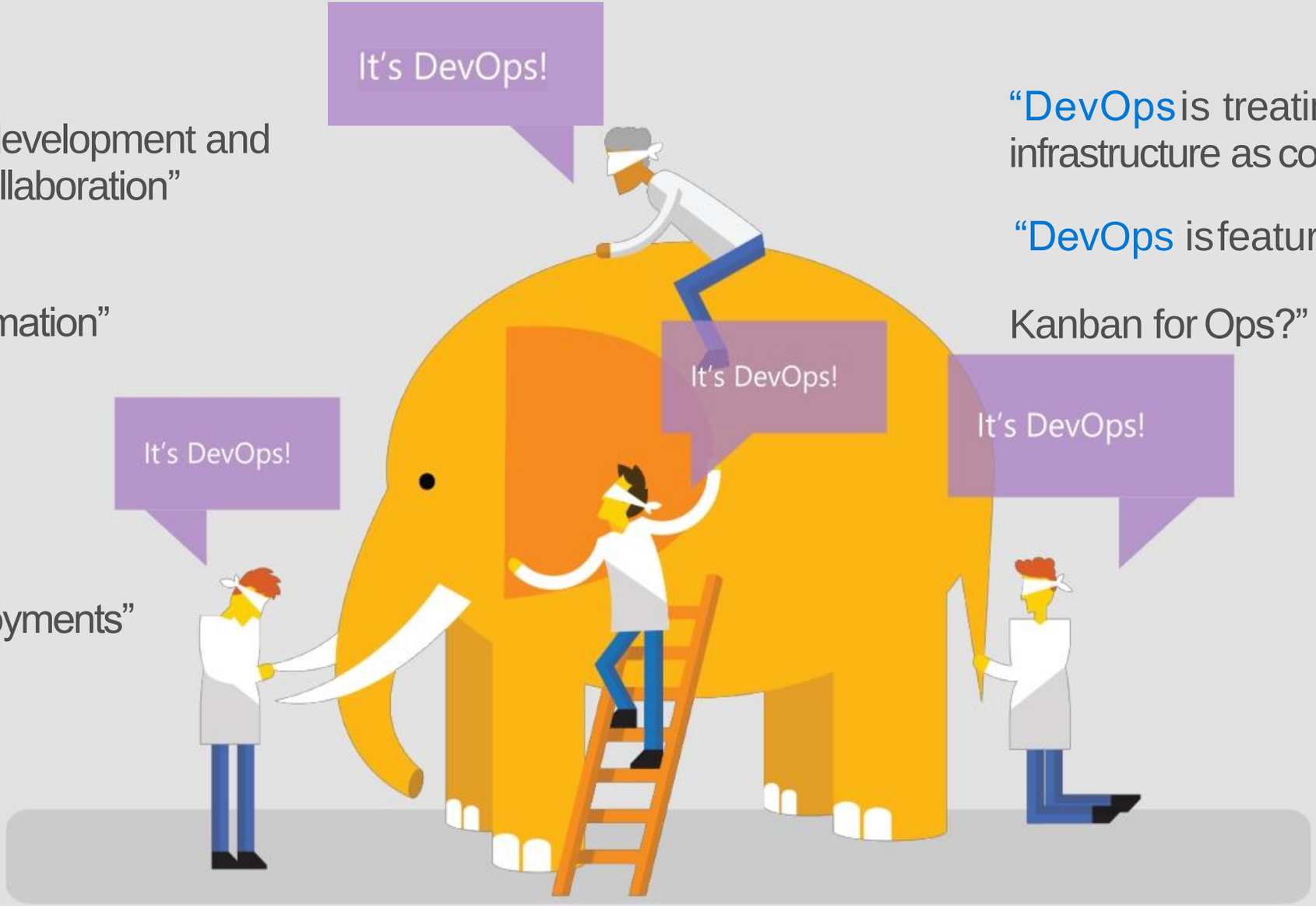
Azure DevOps :
Boards 에서 Pipelines 까지

DevOps는 그동안 너무 이상적이라 실체가 불명확했다

"DevOps is development and operations collaboration"

"DevOps is using automation"

"DevOps is small deployments"



"DevOps is treating your infrastructure as code"

"DevOps is feature switches"

Kanban for Ops?"

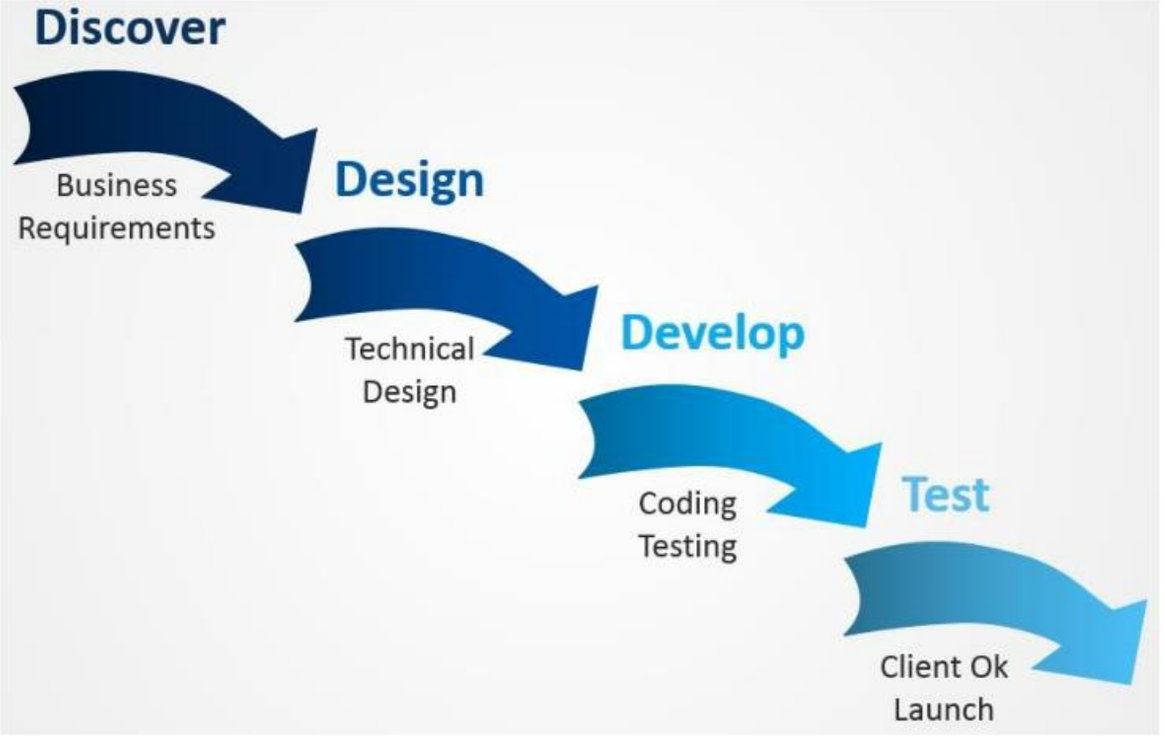
What is CI/CD? DevOps? Pipeline?

- CI/CD 라는 용어는 애자일 소프트웨어 개발 방법론 에서 나옴
- 애자일은 요구사항을 스프린트에 마추어 작게 , 유연하게 개발
- DevOps의 핵심은 개발자와 테스터, 고객이 모든 단계에 참여

- CI / CD는 민첩한 개발을 구현하기 위해 올바른 자동 테스트 도구를 사용하는 DevOps 전략
- CI/CD 를 하기 위해서는 자동화된 도구가 필요 - 이러한 도구들의 묶음을 CI/CD 파이프라인 이라고함
- 즉! 어플리케이션의 통합 및 테스트 단계에서부터 제공 및 배포에 이르는 어플리케이션의 라이프사이클 전체에 걸쳐 자동화와 모니터링을 제공하는 프로세스들의 묶음을 의미
- CI / CD, Agile 및 DevOps 는 같은 목표를 가지고 있음 - **짧은 시간에 더 나은 소프트웨어를 만들기!!**

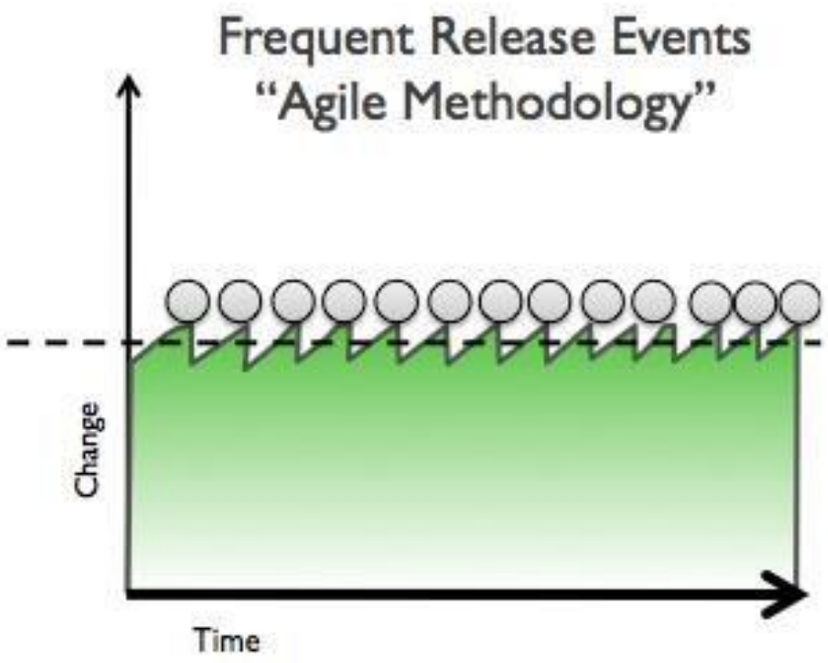
사실 워터폴은 죄가 없다

- 워터폴 모델도 효과가 있다
  - 모든 요구사항이 확실한 경우에
  - 다만, 그런 경우는 극히 드물다

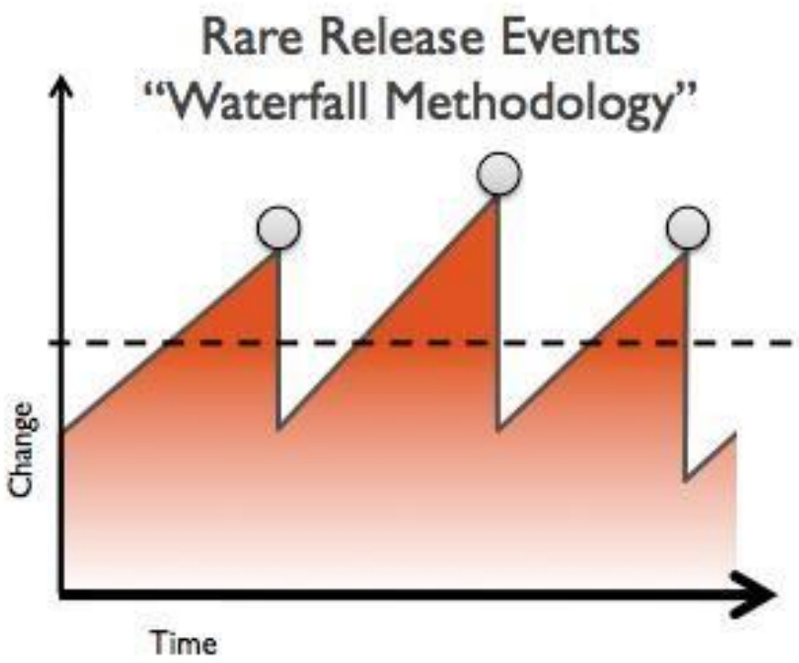


- 추리고 추린 주요 단점
  - 요구사항의 변경이 어렵다
  - 최종 사용자가 참여하지 않는다
  - 모든 것이 완성될 때까지 테스트는 지연된다

애자일의 시작



Smoother Effort
Less Risk



Effort Peaks
High Risk

DevOps의 어플리케이션의 변화

Monolithic 구조에서 Microservice 구조로 변화

•Monolithic: 단일 리소스에 모든 기능을 구현하고 연동하여 간단히 서비스 구성 이 가능하나 단일 형태로 구성이 되어 있고 각 기능이 서로 밀결합 되어 있어 서로 의 종속성이 높아 재개발 혹은 장애에 취약하다.

