AWS Summit

Tokyo

amazon web services

AWS Elastic Beanstalk, AWS OpsWorks, AWS CodeDeploy, AWS CloudFormation を使った自動デプロイ

アマゾン データ サービス ジャパン株式会社
ソリューションアーキテクト
舟﨑 健治

amazon web services

■Gold Sponsors

intel
Empowered by Innovation
NEC
NTT DATA
NetApp
Serverworks
HITACHI
Inspire the Next
TREND MICRO
WORKS APPLICATIONS

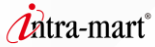
■Global Sponsors

infor

■Silver Sponsors

CTC
Challenging Tomorrow's Changes
FPT
JBCC
SKYARCH
あなたの側で、あなた以上に考える。
iSiD
IT Solution Innovator
ナレコムクラウド
NRI 未来創発
Dream up the future.
野村総合研究所
FUJISOFT
amazon web services

■Bronze Sponsors



■Global Tech Sponsors

■Logo Sponsors

amazon web services

ハッシュタグ **#AWSSummit**
で、**皆さんのツイート**が展示エリア
の**大画面に表示されます**

公式アカウント**@awscloud_jp**
を**フォローすると**、ロゴ入り
コースターを**プレゼント**

【コースター配布場所】
メイン展示会場、メイン会場1F受付、デベロッパーカンファレンス会場

amazon
web services

自己紹介

- 舟﨑 健治（ふなさき けんじ）
- 担当
  - アマゾン データ サービス ジャパン株式会社 ソリューションアーキテクト
  - 業種を問わず幅広くAWS利用をご検討されるお客様をサポート
  - AWS初心者向けWebinarを企画・運営中
- 好きなAWSのサービス：AWS OpsWorks

Agenda

- Introduction
- AWS Elastic Beanstalk
- AWS OpsWorks
- AWS CodeDeploy
- AWS CloudFormation
- まとめ

Introduction

Amazon EC2

ELB

Amazon RDS

Amazon S3

AWS Elastic Beanstalk

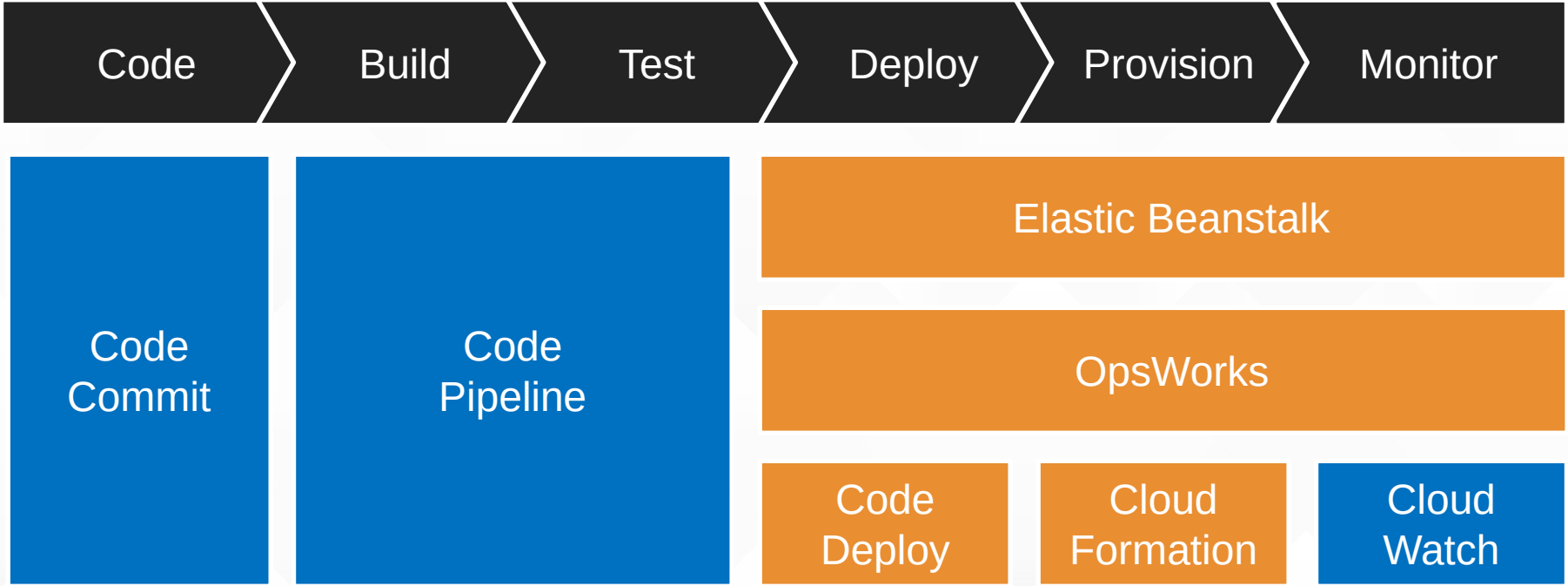
AWS OpsWorks

AWS CodeDeploy

AWS CloudFormation

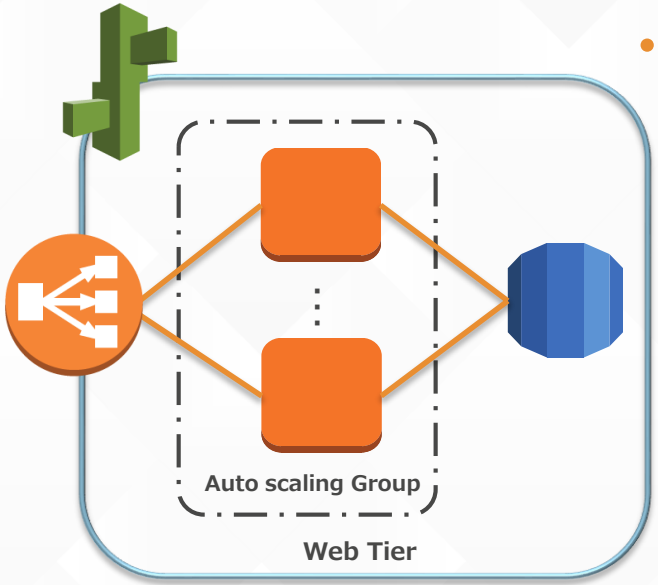
amazon web services

AWSから提供されているデプロイ＆マネージメント関連サービス



AWS Elastic Beanstalk

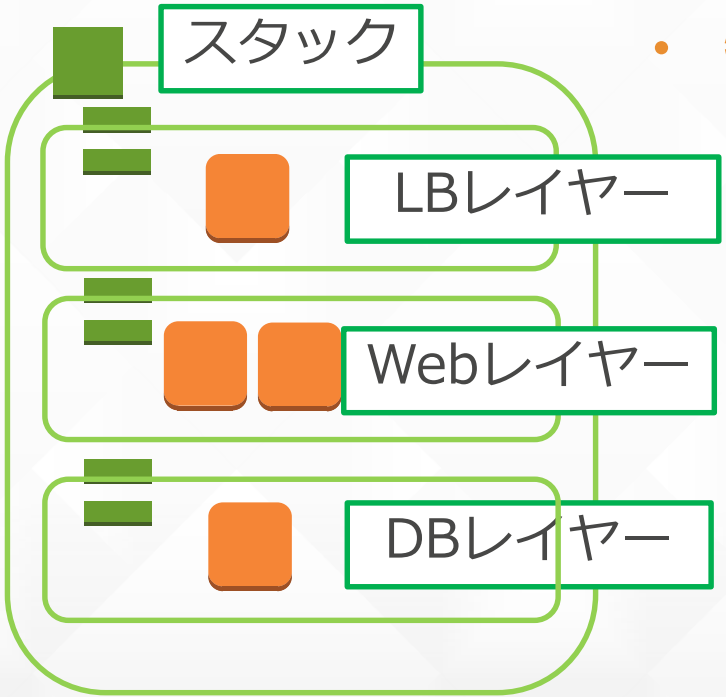
定番構成の構築・アプリデプロイの自動化サービス



- **特徴**
  - 速く簡単にアプリケーションをデプロイ可能
  - インフラストラクチャの準備＆運営からアプリケーションスタックの管理まで自動化
  - Auto Scaling によりコストを抑えながらスケーラビリティを確保
  - Java, PHP, Ruby, Python, Node.js, .NET, Go, Docker などに対応

