



# 私が半年間で学んだ Kubernetesの3つのこと

日本オラクル株式会社  
クラウド・プラットフォーム本部  
Middlewareソリューション部

# 自己紹介



 @cotoc88

- 茂 こと (しげる こと)
- 日本オラクルのソリューションエンジニア
  - Oracle Database 数年
  - Docker/Kubernetes 6ヶ月くらい
  - Vitess/MySQL 3週間くらい
- キーボードはREALFORCE (光らない)



撮影: JapanContainerDays実行委員会

Kubernetesを  
さわったことがありますか？



# Kubernetesとは

- Kubernetesはコンテナ・オーケストレーションツール
  - Google発のオープンソース
- 2014年6月にローンチされ、2015年7月にCloud Native Computing Foundation (CNCF)に開発が移管
  - CNCFは著名な開発者、エンドユーザ、大手クラウドプロバイダ、ベンダーが参加しており、現在はCNCFが主体となり中立的な立場で開発が進められています



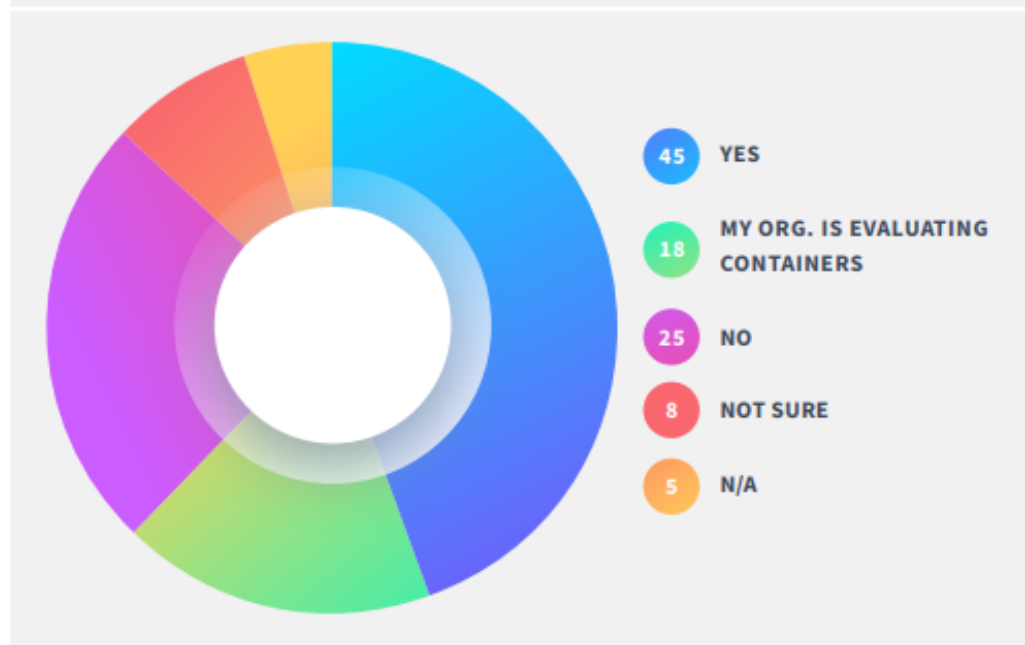
# 私が半年間で学んだKubernetesの3つのこと

1. Kubernetes、なんだかすごそうだ！
  - Kubernetes登場の背景・メリット
2. Kubernetes、なんだか大変そうだ！
  - 最初にぶつかる壁
3. Kubernetesで広がる世界！

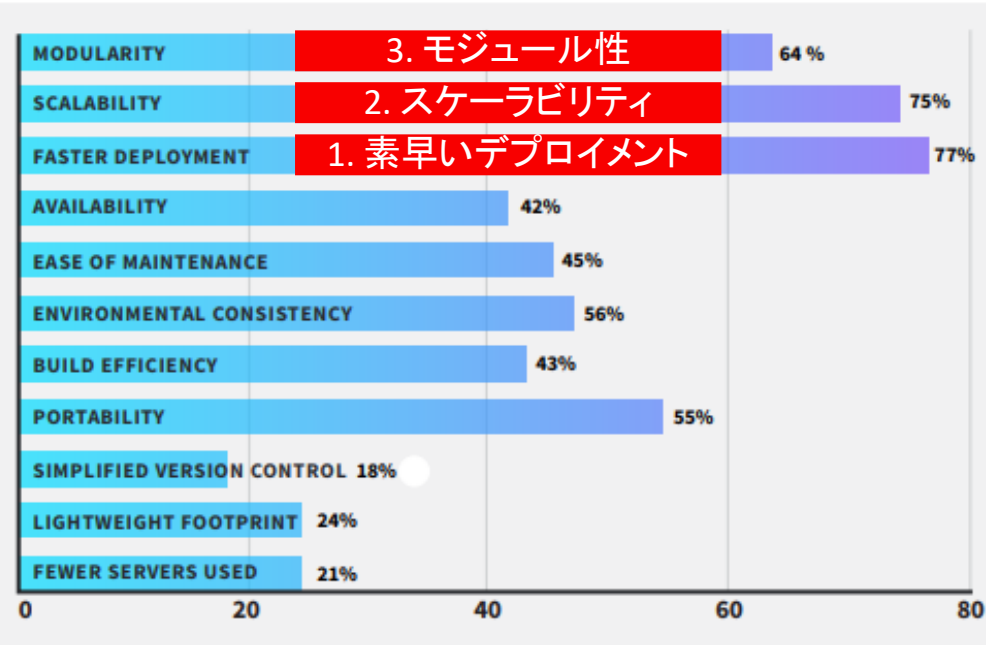
# Container/Kubernetesって なぜ流行ってるの？

# コンテナは流行っているらしい 世界中の企業で採用が進んでいる

**GRAPH 01.** Does your organization currently use container technologies?



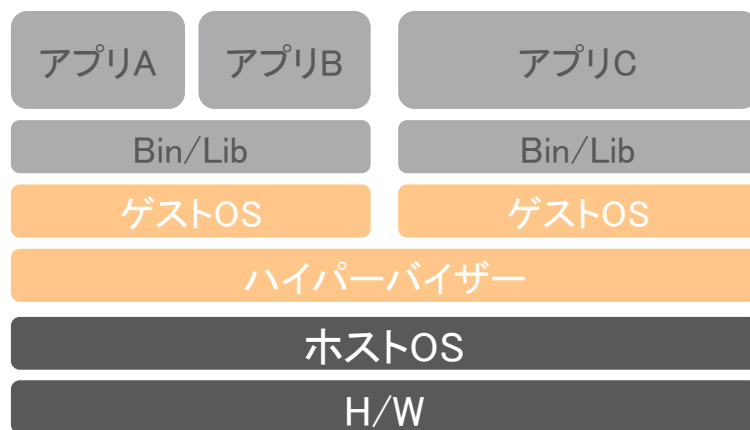
**GRAPH 02.** What benefits do container technologies offer your organization?



# コンテナってなに...?

## コンテナ = 軽量・高速な仮想マシン

- コンテナ型仮想化という技術で実現されている、仮想マシンの一種
- 軽量・高速・高密度にアプリケーションを実行させることが可能



従来型の仮想化



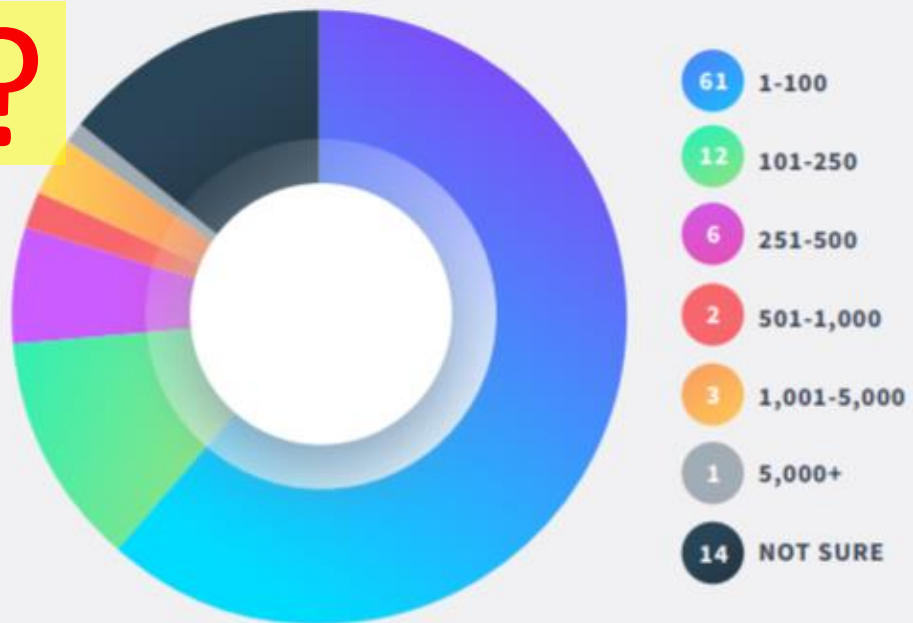
コンテナ型仮想化

- ✓ 軽容量
  - カーネル部分をホストOSと共有するため軽量
  - 高い可搬性
- ✓ 高速
  - 起動が早い
  - オーバーヘッドが少なくパフォーマンスが高い
- ✓ モジュール化
  - 他の構成要素に及ぼす影響が最小限



5000+!?

