

Cloud-Native Day
2020 Korea
LIVE!

마이크로서비스 전환에 필요한
2가지 측면의 접근법

Jung In Lee

vmware®

Today's Sessions

1. 마이크로서비스 디자인
2. 마이크로서비스 구축
3. 마이크로서비스 런타임 & 네트워크
4. 마이크로서비스 모니터링 및 관리

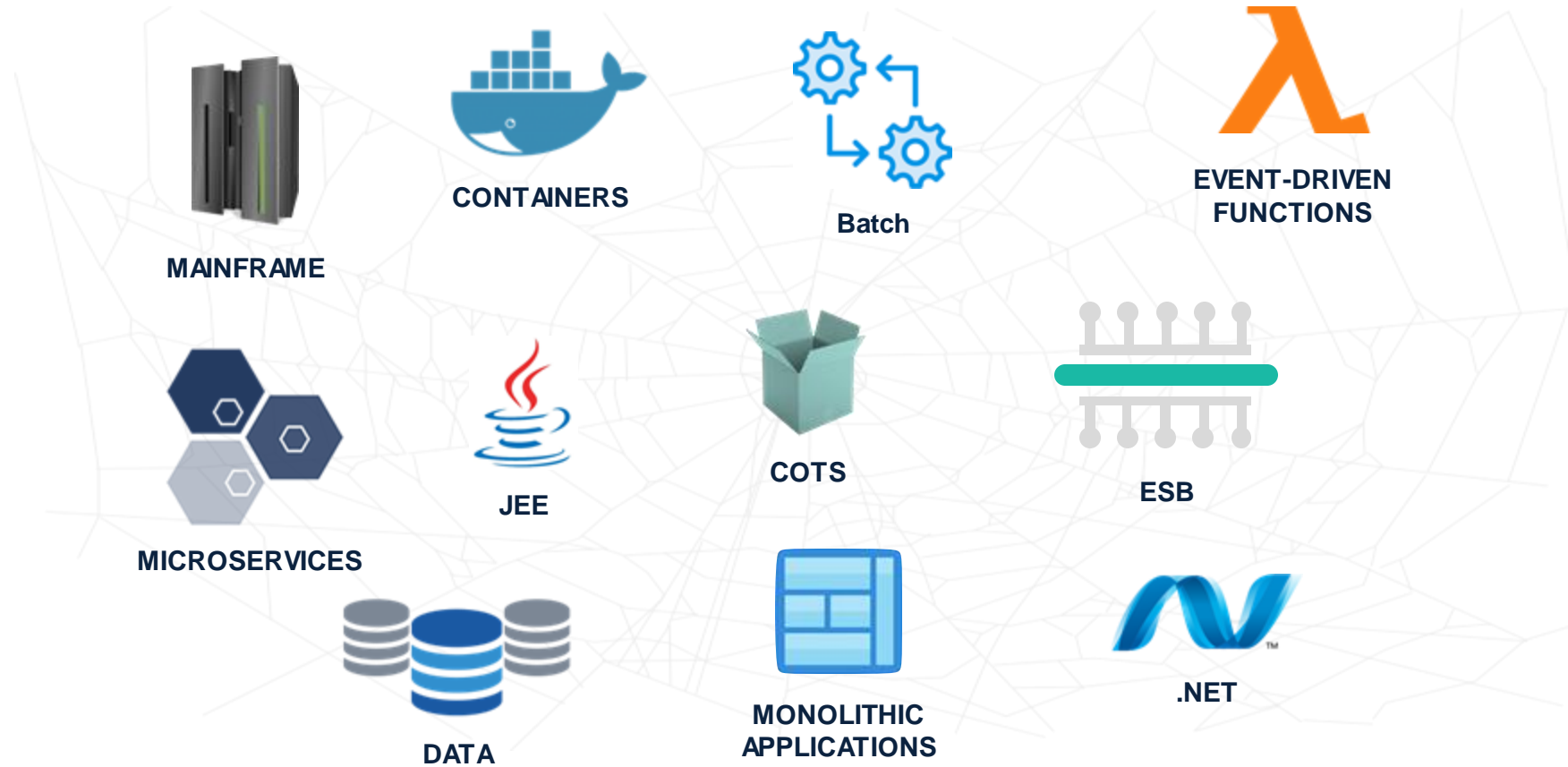
“How do we move toward
modern apps
with greater velocity?”

“How do we operate
modern infrastructure
with more stability at scale?”