•마이크로 서비스: 기능별로 리소스가 분산되어 있으며 서로의 종속성이 낮아 업 데이트 및 장애 발생시 전체 서비스에 심각한 영향을 주지 않으나 리소스의 파편화 로 자동화 및 협업툴을 통하지 않으면 운영의 어려움이 발생할 가능성이 높다.

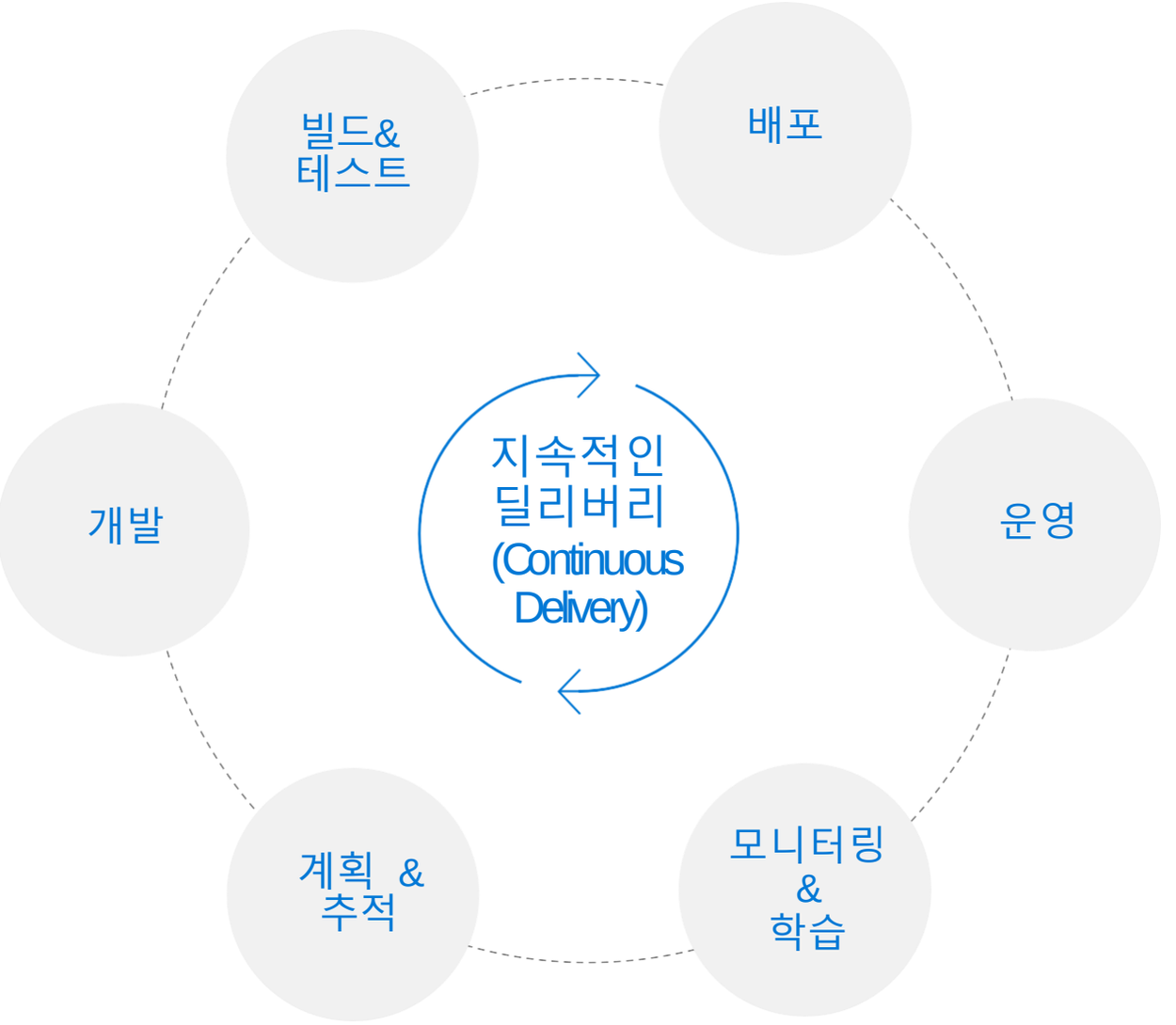
DevOps란 무엇일까요?

사람. 프로세스. 제품.

“

DevOps는 사람, 프로세스, 제품이 함께 결합하여 최종 고객에게 가치를 지속적으로

전달하게 해 줍니다. ”



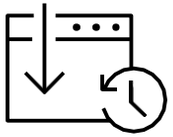
“

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

”

DevOps를 시작하려면

DevOps는 고객에게 지속적인 가치를 제공하기 위해서 소프트웨어의 제공을 자동화 하는 것이며, 그를 위해서 사람, 프로세스 및 도구를 활용하는 것이다. Azure DevOps를 사용하면 IT 부서의 규모나 기존에 사용 중인 도구에 관계없이 소프트웨어를 더욱 빠르고 안정적으로 제공할 수 있다.



지속적인 통합 (CI)

소프트웨어 개발 퀄리티와 속도를 향상시킨다

Azure Pipeline 이나 Jenkins를 사용하여 매번 코드를 커밋할 때마다 클라우드에서 앱을 빌드한다면, 앱은 자동으로 빌드되고 테스트되기에, 버그는 더빠르게 발견된다.

지속적인 딜리버리 (CD)

코드와 인프라가 항상 운영 환경으로 배포가능 상태임을 보장한다

CI(지속적인 통합)과 IaC(코드로 자동화된 인프라)를 결합하면, 언제든지 운영 환경으로 배포할 수 있다는 자신감과 함께 항상 동일한 배포 환경을 얻을수 있다

CI/CD를 통한 지속적인 배포

지속적인 배포(deployment)를 활용하면, CI/CD 가 성공했을 경우 코드 커밋부터 프로덕션까지 전체 프로세스를 자동화할 수 있다

CI/CD를 모니터링 도구와 함께 사용하면, 기능(feature)이 준비되는 즉시 고객에게 안전하게 기능을 제공할 수 있다

CI / CD

Continuous Integration

- 개발자를 위한 자동화 프로세스인 지속적인 통합(Continuous Integration)을 의미
- **개발코드를 통합할 때의 문제점을 해결하고,**
  자동화시켜 지속적으로 유지시키는 방법
- 코드를 커밋만 치면 자동으로 빌드, 통합을 하고, 테스트를 하는 과정을 의미

지속적인 통합(CI)은 개발 팀이 코드의 개발 및 테스트를 단순화하기 위해 사용하는 실행 방안이다. CI는 개발 초기에 버그나 문제를 파악하는 데 도움이 되며, 이를 통해 쉽고 빠르게 문제점을 해결할 수 있다. 자동화된 테스트와 빌드는 CI 프로세스의 일부로 수행된다.
프로세스는 정해진 일정에 따라 수행되거나 혹은 코드가 push될 때마다 실행된다. CI 시스템의 산출물로는 빌드 아티팩트가 생성된다. 아티팩트는 지속적인 딜리버리의 릴리스 파이프라인에서 사용되며, 이를 통해 배포 자동화를 구성할 수 있다.

https://docs.microsoft.com/en-us/azure/devops/pipelines/get-started/key-pipelines- concepts?toc=/azure/devops/pipelines/toc.json&bc=/azure/devops/boards/pipelines/breadcrumb/toc.json&view=vsts

CI / CD

Continuous Delivery/Continuous Deployment

- 지속적인 서비스 제공(Continuous Delivery) / 지속적인 배포(Continuous Deployment)
- 어플리케이션을 항상 신뢰 가능한 수준으로 배포 될수 있도록 지속적으로 관리

- **CI 가 이루어지고 난 후에 운영환경 까지 배포를 수행하여,**
  실제 사용자가 사용할 수 있도록 적용하는 단계

지속적인 딜리버리(CD)는 코드를 하나 이상의 테스트 및 운영 환경에 빌드, 테스트 및 배포하는 프로세스다. 여러 단계로 배포 및 테스트를 실시하면 품질 향상시킬 수 있다. CI 시스템은 인프라와 애플리케이션을 포함하는, 배포 가능한 아티팩트를 생성한다. 자동화된 릴리즈 파이프라인은 이러한 아티팩트를 사용하여 기존 시스템에 새로운 버전과 수정 사항을 릴리즈한다. 모니터링 및 경고 시스템은 지속적으로 실행되어 전체 CD 프로세스에 대한 가시성을 확보한다. 이러한 시스템을 활용하면 에러를 빠르게, 자주 잡을 수 있다.

https://docs.microsoft.com/en-us/azure/devops/pipelines/get-started/key-pipelines- concepts?toc=/azure/devops/pipelines/toc.json&bc=/azure/devops/boards/pipelines/breadcrumb/toc.json&view=vsts