**GRAPH 06.** How many containers is your organization running in production?



# どうやって面倒を見れば...



軽くて集約できるものだからついつい数が増えちゃう

# コンテナ・オーケストレーター

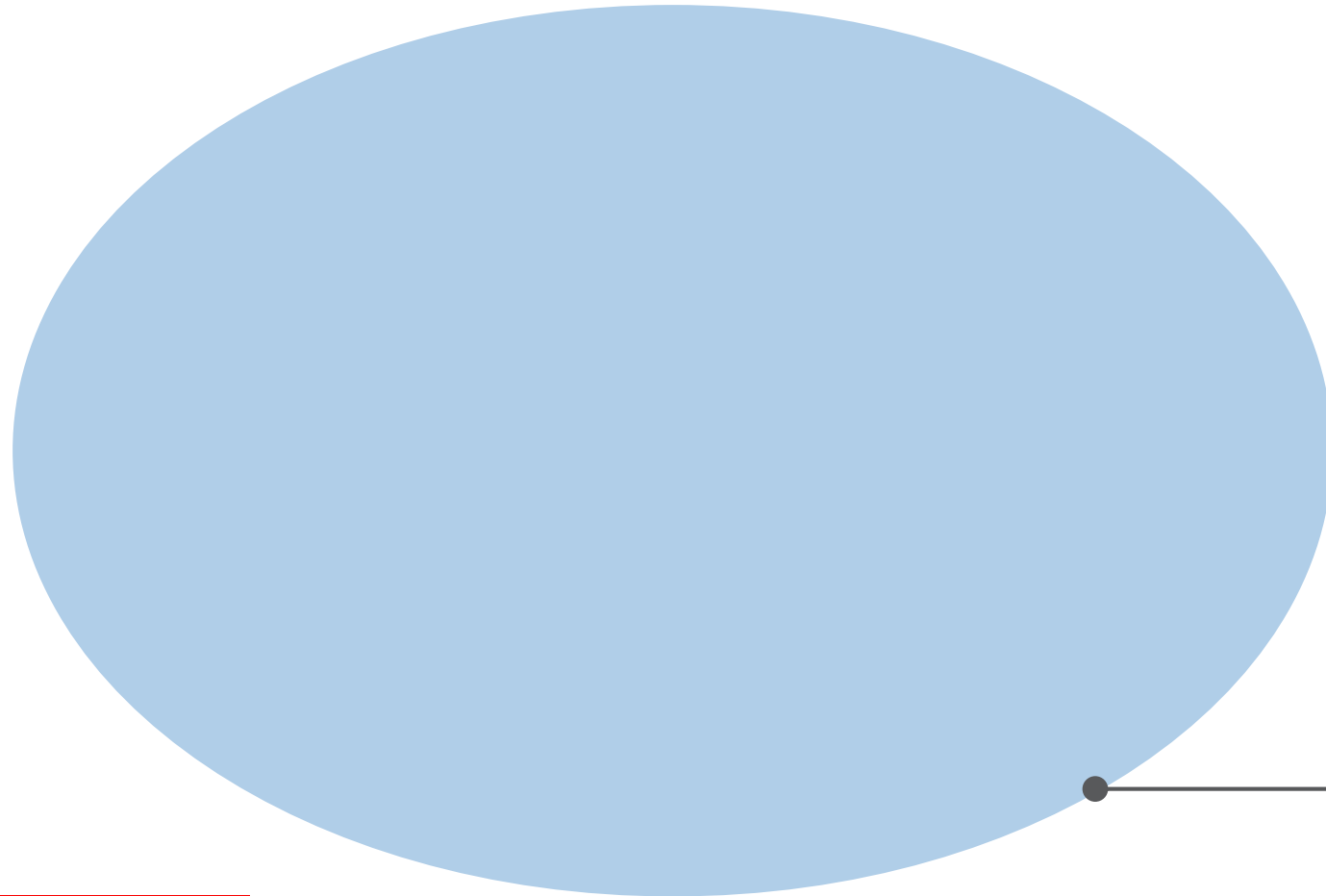
## 大量のコンテナを管理・運用するプラットフォーム

- 複数コンテナのデプロイ、スケーリング等を自動管理するプラットフォーム
  - 複数ホストにコンテナをデプロイ  
(HWを意識しない)
  - 手動／自動でスケーリング
  - 複数コンテナをまとめて制御
  - コンテナの死活監視  
障害時のコンテナ再立ち上げ
  - クラスタ内／外のネットワーク  
アクセスの管理
- Kubernetesがデファクト・スタンダードになりつつある