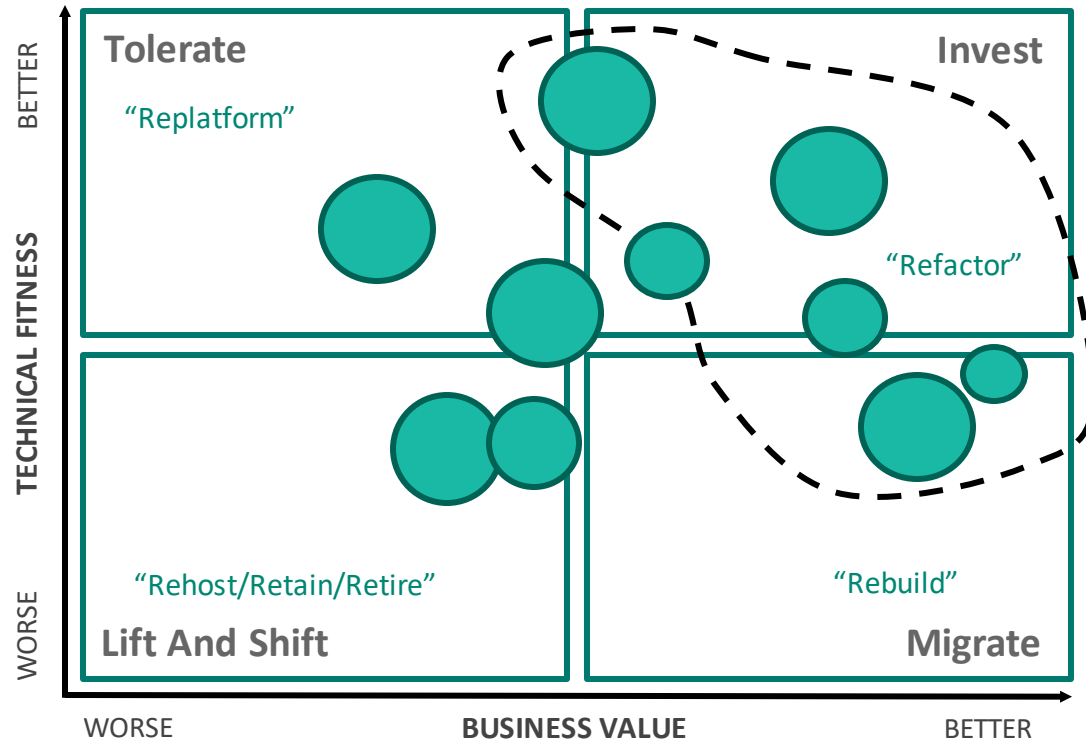
7 Key Questions

1. 모든 Application이 마이크로 서비스에 적합한 것인가?
2. 우리가 얻으려고 하는 마이크로 서비스의 Value는 무엇인가?
3. 어떠한 Criteria에 가장 우선순위를 두고 아키텍처를 설계할 것인가?
4. 전환의 방법은 Big Bang 인가? 혹은 점진적 소멸(Strangler) 방식인가?
5. 어떤 Implementation 패턴을 이용하여 마이크로 서비스를 구현할 것인가?
6. 마이크로 서비스 운영에 필요한 툴에 따라 코딩 패턴의 Dependency는 생기는가?
7. 성공적인 MSA 경험을 다른 개발 조직으로 어떻게 전파할 수 있을까?

1. 모든 Application이 마이크로 서비스에 적합한 것인가?



1. 모든 Application이 마이크로 서비스에 적합한 것인가?



Upload binary or map to source control

Parsers for most common languages (Java, .NET, PHP, Ruby, etc.)

Extensible rules (100s available currently)

UI dashboard to help triage decisioning

Understand technical effort to move your app

The dashboard displays the following information:

- Summary:** Info (260), Low (190), Medium (14), High (14), Total (478).
- CSV Table:**

| Name | Tags | FileType | Target | Type | Advice | Effort | Category | Criticality |
|-----------------------|-------------------------|----------|--------|-------|---|--------|------------|-------------|
| glassj | | java | | | | | | |
| java-glassfish-import | [imports] [third-party] | java | line | regex | Refer to 3rd party organization for cloud affinity of library | | ThirdParty | medium |
- Summary Metrics:** APPS: 13, LOC: 275066, FILES: 3047, FINDINGS: 8102, SCORE: 6.03.
- RECOMMENDATION:** A scatter plot showing technical score vs business value for applications like MuleSoft and Selenium.

2. 우리가 얻으려고 하는 마이크로 서비스의 Value는 무엇인가?



3. 어떠한 Criteria에 가장 우선순위를 두고 아키텍처를 설계할 것인가?

Organization Ownership : 서비스의 담당 현업팀이 분리 되어있는가?

Release Cycle : 서비스의 업데이트 주기를 독립적으로 가져가야 하는가?

Scaling : 서비스 마다 별도의 확장성을 확보해야 하는가?

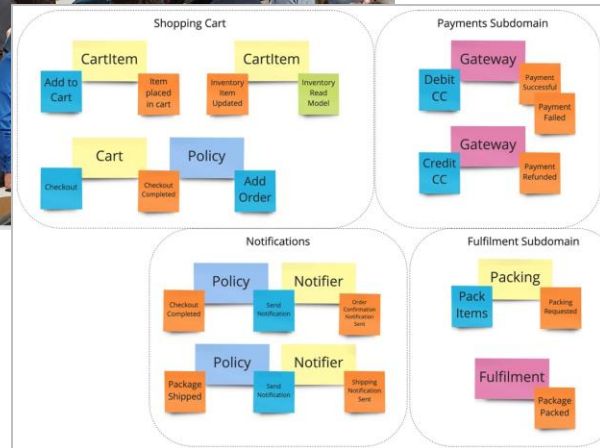
Data and Transaction : 데이터가 분리되어야 하는가?

SPOF : 하나의 서비스에 너무 많은 의존성이 존재하지 않는가?

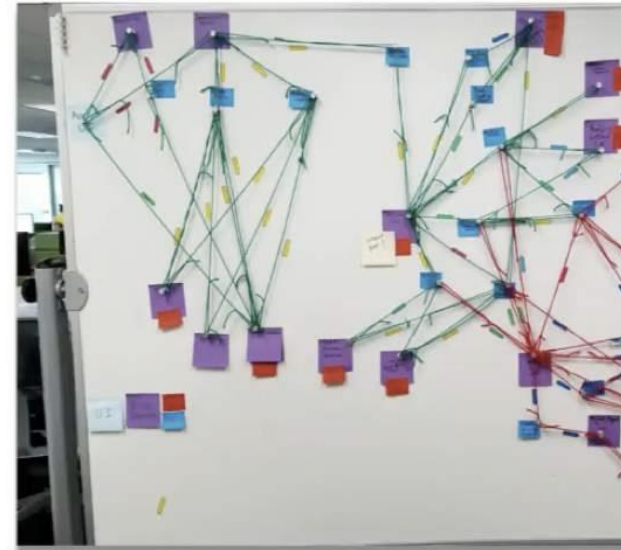
Maintenance Effort : 소수의 개발/운영 조직이 운영을 감당할 수 있는가?

3. 어떠한 Criteria에 가장 우선순위를 두고 아키텍처를 설계할 것인가?

