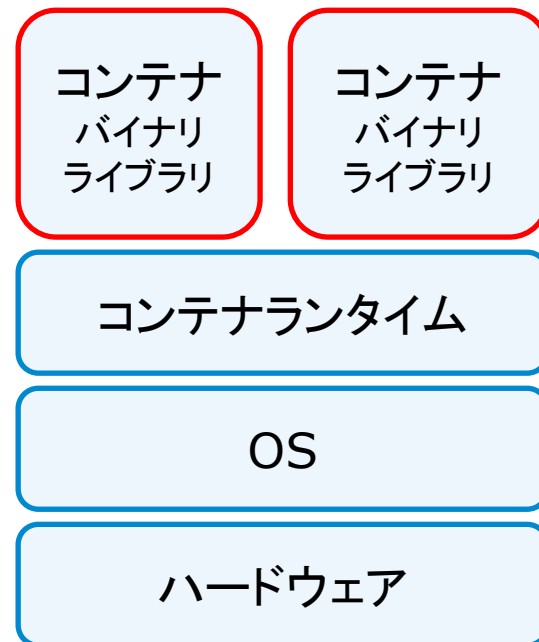
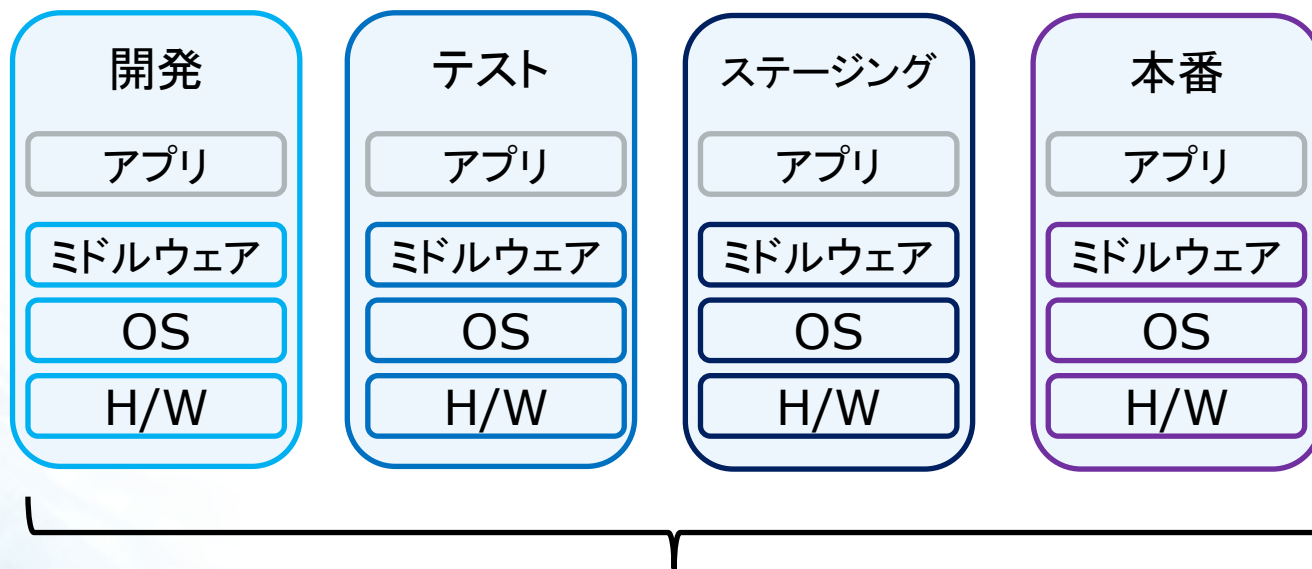


OSS のコンテナ オーケストレーションツール Kubernetes のご紹介

- ホストOS上に論理的な区画(コンテナ)を作ってアプリケーションを動作させるために必要なプログラムなどをまとめたもの
- ホストOS上のリソースをコンテナで共有して利用
- サーバ仮想化に対してオーバヘッドが少なく
- 複数のコンテナを組み合わせて一つのアプリケーションを作る
マイクロサービスとの相性がよい

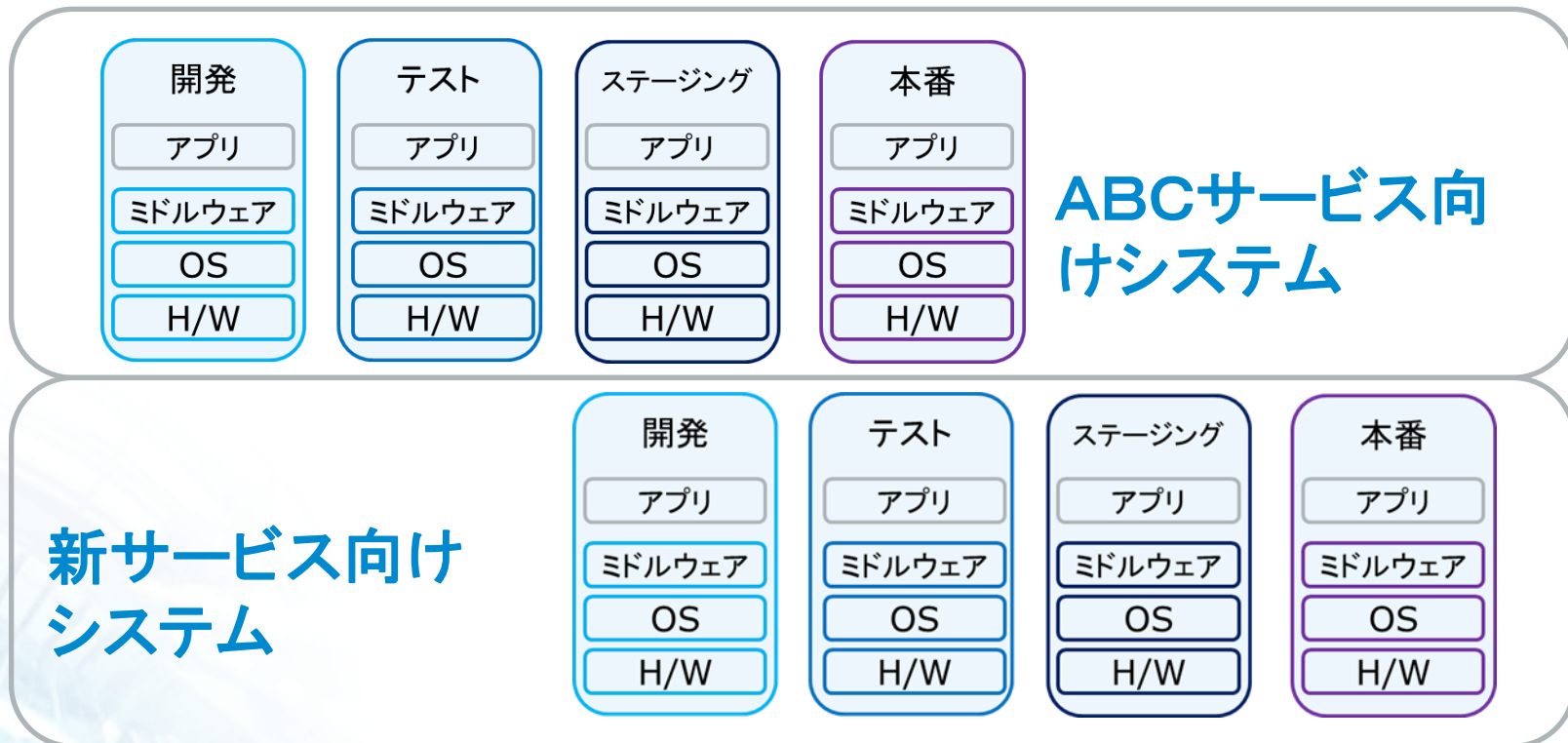


- 開発環境・テスト環境・ステージング環境・本番環境など、複数の環境でインフラ・ミドルウェアの違いに依存する問題が発生しやすい



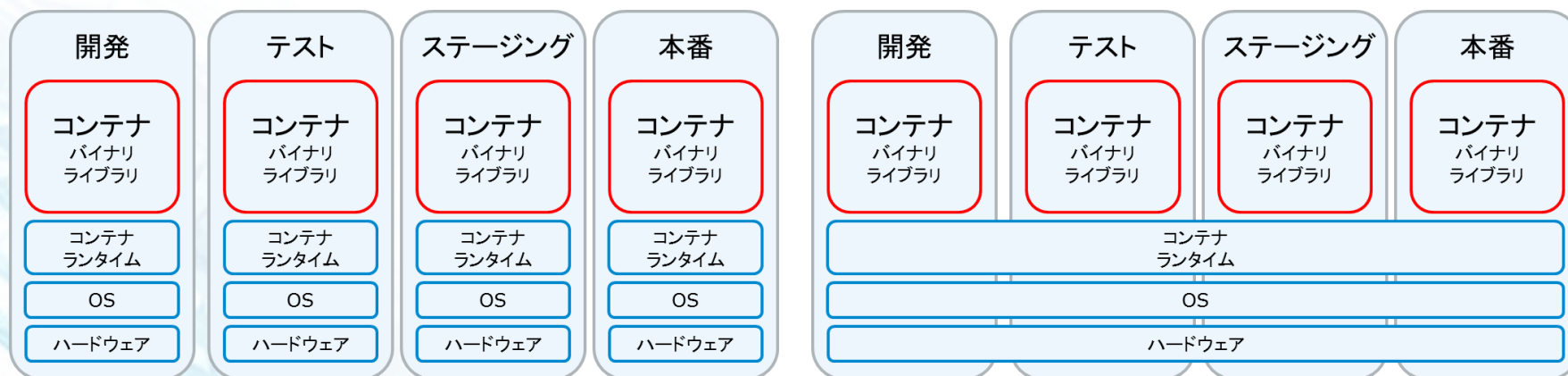
同じ環境??

