 개선	<ul style="list-style-type: none">• 새로운 기능을 통합하거나 클라우드 플랫폼에서 효과적으로 작동하기 위해 애플리케이션에 주요 수정 버전이 필요한 경우• 기존 앱 투자를 사용하려는 경우 확장성 요구 사항을 충족하고, 혁신적인 DevOps 관행을 적용하여 민첩성을 향상하고, 가상 머신의 사용을 최소화	관계형 및 비관계형 데이터베이스를 완전히 관리되는 데이터베이스 솔루션(예: SQL Managed Instance, Azure Database for MySQL, Azure Database for PostgreSQL 및 Azure Cosmos DB)으로 재설계
Rebuild (현대화)	클라우드 네이티브 기술을 사용하여 애플리케이션을 처음부터 재구축	<ul style="list-style-type: none">• 비즈니스 혁신(Azure에서 제공하는 DevOps 방법 포함)을 신속하게 진행하고, 클라우드 네이티브 기술을 사용하여 새 애플리케이션을 빌드하고, AI, 블록체인 및 IoT의 발전을 활용할 준비가 된 경우	Azure Functions, AI, SQL Managed Instance 및 Azure Cosmos DB

Microsoft Cloud Adoption Framework for Azure

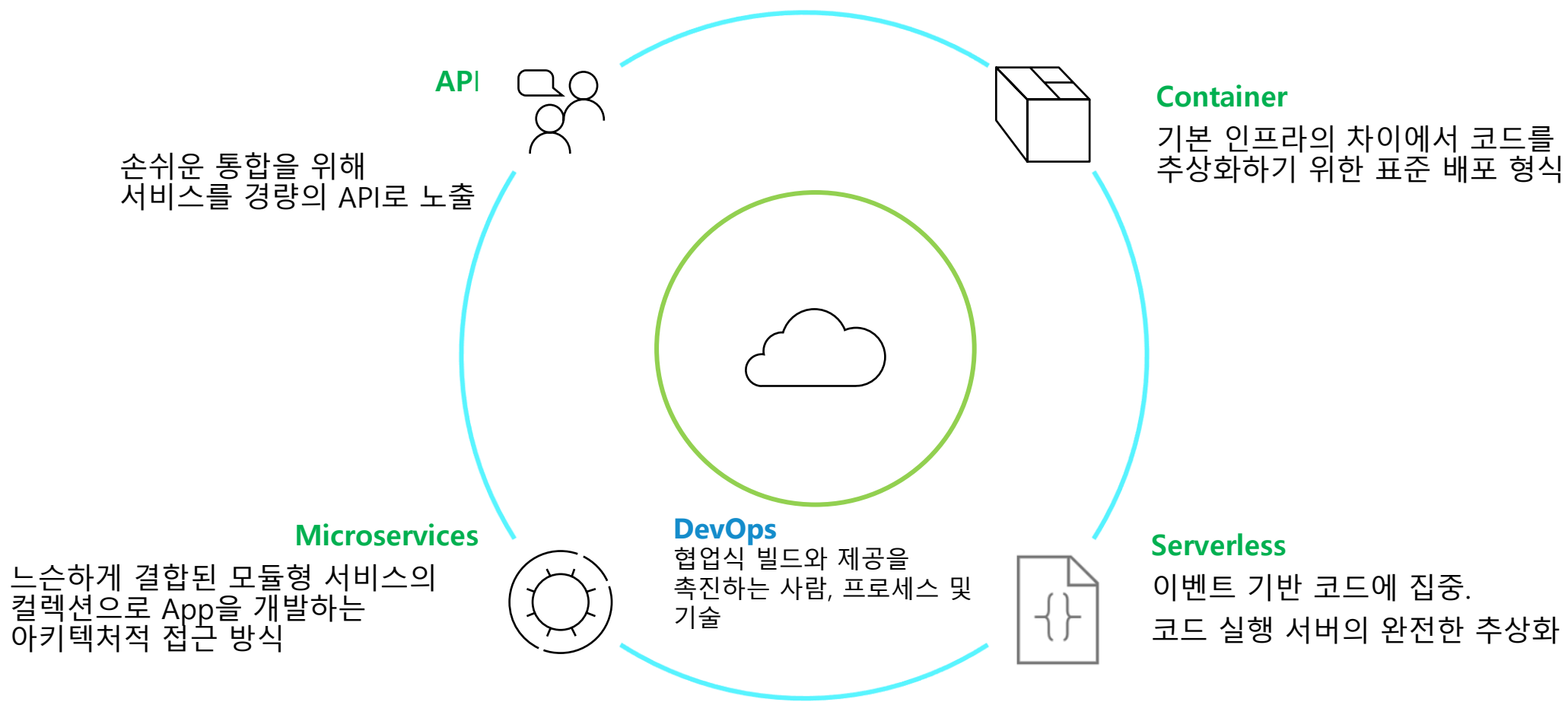
고객이 클라우드에서 성공하는 데 필요한 비즈니스 및 기술 전략을 수립하고 구현하는 데 도움이 되는 입증된 비즈니스 및 기술 지침.



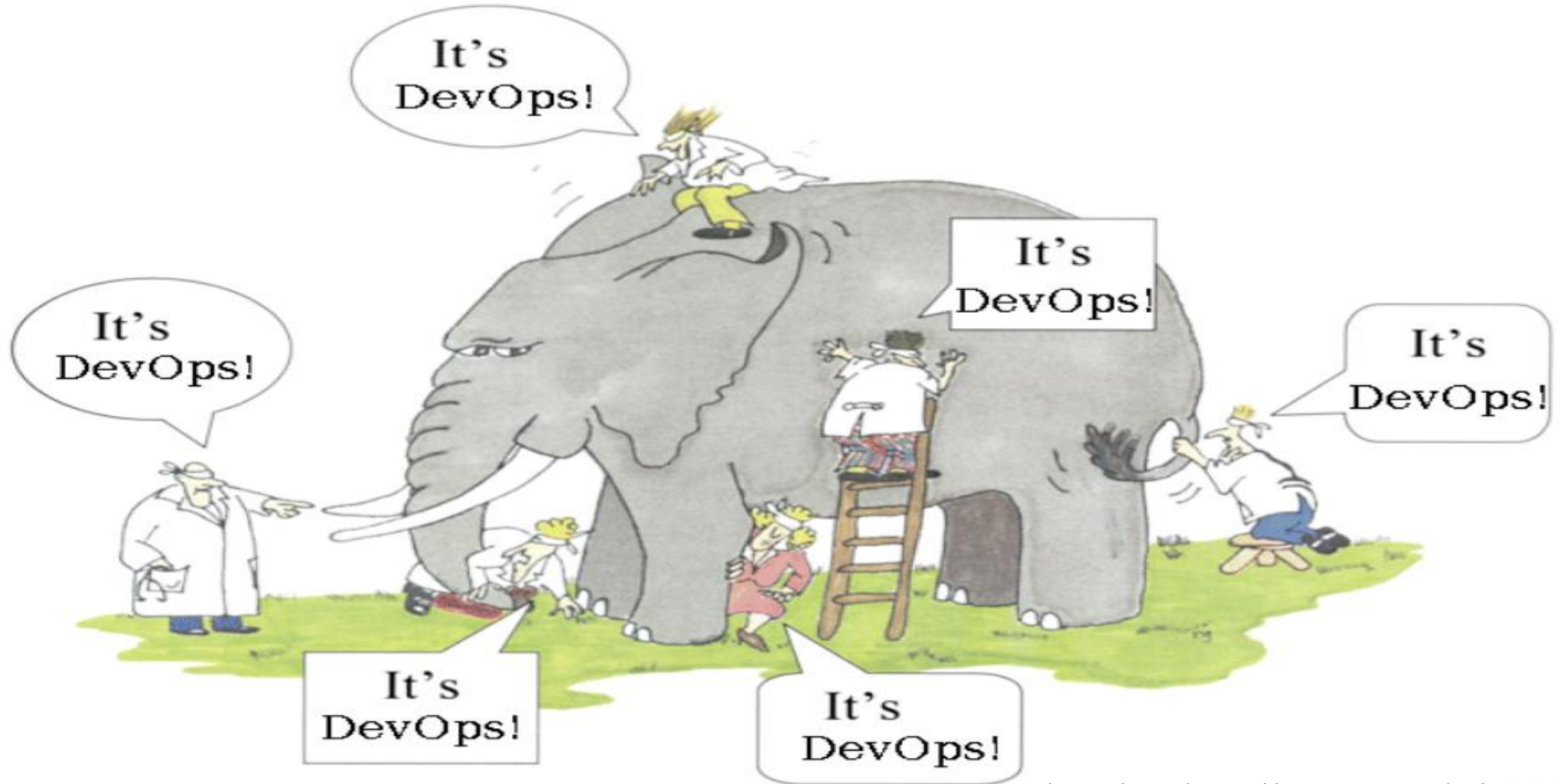
각 모듈은 클라우드 채택의 전체 수명 주기를 통해 비즈니스를 발전시키는 **반복적인 단계**

Cloud-Native 환경에서의 App Modernization

애플리케이션 코드 & 종속성을 컨테이너로 패키징하고, 마이크로서비스로 배포하고, DevOps 프로세스 & 도구를 사용하여 이를 관리하는 것



DevOps의 수 많은 정의...*Buzzword



<https://devopsdays.org/blog/wp-content/uploads/2010/05/itsdevops.png>

DevOps is the **union of people, process, and products** to enable **continuous delivery of value** to your end users - Donovan Brown

