デジタル化に向けたクラウド活用セミナー

オープン・テクノロジー編
インフラ環境に左右されないシステムの実現

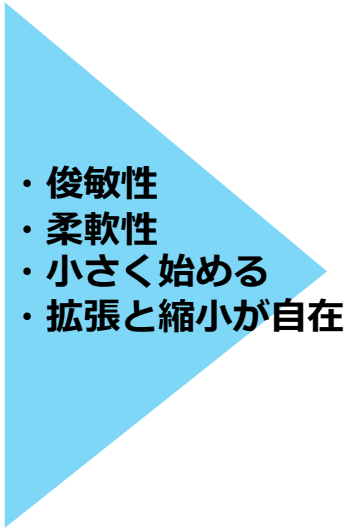
Make your deploy-anywhere future possible

デジタル変革を推進していくには

デジタル変革（DX）の目指す先は、競争に「勝ち残ること」。
その実現には、これまでとは違う新たなテクノロジーが求められます。

DXが目指す先は「勝ち残ること」

- 早く作り、**市場に投入するスピード**
- **新たな顧客体験**を生み出す
- 顧客の**要望をいち早く吸収し、対応**する
- **障害を迅速に解決**し、顧客満足を向上
- 自社の**データ資産の新たな価値**と活用
- 全く**新しい業界への参入**



変化への対応力を実現するテクノロジー

- クラウドネイティブ
- マイクロサービス
- アジャイル開発
- コンテナ技術
- DevOps
- Data & AI

デジタル変革を支えるオープンテクノロジー

エンタープライズレベルでDXを推進していくには
複数の製品/サービスを利用するため、**オープンテクノロジー**が重要

変化への対応力を実現するテクノロジー

- クラウドネイティブ
- マイクロサービス
- アジャイル開発
- コンテナ技術
- DevOps
- Data & AI

求められること

- パブリッククラウド・オンプレミス問わず稼働できる
- ベンダーロックインが除外されている
- 標準化された運用保守、制御可能なセキュリティ対策
- 優れたテクノロジーを常に採用していける

⇨ オープンソースのコラボレーションから、優れたテクノロジー、変化への対応力のあるテクノロジーが生まれる

デジタル変革を実践していくには

クラウドネイティブ技術とは？

パブリッククラウド、プライベートクラウド、ハイブリッドクラウドなどのダイナミックな環境において、スケーラブルなアプリケーションを構築および実行するための能力を組織にもたらします。

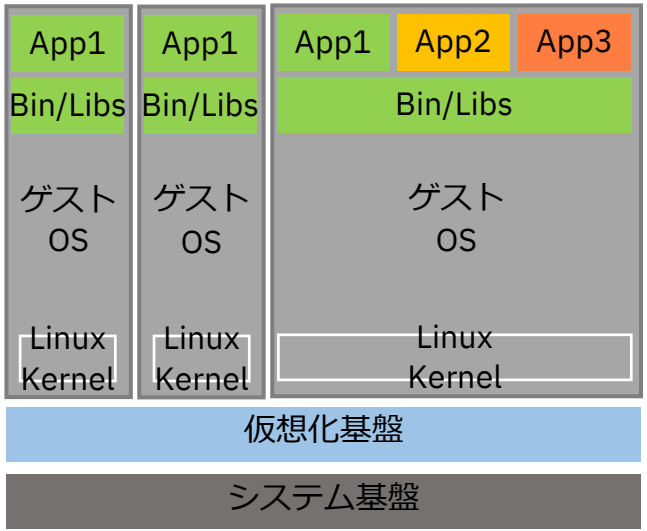
代表的なアプローチとして

- コンテナ
- マイクロサービス
- サービスメッシュ
- イミュータブルインフラストラクチャ

これらにより「回復性」「管理力」「可観測性」を備え、「自動化」と組み合わせることで、変更を最小限の労力で頻繁かつ予測どおりに行うことができます。

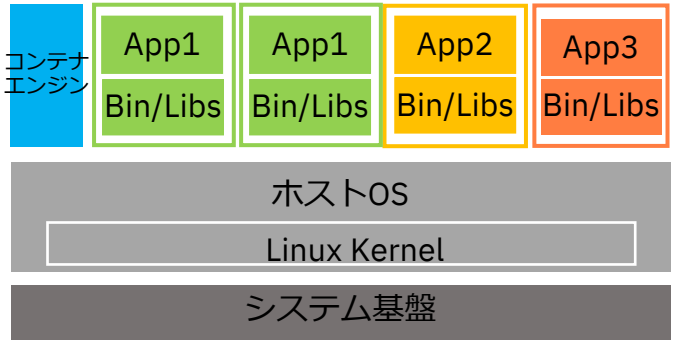
コンテナ化する価値

■従来のVM型仮想化



- ゲストOSのオーバーヘッド
- 低いシステム利用効率
- スケーリング・スピード
- ポータビリティの課題

■コンテナ型仮想化



- Linuxカーネルを共有
- Linuxカーネルの仕組みでコンテナ間の分離を実現
- **高いシステム利用効率**
- **高速起動、スケーリングのスピード**
- **高いポータビリティ**

なぜ コンテナが注目されているか

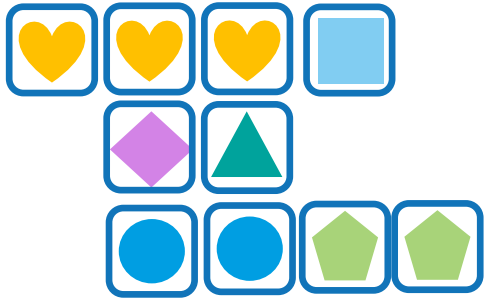
従来に比べ、**拡張性**にすぐれ、**変更改善しやすい**システムを構築できる

モノリシック

必要となる機能を１つのバイナリーで提供
各機能も互いに結び付きが強い

VM型の仮想化環境/IaaS で稼働

マイクロサービス



アプリケーション機能（サービス）ごとに分割し、
サービスごとにパッケージング
各機能の依存性はできるだけ分離
各サービスはRESTなど軽量プロトコルで通信

コンテナー環境で稼働

なぜ コンテナが注目されているか

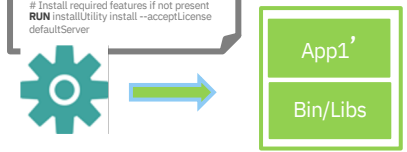
コンテナ活用により、**シンプルに 安心して リリース**が可能 に

– Immutable Infrastructure の実現

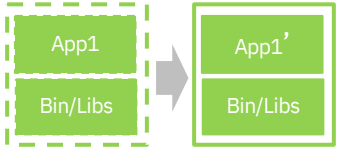
手作業による再現性のない作業（の繰り返し）
+それを補完する **大量のドキュメント**

Dockerfile で常に
クリーンな環境を構築
＋テストで動作確認

```
FROM berty:webProfile7
MAINTENER iwashina@mycompany
COPY /work/productionConfig.xml
/config/
# Install required features if not present
RUN installUtility install --acceptLicense
defaultServer
```