AWS OpsWorks

多様なアーキテクチャをサポートするデプロイ・管理サービス

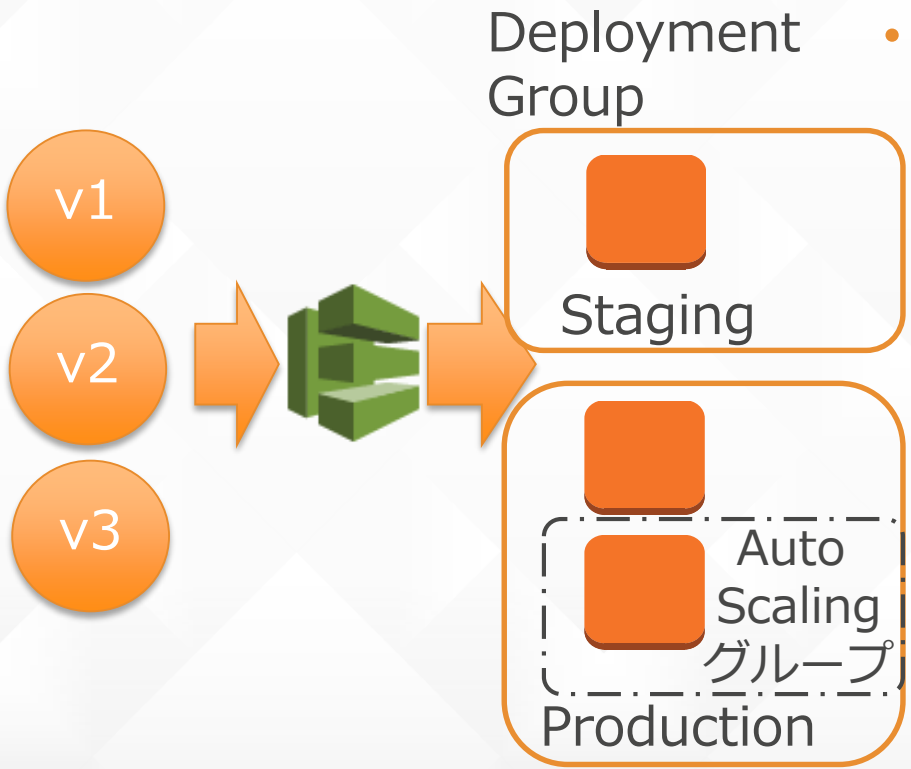


- **特徴**
  - Chefのレシピを使って、デプロイや運用タスクを自動化可能
  - ライフサイクルイベントにより動的な構成変更への対応が可能
  - 継続的な構成管理

amazon web services

AWS CodeDeploy

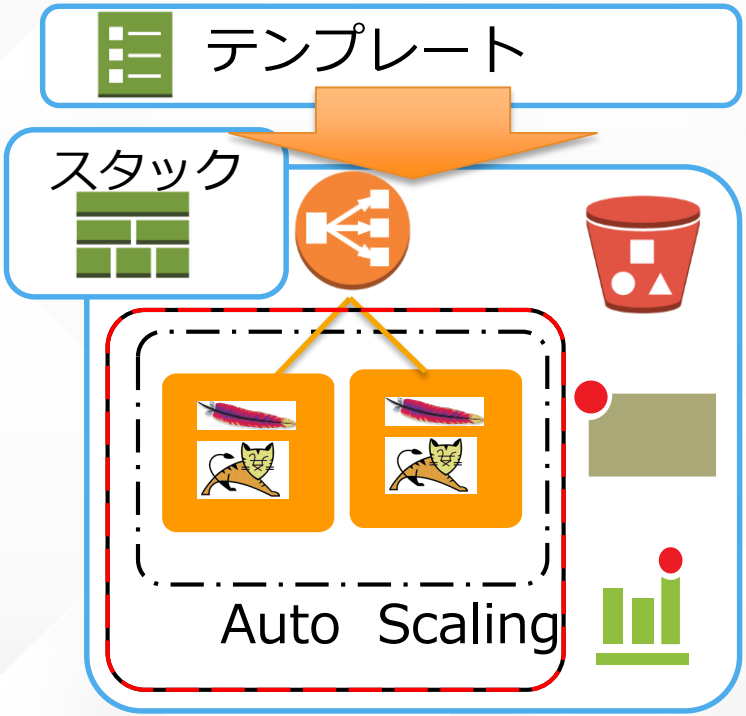
アプリケーションデプロイに特化したサービス



- **特徴**
  - Amazon.comと同様の仕組みで、管理されたデプロイを実現
  - エージェントをインストールするだけでEC2でもオンプレミスでも管理可能
  - グループ内に、一度にデプロイしたり1台ずつデプロイしたりと設定可能

