NTT DATA

AWS上のシステムはこう作る！

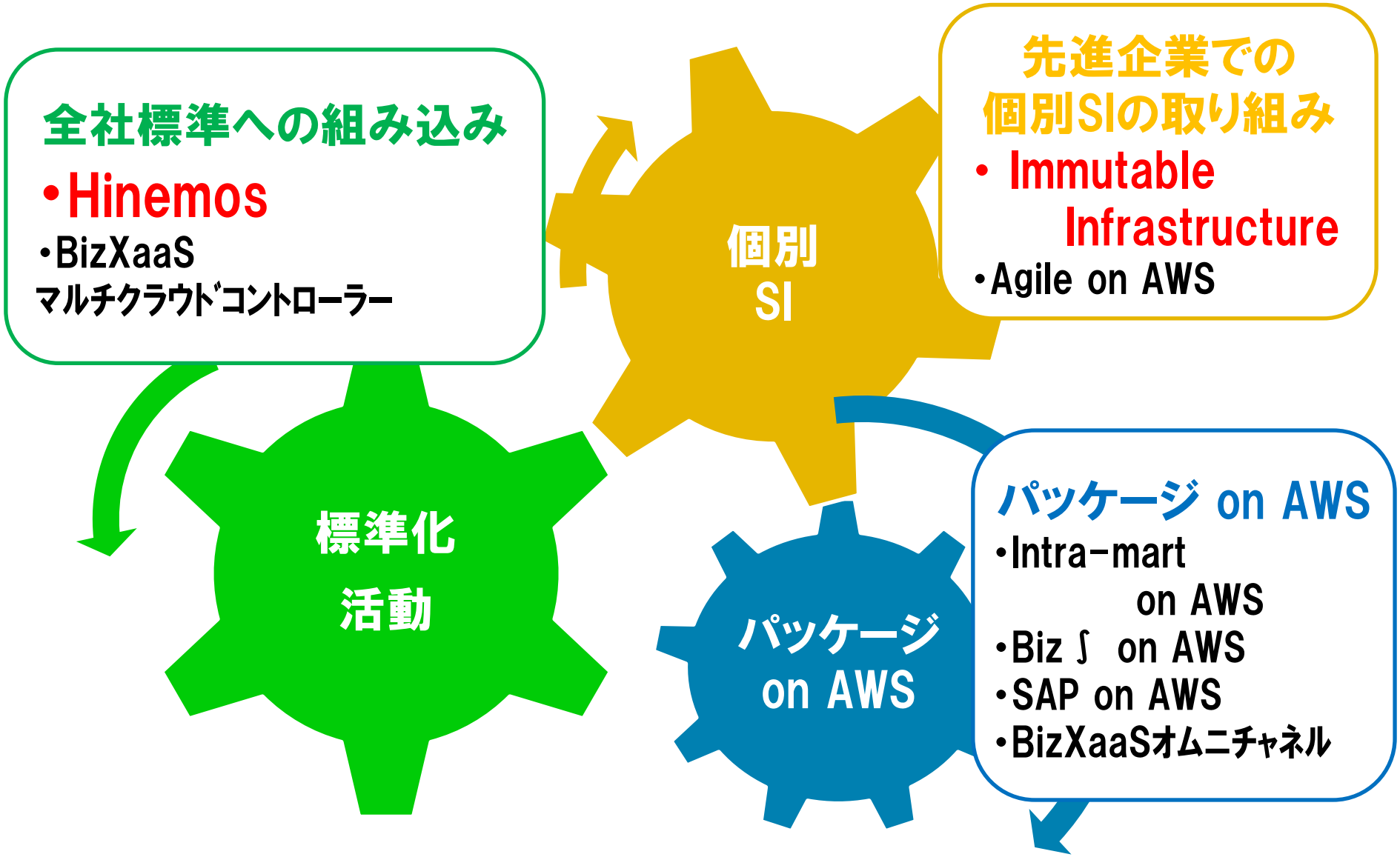
#1 InfrastructureAsCode／ ImmutableInfrastructure を実践した基盤構築自動化

#2 Hinemosで実現するクラウド運用自動化

株式会社NTTデータ

NTTデータのAWSのとりくみ

NTT DATA



NTT DATA

AWS上のシステムはこう作る！

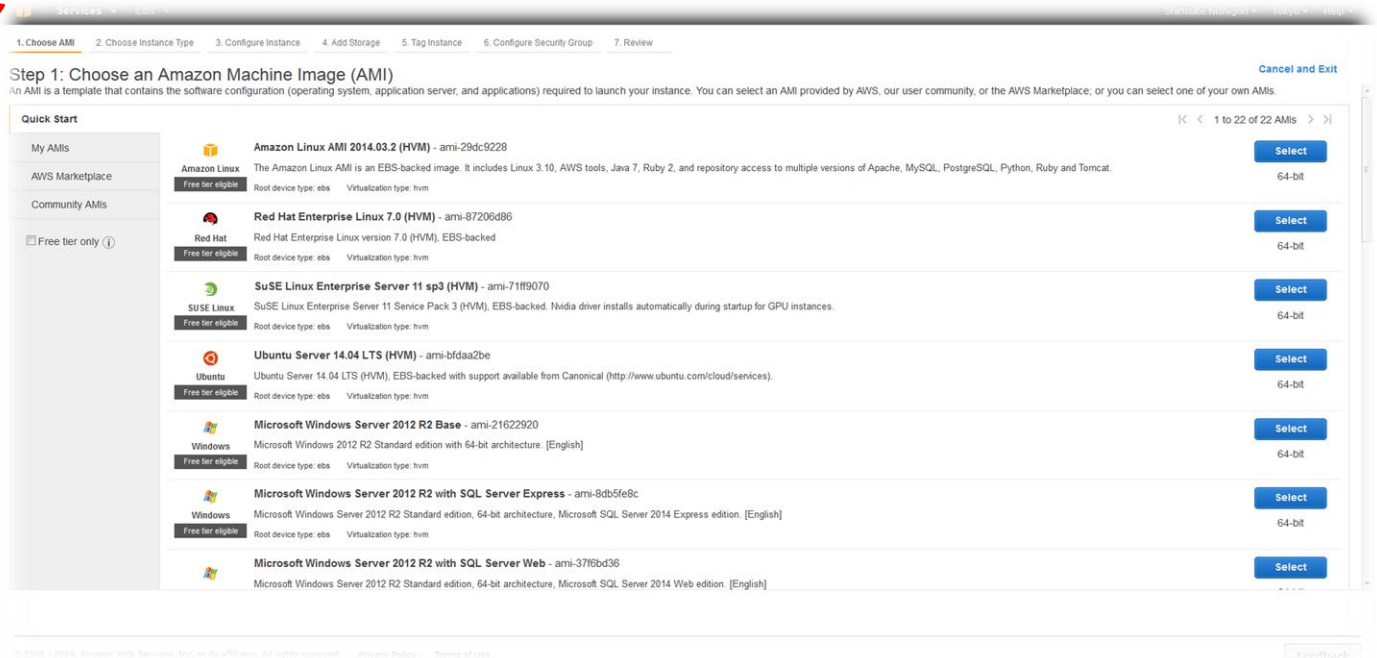
#1 InfrastructureAsCode／ImmutableInfrastructure を実践した基盤構築自動化

株式会社NTTデータ
第三法人事業本部　交通・流通事業部
課長　錦織 真介

It's Cool and Great

1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Tag Instance
6. Configure Security Group
7. Review