Domain Driven Design



Boris Diagram



SNAP-E

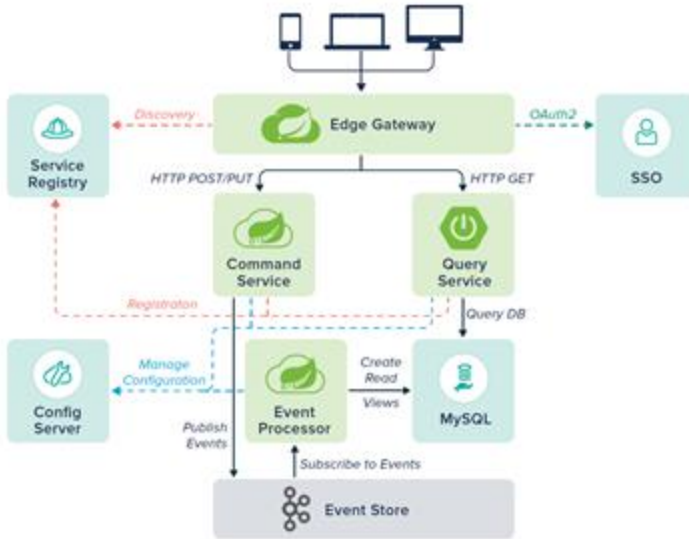


4. 전환의 방법은 Big Bang 인가? 혹은 점진적 소멸(Strangler) 방식인가?

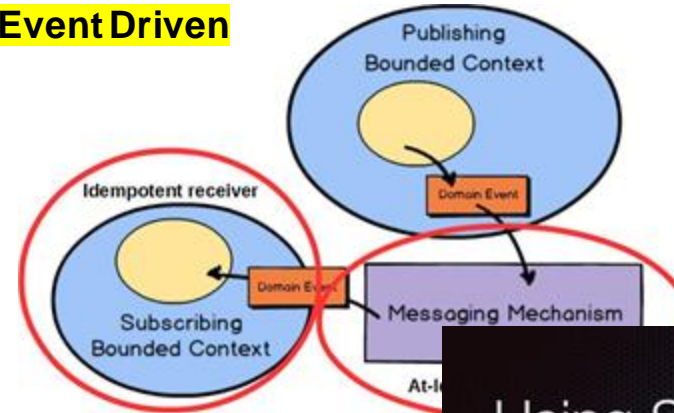


5. 어떤 Implementation 패턴을 이용하여 마이크로 서비스를 구현할 것인가?