***Buzzword** : 명확한 합의와 정의가 없는 용어이다. 한편, 처음에는 명확하게 정의된 용어도, 본래의 의미에서 벗어나 사용되게 되면, 버즈워드로 간주된다.

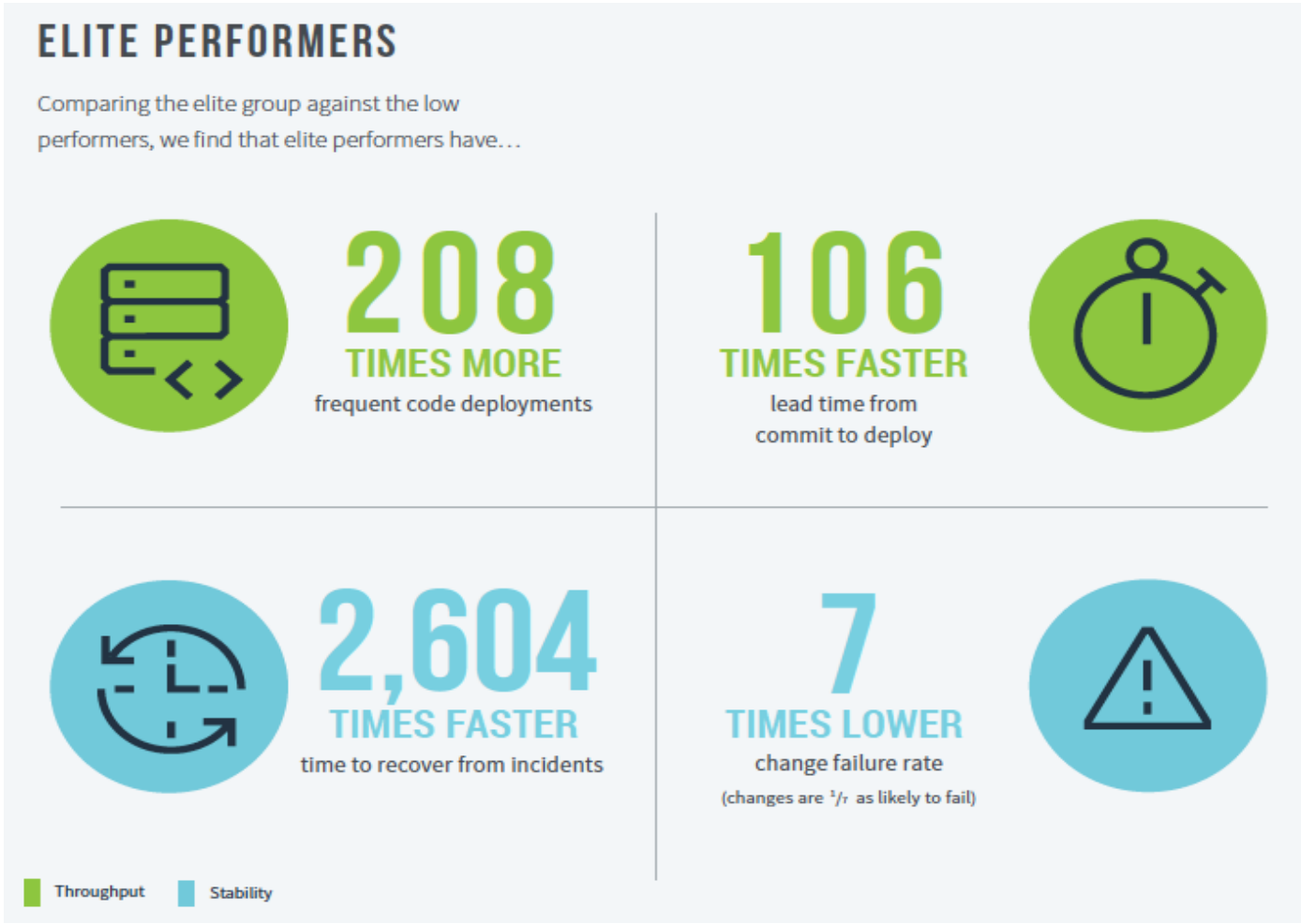
#BuildAfterParty

주요 IT 기업의 DevOps 정의...

- A compound of development (Dev) and operations (Ops), DevOps is **the union of people, process, and technology to continually provide value to customers.**
- ...By adopting a DevOps **culture** along with DevOps **practices** and **tools**, teams gain the ability to **better respond to customer needs...** and **achieve business goals faster.** [MS Azure]
- DevOps는 소프트웨어 제공 **속도**를 높이고 서비스 안정성을 개선하며 소프트웨어 이해관계자 간 **공유 소유권**을 구축하기 위한 **조직적, 문화적 방법론**입니다. [Google]
- DevOps는 애플리케이션과 서비스를 **빠른** 속도로 제공할 수 있도록 조직의 역량을 향상시키는 **문화 철학**, 방식 및 도구의 조합입니다. [AWS]
- DevOps는 개발 및 운영 팀이 협업으로 **속도와 품질, 제어를 유지**하며 애플리케이션을 구축, 테스트, 배치 및 모니터링하는 소프트웨어 제공을 위한 점점 더 보편화되고 있는 **접근 방법**입니다.
- ... 공통 유스 케이스에는 **클라우드 네이티브** 및 모바일 애플리케이션, 애플리케이션 통합, **현대화** 및 다중 클라우드 관리가 포함됩니다. [IBM]
- DevOps is a **set of practices** that works to **automate** and integrate the **processes** between software development and IT teams, so they can **build, test, and release software faster** and **more reliably.** [Atlassian]
- DevOps is an approach to **culture, automation, and platform design** intended to deliver increased business value and responsiveness through **rapid, high-quality service delivery.** This is all made possible through fast-paced, iterative IT service delivery. **DevOps means linking legacy apps with newer cloud-native apps and infrastructure.** [Redhat]

DevOps Elite Performers 사례

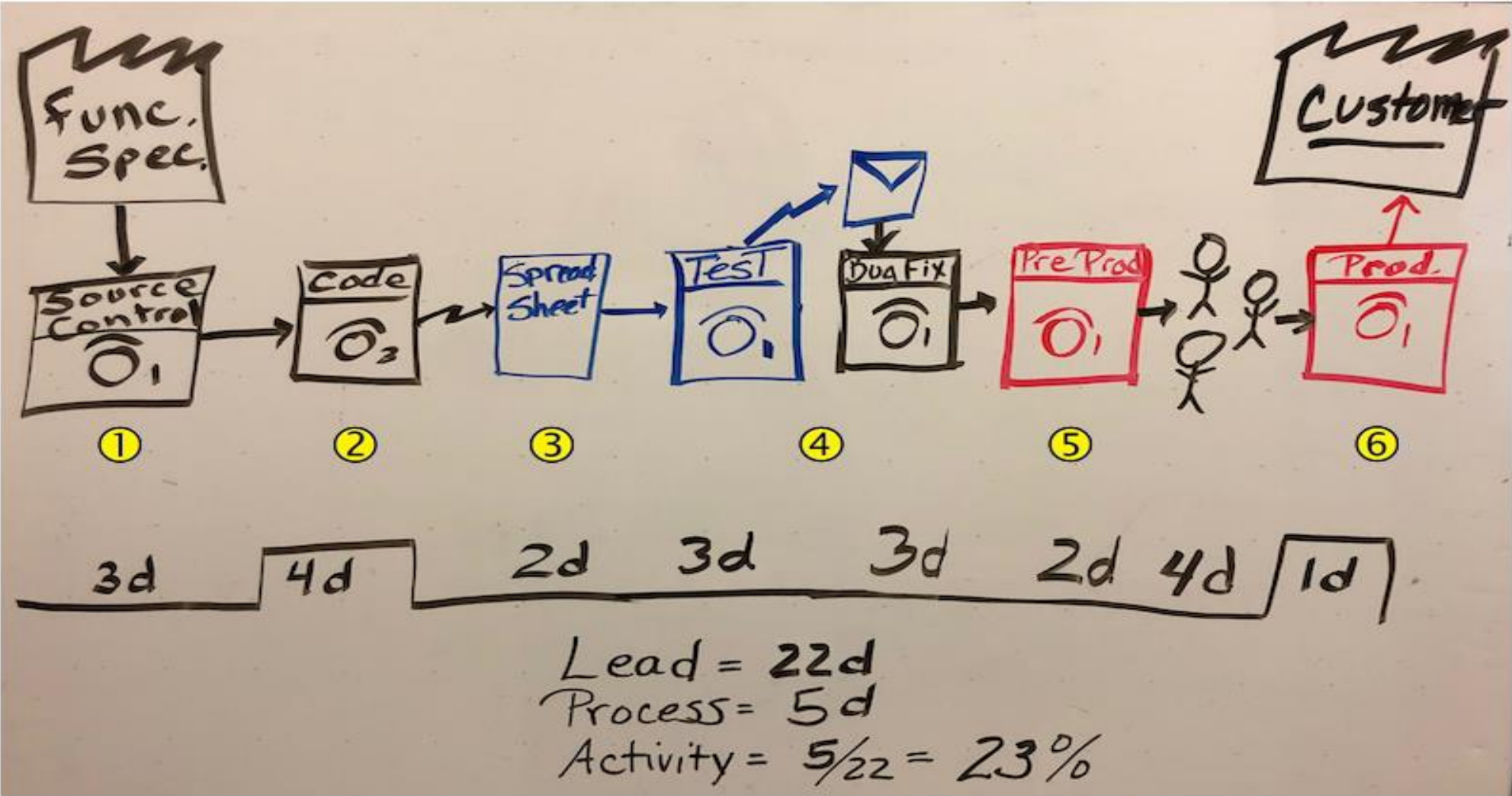
벤치마크는 높은 성과와 엘리트 성과 사이의 격차가 커지는 것을 보여준다.



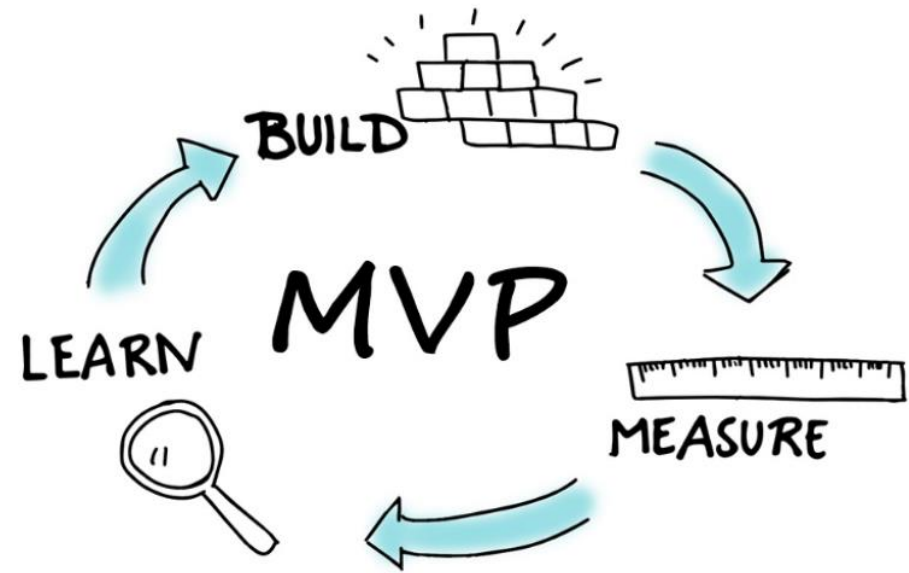
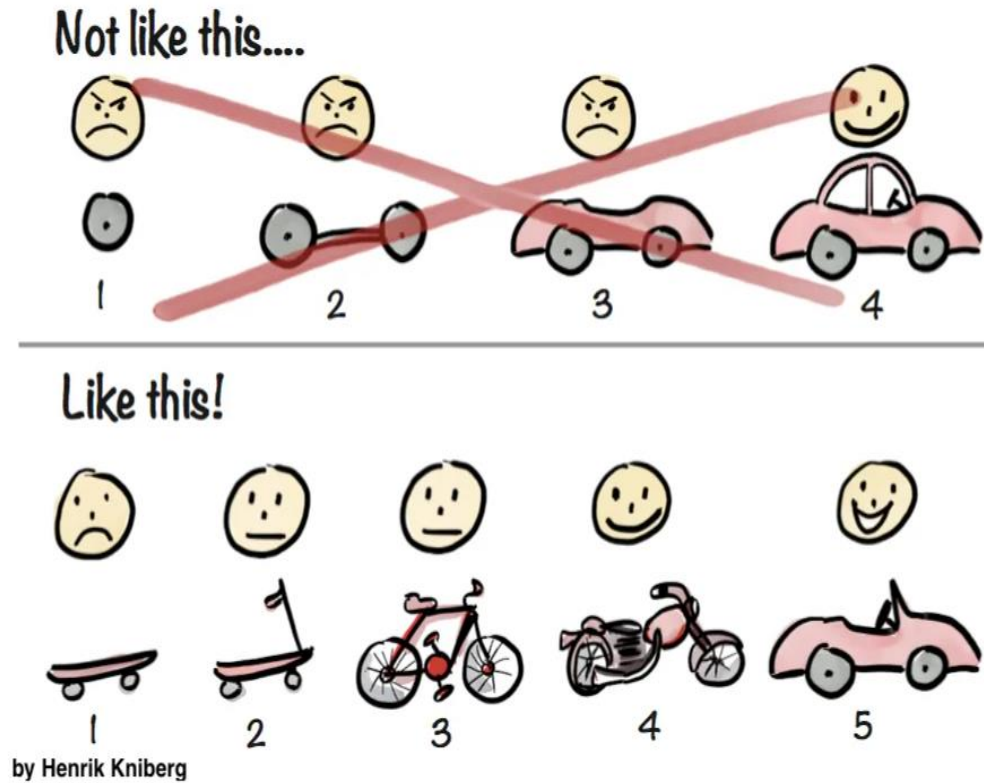
- 잦은 배포 주기
- 신속한 커밋 코드 프로덕션 반영 및 리드 타임
- 사고 발생 후 보다 빠른 서비스 복구
- 낮은 변경 실패율

Value Stream Map (가치 스트림 맵)

팀이 프로세스 단계 어디에서 가치를 만들고 있고 어디에서 낭비가 존재하는지를 시각적으로 보여주는 것이다. 과정의 목적은 최소한의 낭비로 고객에게 최대한의 가치를 전달하는 프로세스에 도달하는 것이다.