7 step magic



AmazonWebService Management Console

The problem of Enterprise Quality

サーバをSpeedyに調達できても・・・・

特殊なミドルウェア導入構築

綿密なテストが必要！

NTT DATA

Our Concept

NTTdata Concept - Detail

従来の基盤開発　必要期間

数ヶ月 → 1ヶ月 → 2ヶ月

機器調達

OS設定、テスト

ミドルウェア設定、テスト

← 数分 →　　← 数分 →

amazon web services

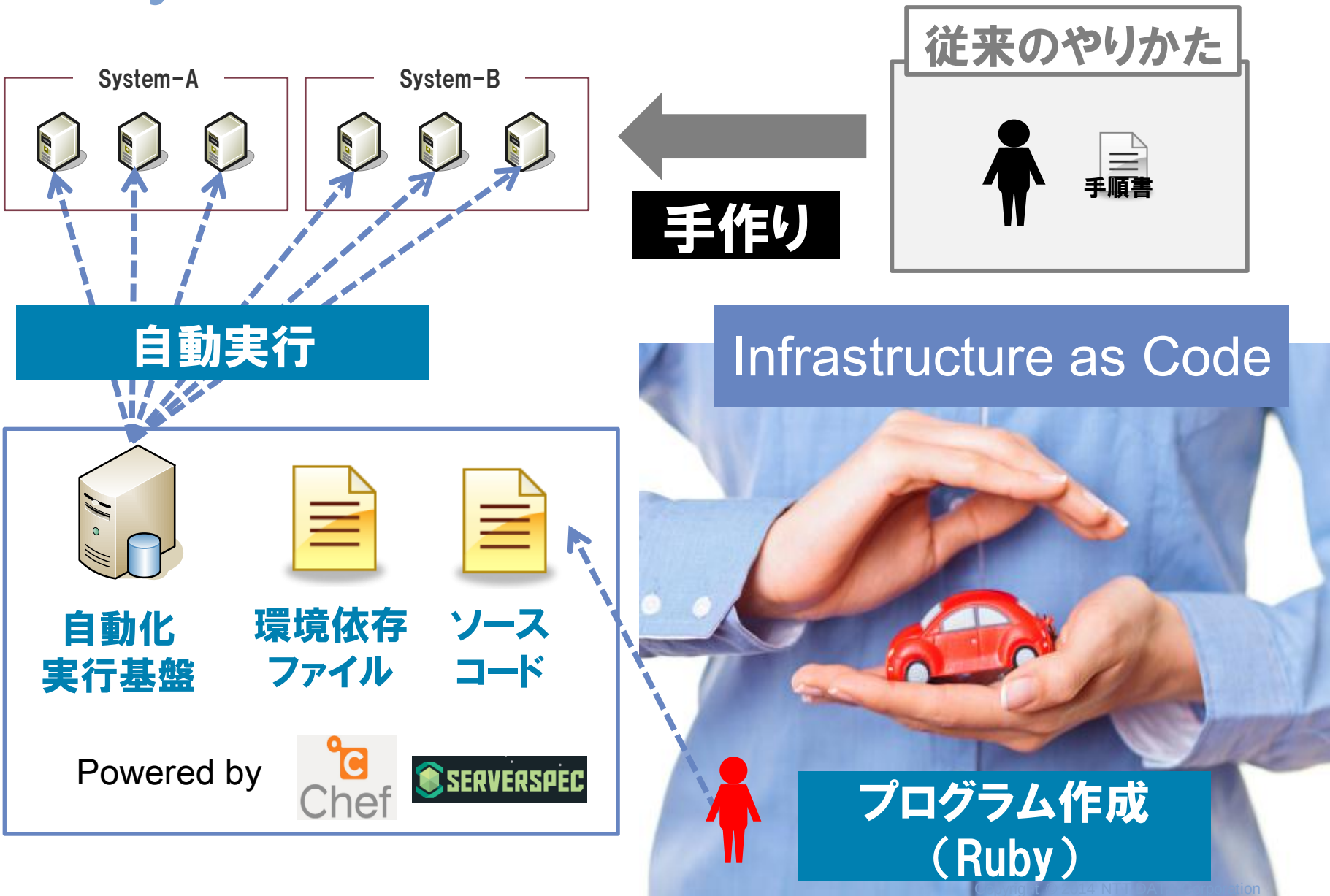
Infrastructure as Code

Chef

SERVERSPEC

NTT DATA

KeyWord : Infrastructure as Code



NTT DATA

Know-how From our experience

本気で“基盤自動化”するために・・・

Point 1
日本文化への適応

Point 2
基盤自動化チームワーク

Point 3
基盤自動化の応用
Immutabel Infrastructure

NTT DATA

Know-how From our experience

本気で“基盤自動化”するために・・・

Point 1
日本文化への適応

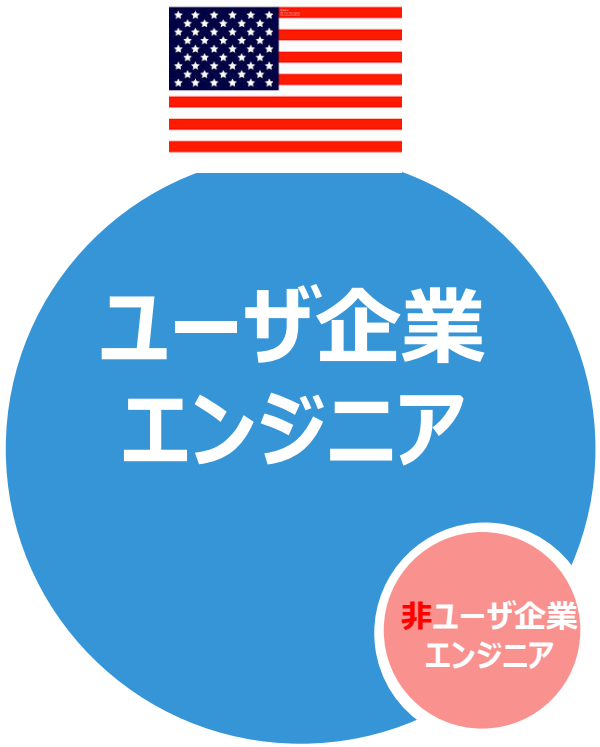
Point 2
基盤自動化チームワーク

Point 3
基盤自動化の応用
Immutabel Infrastructure

NTT DATA

Point 1: Infrastructure as Code on Japanese Culture

基盤自動化を日本でうまく使うには…



日本では、**他社とのコレボレーションを重視した**
エンジニアリングを必要としている

NTT DATA

Point 1: Infrastructure as Code on Japanese Culture

他社とのコレボレーション重視の極み

「日本式運用」
≒ より厳密な管理

- ✓ サービス判定会議
- ✓ 変更管理

http://www.flickr.com/photos/elisamoro/7976145498/

NTT DATA

Point 1: Infrastructure as Code on Japanese Culture

具体的な発生ケース

- ✓ 自動化コードを作成して、サーバを構築しました
- ✓ サービス開始後、不幸なことに基盤バグが発生
- ✓ 緊急でサーバ設定変更が必要に・・・

あなたならどうしますか？

Point 1: Infrastructure as Code on Japanese Culture

答え。

1. 実機(サーバ)を直接なおす
2. 自動化コードを修正し、再実行
3. **故障処理表番号ごとに差分の自動化コードを作成し、実機に適用**

2で対応したいところ。
ただ、変更管理の確実性を
考えると、『3が現実的』

Point 1: Infrastructure as Code on Japanese Culture

当社事例　1

運用に耐える

Chef －＞

NTT DATA

Point 1: Infrastructure as Code on Japanese Culture

Dry-runを活用し、高品質作業

$ knife solo cook root@10.72.39.XX -W

```
Running Chef on 10.72.39.XX...
（略）
Recipe: tomcat::tomcat_tomcat
  * template[etc-default-tomcat] action create
    - Would update content in file /etc/default/tomcat from 90887c to 1edf5d
      --- /etc/default/tomcat 2014-06-26 15:57:01.581131866 +0900
      +++ /tmp/chef-rendered-template20140701-25009-gxmpeu    2014-07-01 17:01:47.577728819 +0900
      @@ -1,7 +1,7 @@
       JAVA_HOME="/opt/java"
       JAVA_OPTS="-Dtomcatprocess -Xloggc:/var/log/tomcat/gc.dat "
      -CATALINA_OPTS="（略）-Xms1024m -XX:NewSize=384m -XX:MaxNewSize=384m -XX:SurvivorRatio=8（略） "
      +CATALINA_OPTS="（略） -Xms1024m -XX:NewSize=512m -XX:MaxNewSize=512m -XX:SurvivorRatio=8（略） "
       LANG=ja_JP.UTF-8
       LC_ALL=ja_JP.UTF-8

Recipe: tomcat::tomcat_tomcat-restart
  * service[tomcat] action restart
    - Would restart service service[tomcat]

Chef Client finished, 2 resources would have been updated
```

Diff結果が表示され値が望んだものかどうか事前に確認できるので、作業失敗を回避！！

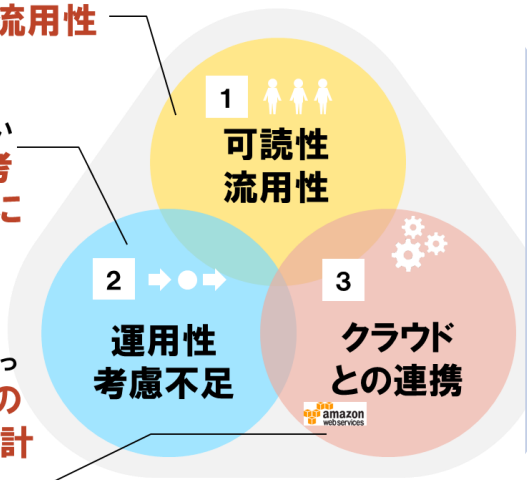
Point 1: Infrastructure as Code on Japanese Culture

当社事例　2

1. 実機(サーバ)を直接なおす
2. 自動化コードを修正し、再実行
3. 故障処理表番号ごとに自動化コードを作成し、実

他人が作ったコードに手をいれる不安は拭いきれないが・・・

- ✓ 複数人で好き勝手にコードをかきはじめると**可読性がおち、流用性が悪くなる**
- ✓ 手作業でサーバ構築をしていた人間は、**冪等性の考慮が足りず、運用にたえられない**
  Ex）pkg install
- ✓ 各種クラウドやツールが持っている**自動化機能の連携が難しく、余計なコードが増える**

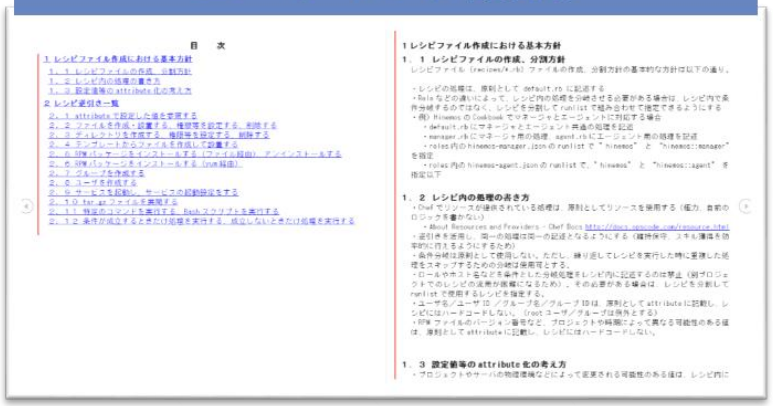


自動化コード・仕組みのルール、ノウハウが重要

Point 1: Infrastructure as Code on Japanese Culture

当社で使われている各種ルール

コーディング規約



目　次

1 レシピファイル作成における基本方針
1．1 レシピファイルの作成、分割方針
1．2 レシピ内の処理の書き方
1．3 設定値等のattribute化の考え方
2 レシピ逆引き一覧
2．1 attributeで設定した値を参照する
2．2 ファイルを作成・設置する、権限等を設定する、利用する
2．3 ディレクトリを作成する、権限等を設定する、削除する
2．4 テンプレートからファイルを作成して設置する
2．5 RPMパッケージをインストールする（ファイル指定）、アンインストールする
2．6 RPMパッケージをインストールする（yum経由）
2．7 グループを作成する
2．8 ユーザを作成する
2．9 サービスを起動し、サービスの起動設定をする
2．10 tar.gzファイルを展開する
2．11 特定のコマンドを実行する（bashスクリプトを実行する
2．12 条件が成立するときだけ処理を実行する、成立しないときだけ処理を実行する

1 レシピファイル作成における基本方針

1．1 レシピファイルの作成、分割方針

レシピファイル（recipes/*.rb）ファイルの作成、分割方針の基本的な方針は以下の通り。

・レシピの処理は、原則として default.rb に記述する
・Role などの違いによって、レシピ内の処理を分岐させる必要がある場合は、レシピ内で条件分岐するのではなく、レシピを分割して runlist で組み合わせて指定できるようにする
・例）Hinemos の Cookbook でマネージャとエージェントに対応する場合
・default.rb にマネージャとエージェント共通の処理を記述
・manager.rb にマネージャ用の処理、agent.rb にエージェント用の処理を記述
・roles 内の hinemos-manager.json の runlist で、"hinemos" と "hinemos::manager" を指定
・roles 内の hinemos-agent.json の runlist で、"hinemos" と "hinemos::agent" を指定以下

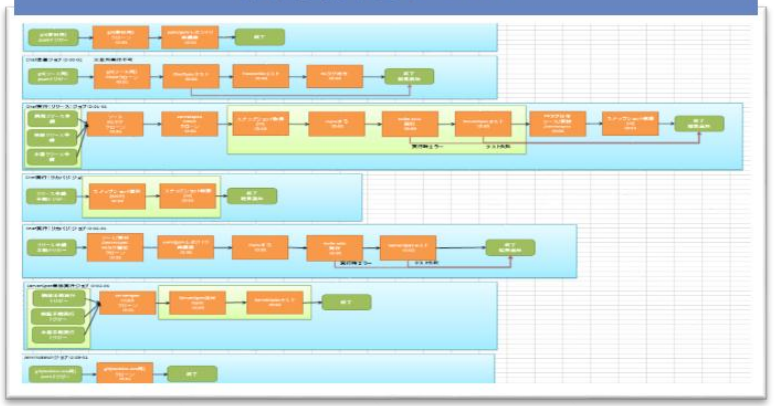
1．2 レシピ内の処理の書き方

・Chef やリソースが提供されている処理は、原則としてリソースを使用する（極力、自前のロジックを書かない）
・About Resources and Providers - Chef Docs http://docs.opscode.com/resource.html
・記引きを実現し、同一の処理は同一の記述となるようにする（保守性向上、スキル獲得を効率的に行えるようにするため）
・条件分岐は原則として使用しない。ただし、繰り返してレシピを実行した時に重複した処理をスキップするための分岐は使用可とする。
・ホストロールやホスト名などを条件とした分岐処理をレシピ内に記述するのは禁止（例プロジェクトでのレシピの変更が困難になるため）。その必要がある場合は、レシピを分割して runlist で使用するレシピを指定する。
・ユーザ名／ユーザID／グループ名／グループIDは、原則としてattributeに記載し、レシピにはハードコードしない。（root ユーザ／グループは例外とする）
・RPM ファイルのバージョン番号など、プロジェクトや時期によって異なる可能性のある値は、原則としてattributeに記載し、レシピにはハードコードしない。

