

コンテナの本番運用に 必要なポイント

NAMU TECH JAPAN
川俣 悠一

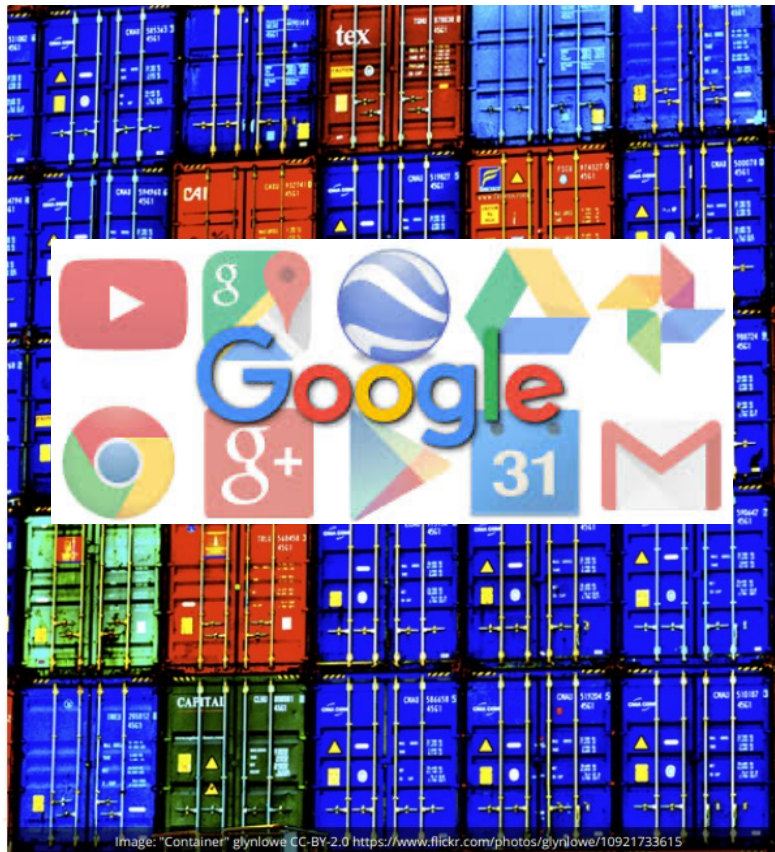
- コンテナ・Kubernetes
- コンテナの本番運用に必要なポイント
- コンテナ利用を容易にするソリューション



コンテナ・Kubernetes

コンテナ = 仮想化の技術

グローバルで多くのコンテナを運営するGoogle



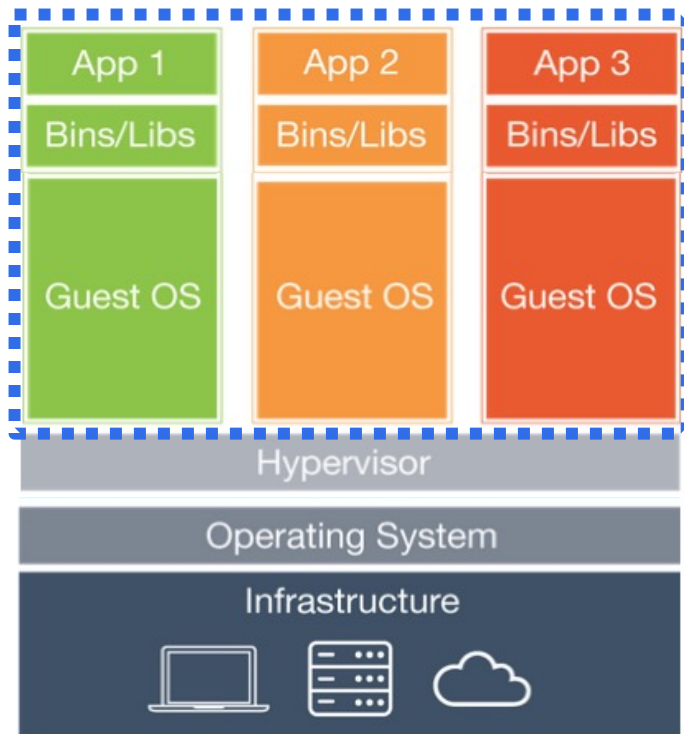
Google and Containers

Everything at Google runs in a container.