“MVP(Minimum Viable Product)”라는 용어는 2001 년 Frank Robinson에 의해 처음 소개 되었으며 “최소 실행 가능한 제품”을 의미한다. 제품 개발에 대한 기존의 접근 방식과 달리 MVP는 신생 기업이 제공하는 제품이 많은 시간을 소비하지 않고 돈을 낭비하지 않고 잠재적인 사용자가 실제로 구매할 수 있는 제품인지 확인할 수 있도록 한다.

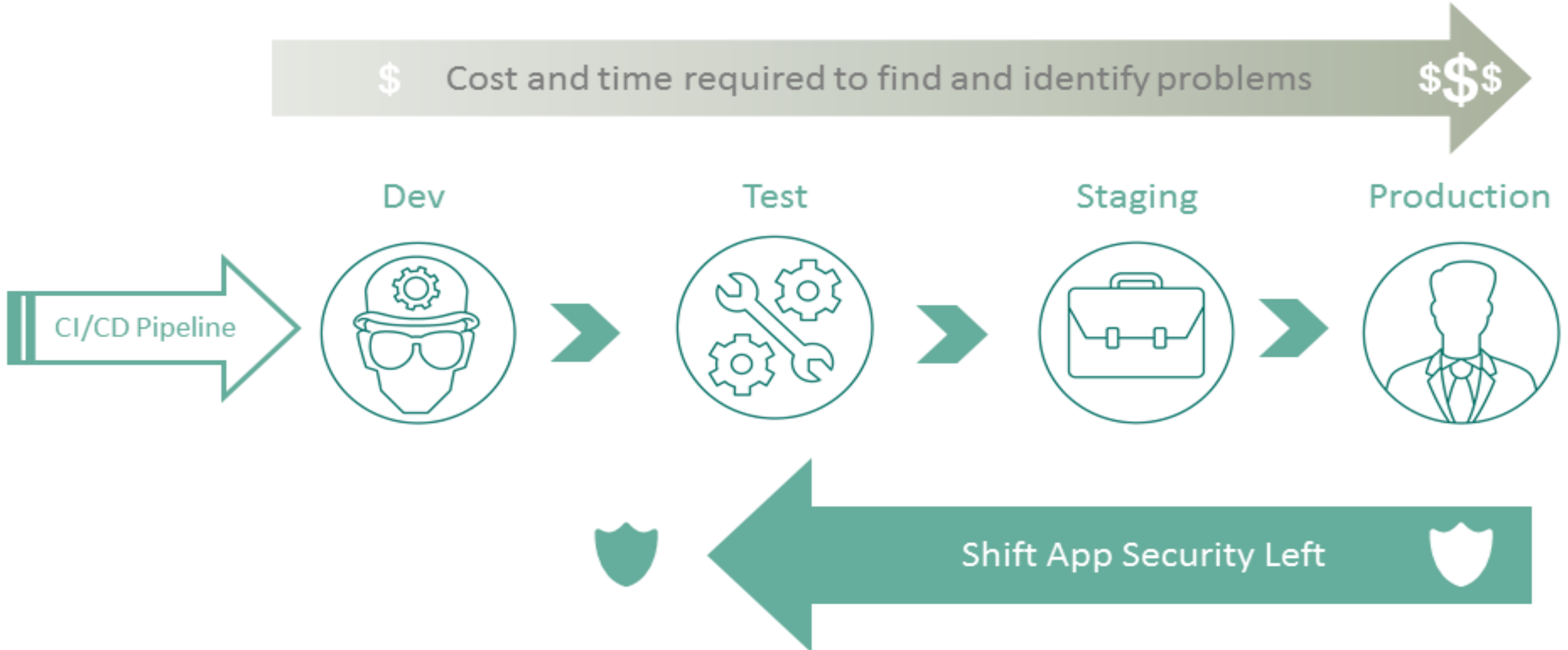


<https://thefactory.tech/how-to-define-your-minimum-viable-product/>

Author/Copyright holder: Henrik Kniberg. Copyright terms and licence: All rights reserved

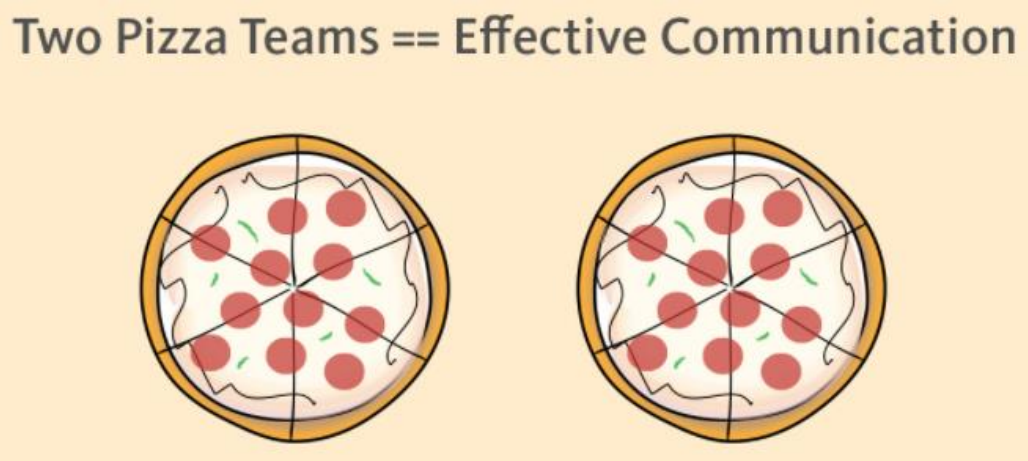
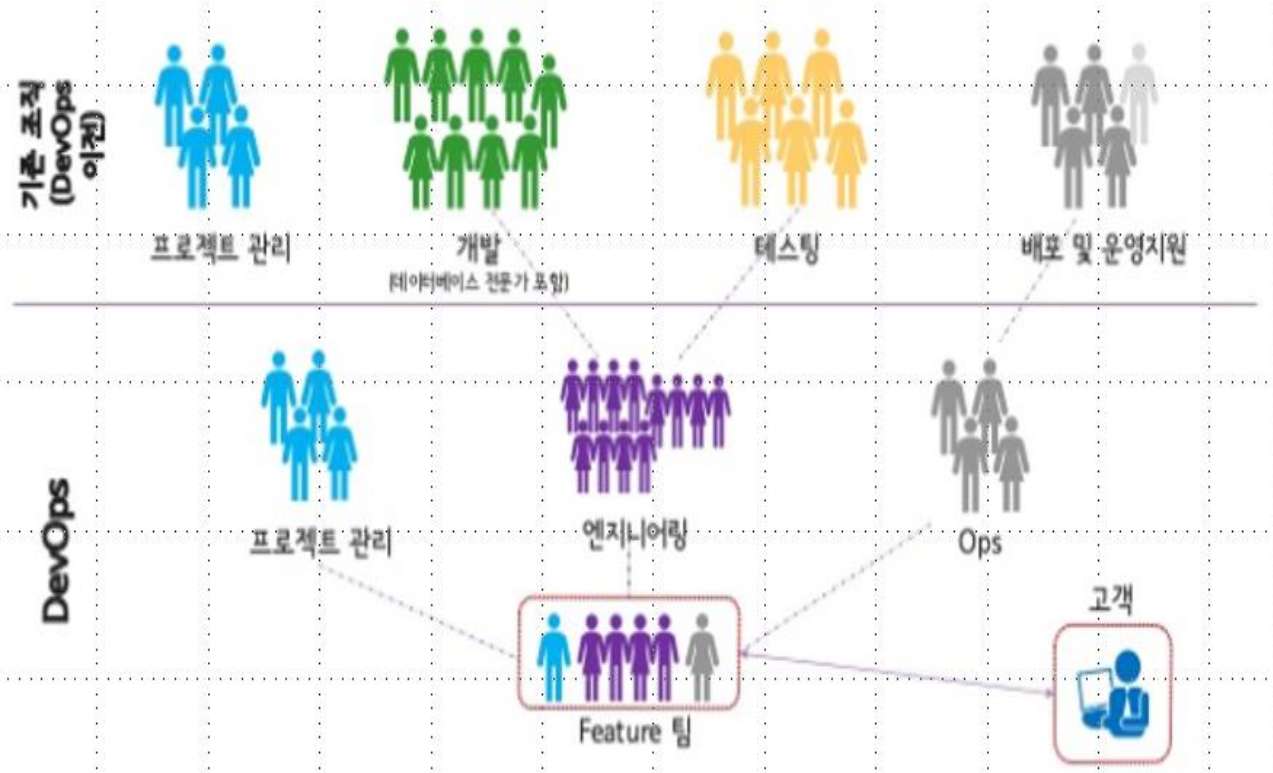
Shift Left

- 애플리케이션 개발 생명 주기 후반(Right)에 수행되던 활동 들을 초반 단계(Left)에서 수행(코드의 안정성과 보안성을 구현)하여 시간과 비용을 절감시키려는 **DevOps** 관행이다.
- 심각한 결함을 조기에 발견할 수 있으며 수정 비용이 저렴하고 일정을 위험에 빠뜨리기 전에 확인 가능하다.



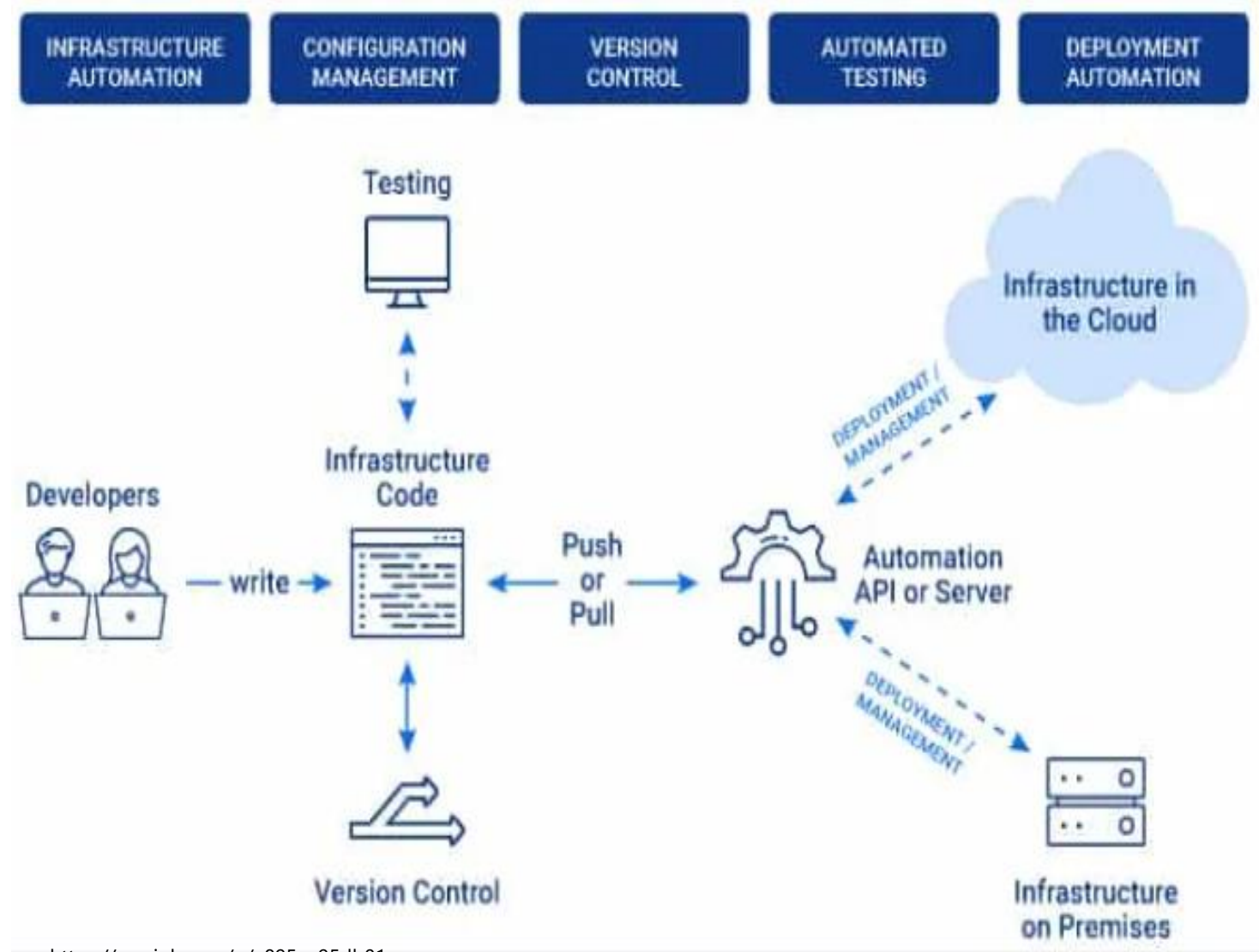
DevOps를 위한 역할과 책임의 이동 : Cross-functional Team