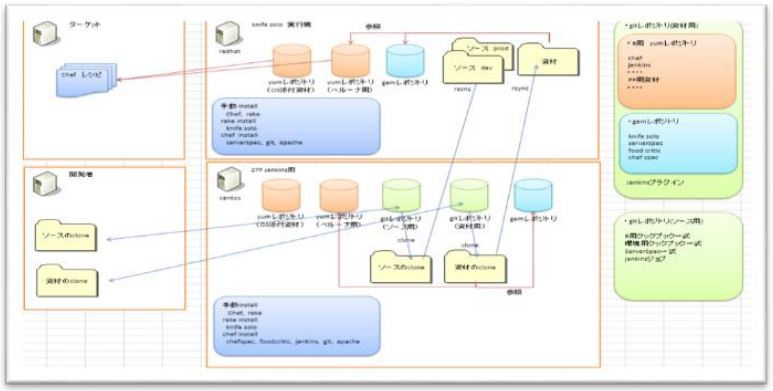
1．3 設定値等のattribute化の考え方

・プロジェクトやサーバの物理環境などによって変更される可能性のある値は、レシピ内に

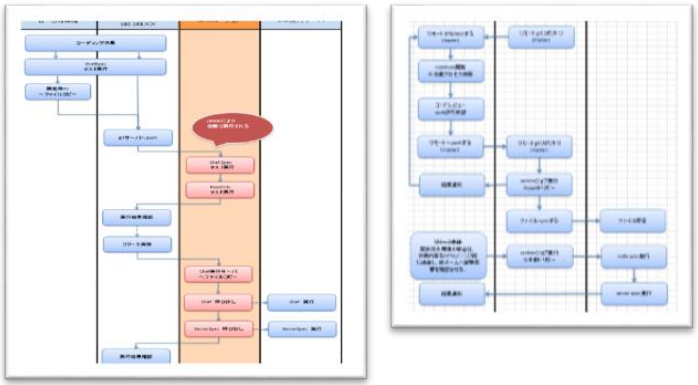
自動化ジョブ



レポジトリ配置



自動化利用フロー



NTT DATA

Know-how From our experience

本気で“基盤自動化”するために・・・

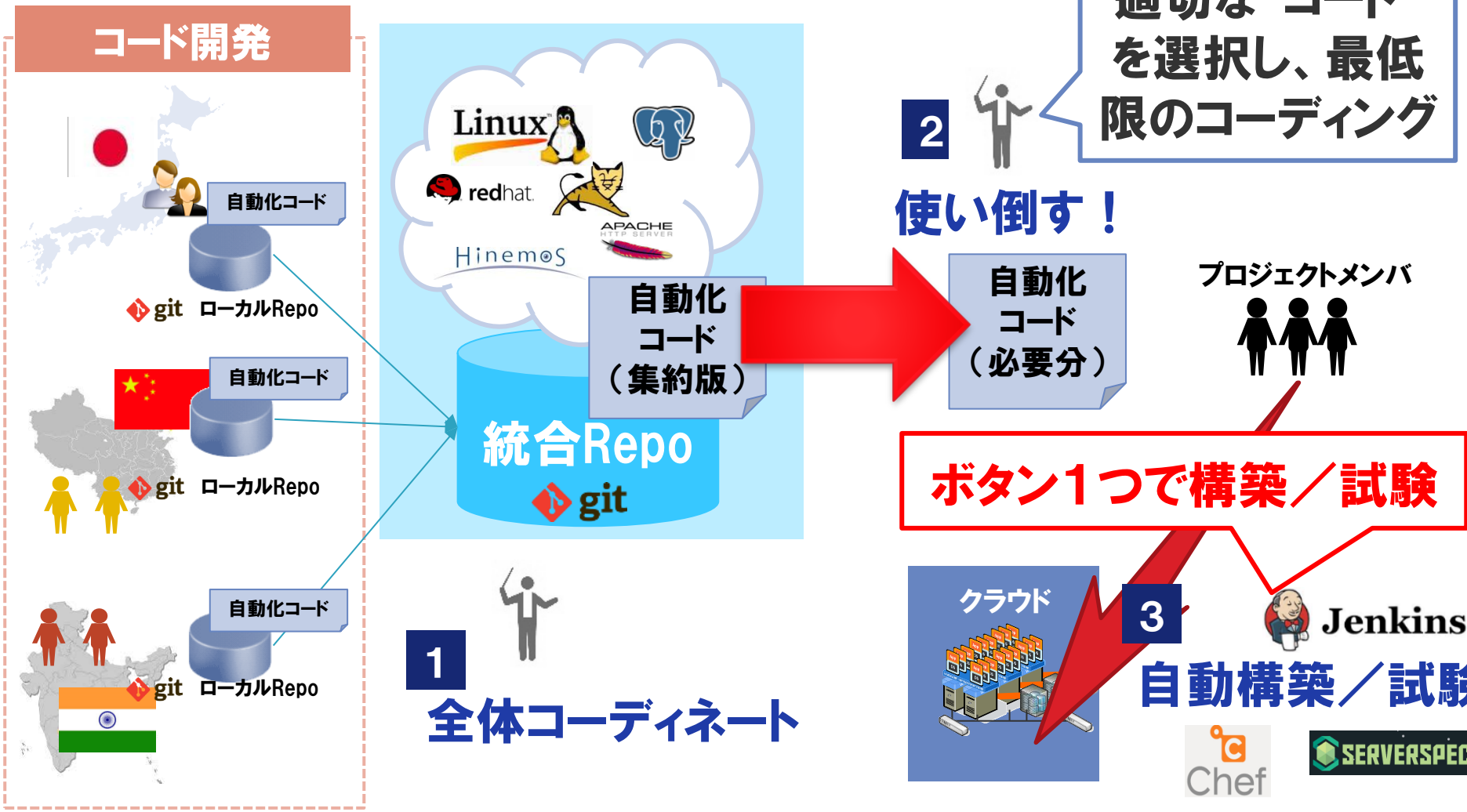
Point 1
日本文化への適応

Point 2
基盤自動化チームワーク

Point 3
基盤自動化の応用
Immutabel Infrastructure

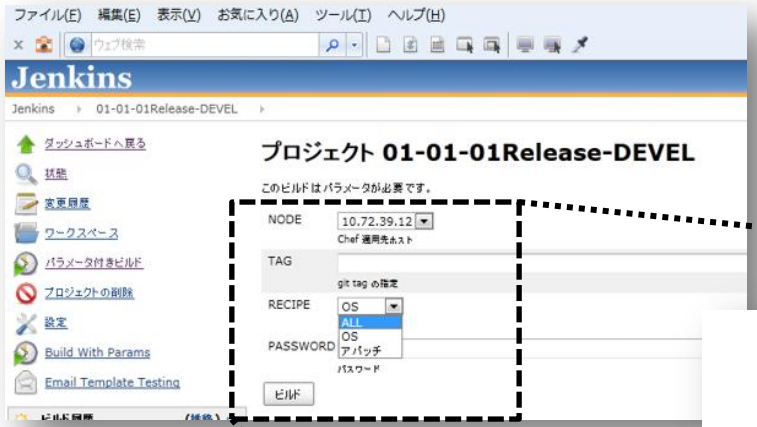
Point 2: We need TeamWork

SIerの力を生かし、海外でコーディング



Point 2: We need TeamWork

自動構築／自動試験　画面イメージ



Chef実行画面

NTT DATA

Point 2: We need TeamWork

自動構築／自動試験　フロー

起動トリガ1
コード
GIT-PUSH

起動トリガ2
画面ボタン押下

1 コードチェック（正当性確認）

2 リリースコード管理

3 Chefコード実行（サーバ自動構築）

4 SERVERSPEC実行（サーバ自動試験）

完了メール発出

画面ワンクリック、または自動コードCheckINで自動構築／試験開始

コードにミスがないか、チェック
ChefSPEC／FoodCritic

不足の事態が発生しても自動リカバリができるための処理を実行 git

Chef

SERVERSPEC

サーバへ自動構築試験コードを実行

NTT DATA

Know-how From our experience

本気で“基盤自動化”するために・・・

Point 1
日本文化への適応

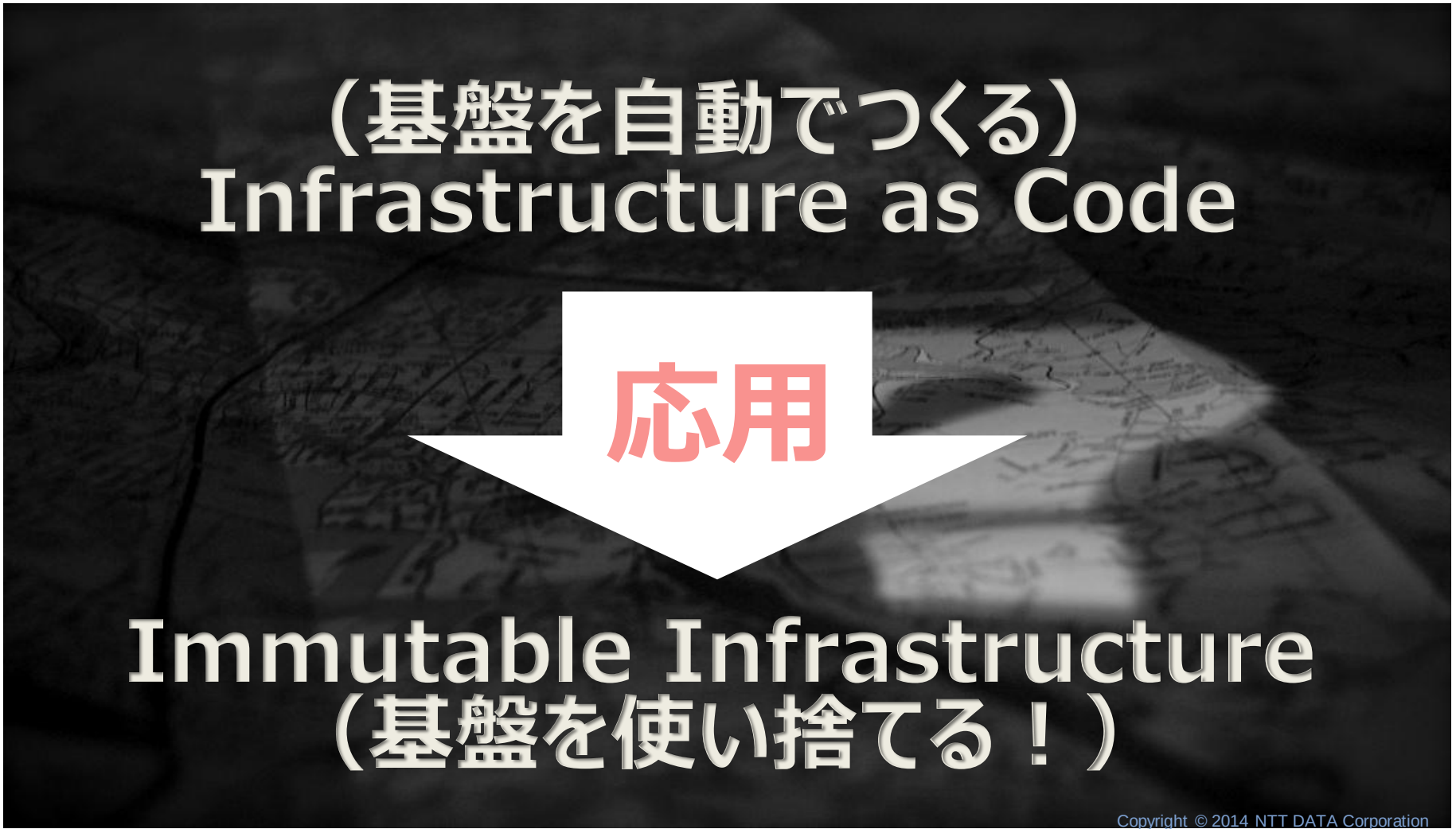
Point 2
基盤自動化チームワーク

Point 3
基盤自動化の応用
Immutabel Infrastructure

NTT DATA

Point 3: Immutable Infrastructure

Infrastructure as Code　応用編



NTT DATA

Point 3: Immutable Infrastructure

基盤を使い捨てる。なんのため？

Immutable Infrastructure
基盤の使い捨て

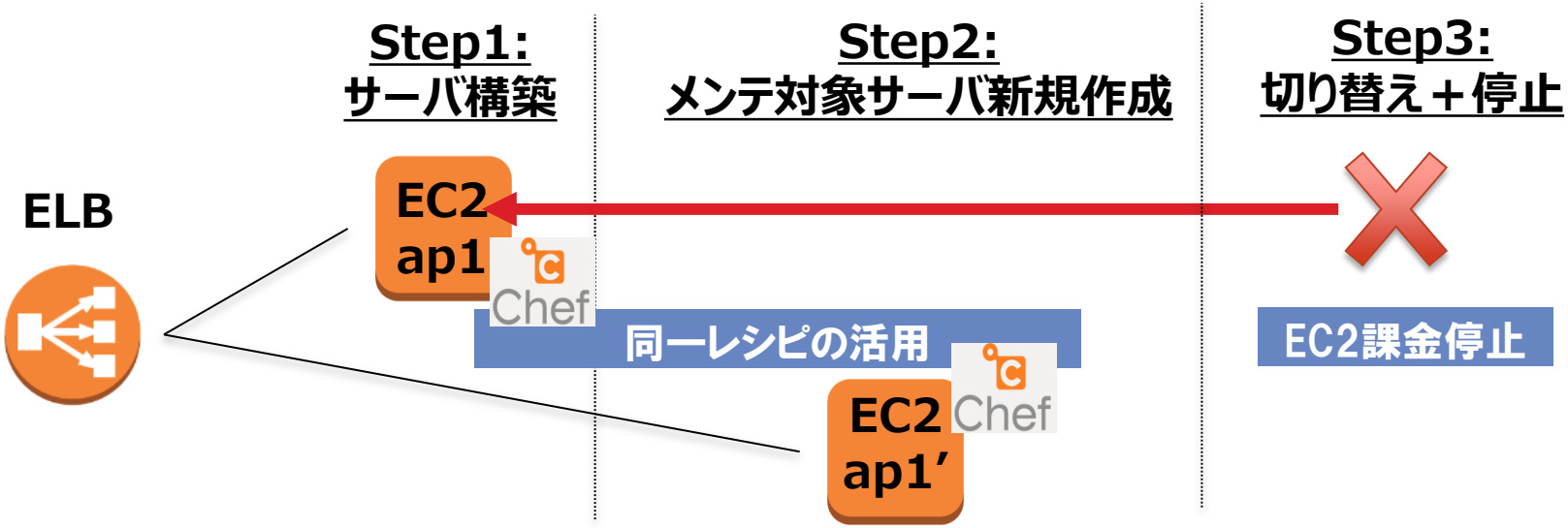
本番サービスを“止めない”こと
を重視してImmutableにしている！！

Point 3: Immutable Infrastructure