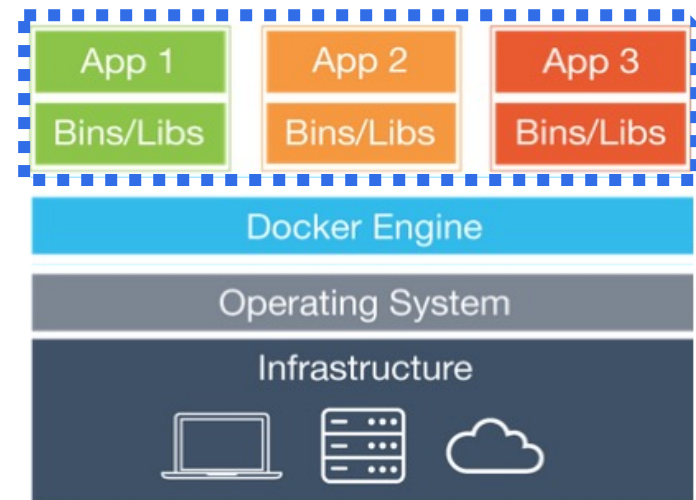
Internal usage:

- Resource isolation and predictability
- Quality of Services
 - batch vs. latency sensitive serving
- Overcommitment (not for GCE)
- Resource Accounting

We start over 2 billion containers per week.



仮想マシン (VM)



コンテナ

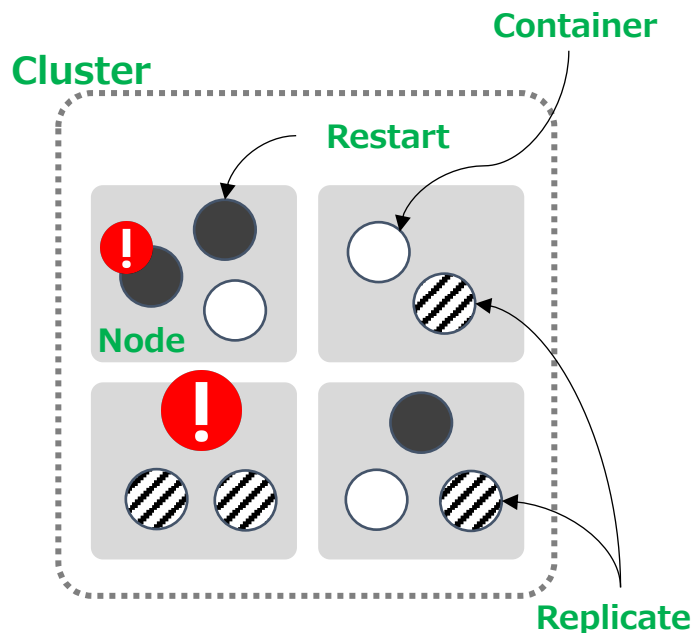
- 仮想化技術のひとつ
- アプリケーションの実行に必要なものだけをパッケージ化して実行
- OSではなく、プロセスレベルで実行されるため、軽量かつ高速
- 代表的なコンテナエンジン： Docker, containerd, CRI-O



kubernetes

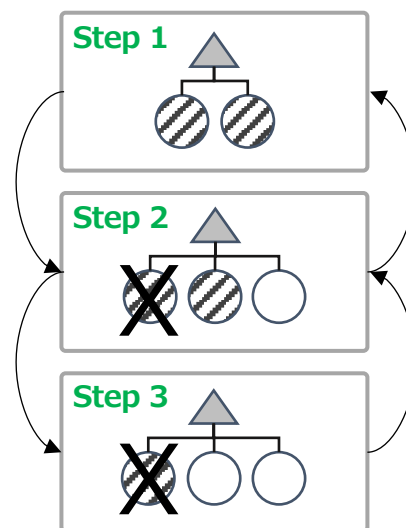
コンテナの運用を自動化する
コンテナオーケストレーションツール

- 複数ホストにおけるコンテナ管理
- アプリケーションのデプロイとアップデート自動化
- コンテナの自動スケーリング
- コンテナの自動復旧
- サービスの宣言的な管理



