Exastro × Ansibleで管理運用するKubernetesマニフェスト

Exastro

アジェンダ

- はじめに 約30分
  - これまでのインフラ管理と今後のインフラ管理に求められることは？
    - コンテナとは？Kubernetesとは？
  - Exastroを使ってどのように効率化できるのか？
- Exastro × Ansibleを活用した 約30分
  Kubernetesマニフェスト管理デモ
- Q&A

はじめに

これまでのインフラ管理の要素を振り返ってみます。
また、今後求められるインフラ管理とは？

Introduction これまでのインフラ管理とは？（オンプレ）

新技術を取り込みにくい

密結合で保守しづらいアプリ

従来のスタティックな環境

手動構築・手動運用

モノリシック
(物理機器)

仮想化技術がまだ普及していなかった頃…

- ✓ サービス停止させないために「**最繁トラヒック×バッファ×冗長2倍**」のリソースを物理的に並べていた
- ✓ また「CPU、Mem、I/O」の選択肢が少なく、**リソースを使いきれなかった**
- ✓ それでもなるべく少ない機器点数で多くのサービスを捌けるように、**複数機能をうまく混載(コンソリ)すること**が”正しい”とされていた
- ✓ **アプリケーションもこれらを前提**に作られていた(それが正しかった)

Introduction 現在のインフラ管理とは？（仮想化・コンテナ）

新技術を取り込みにくい

密結合で保守しづらいアプリ

従来のスタティックな環境

手動構築・手動運用

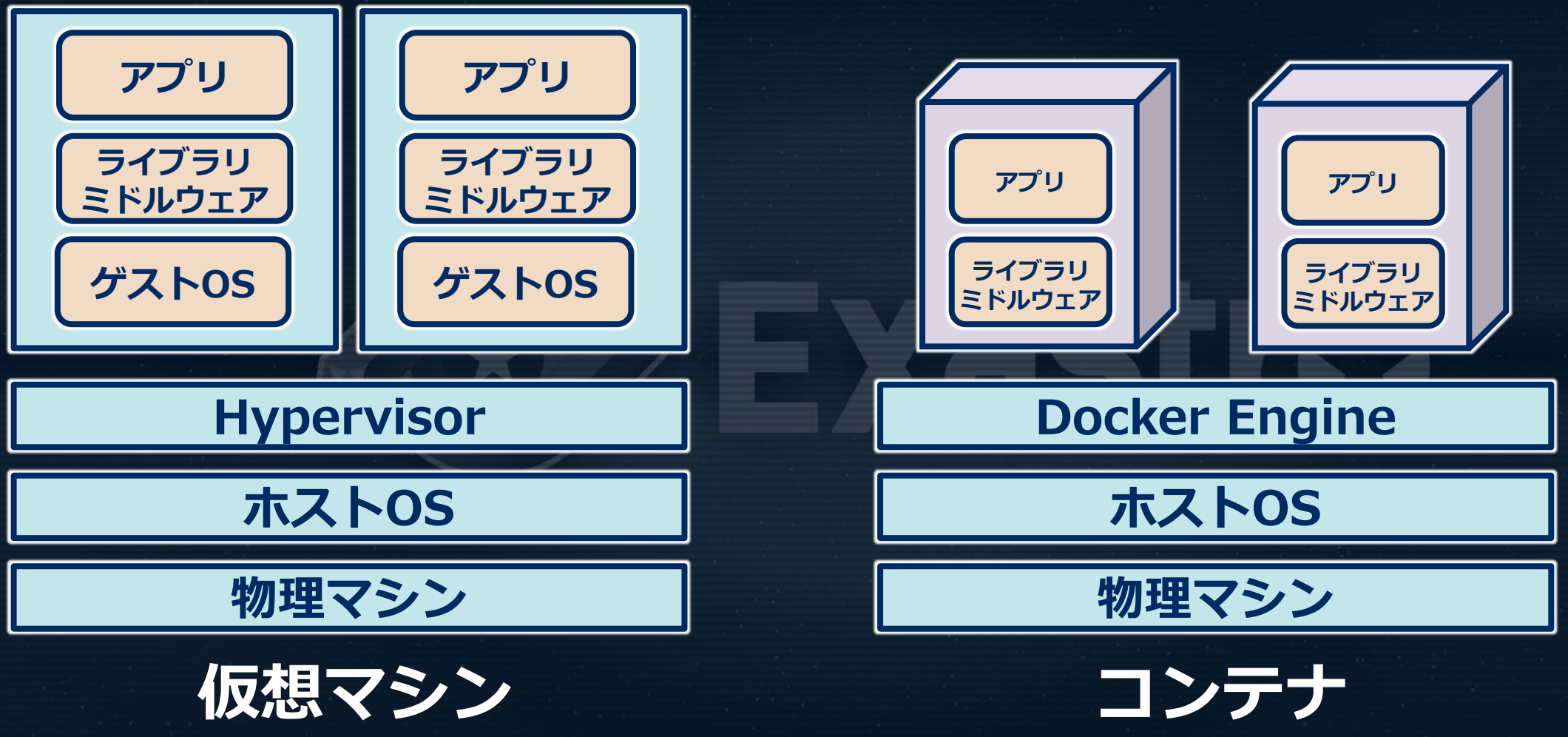
モノリシック
(クラウドリフト)

仮想化技術という武器を手に入れた結果
- ✓「CPU、Mem、I/O」が選択可能に
- ✓「オートスケールWeb」くらいの簡単な動的システムは組めるようになった
- ✓ 結果、システムサイズはコンパクトになりハードウェア寿命から脱却できた
- ✓ 複数機能を混載(コンソリ)させないことが"正しい"と変わり始めた

さらに、コンテナ技術を使うことで
- ✓ 環境のパッケージ化が可能
- ✓ どこでも同じように動くようになる
  →クラウドシフトへの期待

コンテナとは？

✓Dockerに代表される仮想化技術で、非常に軽量なリソースで稼働できる



クラウドネイティブに欠かせないコンテナ・Kubernetesとは？

クラウドネイティブ

コンテナ技術に必要な要素とは？

- ✓ Dockerコンテナ単体のビルドは容易だが、Dockerのみでは複数のコンテナでシステムを組み上げられない
- ✓ 複数のホストへ展開してスケールアウトし、ロードバランシングする仕組み