まるごと
クリーンな環境で
置き換える



– コンテナ可搬性によるリリースの容易性

- 環境をまたいでもリリース漏れが発生しない

– フォールバックの容易性

- リリース後障害が発生した場合も
  すぐに前のバージョンに戻せる

⇨ **ビジネスを**
より安定した基盤で
稼働させることができる

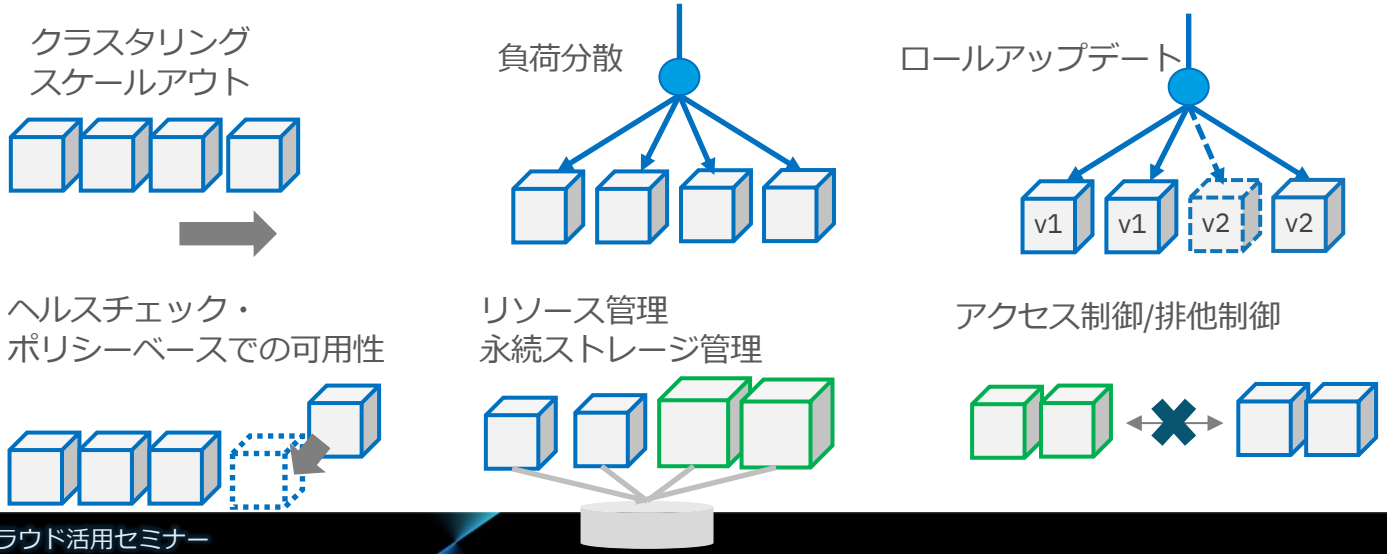
コンテナ・オーケストレーションの必要性

業務量に応じたコンテナの動的な起動／停止、複数コンテナへの負荷分散、リソース割り当ての管理といった機能が求められる

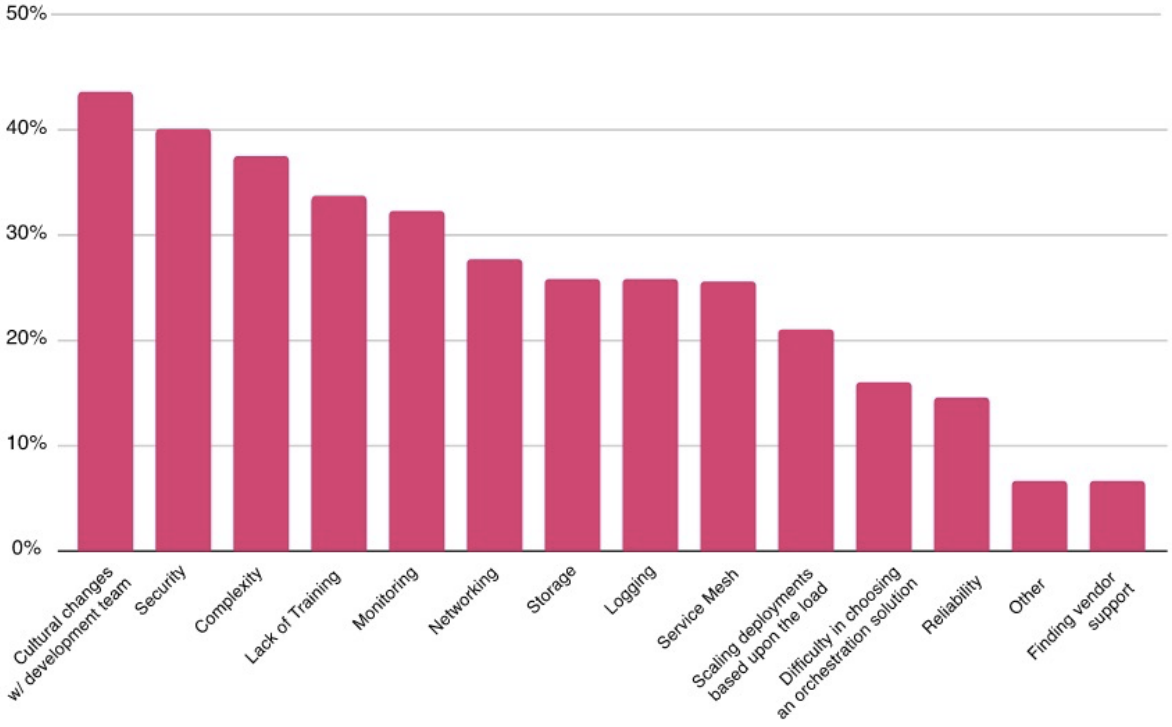
コンテナ・オーケストレーションが必要

kubernetes

■ ポリシーベースの運用管理（宣言的運用）



コンテナ運用時の課題は？



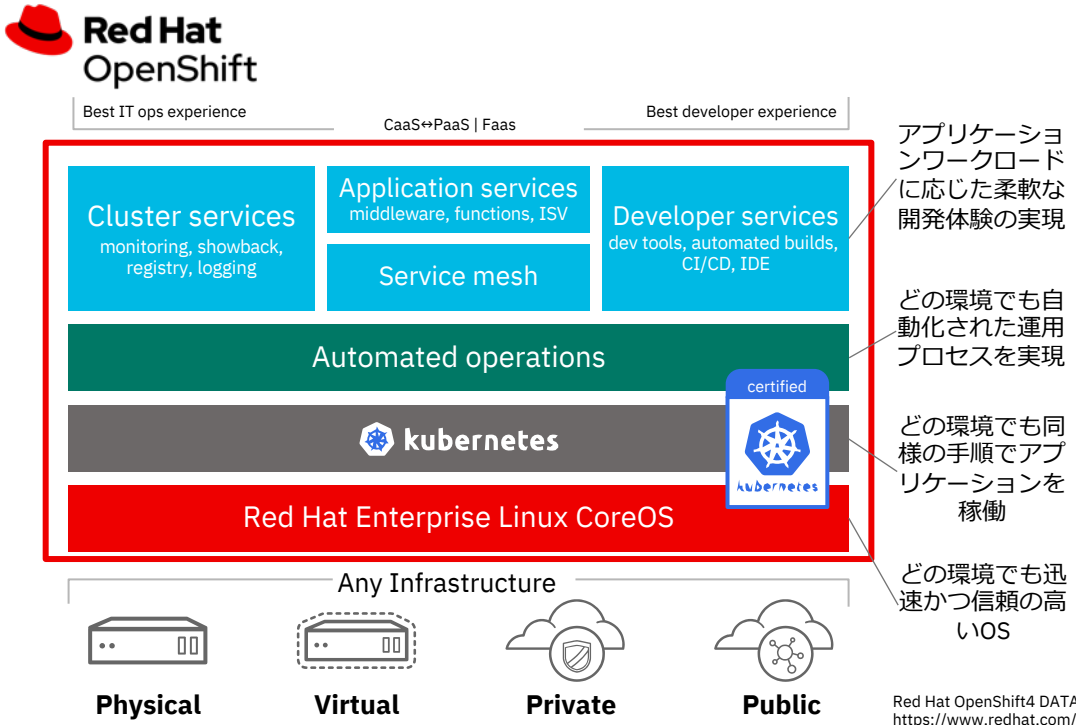
主に挙げられているものは、
- チーム文化の変革
- セキュリティー
- 複雑さ、学習コスト

⇨コンテナ（Kubernetes）を正しく運用するのは難しい

CNCF Survey 2019
https://www.cncf.io/wp-content/uploads/2020/03/CNCF_Survey_Report.pdf

Red Hat OpenShift Container Platform

Kubernetesをエンタープライズ向けに提供するコンテナ・アプリケーション・プラットフォーム



アプリケーションワークロードに応じた柔軟な開発体験の実現

どの環境でも自動化された運用プロセスを実現

どの環境でも同様の手順でアプリケーションを稼働

どの環境でも迅速かつ信頼の高いOS

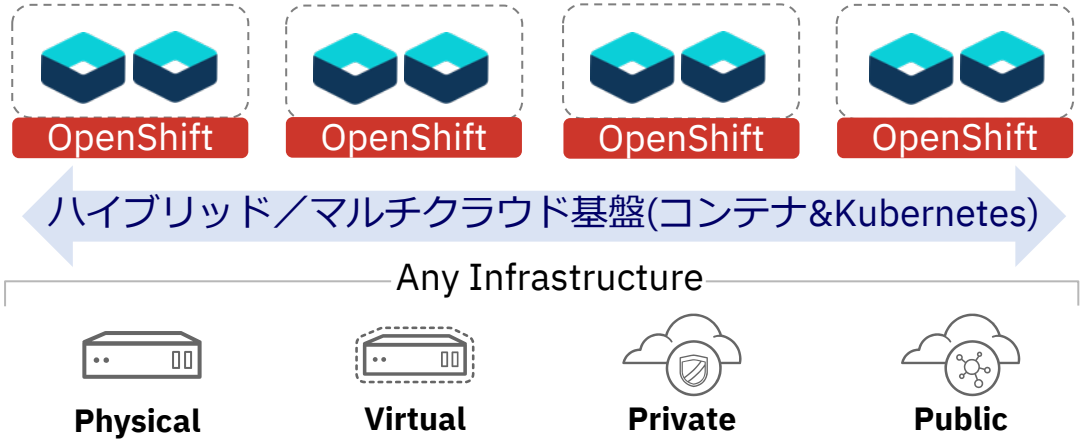
1. 完全に統合され自動化された**統合アーキテクチャ**
2. Kubernetesベースで**オンプレとクラウドのワークロードをシームレス**に結びつける
3. クラウドインフラからOS,アプリケーションサービスまで**自動インストール**
4. OpenShiftプラットフォームとアプリケーションの**更新が容易**
5. 必要なリソースを**オートスケーリング**

Red Hat OpenShift4 DATASHEET
https://www.redhat.com/cms/managed-files/cl-openshift-container-platform-datasheet-f21298pr-202001-en.pdf

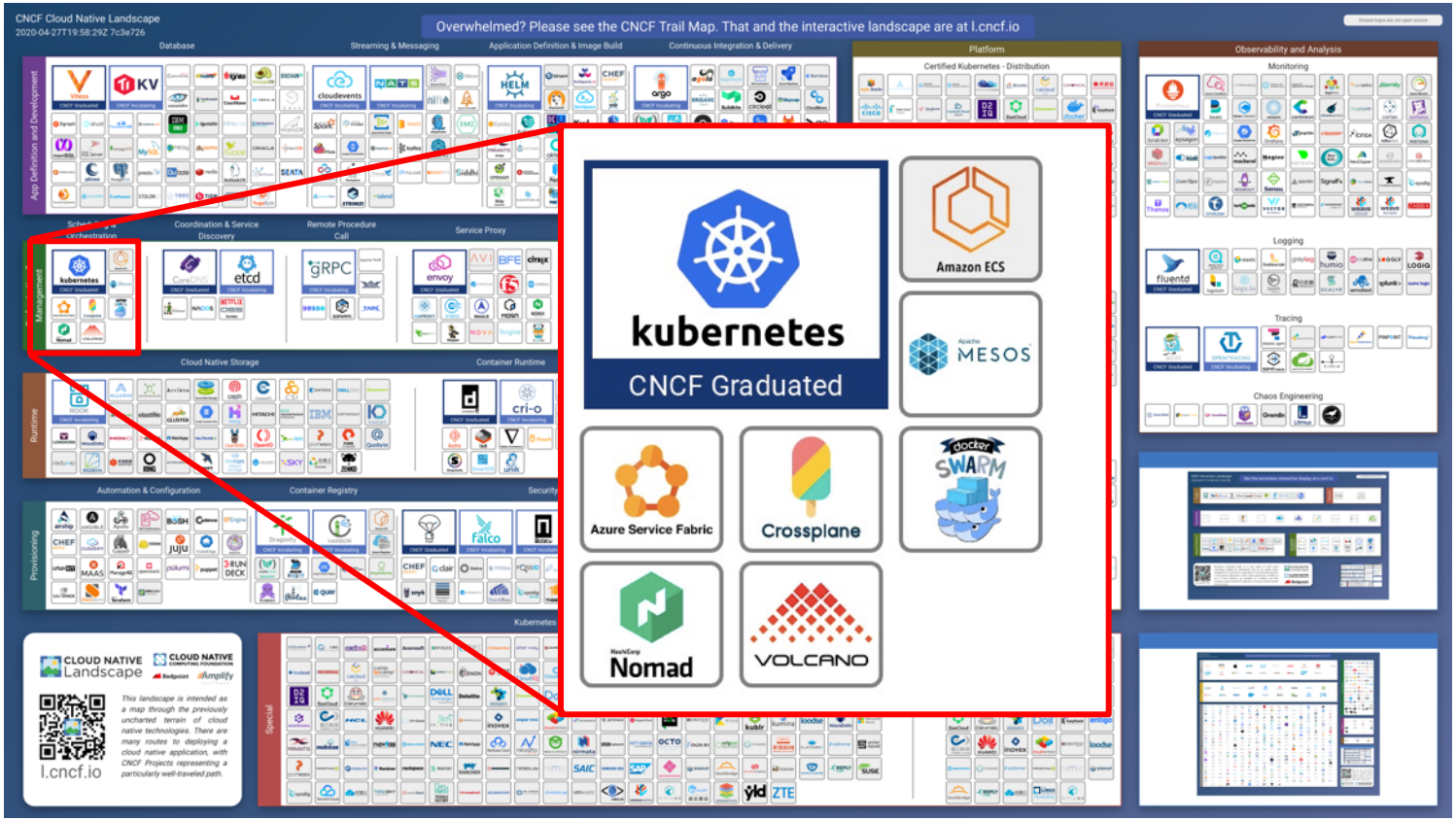
オープンなハイブリット/マルチクラウド基盤の実現

ハイブリッド/マルチクラウド環境を意識しない、統一したコンテナ運用が実現可能となる。

運用プロセスも含めたポータビリティーの実現



クラウドネイティブ技術を支えるツール群



CNCF Landscape
https://landscape.cncf.io/

Cloud Native Trail Map

■ クラウドネイティブ技術を実現するための具体的なステップを提示

1. コンテナ化

- 一般的には Docker コンテナで実施
- いかなるサイズのアプリケーションも依存性もコンテナ化可能
- 時間が経つにつれ、アプリケーションを分割し、マイクロサービス化したくなるだろう