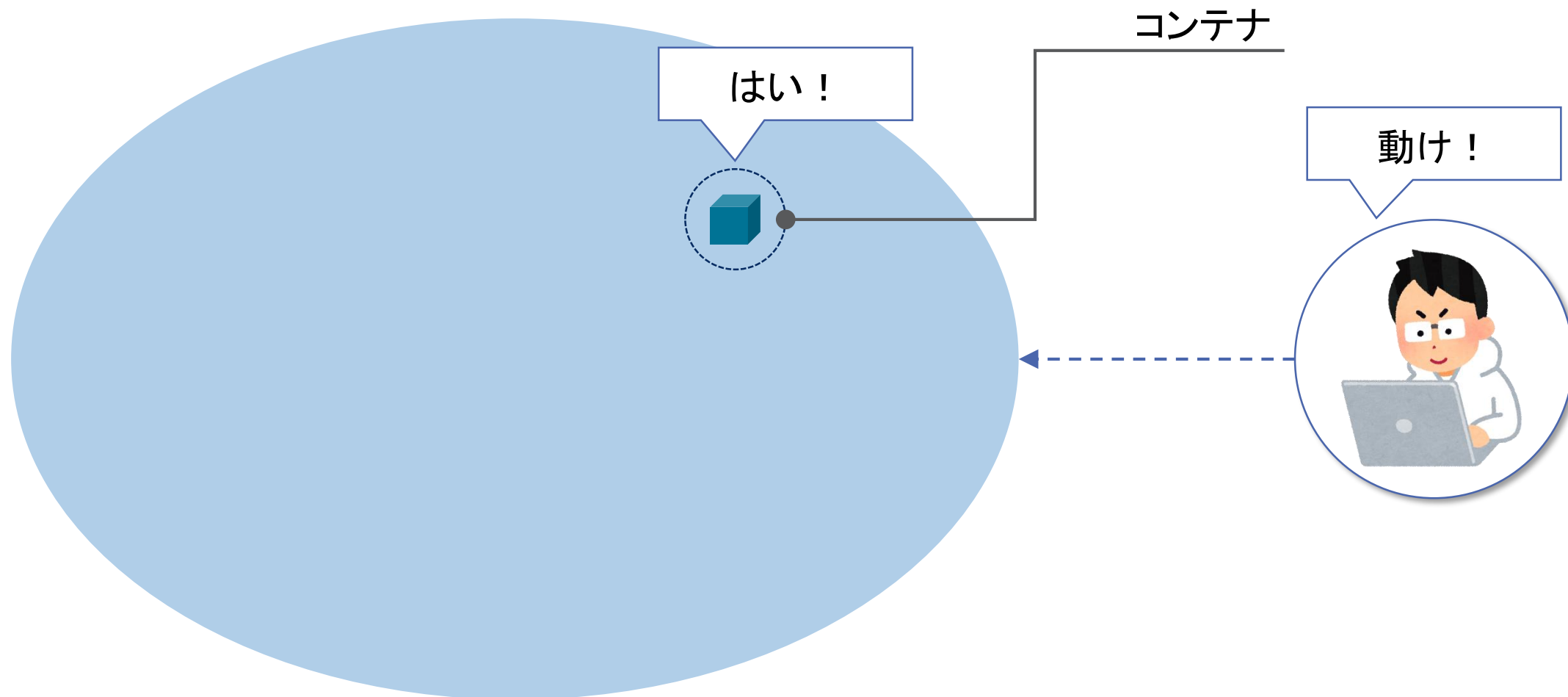
**kubernetes**

# コンテナオーケストレーションが実現すること

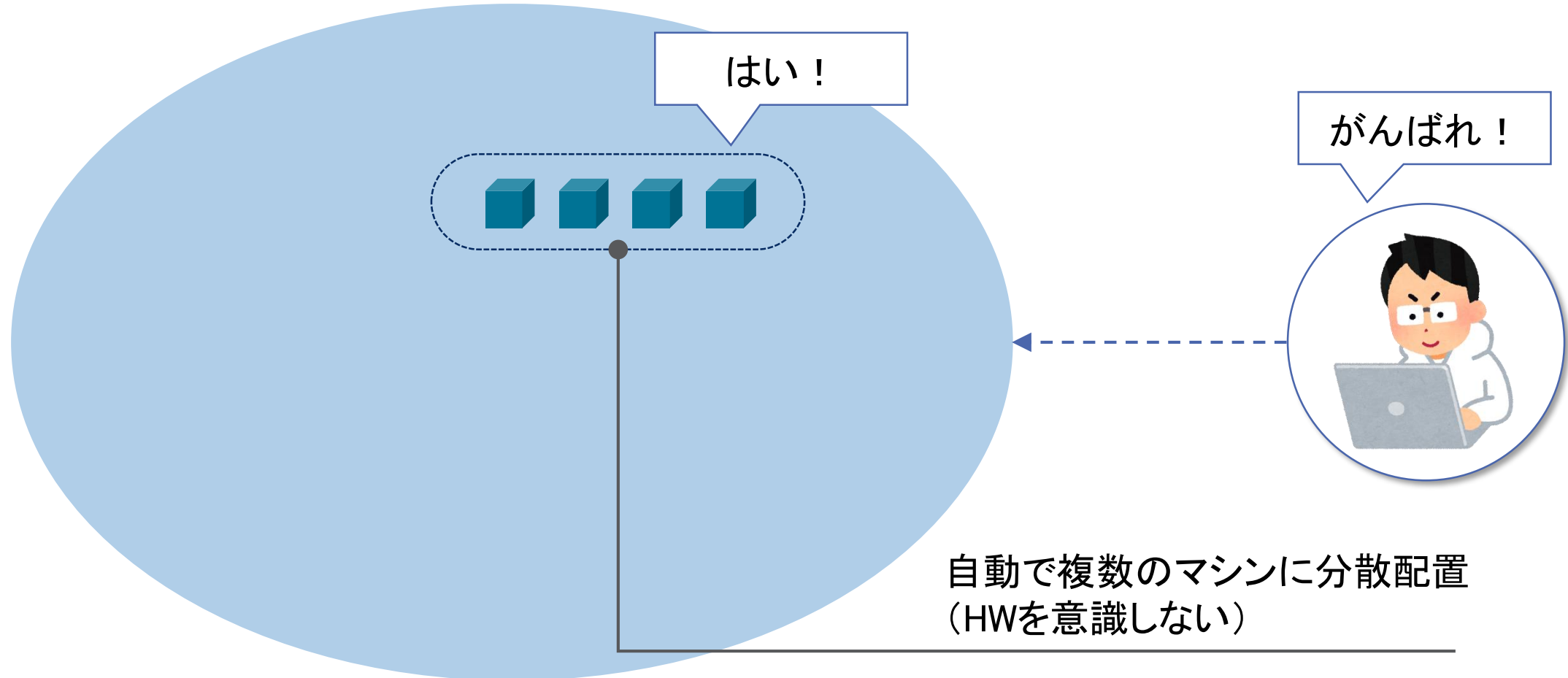


大きなリソースのプール

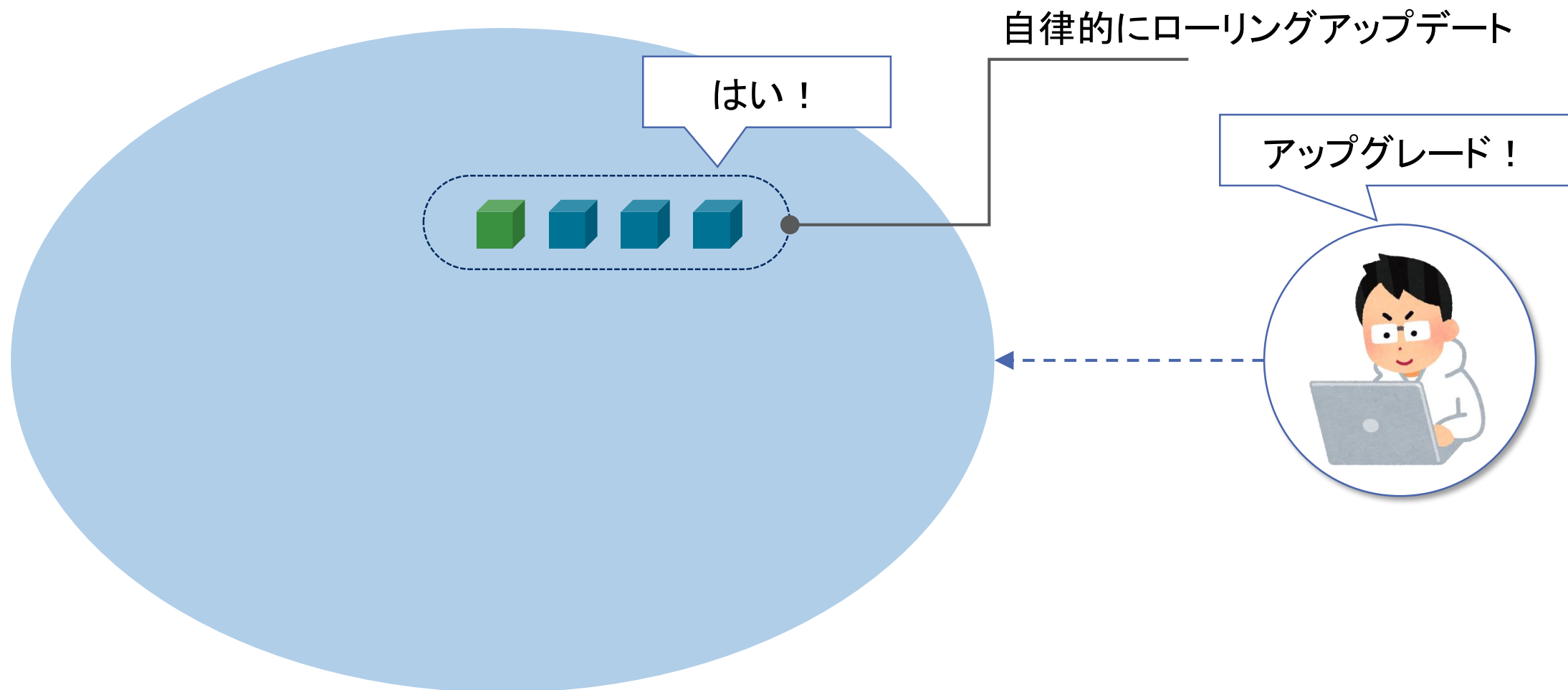
# コンテナオーケストレーションが実現すること



# コンテナオーケストレーションが実現すること

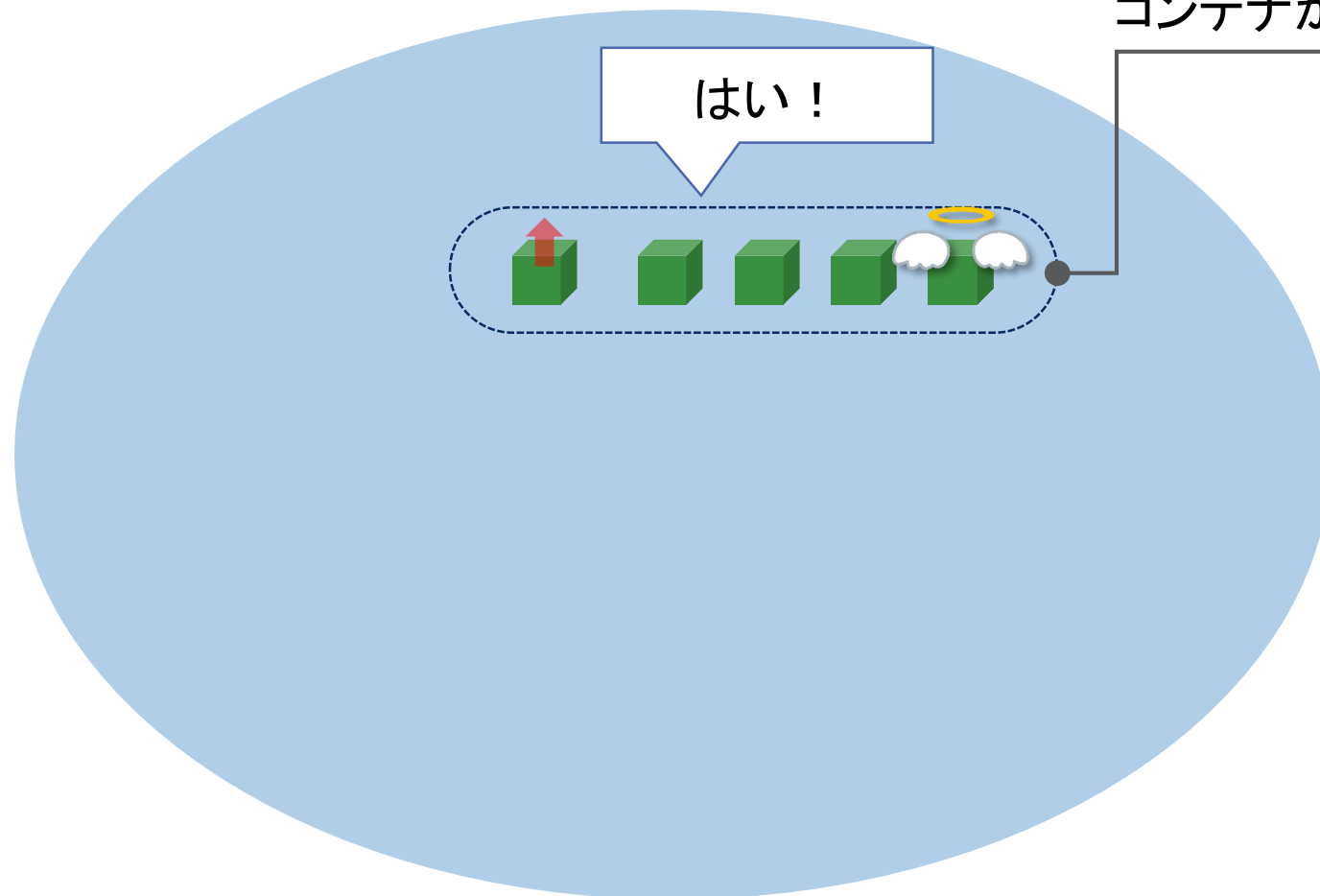