→コンテナオーケストレーション

kubernetes

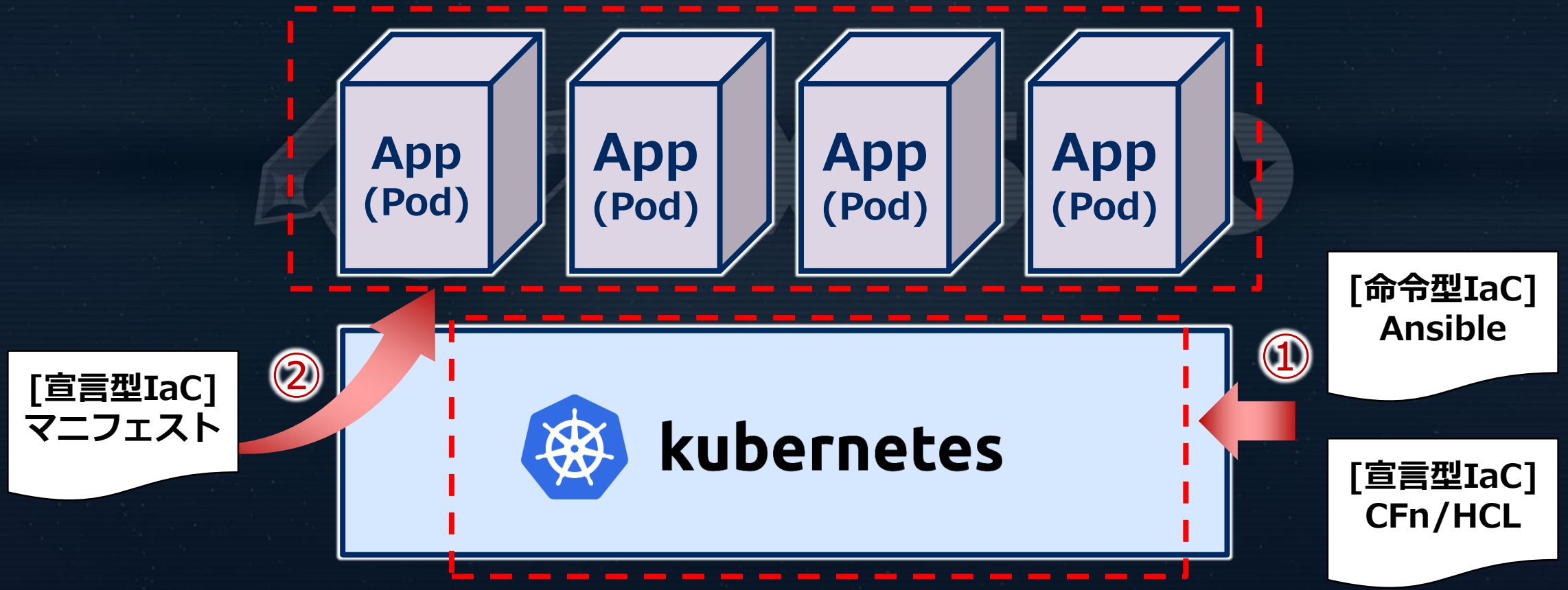
Kubernetesを運用管理するために必要なIaCは？

✓コンテナ基盤の維持管理には2つのIaC(Infrastructure As a Code)が必要となる

① Kubernetes自体の構築やメンテナンスのための構築コード

② Kubernetes上にコンテナ(Pod)をデプロイするための“マニフェスト”

✓本ウェビナーでは②を取り扱います



開発/本番環境の一致

クラウドネイティブな環境の指針 “THE Twelve-Factor App”にて、推奨されている

THE TWELVE-FACTOR APP

X. 開発/本番一致

開発、ステージング、本番環境をできるだけ一致させた状態を保つ

歴史的に、開発環境（開発者が直接変更するアプリケーションのローカルデプロイ）と本番環境（エンドユーザーからアクセスされるアプリケーションの実行中デプロイ）の間には大きなギャップがあった。これらのギャップは3つの領域で現れる。

- **時間のギャップ:** 開発者が編集したコードが本番に反映されるまで数日、数週間、時には数ヶ月かかることがある。
- **人材のギャップ:** 開発者が書いたコードを、インフラエンジニアがデプロイする。
- **ツールのギャップ:** 本番デプロイでは**Apache**、**MySQL**、**Linux**を使うのに、開発者が**Nginx**、**SQLite**、**OS X**のようなスタックを使うことがある。

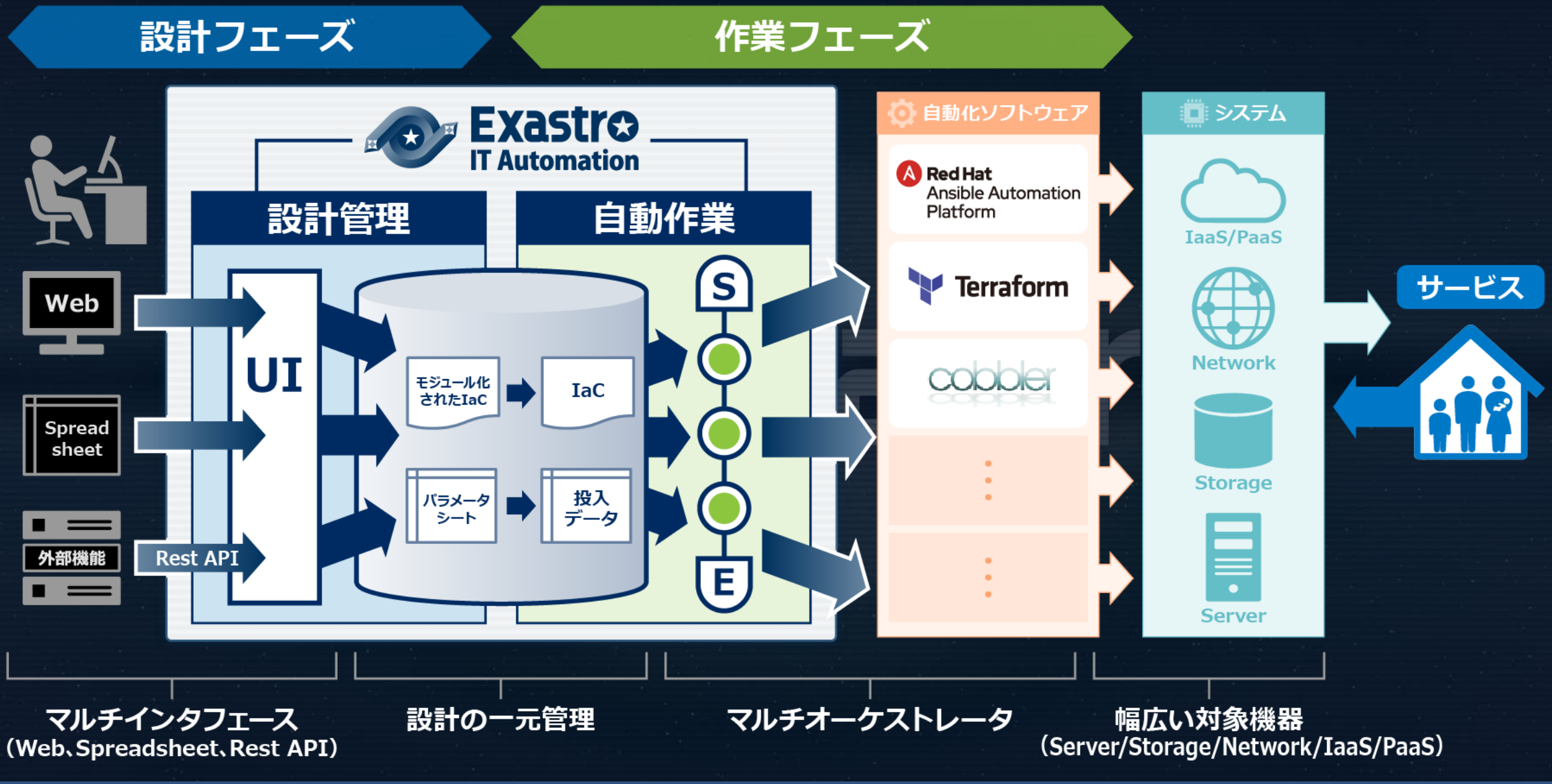
Twelve-Factor Appでは、継続的デプロイしやすいよう開発環境と本番環境のギャップを小さく保つ。 上で述べた3つのギャップを見る。