- サービスが増えると開発環境・テスト環境・ステージング環境・本番環境も増える・・



インフラの準備時間とテスト実施のための時間がかかって
 しまって迅速な開発・サービス提供ができなくなってしまう

- コンテナランタイム上で同じイメージを動かすことで環境の違いのリスクを減らすことができる
- コンテナ間は独立しているので、同じホスト上でも動作させることができる
- サーバ仮想化に対してオーバヘッドが少ないため、サーバの利用効率が上昇する



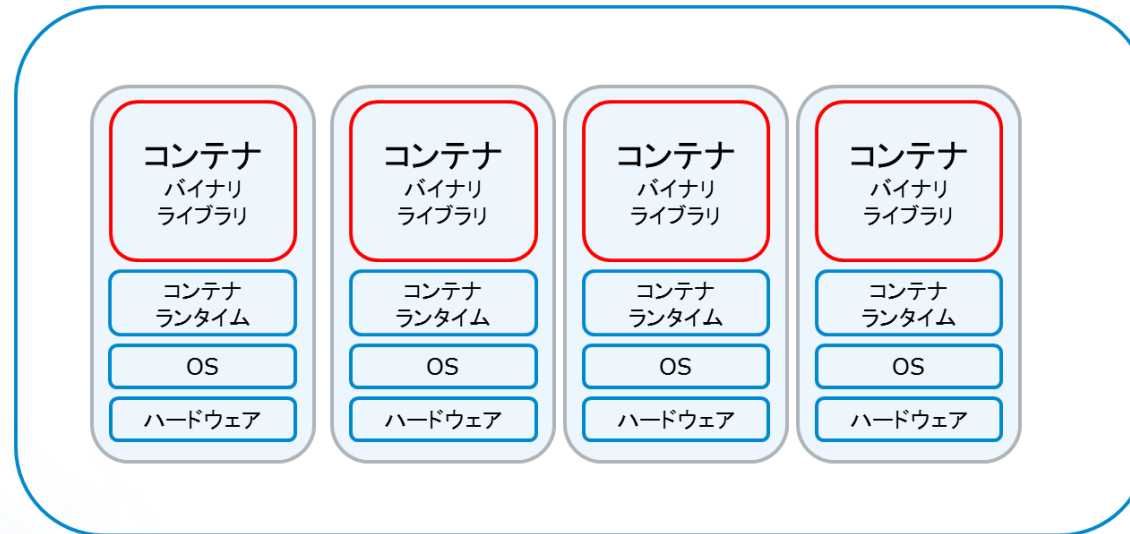
- コンテナの構成ファイルをGitリポジトリで管理・共有できる
- コンテナのイメージをリポジトリで管理・共有できる
- 継続的インティグレーション／継続的デリバリー（CI・CD）を実施しやすく、アプリケーション開発が効率的にできる

CI/CD

アプリケーションソースコードの変更がされたら即座にコンテナを作成して、テスト環境でテストして本番環境にリリース可能な状況にしておく

生産性の向上や環境管理の負荷軽減が期待できる

- コンテナを動かす際には、可用性・性能を考慮して複数のホストマシンから構成されるクラスタが必要

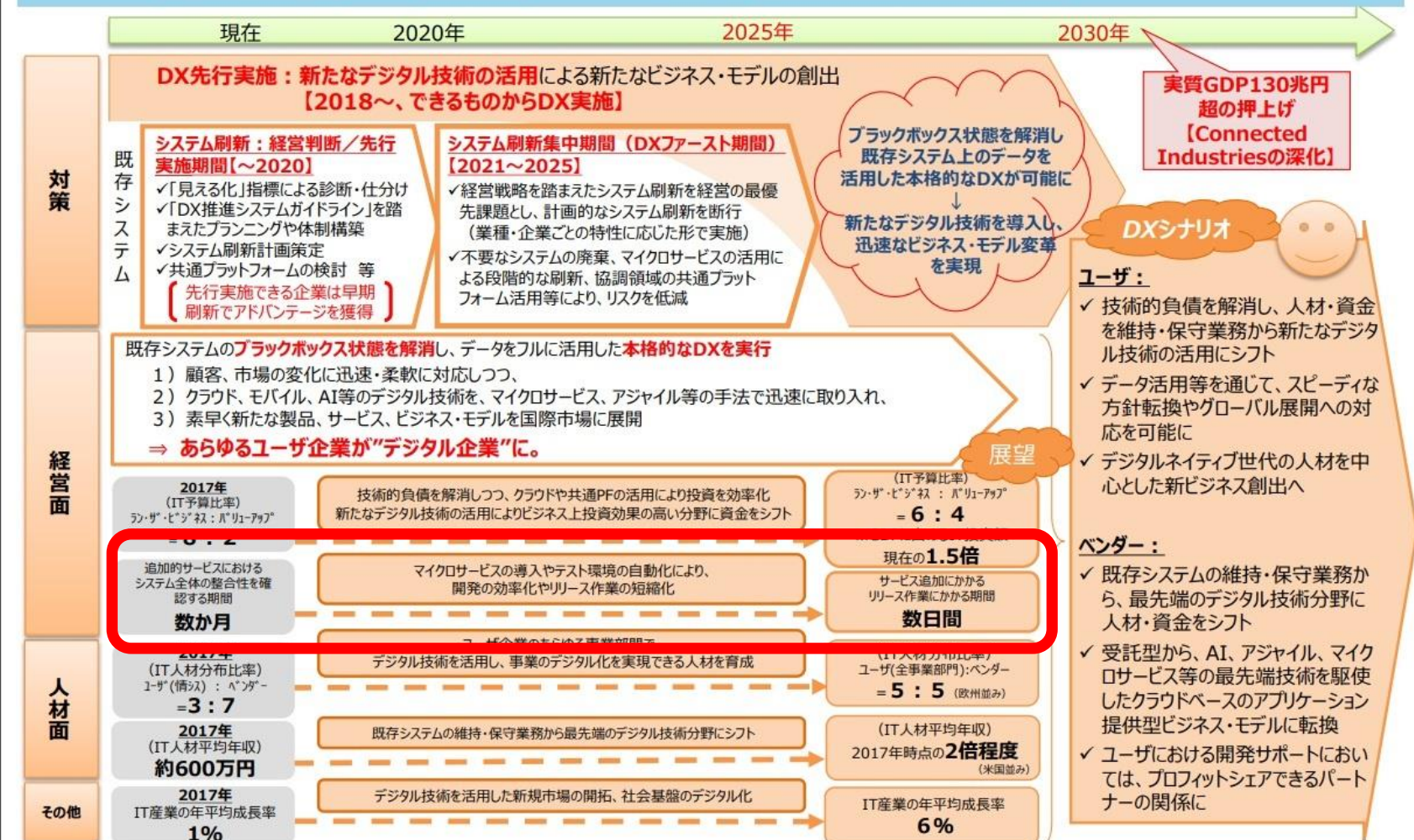


コンテナを管理するための
クラスタプラットフォーム



DX実現シナリオ

【DXシナリオ】2025年までの間に、複雑化・ブラックボックス化した既存システムについて、廃棄や塩漬けにするもの等を仕分けしながら、必要なものについて刷新しつつ、DXを実現することにより、2030年実質GDP130兆円超の押上げを実現。



- Googleでの10年以上の運用経験をもとに設計されたシステムをオープンソース化(2014~)
- CNCFにより開発・メンテナンス
 - ベンダーフリー・認定システム間で互換性が高い
 - kubernetesを中心としたエコシステムの整備
 - Prometheus - fluentd
- 大規模クラスタをサポート

<https://www.cncf.io/certification/software-conformance/>



<https://kubernetes.io/docs/setup/best-practices/cluster-large/>

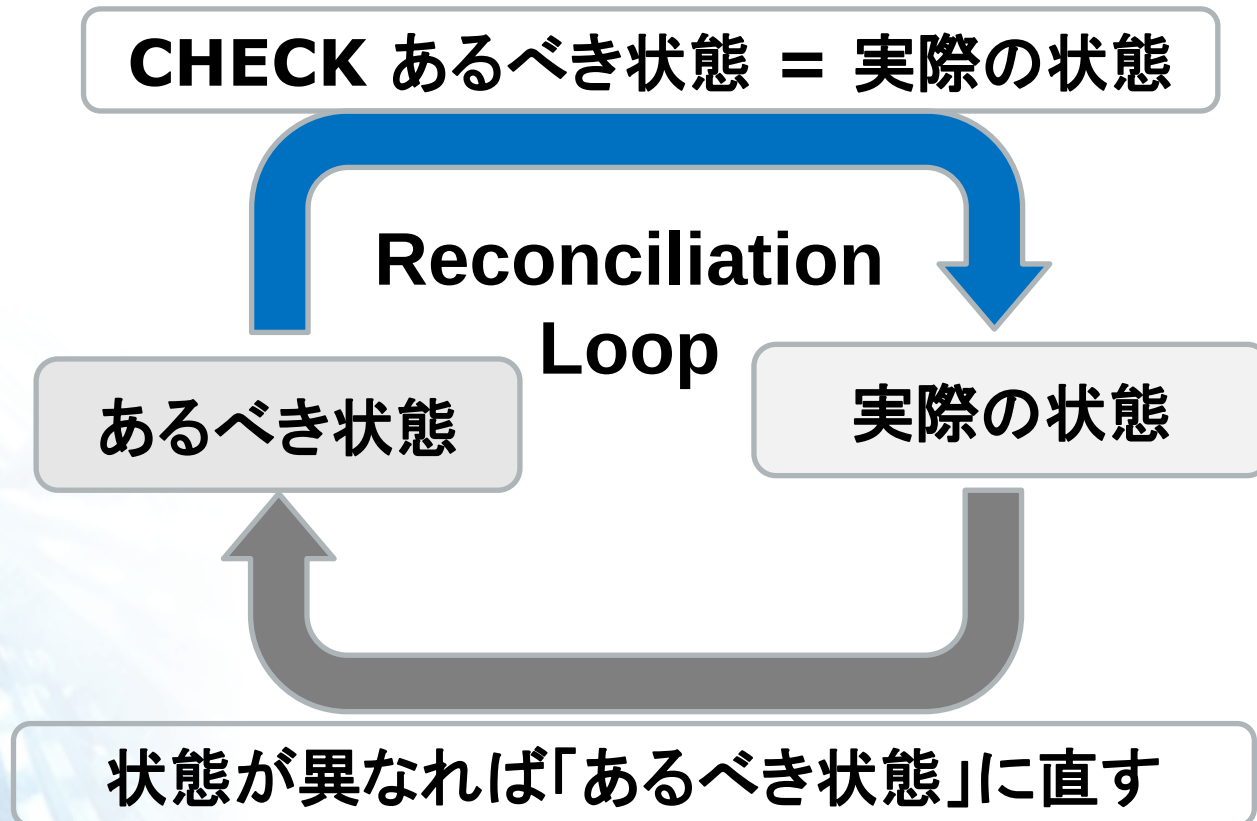
項目	サポート対象
ノード数	5000台以下
Pod数	150,000以下
コンテナ数	300,000以下
1ノードあたりのpod数	100以下