商用サービスサーバを絶対壊さない！

本番システムに手を入れるから壊れる

じゃぁ、手をいれなければ良いんだ！



NTT DATA

Point 3: Immutable Infrastructure

万全の運用を実施するための答え。

1. 実機(サーバ)を直接なおす
2. 自動化コードを修正し、再実行
3. 故障処理表番号ごとに差分の自動化コードを作成し、実機に適用

4.基盤を使い捨てる！

変更加えたレシピで、新規サーバ構築して、切替作業を実施することで、安全性UP

NTT DATA

Session Ending

エンタープライズシステムにおけるAWSのスピードを最大化するInfrastructure as Code

- ✓ **厳密な“日本式システム管理”に耐えられる仕組みや工夫が重要（Dry-runやルール作り）**
- ✓ **高難易度技術を簡単にするための“仕掛け”や、効率的に作るための“体制”を整備し、効果倍増**
- ✓ **基盤自動化を応用することで、手軽にImmutableInfrastructure**

NTT DATA

Next Session

オープンイノベーション
～　なんでも組み合わせる新発想　♪　～

×

×

クラウドベストプラクティス

基盤自動化

基盤作業グローバル化（オフショア化）

amazon web services

Chef

SERVERSPEC

Hinemos

NTT DATA

AWS上のシステムはこう作る！

#2 Hinemosで実現するクラウド運用自動化

株式会社NTTデータ
基盤システム事業本部　システム方式技術事業部
長妻 賢

NTT DATA

構築しっぱなしにしていませんか！？

運用管理こそがシステムの本番です！

運用管理って？

NTT DATA

システム運用

システム運用(－うんよう)とは、主にコンピュータ上で稼動し、さまざまなサービスを提供しているシステムが停止することなく、利用顧客に対してつつがなくサービスを提供できるよう当該環境を維持管理すること。

……

保守運用

保守運用ではサービスが・・・・・メーカーが提供する障害対策**パッチ情報などを確認してシステムへ適用する定例保守**作業、日々蓄積される業務データなどを有事の際に復旧させる事ができるよう**バックアップの取得、保管を行うバックアップ作業など**が保守運用にあたる。

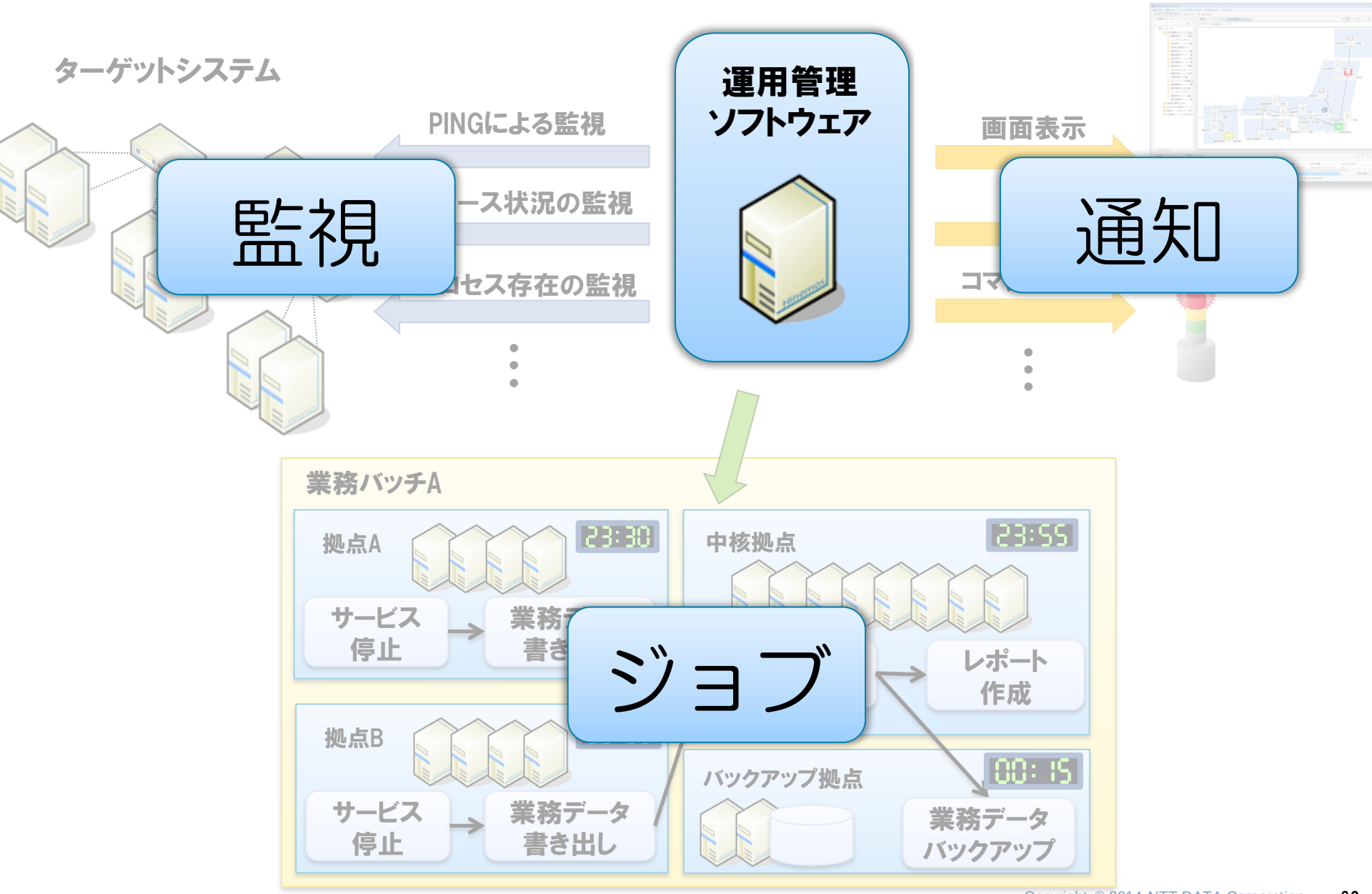
障害対応

障害が発生した際にいち早くそれを検知する為の仕組みを組み込んで**日々確認作業を行う監視**運用、障害箇所を調査・特定し**サービス復旧に向けて作業を行う障害対策**などが障害対応にあたる。

ウィキペディア，2008,「システム運用」『ウィキペディア日本語版』,
（2014年6月24日取得，http://ja.wikipedia.org/w/index.php?title=システム運用&oldid=21798980）.