AWS CloudFormation

設定管理 & クラウドのオーケストレーション サービス



特徴

- テンプレートを元に、EC2やELBといったAWSリソースの環境構築を自動化
- JSONフォーマットのテキストで、テンプレートを自由に記述可能

amazon web services

各種サービスを使ったWordpressのデプロイ手法

WordPress

AWS Elastic Beanstalk

AWS OpsWorks

AWS CodeDeploy

AWS CloudFormation

プロビジョニング

デプロイ

環境のカスタマイズ

各種サービスを３つの観点からご紹介

amazon web services

WordPressとは

- オープンソースのブログ／CMSプラットフォーム
- 動作環境
  - PHP
  - MySQL

構築するシステムの構成イメージ

注意事項

- どのAWSデプロイメント＆マネージメントサービスがベストであるかをお伝えする内容ではありません。

- 特定のWebアプリケーションのデプロイを各種サービスで実行する手法をご理解頂くことで、サービス選定の一助となれば幸いです。

AWS Elastic Beanstalk

amazon web services

AWS Elastic Beanstalk

Code | Build | Test | Deploy | Provision | Monitor

Code Commit

Code Pipeline

Elastic Beanstalk

OpsWorks

Code Deploy

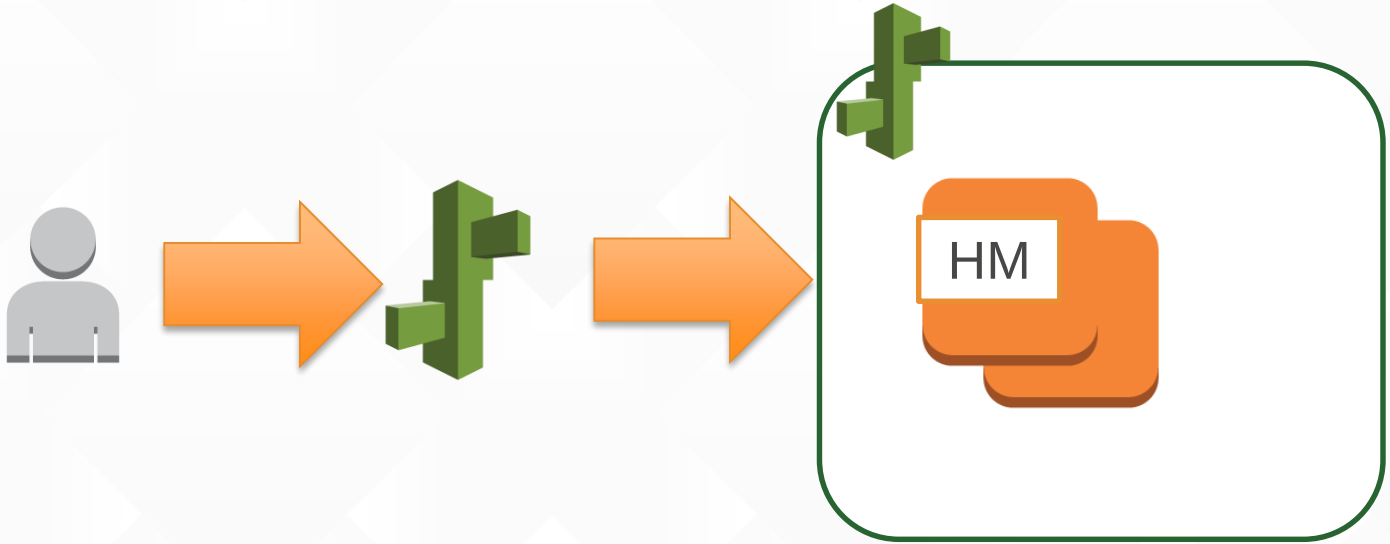
Cloud Formation

Cloud Watch

amazon web services

AWS Elastic Beanstalkによるプロビジョニング

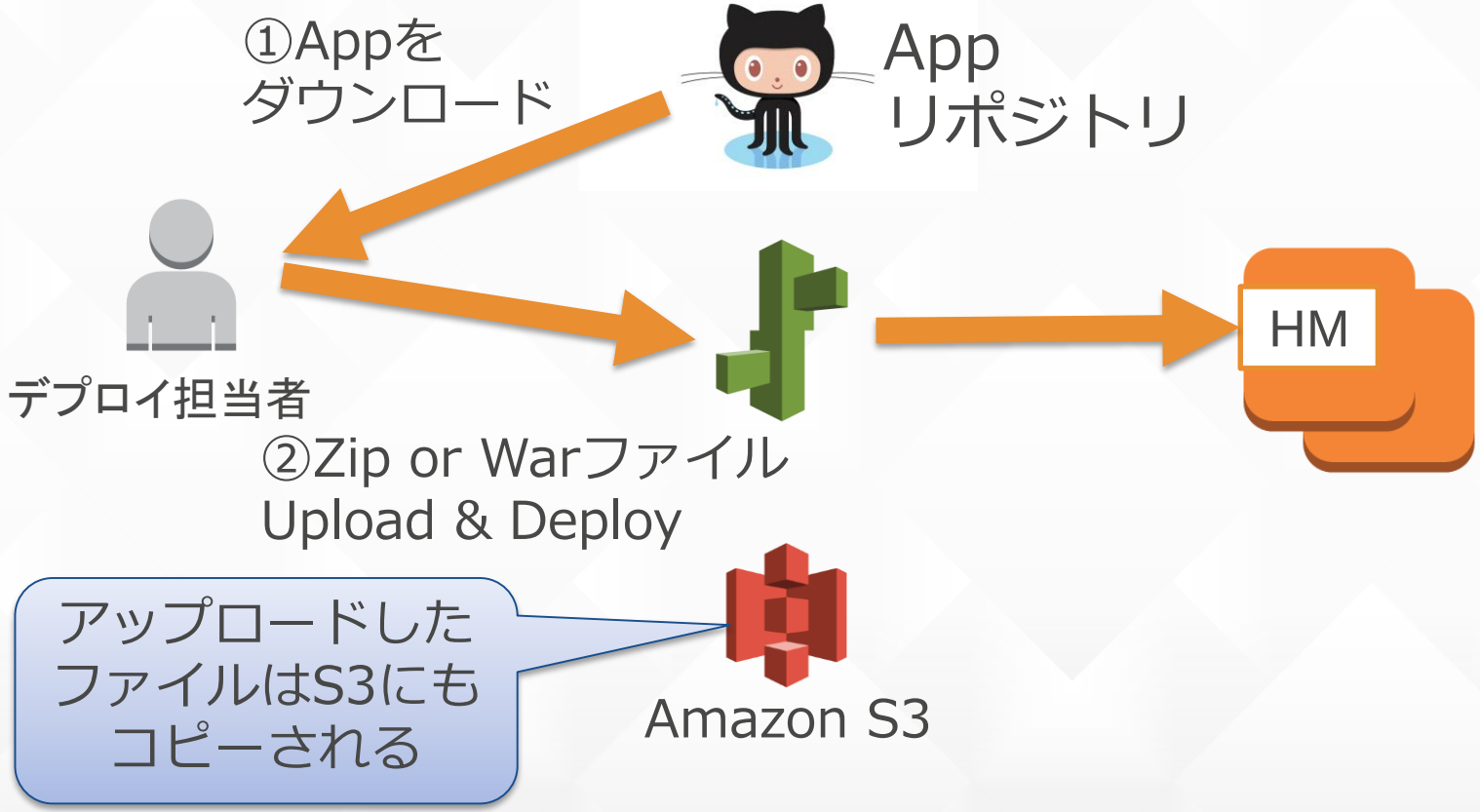
プロビジョニング



EC2インスタンス内部で
Host Managerが動作

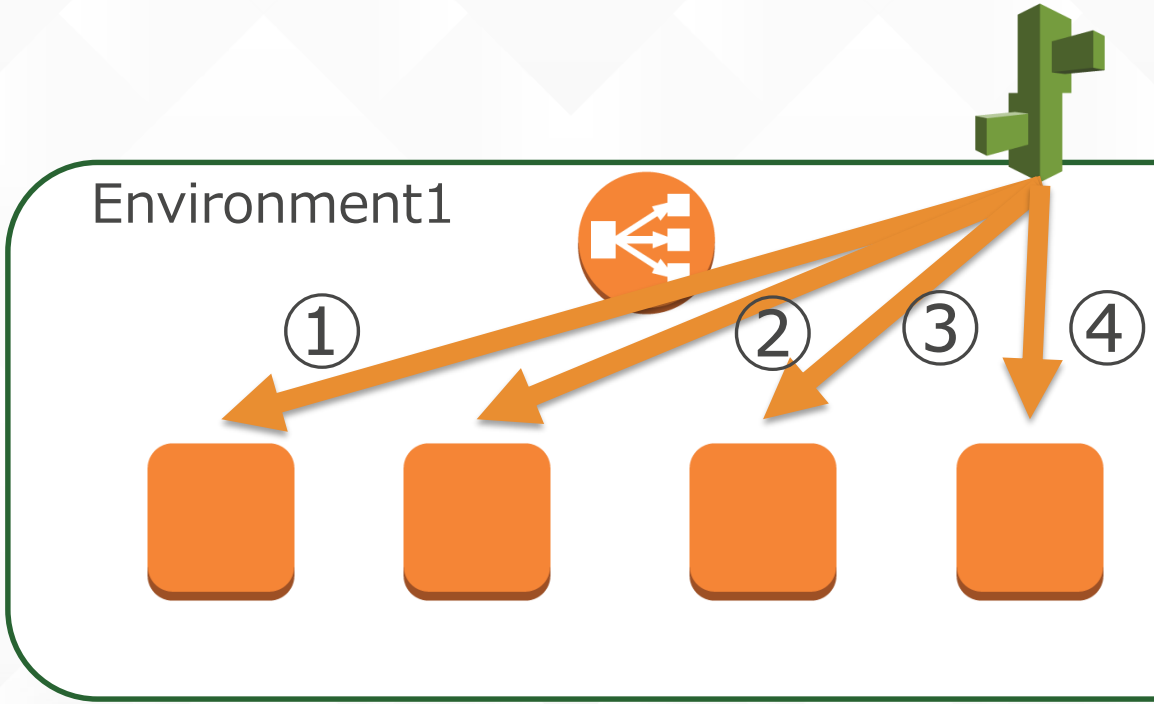
AWS Elastic BeanstalkとGitHubを使ったデプロイ

デプロイ



amazon web services

AWS Elastic Beanstalkを使った順次バッチ処理でのデプロイ

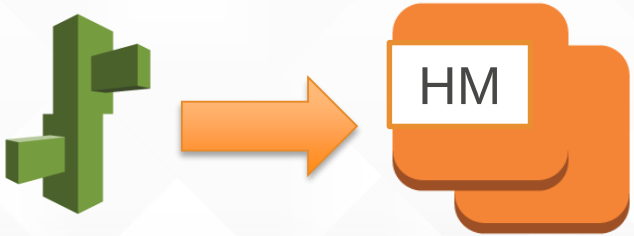


デプロイを
1度に1台ずつ、
1度に全体の25%ずつ、
1度に全部、
など調整が可能

アプリケーションコンテナをカスタマイズ

環境のカスタマイズ

- Elastic Beanstalk Configuration Fileで動作中のコンテナをカスタマイズ
  - 独自AMIを用いなくても必要なコンポーネントの追加が可能