빌드, 검증, 배포 및 전달 단계를 통해 소프트웨어 변경의 이동을 더욱 능률화하며 프로덕션 지원 전 과정을 통해 설계에서부터 소프트웨어 애플리케이션의 모든 소유권을 가진 **교차 기능(cross-functional) 팀**에게 권한을 부여함으로써 **민첩한 개발 관행을 확장**한다.



Infrastructure as Code

- IaC는 인프라의 모양을 정의하기 위한 **선언적 모델**이다. 이 방법에서는 모든 연결을 찾을 수 있다. 다른 프로젝트, 다른 팀과 구성의 비트와 조각을 공유 할 수 있다. 다른 사람들에게 변경사항을 검토하게 할 수 있다.
- IaC는 인프라와 모든 비트 및 조각 (네트워크에서 VM으로의 로드 밸런서)을 환경에 대한 **단일 진실 공급원(SSOT)**으로 관리하는 방법이다.



<https://morioh.com/p/a895aa95db01>

SSOT(Single Source Of Truth)

정보 모형과 관련된 데이터 스키마를 모든 데이터 요소를 한 곳에서만 제어 또는 편집하도록 조직하는 관례를 이른다. 데이터 요소로의 가능한 연결(관계 스키마 내의 다른 영역이나 원거리의 연방 데이터베이스에서도 올 수 있음)은 모두 참조로만 이루어진다. 데이터가 위치한 다른 모든 곳들은 단지 으뜸되는 "진실 공급원"의 위치를 참조하기만 하므로, 으뜸되는 위치의 데이터 요소를 갱신하면 수정 사항 반영을 빠뜨릴 데이터의 사본이 존재할 가능성 없이 시스템 전체에 전파되게 된다.?????????? **WT...H**

조직 내 구성원이 동일한 정보에 접근할 수 있도록 보장 (Atlassian)


일반적으로 코드 저장소를 가리키는 용어

Autodesk + FMI에 따르면 [재작업의 52%](#)는 취약한 데이터와 잘못된 의사소통으로 인해 발생




멱등성 (Idempotence)

멱등성(Idempotence)이란, 몇 번이고 같은 작업을 실행하더라도 같은 결과로 수렴되는 것을 의미하며,



서버와 관련해서 말하면 몇 번이고 프로비저닝 작업을 실행하더라도 같은 서버 상태가 된다는 것을 가리킨다.



Terraform이나 Chef, Ansible, Puppet과 같은 도구는 모두 이러한 방식에 기반을 두고 있다.



Immutable Infrastructure

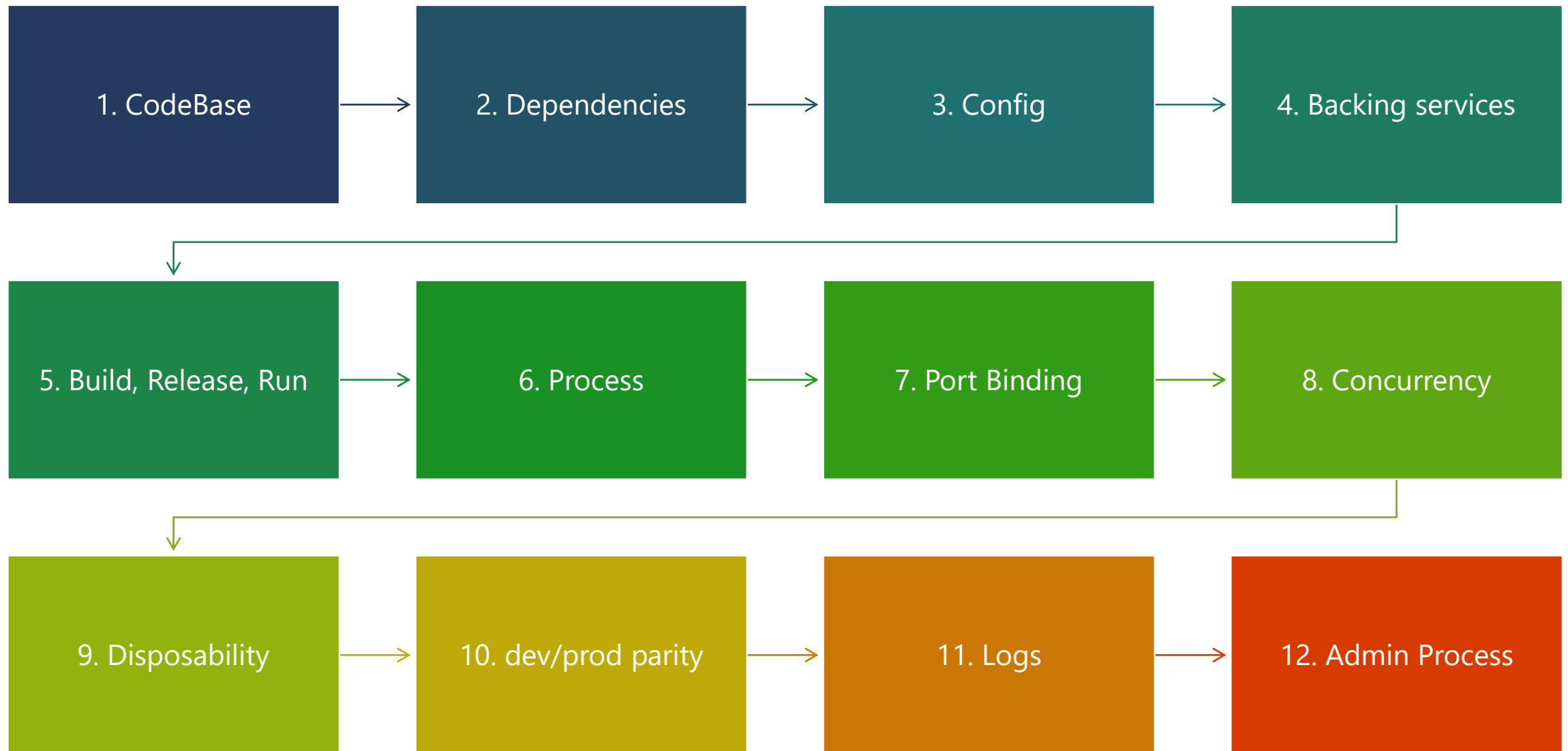
- **Immutable Infrastructure**란, 불변하는 인프라라는 의미며, 서버를 예로 들면 일단 서버를 구축했으면 그 후에는 서버의 소프트웨어에 변경을 가하지 않는 것을 가리킨다.
- 즉, 기존 서버에 변경을 가하는 대신에 새로운 서버를 만들어 기존 서버를 모두 대체하는 접근 방식이다.
- Immutable Infrastructure를 실현하는 대표적인 기술로 컨테이너 기술(**Docker** 등)이 있다.



Pet vs Cattel



12 Factors



Azure Container Instance

Azure Container Instances는 간단한 애플리케이션, 작업 자동화 및 빌드 작업 등 격리된 컨테이너에서 작동할 수 있는 모든 시나리오에 적합한 솔루션입니다.



서버를 관리할 필요 없이 컨테이너 실행

ACI(Azure Container Instances)에서 워크로드를 실행하면, 워크로드 실행용 인프라 관리에 시간을 쓰지 않고 애플리케이션 설계 및 구축에 집중할 수 있습니다.



주문형 컨테이너로 민첩성 증가

단일 명령을 사용하여 전례 없는 단순성과 속도감으로 클라우드에 컨테이너를 배치할 수 있습니다. ACI를 사용하면 귀사에서 필요할 때마다 까다로운 워크로드에 대해 추가적인 컴퓨팅을 제공할 수 있습니다. 예를 들어, Virtual Kubeletuse과 함께 ACI를 사용하면, 트래픽이 급증할 때 귀사의 Azure Kubernetes Service(AKS) 클러스터로부터 탄력적으로 확장된 지원을 받을 수 있습니다.



하이퍼바이저 격리를 통해 애플리케이션 보호

경량 컨테이너의 효율성을 유지하면서 컨테이너 워크로드에 대한 가상 머신의 보안을 확보할 수 있습니다. ACI는 각 컨테이너 그룹에 하이퍼바이저 격리 기능을 제공하여 컨테이너가 커널을 공유하지 않고 격리된 상태로 실행되도록 합니다.

Azure Webapp for Container

Azure Web App for Containers는 Azure App Service 플랫폼의 구성 요소이며 웹 애플리케이션의 런타임, 프레임워크, 도구를 더 많이 제어해야 하는 개발자에게 탁월한 옵션입니다.



몇 초 안에 Azure에 배포

컨테이너 기반 웹앱을 간단히 배포할 수 있습니다. Docker 허브 또는 전용 Azure Container Registry에서 컨테이너 이미지를 끌어오면 Web App for Containers 이 기본 종속성을 포함하는 컨테이너화된 앱을 몇 초 안에 프로덕션으로 배포합니다. 플랫폼에서 OS 패치, 용량 프로비전 및 부하 분산을 자동으로 처리합니다.

더 빠르게 업데이트 제공

Docker 허브, Azure Container Registry 및 Visual Studio Team Services의 CI/CD(연속 통합/연속 배포) 기능을 통해 컨테이너 이미지 배포를 자동화하고 간소화하세요. App Service는 소스 코드가 변경될 때마다 앱이 업데이트되도록 선택한 레지스트리와 연결을 생성합니다. 스테이징 환경에서 성능 및 품질 테스트를 예약하고 배포 슬롯을 사용하여 몇 초 안에 스테이징에서 프로덕션으로 전환하거나 가동 중지 시간 없이 이전 버전으로 롤백하세요.

필요 시 쉽게 확장

애플리케이션의 필요에 따라 자동으로 확장합니다. 세분화된 크기 조정 규칙을 사용하여 워크로드 내의 최대 작업량을 자동으로 처리하고 사용량이 적은 시간 동안은 비용을 최소화할 수 있습니다. 마우스 클릭 몇 번으로 여러 위치에서 데이터 및 호스트 서비스를 배포하세요.