- Kubernetes X.Y.Z
 - X メジャーバージョン
 - Y マイナーバージョン
 - Z パッチバージョン
 - 最新 1.16.2 (11/4時点)
- マイナーバージョンのリリースサイクル 約3カ月
- 3つのマイナーバージョンをメンテナンス

一つのマイナーバージョンは
約9か月間のみメンテナンスされる

<https://kubernetes.io/docs/setup/release/version-skew-policy/>

- ユーザが宣言的に「あるべき状態」を定義
- Kubernetesは「あるべき状態」になるように「自律的」に働く



- 宣言1: Nginx を3つ動かす
1つのNginxに障害が発生して、合計2つしか動いていない場合には新規で起動
⇒ 障害の自動修復
- 宣言2: Nginxの平均CPU利用率が50%になるようにNginxを自動的に増やす
⇒ 水平オートスケール
- 宣言3: ノードのリソースが不足したらノードを追加
⇒ ノードの水平オートスケール

- 宣言1: Nginx を3つ動かす

Nginxが2つしか動いていない場合には新規で起動

⇒ 可用性

Kubernetesの特徴

「あるべき状態」になるように
「自律的」に働く

⇒ ノードの水平オートスケール



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: redis-master
  labels:
    app: redis
spec:
  selector:
    matchLabels:
      app: redis
      role: master
      tier: backend
```

replicas: 1

```
template:
  metadata:
    labels:
      app: redis
      role: master
      tier: backend
```