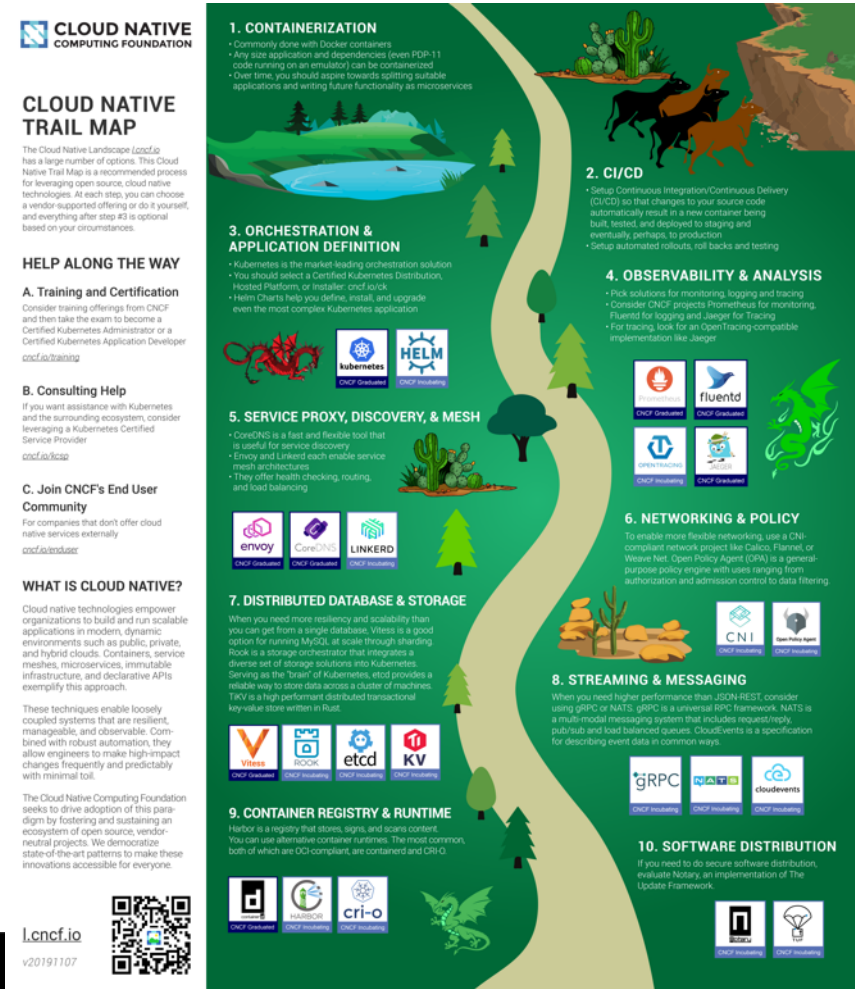
2. CI/CD（継続デリバリの仕組み）

- 継続インテグレーション・継続デリバリの仕組みをセットアップ
- コード変更が 新しいコンテナ・イメージとして、自動的にビルドされ、テストされ、デプロイされるように

3. オーケストレーション&アプリ定義

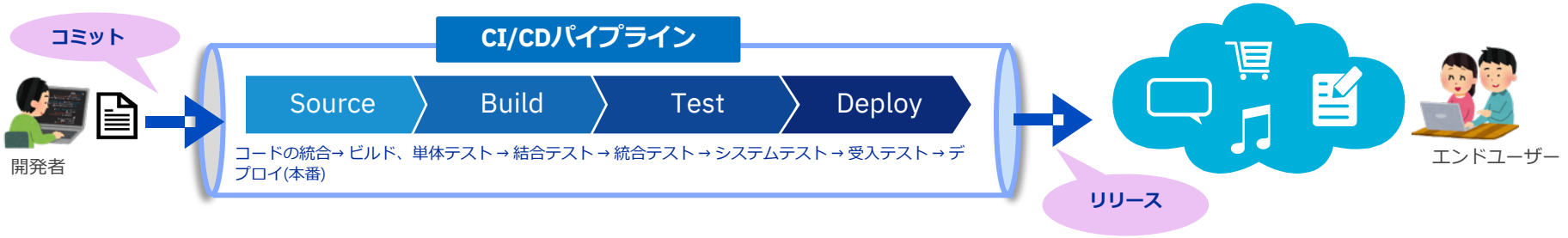
- 互換性試験に合格している Certified Kubernetes 環境を選択
- もっとも複雑なアプリケーションであってもHELMで 定義して、導入して更新していくことが可能

Cloud Native Trail Map
https://github.com/cncf/trailmap/blob/master/CNCF_TrailMap_latest.pdf



CI/CDパイプライン

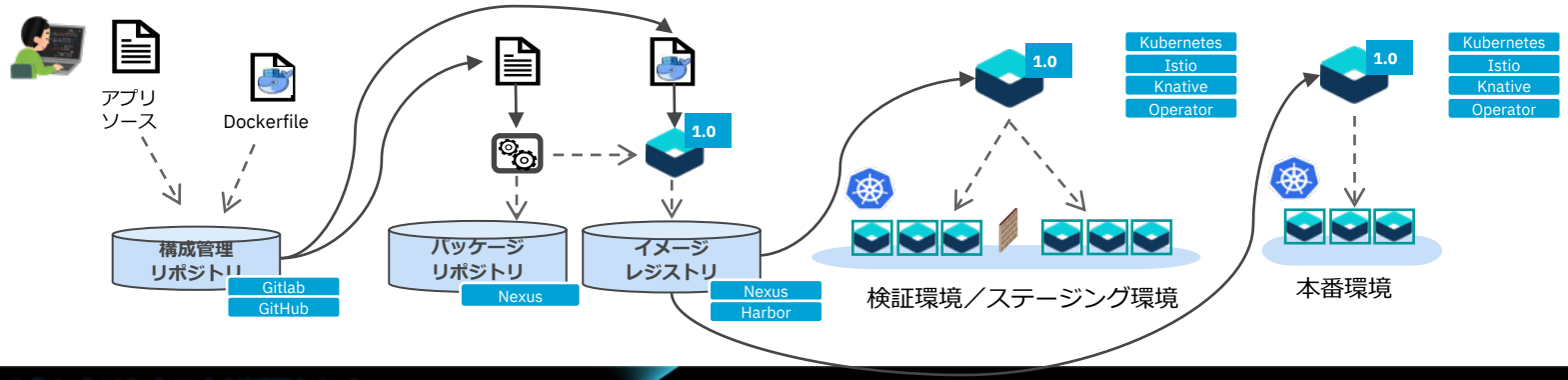
- 一連のプロセスを「1本のパイプライン」に見立て、成果物とプロセスを管理する考え方
  - 一連のプロセス = 例：開発における「ビルド」「デプロイ」「テスト」「リリース」のプロセス
  - プロセスを自動化し、正確性とスピードを上げる
  - リリース候補は必ずパイプラインを通過させ、品質を確保する
- CI/CDパイプライン（デリバリー・パイプライン）
  - 継続的インテグレーション（CI）／継続的デリバリー（CD）のためのパイプライン
  - 上記の開発プロセスを自動化し、開発生産性や改善スピード・リリース頻度を向上



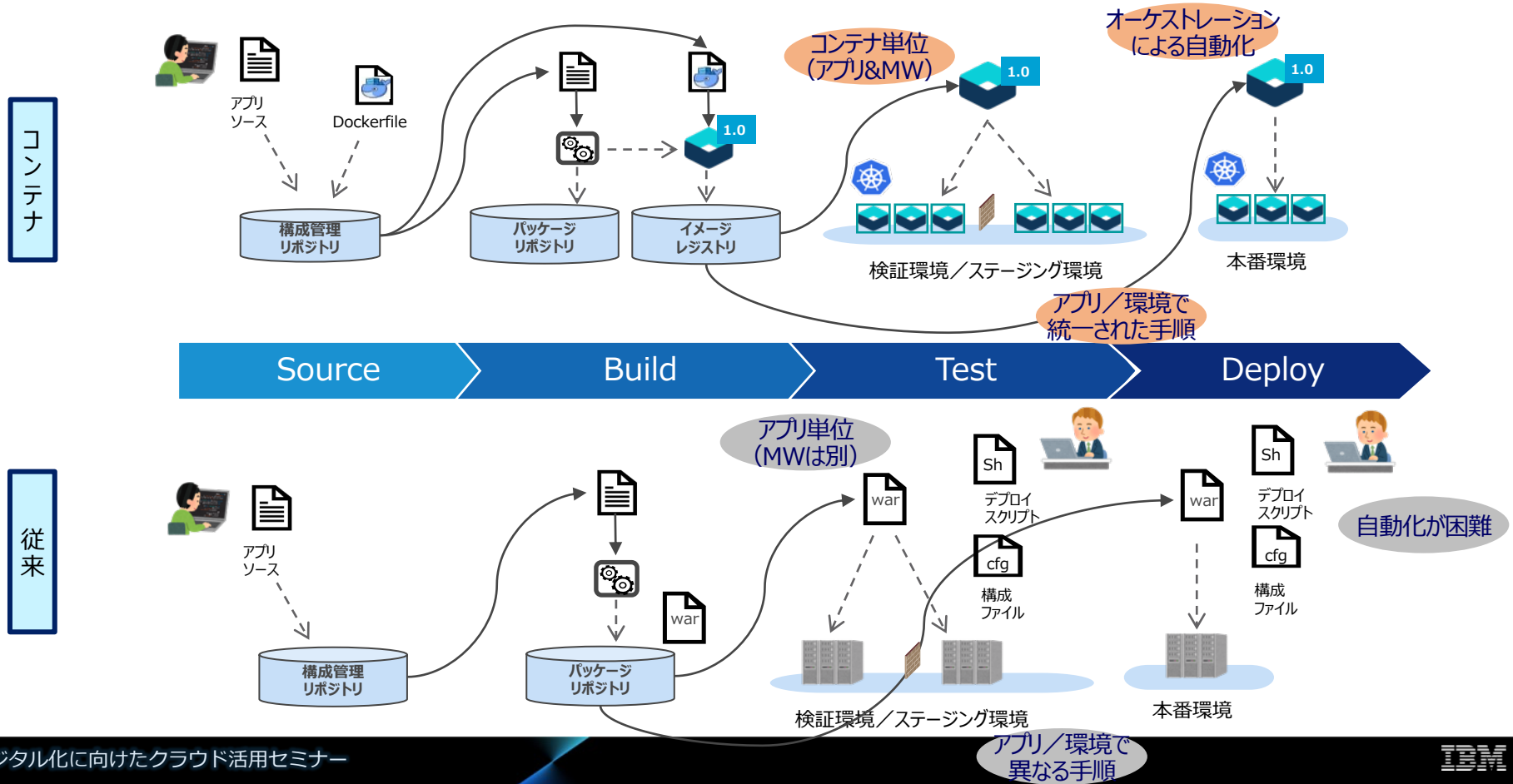
コンテナベースの開発の流れ

- アプリケーションと実行環境をパッケージ化した**コンテナ単位でデプロイ**
- ビルド済のコンテナイメージを**複数の実行環境に統一された方法でデプロイ**
- コンテナ・オーケストレーション基盤やツールによる**自動化**

Source	Build	Test	Deploy
・ベースイメージの取得 ・アプリの設計・実装 **・Dockerfileの作成** ・manifestの作成 ・リポジトリへの登録	・アプリのコンパイル ・単体テスト ・アプリのパッケージング **・コンテナイメージのビルド** **イメージのレジストリ登録**	**・検証／ステージング環境へのデプロイ** ・統合／受入テストの実施	**・本番環境へのデプロイ**