Azure Kubernetes Service

AKS(Azure Kubernetes Service)는 운영 오버헤드를 Azure로 오프로드하여 Azure에서 관리되는 Kubernetes 클러스터 배포를 단순화합니다.



인프라를 관리할 필요가 없고 [KEDA](#)를 통해 이벤트 구동 자동 스케일링 및 트리거를 추가하는 기능이 포함된 용량의 [탄력적 프로비저닝](#)



[Visual Studio Code Kubernetes 도구](#), [Azure DevOps](#), [Azure Monitor](#)를 통한 더 빠른 엔드투엔드 개발 환경



[Azure Active Directory](#)를 사용한 가장 포괄적인 인증과 권한 부여 기능 및 [Azure Policy](#)를 사용하여 여러 클러스터에서 동적 규칙 적용



다른 클라우드 공급자보다 많은 지역에서 사용 가능

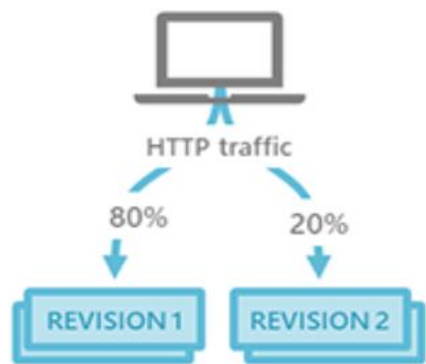
Azure Container Apps - Overview

Azure Container Apps를 사용하면 서버리스 플랫폼에서 마이크로서비스 및 컨테이너화된 애플리케이션을 실행할 수 있습니다. **Focus on apps, not infrastructure.**



Azure Container Apps: Example scenarios

PUBLIC API ENDPOINTS



HTTP requests are split between two versions of the container app where the first revision gets 80% of the traffic, while a new revision receives the remaining 20%.

AUTO-SCALE CRITERIA

Scaling is determined by the number of concurrent HTTP requests.

BACKGROUND PROCESSING



A continuously-running background process that transforms data in a database.

AUTO-SCALE CRITERIA

Scaling is determined by the level of CPU or memory load.

EVENT-DRIVEN PROCESSING

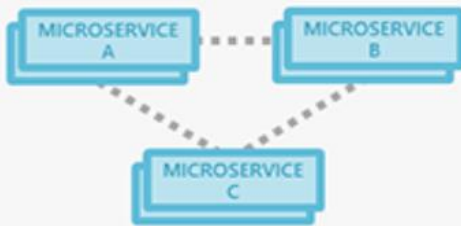


A queue reader application that processes messages as they arrive in a queue.

AUTO-SCALE CRITERIA

Scaling is determined by the number of messages in the queue.

MICROSERVICES



Deploy and manage a microservices architecture with the option to integrate with Dapr.

AUTO-SCALE CRITERIA

Individual microservices can scale according to any KEDA scale triggers.

Serverless

- Only some frameworks supported
- Have to use specific coding patterns
- **Can't host a website**
- Locked to platform

AWS App Runner / GCP Cloud Run

- **Only scales to HTTP**
- **No scale to zero (AWS)**
- Still quite locked to platform

Azure Container Apps - Features

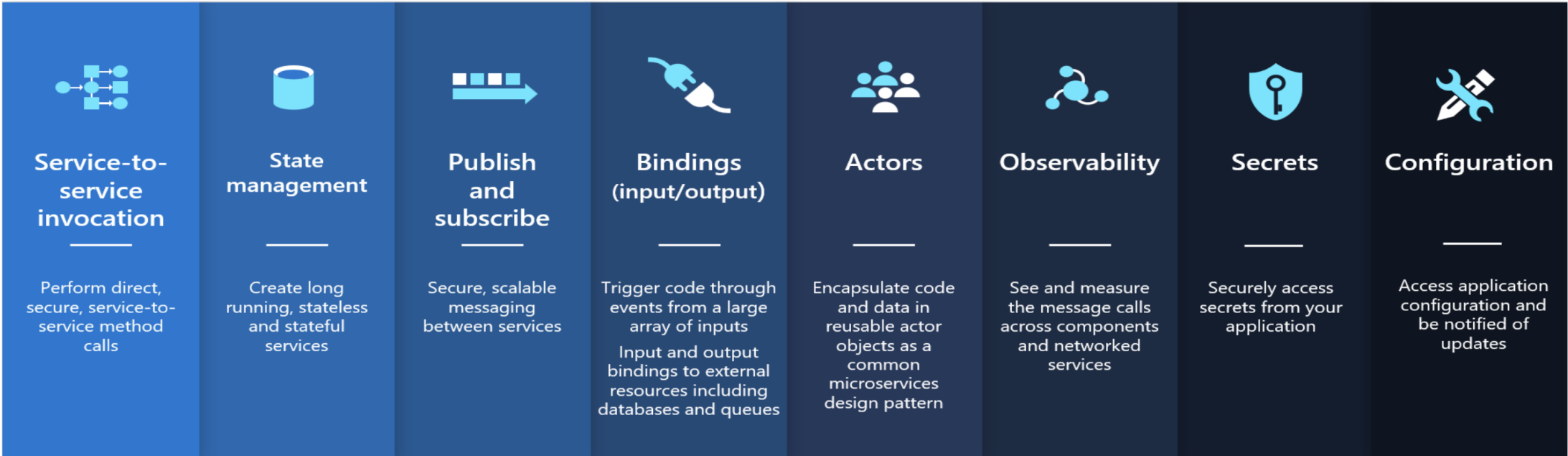
Azure Container Apps은 다음과 같은 기능을 제공합니다.

- [Run multiple container revisions](#)
- [Autoscale](#)
- [Enable HTTPS ingress](#)
- [Split traffic](#)
- [Use internal ingress and service discovery](#)
- [Build microservices with Dapr](#)
- [Run containers from any registry](#)
- [Use the Azure CLI extension or ARM templates](#)
- [Provide an existing virtual network](#)
- [Securely manage secrets](#)
- [View application logs](#)

KEDA가 지원하는 스케일 트리거를 기반으로 앱을 자동 확장. 대부분의 애플리케이션을 0로 스케일 가능 (단, **CPU나 Memory 기반 스케일은 0 불가 --> 1**)

Dapr (Distributed Application Runtime)

Dapr란 빌딩 블록의 집합이자 마이크로 서비스를 빌드하는 API입니다. **사이드카 패턴** 사용.



```
POST http://localhost:3500/v1.0/invoke/cart/method/neworder
GET http://localhost:3500/v1.0/state/inventory/item67
POST http://localhost:3500/v1.0/publish/shipping/orders
GET http://localhost:3500/v1.0/secrets/keyvault/password
```

KEDA (Kubernetes Event-driven Autoscaling)

KEDA는 Kubernetes 기반 Event Driven Autoscaler입니다. KEDA를 사용하면 처리해야 하는 이벤트 수를 기준으로 Kubernetes의 모든 컨테이너의 스케일링을 실행할 수 있습니다.



Event-driven

Intelligently scale your event-driven application



Autoscaling Made Simple

Bring rich scaling to every workload in your [Kubernetes](#) cluster



Built-in Scalers

Out-of-the-box scalers for various vendors, databases, messaging systems, telemetry systems, CI/CD, and more



Multiple Workload Types

Support for variety of workload types such as deployments, jobs & custom resources with `/scale` sub-resource



Vendor-Agnostic

Support for triggers across variety of cloud providers & products



Azure Functions Support

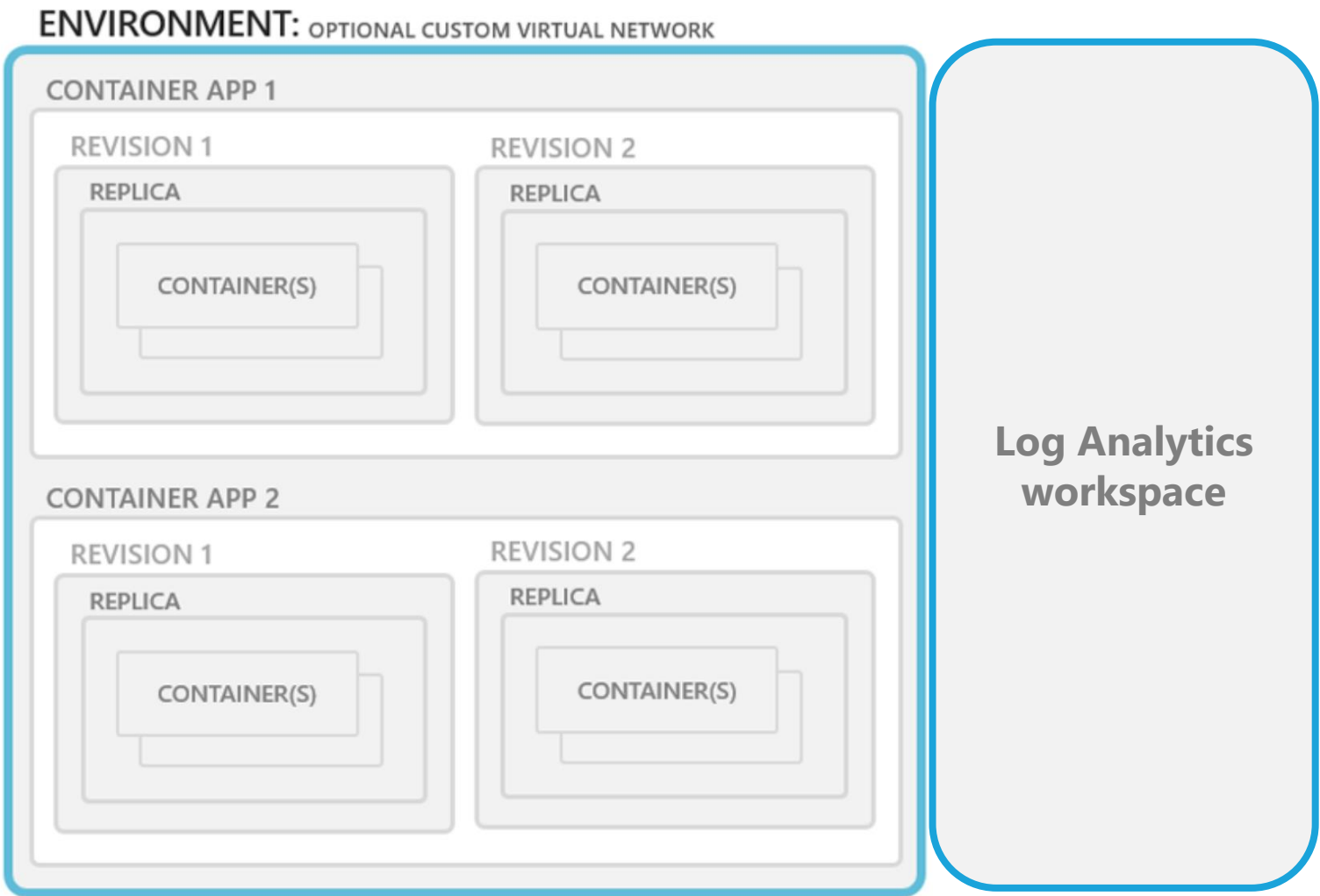
Run and scale your Azure Functions on Kubernetes in production workloads

Azure Container Apps environments

동일한 가상 네트워크에 배포된 컨테이너 앱 컬렉션 주위로 격리 및 관찰 가능성에 대한 경계를 정의합니다.



Environments are an isolation boundary around a collection of container apps.