# コンテナオーケストレーションが実現すること



# コンテナオーケストレーションが実現すること

コンテナが落ちたら自動で新たに立ち上げる



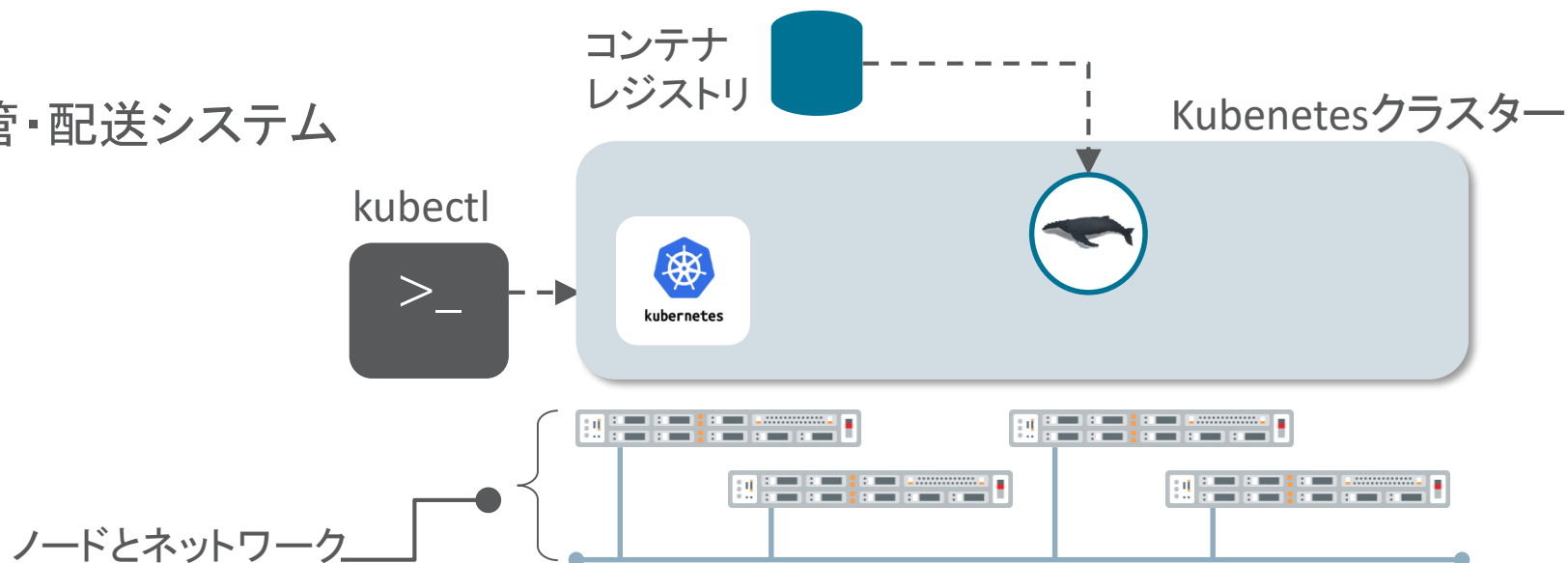


なんだかすごそうだ！

よし、  
さっそくやってみよう！

# Kubernetes利用環境の主要な構成要素

- 大きく分けて、**Kubernetesクラスター**、**kubectl**、**コンテナレジストリ**で構成
  - kubectl
    - Kubernetesクラスターの管理操作を行うためのコマンドラインツール