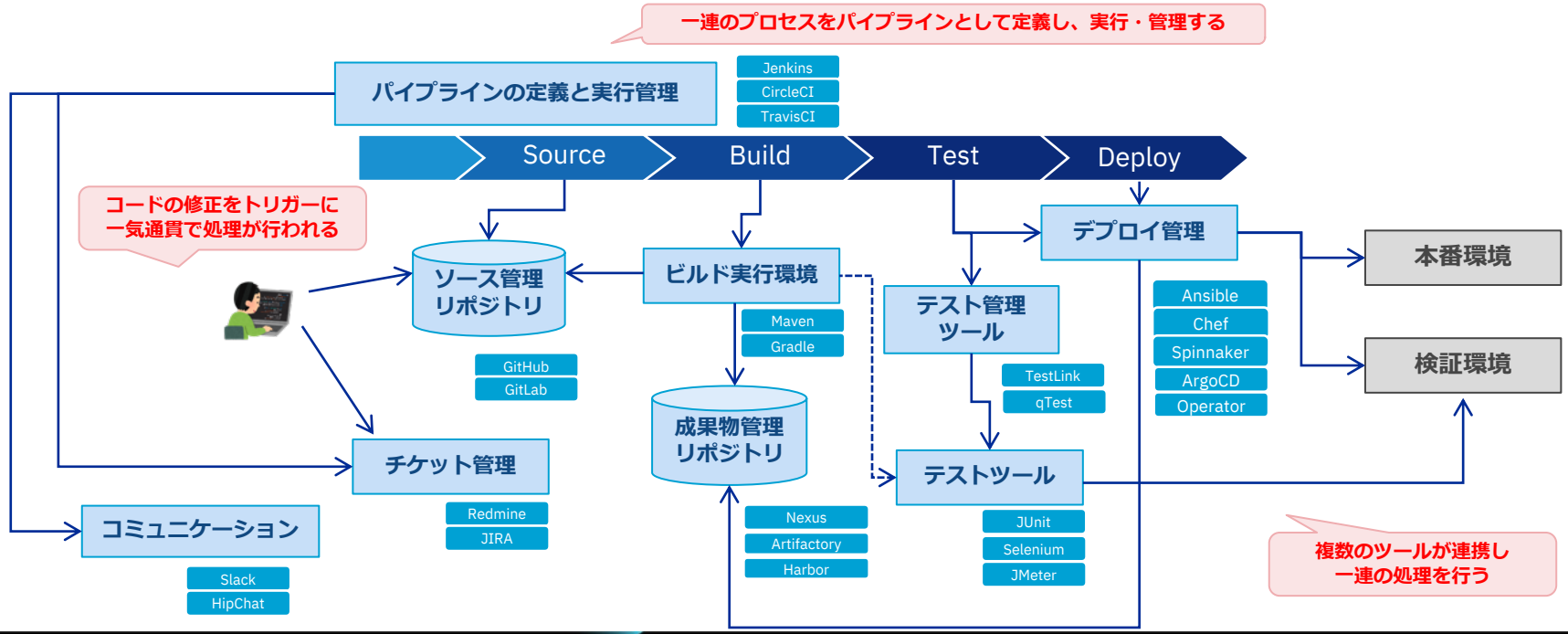


従来（非コンテナベース）との流れの比較

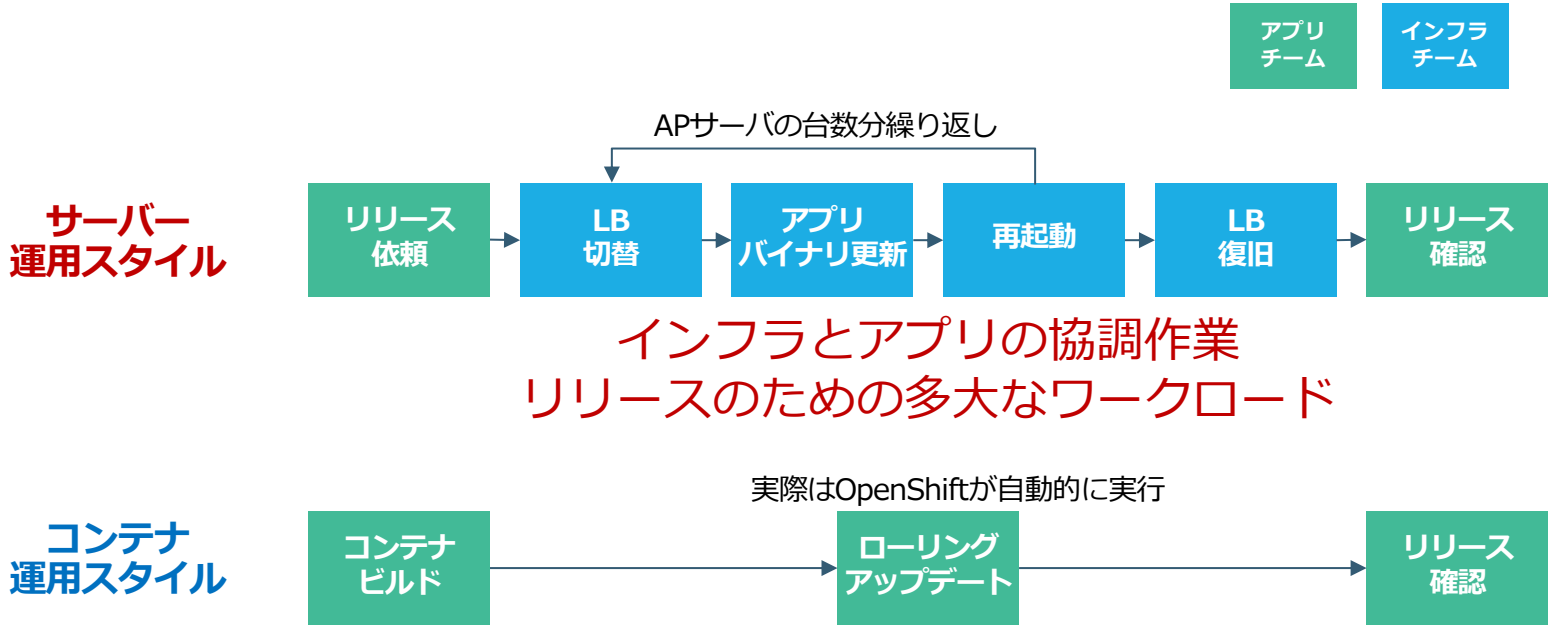


CI/CDパイプラインの構成

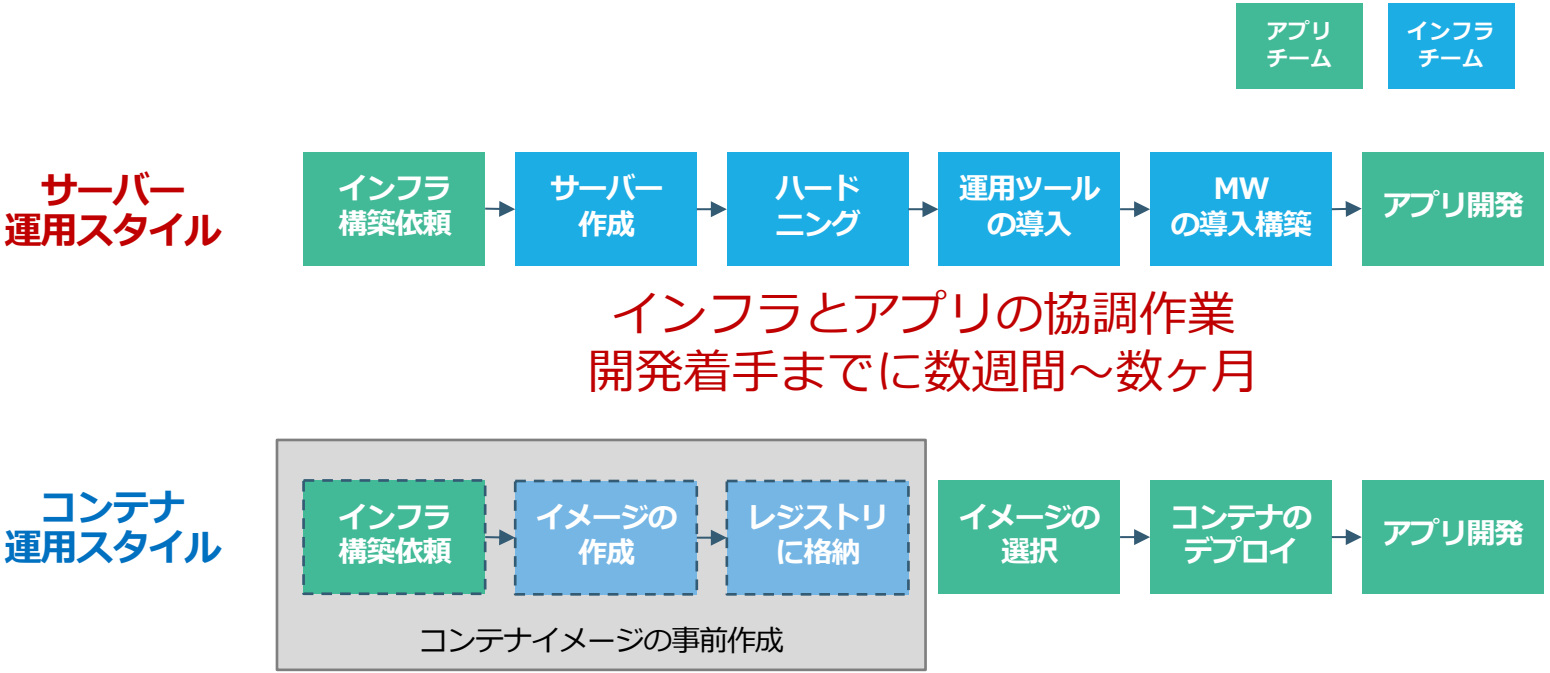
- 複数のツール/ミドルウェア/サービスを組合わせて構成する
  - テストの自動化はどこまで組み込むか、デプロイ処理の実装はどこまで自動化するか。
  - オープンソースを組合わせるか、Cloud Provider提供のサービスを使うか、ツールセットを使うか。



コンテナ化による変化：リリース作業



コンテナ化による変化：環境構築

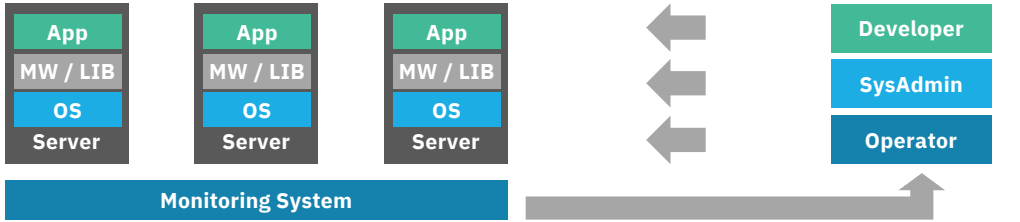


コンテナ化による変化：障害対応

アプリ チーム　インフラ チーム

サーバー運用スタイル

サーバに直接ログインしてトラブルシュート

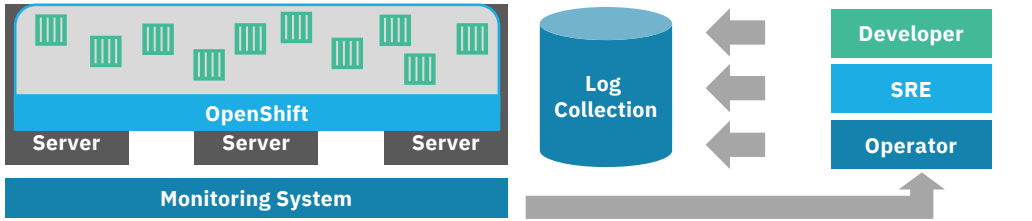


監視アラートをトリガーとして

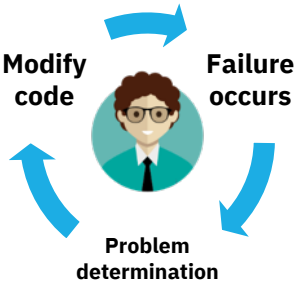
変更記録は帳票として蓄積されていく
低再現性、低品質、非効率

コンテナ運用スタイル

コンテナにはログインせずログを見る



監視アラートをトリガーとして

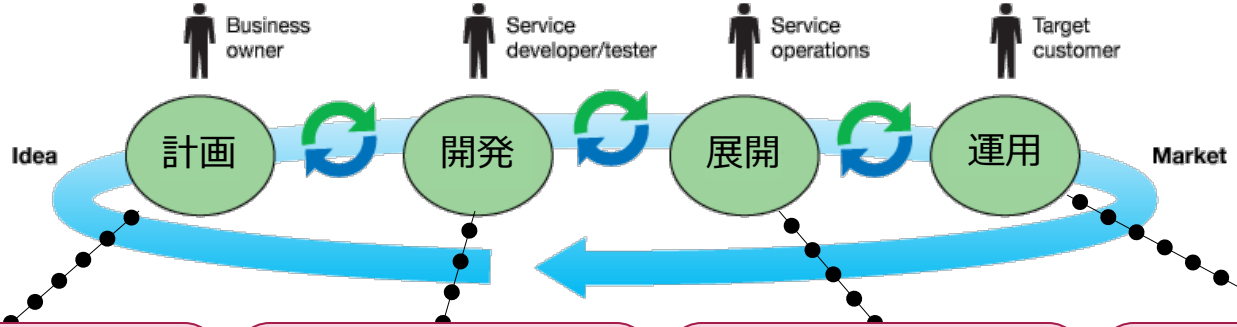


変更はコードとして蓄積されていく
冪等性、高品質、高効率

- ✓ 問題があればOpenShiftのコンソールから修復作業
- ✓ 必要があればコンテナを再ビルドしてリリース

DevSecOps(継続的な開発運用とセキュリティー)の実践

DevOps 開発モデルをベースに、セキュリティーの検査・対策を可能な限り開発時に行う（Shift Left)