- 時間のギャップを小さくする: 開発者が書いたコードは数時間後、さらには数分後にはデプロイされる。
- 人材のギャップを小さくする: コードを書いた開発者はそのコードのデプロイに深く関わり、そのコードの本番環境での挙動をモニタリングする。
- ツールのギャップを小さくする: 開発環境と本番環境をできるだけ一致させた状態を保つ。

https://12factor.net/ja/dev-prod-parity

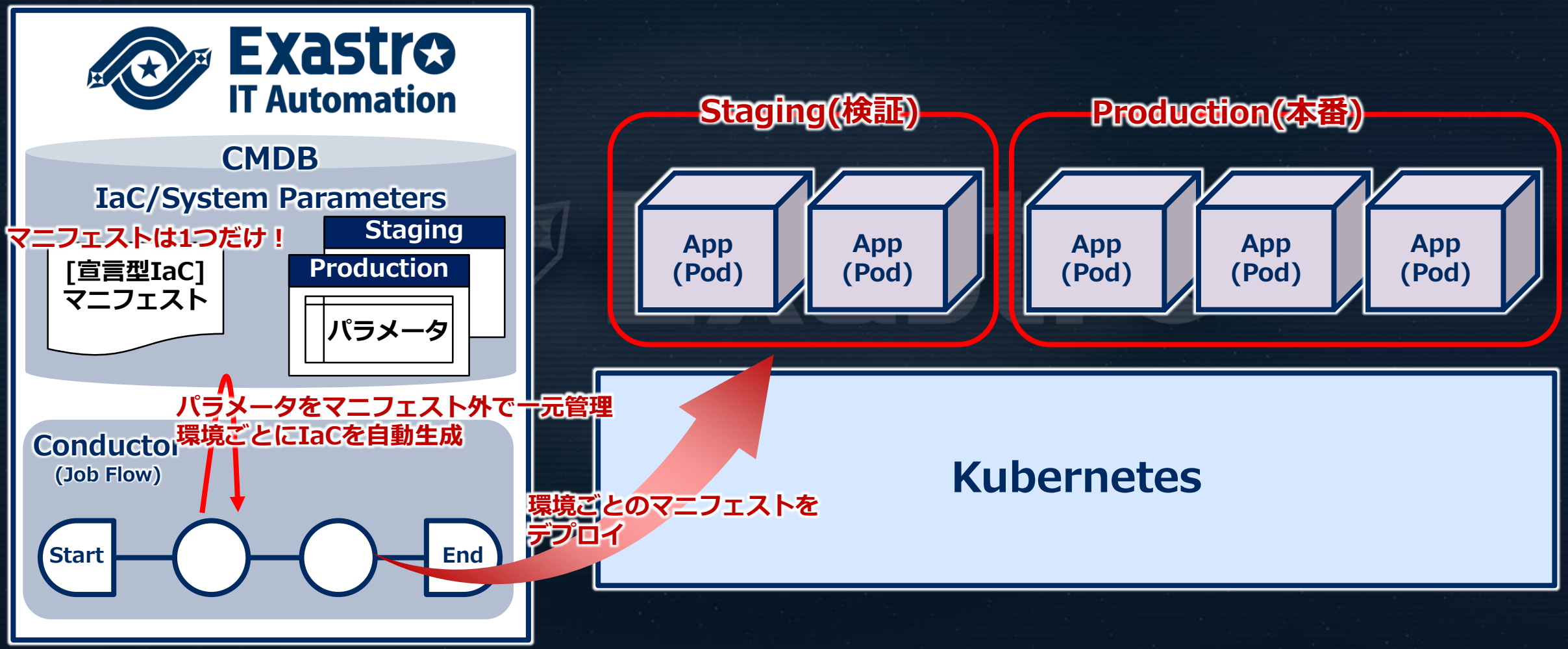
クラウドネイティブでは
環境の変更頻度が高い
→環境差分を減らす工夫が必要

Exastro IT Automation：システム情報をデジタル管理するためのフレームワーク



Exastroが使えます！

✓マニフェストの中で環境に依存するパラメータをExastro上で管理し、
一貫したマニフェストにして環境差分を限定可能！



今回のデモでのポイント

Exastroで開発・本番での環境一致を支援します！

✓マニフェスト（IaC）をExastroでテンプレート管理

1つのマニフェストファイルをテンプレートとして登録する事で
複数の環境へデプロイ可能です！

✓パラメータシートを作成し、マニフェストの環境依存パラメータを外部管理

随時変更するパラメータはIaCのコードを変更することなく、
Exastroから自動でIaCを生成します！

※本日使用する手順は別途公開予定です

[Tips]AnsibleでのKubernetesモジュールについて