  - Kubernetesクラスター
    - クラスター本体、実際のワークロードが可動する場所
  - コンテナレジストリ
    - コンテナイメージの保管・配送システム



# Kubernetes上でコンテナが動作する様子

- Kubernetesでアプリケーションを動かすときの最も基本的な構成

- Pod / Container

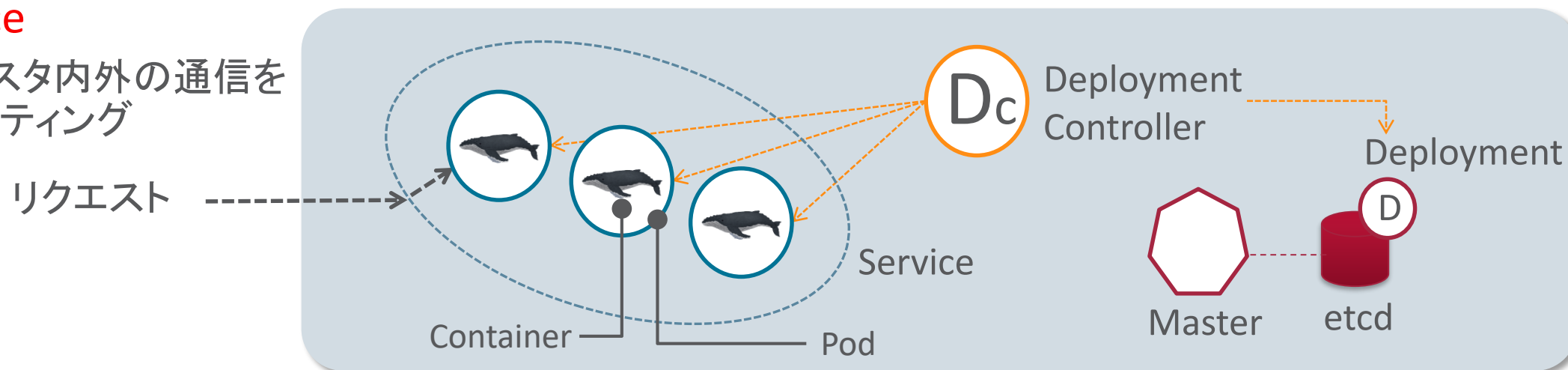
- 複数のコンテナを束ねる単位

- Deployment / Deployment Controller