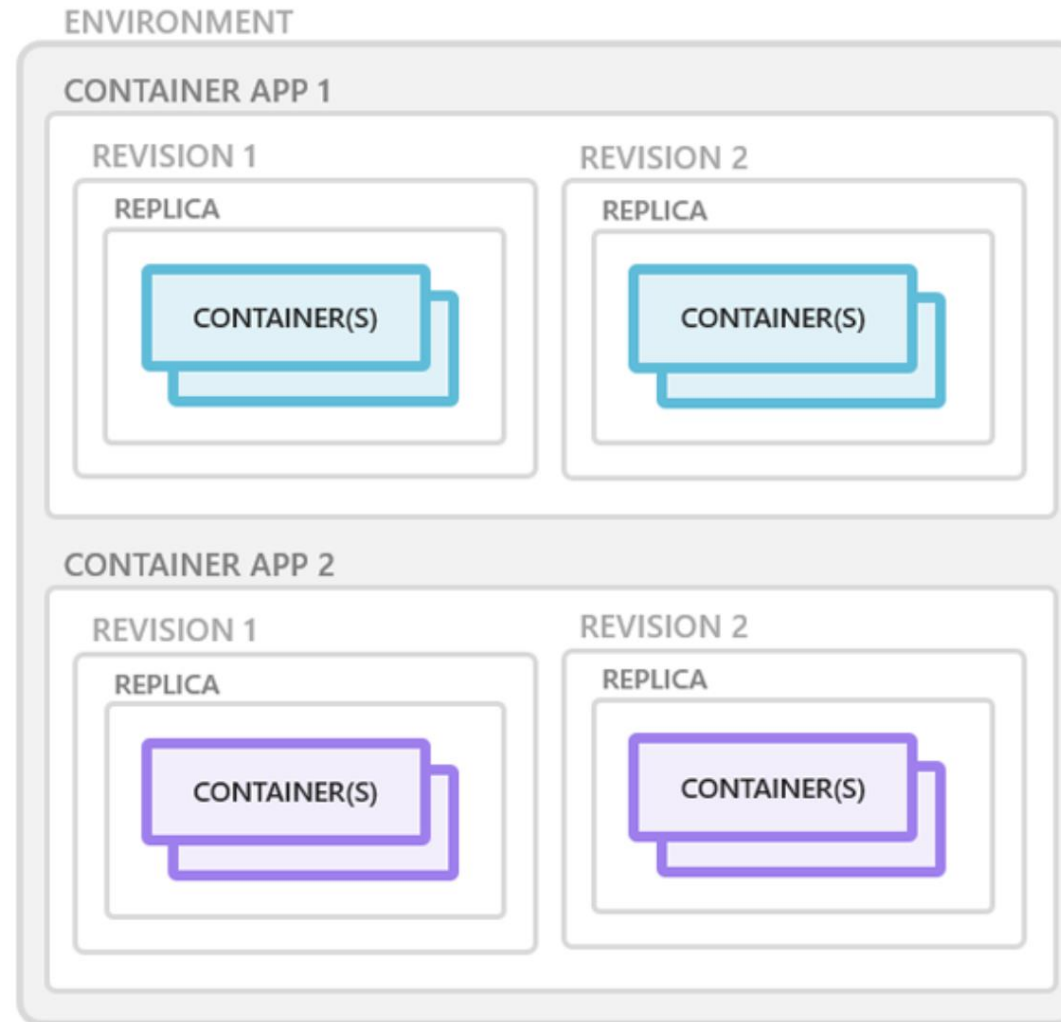


Azure Container Apps - Containers

컨테이너는 revision 스냅 샷 내부의 파드에서 함께 그룹핑되며 원하는 런타임, 프로그래밍 언어 또는 개발 스택을 사용할 수 있습니다.



Containers for an Azure Container App are grouped together in pods inside revision snapshots.



리소스 할당 시 컨테이너 앱의 모든 컨테이너에 대해 요청된 CPU 및 Memory의 총량은 다음 조합 중 하나로 합산되어야 합니다.

vCPUs (cores)	Memory (GiB)
0.25	0.5Gi
0.5	1.0Gi
0.75	1.5Gi
1.0	2.0Gi
1.25	2.5Gi
1.5	3.0Gi
1.75	3.5Gi
2.0	4.0Gi

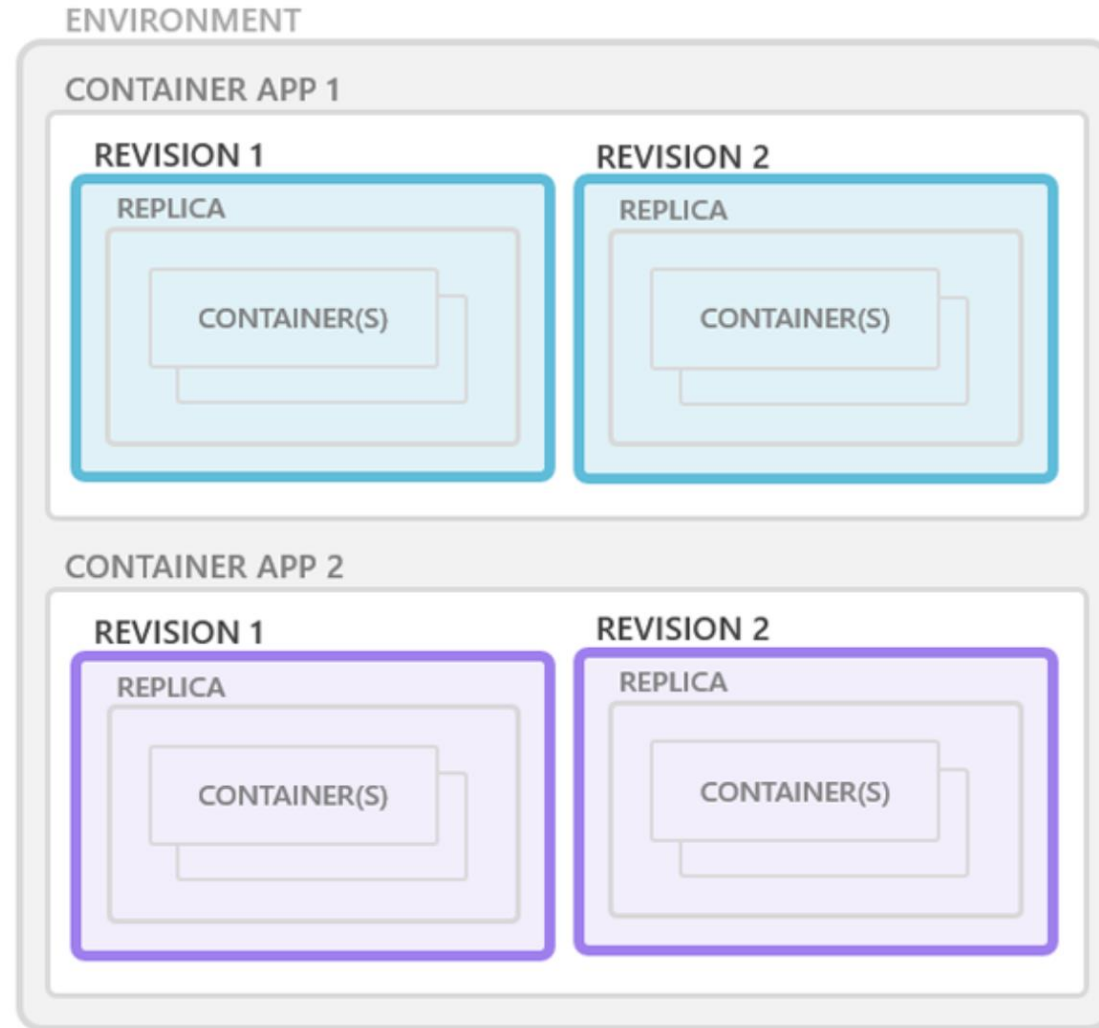
```
"containers": [  
  {  
    "name": "main",  
    "image": "[parameters('container_image')]",  
    "env": [  
      {  
        "name": "HTTP_PORT",  
        "value": "80"  
      },  
      {  
        "name": "SECRET_VAL",  
        "secretRef": "mysecret"  
      }  
    ],  
    "resources": {  
      "cpu": 0.5,  
      "memory": "1Gi"  
    },  
  }  
<이하 생략>
```

Azure Container Apps - Revisions

Revisions는 하나의 컨테이너 앱에 대한 불변의(Immutable) 스냅 샷입니다.

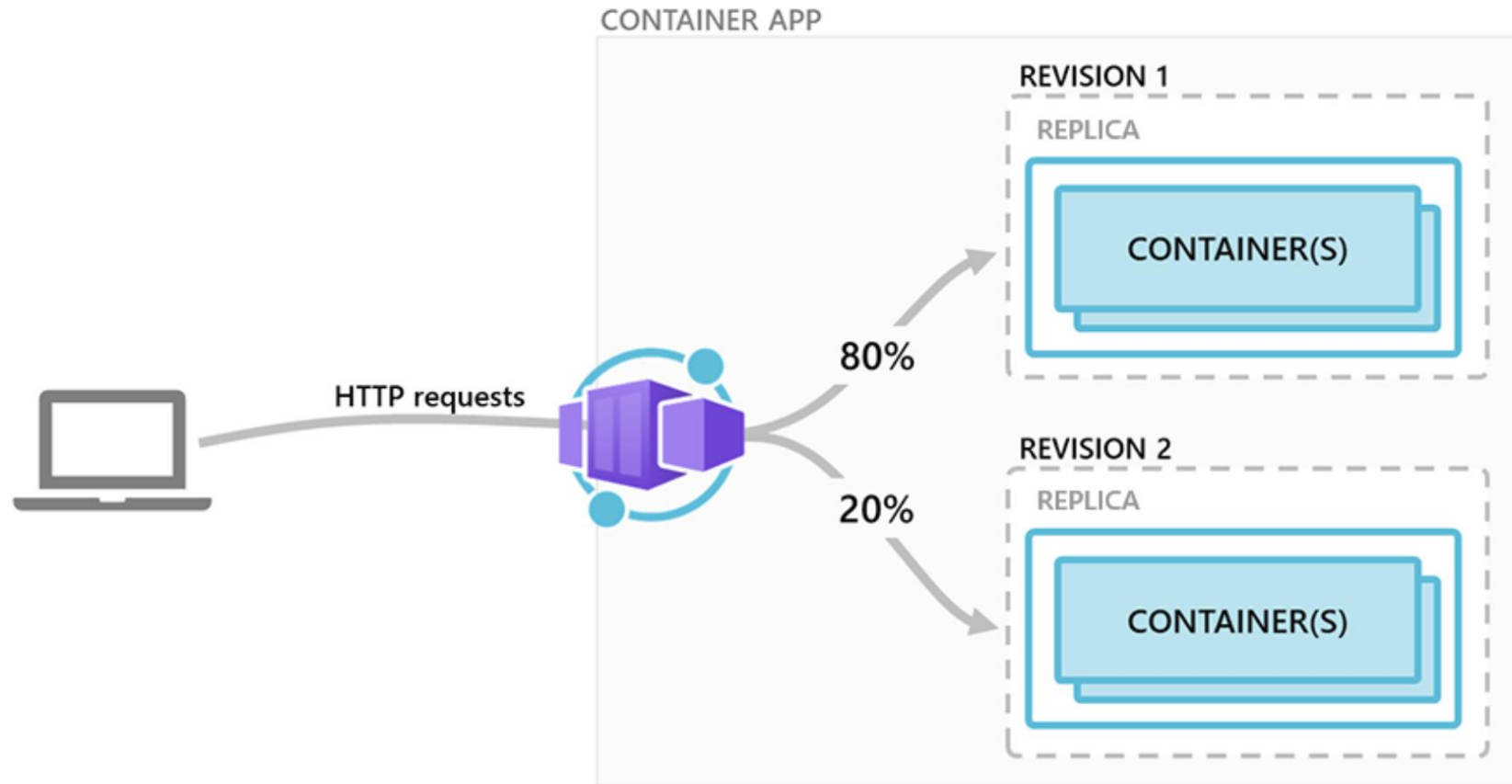


Revisions are immutable snapshots of a container app.