運用管理ソフトウェアって？

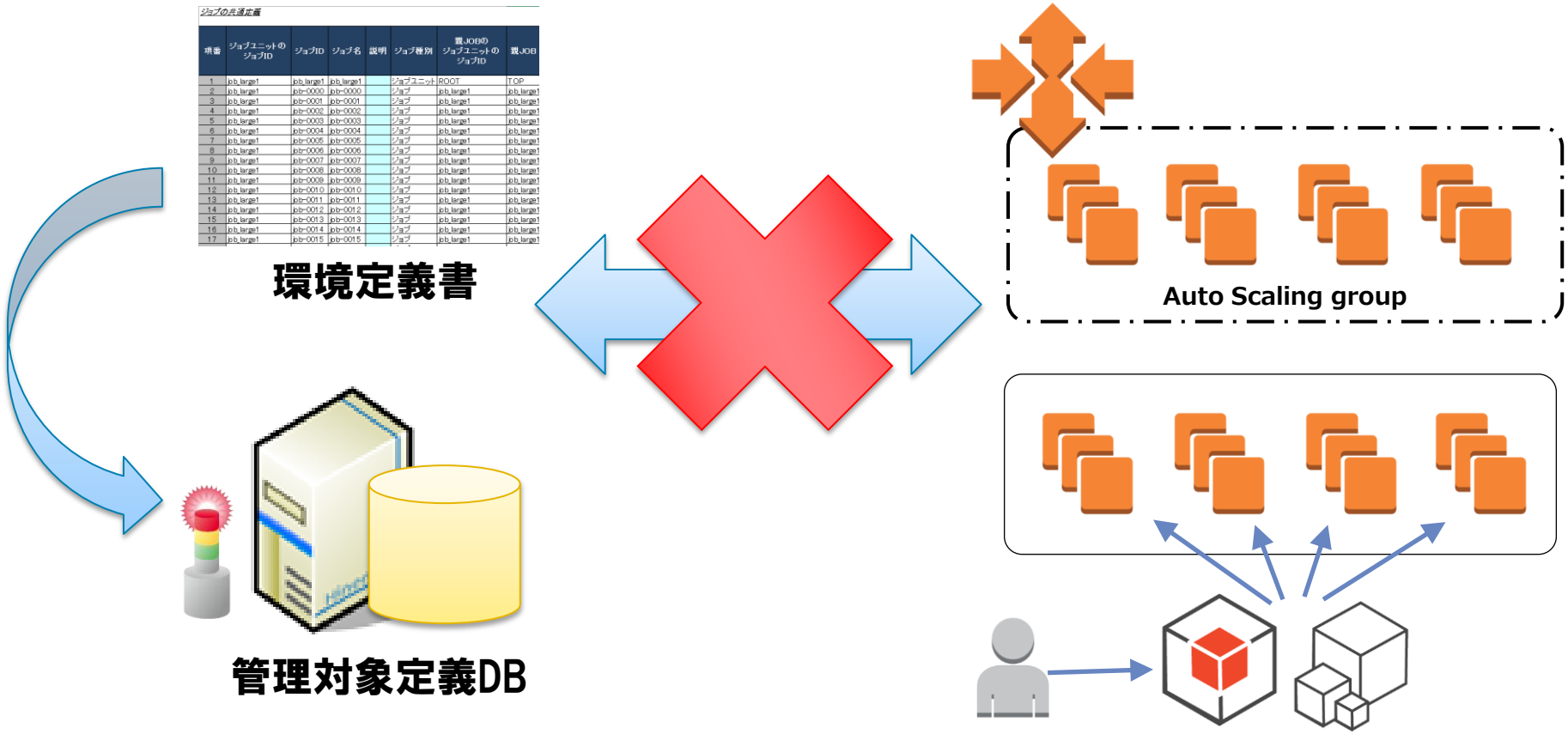
NTT DATA



クラウド上のシステムを運用管理する難しさ

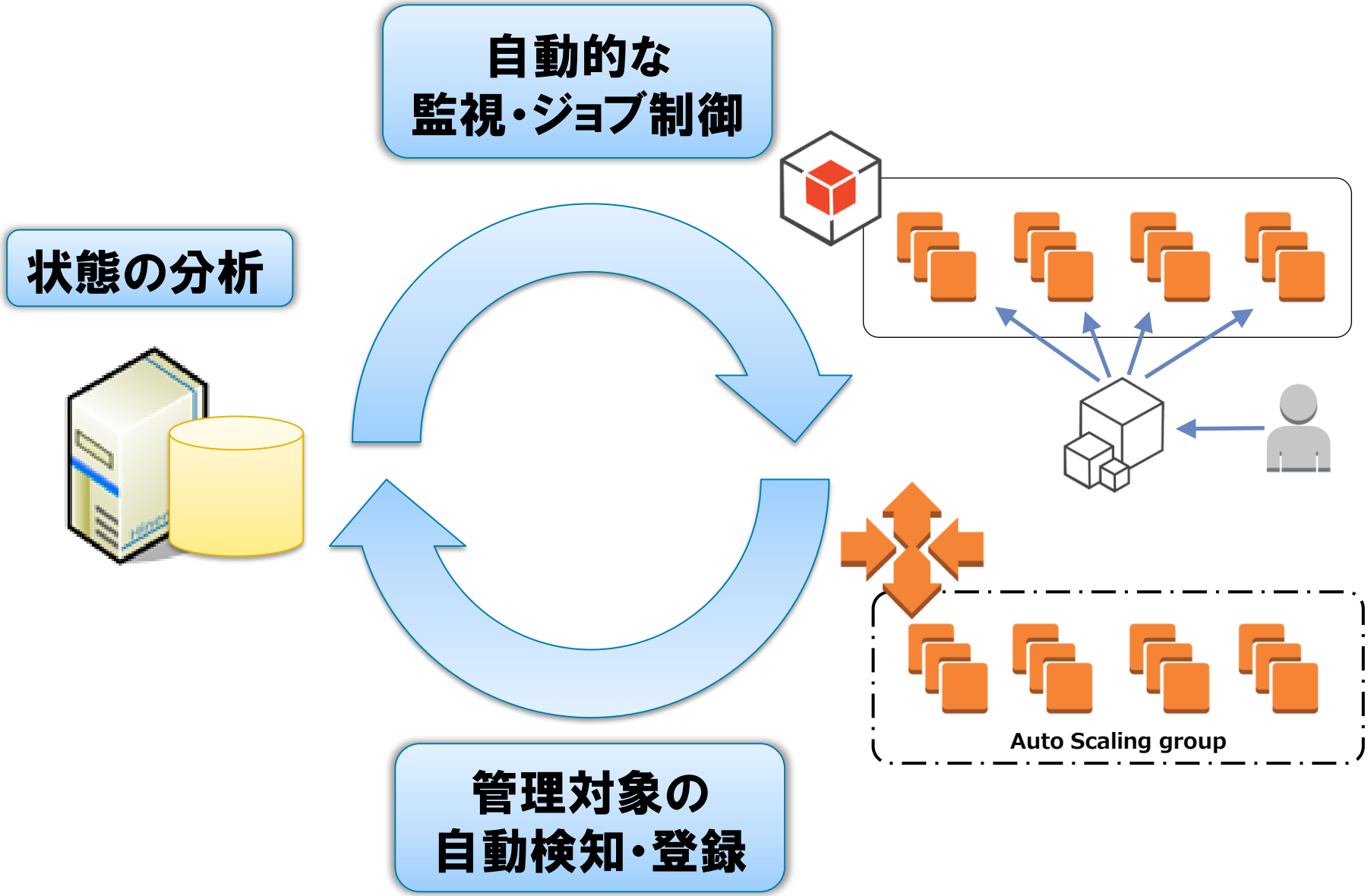
NTT DATA

オンデマンド・スケーラブルにリソースが確保できるクラウドの最大のメリットが、運用管理を難しくしています。



クラウドのメリットを活かした運用管理

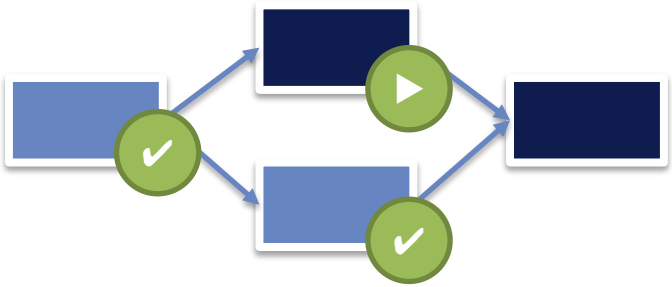
NTT DATA



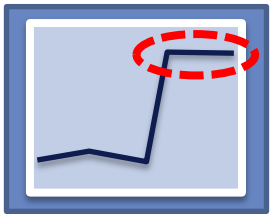
HinemoS ならそんな運用管理が実現できます

運用管理に必要な多様な機能を備えたソフトウェア

監視



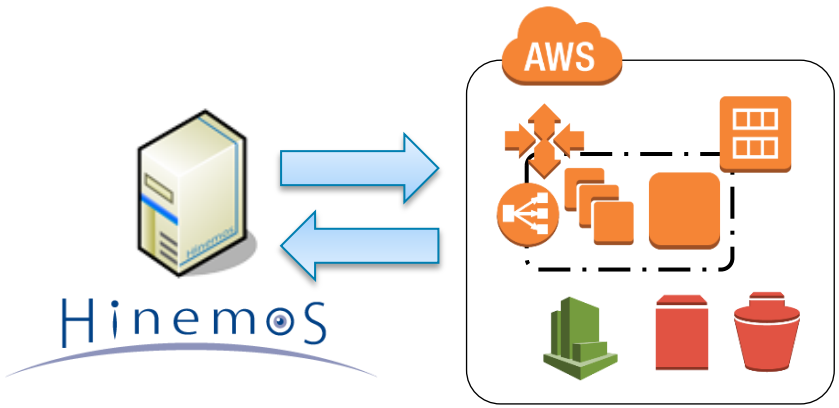