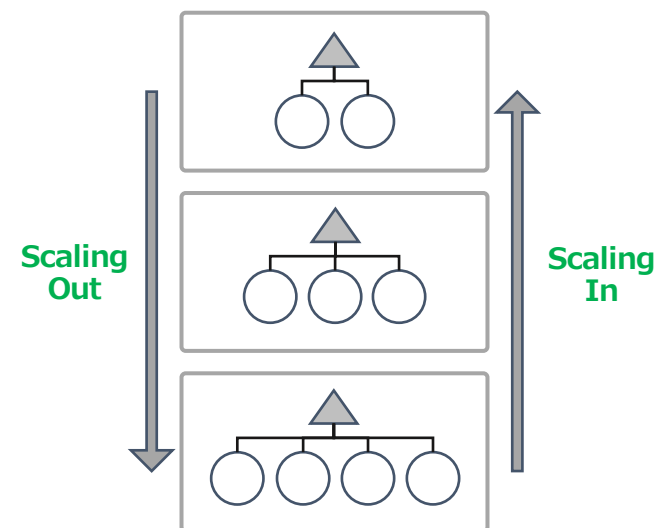
Self Healing

コンテナの自動復旧



Rolling Update

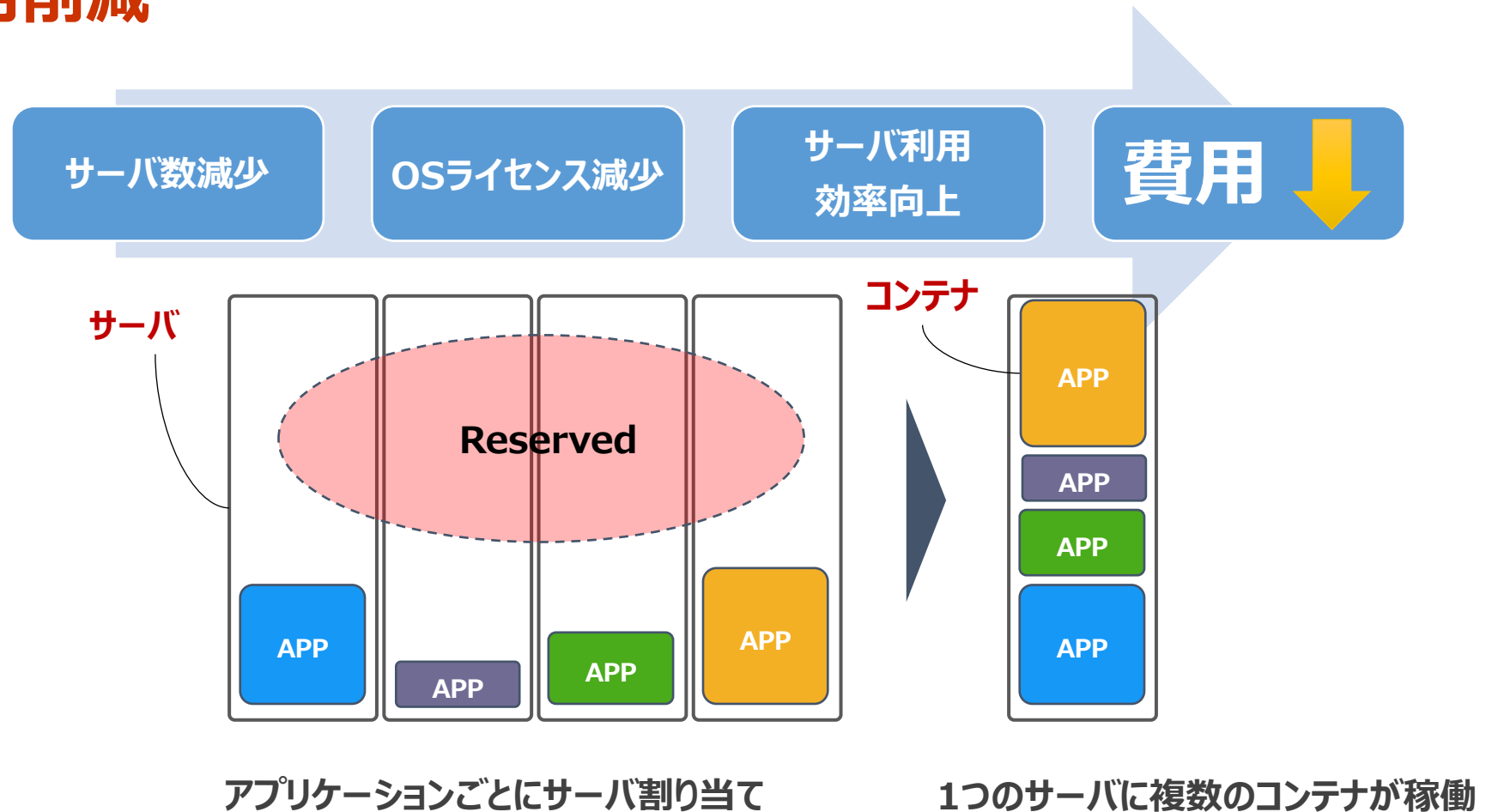
アプリケーション
無停止アップデート



Auto Scaling

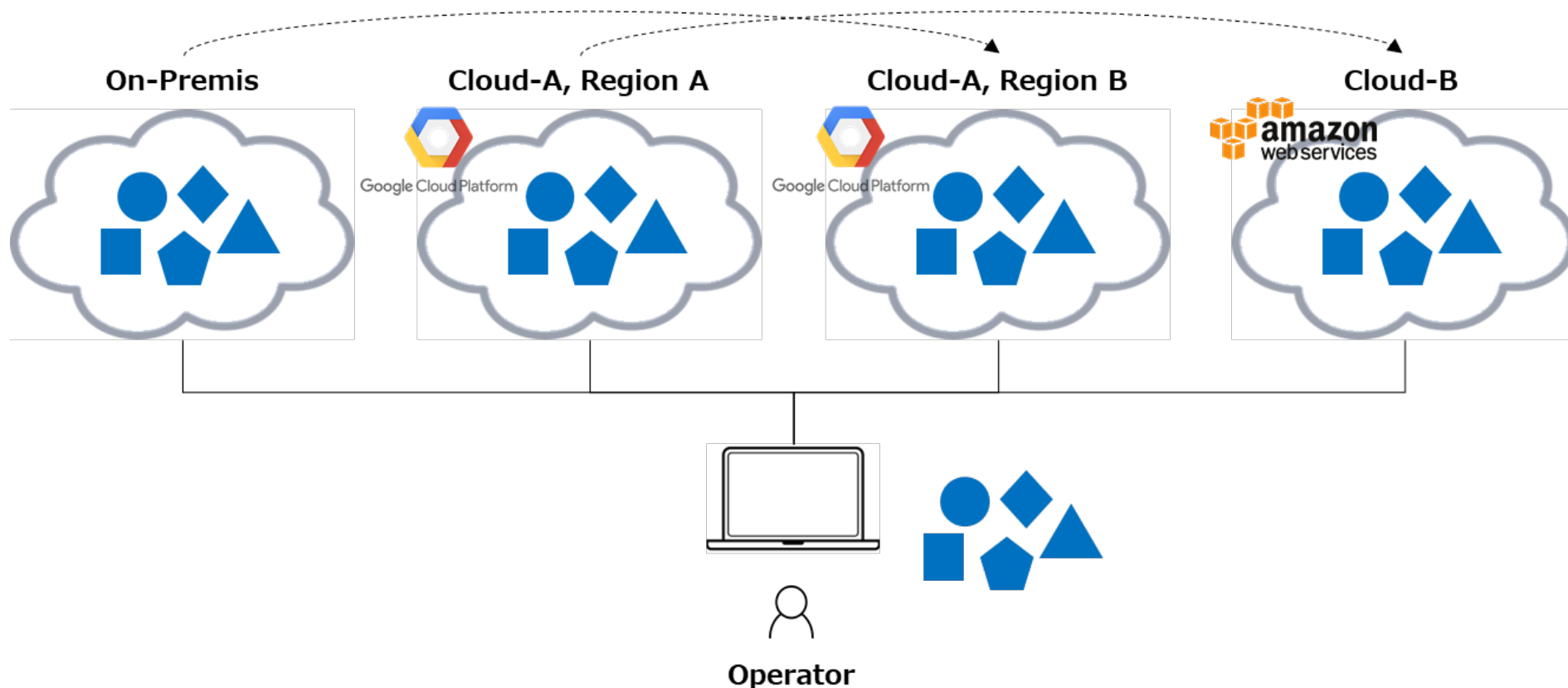
コンテナ数を
自動調整

費用削減

