研究開発部門における
クラウド活用効率化の
取り組み事例

テクノロジー・イノベーションセンター 主任技師
前川博志

はじめに:
ダイキン工業について

企業概要

（2023年3月末現在）

会社名	ダイキン工業株式会社 1963年（昭和38年）大阪金属工業株式会社から社名変更
創　業	1924年（大正13年）10月25日大阪市で創業 創業者：山田晁
設　立	1934年（昭和9年）2月11日
資本金	850億円
グループ従業員数	連結96,337名
会長・社長	会長：井上礼之　社長兼CEO：十河政則
本　社	大阪市北区
グループ会社数	連結子会社347社（国内30社、海外317社）

当社をあらわすキーワード

創業1924年
99年の歴史

人を基軸に
おく経営

空調機器と冷媒を
両方手がけている
総合空調メーカー

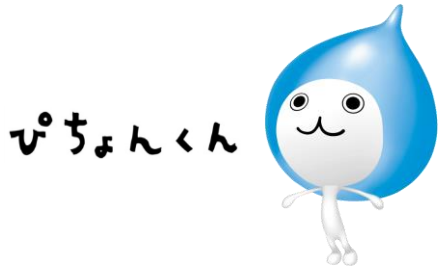
グローバル生産拠点
110カ所以上

全社売上高
3兆円以上

従業員**9.6万人**
海外従業員比率は8割以上

170カ国以上へ
事業展開

海外売上高比率
83%



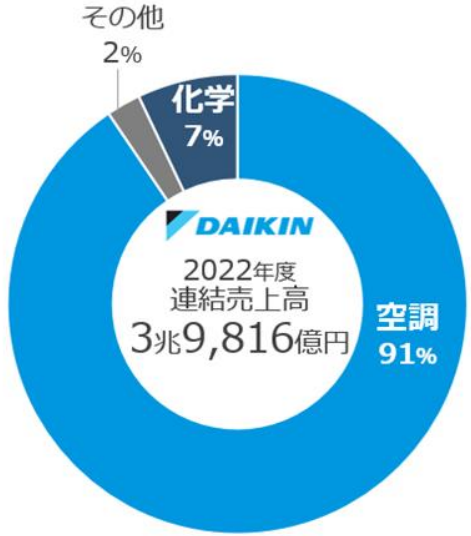
事業内容

空調事業

住宅用

業務用

サービス



その他事業

油圧機器

酸素濃縮機

化学事業

冷媒

半導体用途

自動車用途

空調事業の主なグローバル生産拠点ー市場最寄り化ー

市場最寄化生産をベースに、世界27カ国87ヶ所以上※に生産拠点を構築

※フィルタ、低温を含む

欧州

- **ダイキンヨーロッパ（ベルギー：1972）**
  業務用エアコン、暖房製品
- **ダイキンインダストリーズチェコ（2003）**
  住宅用エアコン
- **ダイキンアプライドヨーロッパ（伊：2007買収）**
  チラー、ターボ冷凍機
- **ダイキントルコ（2011）**
  住宅用エアコン、暖房製品

日本

- **滋賀製作所（滋賀県草津市：1970）**
  住宅用エアコン
- **堺製作所（大阪府堺市：1937）**
  業務用エアコン

堺製作所臨海工場 新1号工場（2018）

米国

- **ダイキンアプライド・アメリカズ（ヴァージニア州スタントン：2007買収）**
  大型チラー、ターボ冷凍機
- **ダイキンコンフォートテクノロジーズノースアメリカ（テキサス州ヒューストン：2012買収）※旧グッドマン**
  住宅用ユニタリ、ガスファーネス、業務用エアコン

ダイキン・テキサス・テクノロジーパーク（2017）

インド

- **ダイキンエアコンディショニングインド（2009）**
  住宅用エアコン、業務用エアコン、水冷チラー、空冷チラー

中国

- **大金空調（上海）有限公司（1995）**
  業務用エアコン、全熱交換器、空冷チラー
- **大金空調（蘇州）有限公司（2011）**
  住宅用エアコン、業務用エアコン
- **マッケイ（武漢 2007買収）**
  水冷チラー、ターボ冷凍機など
- **マッケイ（深セン 2007買収）**
  空冷チラー、ファンコイルユニットなど

アジア

- **ダイキンインダストリーズタイランド（1990）**
  住宅用エアコン、業務用エアコン
- **ダイキンマレーシア（2007買収）**
  住宅用エアコン、業務用エアコン、チラー
- **ダイキンエアコンディショニングベトナム（2018）**
  住宅用エアコン

南米

- **ダイキンエアコンディショニングアマゾナス（2012）**
  住宅用エアコン、業務用エアコン

社内外との協創を通じたイノベーションの加速

空調、化学等のコア技術を追求するとともに、自前主義を脱却し
社内外の異分野技術を取り入れた“オープン・イノベーション”の創出をめざす

技術開発拠点　テクノロジー・イノベーションセンター（TIC）

- 総床面積：約5.8万㎡、6 階建て
- 所在地：大阪府摂津市（当社淀川製作所内）
- 投資額：約380億円
- 開所：2015 年11 月25日
- 人員数：700人規模