- Podの状態(レプリカ数など)を定義するデータ/制御プロセス

- Service

- クラスタ内外の通信をルーティング



Kubernetesクラスタ

# Kubernetesは宣言的オペレーション

## 最終的にどうあるべきかということを宣言的に記述

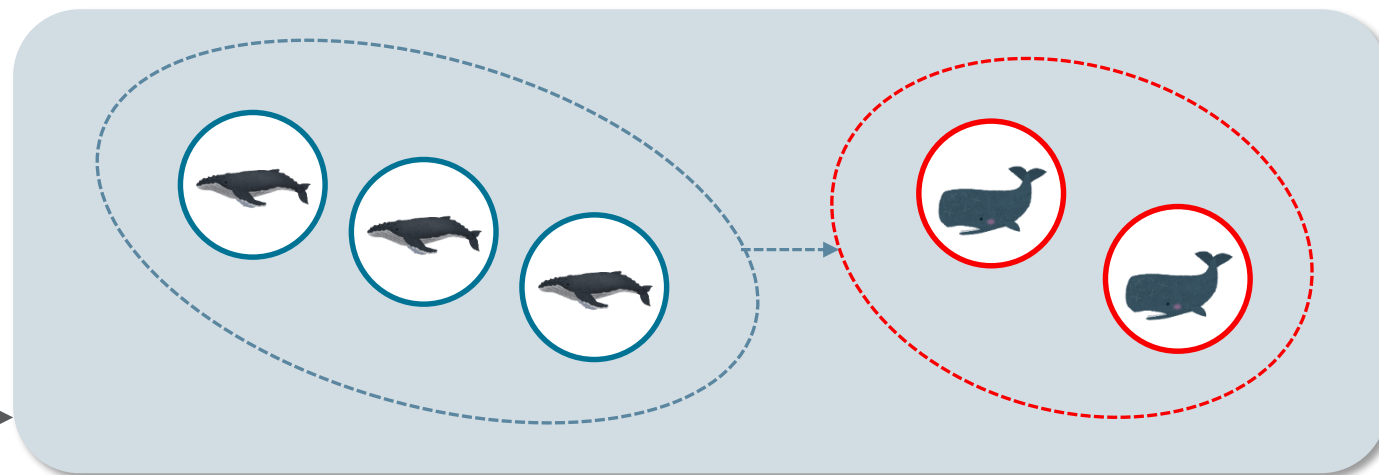
- クラスタに配備する要素を全てマニフェスト・ファイル(=コード)に記述
  - 環境構成の変更内容をバージョン管理し、追跡可能にする
  - 運用オペレーションはコードの変更によって実施し、作業を簡素化する