Kubernetesマニフェストの適用で、以下のAnsibleモジュールが利用可能

✓k8s – Manage Kubernetes (K8s) objects

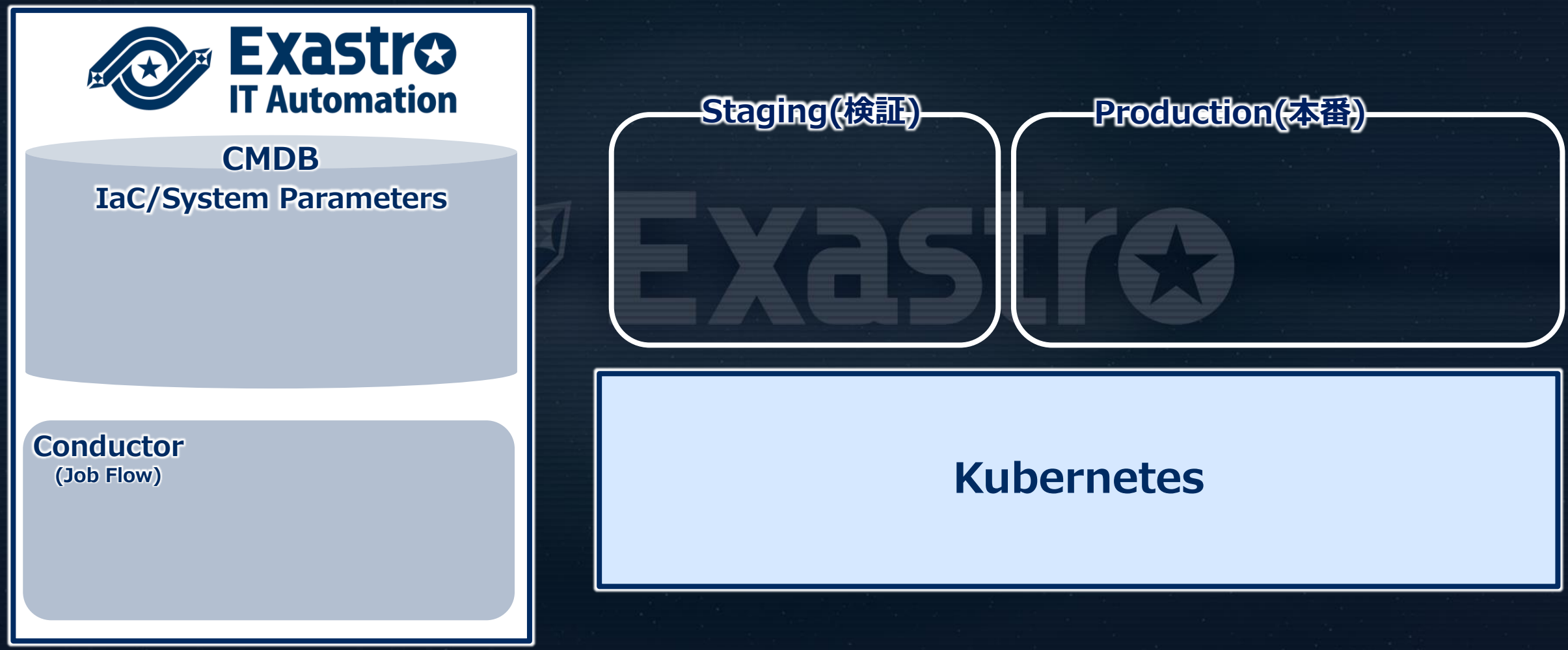
- Ansibleの実行サーバをMasterNodeとする（≒Kubectlコマンド相当の操作が可能）
- 必要な設定
  - Ansibleサーバ(ITAサーバ)：
    モジュールのインストール
    ```
    $ ansible-galaxy collection install community.kubernetes
    ```
  - 実行先サーバ(k8s MasterNode)：
    ライブラリのインストール
    ```
    $ pip install openshift
    ```

参考URL

- https://docs.ansible.com/ansible/latest/collections/community/kubernetes/k8s_module.html
- https://blog.mosuke.tech/entry/2019/08/21/ansible-for-k8s-resources/

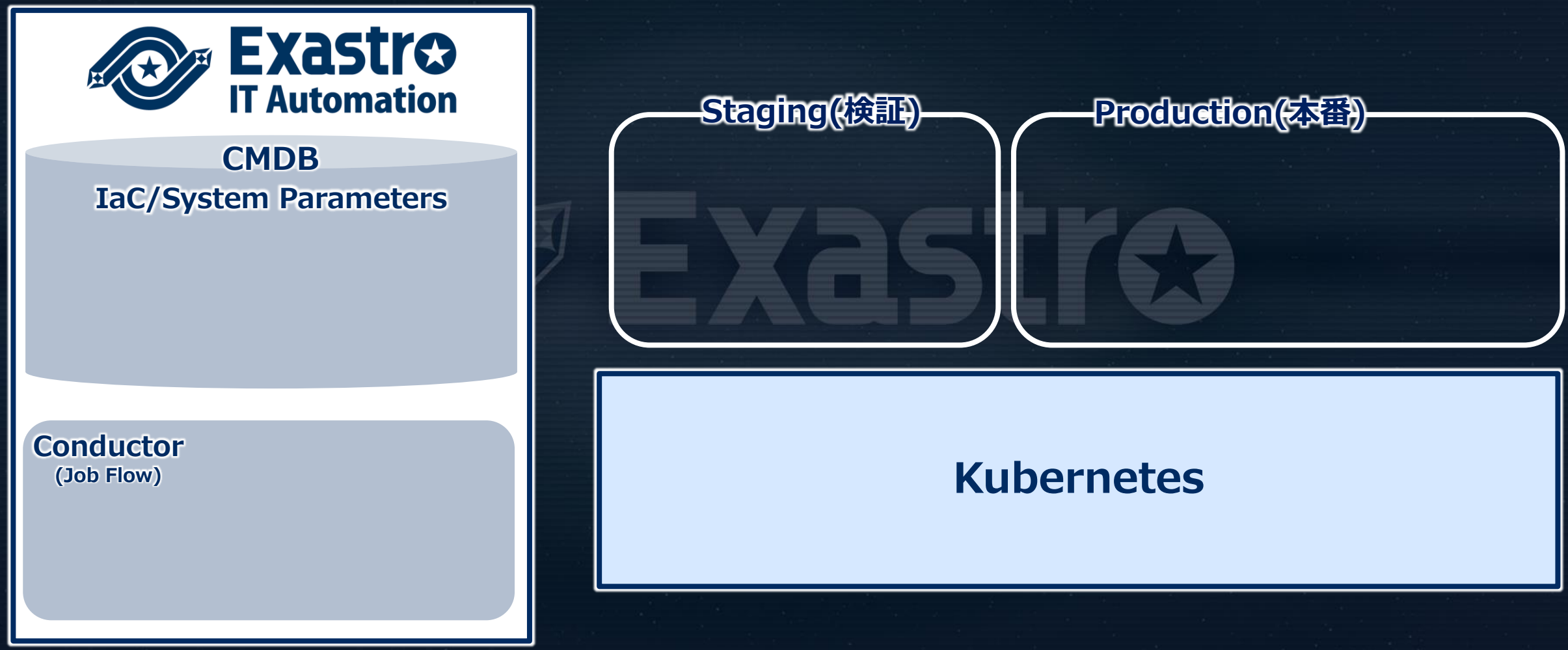
-デモ- 初期状態

✓Exastro：未セットアップ状態（Version 1.7.2）
✓Kubernetes：NameSpaceのみ存在(Staging/Production)