産学・研究機関と新たな空気価値創造の取り組み

東京大学

「空気の価値化」を実現する未来技術やビジネスモデルを東京大学の教員陣3,000人や東大発のベンチャー企業群とともに創出し社会実装する

大阪大学

「ダイキン情報技術大学」を設立、AI・IoT人材を大阪大学教員と共に育成。また、新キャンパス（箕面）を舞台に次世代スマートビルの実現もめざす

京都大学

「ニューノーマル時代を見据えた研究開発テーマ」を掲げ、ヘルスケアやエネルギーからアジア・アフリカ地域研究まで幅広い領域で成果創出に取り組む

他社との主な取り組み

日立製作所

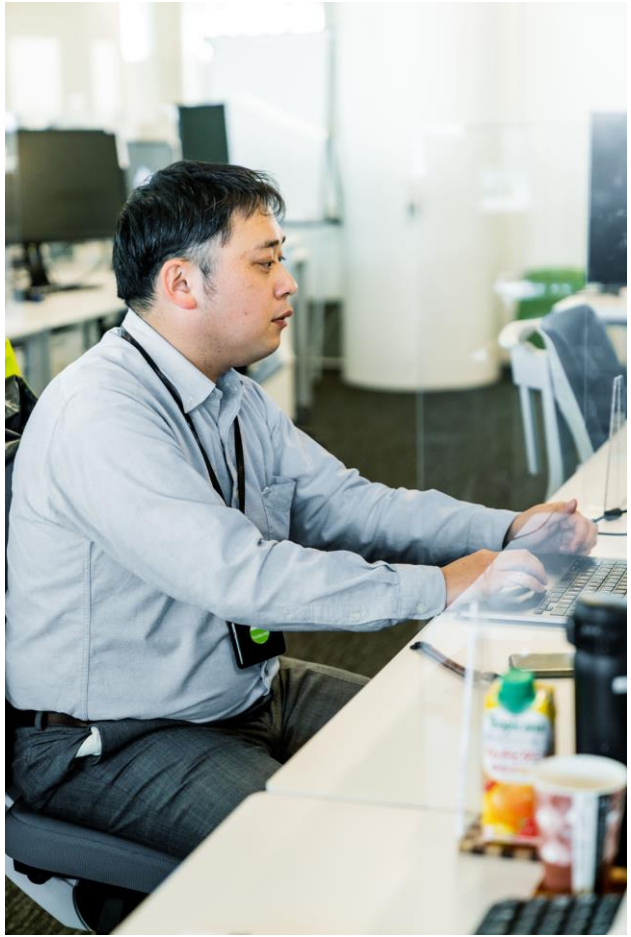
ものづくりに必要な「技能伝承」にAIを活用。品質の安定や生産性向上、人材育成を推進

Point 0

異業種20社以上で知的生産性向上を実現する「未来のオフィス空間」づくりをめざした『CRESNECT』プロジェクトの一環。パナソニック、オカムラ、ライオン、TOTOなどの社員が中心となり、IoTを駆使してサービスを開発

私について簡単に

自己紹介



前川 博志

分析機器メーカー@京都
⇒ IT系ベンチャー@東京
⇒ ダイキン工業

空調IoTプラットフォームDK-Connect開発のSRE
⇒ サービスマンサポートシステム全体のアーキテクト

⇒ AWSベストプラクティス/リファレンス作成
+ 関連プロジェクトのアジャイルコーチ

実現したいこと

研究開発組織における現在の課題

背景

- 5カ年計画でも大きなテーマとして、「顧客とつながるソリューション事業の推進」が示される
  ⇒ **ソリューション企画の推進が必要**
- スピード感をもってソリューション構築するため、
  AWSを始めとしたクラウドの活用が重要

現在の課題

- ソリューション製品に対する**標準的な設計が存在しない**ため、アーキテクチャが安定しない
  ⇒ 同様の課題に対して車輪の再発明多発、過去トラの共有も不十分で同様の課題を再生産
- 結果的に、**開発期間長期化、品質低下、ベンダーへの過度な依存**といった問題が発生

ダイキン内で活用できる「設計の標準化」が必要

第一の矢

AWS ベストプラクティス

AWSベストプラクティス

- オージス総研 / AWS などの**有識者協力の下、TICにおける標準的なAWS実装を定め、ベストプラクティスとして策定**。アプリケーションを効率的に構築するポイントから、セキュリティ対応、監査対応なども含めて記述