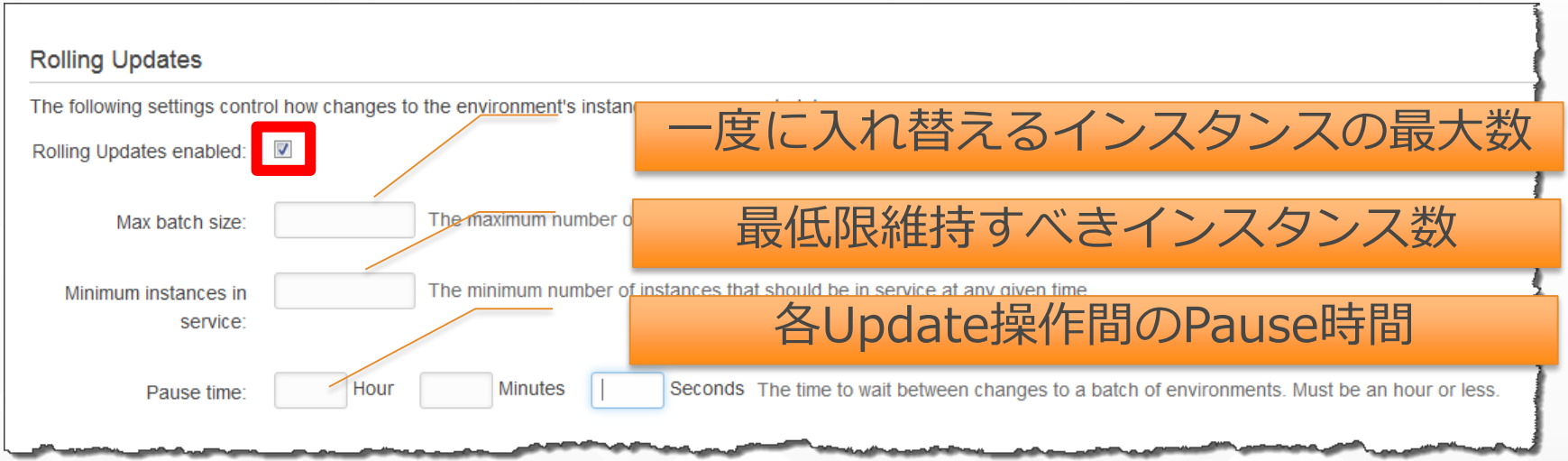

```
packages:
   yum:
      git: []

commands:
   01prepare:
      command: “scripts/prepare.sh”
```

yumでgitをインストールしつつ、特定のシェルスクリプトを実行する例

環境設定のRolling Update

Auto Scalingグループ内にあるインスタンスの置換えを伴う操作を一部ずつ実行(*)

– 内部的にはCloudFormationのUpdate Policyを利用



(*) デプロイの場合は、前述の順次バッチ実行を活用

AWS OpsWorks

amazon web services

AWS OpsWorks

Code Commit

Code Pipeline

Elastic Beanstalk

OpsWorks

Code Deploy

Cloud Formation

Cloud Watch

amazon web services

AWS OpsWorksによるプロビジョニング

プロビジョニング

①PHP App Serverレイヤーを作成

②PHP App Serverレイヤーにインスタンスを追加

③インスタンスを起動

PHP App Serverレイヤー

④OpsWorksエージェントが自動インストールされる

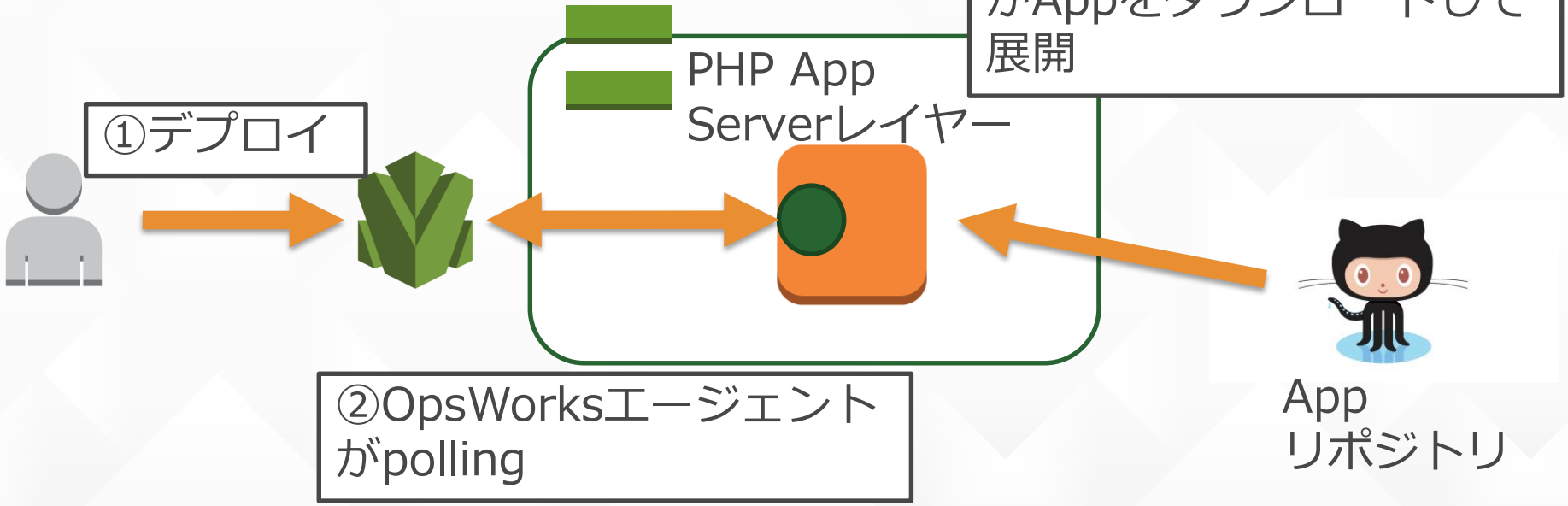
Cookbook リポジトリ

⑤エージェントがレシピをダウンロードして実行

amazon web services

AWS OpsWorksによるデプロイ

デプロイ



AWS OpsWorksのデプロイJSONを使った構成情報の取得

deploy JSONの例

```
"deploy": {
  “wordpress”: {
    "database": {
      "host": “dbhostname",
      "database": “dbname",
      "username": "awsuser",
      "password": "mypassword",
```