Azure Container Apps – Revisions (계속)

revision-scope 으로 앱을 변경할 때마다 새로운 Revision을 만들어 앱에 대한 업데이트 릴리스 관리를 도우며, 활성 상태 Revision 및 각 활성 Revision으로의 외부 트래픽 라우팅을 제어 할 수 있습니다.



revision-scope changes 또는 **application-scope changes**

두 가지 범주 변경 사항 구분되며 Revision-scope는 앱 배포 시 새로운 Revision 생성을 트리거 하며 Application-scope은 그렇지 않습니다.

single revision mode 또는 ***multiple revision mode***

기본적으로는 single revision mode로 오직 하나의 revision만 활성화되며 새로운 revision 생성시 최신 revision이 활성 revision을 대체

다중 revision을 동시에 실행하여 Revision 관리 페이지에서 각 revision으로의 트래픽 백분율로 제어 가능

Revision 관리 페이지에서 revision에 대한 활성/비활성 제어 가능

Revision Labels

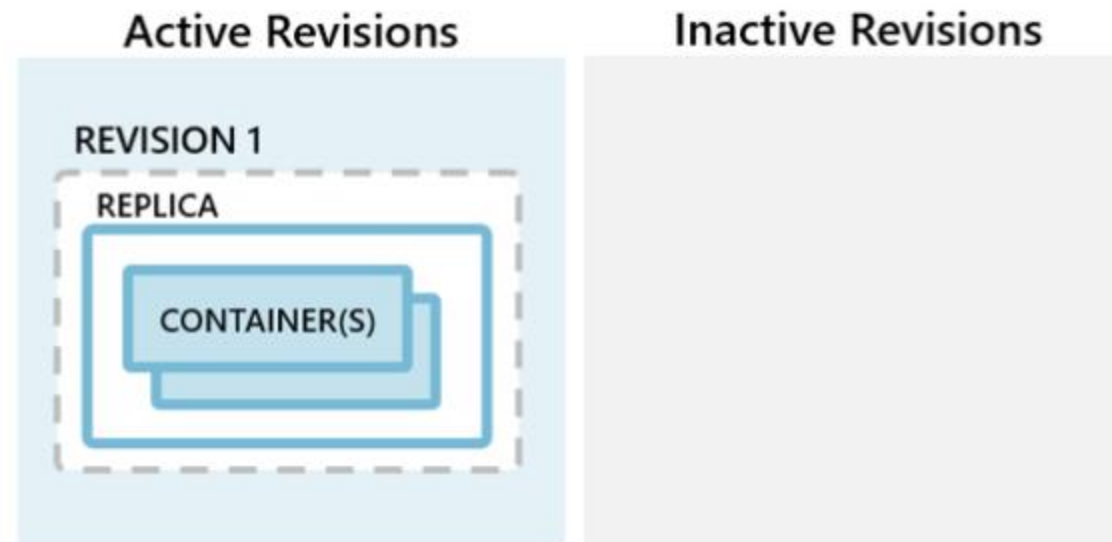
revision 간의 트래픽 제어 수단으로 이용 가능하며 새로운 revision에 대한 테스트용으로 유용

Application Lifecycle

Azure Container Apps 애플리케이션 수명 주기는 revision을 중심으로 진행됩니다.



Upon **deployment**,
the first revision is
automatically
created.



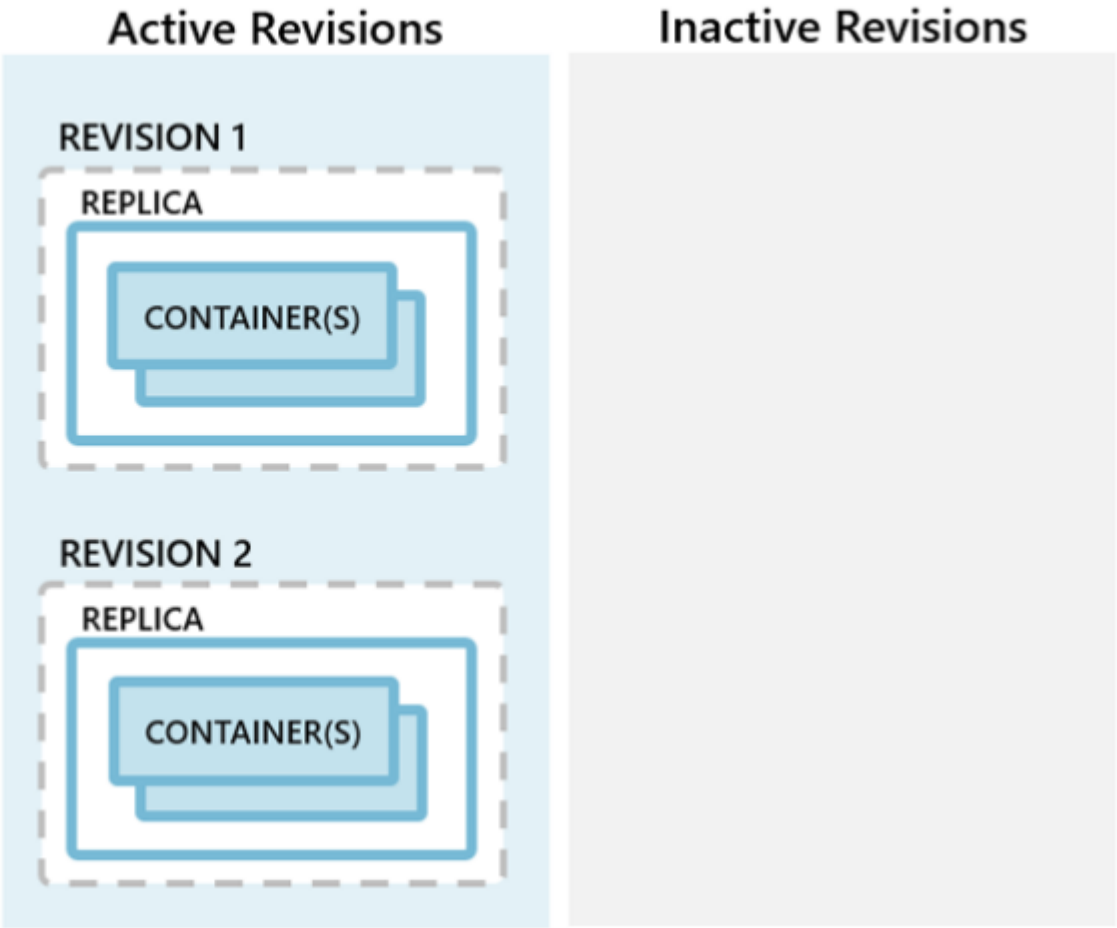
컨테이너 앱이 배포되면 첫 번째 개정이 자동으로 생성됩니다.

Application Lifecycle (계속)

컨테이너 앱이 revision-scope 변경으로 업데이트되면 새 revision 이 생성됩니다. 이전 revision을 자동으로 비활성화할지 또는 계속 사용토록 허용할지 여부 선택 할 수 있습니다.



As the container app is **updated**, a new revision is created.

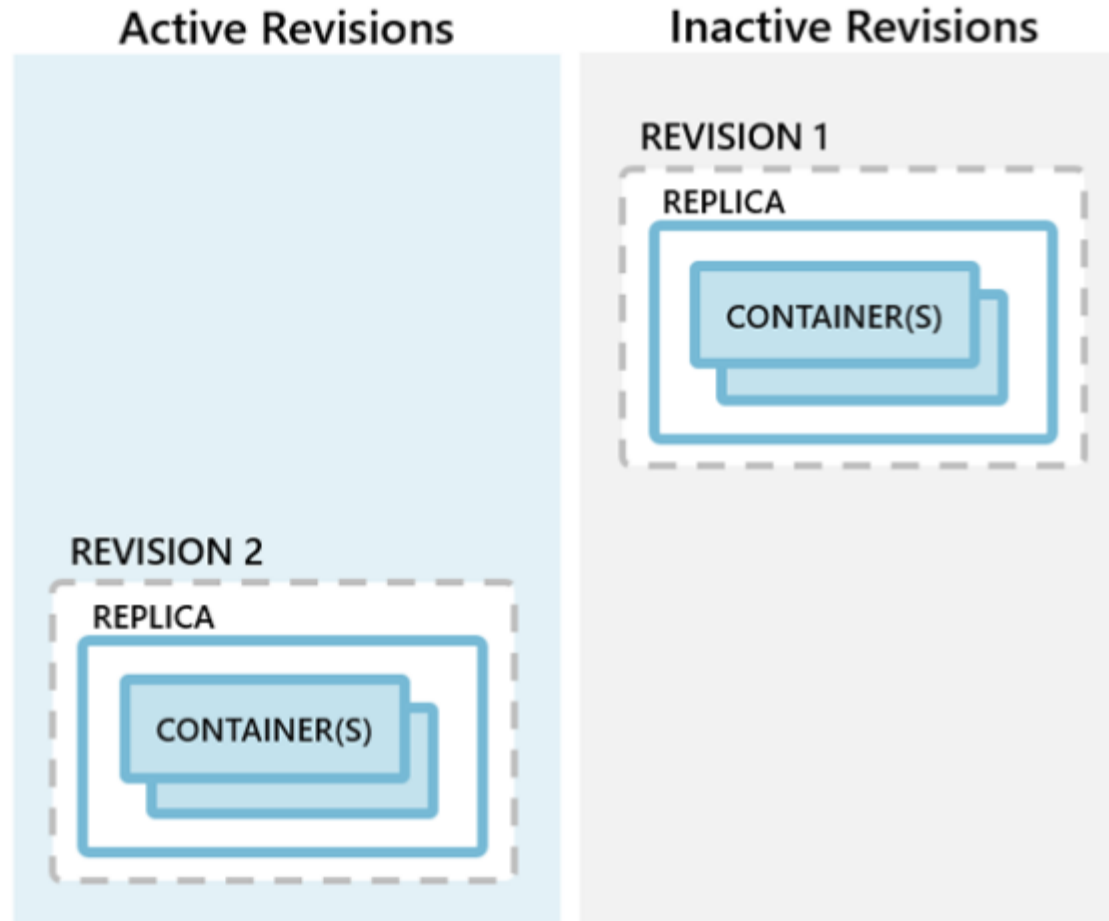


Application Lifecycle (계속)

revision이 더 이상 필요하지 않으면 나중에 다시 활성화하는 옵션을 사용하여 revision을 비활성화할 수 있습니다. 비활성화하는 동안 revision 내 컨테이너가 종료됩니다.



Once a revision is no longer needed, you can **deactivate** individual revisions, or choose to automatically deactivate old revisions.



Application Lifecycle (계속)

컨테이너는 다음 상황에서 종료됩니다.



- 컨테이너 앱이 확장됨에 따라
- 컨테이너 앱이 삭제됨에 따라
- Revision이 비활성화됨에 따라

Azure Container Apps 과거와 현재 그리고 전망...



김영대 ▶ 슬기로운 Azure생활

2021년 11월 8일 · 🌐

[Azure Container Apps 소개]

이번 Ignite 컨퍼런스에서 AKS(Azure Kubernetes Service)를 보완하는 새로운 컨테이너 서비스인 Azure Container Apps의 미리 보기 출시를 발표했습니다. 이 마이크로서비스를 위해 특별히 구축되었다고 언급했습니다.