	計画	開発	展開	運用
脅威	- 不適切なベースイメージの選択 - 不正・脆弱な外部ライブラリの利用 - 運用終了時の監査データ誤削除・暗号鍵の未廃棄	- イメージレジストリへの攻撃　（イメージ改ざん・漏洩） - イメージ経由の秘密漏洩	- 脆弱なイメージの誤デプロイ - 隣接コンテナ間の攻撃 - ホストへの攻撃	- 不正アクセスによるアプリ改ざん - デプロイ後に発見された脆弱性への攻撃 - 出所不明なコンテナ
対策	- 適切なベースイメージ・外部ライブラリの選定 - 適切な運用・廃棄プランの策定	- ベースイメージの署名検証 - イメージレジストリの保護 - 適切な秘密管理	- 適切なイメージデプロイ - コンテナ権限最小化 - コンテナ隔離 - 秘密の保護	- 変更検出・脆弱性検査 - インフラ保護 - コンテナ間隔離 - 不正コンテナ検出

オープンテクノロジーを活用したIBMのソリューション

IBMは、オープンテクノロジーを活用して、構想策定～開発～運用/保守までを支援するソリューションを提供し、お客様のデジタル変革をお手伝いします。

構想策定	• コンテナ化アセスメント • DevSecOpsプロセス＆ツールアセスメント • DevOpsコンサルティング
開発	• クラウド・アプリケーション構築サービス • ソースコード自動生成（IBM 超高速開発ソリューションAWAG） • アジャイル開発 • DevOps as a Service
運用/保守	• クラウドアプリケーションの運用・保守 （ソリューション・オペレーション・センター）

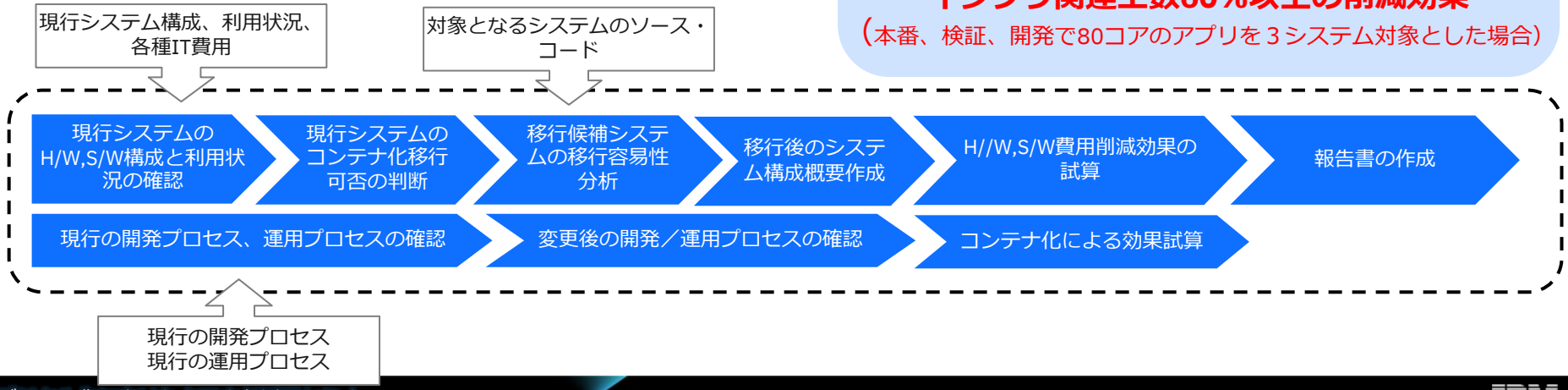
事例：コンテナ化アセスメント

【概要】

- ✓ 既存のアプリケーションのH/W,S/W構成、利用状況を基にコンテナ化の実現可能性を検討
- ✓ 個々のアプリケーションのソースを基に、コンテナ化に必要な改修箇所の明確化を図る
  - 解析ツールの利用により、コンテナ化に必要な改修箇所を特定可能
- ✓ コンテナ化による費用対効果の算定
  - H/W費用、S/W費用の削減効果の算定
  - 運用・保守に関わる作業の自動化によるコスト削減および期間短縮効果の算定
- ✓ 報告書の作成

【結果：例】

コンテナ化により、
CPUコア数20%以上、SW費用40%以上、
インフラ関連工数60%以上の削減効果
（本番、検証、開発で80コアのアプリを３システム対象とした場合）



事例：アプリケーション構築サービス 他

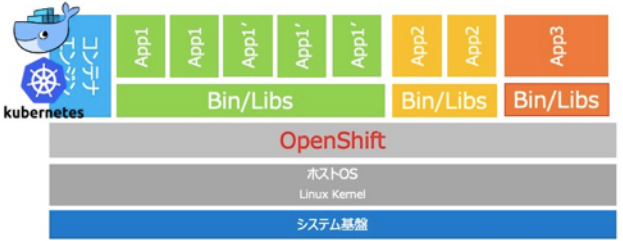
お客様の課題

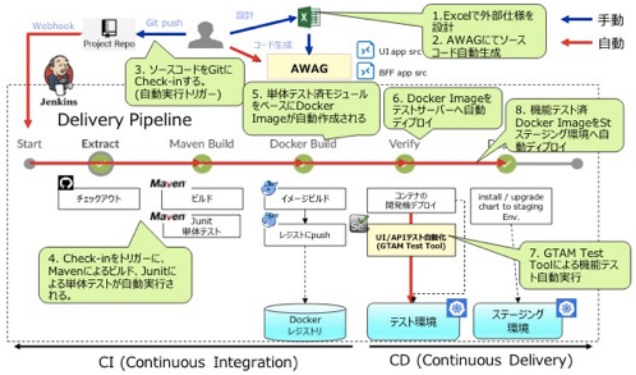
- 機能拡張の対応スピードが遅い
- 開発サイクルが長く、効果的な開発プロセスがない
- 現行アーキテクチャによりシステムが安定しない
- ピーク時の応答時間が十分でなく、解決のために HWを増強してきた

対応ソリューション

- **コンテナ技術の活用**
  - コンテナを活用した並列化、高可用性を実現
  - ベンダー依存の低減と高信頼性の両立を図り、**OpenShiftによるコンテナ基盤の採用**

- 業務要件へ柔軟に対応できる仕組みの導入
  - ✓ 継続的な機能拡張を効率的に実現するため、**アジャイル及びCI/CDの導入**
  - ✓ **弊社開発アセットを利用**することにより、高品質で短期間での実現が可能





デジタル化に向けたクラウド活用セミナー

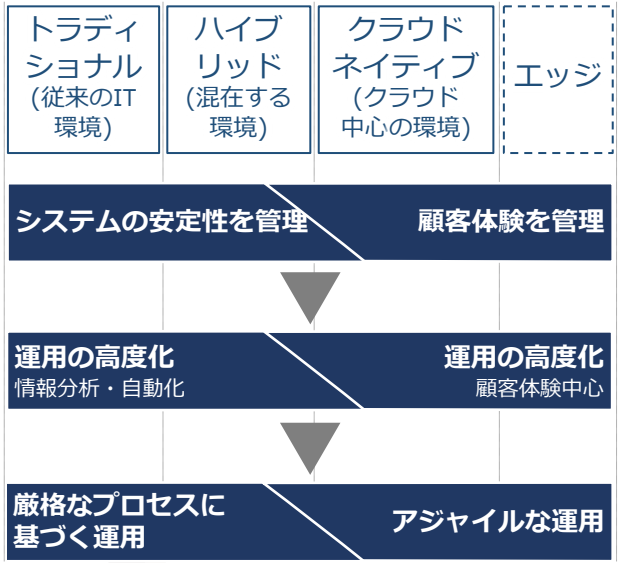
オープン・テクノロジー編
インフラ環境に左右されないシステムの実現

IBM ソリューションのご紹介

Make your deploy-anywhere future possible

オープン・テクノロジー化により求められる運用とは

レガシーとクラウドネイティブが共存するDX時代のシステム運用管理には、
従来の運用からクラウドを意識した運用への変革が求められています。



- ハイブリッド&マルチクラウドの世界では、クラウドベンダーや仮想化テクノロジーにより**必要なスキルセットが異なる**ため、従来型の運用だけでは対処できない
- トラディショナルなIT環境とクラウドネイティブ環境が混在する**クラスター間の可視性**がなく、一元管理することが必要
- 複雑なクラウドやオンプレミス環境でアプリの問題が発生した場合の**問題判別と解決のスピード**が遅い
- 60%以上のお客様は、複雑なマルチクラウド環境を管理し運用するためのツールやプロセスを持っていない

Source: IBM IBV Study – Assembling your cloud orchestra

クラウドネイティブ・ソフトウェア IBM Cloud Paks

IBM Cloud Paksは、エンタープライズにおけるユースケース別に製品化しています。
現在6製品を提供しており、今後も拡充を計画しています。

Cloud Pak for Applications	Cloud Pak for Data	Cloud Pak for Integration	Cloud Pak for Automation	Cloud Pak for Multicloud Management	Cloud Pak for Security
アプリケーションのビルド デプロイ 実行	データの収集 編成 解析	アプリケーション/データ/クラウドサービス/API の統合	ビジネス・プロセス 意思決定 コンテンツの変革	マルチクラウドの可視性 ガバナンス 自動化	セキュリティーのデータ、ツール、ワークフローを結びつける
コンテナ化されたIBMソフトウェア	コンテナ化されたIBMソフトウェア	コンテナ化されたIBMソフトウェア	コンテナ化されたIBMソフトウェア	コンテナ化されたIBMソフトウェア	コンテナ化されたIBMソフトウェア
コンテナプラットフォーム運用サービス Red Hat OpenShift	コンテナプラットフォーム運用サービス Red Hat OpenShift	コンテナプラットフォーム運用サービス Red Hat OpenShift	コンテナプラットフォーム運用サービス Red Hat OpenShift	コンテナプラットフォーム運用サービス Red Hat OpenShift	コンテナプラットフォーム運用サービス Red Hat OpenShift