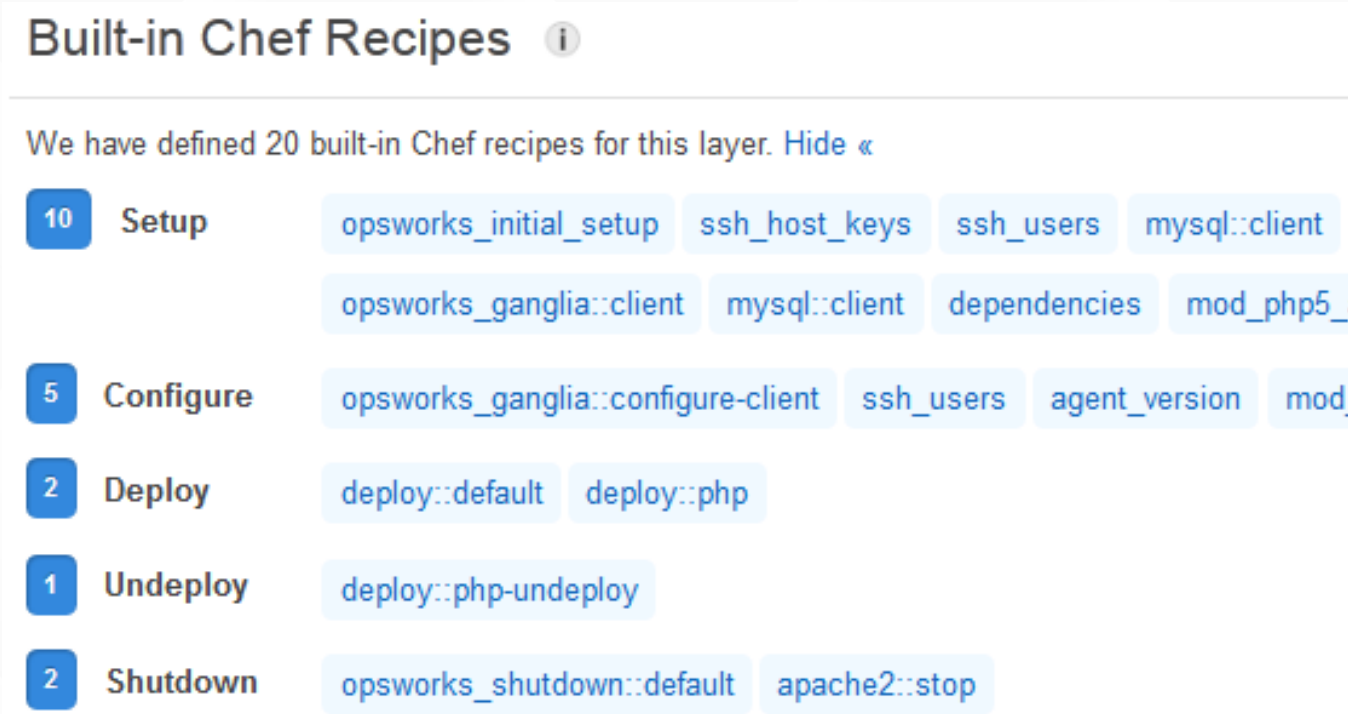
レシピから取得する例

```
dbhost = node[:deploy][:wordpress][:database][:host]
dbname = node[:deploy][:wordpress][:database][:database]
dbuser = node[:deploy][:wordpress][:database][:username]
dbpass = node[:deploy][:wordpress][:database][:password]
```