그래서~~~~ 제가 귀한 월요일 아침 약 30분 정도 시간을 들여 퀵스타트 πππ

No capacity available in the region 'eastus' for provisioning an environment in another region.

머지??? 포기 안하고 이래 저래 시도했지만...

"At this stage of the preview we only support canadace"

이런... 진작 문서에 넣어주시지...πππ

여튼 이거 엄청 기대되는 서비스네요~~~~!!^^

<https://azure.microsoft.com/ko-kr/services/container-apps/>



김영대 ▶ 슬기로운 Azure생활

2021년 11월 23일 · 🌐

와~~~ Azure DevOps CI/CD 파이프라인을 통한 배포
~~~~!!^^ 부산 세미나에서 Demo로 보여 드려야G~

azuredevops\_demo.web Home Privacy

Hello~~~~~ Bus  
Deploying a Dot



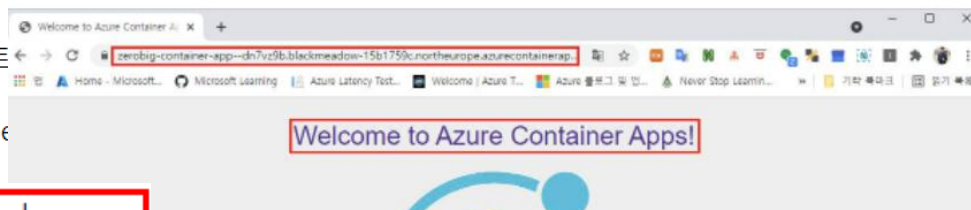
김영대 ▶ 슬기로운 Azure생활

2021년 11월 9일 · 🌐

와~~~ 드디어

첫, Container app 배포~!!^^

여러 버전의 앱을 만들어 각 앱 별로 트래픽 비율 설정, 자동 크기 조정 등 신기하고 잼나는 기능은 차차~~~~^^



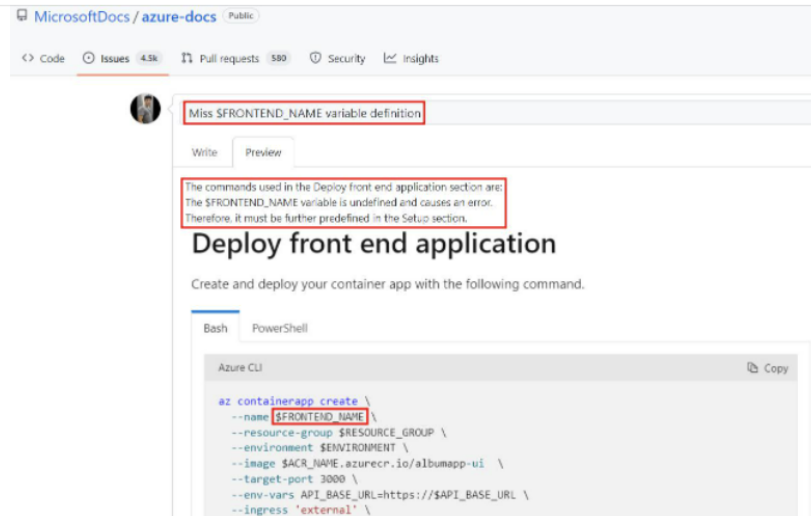
김영대 ▶ 슬기로운 Azure생활

6월 4일 오후 1:16 · 🌐

Azure Container Apps 관련 Contribution 하나 더 추가~^^

Contribution 올리면 피드백이 바로 바로 오네요. 그만큼 MS에서 이 서비스를 중요하게 생각하고 있다는 거겠죠?

Azure 컨테이너 서비스의 떠 오르는 별, Go, go, Azure Container Apps!!!



Canada Central

West Europe

North Europe

East US

East US 2

East Asia

Australia East

Germany West Central

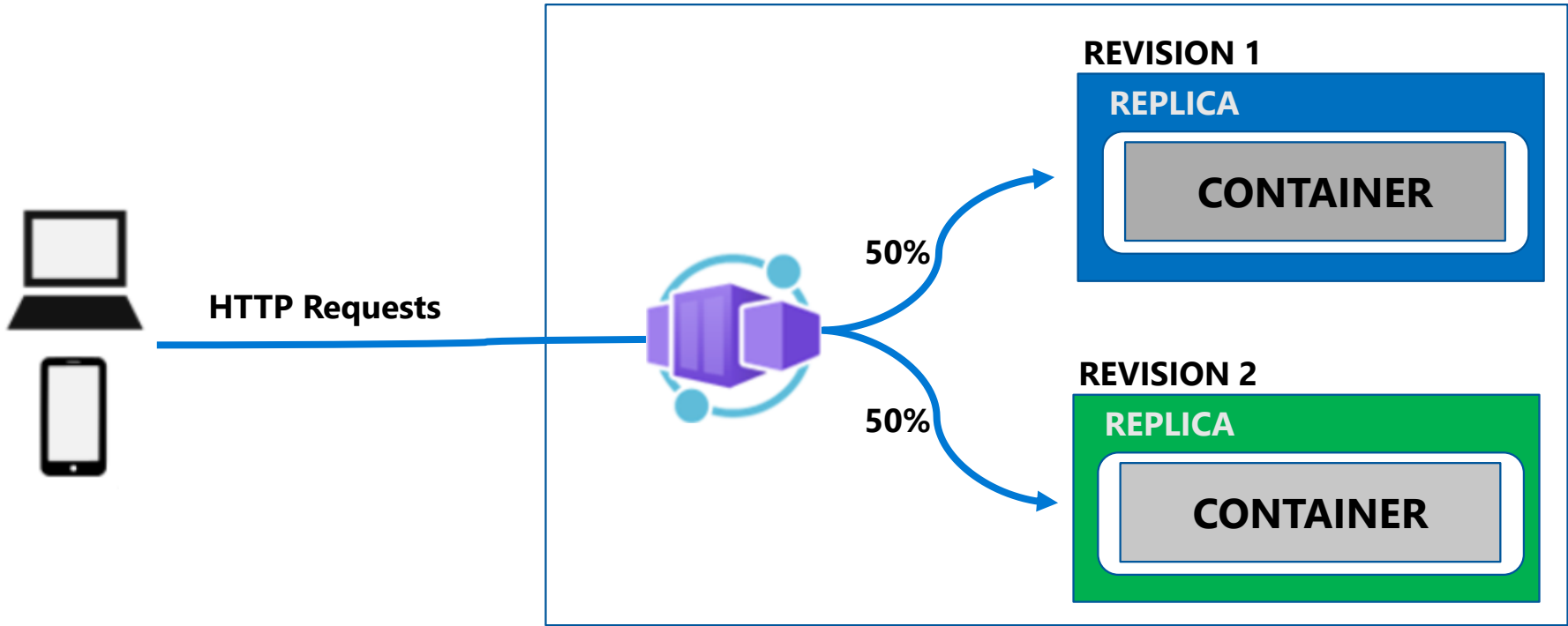
Japan East

UK South

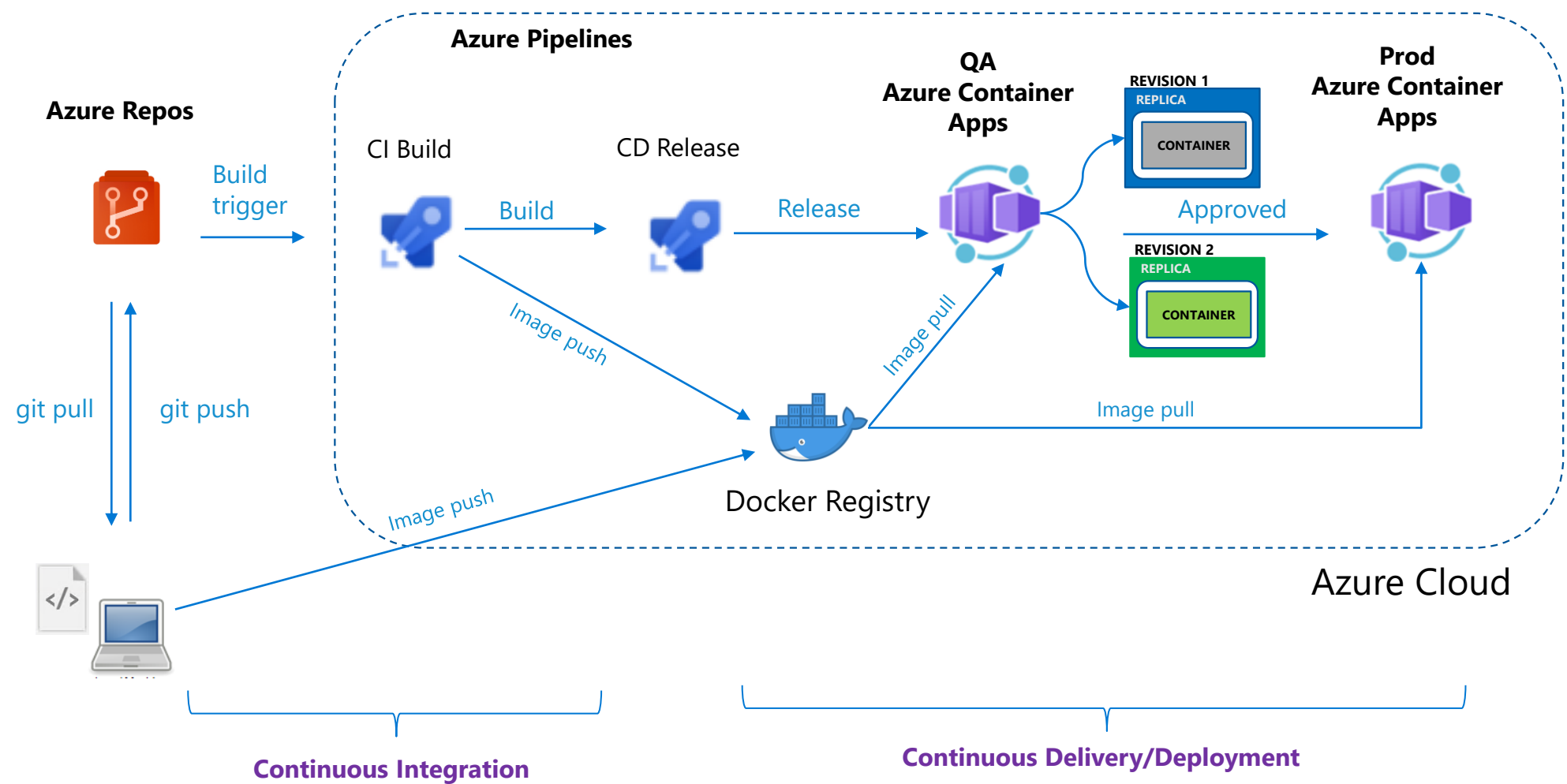
West US

Demo

Blue – Green 배포 시나리오 데모



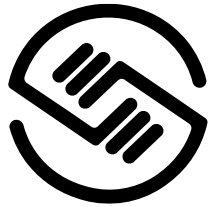
Azure CI/CD 파이프라인을 사용한 애플리케이션 배포 – Azure Container Apps











**CLOUDMATE**  
Managed Service Expert



Microsoft

Build 2022 After Party Korea는 클라우드메이트, 마이크로소프트의 후원을 받아 진행하고 있습니다.

#BuildAfterParty

# 당신의 목소리를 들려주세요!

Build 2022 After Party Korea는 여러분의 목소리를 기다립니다.  
가감 없는 목소리가 발표자 분에게 매우 큰 힘이 됩니다.  
앞으로 더 좋은 행사가 될 수 있도록 목소리를 내주세요.  
감사합니다!

세션에 대한 목소리 (익명):

<https://bit.ly/build2022afterpartykorea-session-survey>