IBM **Cloud** | AWS Azure GCP Other Cloud | Edge | Private | Systems

クラウドネイティブ・ソフトウェア IBM Cloud Paks

IBM Cloud Paksは、エンタープライズにおけるユースケース別に製品化しています。
現在6製品を提供しており、今後も拡充を計画しています。

Cloud Pak for Applications

アプリケーションの
ビルド
デプロイ
実行

コンテナ化された
IBMソフトウェア

コンテナ
プラットフォーム
運用サービス
Red Hat OpenShift

Cloud Pak for Data

データの
収集
編成
解析

コンテナ化された
IBMソフトウェア

コンテナ
プラットフォーム
運用サービス
Red Hat OpenShift

Cloud Pak for Integration

アプリケーション/
データ/
クラウドサービス/
API の統合

コンテナ化された
IBMソフトウェア

コンテナ
プラットフォーム
運用サービス
Red Hat OpenShift

Cloud Pak for Automation

ビジネス・プロセス
意思決定
コンテンツの変革

コンテナ化された
IBMソフトウェア

コンテナ
プラットフォーム
運用サービス
Red Hat OpenShift

Cloud Pak for Multicloud Management

マルチクラウドの
可視性
ガバナンス
自動化

コンテナ化された
IBMソフトウェア

コンテナ
プラットフォーム
運用サービス
Red Hat OpenShift

Cloud Pak for Security

セキュリティーの
データ、ツール、
ワークフローを
結びつける

コンテナ化された
IBMソフトウェア

コンテナ
プラットフォーム
運用サービス
Red Hat OpenShift

IBM **Cloud**　Other Cloud　Edge　Private　Systems

IBM Cloud Pak for Multicloud Management とは

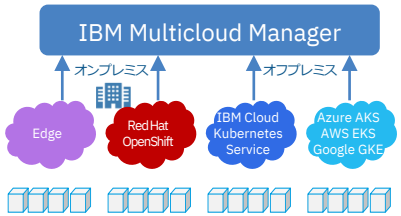
IBM Cloud Pak for Multicloud Management は、
複雑化するハイブリッド・マルチクラウド環境を統合管理し運用の自動化を推進します。

プロビジョニング （システム構築）　→　監視

クラスター間の可視性向上と必要なスキル・セットの統一

問題判別と解決スピード向上

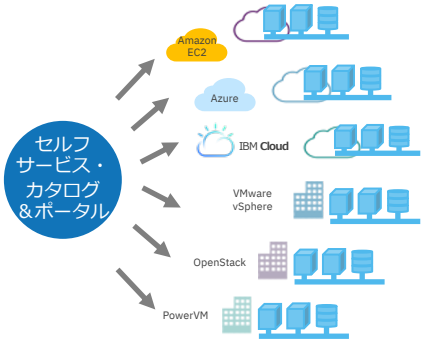
マルチクラスターを一元管理する**統合コンテナ・オーケストレーション**



IBM Multicloud Manager

VM&コンテナ基盤の**払出し/管理の自動化**

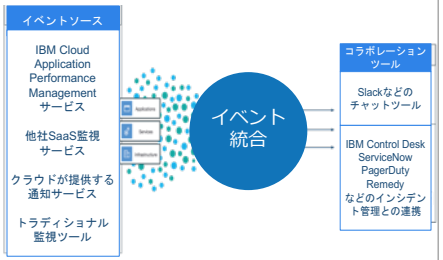
- マルチクラウドでIaaSレベルのデプロイ
- 事前定義されたデプロイ・パターンを提供



IBM Cloud Automation Manager
Ansible Automation / CloudForms

複数イベントソースからの**イベント統合**

- インシデントの優先順位づけと集約
- アラート通知の自動化
- ガイド通知・タスクの自動化



IBM Cloud
Event Management

SoR・SoE システムの**統合モニタリング**

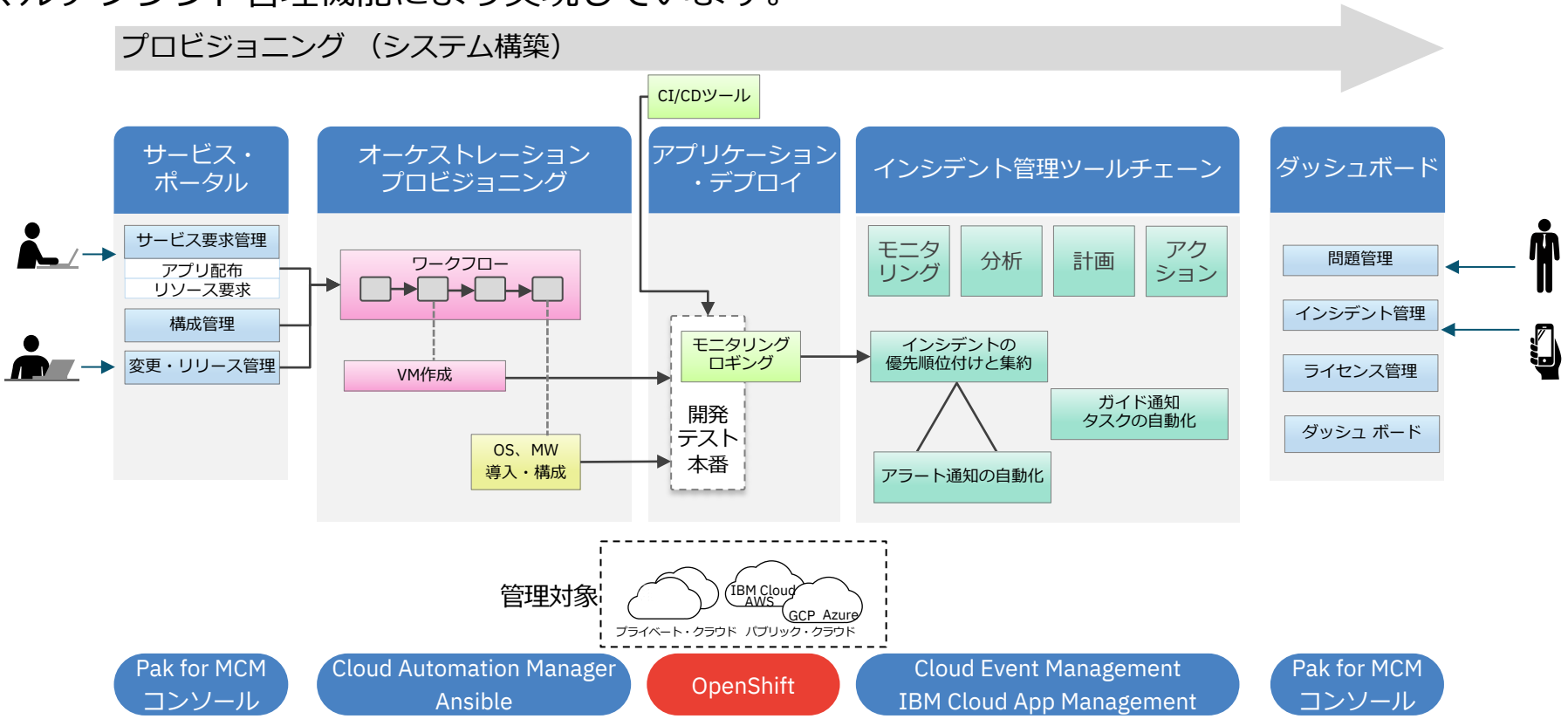
システム（オンプレミス）　システム（クラウド）

IBM Cloud
App Management

Red Hat OpenShift

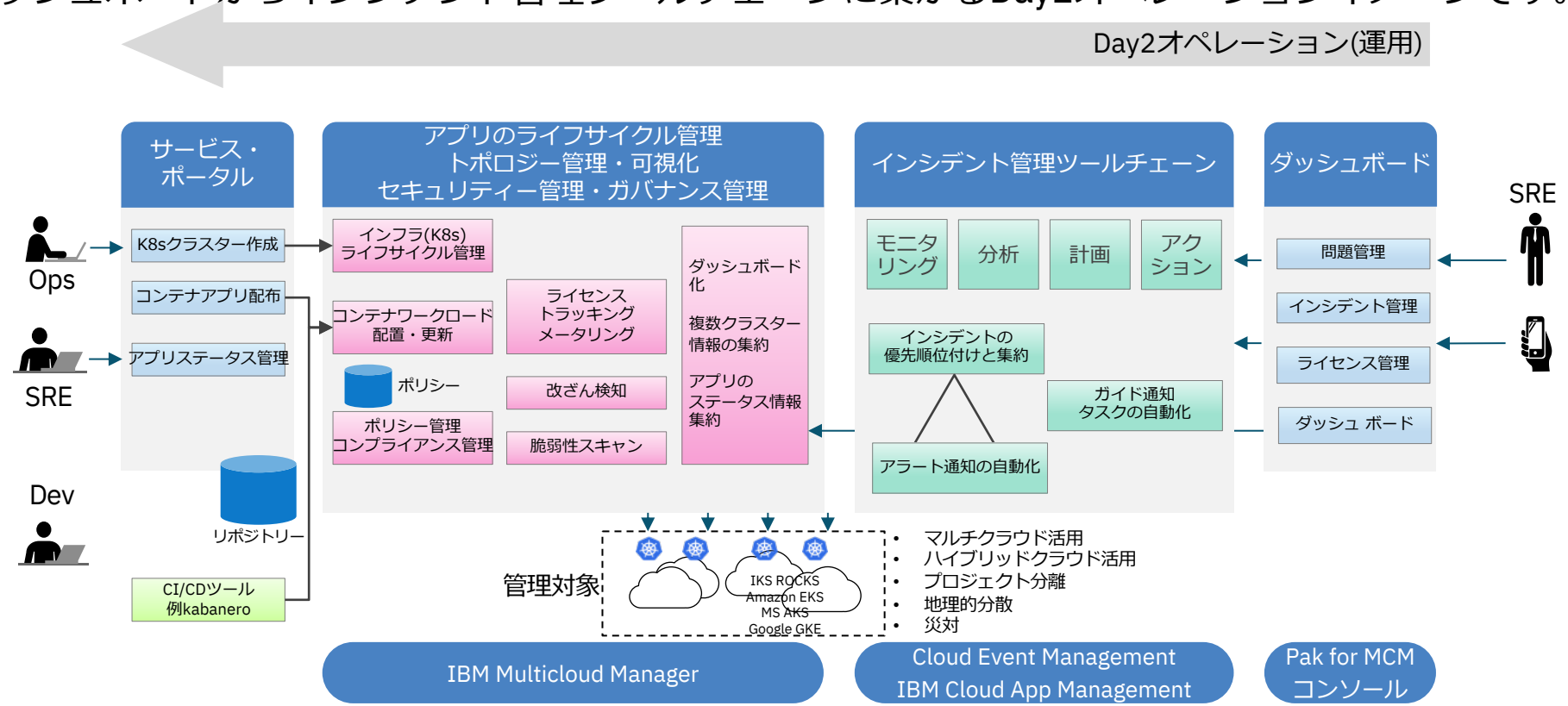
IBM Cloud Pak for Multicloud Management のユースケース(1)

ハイブリッド&マルチクラウド管理におけるプロビジョニング（システム構築）イメージです。
マルチクラウド管理機能により実現しています。

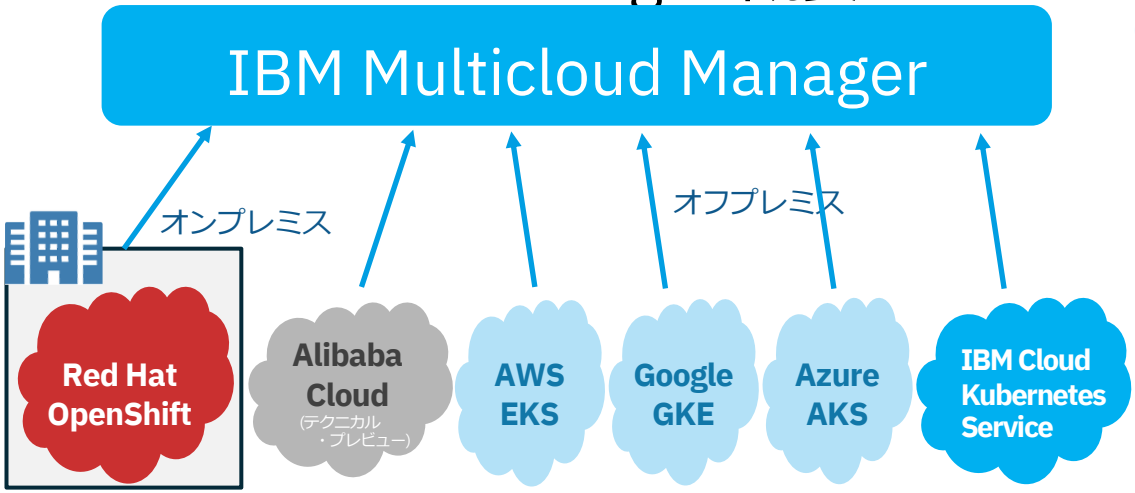


IBM Cloud Pak for Multicloud Management のユースケース(2)