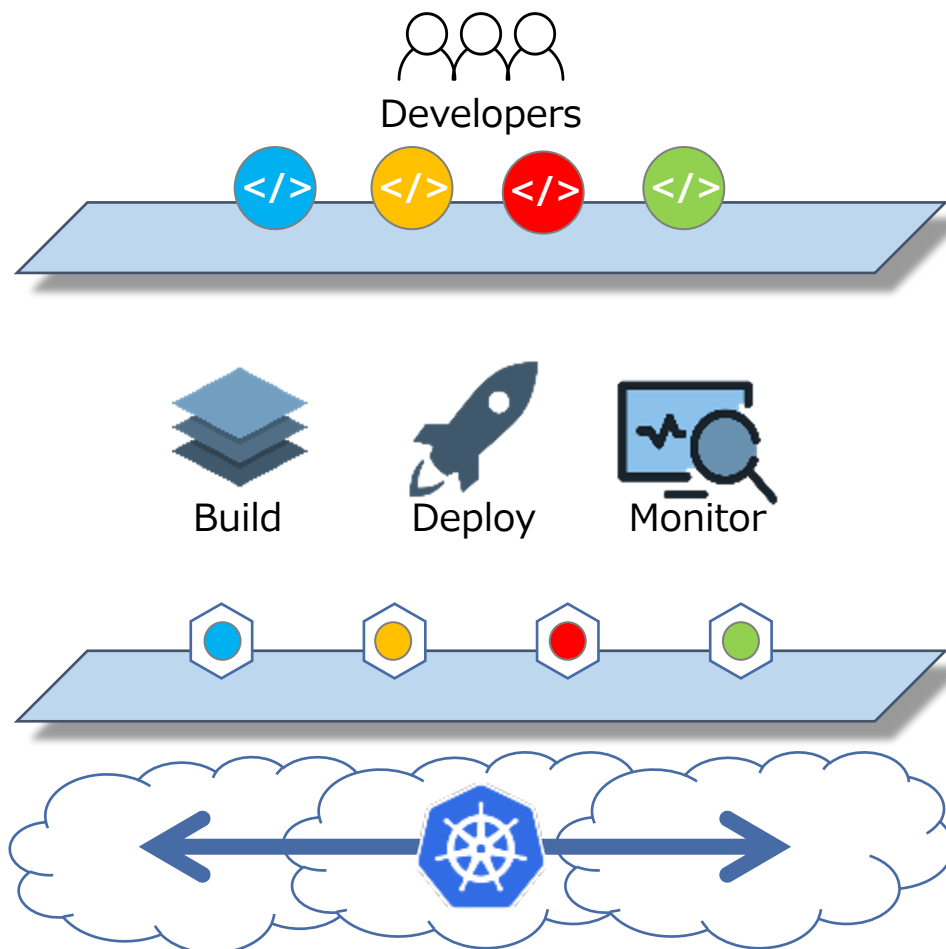


ポータブル

インフラ環境にかかわらず、アプリケーションの自由に移動可能
Vendor Lock-in の懸念が不要



開発運用の効率化



開発チーム

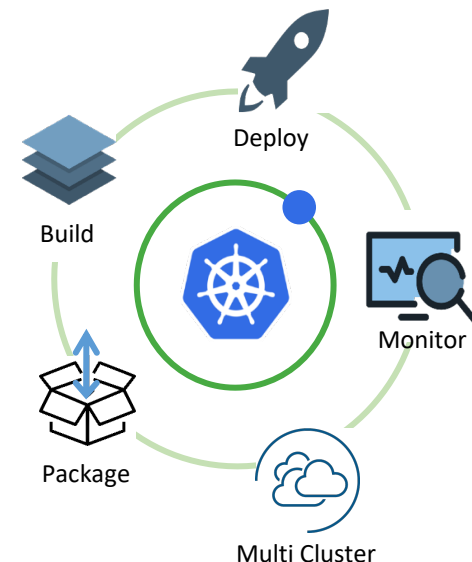
開発により集中
より良いアプリケーションを
迅速に提供

アプリケーション管理

アプリケーション管理の自動化
安定的に管理

クラウドネイティブ

どのような Cloud、Kubernetes環境でも