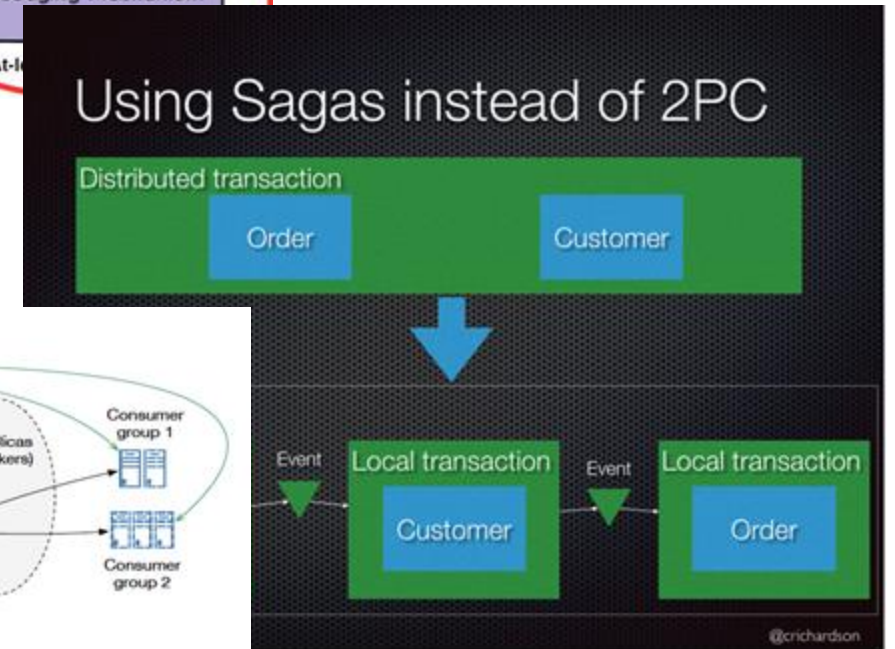
CQRS



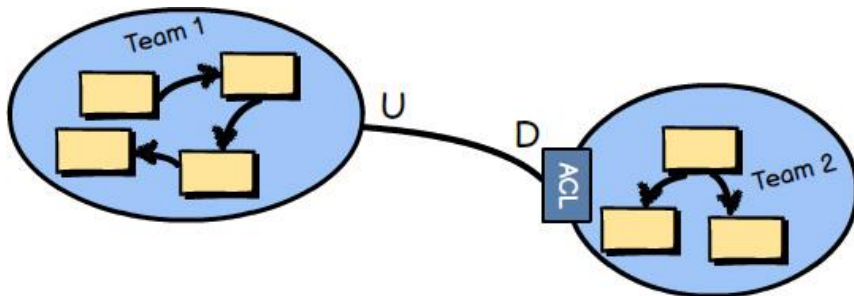
Event Driven



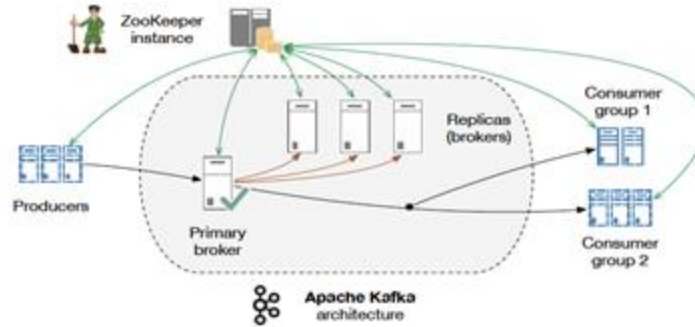
Saga for distributed txns



Anti Corruption Layer



Messaging Queue

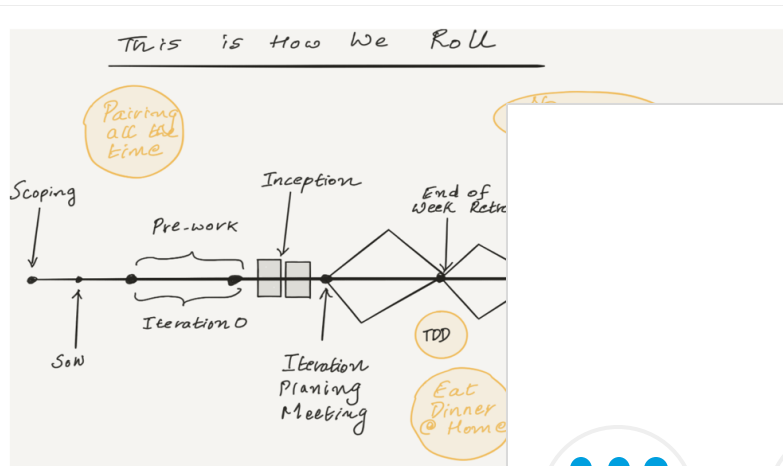
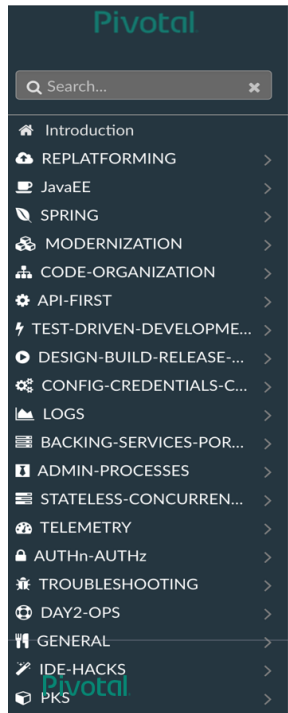


6. 마이크로 서비스 운영에 필요한 툴에 따라 코딩 패턴의 Dependency는 생기는가?



7. 성공적인 MSA 경험을 다른 개발 조직으로 어떻게 전파할 수 있을까?

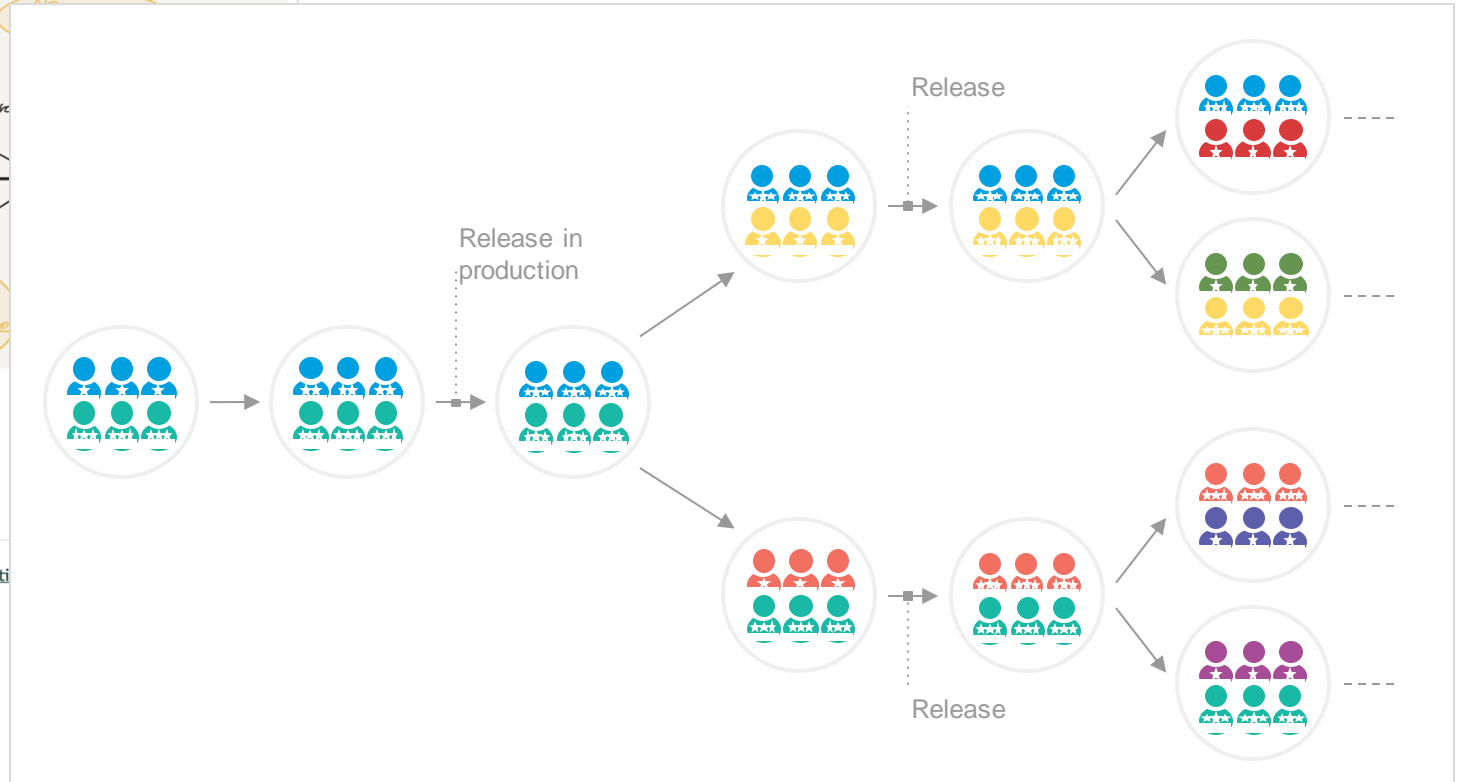
Cookbook and Recipes



Most recent recipes

- 12-01-2017 - [README](#)
- 11-29-2017 - [Secure caching](#)
- 11-27-2017 - [Configure PCF Autoscaler](#)
- 11-24-2017 - [Building and Deploying your first SpringBoot App](#)
- 11-17-2017 - [How to manage keys and certificates for X.509 authentication](#)
- 11-16-2017 - [Blue-green deployments](#)

Pollination Mode Scaling



7 Key Questions

1. 모든 Application이 마이크로 서비스에 적합한 것인가?