ジョブ制御



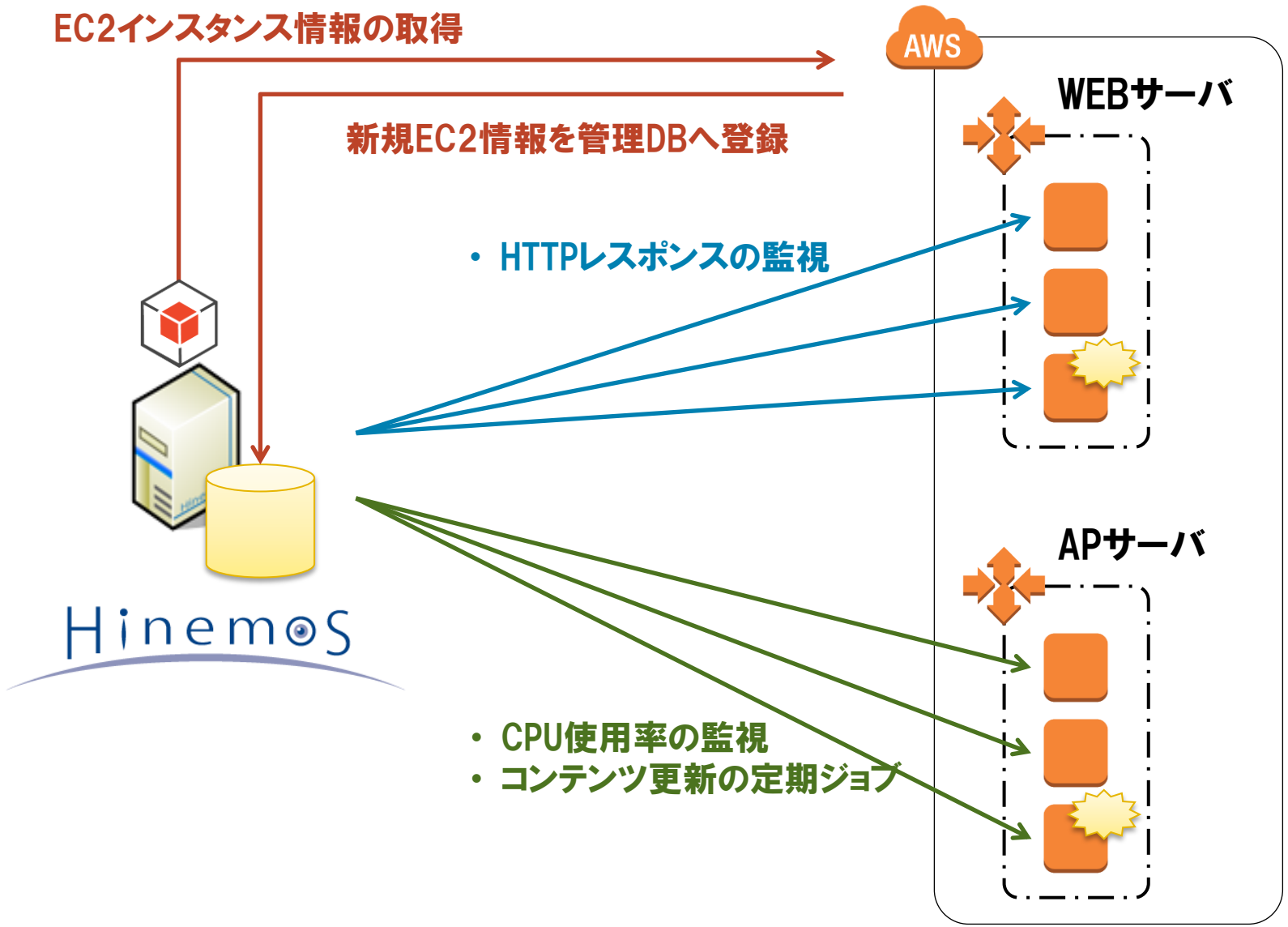
パフォーマンス管理

クラウドにいち早く対応！

- 構成変更への自動追随
- クラウドインスタンスの各種制御
- 課金情報の可視化・監視



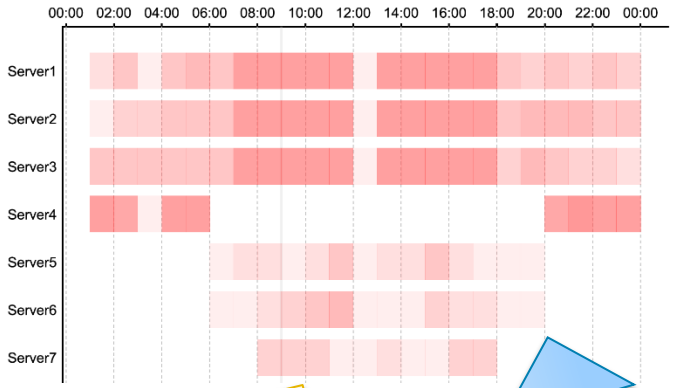
サーバ登録から監視・ジョブまでシームレスに



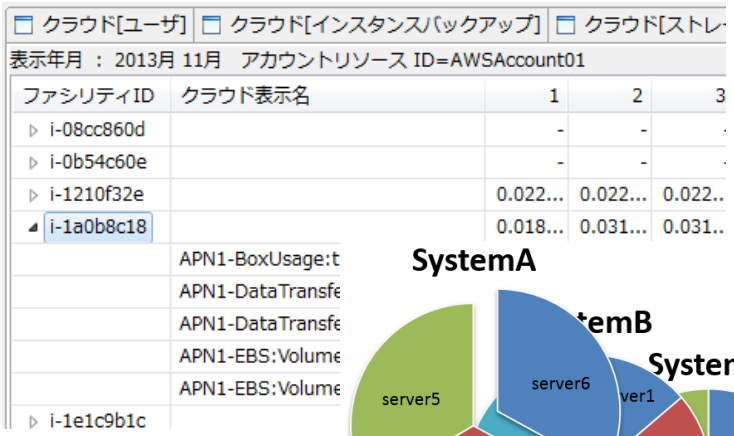
システムの様々な分析をワンストップに！
クラウド利用料金分析もお手の物

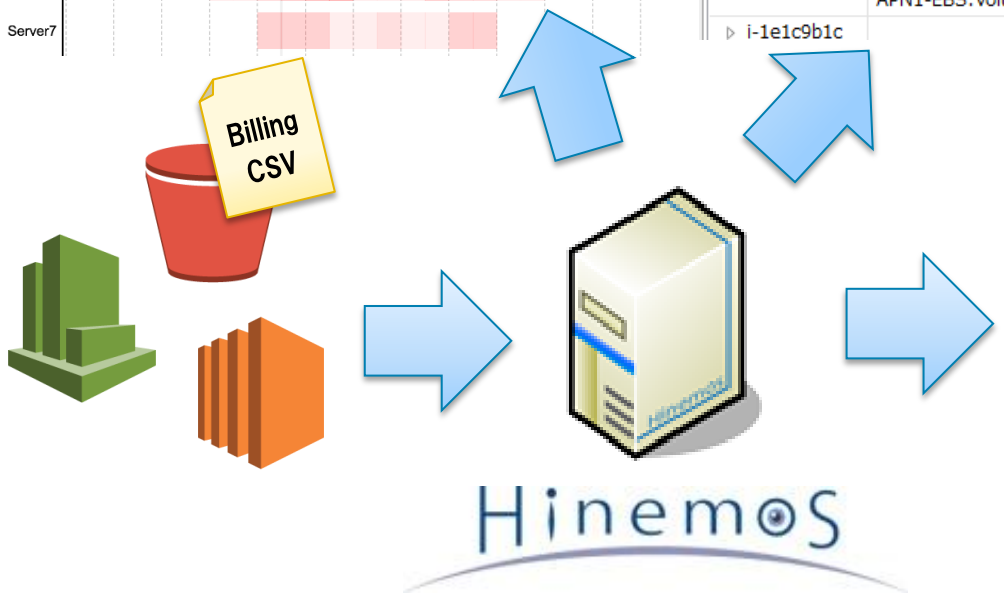
NTT DATA

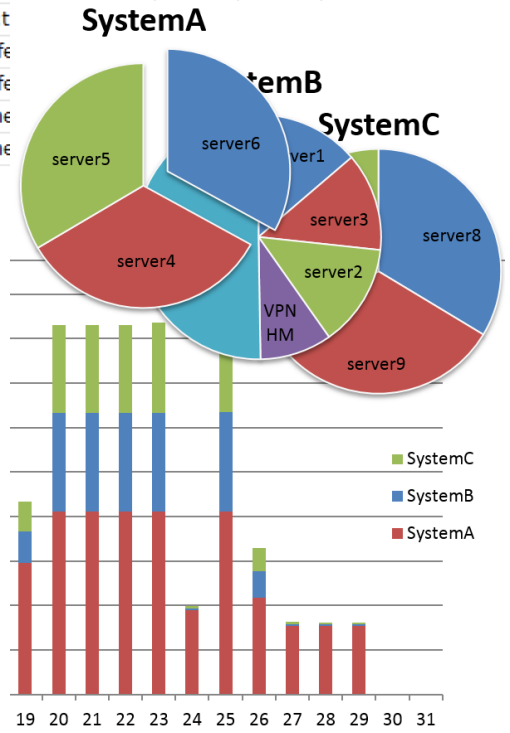
サーバリソースの可視化



サーバ・システム単位の課金情報