-デモ- 1. 機器一覧の登録

✓Kubernetes MasterNodeを機器一覧に登録します
（AnsibleにてKubernetesを操作するため）

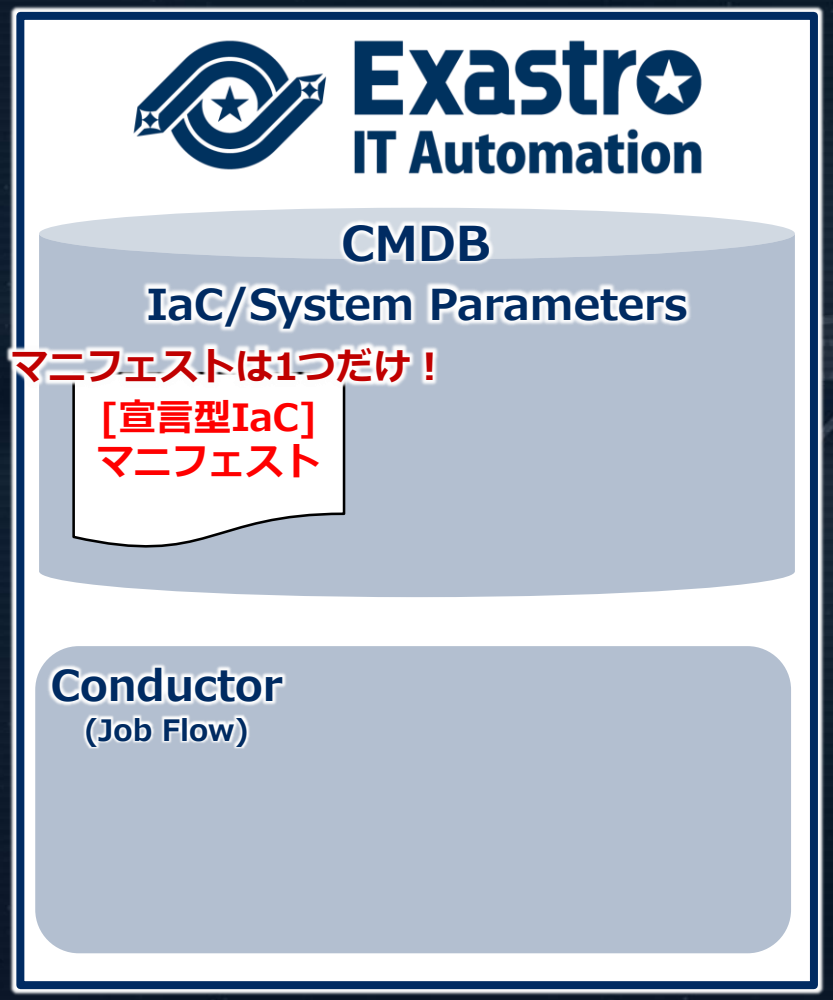


-デモ- 2. Kubernetesマニフェストのアップロード（テンプレート化）

✓利用するマニフェストファイルをアップロードします

Ansible共通>テンプレート管理（※）

※Exastro-ITA_利用手順マニュアル_Ansible-driver.pdf (exastro-suite.github.io)
5.2.4 テンプレート管理