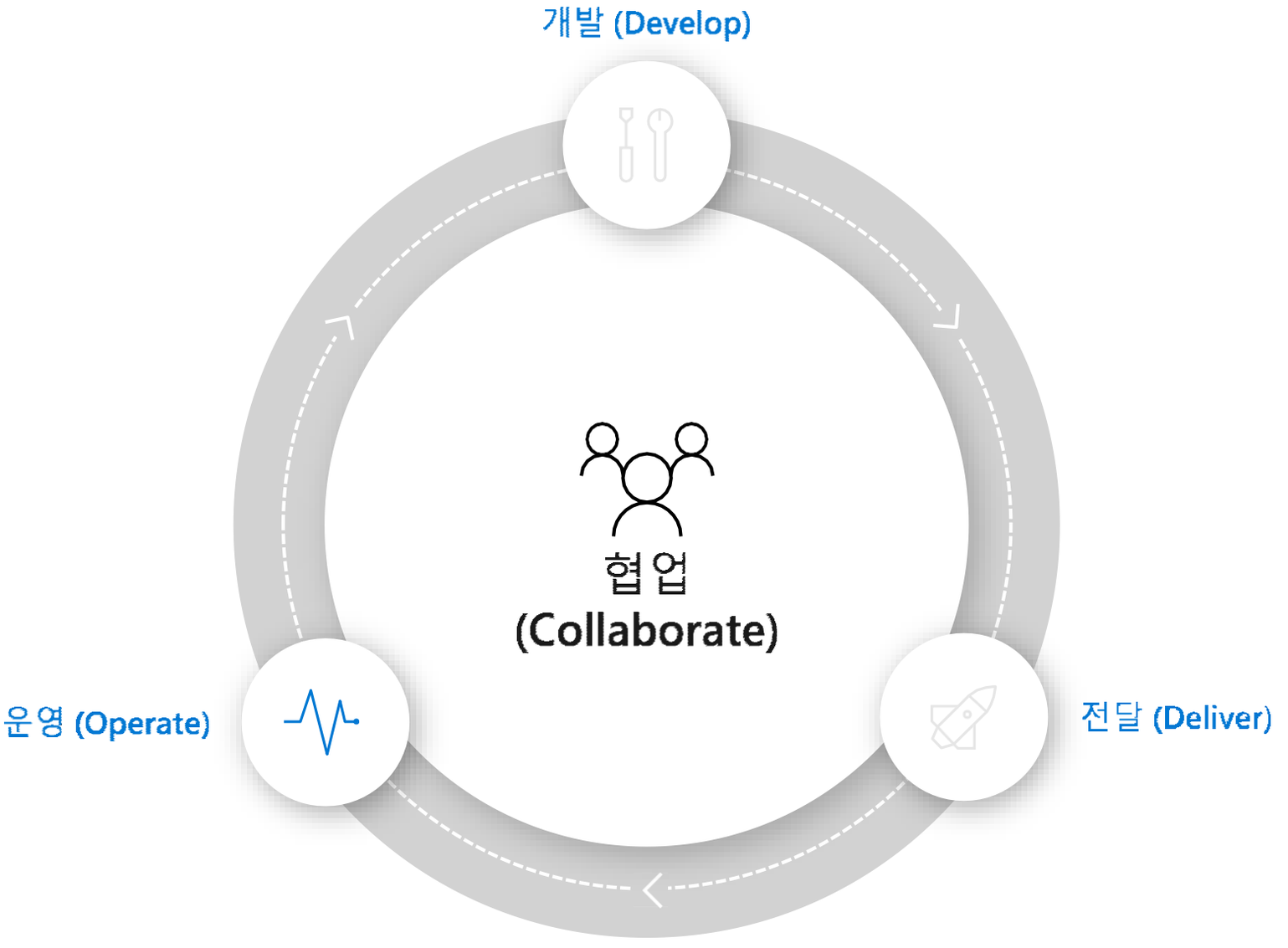
DevOps로 가치를 가속화합니다

목표 및 도구 공유

협업

프로세스 자동화

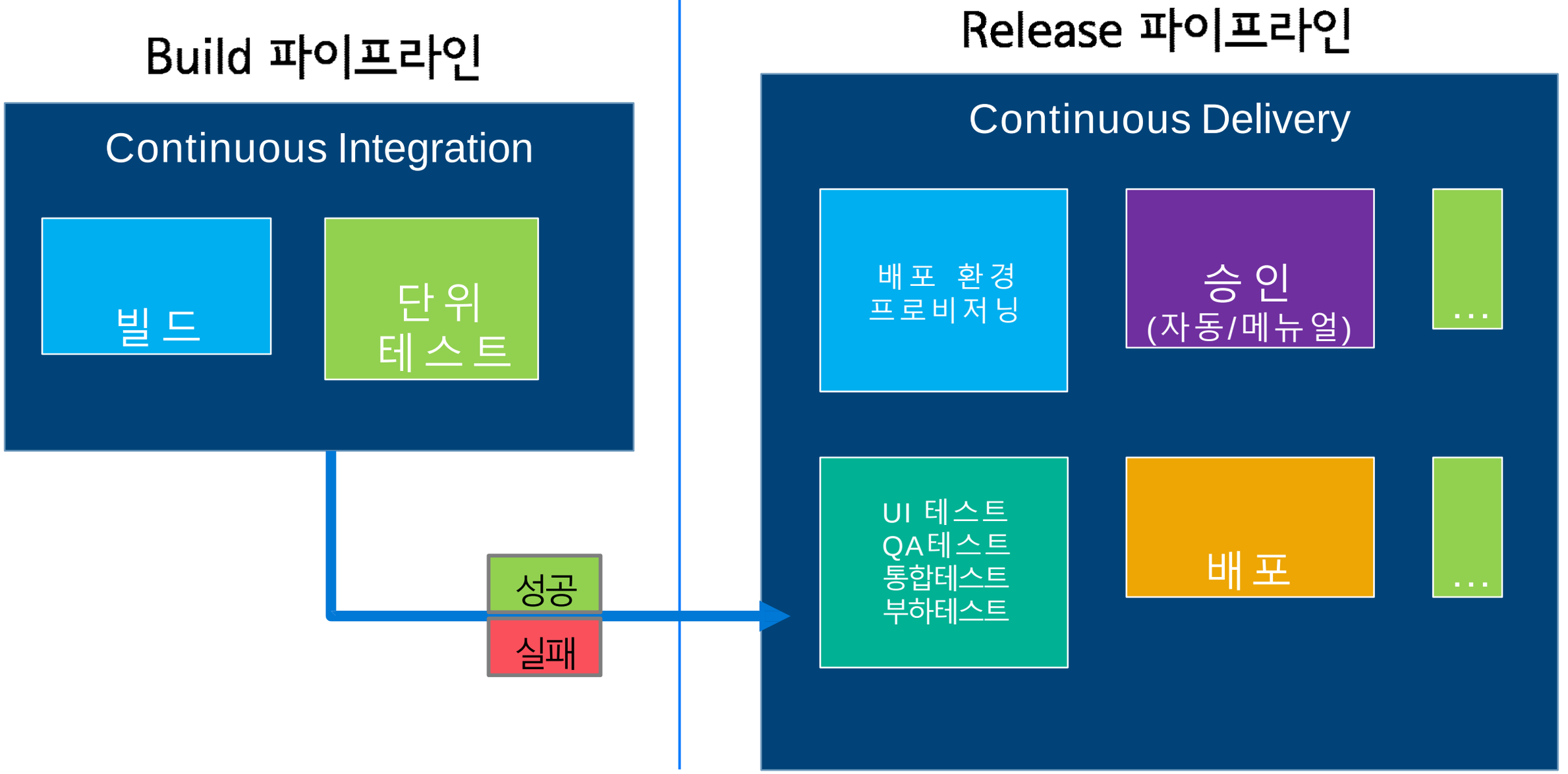
지속적인 딜리버리 및 개선



DevOps 관련 용어정리

1. Code : 소스코드를 개발하고 컴파일 및 코드를 배포 가능한 파일 (Artifact) 구성상태로 만드는 단계
2. 컴파일 (Compile) : 소스코드를 분석해 기계어(Binary Code) 로 번역
3. Build : Binary Code 와 Asset(Library, HTML, Image 등) 등을 모아 실행 가능한 상태로 만드는것
4. Artifact (Binary) : 배포가능한 형태의 컴파일 또는 빌드된 소스코드 (Compiled Source (WAR, JAR), TAR & ZIP, No Compile Source)
5. 저장소 (Repository) : 아티팩트(패키지) 들을 배포하기 위한 저장소
6. 테스트 : 개발된 코드 기능을 테스트 또는 전체 서비스의 기능을 테스트하는 단계
    - Unit 테스트: 각 기능별로 동작 유무를 확인할수 있는 custom script
    - Integration 테스트 : unit 테스트의 묶음 혹은 전체 서비스 동작 유무를 확인할수 있는 custom script
    - Load 테스트: 네트워크 레벨의 부하를 일으켜 평균 응답시간 혹은 TPS(Transaction Per Second) 측정
7. Deploy & Release : 배포가능한 파일(artifact) 을 통해서 소스코드를 배포 및 인프라 리소스에 대한 형상을 생성(변경, 삭제) 하여 서비스를 투입(Release) 한느 단계
8. 모니터 : 인프라, 애플리케이션의 시스템과 서비스 레벨의 지표와 로그를 모니터링하여 개성 사항을 도출하는 단계

CI 다시 이해하기


Build 파이프라인
Continuous Integration
빌드
단위
테스트
성공
실패
Release 파이프라인
Continuous Delivery
배포 환경
프로비저닝
승인
(자동/메뉴얼)
...
UI 테스트
QA테스트
통합테스트
부하테스트
배포
...