インフラ環境の違いを考慮不要

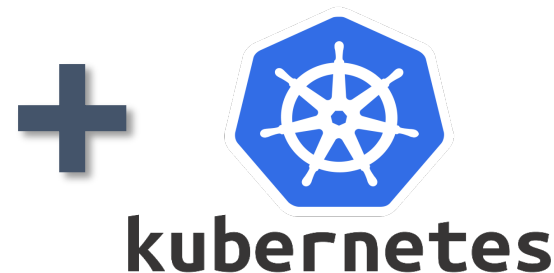




コンテナの本番運用に必要なポイント

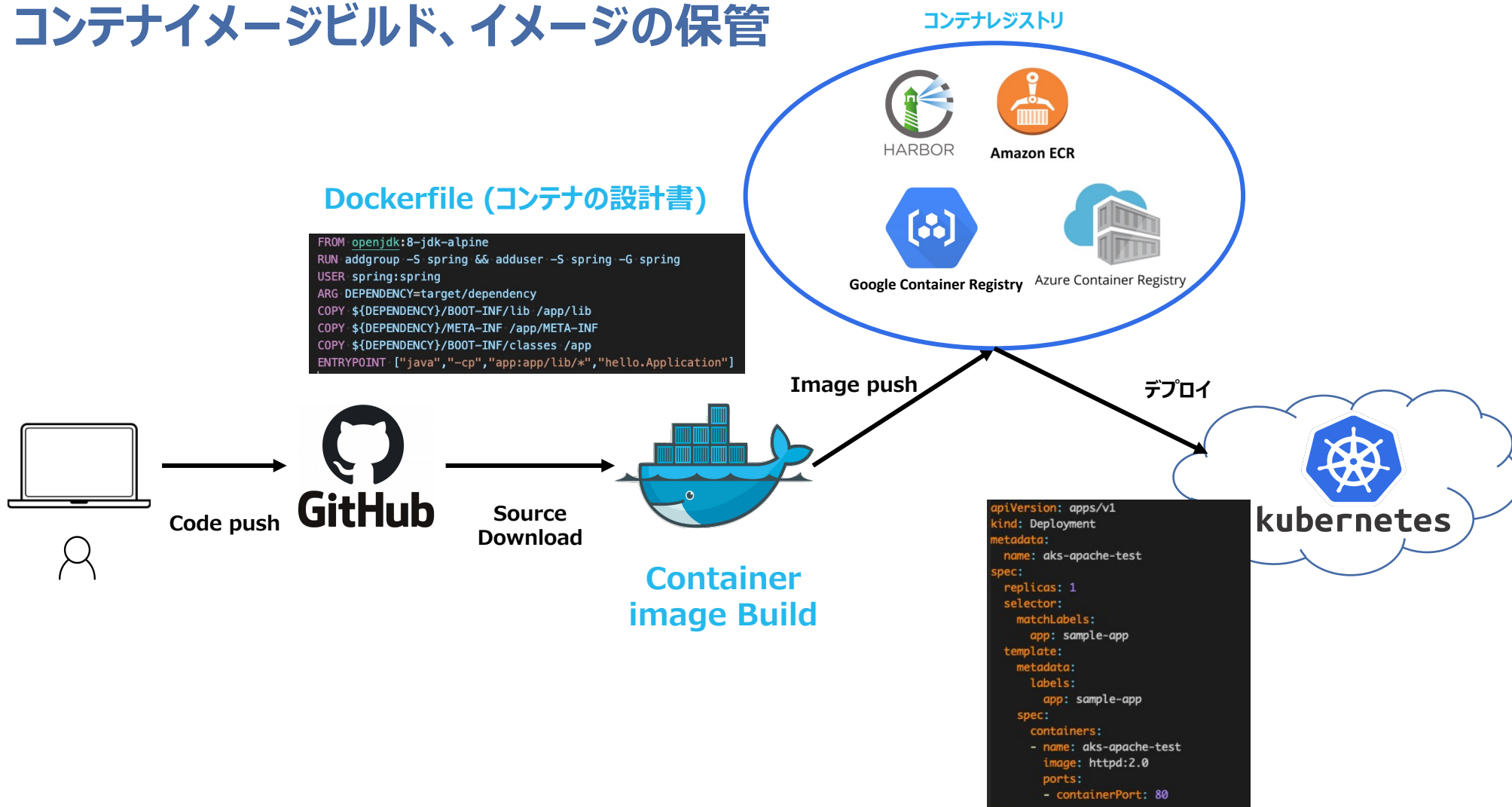
Kubernetes : 課題

1. コンテナイメージのビルド・保管
2. CI / CD 環境
3. コンテナ毎の監視、アラート
4. マニフェストファイル(リソース定義yaml等)の作成・管理
5. 複数のクラスタ管理