[Tips]今回利用するマニフェスト(nginx_manifest.yaml)について

✓Kubernetes公式のチュートリアルのマニフェストをベースにしています

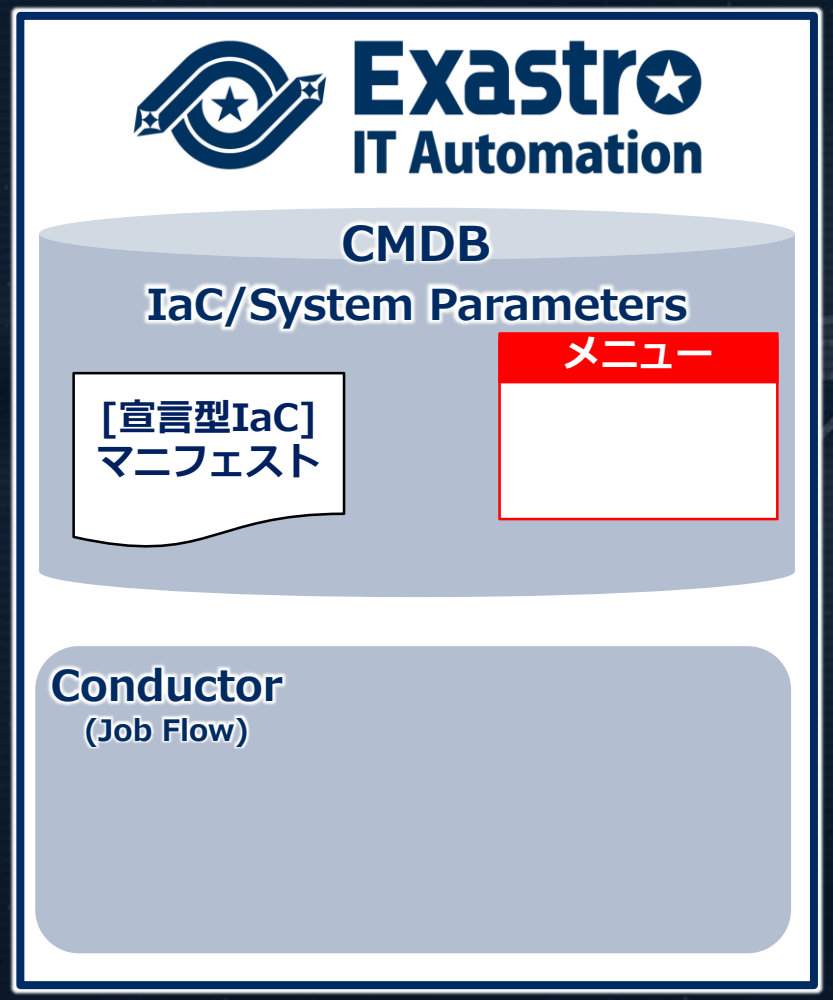
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: {{ replicas }}
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: {{ image }}
        ports:
        - containerPort: 80
～以下略～
```

[ファイルについて]
NGINXのPodを立ち上げるマニフェスト

[変数(カッコについて)]
パラメータはJinja2の形式で記入します
{{ replicas }} ：Podをいくつ立ち上げるか
{{ image }} ：コンテナイメージ(バージョン)
{{ nodeport }} : 接続ポート

-デモ- 3. メニュー（パラメータシート）の作成

✓マニフェスト内のパラメータを管理するためのメニューを作成します
（パラメータの入力自体は次で設定）



[Tips]縦メニューについて

✓縦メニュー

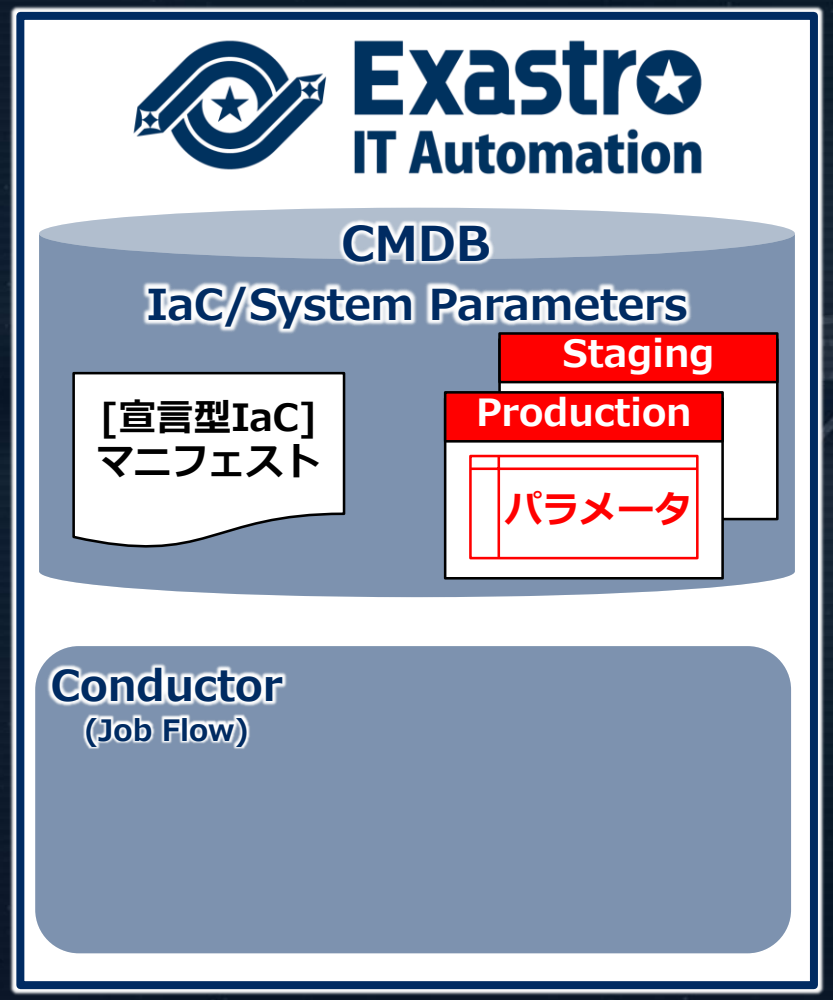
✓繰り返し定義されるようなパラメータを、1つのオペレーション上で同時に利用可能（代入順序の設定が必要）

例）「通常メニュー」と「縦メニュー」それぞれで
「IPアドレス」＋「ドメイン」のパラメータシートを作成した場合

※利用手順マニュアル メニュー作成機能
4 メニュー（パラメータシート/データシート）説明

-デモ- 4. オペレーション(環境)・パラメータ登録

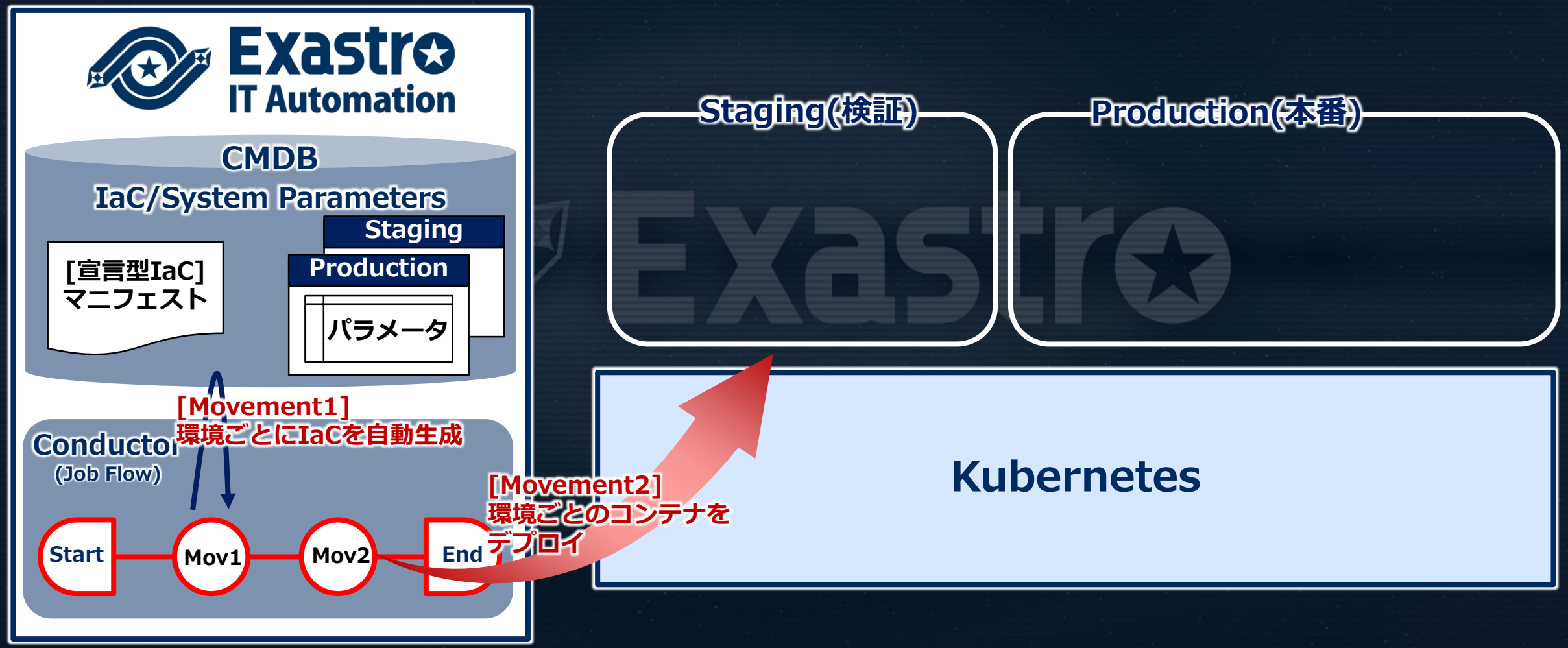
✓StagingとProductionのそれぞれをオペレーションで分割します
✓環境ごとにパラメータを設定します



-デモ- 5. Movement・Conductor作成

✓環境を設定するためのMovementとConductorを作成します

✓Mov1はマニフェストの自動生成、Mov2はPodのデプロイ



[Tips]今回利用するPlaybookについて　Mov1：CreateManifest

✓パラメータを埋め込んだマニフェストファイルを作成します