- ✓ アプリ x 3
- ✓ データストア x 2
- ✓ ネットワークはアプリ→ データストアを許可
- ✓ ...etc



Kubernetesに適用

manifestファイル



Kubernetesクラスタ

# マニフェスト・ファイルの例

- yaml形式で記述するのが主流
- 右例では以下のような構成を定義
  - 3つのPodで冗長化
  - コンテナイメージは、  
cotoc/cowweb:v1.0
  - ポート8080を公開してアプリに流す
  - ...

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: cowweb
5  spec:
6    replicas: 3
7    selector:
8      matchLabels:
9        app: cowweb
10   strategy:
11     type: Recreate
12   template:
13     metadata:
14       labels:
15         app: cowweb
16         version: v1.0
17     spec:
18       containers:
19         - name: cowweb
20           image: cotoc/cowweb:v1.0
21           ports:
22             - name: api
23               containerPort: 8080
24   ... (以下略) ...
```

# マニフェスト・ファイルの運用

## 本番環境で運用するためには様々な考慮が必要

### Agenda

- コンテナのライフサイクル（起動時）
- コンテナのライフサイクル（停止時）
- ヘルスチェック
- メンテナンスとアップデート
- スケジューリング
- リソースの割り当てと基準
- カーネルパラメータのチューニング
- インターネットからのアクセス制御

Kubernetes完全ガイド impress top gearシリーズ



<https://speakerdeck.com/masayaaoiyama/jkd1812-prd-manifests>

 @amsy810

# 単純なアプリケーションでも3桁・・・

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: wordpress
5  labels:
6    app: wordpress
7  spec:
8  ports:
9    - port: 80
10 selector:
11   app: wordpress
12   tier: frontend
13 type: LoadBalancer
14 ---
15
16
17 apiVersion: v1
18 kind: PersistentVolumeClaim
19 metadata:
20   name: wp-pv-claim
21 labels:
22   app: wordpress
23 ---
24
25 apiVersion: v1
26 kind: Service
27 metadata:
28   name: wordpress-mysql
29 labels:
30   app: wordpress
31 spec:
32 ports:
33   - port: 3306
34 selector:
35   app: wordpress
36   tier: mysql
37 clusterIP: None
38 ---
39
40 apiVersion: v1
41 kind: PersistentVolumeClaim
42 metadata:
43   name: mysql-pv-claim
44 labels:
45   app: wordpress
46 ---
47
48 apiVersion: apps/v1
49 kind: Deployment
50 metadata:
51   name: wordpress
52 labels:
53   app: wordpress
54 spec:
55   replicas: 1
56   selector:
57     matchLabels:
58       app: wordpress
59       tier: frontend
60   template:
61     metadata:
62       labels:
63         app: wordpress
64         tier: frontend
65     spec:
66       accessModes:
67         - ReadWriteOnce
68       resources:
69         requests:
70           storage: 20Gi
71       volumeMounts:
72         - mountPath: /var/www/html
73           name: wordpress-persistent-storage
74           persistentVolumeClaim:
75             claimName: wp-pv-claim
76 --- 以下略 ---
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
```



# なんだか大変そうだ...

通称:YAMLの壁

# でも、その先に待つのは！

クラスター内/外のネットワーク  
アクセスの管理

複数ホストにコンテナをデプロイ  
(HWを意識しない)

障害時のコンテナ再立ち上げ



手動／自動でスケーリング

複数コンテナをまとめて制御

## kubernetes

コンテナの死活監視

# Cloud Nativeプロダクトと組み合わせで最高のDevOpsを！



# DevOpsを実現するためのCloud Nativeプロダクト群

- コンテナのCI/CD
  - 自動でビルド,テスト,デプロイを実施
- Wercker
  - 2012年オランダ発、現在はOracleが提供しているCIツール
- Spinnaker
  - Netflixが開発したCDツール
    - カナリーリリースの自動解析など
- 運用・監視
  - k8sクラスターやPodの監視
- Prometheus
  - システムのモニタリング・アラートのためのソフトウェア
  - K8sとの連携機能あり
- Grafana
  - メトリックの可視化ツール
    - 豊富なグラフテンプレート



# スケーラブルなアーキテクチャに不可欠なインフラ

- サービスメッシュ
  - マイクロサービス間に張りめぐされたメッシュ状のトラフィックや経路の制御
- Istio
  - Pod間の通信にプロキシ(Envoy)を挟むことでトラフィックのモニタリングやコントロールを行う
  - アプリの変更なし
- 分散データベース
  - k8s上でスケーラブルな分散データベース
- Vitess
  - シャーディング技術を用いることでMySQLをスケーラブルに扱うことができるクラスタリングシステム



# 本日お話した3つのこと

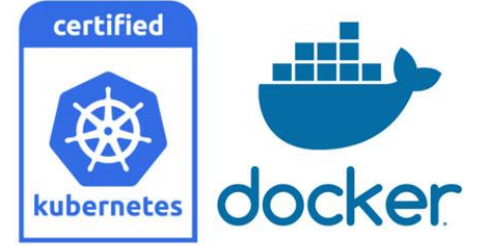
1. Kubernetes、なんだかすごそうだ！
  - Kubernetes登場の背景・メリット
2. Kubernetes、なんだか大変そうだ！
  - 最初にぶつかる壁
3. Kubernetesで広がる世界！