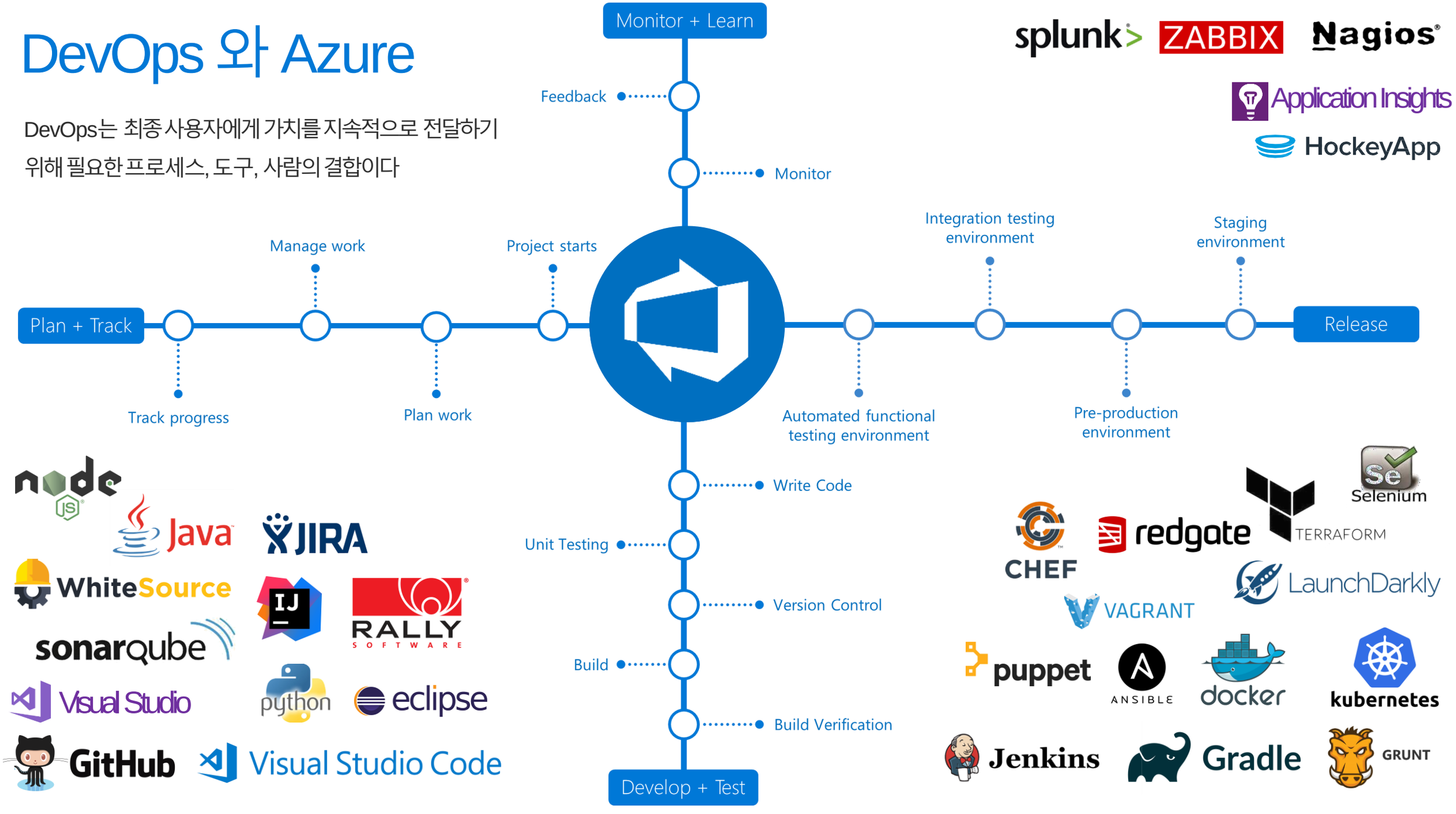
Azure DevOps

https://azure.com/devops

Azure DevOps

DevOps 와 Azure

DevOps는 최종 사용자에게 가치를 지속적으로 전달하기 위해 필요한 프로세스, 도구, 사람의 결합이다



splunk> ZABBIX Nagios

Application Insights

HockeyApp

node JS

Java

JIRA

WhiteSource

IJ

RALLY SOFTWARE

sonarqube

Visual Studio

python

eclipse

GitHub

Visual Studio Code

Se Selenium

redgate

TERRAFORM

CHEF

LaunchDarkly

VAGRANT

puppet

ANSIBLE

docker

kubernetes

Jenkins

Gradle

GRUNT

Azure DevOps

https://azure.com/devops

Azure Boards

팀(team) 간에 작업을 계획, 추적 및 논의하여, 최종 사용자에게 보다 신속하게 가치를 제공한다

Azure Repos

클라우드에서 호스팅되는 무제한 private Git Repo. 풀 리퀘스트 협업, 진보된 파일 관리 등 제공

Azure Pipelines

모든 언어, 플랫폼, 클라우드를 위한 CI/CD 지원. GitHub 등의 Git 공급자에 연결하여 원하는 클라우드로 지속적인 배포 실행

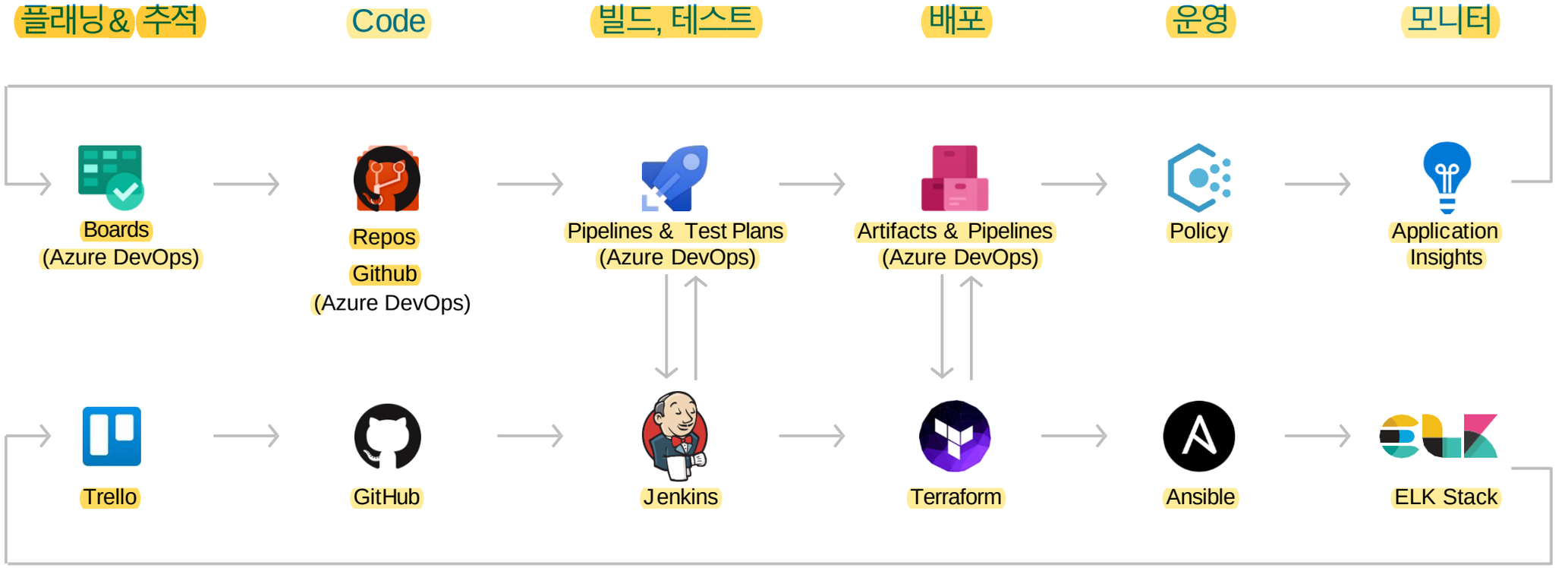
Azure Test Plans

제품을 신뢰성 있게 출시할 수 있도록 돕는 테스트 관리 도구 및 탐색적 테스트 도구 집합

Azure Artifacts

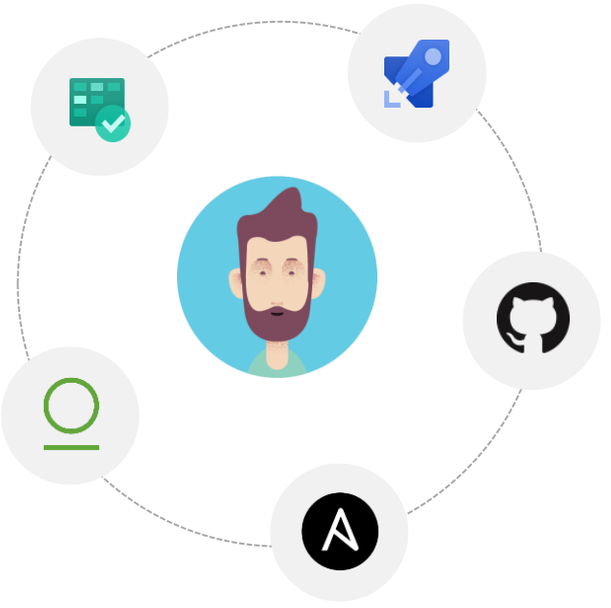
패키지를 생성, 호스트, 공유한다. CI/CD 파이프라인에 아티팩트를 손쉽게 추가할 수 있다

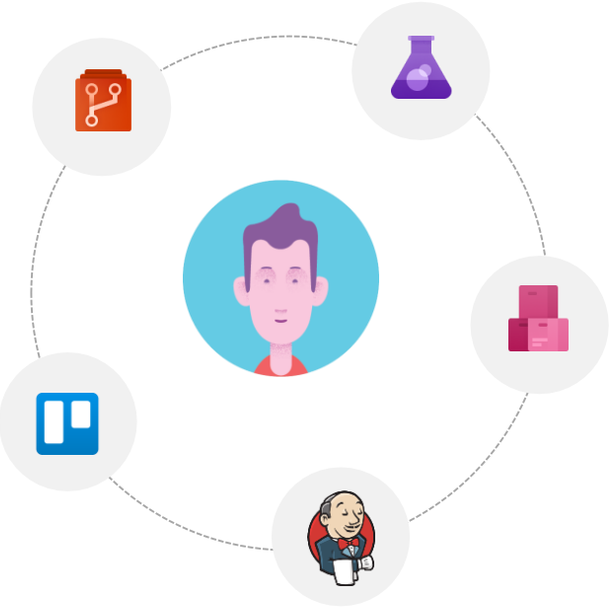
Azure 프레임워크 상에서의 DevOps



Azure DevOps: 좋아하는 클라우드와 도구를 선택

Azure DevOps에서는 개발자들이 자신들이 좋아하는 도구를 선택할 수 있다





Microsoft, 오픈 소스 혹은 좋아하는 써드파티 도구를 적절히 혼합하여 워크플로우를 생성

AWS로도, 쿠버네티스로도 배포할 수 있어요!!!

특정 클라우드, 온프레미스, 혹은 둘 모두에서 제공하는 서버들에 배포

Azure Pipelines

클라우드에서 호스팅되는 파이프라인. Linux, Windows, macOS 지원. 오픈소스의 경우 시간제약 없이 지원

모든 언어, 모든 플랫폼, 모든 클라우드

Node.js, Python, Java, PHP, Ruby, C/C++, .NET, Android, iOS 앱을 빌드, 테스트, 배포. Linux, macOS, Windows 상에서 병렬로 실행. Azure, AWS, GCP 심지어 온프레미스로도 배포 가능

확장 가능

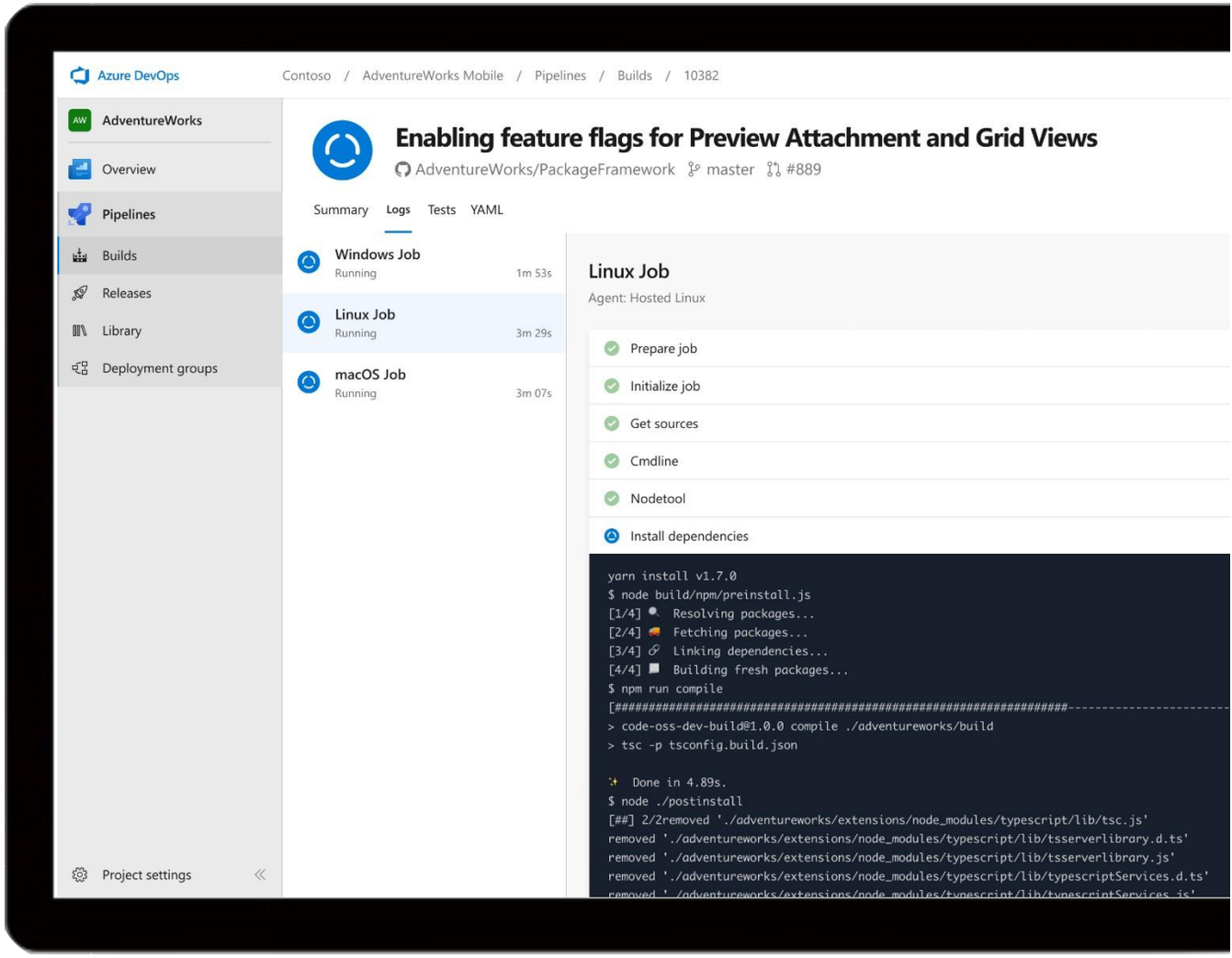
Slack에서 SonarCloud에 이르기까지 수백 개의 확장 기능과 함께 다양한 커뮤니티 기반의 빌드, 테스트 및 배포 태스크를 사용할 수 있다. YAML, 리포트 등도 지원

컨테이너와 쿠버네티스 지원

Docker Hub나 Azure Container Registry와 같은 컨테이너 레지스트리에 이미지를 빌드하고 푸시할 수 있다. 개별 호스트 또는 Kubernetes에 컨테이너 배포가능

오픈소스에 최적

모든 오픈 소스 프로젝트에서 신속한 CI/CD 파이프라인을 구성할 수 있다. 오픈 소스 프로젝트의 경우, Linux, MacOS 및 Windows에서 최대 10개의 무료 병렬 Job을 제공하며 무제한의 빌드 시간을 지원한다



https://azure.com/pipelines

Build/Release 파이프라인

Build your app

.NET Core


Anaconda


Android

ASP.NET


C/C++ with GCC


C/C++ with VC++

Docker


Go


Java


JavaScript and Node.js

PHP


Python


Ruby


UWP


Xamarin


Xcode

https://docs.microsoft.com/en-us/azure/devops/pipelines/?view=azure-devops#pivot=index&panel=indexA

Build/Release 파이프라인

Deploy your app


Azure Kubernetes Service


Azure Stack

Azure SQL database


Azure Web Apps


Linux VM

npm


NuGet


Virtual Machine Manager


VMware


Web App for Containers


Windows VM






https://docs.microsoft.com/en-us/azure/devops/pipelines/?view=azure-devops#pivot=index&panel=indexA