2. 우리가 얻으려고 하는 마이크로 서비스의 Value는 무엇인가?

3. 어떠한 Value를 가장 우선순위를 두어, 아키텍처를 설계할 것인가?

4. 전환의 방법은 Big Bang 인가? 혹은 점진적 소멸(Strangler) 방식인가?

5. 어떤 Implementation 패턴을 이용하여 마이크로 서비스를 구현할 것인가?

6. 마이크로 서비스 운영에 필요한 툴에 따라 코딩 패턴의 Dependency는 생기는가?

7. 성공적인 MSA 경험을 다른 개발 조직으로 어떻게 전파할 수 있을까?

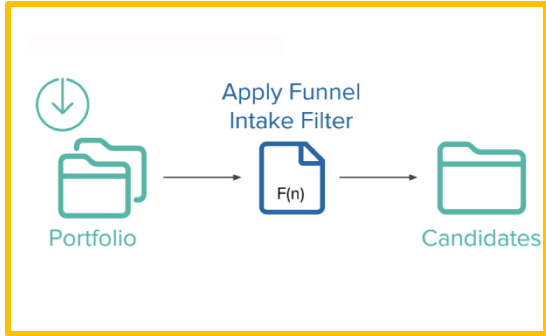
1. VMware
Journey Navigator
Program

2. VMware
App Navigator
Program

3. VMware
Modernization
Program

Application Transformation Process

1. VMware Journey Navigator Program



2. Event Storm the legacy space
Highly collaborative modeling tornado!
Make sense of a complex business domain with a common language for different tribes. Identify trouble spots & starting points.

1. Define Objectives & Key Results
Decide on direction for outcomes & approach. Agree on objectives and refine key results to measure those.

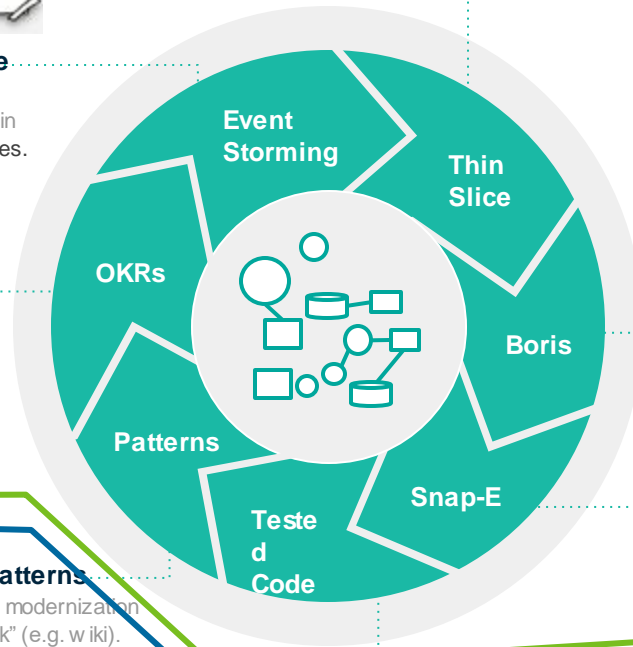
We will objective
as measured by key results



7. Cloud Modernization Patterns
Reuse and document the learned modernization patterns as recipes in a "cookbook" (e.g. wiki). Consolidate knowledge and accelerate team productivity.

3. VMware Modernization Program

2. VMware App Navigator Program



3. Select "Thin Slice(s)"

Cut the monolithic business cake and pick a sweet piece to start the modernization work.



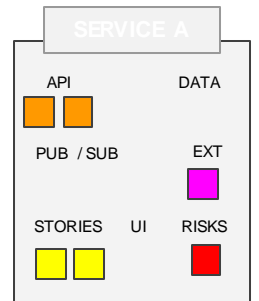
4. "Boris" crafts the desired notional architecture

Model the API capabilities. Model the business capabilities.



5. Snap-E

Fill the backlog with actionable user stories.



6. Produce tested and working code

Take low-level design decisions, select technical patterns to use, and implement user stories. Time-bounded set of 1-week iterations.