ハイブリッド＆マルチクラウド管理におけるKubernetesクラスターの一元管理の例です。
ダッシュボードからインシデント管理ツールチェーンに繋がるDay2オペレーションイメージです。



IBM Multicloud Manager 概要

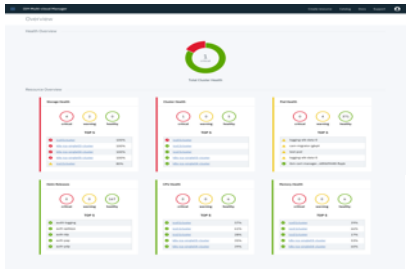


1 マルチクラスタの運用と可視性
- K8sクラスタの払出し(Cluster API)、インポート
- 複数クラスターの健全性の可視化
- SRE向け統合サーチ機能

2 アプリケーション・セントリック管理
- 複数クラスターへのアプリケーション配布
- 複数クラスタ間の動的移動
- 複数クラスタ間のIstioサービス・ディスカバリ

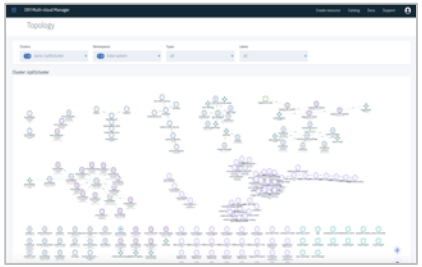
3 ガバナンス&リスク管理
- ポリシーベースのセキュリティ強化
- 脆弱性アドバイザー、変更アドバイザー++

ダッシュボード



マルチクラスタ管理

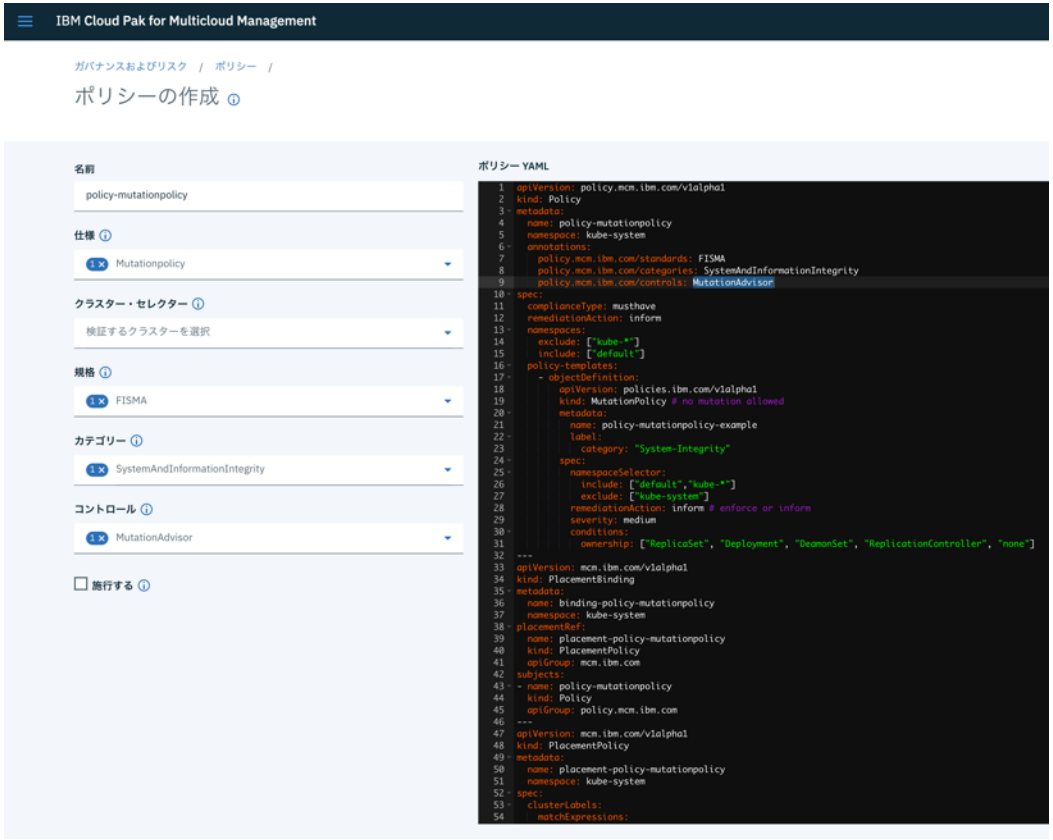
全体俯瞰



リモート・クラスタへの
アプリケーション・デプロイ

ガバナンス&リスク管理

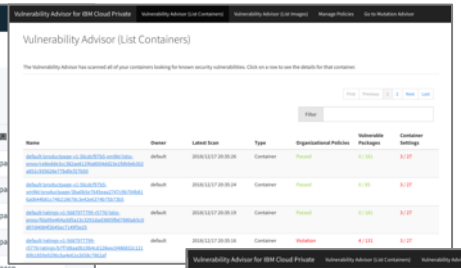
■あるべき定義との差異をポリシー・ベースで確認

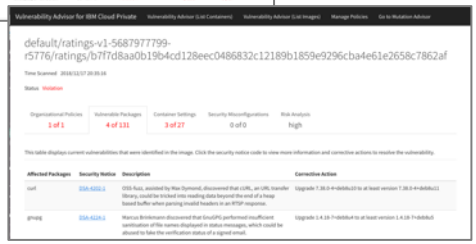


- ■ 構成ポリシー・コントローラー
  - – 構成情報のリストを取得し条件と一致するか確認
- ■ IAMポリシー・コントローラー
  - – RBACとロール・バインディングのポリシーをチェック
- ■ 監査ロギング・ポリシー・コントローラー
  - – 主要なコア・サービスの監査機能の稼働を確認
- ■ 証明書ポリシー・コントローラー
  - – ポリシーに準拠しない証明書を検知
- ■ Secret暗号化ポリシー・コントローラー
  - – ポリシーに準拠しないSecretの暗号化を検知
- ■ CIS・ポリシーコントローラー
  - – Center for Internet Security (CIS) のK8s基準への対応をチェック
- ■ 脆弱性ポリシー・コントローラー
  - – 脆弱性が検知されたPodを通知
- ■ ミューテーション・ポリシー・コントローラー
  - – 変更があったPodを通知

ガバナンス&リスク管理：脆弱性アドバイザー

■ イメージ・レポジトリとの統合

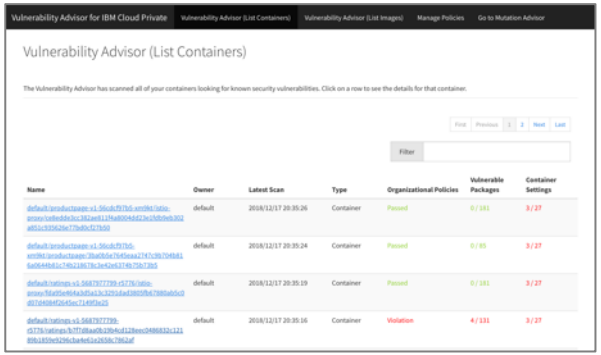




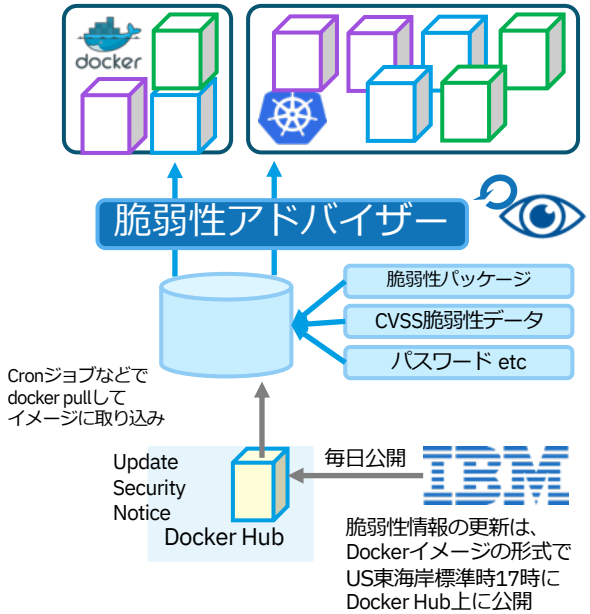
■ スキャン項目

・既知の脆弱性があるパッケージのスキャン
・ITCS104をベースに構成された27のデフォルト・ルール
・rootkitが導入されているかのスキャン

■ 稼働インスタンスのスキャン



イメージ・レポジトリ　コンテナー・インスタンス



ガバナンス&リスク管理：Mutationアドバイザー

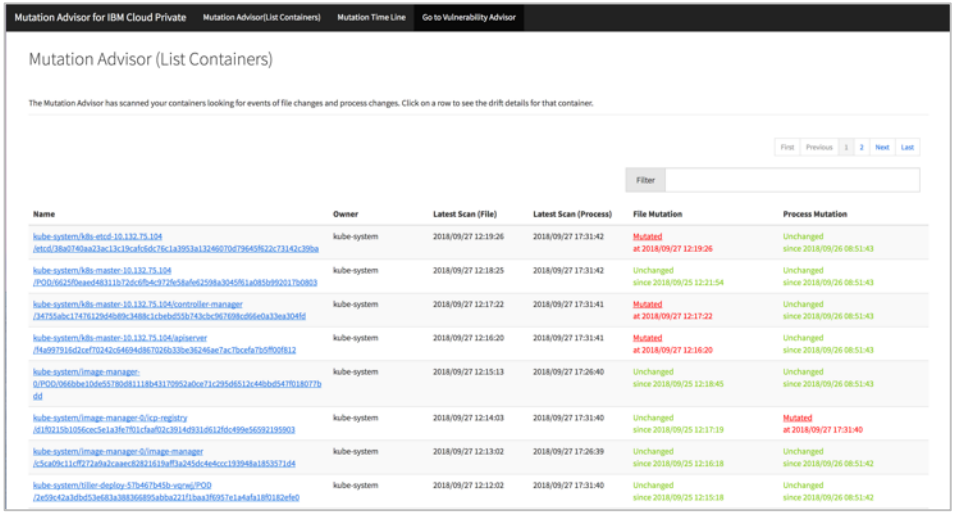
■ 変更検知機能

–各種業界IT標準では変更検知機能が要求されている

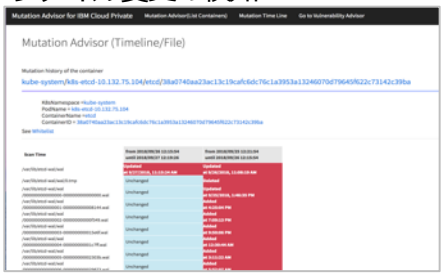
例：Payment Card Industry Data Security Standard (PCI/DSS)