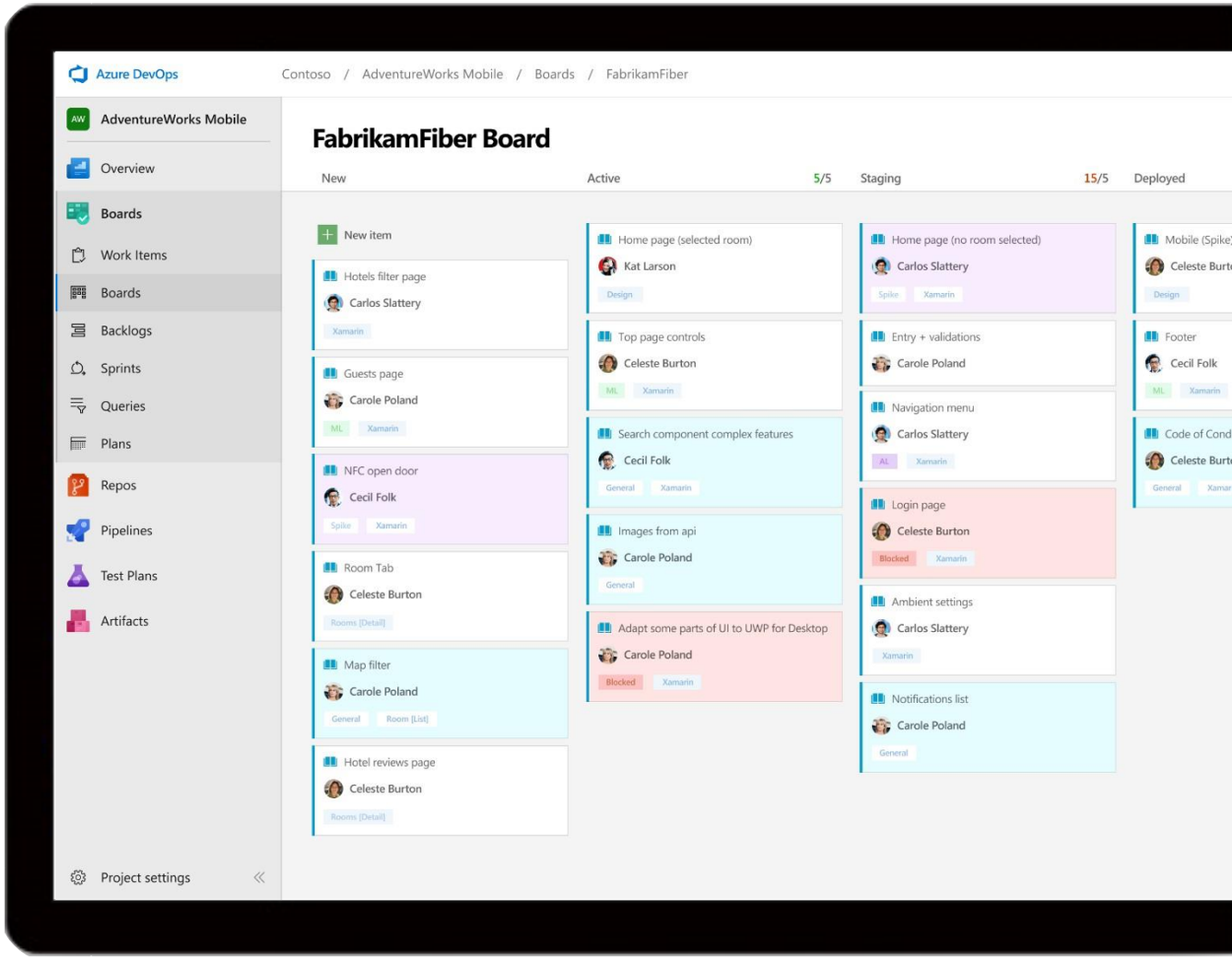
Azure Boards

Kanban 보드, 백로그, 팀 대시보드 및 사용자 정의 보고서를 통해서 업무를 추적한다

아이디어에서 릴리스로 연결
모든 개발 단계에서 모든 아이디어를 추적하고
팀원들이 모든 코드 변경을 작업 항목에
직접 연결하도록 관리한다

Scrum 지원
내장된 스크럼 보드 및 계획 도구를
사용하여 팀이 스프린트, 스탠드업 및 플래닝
회의를 실행할 수 있도록 지원

프로젝트 통찰력
강력한 분석 도구 및 대시보드 위젯을
사용해서 프로젝트의 상태에 대한 새로운
통찰력을 얻을 수 있다



https://azure.com/devops

Azure Repos

무제한 Private Git repo 호스팅 및 TFVC 지원,
취미 프로젝트부터 세계 최대의 Git 리포로 확장가능

여러분의 Git 클라이언트 연계

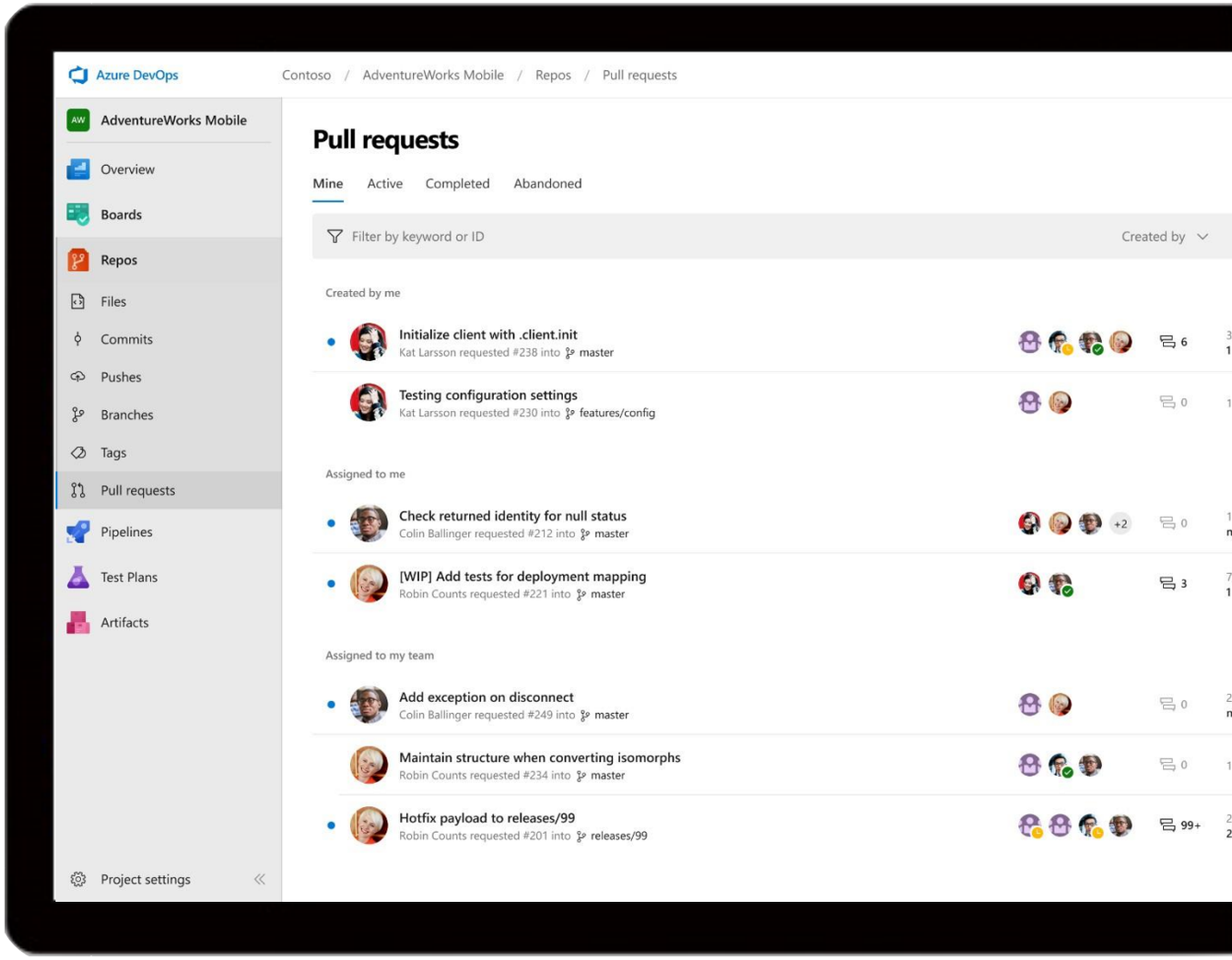
여러분의IDE, 편집기또는Git 클라이언트에서 Git repo로 안전하게 연결하고 코드를 푸시

Web Hook 및 API 통합

마켓플레이스에서 확장을 추가하거나 웹훅이나 REST API를 사용하여 직접구축 가능

시멘틱 코드 검색

클래스와 변수를 이해하는 코드 인식 검색을 사용하여 원하는 것을빠르게 찾을 수 있다



https://azure.com/devops

Azure Test Plans

전체적인(End-to-End) 추적성 확보, 브라우저에서 테스트를 실행하고 결함을 기록, 테스트 수명주기 전반에 걸쳐 품질을 주적하고 평가

풍부한 데이터 캡쳐

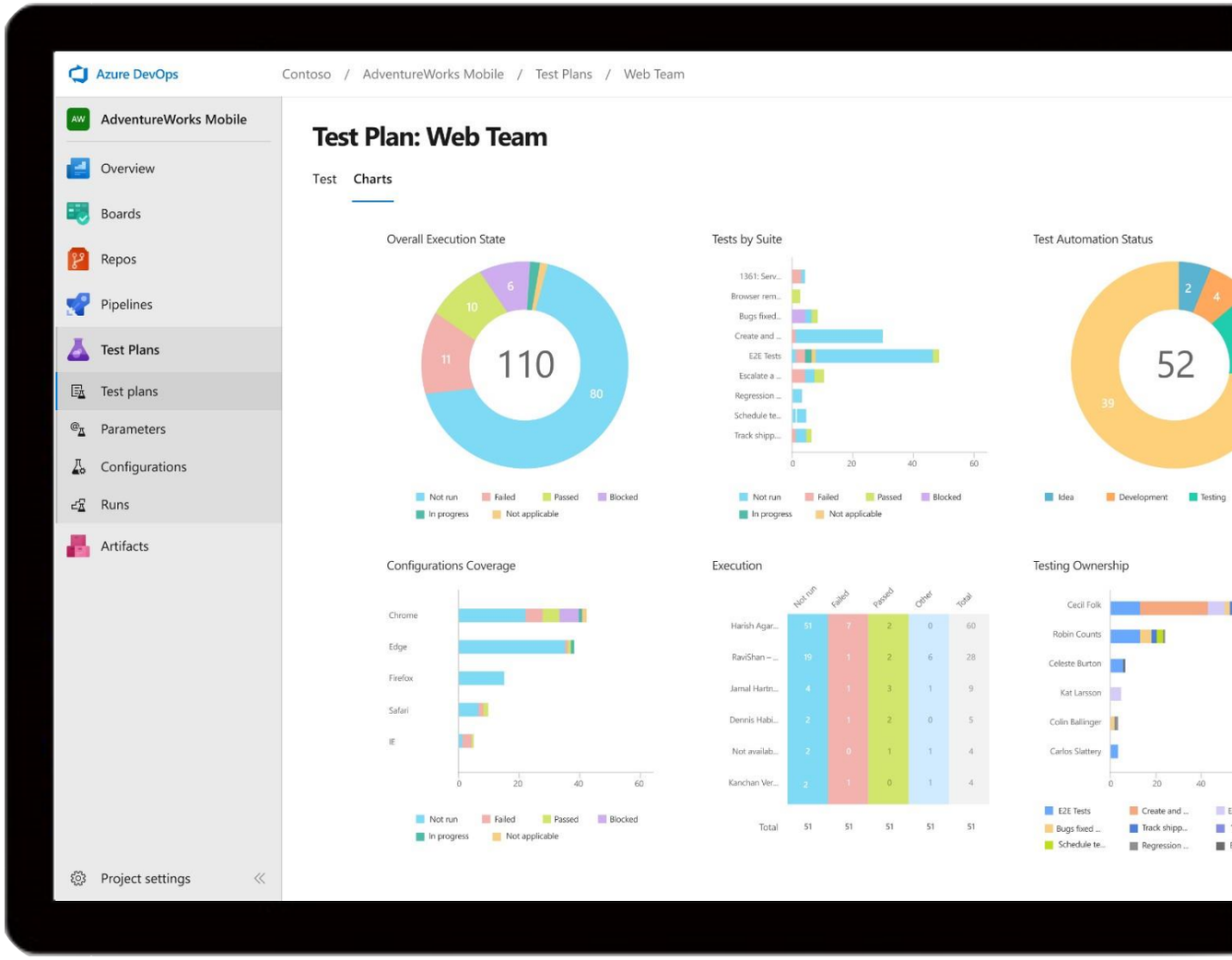
발견된 결점을 조치할 수 있도록 테스트를 실행하면서 풍부한 시나리오 데이터를 캡처. 테스트 사례나 테스트 단계 없이 사용자 스토리를 테스트. 탐색적 테스트 세션에서 직접 테스트 사례를 생성할 수 있다.