```
spec:
  containers:
    - name: master
```

image: redis:5.0.4

```
resources:
  requests:
```

cpu: 100m

memory: 100Mi

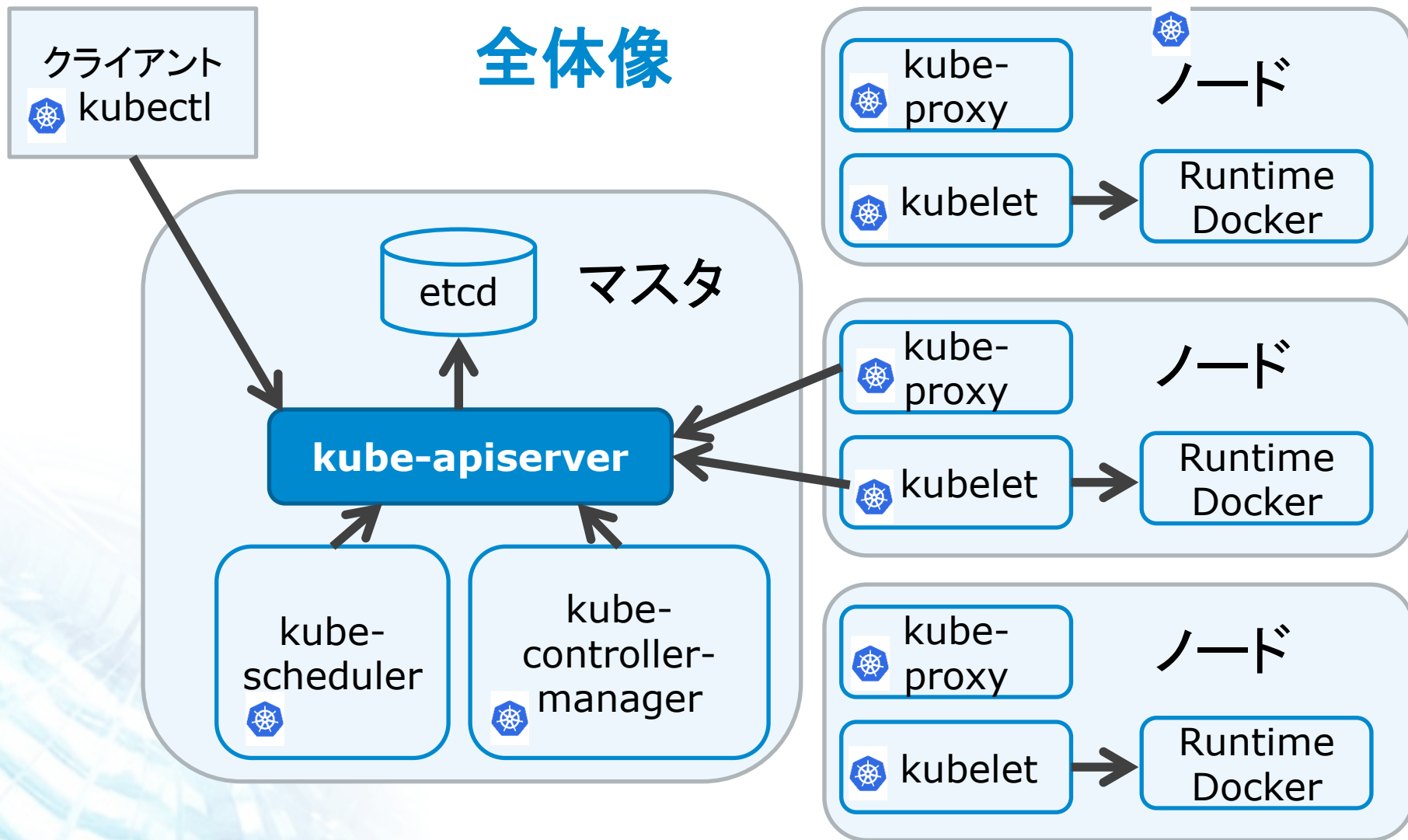
```
ports:
  - containerPort: 6379
```

構成をファイルで定義できる
Githubなどで管理しやすい

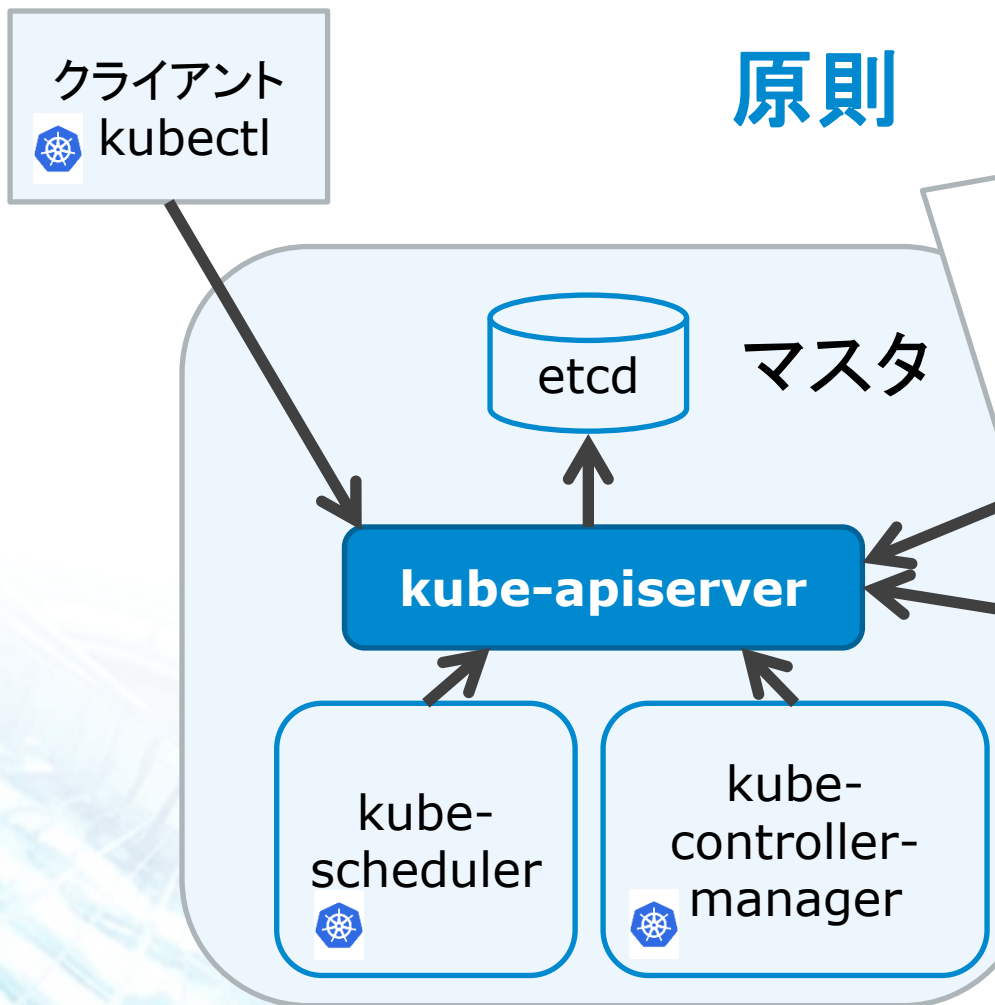
複製数の指定

コンテナイメージの指定

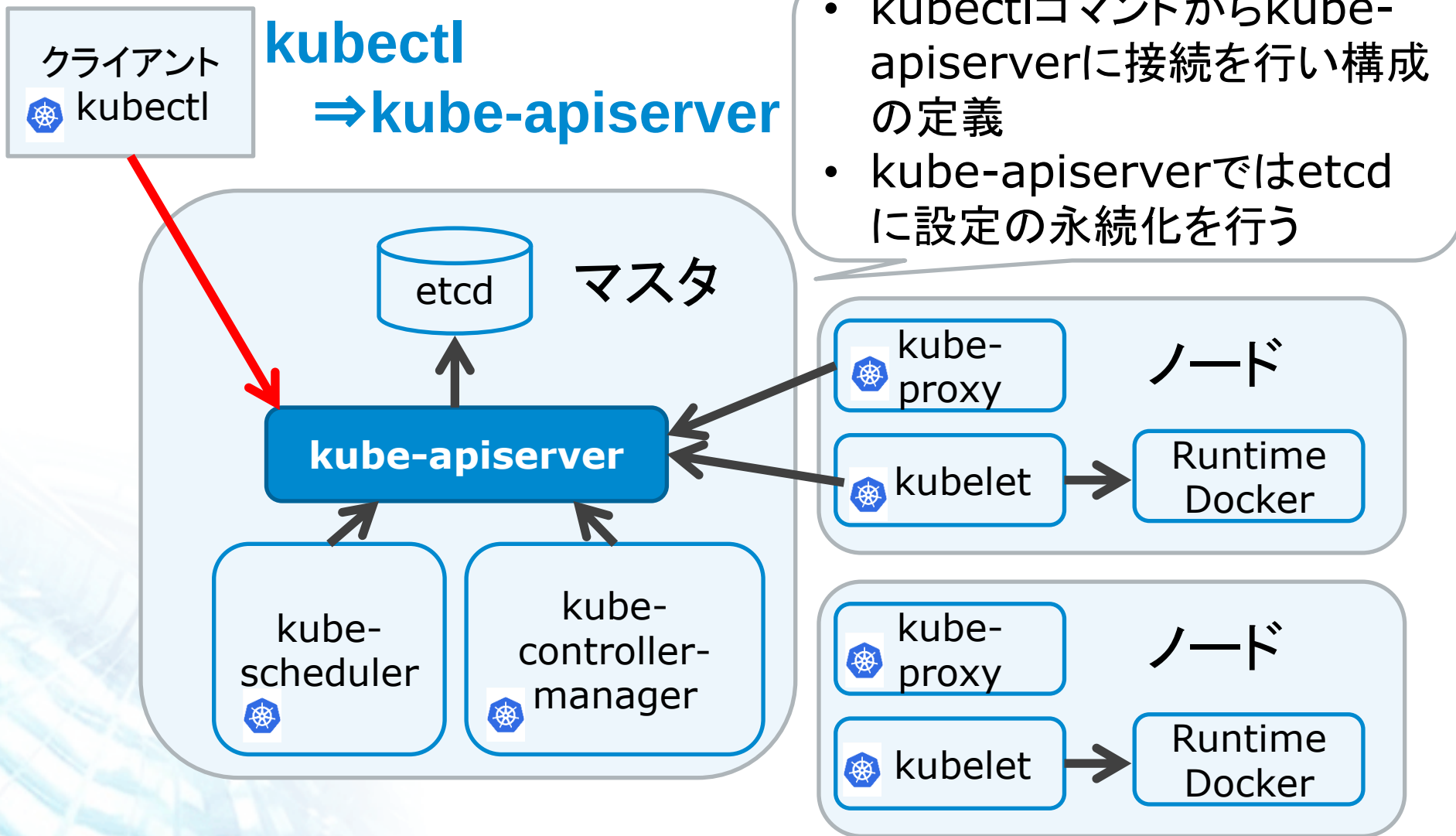
H/Wリソースの設定

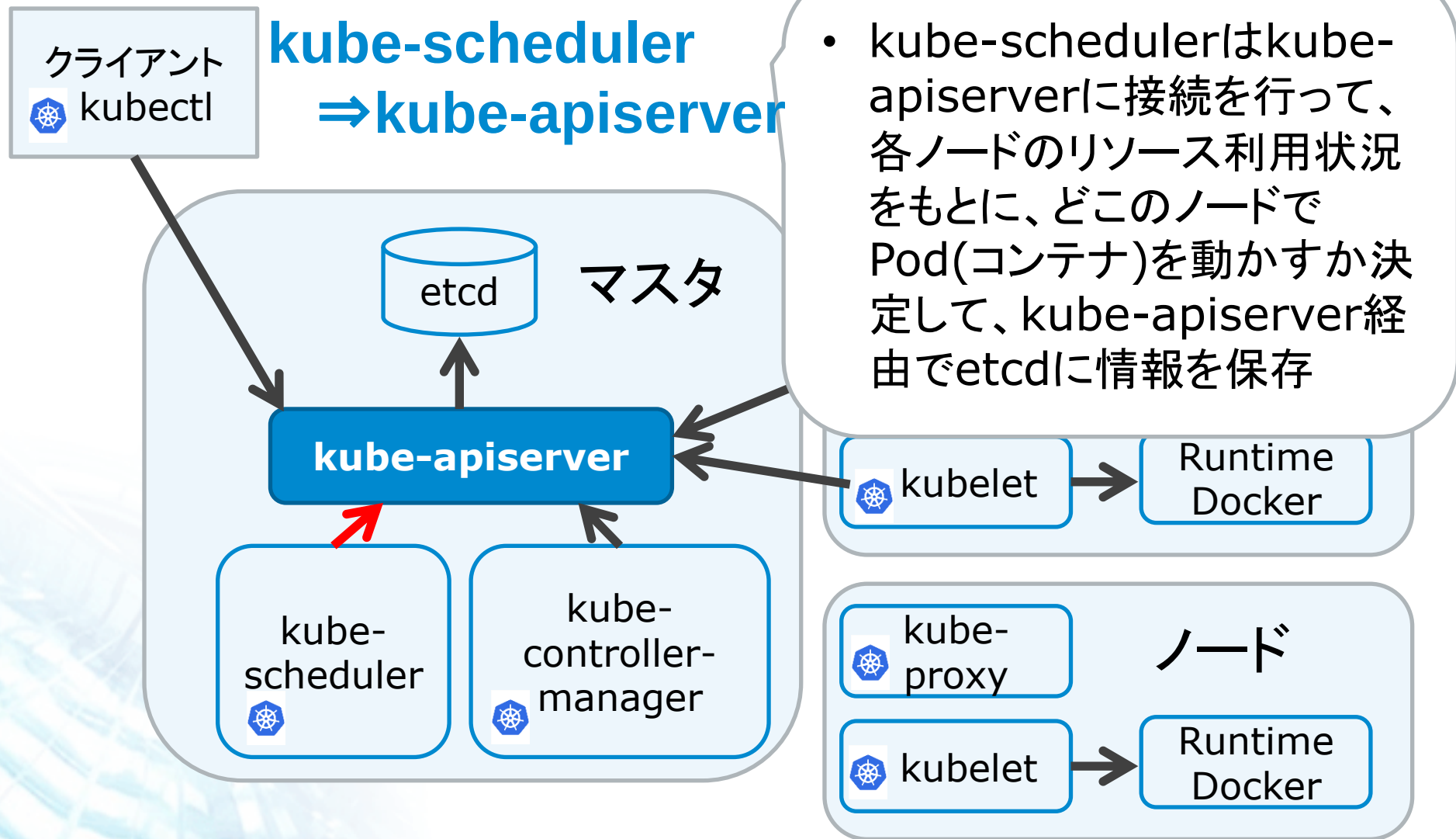


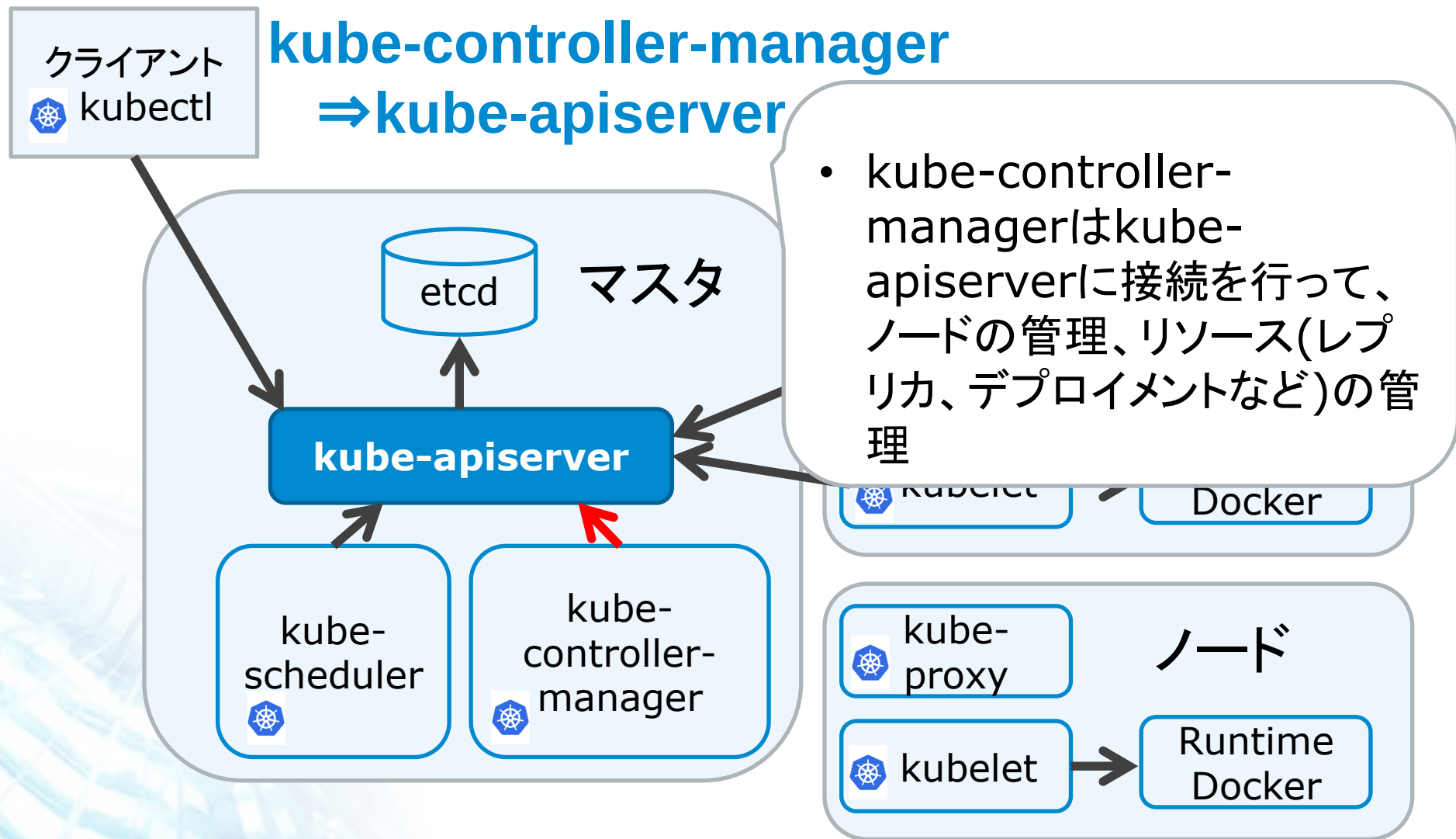
原則

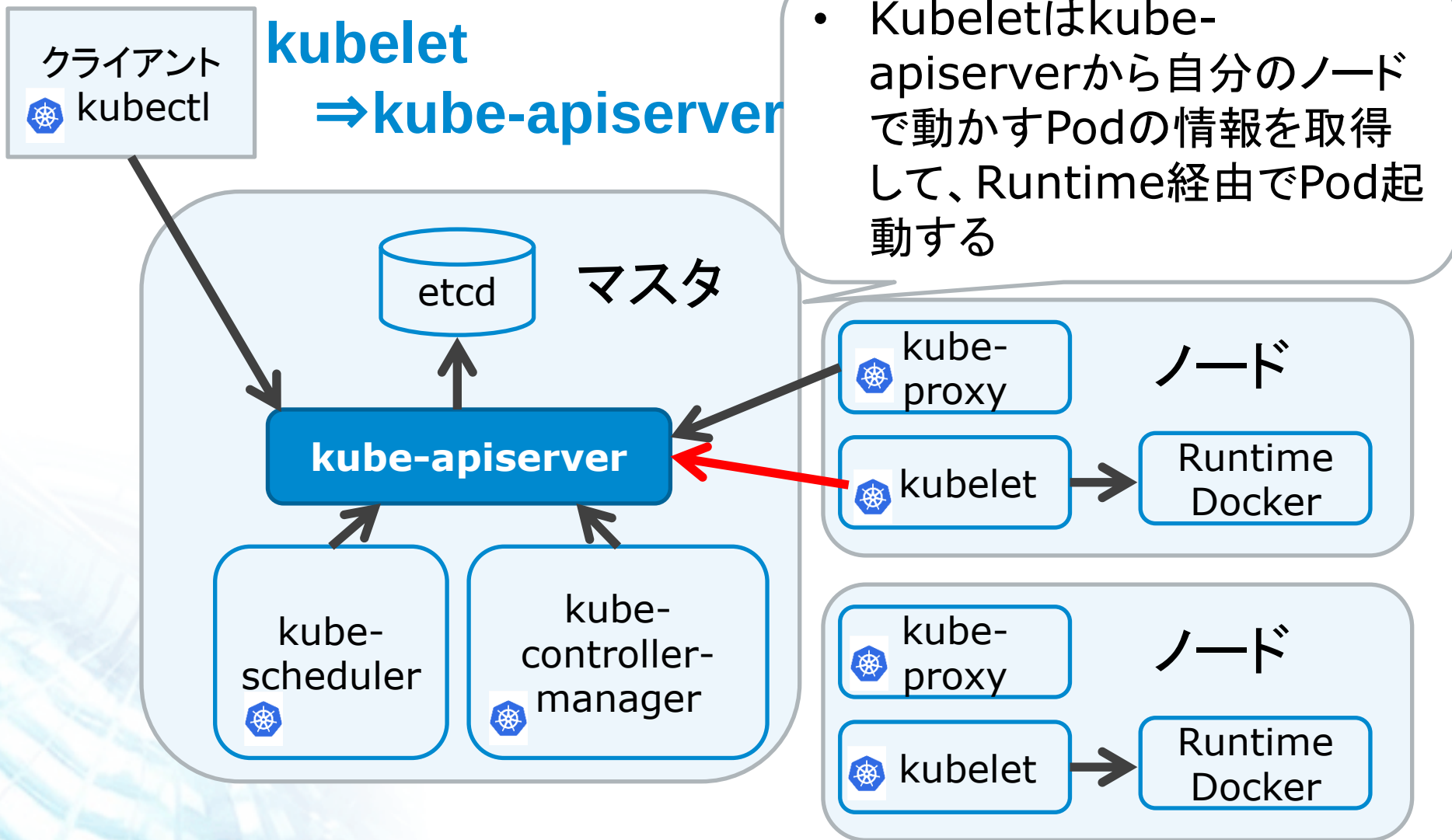


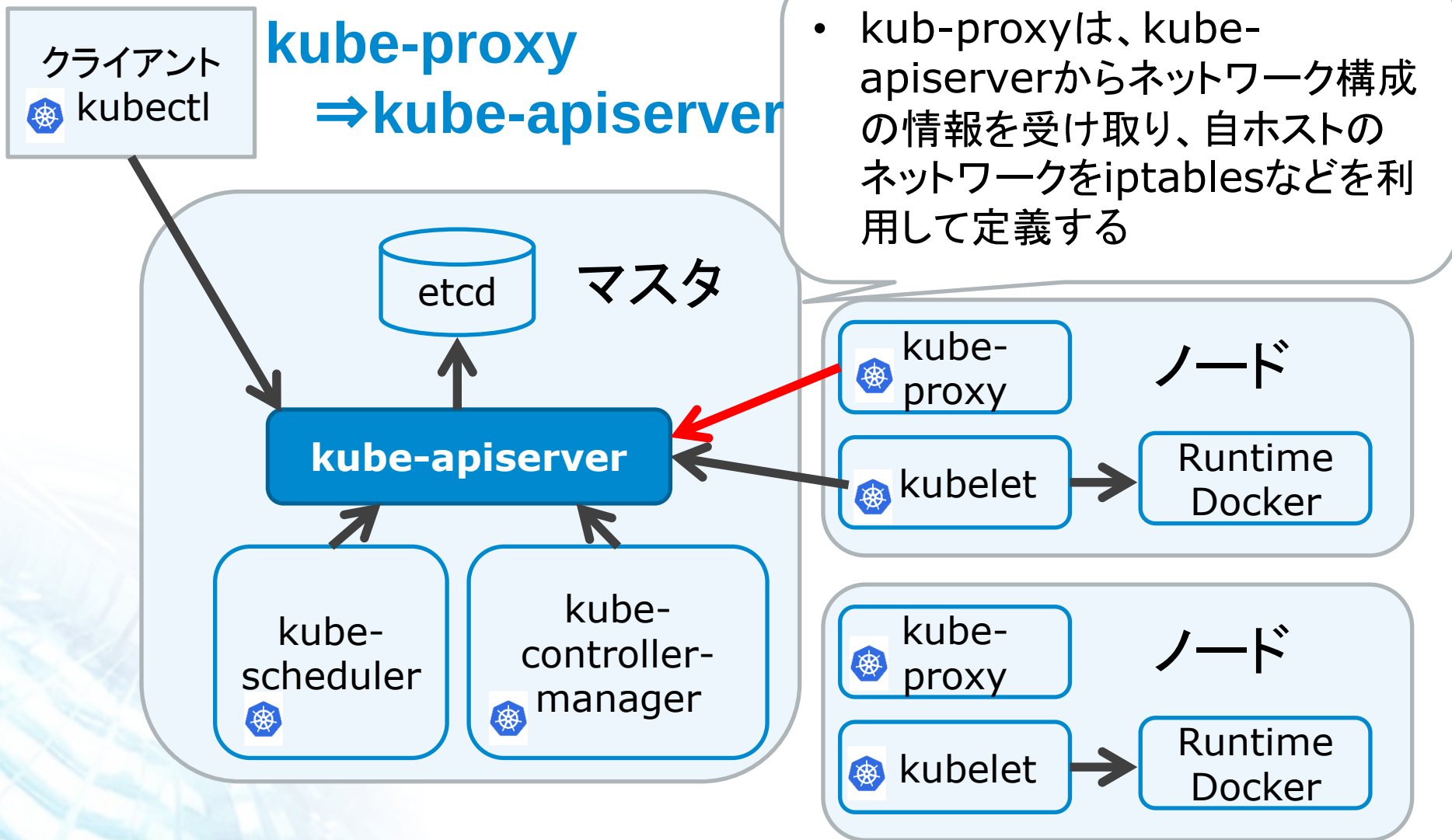
- 各コンポーネント  は kube-apiserverを通してやりとりを行う
- データはetcdに保存される
- kube-apiserverからのみ etcdへアクセスできる
- 部分的な障害がサービス停止にならないようなアーキテクチャ
- 他のコンポーネントに接続ができなくても、各コンポーネントは直近の内容をもとに動作を続ける

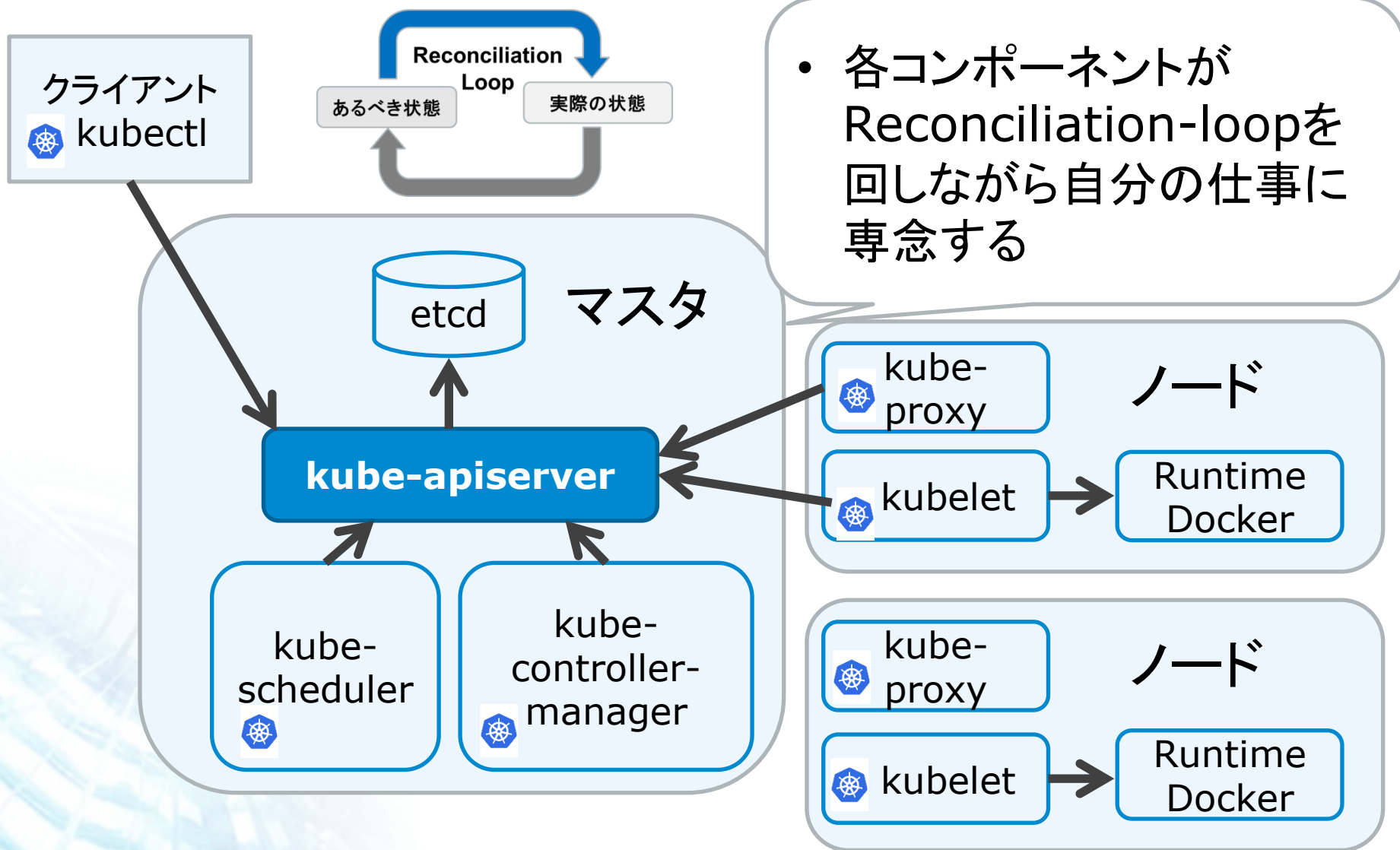




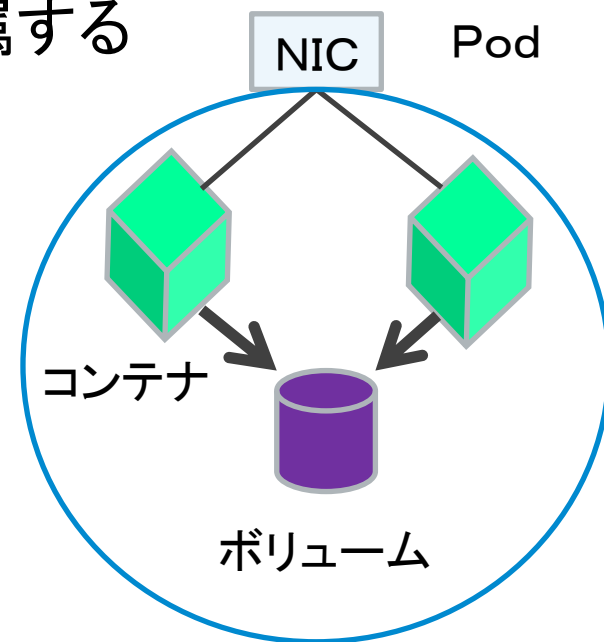
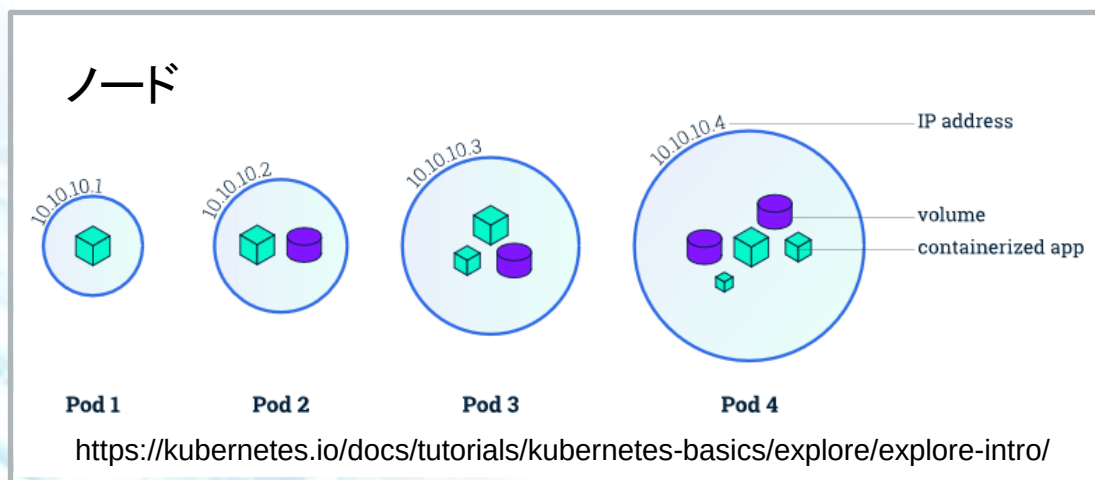








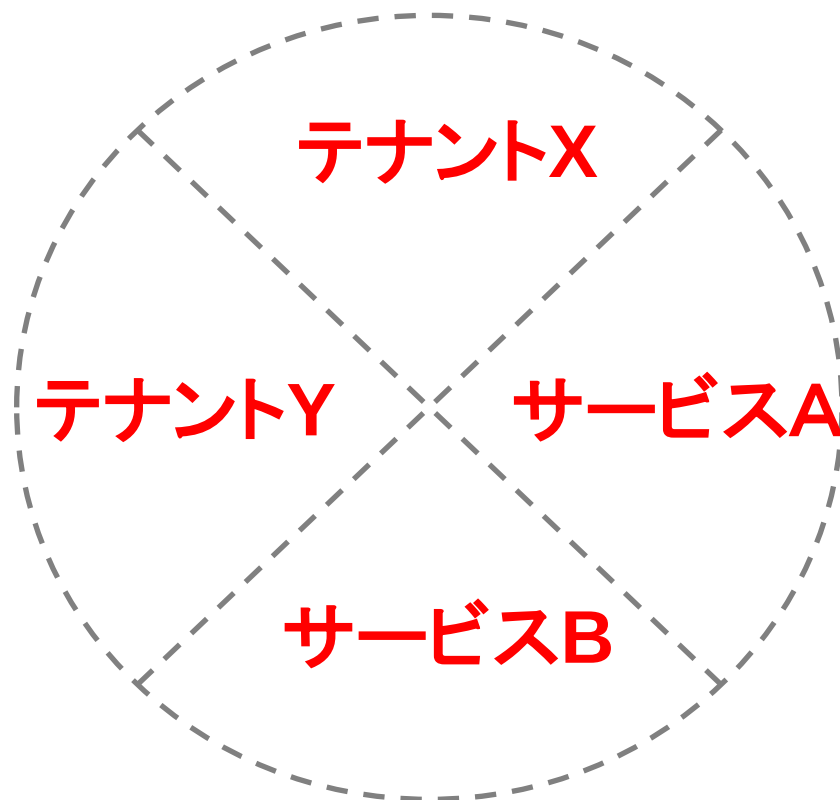
- kubernetesにデプロイする「論理ホスト」
 - 1つ以上のコンテナ
 - 共有ネットワーク
 - 共有ボリューム
 - 同じPod内のコンテナは同一ノードにデプロイされる
 - PodはいずれかのNamespaceに属する



- 一つのk8sクラスター上で複数の仮想クラスターの動作をサポート

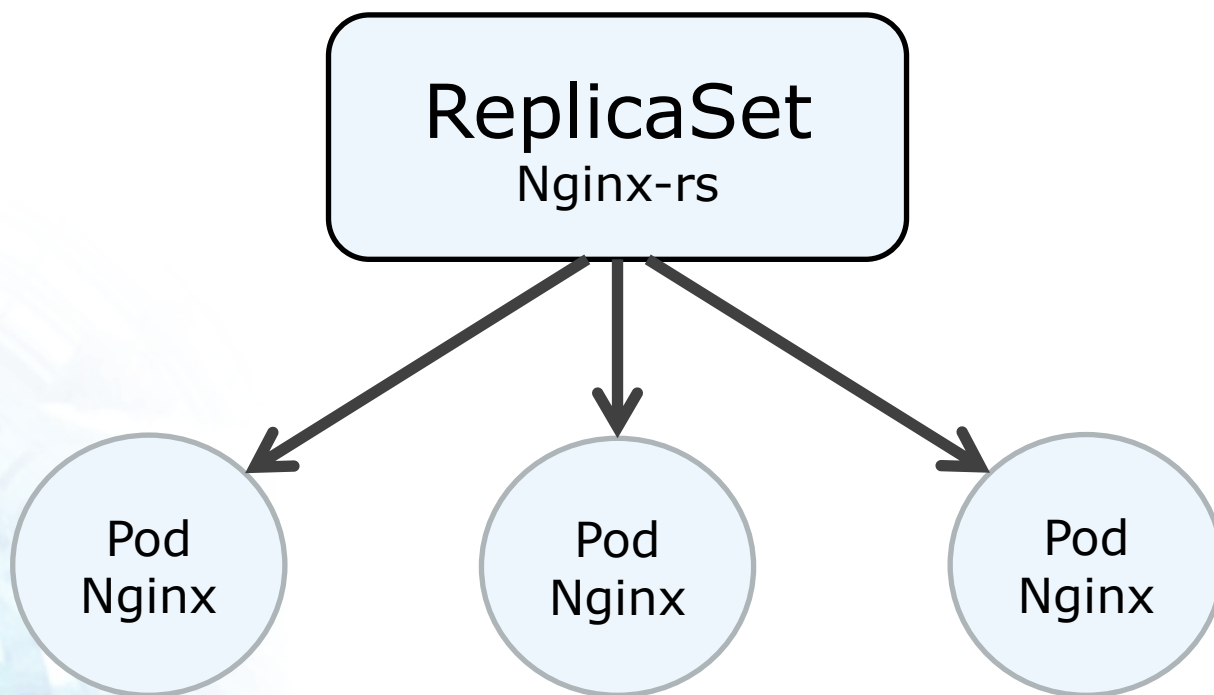


開発・テスト環境

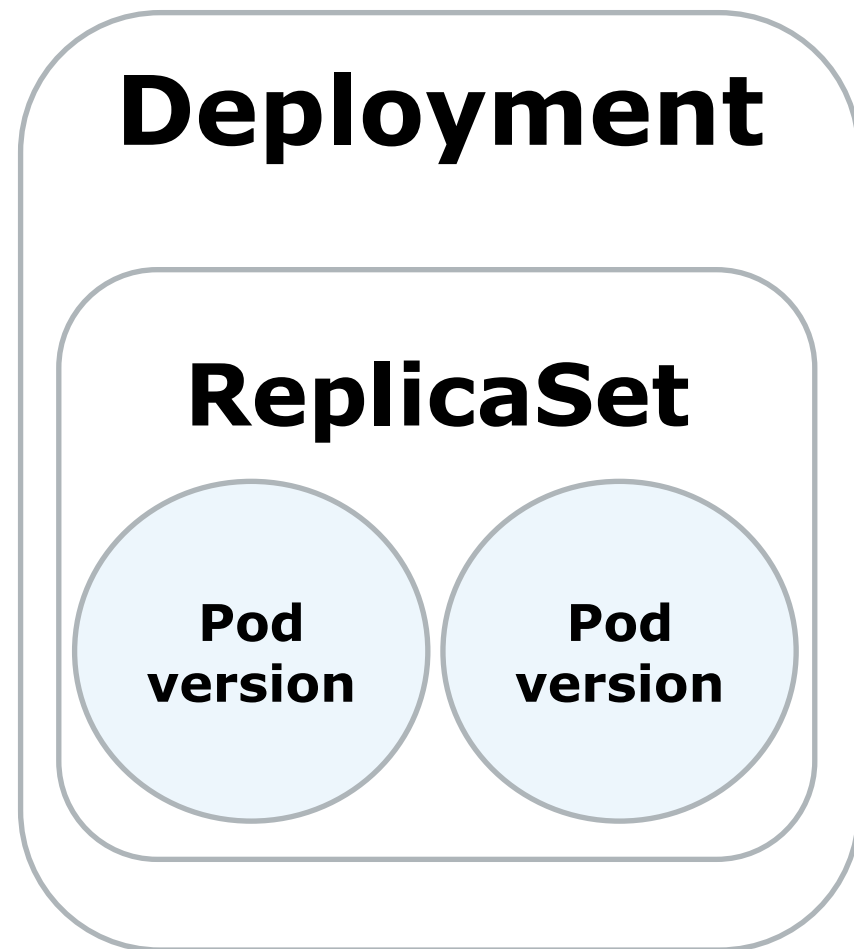


本番環境

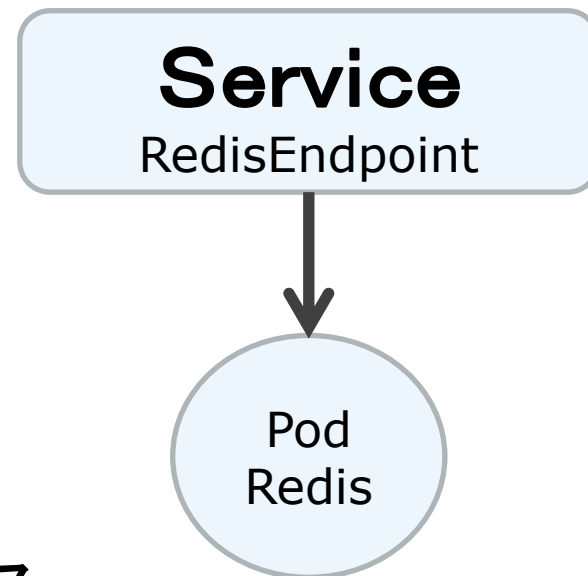
- Podの上位リソース
- 同じPodを指定数だけ起動して管理するリソース
(コントローラが指定数動いているか確認する)



- ReplicaSetの上位リソース
- バージョン管理
特定のバージョンにアップデート/ロールバック
- アップデート戦略
 - Recreate
Podをすべて停止、新しいバージョンで一斉に起動
 - RollingUpdate
少数の Pod を順次アップデート

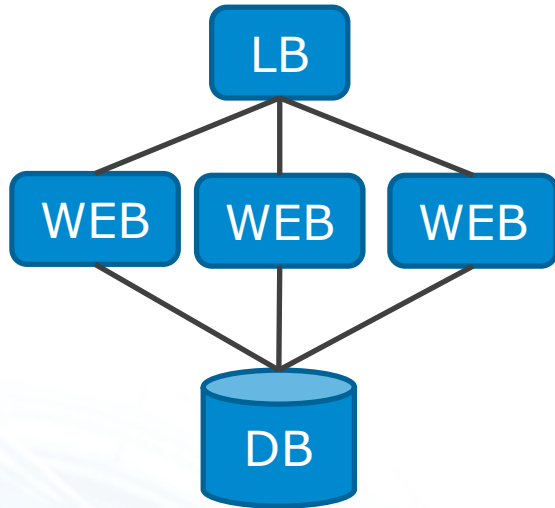


- PodのIPアドレスは不定
- ServiceはPodへのエンドポイントを提供する
 - クラスタ内部向け
 - 外向け
 - クラスタ内外の負荷分散
- Kubernetes内部でもっているDNSを利用して名前解決
- アプリからはサービス名でアクセス

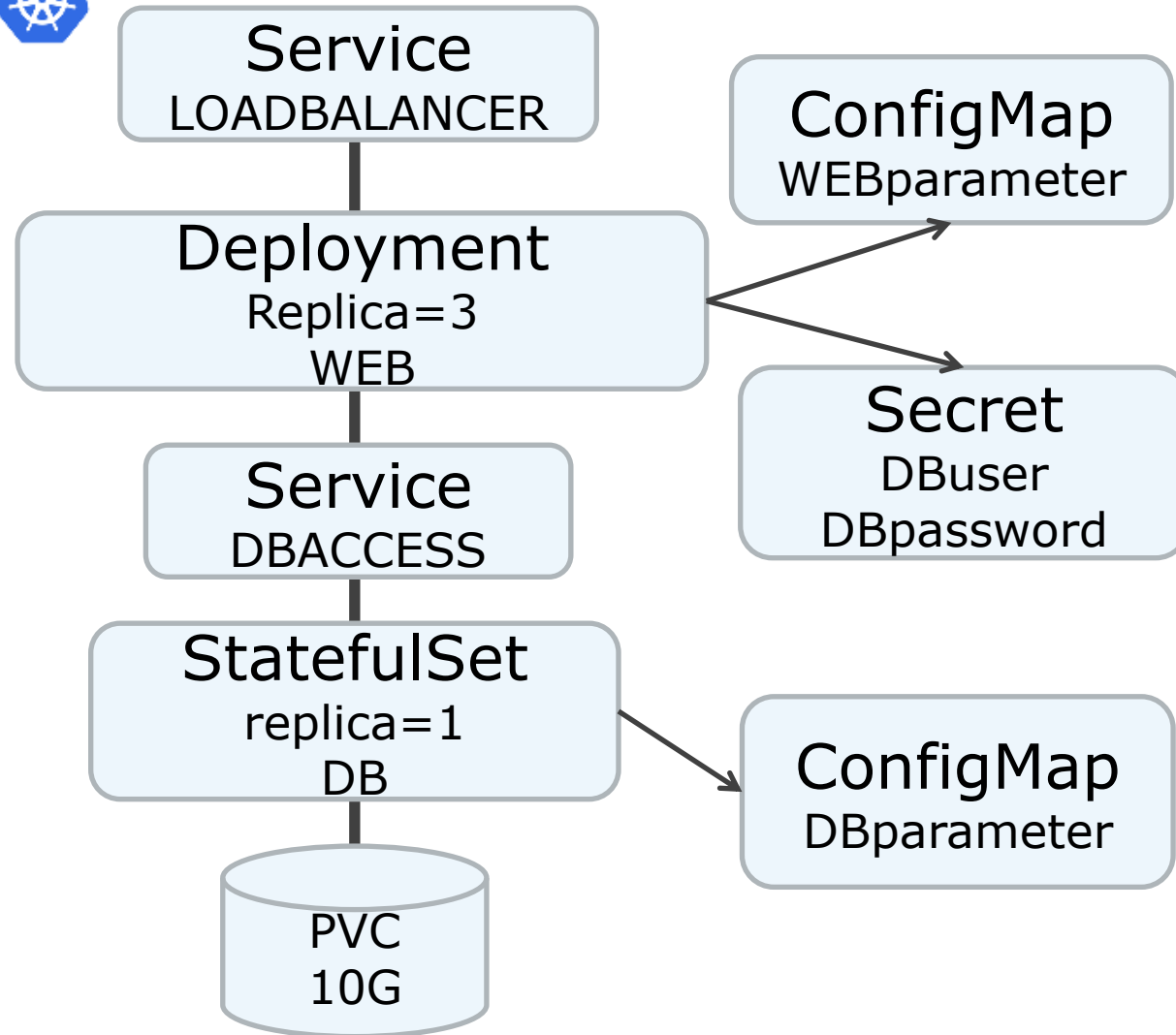


- DaemonSet
 - 各ノードで一つだけ動くPod
- Statefulset
 - 状態を保持するアプリケーション
 - ネットワーク識別子、永続的なストレージを利用できる
- Job/CronJob
 - 処理が終わったら終了するPod
- PersistentVolume/PersistentVolumeClaim
 - 永続的なボリューム
- ConfigMap/Secret
 - 設定・パスワード