**YAMLの壁を乗り越えて、KubernetesとCloud Nativeプロダクトを活用して最高の開発/プロダクション環境を実現しよう！**

さいごのお知らせ

# Containers on Oracle Cloud

OracleもKubernetesのサービスを提供しています



- コンテナベースのマイクロサービスおよびサーバーレス・アプリケーションを構築、配置、管理するためのフルマネージド・サービスを提供します



- **Oracle Container Engine for Kubernetes (OKE)**
  - マネージドKubernetesサービス



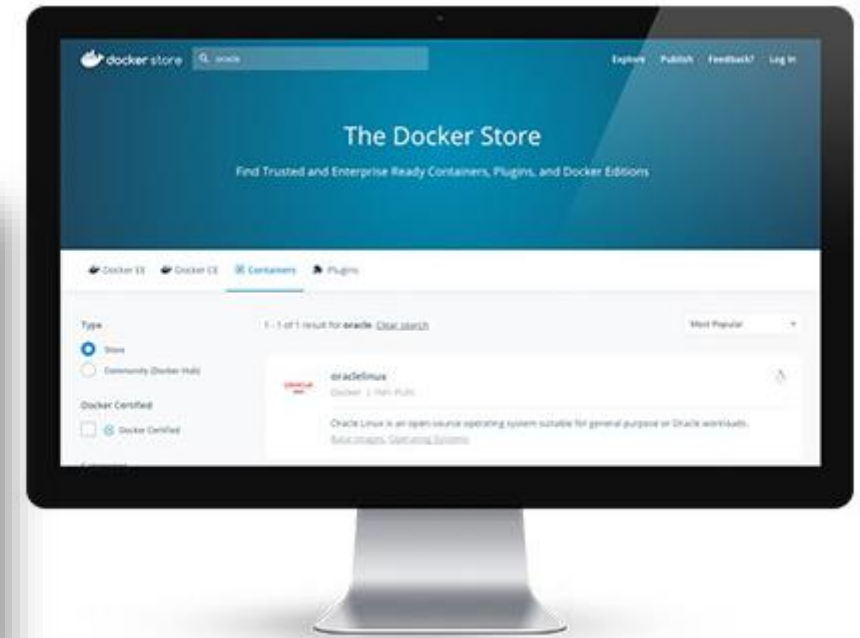
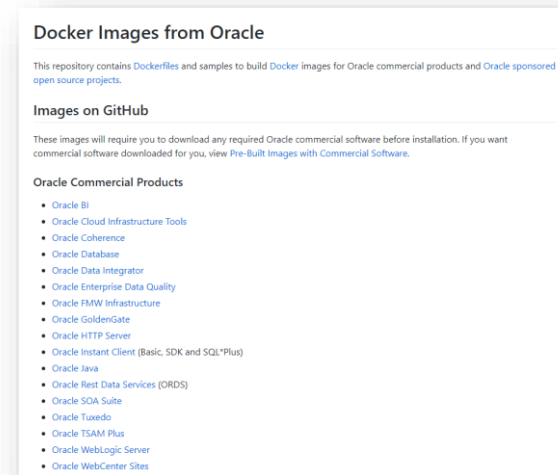
- **Oracle Cloud Infrastructure Registry (OCIR)**
  - マネージドなコンテナレジストリ



# Oracle on Docker Store

## Oracleの製品がDockerで簡単に動かせるようになりました

- Docker StoreでOracle Database、Oracle WebLogic Server、Oracle Coherence、Oracle Javaなどの主要製品を公式なDockerイメージとして利用できるようになりました
  - OTN開発者ライセンスで無償利用可能



<https://store.docker.com/publishers/oracle>

# Container/Cloud Nativeの世界はコミュニティが活発 もっと詳しくCloud Nativeな技術を共有し学び会える場



※ <https://containerdays.jp>

## Community

[Docker Tokyo](#)

[Kubernetes Meetup Tokyo](#)

[Rancher JP](#)

[Japan OpenStack User Group](#)

[Container SIG](#)

[Mesos User Group Tokyo](#)

[PaaS JP](#)

[Bluemix Users Group](#)

[Serverless Community \(JP\)](#)

[GitLab Tokyo](#)

[Cloud Native Meetup Tokyo](#)

[Cloud Native Developers JP](#)

[Cloud Native Deep Dive](#)

[NoOps Japan](#)

※来年はCloudNative Daysに名称が変更



## cndjp - Cloud Native Developers JP

- **Cloud NativeなOSSスタック**を取り上げる勉強会シリーズ、およびコミュニティ
- OSS中心(最近はOSSとも限らない傾向)
- 楽しく学ぶ、深く学ぶ

# Oracle Cloud Hangout Cafe (OchaCafe)

## カフェで一息つきながらしっかり学べる、Cloud Nativeな勉強会シリーズ

12月  
20

OchaCafe#1 - Kubernetesで作るコンテナベース  
CI☆CDの夕べ

Oracle Cloud Hangout Cafe #1 増席しました!

主催: Oracle Code



ハッシュタグ: #ochacafe

	テーマ	キーワード
12/20	コンテナを使ったCI/CD	Wercker,Kubernetes,Spinnaker
1月開催予定	Microservicesの運用・管理	Grafana,Prometheus,istio,
2月開催予定	Microservicesな Javaアプリケーション	Helidon,MicroProfile,GraalVM
3月開催予定	明日から使える Enterprise Blockchain	Hyperledger Fabric
4月開催予定	避けては通れない 認証/認可	OAuth 2.0,OpenID Connect
5月開催予定	OpenAPIのエコシステム	OpenAPI

※発表順序・テーマは変更する可能性があります。

#ochacafe

<https://ochacafe.connpass.com/event/108009/>

## Safe Harbor Statement

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

ORACLE®