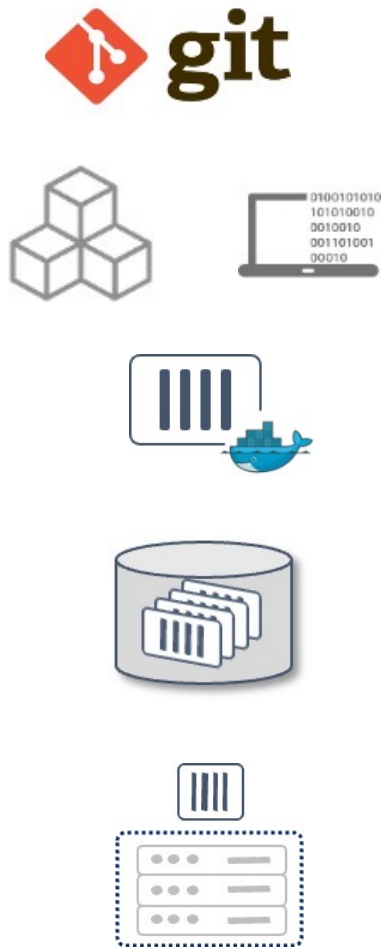


**コンテナを運用する上では上記課題を解決する必要がある。
そのため、Kubernetes以外に多くのソリューションを
組合せて利用しなければならない。**

コンテナイメージビルド、イメージの保管



CI/CD 環境



Application コードダウンロード

単位テスト

イメージビルド

イメージ登録

コンテナ配布/アップデート

パイプライン
Job

コンテナ毎の監視、アラート



Prometheus



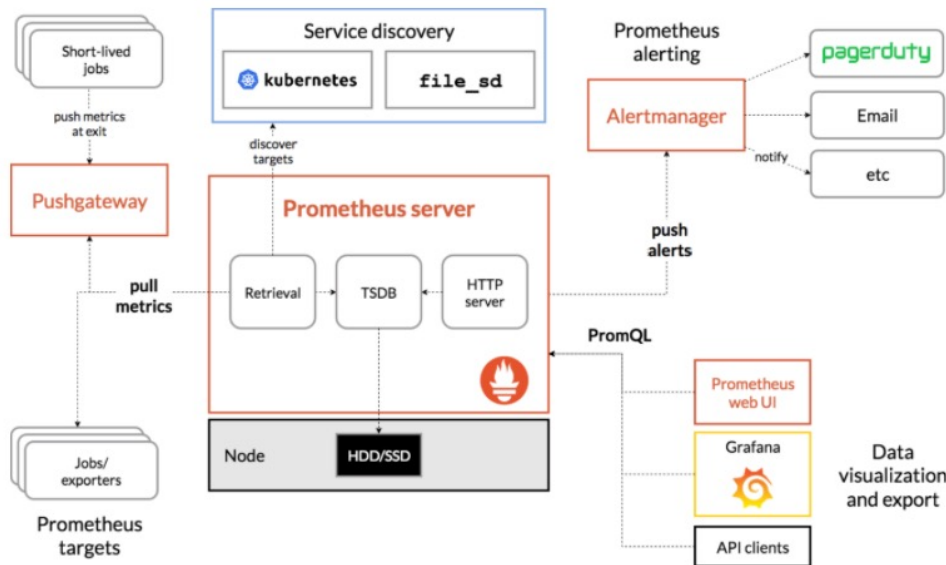
Grafana



Istio



Prometheus architecture



Source: <https://prometheus.io/docs/introduction/overview/>

マニフェストファイル(リソース定義yaml等)の作成・管理

yaml 作成

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: aks-apache-test
spec:
  replicas: 1
  selector:
    matchLabels:
      app: sample-app
  template:
    metadata:
      labels:
        app: sample-app
    spec:
      containers:
        - name: aks-apache-test
          image: httpd:2.0
          ports:
            - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: aks-apache-test
spec:
  type: ClusterIP
  ports:
    - port: 80
  selector:
    app: sample-app
```

コンテナの設定変更のため yaml修正

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: aks-apache-test
spec:
  replicas: 3
  selector:
    matchLabels:
      app: sample-app
  template:
    metadata:
      labels:
        app: sample-app
        environment: test
    spec:
      containers:
        - name: aks-apache-test
          image: httpd:3.0
          ports:
            - containerPort: 80
      nodeSelector:
        disktype: fast
---
apiVersion: v1
kind: Service
metadata:
  name: aks-apache-test
spec:
  type: ClusterIP
  ports:
    - port: 80
  selector:
    app: sample-app
```