構成図



リソースを組み合わせてYAMLファイルで構成定義



- リソースを自分で作ることもできる
⇒ 独自のReconciliation Loopの開発
- オペレータの運用ノウハウをコードで表現できる
- <https://operatorhub.io/>で公開されている



例: Zalando社:Postgres-Operator

PostgreSQL SR構成を管理してくれるリソース
Primary-Standby(マスター-スレーブ)の状態管理



Postgres-Operator
provided by Zalando SE

Postgres operator creates and
manages PostgreSQL clusters
running in Kubernetes.

VMによる運用

- システム管理者
 - VMの払い出し
 - ネットワークの定義
 - ホストの監視・管理
 - ミドルウェアの冗長化対策
- 開発者
 - sshなどでVMにアクセスして環境の構築

コンテナによる運用

- システム管理
 - kubernetesクラスタの監視・管理
- 開発者
 - ファイルベースで構成を定義
 - Gitなどでアプリケーションと構成を管理

Pros

- コンテナ開発による生産性の向上
- コンテナによる環境管理の負荷軽減
- コンテナによる効率的なサーバ運用
- 自動スケーリング
- 障害の自動修復
- ファイルベースによる構成、バージョン管理
- 特定のベンダに支配されない標準技術

Cons

- 学習コストが高い
- Kubernetesのリリースが早くて追従が大変
- エコシステム(関連ソフトウェア)も進歩が速く追従が大変

- オンラインコースの活用
 - ブラウザでkubernetesクラスタを操作できる

The screenshot displays the Udemy website interface. At the top, a teal banner reads '今すぐ課題の解決に向けてスタートしよう | 30日間返金保証だから安心。今すぐ登録しよう。' (Start now to solve your problems | 30-day money-back guarantee, so you can be安心. Register now). Below this is the Udemy logo and navigation links. The main content area features a course titled 'Kubernetes for the Absolute Beginners - Hands-on' with a description: 'Learn Kubernetes in simple, easy and fun way with hands-on coding exercises. For beginners in DevOps.' The course has a 4.6 rating from 6020 reviews and 28,531 students. The creator is Mumshad Mannambeth, and it was last updated on 7/2019. The course is available in English, with subtitles in English and Italian. A search bar at the bottom of the course page contains the text 'あなたのペースで学ぼう' (Learn at your own pace) and 'いつでもどんなトピックでも学べる。