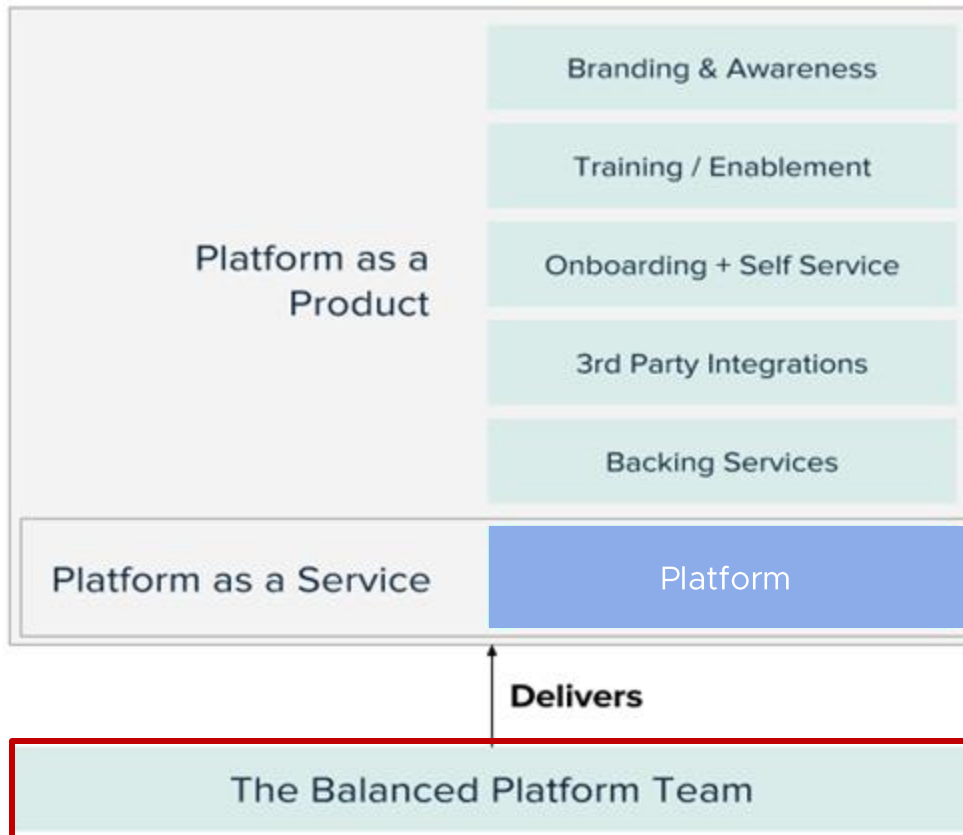


4 Key Questions

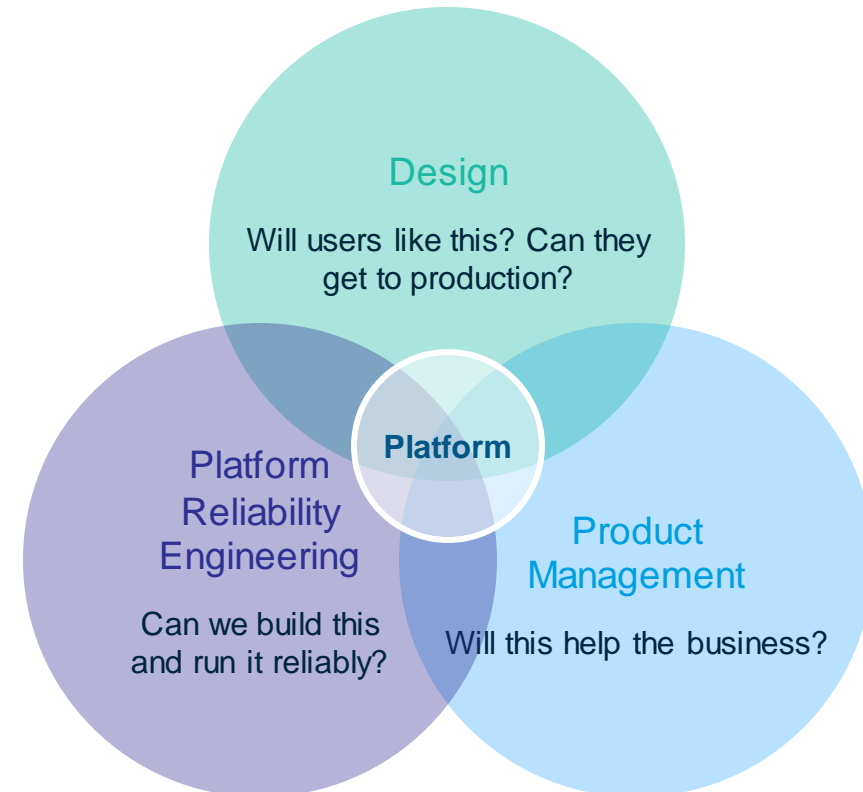
1. Platform as a Product 사상을 이해하고 있는 팀인가?
2. 플랫폼 Team의 목표에 따른 역할과 그에 따른 역량 강화는 어떻게 할 것인가?
3. K8S 기반의 플랫폼 위에 어떤 Tooling을 선택하고 적용할 것인가?
4. 더 많은 애플리케이션을 Hosting 하기 위한 플랫폼 팀의 전략은 무엇인가?