多数の yaml を管理しなければいけなくなる。

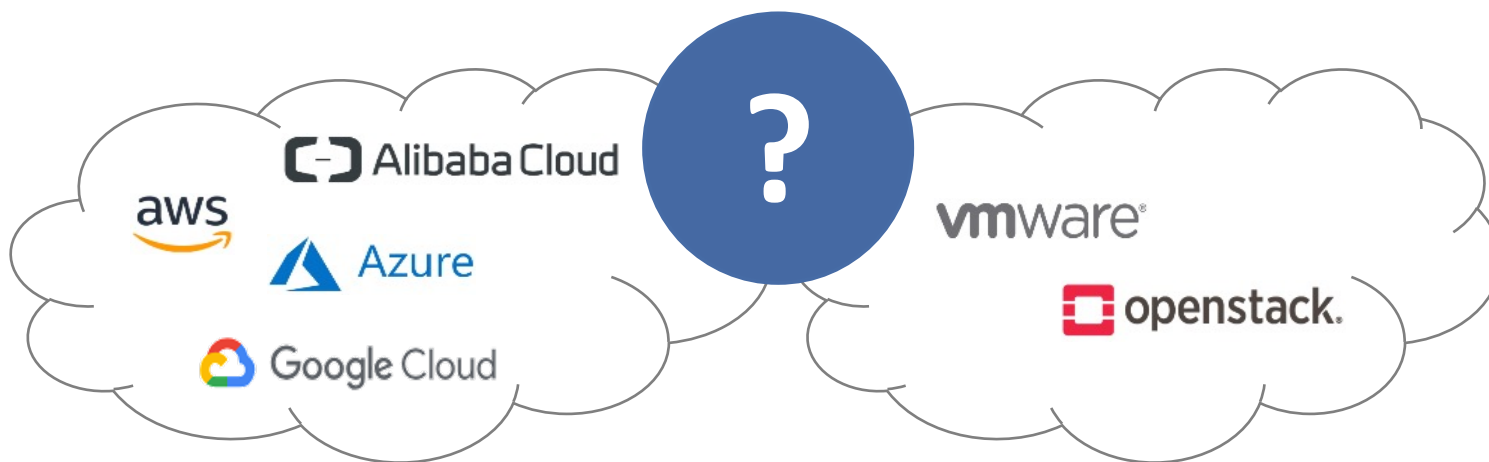
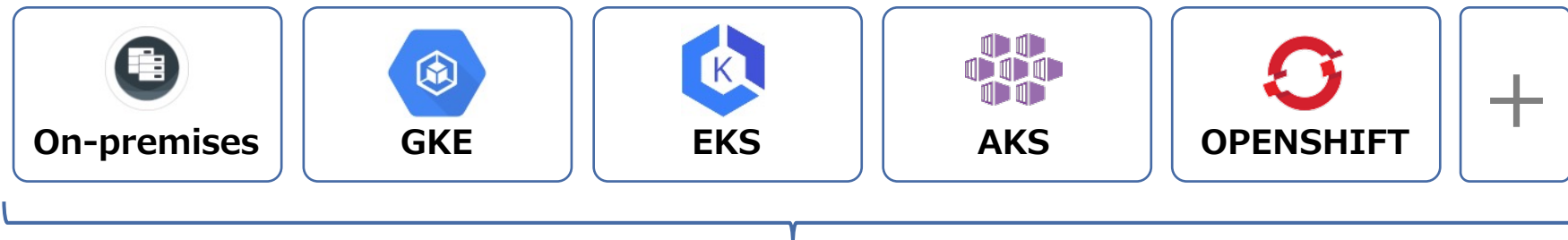
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: aks-apache-test
spec:
  replicas: 1
  selector:
    matchLabels:
      app: sample-app
  template:
    metadata:
      labels:
        app: sample-app
    spec:
      containers:
        - name: aks-apache-test
          image: httpd:2.0
          ports:
            - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: aks-apache-test
spec:
  type: ClusterIP
  ports:
    - port: 80
  selector:
    app: sample-app
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: aks-apache-test
spec:
  replicas: 1
  selector:
    matchLabels:
      app: sample-app
  template:
    metadata:
      labels:
        app: sample-app
    spec:
      containers:
        - name: aks-apache-test
          image: httpd:2.0
          ports:
            - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: aks-apache-test
spec:
  type: ClusterIP
  ports:
    - port: 80
  selector:
    app: sample-app
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: aks-apache-test
spec:
  replicas: 1
  selector:
    matchLabels:
      app: sample-app
  template:
    metadata:
      labels:
        app: sample-app
    spec:
      containers:
        - name: aks-apache-test
          image: httpd:2.0
          ports:
            - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: aks-apache-test
spec:
  type: ClusterIP
  ports:
    - port: 80
  selector:
    app: sample-app