웹 및 데스크톱에서 테스트

실행 중인 응용 프로그램을 테스트. 데스크톱 또는 웹 시나리오 전반에 걸쳐 스크립트로 작성된 테스트 수행. 클라우드에서 온프레미스 애플리케이션을 테스트하거나 혹은 그 반대로 테스트 수행 가능

전체적인 추적성 확보

여러분의 엔지니어와 사용자 승인 테스트 관계자가 동일한 테스트 툴을 활용. 도구가 필요한 경우에만 비용을 지불



https://azure.com/devops

Azure Artifacts

공용 및 프라이빗 소스로부터 Maven, npm, NuGet 패키지 피드를 생성하고 공유, CI/CD 파이프라인에 완벽하게 통합

모든 패키지 유형을 관리

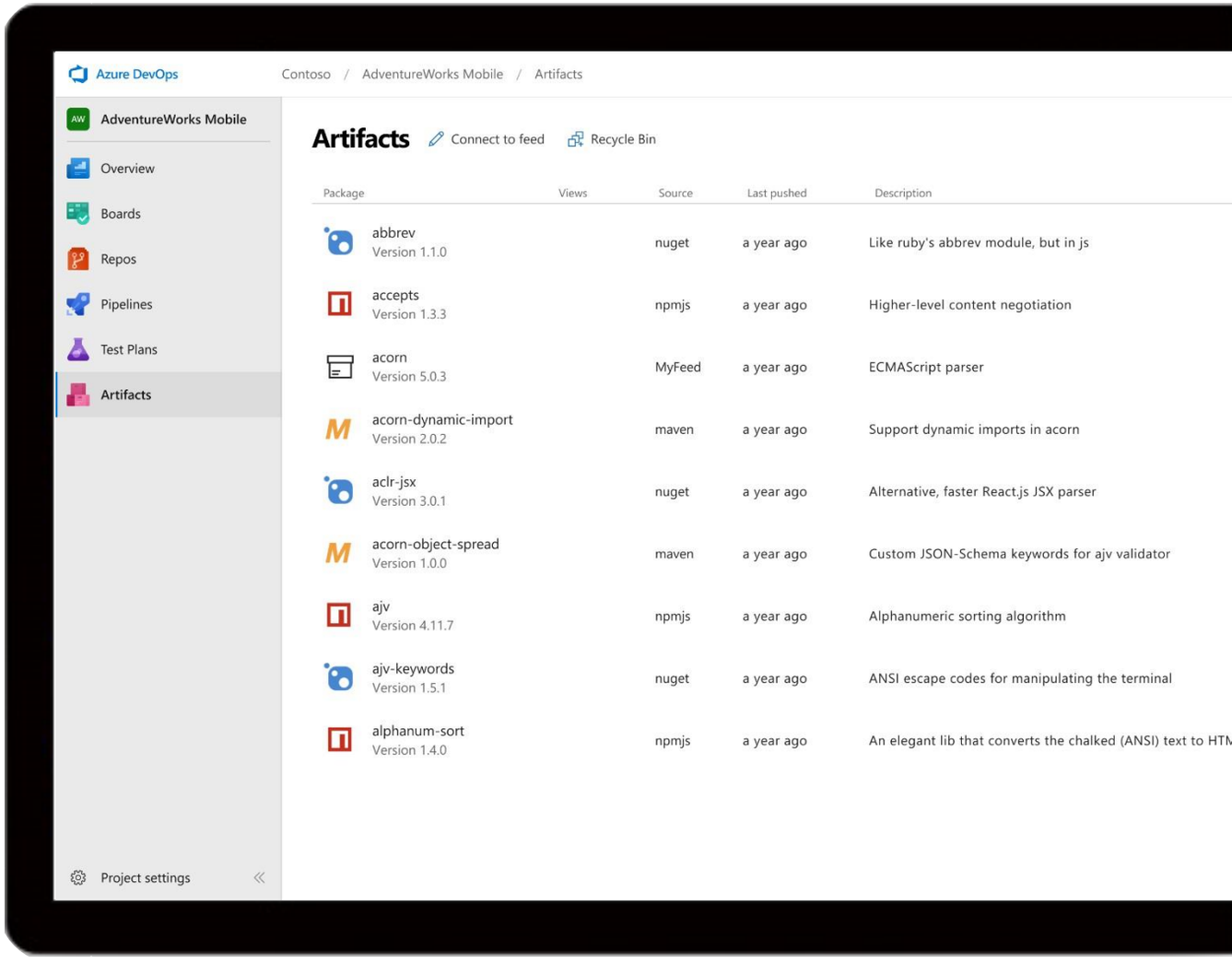
Maven, npm 및NuGet에 대한 범용 아티팩트 관리

패키지를 파이프라인에 추가

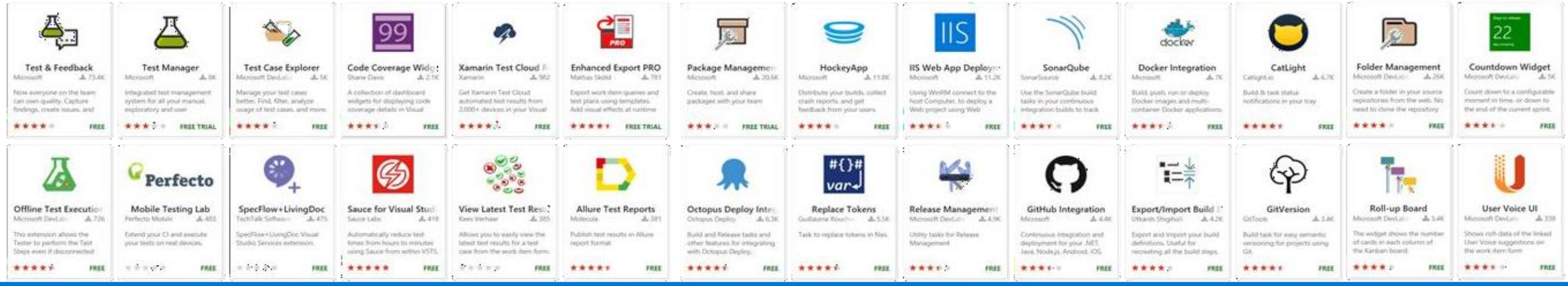
패키지를 공유하고 내장된CI/CD, 버전 관리, 테스팅 사용

효율적인 코드 공유

소규모 팀과 대기업간에 쉽게 코드를 공유

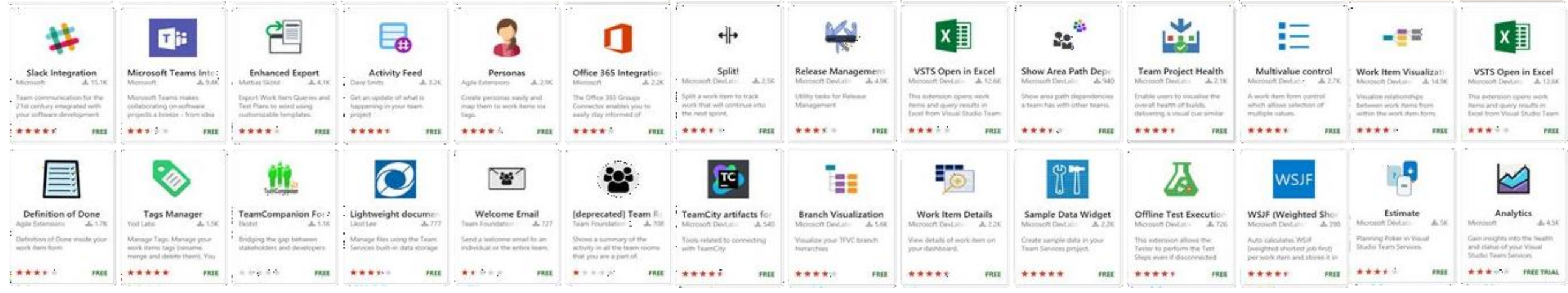


https://azure.com/devops



확장 기능

700개 이상의 Azure DevOps 및 TFS 용 확장 기능
대중적인 상용 및 오픈 소스 도구와의 통합을 제공



실습

그 전에 잠깐!

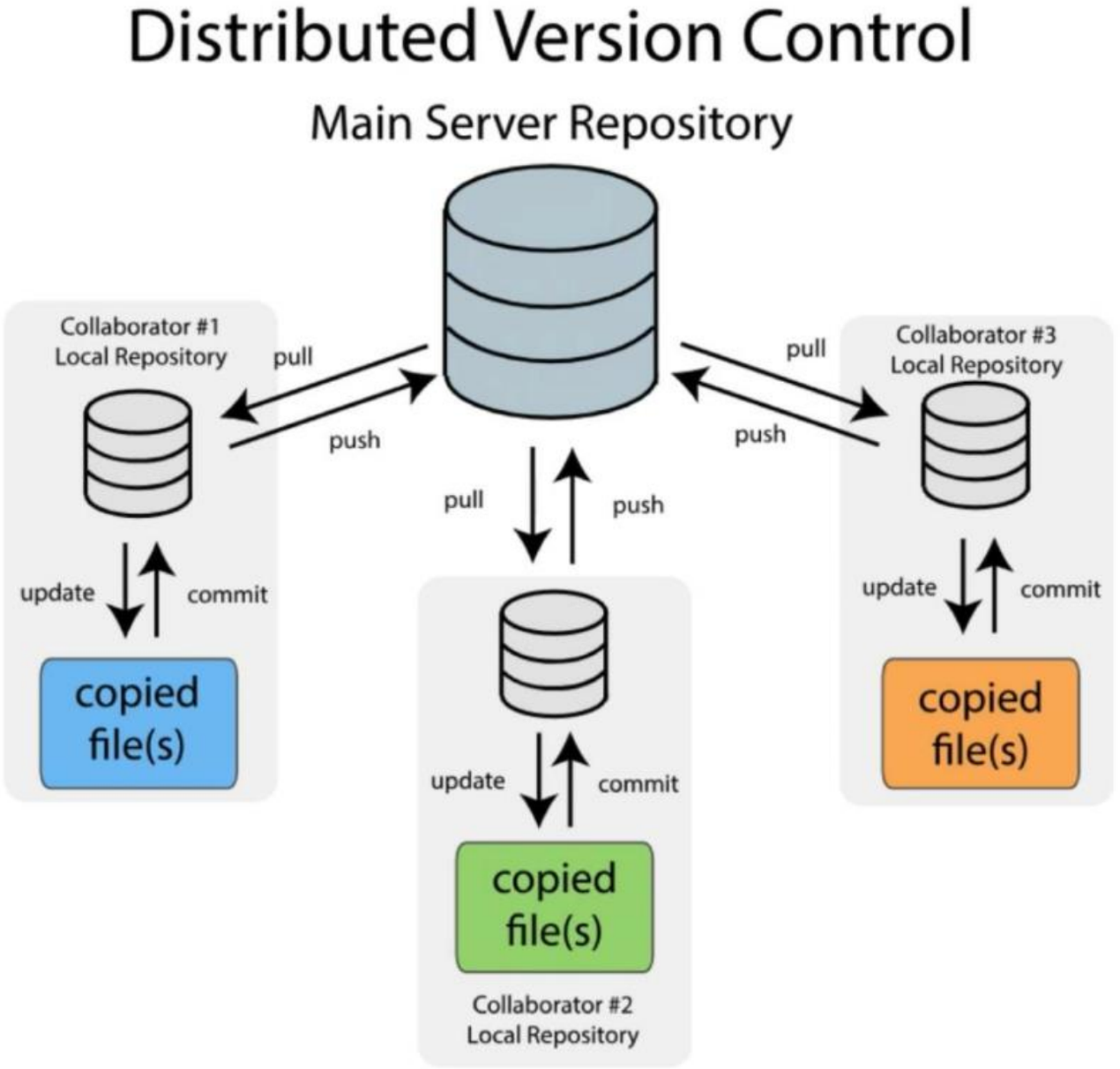
GitHub 에 대해 알아보자!