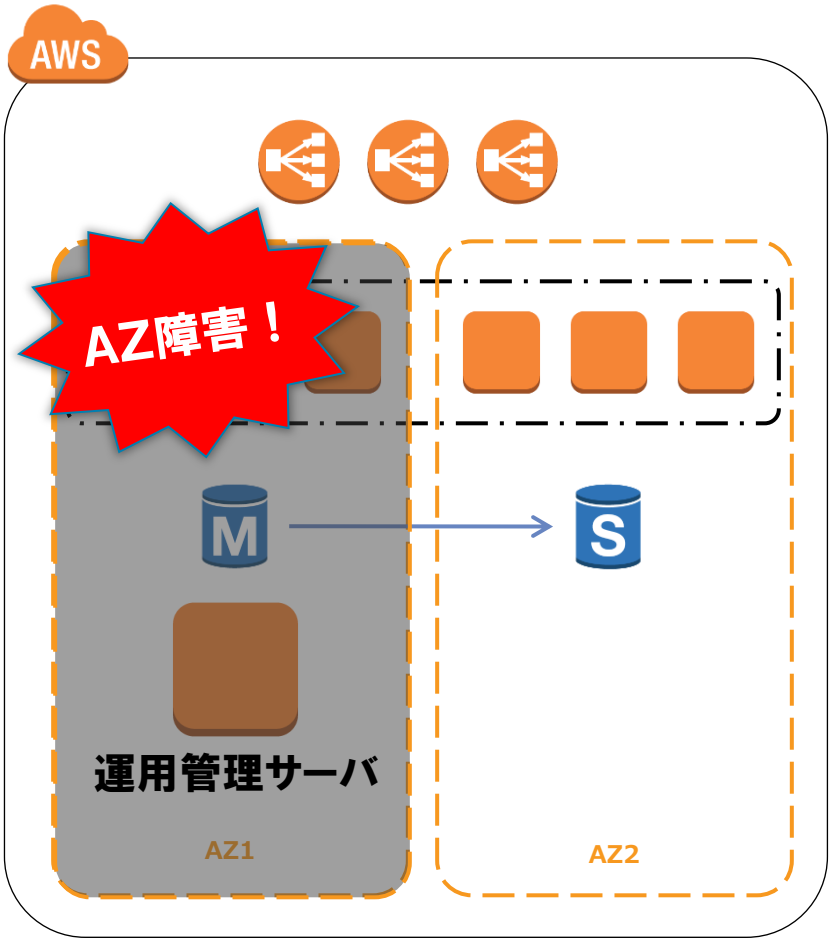
NTT DATA

業務継続の鍵は、運用管理にあり！

止められないシステム、どう自動運用する？

高いサービスレベルを求められるシステム

多重化してシステムの継続性を向上させたけれども・・・

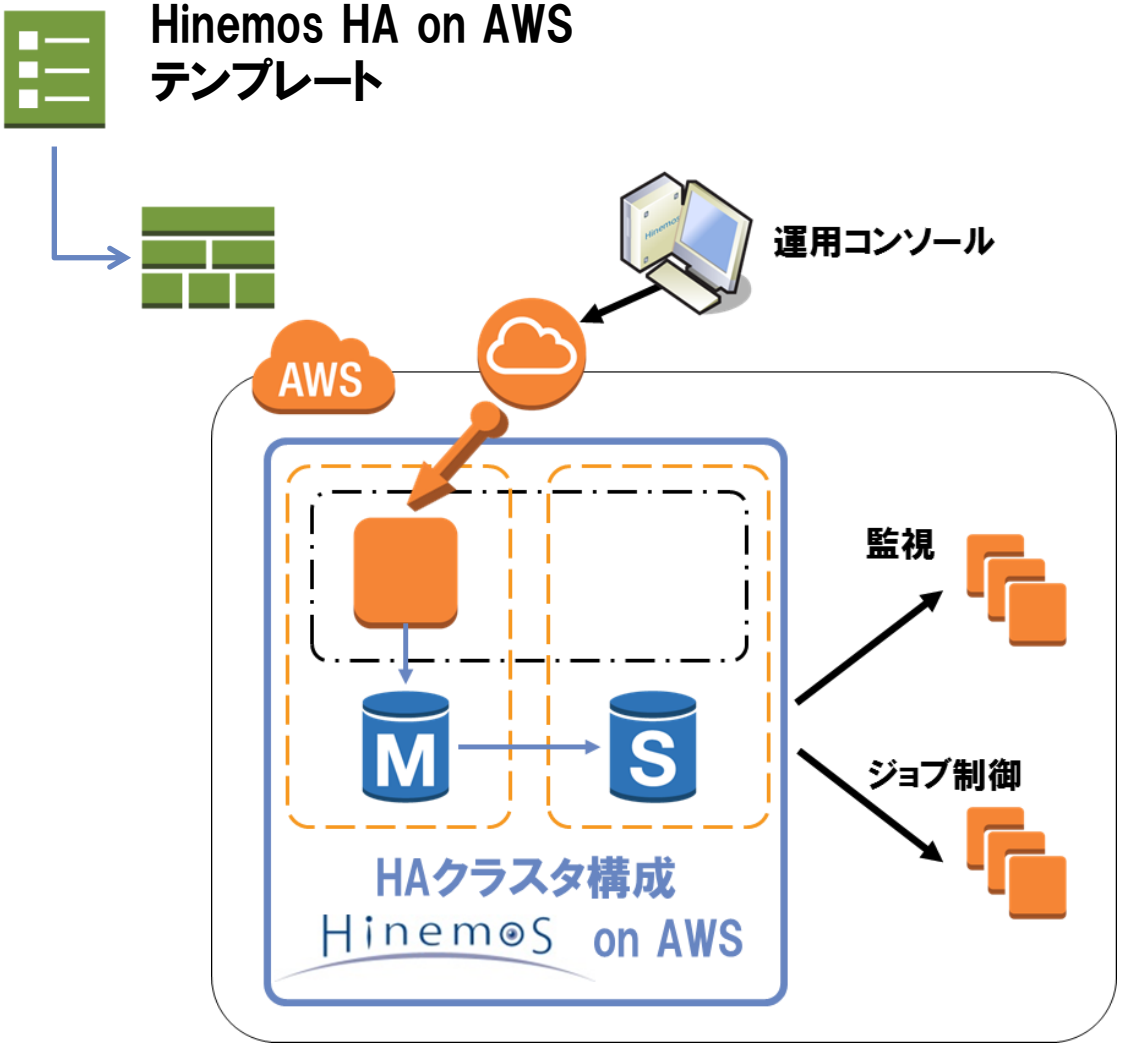


運用管理サーバは機能不全！！

- 障害の回復処理はできた・・・？
- SEは障害に気づけた・・・？
- バックアップは取れてるの・・・？
- さらなる障害が起きたら・・・？

運用管理をノンストップに！

Hinemos HA on AWS



9月リリース予定！
ブースにてデモ実施中

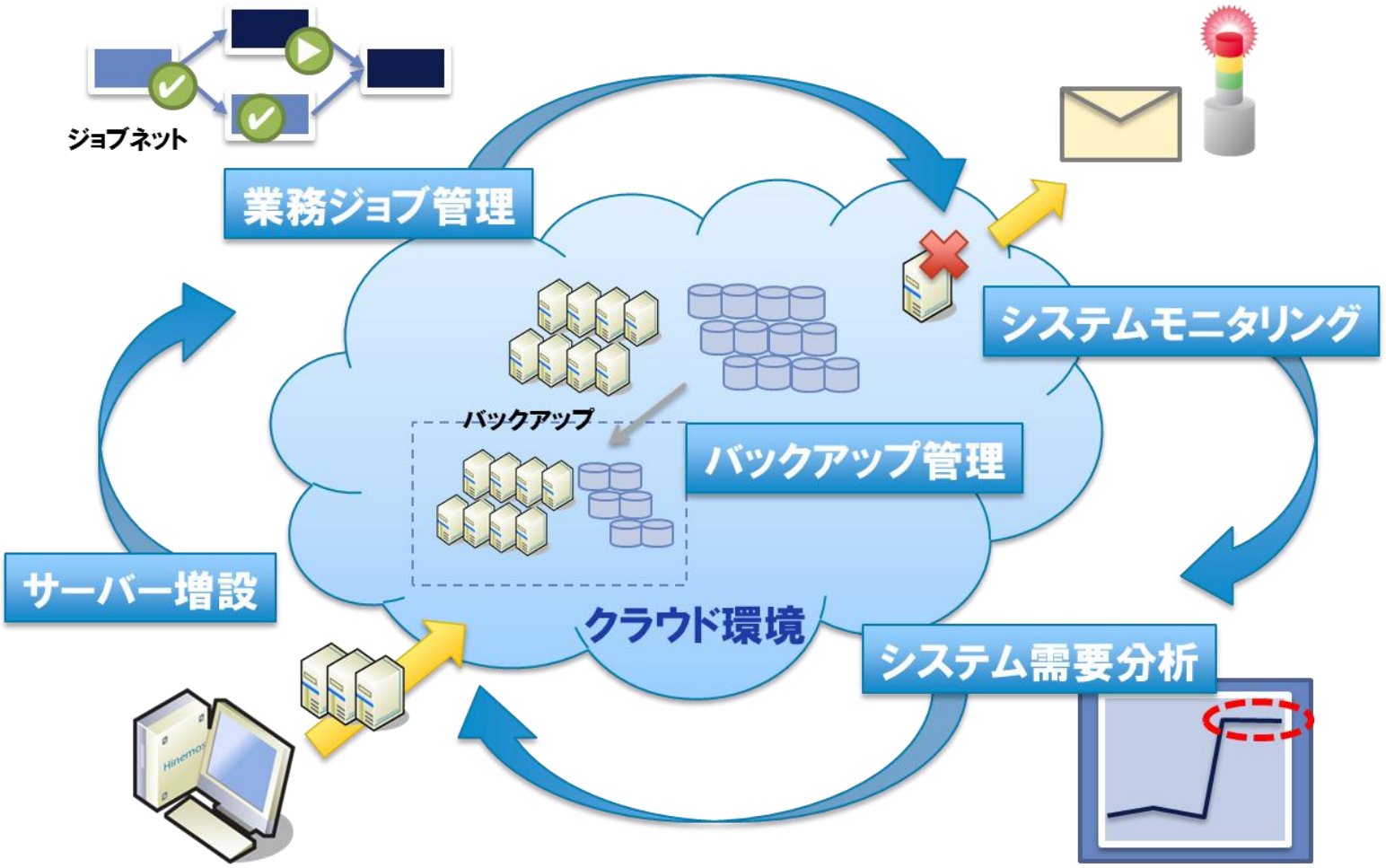
- **簡単なデプロイ**
- **HAを意識不要**
- **ジョブ、監視を自動継続**