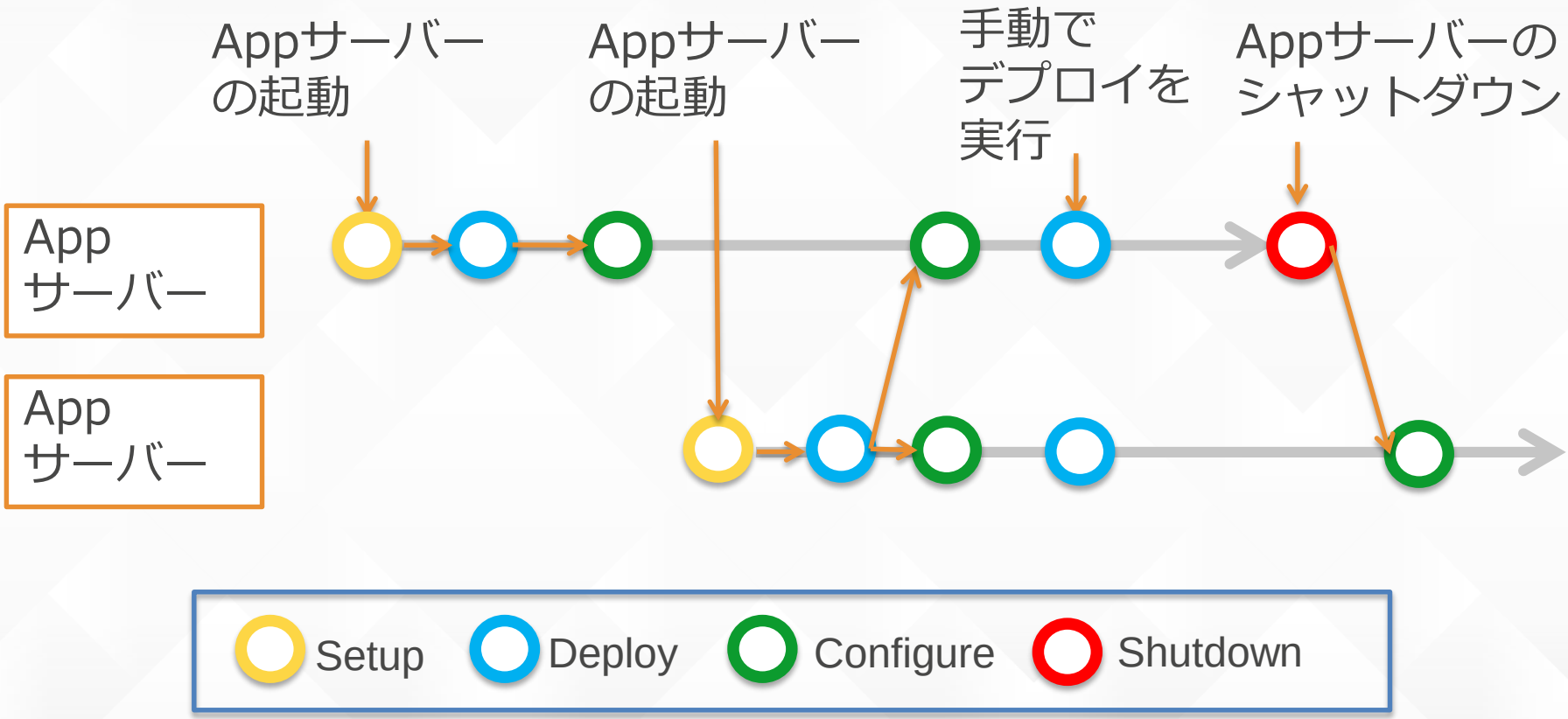
amazon web services

AWS OpsWorksの5つのライフサイクルイベント

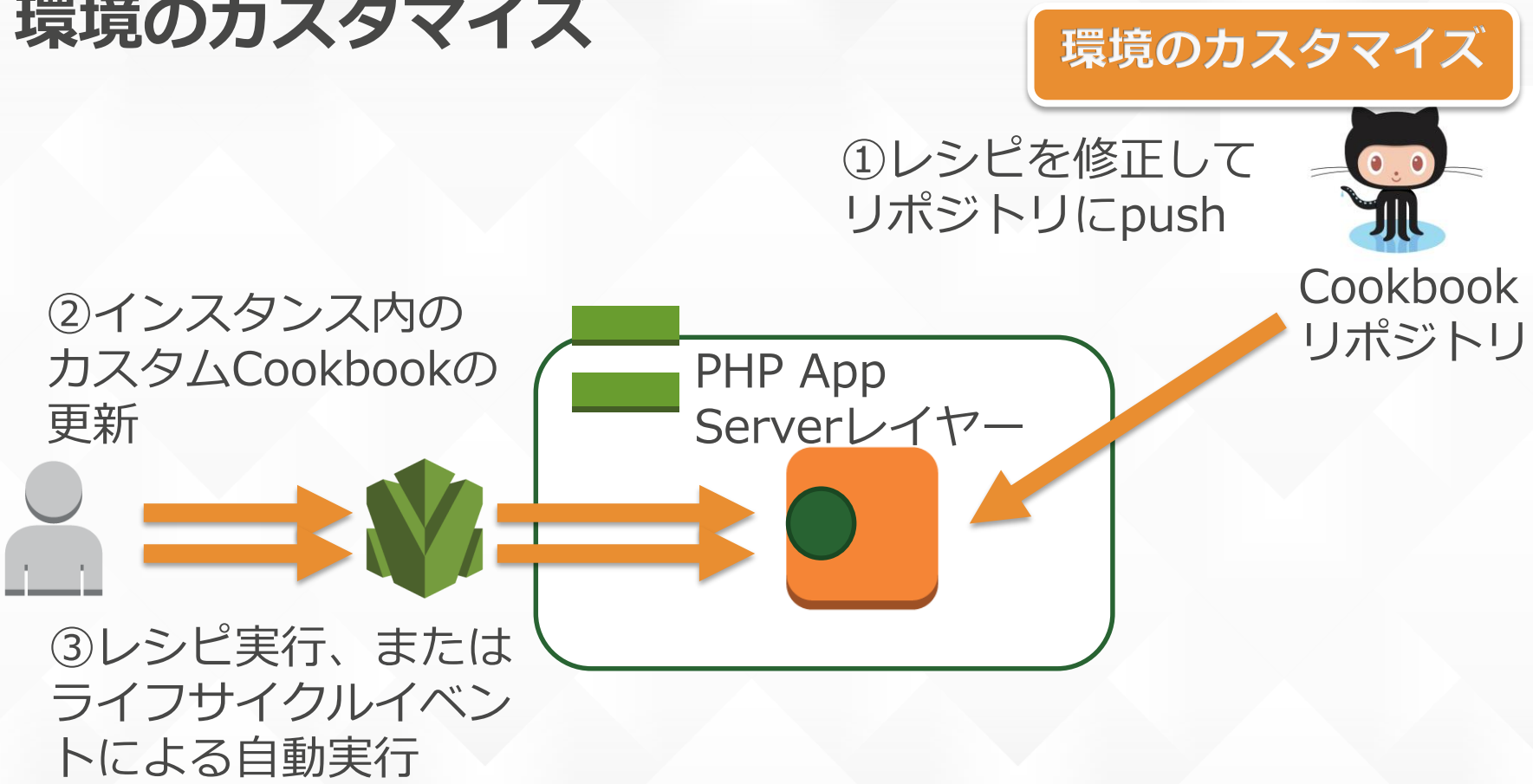
- Setup
- Configure
- Deploy
- Undeploy
- Shutdown



ライフサイクルイベントによる自動デプロイ



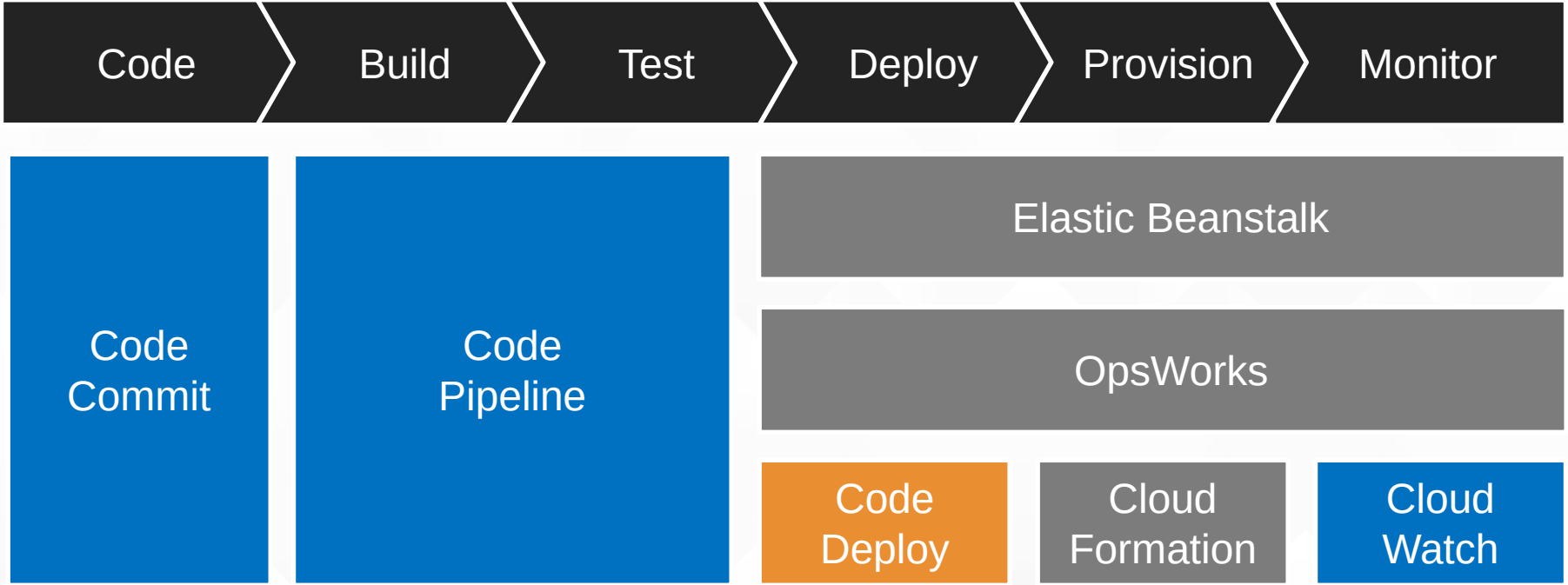
環境のカスタマイズ


環境のカスタマイズ
①レシピを修正して
リポジトリにpush
Cookbook
リポジトリ
②インスタンス内の
カスタムCookbookの
更新
PHP App
Serverレイヤー
③レシピ実行、または
ライフサイクルイベン
トによる自動実行