```
- name: Create Manifest File
  template:
    src: "{{ item.0 }}"
    dest: "/tmp/{{ item.0 | basename }}"
  vars:
    replicas: "{{ item.1 }}"
    image: "{{ item.2 }}"
    nodeport: "{{ item.3 }}"
  with_together:
  - "{{ VAR_TPF_FILE }}"
  - "{{ VAR_replicas }}"
  - "{{ VAR_image }}"
  - "{{ VAR_nodeport }}"
```

[Playbookについて]
マニフェストテンプレートにパラメータを設定し、
環境ごとのマニフェストファイルを作成します。

[各変数について]
{{ VAR_TPF_FILE }}は作成するマニフェスト
テンプレートを指定します。
それぞれのテンプレート内の変数は以下のように
変換されます。
VAR_replicas ⇔ replicas
VAR_image ⇔ image
VAR_nodeport ⇔ nodeport

[Tips]今回利用するPlaybookについて　Mov2：ApplyManifest

✓Podをデプロイします

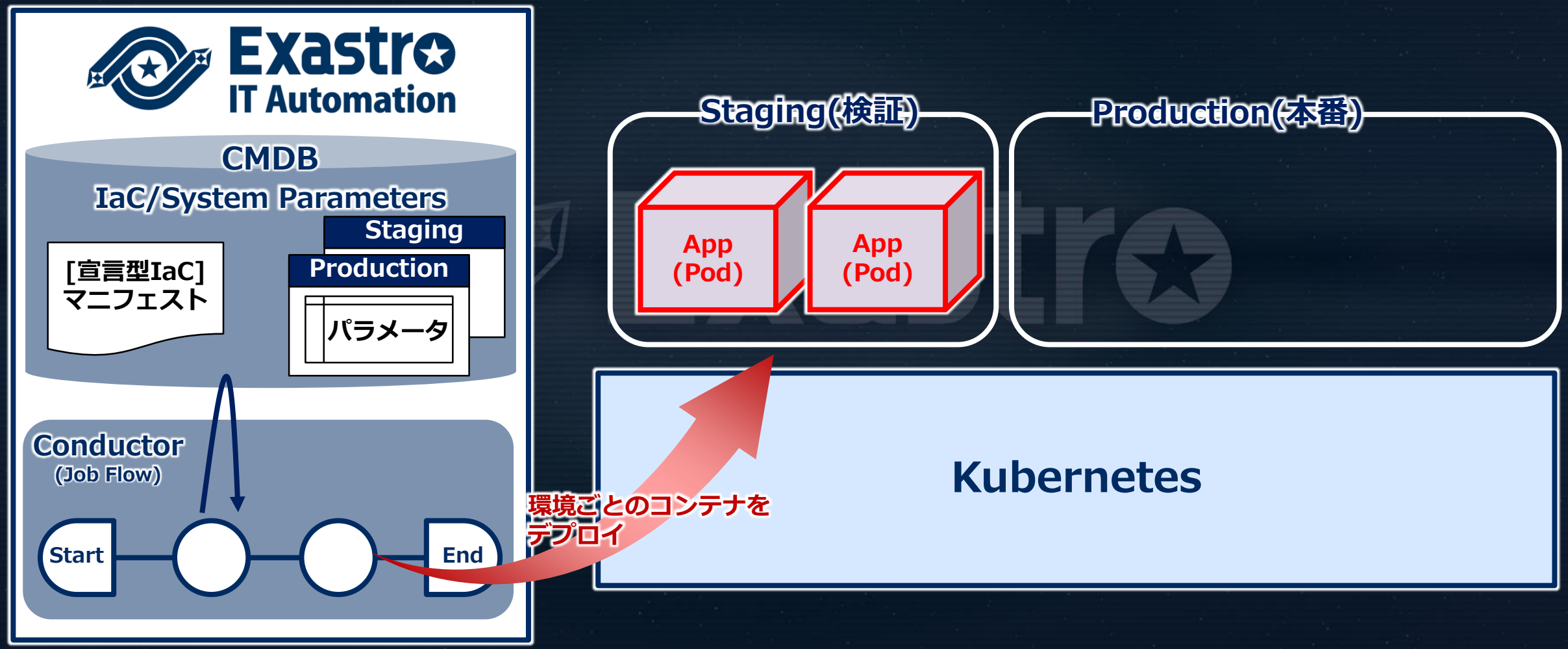
```
- name: Apply Manifest
  k8s:
    namespace: "{{ item.1 }}"
    state: present
    src: "/tmp/{{ item.0 | basename }}"
  with_together:
  - "{{ VAR_TPF_FILE }}"
  - "{{ VAR_namespace }}"