✓ クリティカルなシステム・ファイル、構成ファイル、コンテンツ・ファイルへの許可を得ない変更は、最低限週次で担当者に通知されなければならない

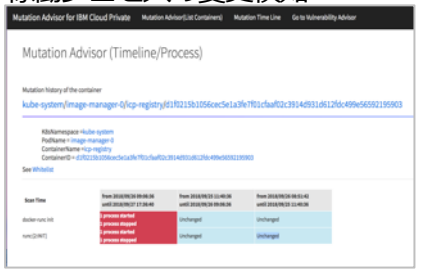
Mutation Advisor



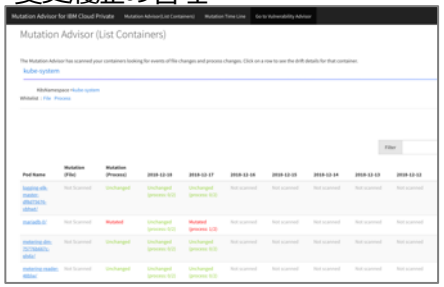
ファイル変更の検知



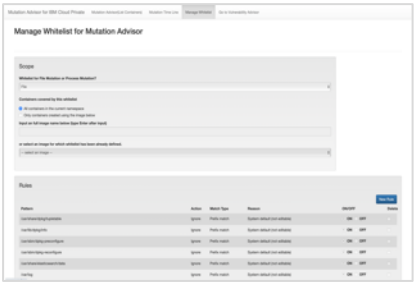
稼働プロセスの変更検知



変更履歴の管理



ホワイトリストの設定



IBM Cloud Automation Manager 概要

■ 共通のテクノロジーで、マルチクラウドを管理
- –特定クラウドに依存したくない
- –あらゆるクラウドの操作を覚えることはできない

Terraform

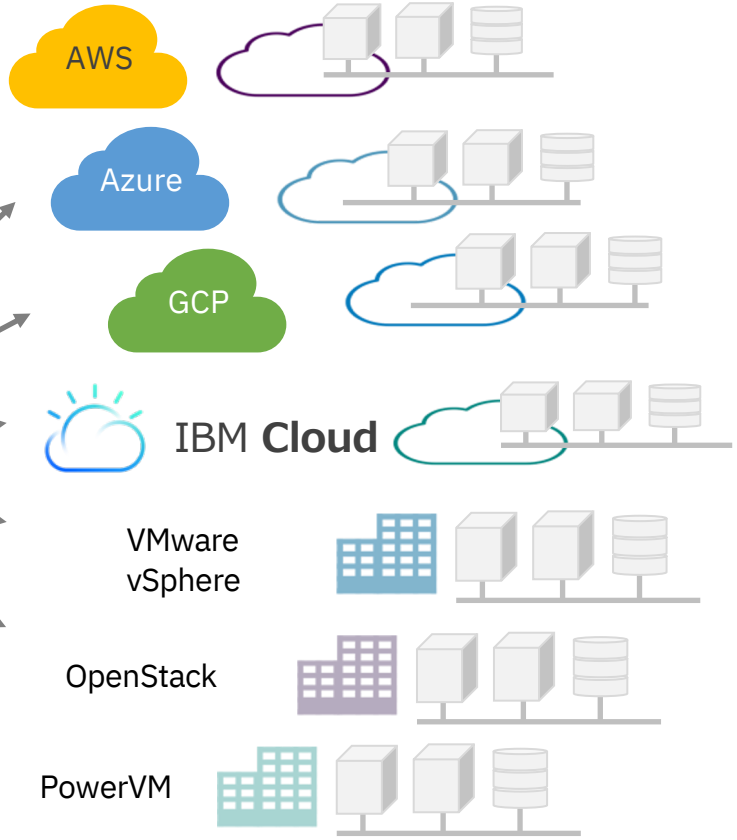
仮想化環境や多様なクラウド環境の
構成を自動化する
オープンソース・テクノロジー
https://www.terraform.io/

あるべき姿をコードとして記述

```
# Specify the provider and access details
provider "aws" {
  access_key = "${var.aws_access_key}"
  secret_key = "${var.aws_secret_key}"
  region = "${var.aws_region}"
}

# Our default security group to access
# the instances over SSH and HTTP
resource "aws_security_group" "default" {
  name = "terraform_example"
  description = "Used in the terraform"

  # SSH access from anywhere
  ingress {
    from_port = 22
    to_port = 22
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
```



Infrastructure as Code

基盤環境の構築も
アプリケーション・コードのように管理・実行

基盤自動化ソリューションの位置付け

Terraform Automation

■ 「宣言的」自動化

- クラウド環境・仮想化環境(Cloud Provider)のリソース払出し
- 「あるべき状態」を 宣言、Terraform が自動的に あるべき状態に構成
- 環境が逸脱した状態である場合には、「あるべき状態」にするため なにを変更するか 示した(terraform plan)上で、実行することが可能
  - このため実行した環境の「ステート情報」を持っている

■ オーケストレーション・ツール

- 複数構成ファイル（tfファイル）に定義されたリソースの依存関係を自動的に判別し、処理の実行順序に反映。依存関係の明示的指定も可能
- 複数リソースの「あるべき姿」をそれぞれのファイルに記載しておくことで、統合して実行してくれる
- OS払出し以降は構成管理ツール(Cloud Provisioner)と連携

クラウド・リソース のプロビジョニング

Ansible Automation

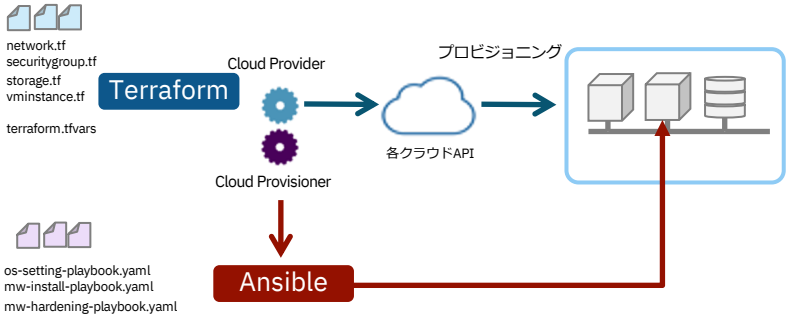
■ 「宣言的」自動化 +「手続き型」自動化

- エージェントレスで 構成自動化が可能となることが大きな魅力
  - SSHがターゲットに構成されていればよく、既存環境の自動化が可能
- あるべき姿を宣言する「宣言的自動化」、できるだけこちらで実装。
- 宣言的な方法で 記載することが難しい、泥臭い処理や独自の運用処理も「手続き型」で自動化することが可能
  - ただし手続き型で記載すると「べき等性」を実現するためには、処理が複雑化する

■ 構成管理ツール・Day2運用の自動化ツール

- OS設定やハードニング、ミドルウェアの構成・設定
- 意図した通りに構成されているかも Ansible で確認
- 繰返し実施するDay2運用の処理、繰返し実施する障害対応の自動化

OS・ミドルウェアの導入構成、Day2運用



CloudForms

■ クラウド環境・仮想化環境の 構成情報 自動集約管理

- 払い出された環境情報を自動的にスキャニングして マシン情報、基盤情報を収集
- ダッシュボードを利用して全体を把握、事前定義された様々な観点でのレポート生成
- 環境のライフサイクル管理(利用期限設定)、ポリシーベースでの環境コントロールとアラート通知

マルチクラウド・ハイブリッドクラウド全体の可視化、集約管理

IBM Cloud Event Management 概要

■オンプレ&クラウド環境のインシデント ライフサイクルを一元管理
■イベント/インシデントの相関処理、優先順位付け、解決を統合管理
■担当者への通知/インシデント管理ツール連携/運用対応手順を自動化

アプリケーション、パフォーマンス、オペレーション・イベントの収集

優先順位付け、集約、インシデント情報、オペレーション・ガイド表示の自動処理

チーム間の情報交換を効率的に実施

IBM Cloud App Management 概要

ハイブリッド・マルチクラウド環境のアプリケーションを統合管理

- ハイブリッド・マルチクラウド環境 Application Performance Management ソリューション
  - クラウド・ネイティブ・アプリ：
    Kubernetes環境および信頼性への影響の把握
  - レガシー・アプリ：
    幅広いインフラとアプリをサポート
  - ブラックボックス・モニタリングにも対応
- アプリ管理を容易にするサービスレベル・モニタリング
  - ゴールデン・シグナルによるモニタリング
  - 1ホップのサービストポロジーを提供
  - 専門知識への依存度を軽減
- ダイナミックなクラウド・ワークロードをしっかり管理
  - インフラへの変更とインシデントの相関を関係を表示
  - イベント管理とランブックオートメーションと連携し運用をスケーラブルに

まとめ

- 「インフラ環境に左右されないシステム」を実現するためには、コンテナ / マイクロサービス / CI CD / DevSecOps などのオープン・テクノロジーの活用が重要です。
- Kubernetes をエンタープライズ向けに提供する OpenShift がコンテナ基盤運用に伴う複雑性を簡易化します。
- IBMは構想策定～開発～運用/保守の全てのフェーズにおいてオープン・テクノロジーを活用したお客様のデジタル変革をご支援します。
- Cloud Pak for Multicloud Management はアプリケーションがオンプレミス、クラウド、エッジのどこにあろうと、一貫性のあるイベント管理・アプリケーション管理・インフラストラクチャー管理・マルチクラスター管理を提供し、「インフラ環境に左右されないシステム」の運用を実現します。

ワークショップ、セッション、および資料は、IBMまたはセッション発表者によって準備され、それぞれ独自の見解を反映したものです。それらは情報提供の目的のみで提供されており、いかなる参加者に対しても法律的またはその他の指導や助言を意図したものではなく、またそのような結果を生むものでもありません。本講演資料に含まれている情報については、完全性と正確性を期するよう努力しましたが、「現状のまま」提供され、明示または暗示にかかわらずいかなる保証も伴わないものとします。本講演資料またはその他の資料の使用によって、あるいはその他の関連によって、いかなる損害が生じた場合も、IBMは責任を負わないものとします。 本講演資料に含まれている内容は、IBMまたはそのサプライヤーやライセンス交付者からいかなる保証または表明を引きだすことを意図したものでも、IBMソフトウェアの使用を規定する適用ライセンス契約の条項を変更することを意図したものでもなく、またそのような結果を生むものでもありません。

本講演資料でIBM製品、プログラム、またはサービスに言及していても、IBMが営業活動を行っているすべての国でそれらが使用可能であることを暗示するものではありません。本講演資料で言及している製品リリース日付や製品機能は、市場機会またはその他の要因に基づいてIBM独自の決定権をもっていつでも変更できるものとし、いかなる方法においても将来の製品または機能が使用可能になると確約することを意図したものではありません。本講演資料に含まれている内容は、参加者が開始する活動によって特定の販売、売上高の向上、またはその他の結果が生じると述べる、または暗示することを意図したものでも、またそのような結果を生むものでもありません。 パフォーマンスは、管理された環境において標準的なIBMベンチマークを使用した測定と予測に基づいています。ユーザーが経験する実際のスループットやパフォーマンスは、ユーザーのジョブ・ストリームにおけるマルチプログラミングの量、入出力構成、ストレージ構成、および処理されるワークロードなどの考慮事項を含む、数多くの要因に応じて変化します。したがって、個々のユーザーがここで述べられているものと同様の結果を得られると確約するものではありません。

記述されているすべてのお客様事例は、それらのお客様がどのようにIBM製品を使用したか、またそれらのお客様が達成した結果の実例として示されたものです。実際の環境コストおよびパフォーマンス特性は、お客様ごとに異なる場合があります。

IBM、IBM ロゴ、ibm.com、IBM Cloud Pak は、 世界の多くの国で登録されたInternational Business Machines Corporationの商標です。他の製品名およびサービス名等は、それぞれIBMまたは各社の商標である場合があります。現時点での IBM の商標リストについては、www.ibm.com/legal/copytrade.shtml をご覧ください。

Red Hat®, JBoss®, OpenShift®, Ansible®, CloudForms® are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

Linuxは、Linus Torvaldsの米国およびその他の国における登録商標です。

VMware およびVMware vSphereは、VMware, Inc.またはその子会社の米国およびその他の地域における登録商標または商標です。