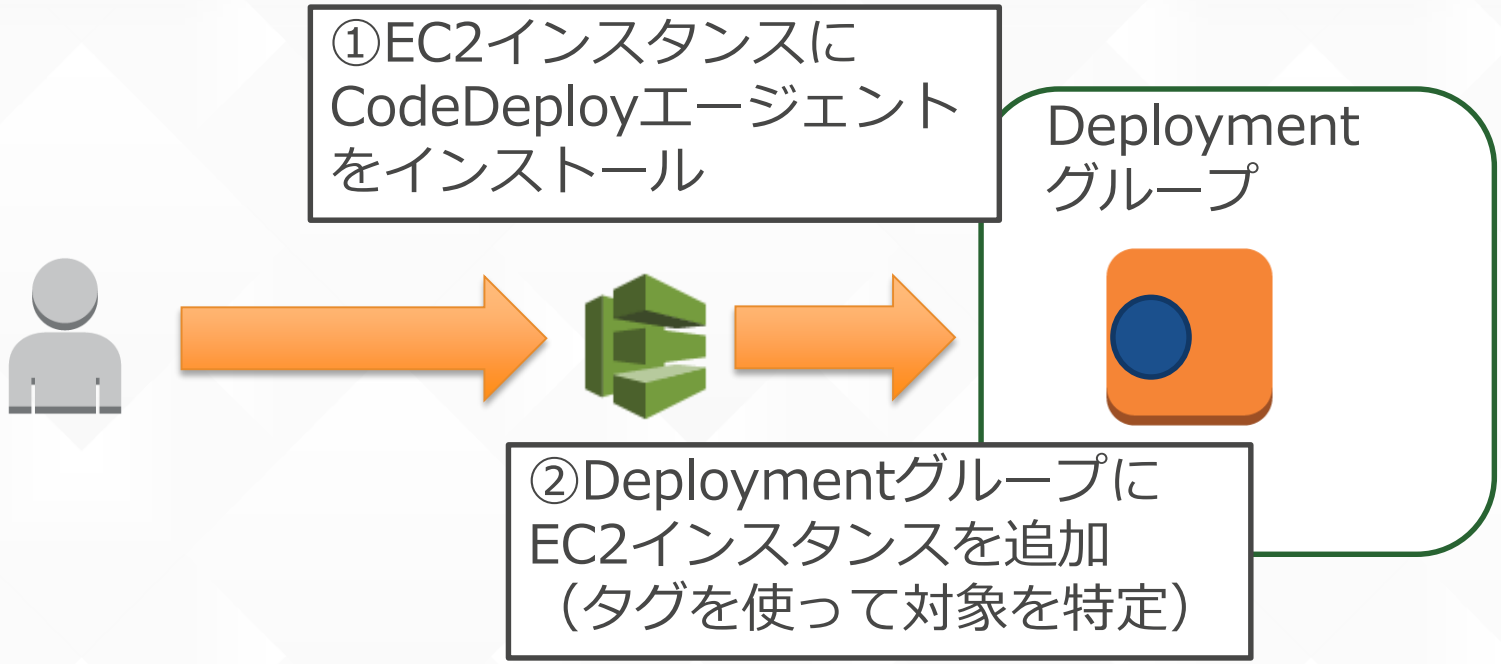
AWS CodeDeploy

amazon web services

AWSから提供されているデプロイ＆マネージメント関連サービス



CodeDeployによるデプロイのための事前準備



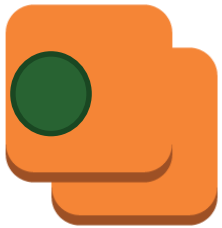
CodeDeployによるデプロイ

デプロイ

②Deployment Groupにデプロイ

③CodeDeployエージェントがpolling

デプロイ担当者



開発者

④GitHubからソースコードバンドルを取得してデプロイ

①WordPressにAppSpecファイルを含める

amazon web services

Deployment Configuration

どれを
デプロイするか？

Revision 1

Revision 2

Revision 3

どうやって
デプロイするか？

Deployment
Configuration

OneAtATime

AllAtOnce

HalfAtATime

どこに
デプロイするか？

Deployment Group

appspec.ymlのhooks

環境のカスタマイズ

```
hooks:
  BeforeInstall:
    - location: scripts/install_dependencies.sh
      xxxxx
  AfterInstall:
    - location: scripts/change_permissions.sh
      xxxx
  ApplicationStart:
    - location: scripts/start_server.sh
      xxxx
  ApplicationStop:
    - location: scripts/stop_server.sh
      xxxx
```

シンプルに既存の
スクリプトを利用可能

amazon web services

AWS CloudFormation + AWS CodeDeploy

- CloudFormationを使って、CodeDeployエージェントがインストールされたEC2インスタンスをプロビジョニング

プロビジョニング

継続的なデプロイ

AWS CloudFormation

AWS CodeDeploy

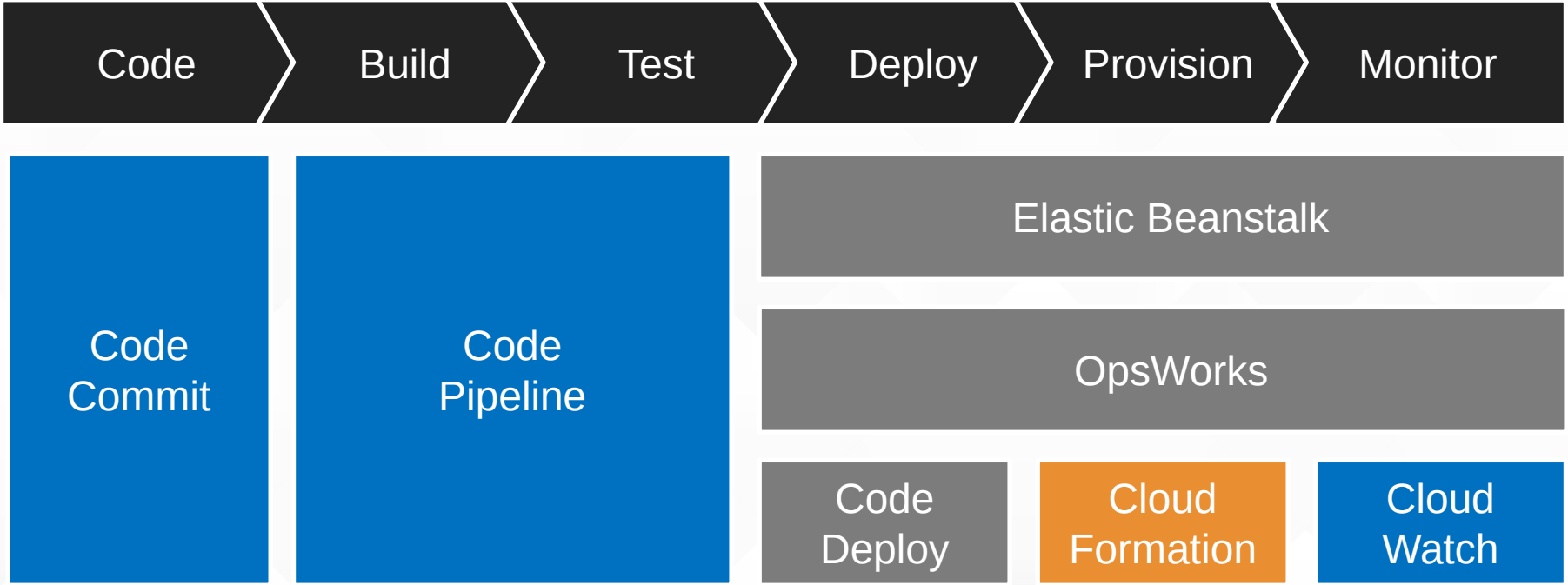
※CloudFormationを使用しなくてもCodeDeployをご利用可能
※CloudFormationではCodeDeployのDeployment Group等のリソースは未サポート（2015/6/2現在）