専門家が教える何千ものコースをご用意。' (You can learn about any topic at any time. We have thousands of courses taught by experts for you). The bottom of the page features a red banner with three icons and text: '100000のオンラインコース 様々な新しいトピックが追加されています' (100,000 online courses, various new topics are being added), '専門的指導 安心して受講を始めよう' (Specialized guidance, start your course with confidence), and '学習期限なし あなたのペースで学ぼう' (No learning deadline, learn at your own pace).

今すぐ課題の解決に向けてスタートしよう | 30日間返金保証だから安心。今すぐ登録しよう。

Udemy カテゴリー 何でも検索 ログイン 新規登録

ITとソフトウェア > ネットワークとセキュリティ > Kubernetes

Kubernetes for the Absolute Beginners - Hands-on

Learn Kubernetes in simple, easy and fun way with hands-on coding exercises. For beginners in DevOps.

★★★★★ 4.6 (6020件の評価) 28531人の受講生が登録

作成者 Mumshad Mannambeth 最終更新 7/2019

🗣 英語 🗣 英語, イタリア語 [自動生成], 4以上

学習内容

- ✓ Gain basic understanding of Kubernetes Fundamentals
- ✓ Develop Kubernetes Configuration Files in YAML
- ✓ Deploy Kubernetes Cluster on local systems
- ✓ Deploy Kubernetes on Google Cloud Platform
- ✓ Deploy Applications on Kubernetes
- ✓ Setup ReplicaSets, Service Deployments on Kubernetes

あなたのペースで学ぼう

いつでもどんなトピックでも学べる。専門家が教える何千ものコースをご用意。

学びたいコースを検索

100000のオンラインコース 様々な新しいトピックが追加されています

専門的指導 安心して受講を始めよう

学習期限なし あなたのペースで学ぼう

- Certified Kubernetes Administrator (CKA) 試験

- Kubernetesの管理者に必要なスキル、知識、および能力が問われる試験
- ハンズオン形式の試験 試験時間3時間



- Certified Kubernetes Application Developer (CKAD) 試験

- Kubernetes上でアプリケーションを設計・構築・設定・展開できることを証明する資格試験
- ハンズオン形式の試験 試験時間 2時間

- **kubernetes slack** <https://slack.k8s.io/>
#jp-users #jp-events チャンネル
- **Kubernetes Meetup Tokyo**
<https://k8sjp.connpass.com/>
毎月開催 19:00~
200~300人規模のイベント
youtube配信あり



CLOUD NATIVE TRAIL MAP

The Cloud Native Landscape landscape.cncf.io has a large number of options. This Cloud Native Trail Map is a recommended process for leveraging open source, cloud native technologies. At each step, you can choose a vendor-supported offering or do it yourself, and everything after step #3 is optional based on your circumstances.