형상관리 - Git

1. Branch and Merge : 새로운 기능 패치시 브랜치를 생성하고 다시 메인코드로 병합가능
2. Small and Fast : 로컬에서 우선작업하여 빠름
3. 분산형 데이터 모델
4. 데이터 안전 : 모든 파일 체크섬 검사
5. 스테이징 모드 : local repo 와 remote repo 로 2단계 저장소를 가짐

https://git-scm.com/about/

GitHub 구조 이해



출처 - https://bit.ly/2WVzFpf

3 State of Git

Modified : 작업디렉토리에서 데이터가 변경된 상태

Staged : 작업 디렉토리에서 변경된데이타가 commit 되기위해 marked 된 상태

Committed : Local Repo 에 저장된 데이타상태

4 area of Git

: Content 가 생성되고 edit 되고 delete 되는 공간, github 로 부터 git clone 을 받을경우 해당영역이 working directory 가 된다.

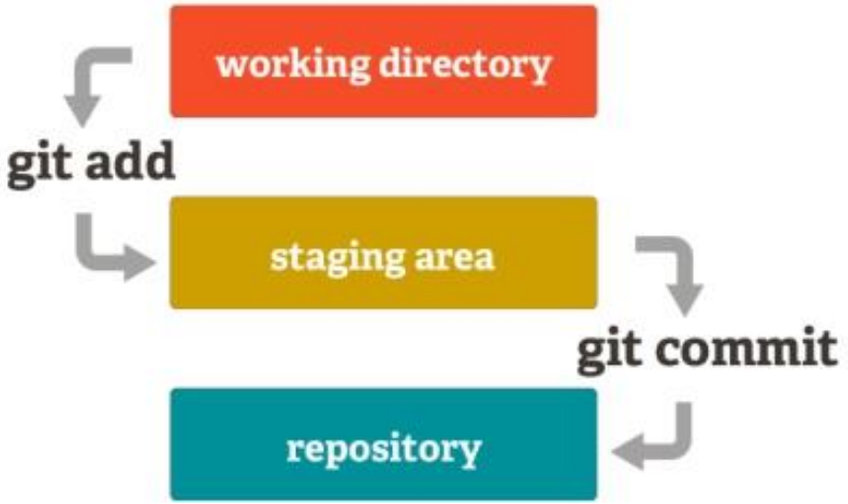
: Committed 되기 위해 대기하는 공간
git add 명령어로 local repo 에 commit 되기 전의 상태

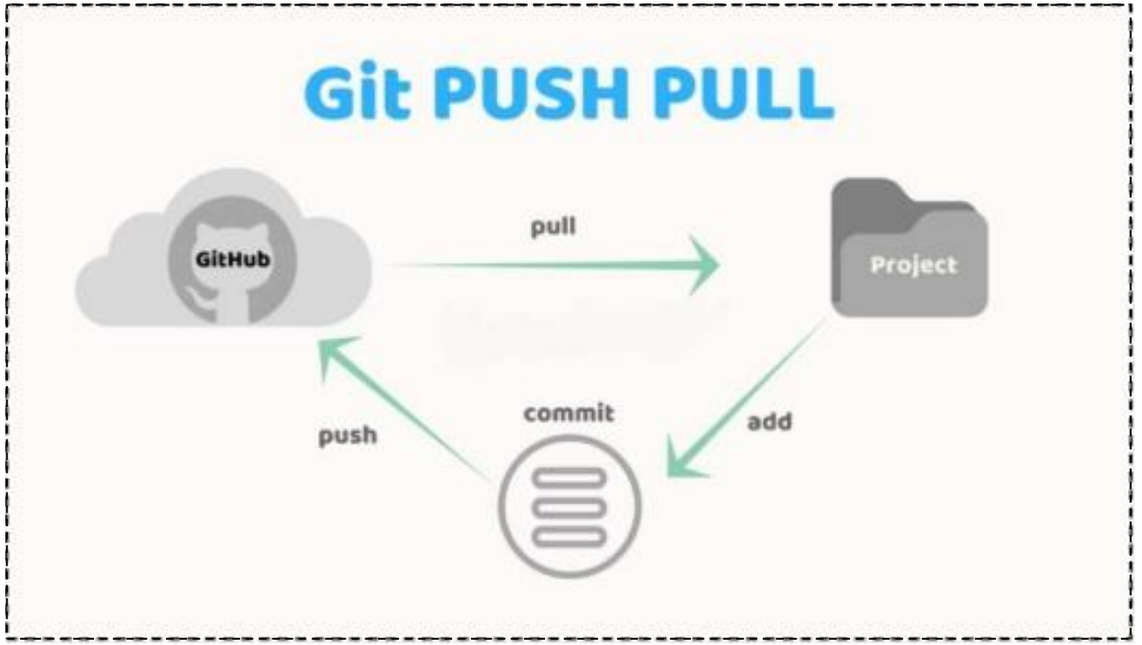
: Commit 된 이후 저장되는 로컬 레포지토리

: 중앙 원격 레포지토리, GitHub

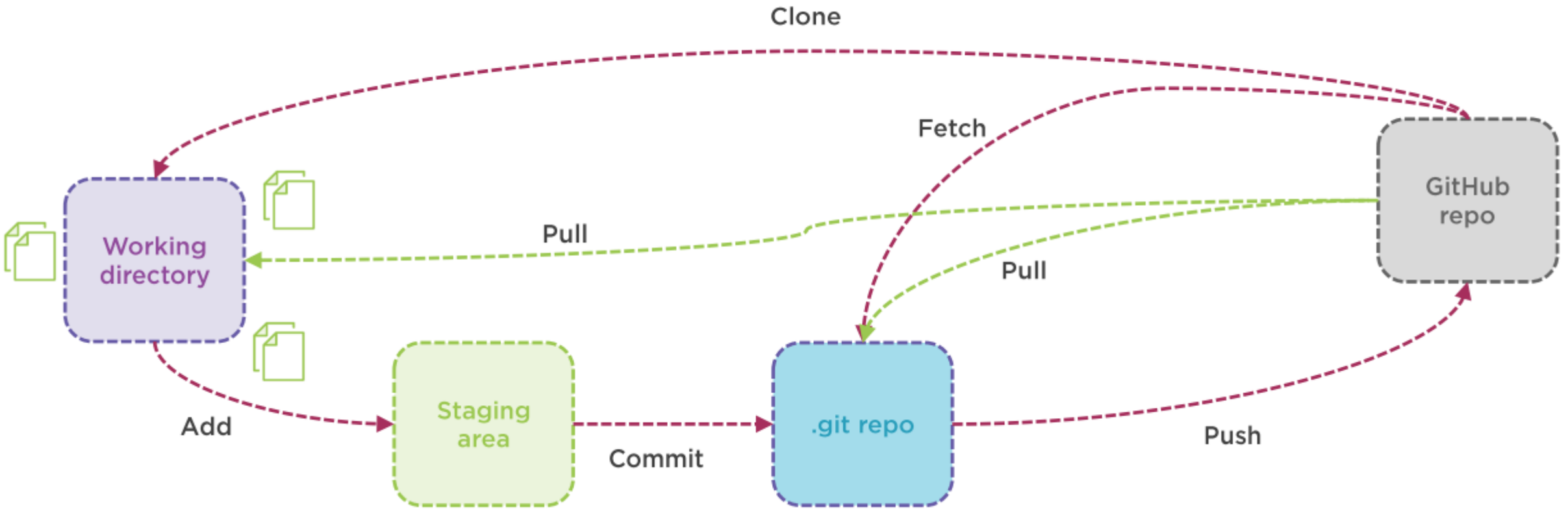
Git 사용해 보기

- github.com에서 repository 생성
- git init : 로컬 repo 생성
- git config
  - git config --global user.name
  - git config --global user.email
- git status
- git add
- git commit
- git remote : 원격 repo 로 부터 project 다운로드
- git clone
- git pull
- git push