```

[Playbookについて]
Manage Kubernetes (K8s) objectsモジュールを利用して、Mov1で生成されたマニフェストファイルを環境へ適用していきます。

説明②
{{ VAR_TPF_FILE }}は適用するマニフェストファイルを指定し、{{ VAR_namespace }}は適用するnamespaceを指定します。

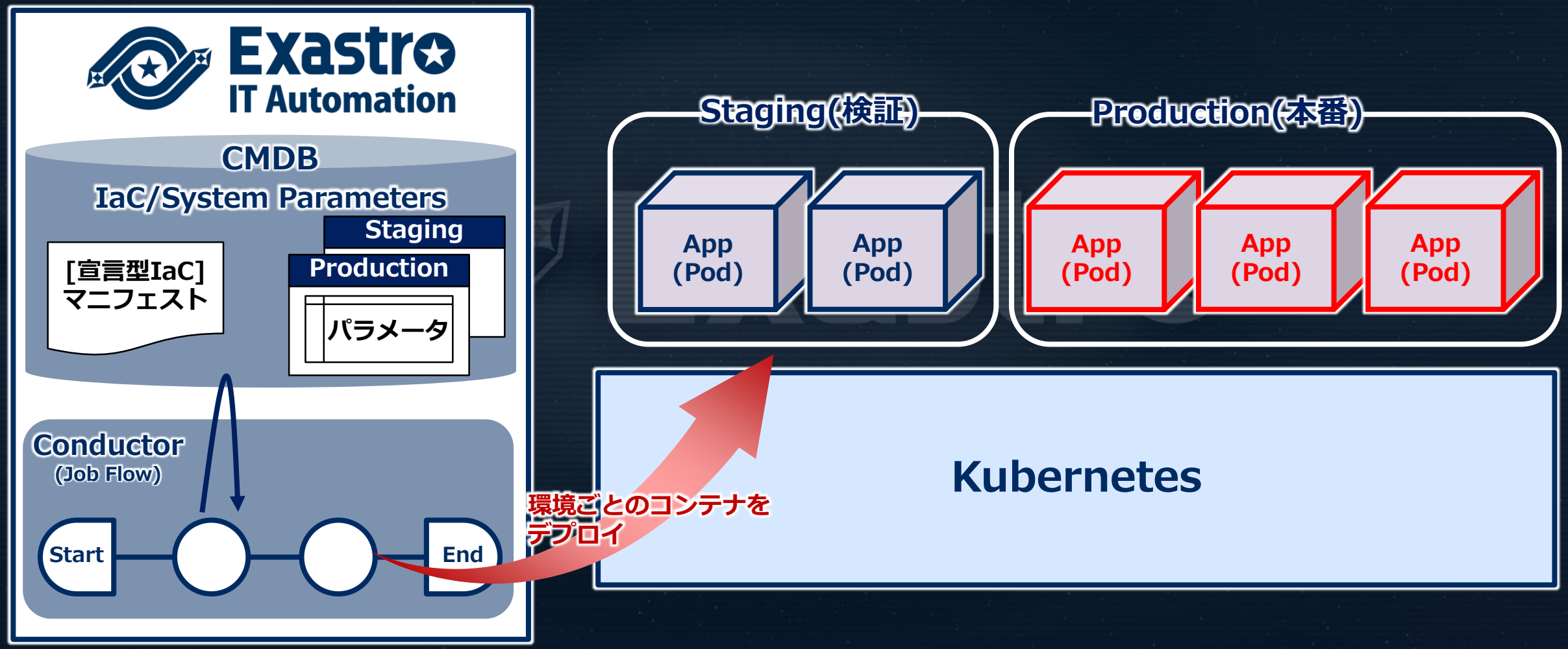
-デモ- 6. Conductorの実行(Staging)

✓オペレーションStagingを実行して、Staging環境のPodを作成します



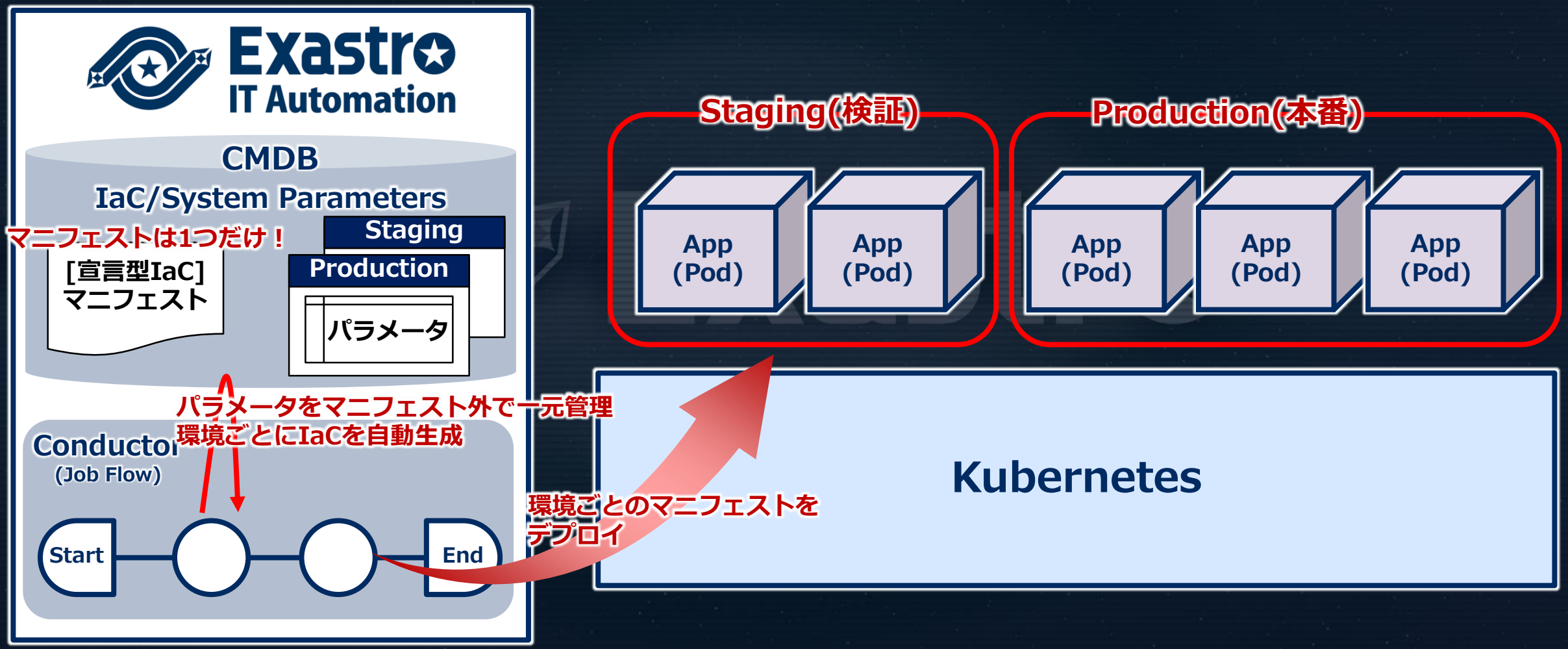
-デモ- 7. Conductorの実行(Production)

✓オペレーションProductionを実行して、Production環境のPodを作成します



まとめ

✓マニフェストの中で環境に依存するパラメータをExastro上で管理し、
一貫したマニフェストにして環境差分を限定可能！



[参考]Exastro IT Automation 関連記事

✓Qiita記事

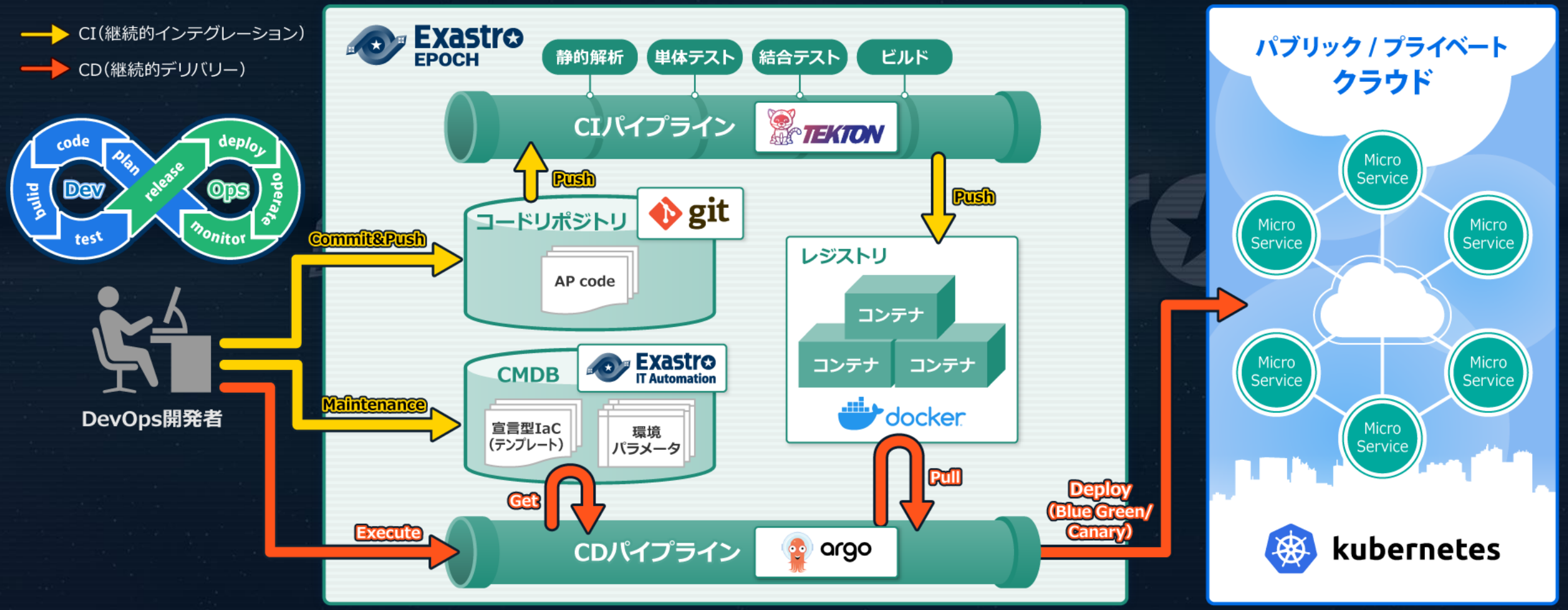
- Exastro IT Automationをインストールしてみた（v1.7.2）
- Exastro IT Automationのキホンの”キ”
- Exastro IT Automation ver1.6.0のキホンの"キ“

✓公式サイト

- Exastroコミュニティ (Github)

[参考]8月のウェビナーにて、Exastro EPOCHをご紹介予定！

Exastro EPOCHは「クラウドネイティブシステム開発を加速するためのフレームワーク」です




Exastro