HELP ALONG THE WAY

A. Training and Certification

Consider training offerings from CNCF and then take the exam to become a Certified Kubernetes Administrator or a Certified Kubernetes Application Developer cncf.io/training

B. Consulting Help

If you want assistance with Kubernetes and the surrounding ecosystem, consider leveraging a Kubernetes Certified Service Provider cncf.io/csp

C. Join CNCF's End User Community

For companies that don't offer cloud native services externally cncf.io/enduser

WHAT IS CLOUD NATIVE?

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

The Cloud Native Computing Foundation seeks to drive adoption of this paradigm by fostering and sustaining an ecosystem of open source, vendor-neutral projects. We democratize state-of-the-art patterns to make these innovations accessible for everyone.

landscape.cncf.io

v20181211



1. CONTAINERIZATION

- Commonly done with Docker containers
- Any size application and dependencies (even PDP-11 code running on an emulator) can be containerized
- Over time, you should aspire towards splitting suitable applications and writing future functionality as microservices

3. ORCHESTRATION & APPLICATION DEFINITION

- Kubernetes is the market-leading orchestration solution
- You should select a Certified Kubernetes Distribution, Hosted Platform, or Installer: cncf.io/kick
- Helm Charts help you define, install, and upgrade even the most complex Kubernetes application



5. SERVICE PROXY, DISCOVERY, & MESH

- CoreDNS is a fast and flexible tool that is useful for service discovery
- Envoy and Linkerd each enable service mesh architectures
- They offer health checking, routing, and load balancing



7. DISTRIBUTED DATABASE & STORAGE