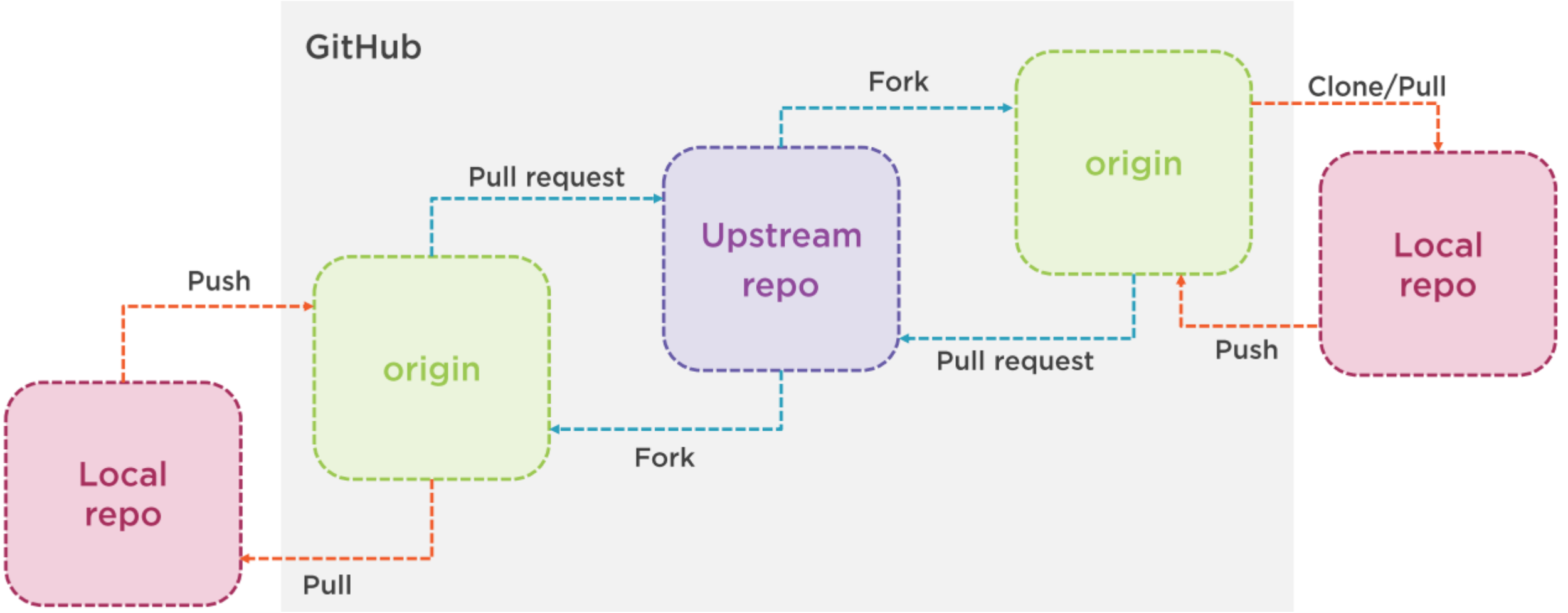




GitHub 구조 이해

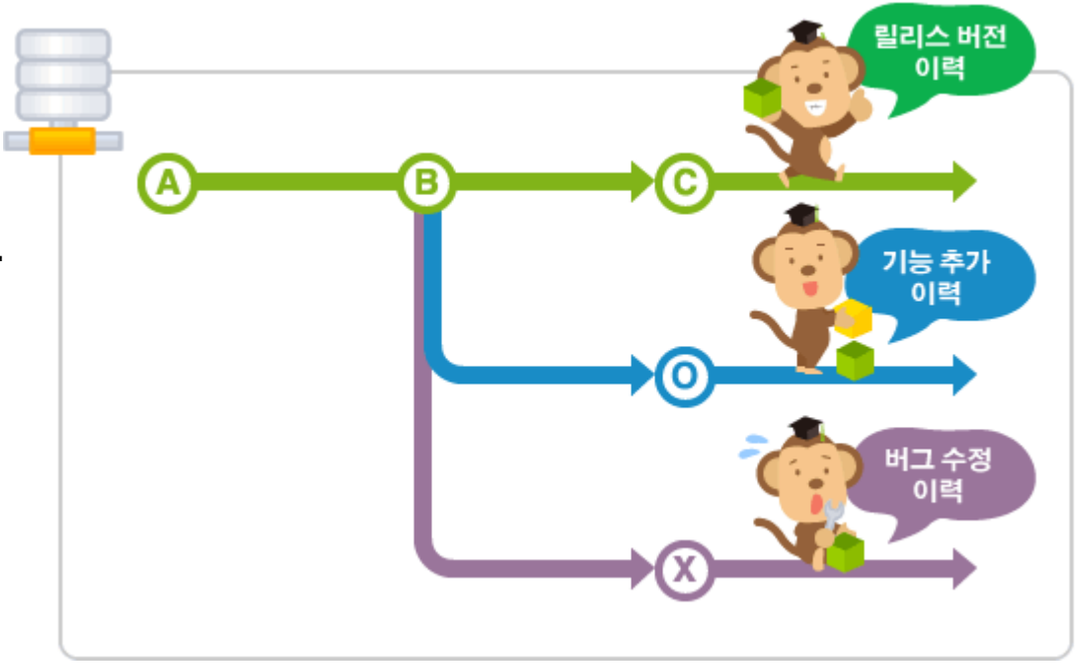

Clone
Fetch
GitHub repo
Pull
Working directory
Pull
Add
Staging area
.git repo
Commit
Push

Forking on Github

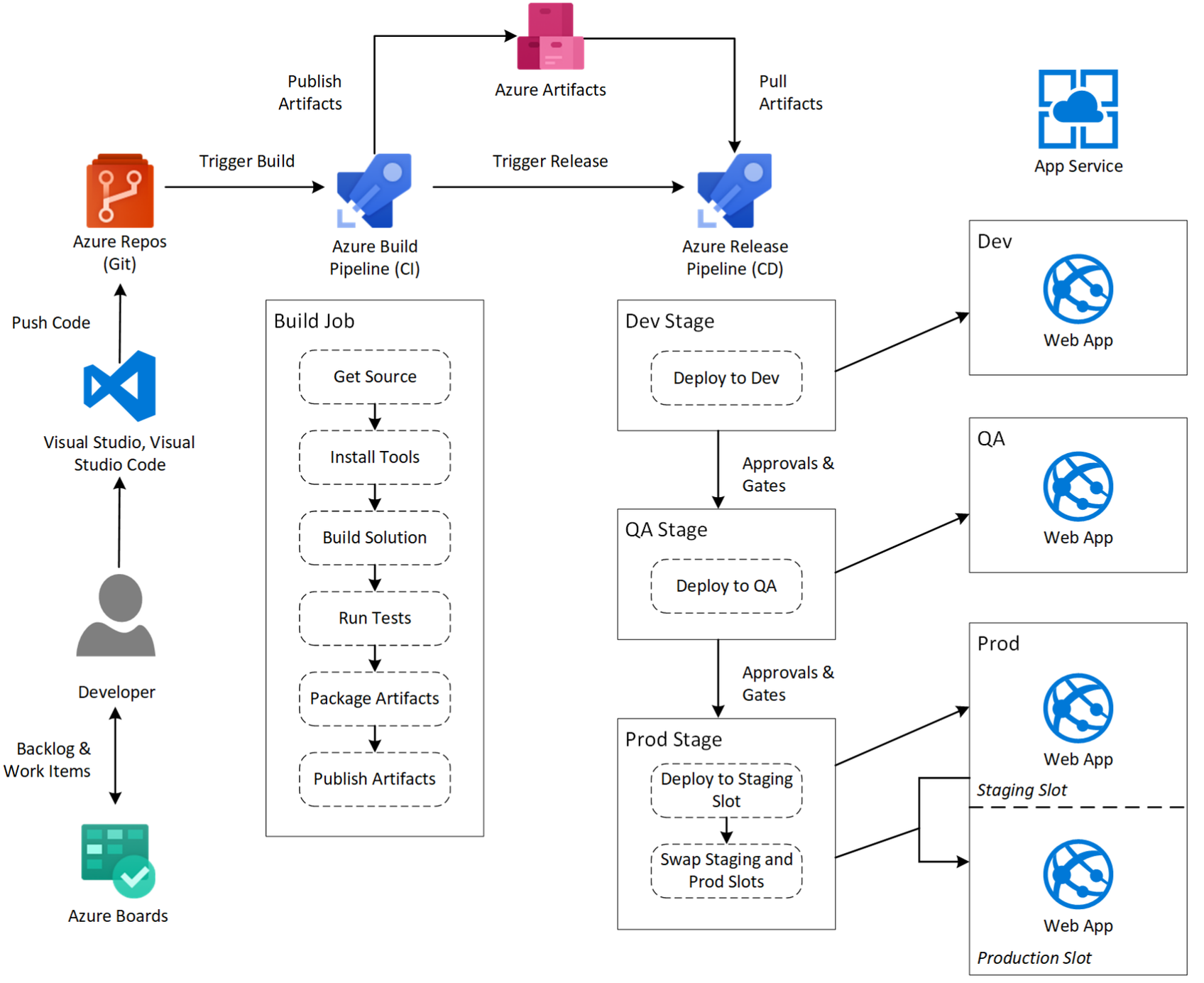


Branch

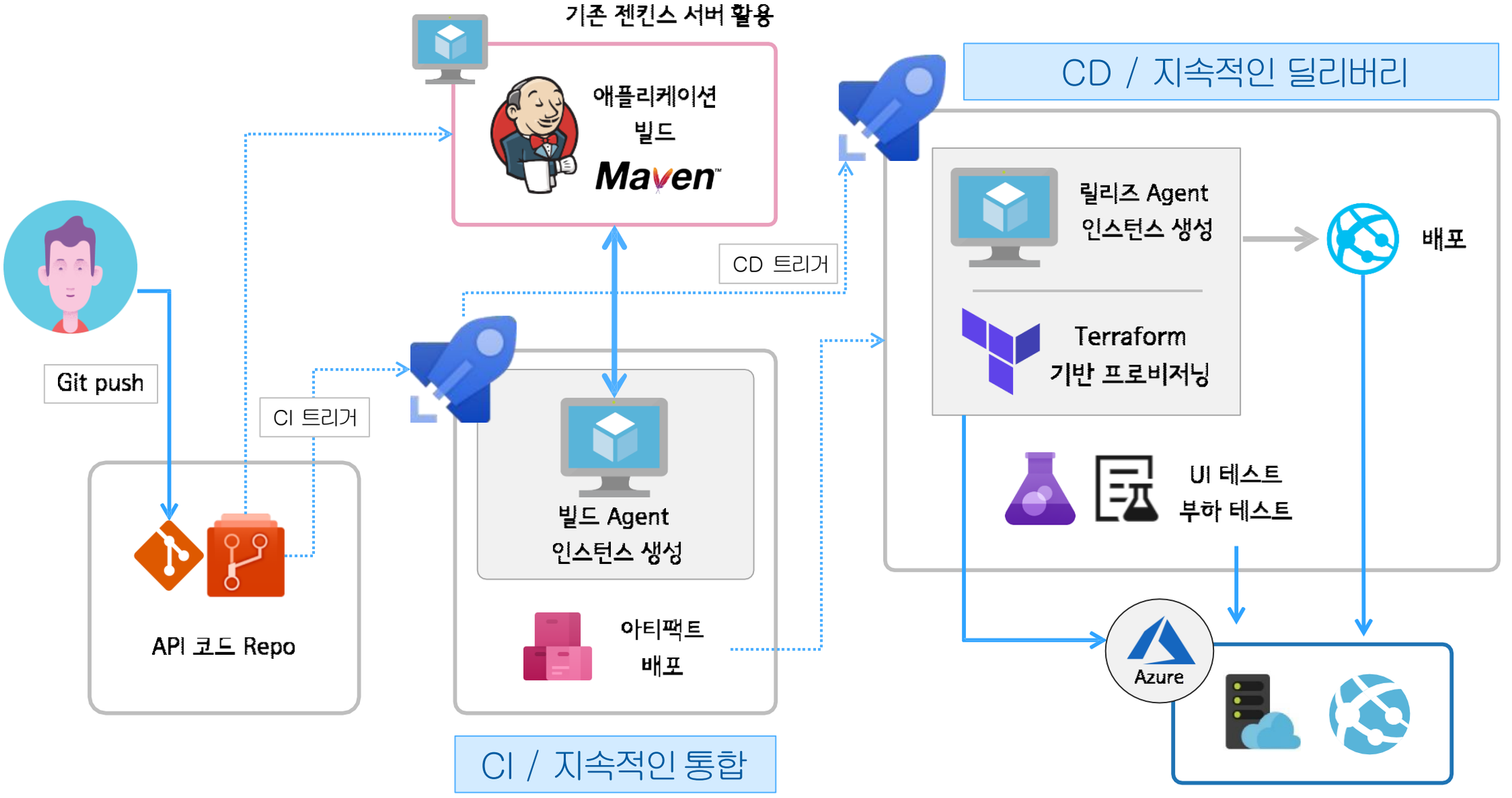
동일한 소스코드로 작업하는 여러 개발자들이
각자 독립적인 작업영역(저장소) 안에서 마음대로
소스코드를 변경하여 서로다른 버전의 코드를
만들수 있다. 필요시 변경된 내용은 원래의
버전과 비교해서 하나의 새로운 버전을 만들수도 있다.



실습환경


Publish Artifacts
Azure Artifacts
Pull Artifacts
App Service
Trigger Build
Trigger Release
Azure Repos (Git)
Azure Build Pipeline (CI)
Azure Release Pipeline (CD)
Push Code
Visual Studio, Visual Studio Code
Developer
Backlog & Work Items
Azure Boards
Build Job
Get Source
Install Tools
Build Solution
Run Tests
Package Artifacts
Publish Artifacts
Dev Stage
Deploy to Dev
Approvals & Gates
QA Stage
Deploy to QA
Approvals & Gates
Prod Stage
Deploy to Staging Slot
Swap Staging and Prod Slots
Dev
Web App
QA
Web App
Prod
Web App
Staging Slot
Web App
Production Slot

#2 Demo


기존 젠킨스 서버 활용
애플리케이션
빌드
Maven
CD / 지속적인 딜리버리
릴리즈 Agent
인스턴스 생성
배포
CD 트리거
Terraform
기반 프로비저닝
Git push
CI 트리거
UI 테스트
부하 테스트
빌드 Agent
인스턴스 생성
아티팩트
배포
API 코드 Repo
Azure
CI / 지속적인 통합