amazon web services

AWS CloudFormation

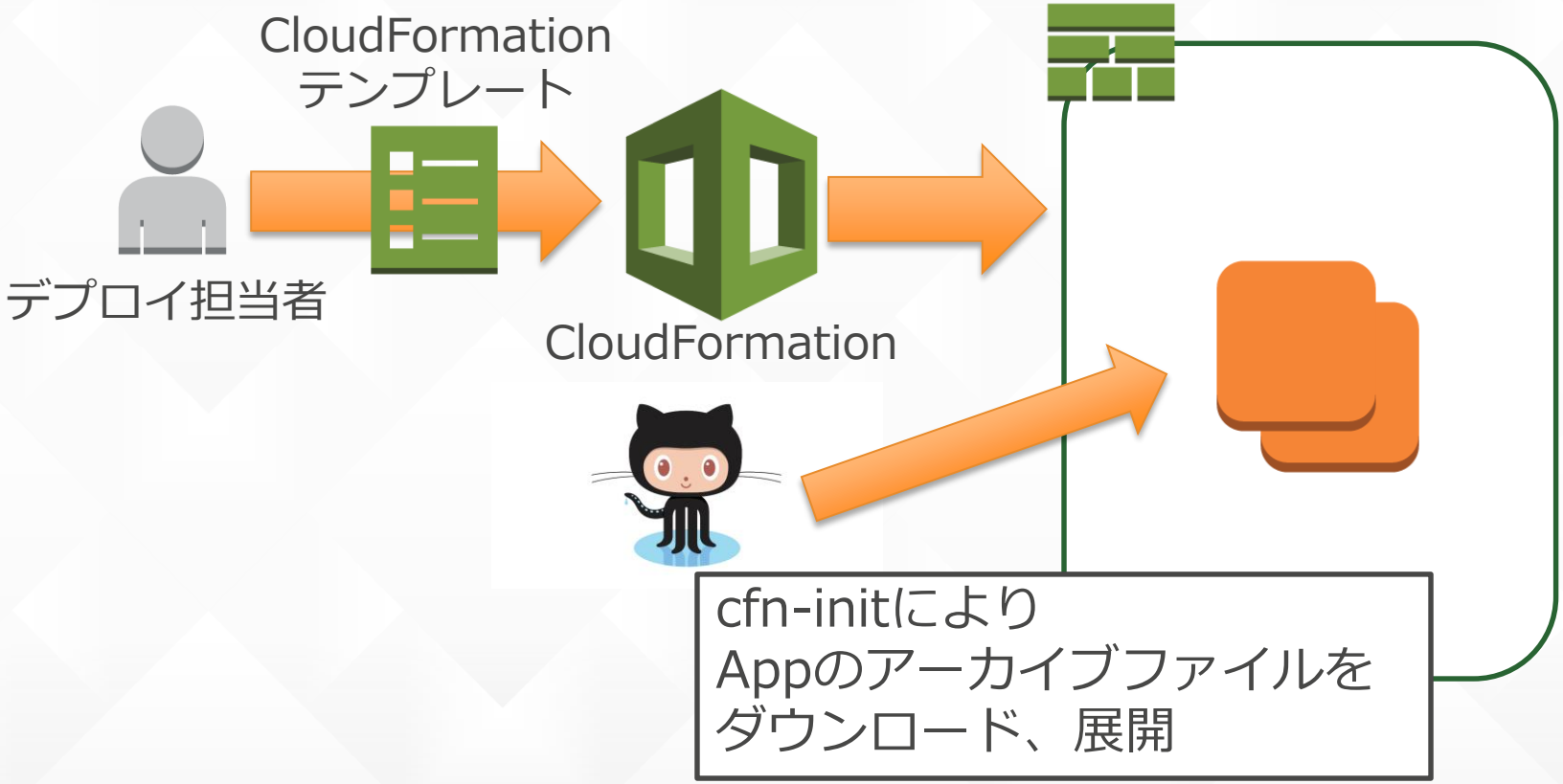
amazon web services

AWSから提供されているデプロイ＆マネージメント関連サービス



AWS CloudFormationによるプロビジョニングおよびデプロイ

プロビジョニング&デプロイ



AWS CloudFormation Template

```
{
   "AWSTemplateFormatVersion" : “2010-09-09",

   “Parameters" : {
   },

   "Resources" : {
   },

   "Outputs" : {
   }
}
```

CloudFormation実行時に
ユーザ入力を求める
パラメータを宣言

CloudFormationで作成する
AWSリソースを宣言

amazon web services

AWS CloudFormation Template

```
"Resources" : {
   "WebServer": {
    "Type" : "AWS::EC2::Instance",
    "Metadata" : {
     "AWS::CloudFormation::Init" : {
      "configSets" : {
       "wp_install" : ["install_wp", "configure_wp" ]
      },
      "install_wp" : {
      },
       “configure_wp" : {
      }
        . . . .
```

AWS CloudFormation Template

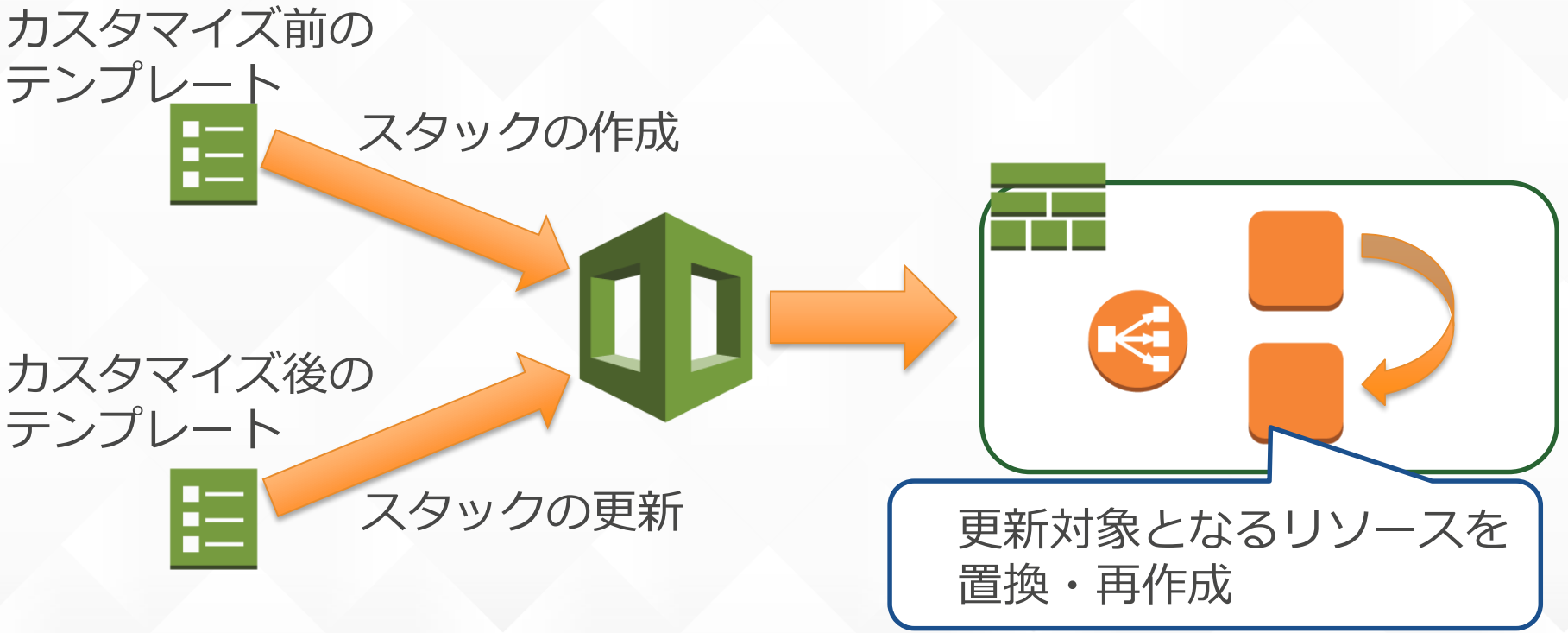
```
"install_wp" : {
    "packages" : {
    },

    "sources" : {
    },

}
```

インストールするパッケージ名

WordPressのコードリポジトリ

amazon web services

環境のカスタマイズ



各種サービスの機能比較

	Elastic Beanstalk	OpsWorks	CodeDeploy	Cloud Formation
プロビジョニング	可能	可能	不可 既存のEC2インスタンスを利用可能	可能
デプロイ	Host Managerと連携して実行	OpsWorksエージェントと連携して実行	CodeDeployエージェントと連携して実行	cfn-initと連携して実行
環境のカスタマイズ	Configuration Fileを活用	カスタムレシピを活用	appspec.ymlを活用	Templateを活用

まとめ

まとめ

- デプロイ＆マネージメントサービスを活用することで、簡単にデプロイ自動化が可能
- それぞれのサービスでデプロイ手順は異なる
- お客様のシステムの要件と照らし合わせて最適なサービスを活用すべき



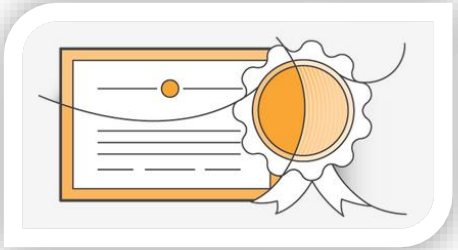

AWSトレーニング @ AWS Summit Tokyo

セルフペースラボ：@パミール1F 瑞光

AWS クラウドに実際に触れてみませんか？
ご自分の AWS アカウントをおつくりいただけなくても、
AWS クラウドを体験いただけます。

AWS認定試験（有償）：@ パミール1F 黄玉

特設認定試験会場を AWS Summit Tokyo 2015 会場に開設
Devopsエンジニア-プロフェッショナル認定試験を先行受験いただけます。



AWS認定資格者取得専用ラウンジ：@ パミール1F 青玉

他の AWS 認定資格をお持ちの方とのネットワーキングにぜひラウンジをご活用ください。
お席や充電器、お飲物などを用意し、皆様をお待ちしております。

amazon web services

Thank You

amazon web services