When you need more resiliency and scalability than you can get from a single database, Vitess is a good option for running MySQL at scale through sharding. Rook is a storage orchestrator that integrates a diverse set of storage solutions into Kubernetes. Serving as the "brain" of Kubernetes, etcd provides a reliable way to store data across a cluster of machines.



9. CONTAINER REGISTRY & RUNTIME

Harbor is a registry that stores, signs, and scans content. You can use alternative container runtimes. The most common, all of which are OCI-compliant, are containerd, rkt and CRI-O.



2. CI/CD

- Setup Continuous Integration/Continuous Delivery (CI/CD) so that changes to your source code automatically result in a new container being built, tested, and deployed to staging and eventually, perhaps, to production
- Setup automated rollouts, roll backs and testing

4. OBSERVABILITY & ANALYSIS

- Pick solutions for monitoring, logging and tracing
- Consider CNCF projects Prometheus for monitoring, Fluentd for logging and Jaeger for Tracing
- For tracing, look for an OpenTracing-compatible implementation like Jaeger



6. NETWORKING

To enable more flexible networking, use a CNI-compliant network project like Calico, Flannel, or Weave Net.



8. STREAMING & MESSAGING

When you need higher performance than JSON-RPC, consider using gRPC or NATS. gRPC is a universal RPC framework. NATS is a multi-model messaging system that includes request/reply, pub/sub and load balanced queues.



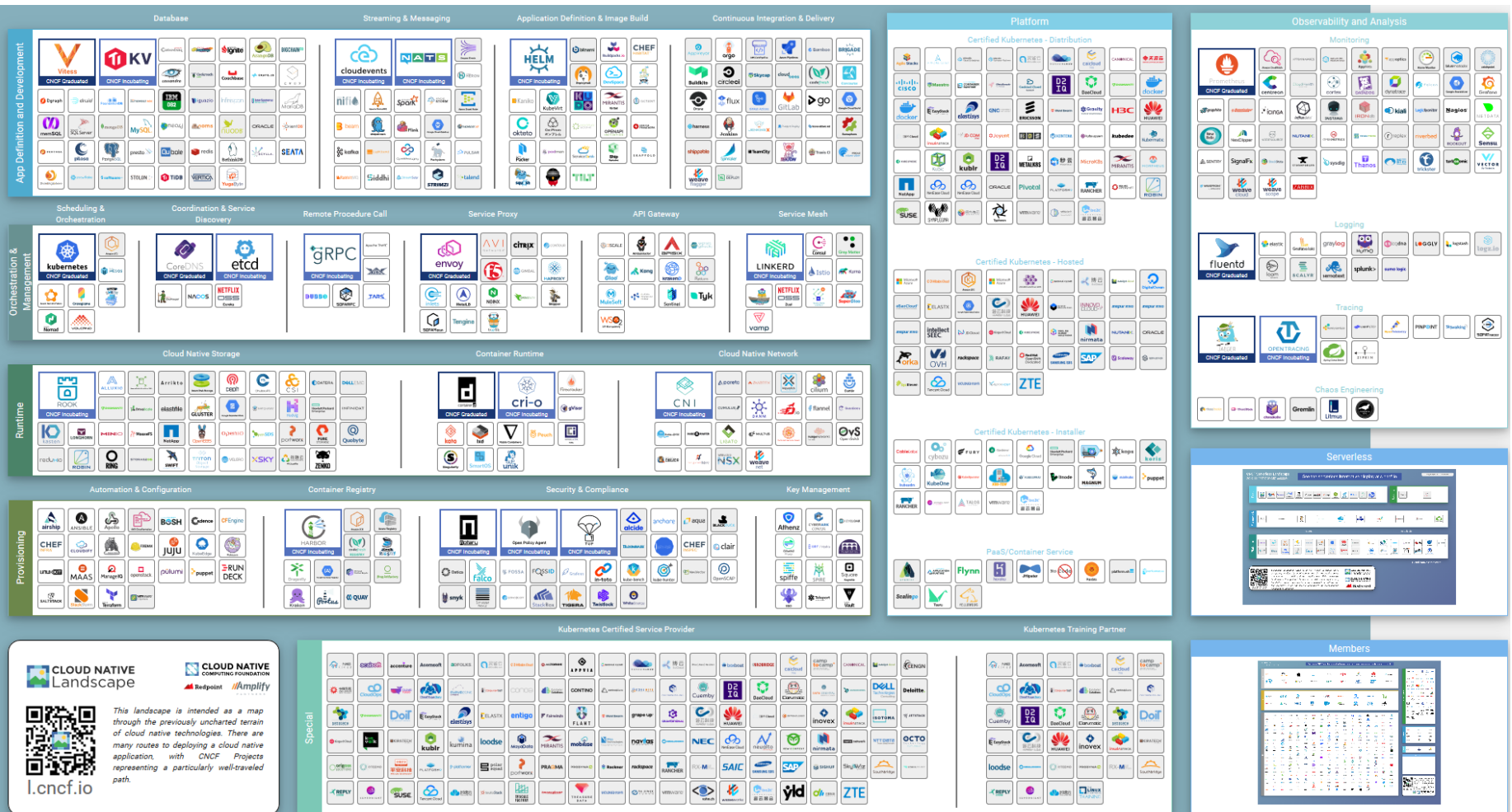
10. SOFTWARE DISTRIBUTION

If you need to do secure software distribution, evaluate Notary, an implementation of The Update Framework.



<https://github.com/cncf/landscape/blob/master/README.md#trail-map>

SRA OSS, INC. CNCF landscape



ご清聴ありがとうございました。