1. Platform as a Product 사상을 이해하고 있는 팀인가?
2. 플랫폼 Team의 목표에 따른 역할과 그에 따른 역량 강화는 어떻게 할 것인가?

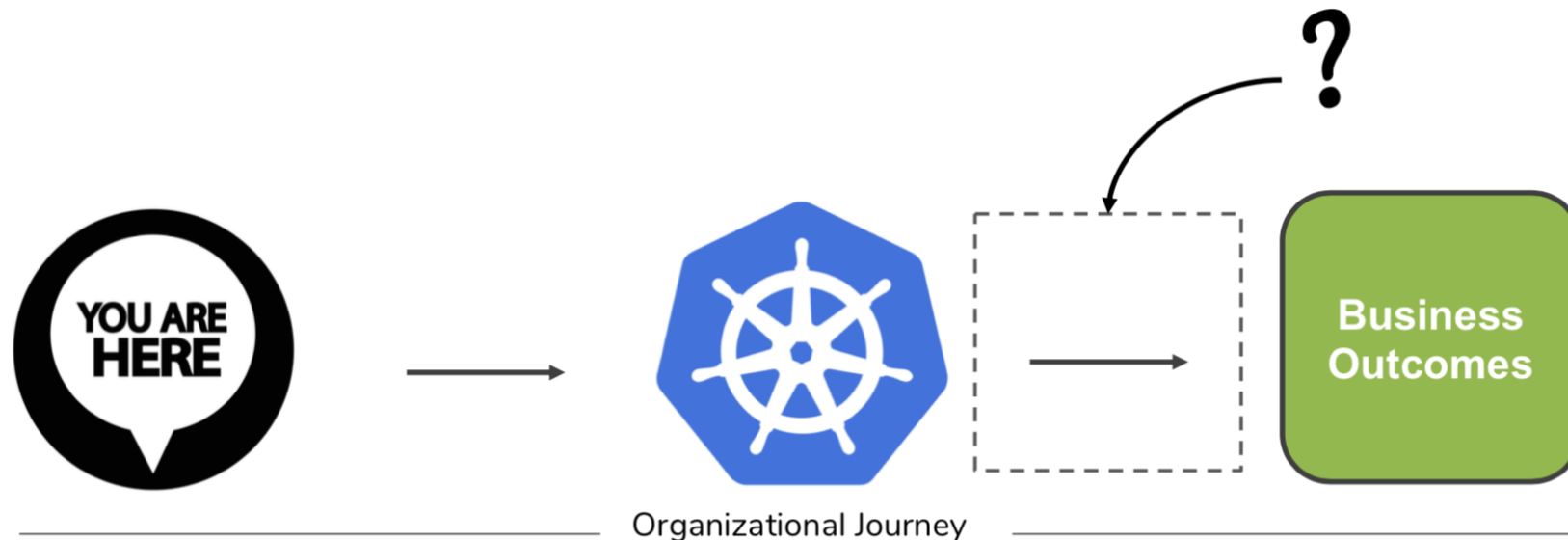
Platform as a Product



Balanced Platform Team



- 3. K8S 기반의 플랫폼 위에 어떤 Tooling을 선택하고 적용할 것인가?
- 4. 더 많은 애플리케이션을 Hosting 하기 위한 플랫폼 팀의 전략은 무엇인가?

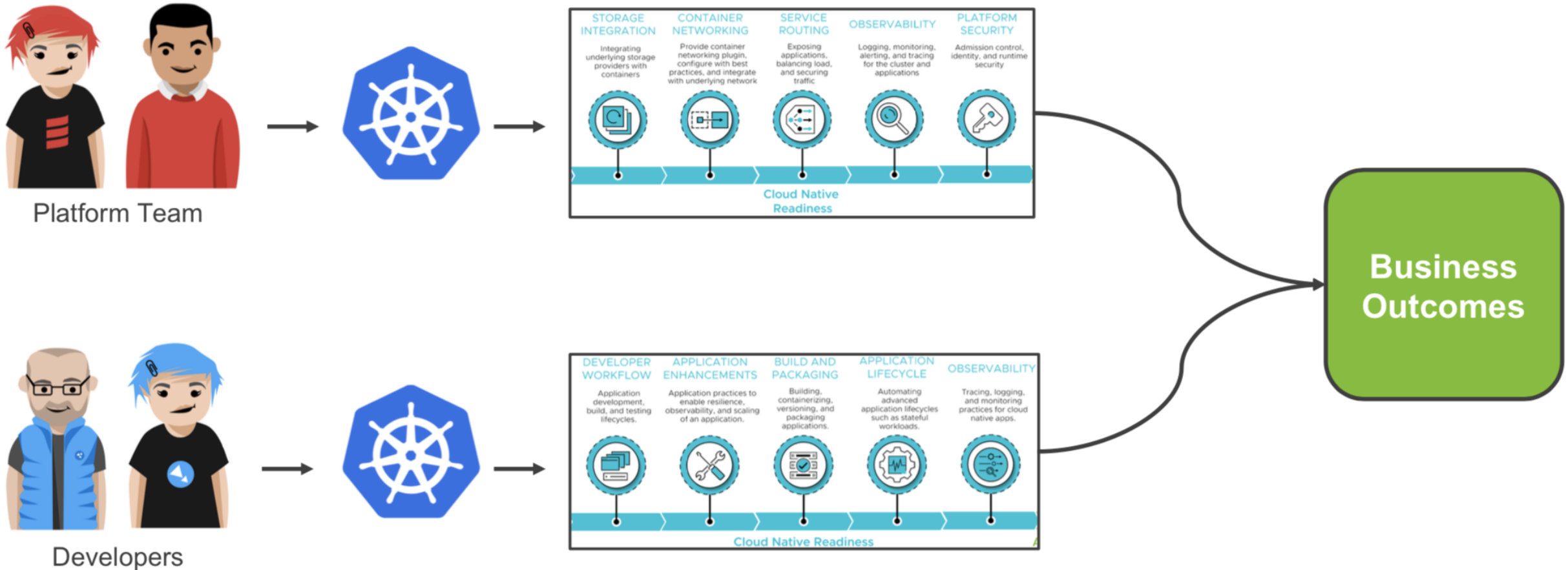


- 3. K8S 기반의 플랫폼 위에 어떤 Tooling을 선택하고 적용할 것인가?
- 4. 더 많은 애플리케이션을 Hosting 하기 위한 플랫폼 팀의 전략은 무엇인가?

The image displays a large grid of cloud-native tooling, organized into several functional categories:

- App Definition and Development:** Includes Database (e.g., Vitess, KV), Streaming & Messaging (e.g., cloudevents, NATS), Application Definition & Image Build (e.g., HELM, CHEF), and Continuous Integration & Delivery (e.g., argo, flux).
- Orchestration & Management:** Includes Scheduling & Orchestration (e.g., kubernetes), Coordination & Service Discovery (e.g., CoreDNS, etcd), Remote Procedure Call (e.g., gRPC), Service Proxy (e.g., envoy), API Gateway (e.g., Kong), and Service Mesh (e.g., LINKERD).
- Runtime:** Includes Cloud Native Storage (e.g., minio), Container Runtime (e.g., cri-o), and Cloud Native Network (e.g., CNV).
- Provisioning:** Includes Automation & Configuration (e.g., Ansible, Chef), Container Registry (e.g., Harbor), Security & Compliance (e.g., Falco), and Key Manage (e.g., Vault).
- Platform:** Includes Certified Kubernetes - Distribution (e.g., OpenShift), Certified Kubernetes - Hosted (e.g., AWS EKS), and Certified Kubernetes - Installer (e.g., kubeadm).
- Observability and Analysis:** Includes Monitoring (e.g., Prometheus), Logging (e.g., fluentd), Tracing (e.g., Jaeger), and Chaos Engineering (e.g., Chaos Mesh).
- Tools:** A collection of various utility tools.
- Framework:** A collection of framework tools.
- Hosted Platform:** A collection of hosted platform tools.
- Immutable Platform:** A collection of immutable platform tools.