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: aks-apache-test
spec:
  replicas: 1
  selector:
    matchLabels:
      app: sample-app
  template:
    metadata:
      labels:
        app: sample-app
    spec:
      containers:
        - name: aks-apache-test
          image: httpd:2.0
          ports:
            - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: aks-apache-test
spec:
  type: ClusterIP
  ports:
    - port: 80
  selector:
    app: sample-app
```

複数のクラスタ管理



Multiple Cloud / Hybrid Cloud



コンテナ利用を容易にするソリューション

OSSを自ら構成、
運用する



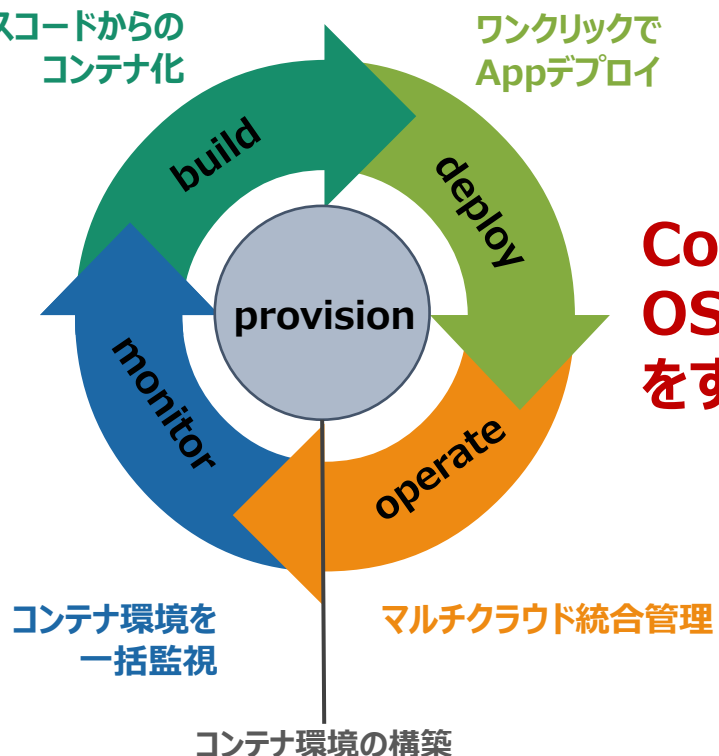
商用コンテナプラットフォームで
運用する





Cocktail Cloud

All-in-One コンテナ管理プラットフォーム
Web UIベースで簡単に利用可能



**Cocktail Cloudは、自社で開発したものと
OSSを組み合わせでコンテナの本番運用に必要なものを
すべてご用意！！**

The diagram illustrates the relationship between Platform, Observability and Analysis, and Monitoring. A large blue arrow points from the 'Platform' section to the 'Monitoring' section.

Platform

Certified Kubernetes - Distribution

Certified Kubernetes - Hosted

Observability and Analysis

Monitoring

Logging

Tracing

Chaos Engineering

A grid of 100 logos representing various companies, likely sponsors or partners of the event. The logos are arranged in 10 rows and 10 columns. Some recognizable logos include Intel, Oracle, SAP, Microsoft, and others.

1. Cocktail Cloudパートナー

License (Enterprise、オンラインサブスクリプション)
Distributor

2. OEM、ODM

Enterprise版、Online版を御社ブランドで提供

3. 技術サポート

コンテナ化
MSA (マイクロサービスアーキテクチャ) コンサルティング
OSS技術サポート



Cocktail Cloud

ありがとうございました。

NAMU TECH JAPAN

<https://namutech.co.jp>

cocktail@namutech.co.jp