NTT DATA

ということで
クラウド上のシステムを自動運用するなら・・・

AWSにいち早く対応した Hinemos でトータルに管理！

NTT DATA



数字で見るHinemos

NTT DATA

460,000 Downloads

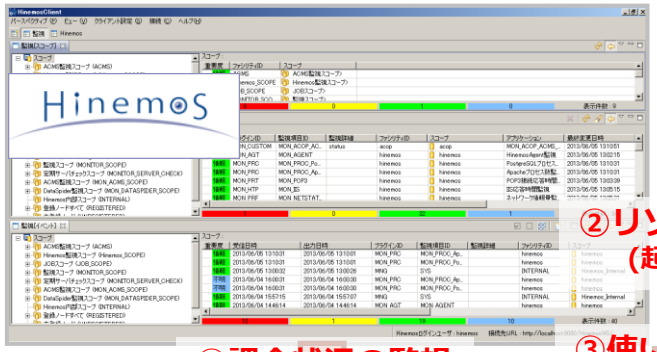
500+ Systems

39 Partners

34 Versions

1000+ Nodes

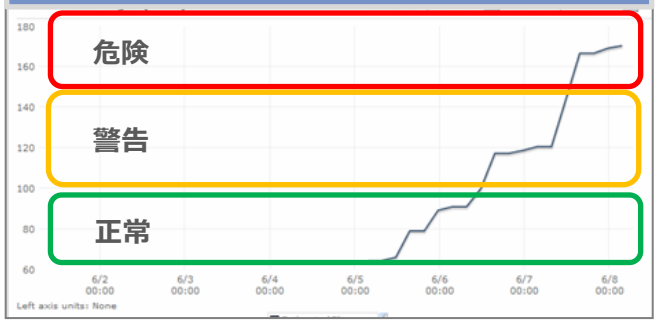
AWS利用料金の「監視」・「抑制」・「制御」にHinemosを活用！

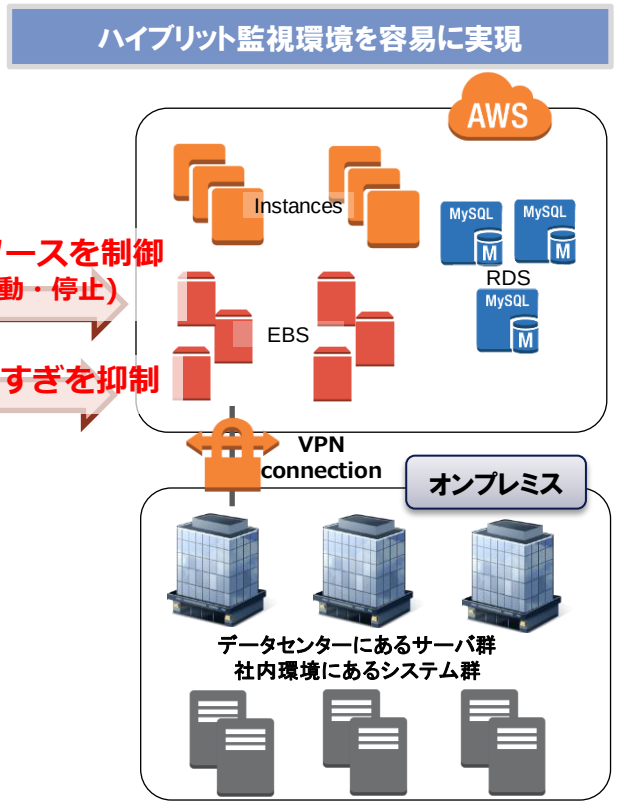


①課金状況の監視

②リソースを制御
（起動・停止）

③使いすぎを抑制





お客様名
株式会社アールシーコア様

提供Hinemosパートナー
株式会社ホロンテクノロジー

Holon Technology

システム構成
AWSとオンプレミス環境との
ハイブリッドクラウド

Hinemos導入メリット
AWS利用料金の抑制
運用の自動化

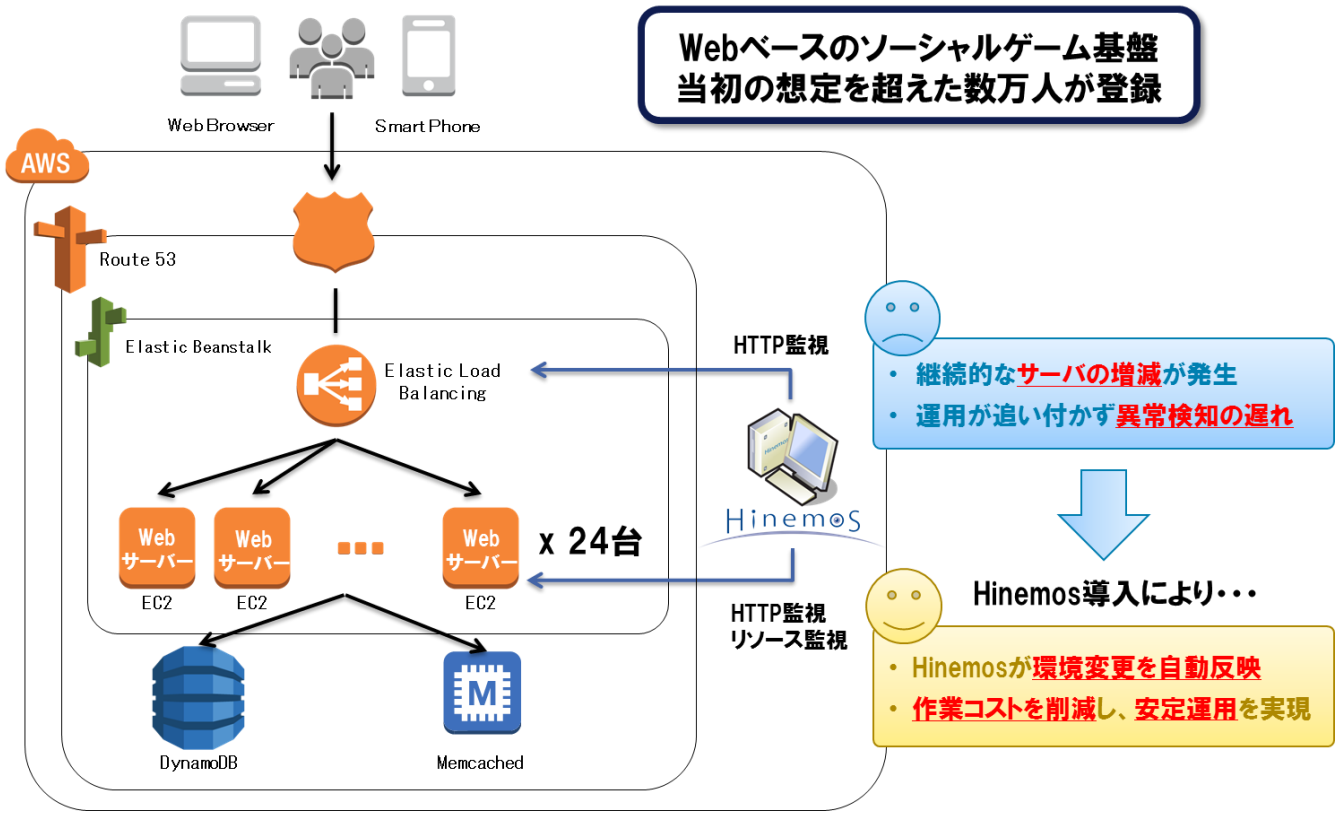
Hinemos関連サービス
CloudAL（クラウダル）

お客様の声

【監視】:「使い過ぎてしまうのでは」という不安を、Hinemosが監視してくれるので、安心して利用することができた
【抑制】:「使わない時は停止できる」というAWSのメリットをHinemosが柔軟にコントロールしてくれるので、利用料金を抑制できた
【制御】:オンプレミス環境とAWS環境を一元管理できるので、定常業務（バックアップ等）の自動化が制御でき、運用負荷を軽減できた

事例：株式会社DMM.com様（株式会社アトミテック提供）

拡張を続けるシステムの監視を自動化し、安定稼働を実現



お客様名
株式会社DMM.com様

提供Hinemosパートナー
株式会社アトミテック

システム構成
AWS上のWebベースの
ソーシャルゲーム基盤

Hinemos導入メリット
度重なるシステム拡張へ
自動的に追従できる運用

Hinemos利用機能
Hinemosクラウド管理オプション

お客様の声

・Web サーバーの障害をいち早く察知できるようになったため、これまでに比べ収益低下に繋がる時間を短縮できようになった。
・度重なるシステムの拡張にも自動で追随し、稼働中のシステムの状態を正確に把握が可能になった。
・今後の拡張計画において、運用のコストを削減し、安心してビジネスを展開できるようになった。

HinemosパートナーのAWS関連ソリューション

NTT DATA

AWSへのシステム移行・導入を
Hinemosパートナーが強力に支援します！

Holon Technology

クラウド運用管理支援サービス

CloudAL（クラウダル）

クラウドのコスト・スピード・柔軟性を
活かした運用（監視/ジョブ/運用）を支援

株式会社 クニエ

クラウドへの

ジョブ管理環境移行サービス

Hinemosによりクラウド活用の
コスト削減効果を高めます

株式会社 クニエ

Hinemos SAP連携

ソリューション

Hinemosのジョブ/監視機能でSAP製品の
統合管理を実現・ITコストを大幅に削減

Hinemosに興味を持ったら・・・

セミナ ※

- Hinemosで実現！ 物理・仮想・**クラウド環境の運用自動化**セミナ
  - 2014年7月23日（水）13:30～

- 『Hinemos with AWS』クラウド運用ソリューション紹介セミナ
  - 2014年9月3日（水）13:30～

書籍

- Hinemos統合監視［実践］入門（技術評論社）
  - 2014年9月頃発売予定

WEB・お問い合わせ

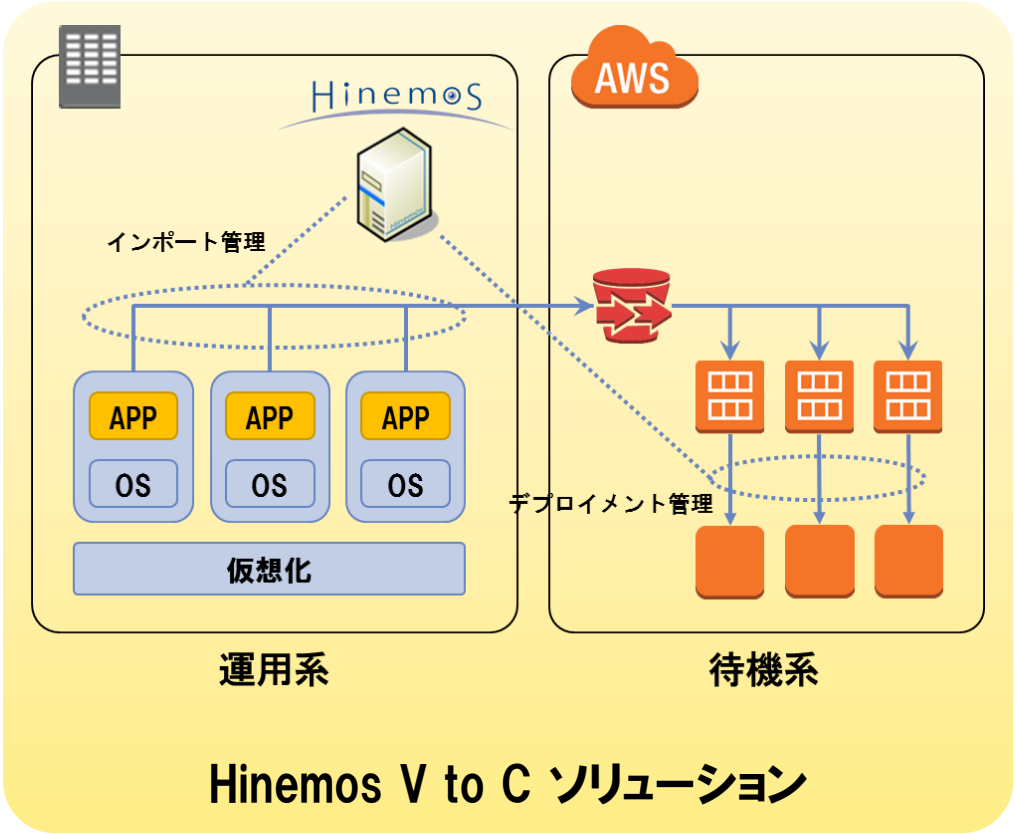
Hinemos　検索

※ 詳細はお手元の資料、またはHinemosで検索

NTTデータブースへ是非お越しください

Hinemos ブース

- Hinemosではじめる
  クラウド管理
- Hinemos HA on AWS
- 課金配賦機能
  （クラウド利用料金分析）
- Hinemos V to C（DR）
  ソリューション



Hinemos V to C ソリューション

BizXaaSマルチクラウドコントローラ ブース

「システム構成」をテンプレート管理し、AWS,プライベートクラウドにワンクリックでデプロイ

Intra-mart on AWS ブース

3300社を超える導入実績を誇るビジネスアプリケーション基盤Intra-mart
AWSでスピーディに構築する実績が増えています