- 3. K8S 기반의 플랫폼 위에 어떤 Tooling을 선택하고 적용할 것인가?
- 4. 더 많은 애플리케이션을 Hosting 하기 위한 플랫폼 팀의 전략은 무엇인가?



4 Key Questions

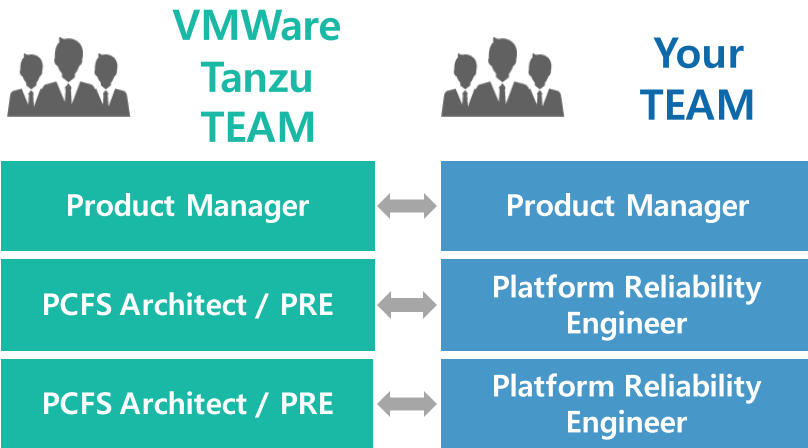
1. Platform as a Product 사상을 이해하고 있는 팀인가?
2. 플랫폼 Team의 목표에 따른 역할과 그에 따른 역량 강화는 어떻게 할 것인가?
3. K8S 기반의 플랫폼 위에 어떤 Tooling을 선택하고 적용할 것인가?
4. 더 많은 애플리케이션을 Hosting 하기 위한 플랫폼 팀의 전략은 무엇인가?

Platform
Dojo

K8S
Service

Platform Dojo & K8S Services

1. Platform Dojo



플랫폼 팀은 플랫폼의 공통서비스 개발, 브랜드, 프로모션, 교육, 채용 등에 걸친 포괄적인 역할 정의가 필요합니다.

Platform as a Product

제품(Product)는 항상 "유지"만 되어서는 안되고, 요구에 따라 "발전" 되고 "업데이트" 되어야 합니다.

SRE

오퍼레이션을 "소프트웨어" 방식으로 보며, Toil을 최소화 하며, 어플리케이션의 속도와 안정성을 함께 추구하는 팀입니다.

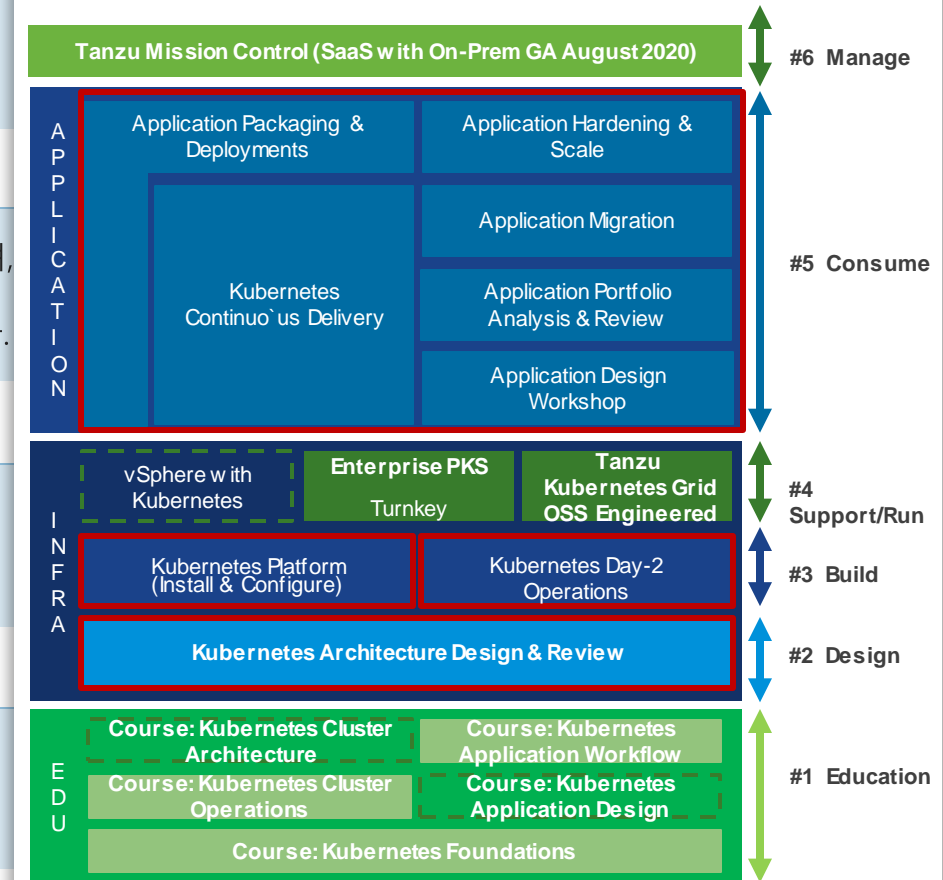
Dedicated Balanced Team

작은 규모의 플랫폼 팀은 수천 개의 앱과 "수천 명의 개발자"의 요구를 받아들일 수 있어야 합니다.

Platform as a Product

플랫폼이 Production으로 갈 수 있는 유일한 통로로서 역할을 해야 합니다.

2. K8S Services



“How do we move toward
modern apps
with greater velocity?”

“How do we operate
modern infrastructure
with more stability at scale?”

Cloud-Native Day
2020 Korea
LIVE!

Thank you

Jung In Lee (ljungin@vmware.com)