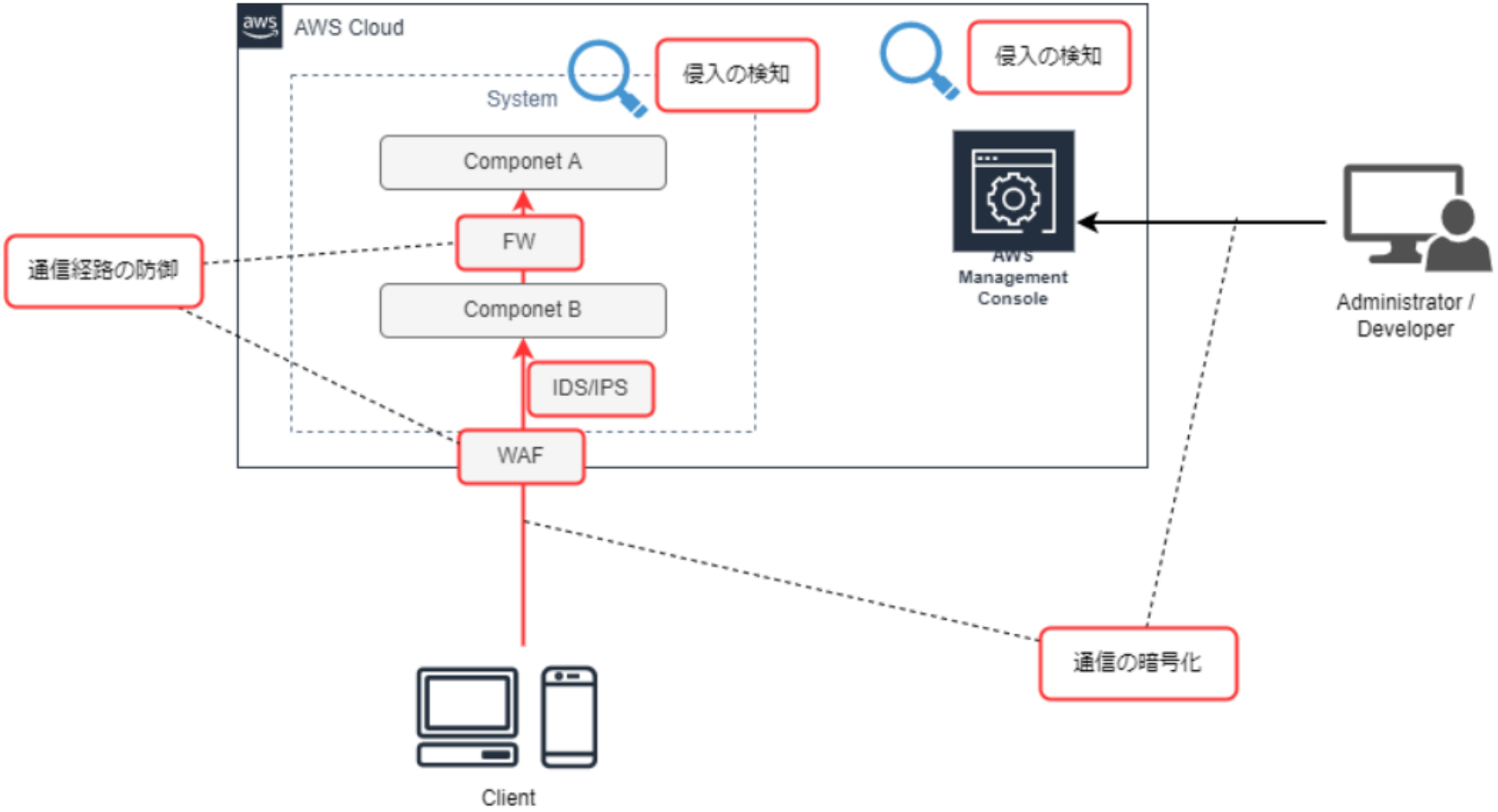
目次

- アドバイスは受けながらも、文書は**自分ごととできるよう基本的に社員自らが執筆**。
- **Githubを活用**したWebベースのドキュメントとすることで、更新作業やレビュー作業を効率化し、最新情報への更新を継続的に実施

ベストプラクティス例① 計算リソースの選択

自由度	高い ⟷ 低い		
管理工数	多い ⟷ 少ない		
スケールの単位	インスタンス	アプリケーション(コンテナ)	関数
AWSの責任範囲	ハードウェア以下	OS以下	ランタイム以下
特徴	・OS、ネットワーク、ストレージのレベルで構成を制御できる ・好みのOSを利用できる ・OS以上の全てを自分でコントロールできる	・サーバを自分で構成して実行できる ・アプリケーションの構成を制御できる ・スケールを自分でコントールできる	・必要なときだけコードを実行できる ・インフラの構成・管理を行う必要がない

ベストプラクティス例② セキュリティの考え方



その他ベストプラクティスのポイント

- **各章の冒頭に、概要と最低限気を付けるべきポイントを明示**
  - 対象読者
  - 要約
  - 得られる知識
  - 実装上のポイント

- **一般論にとどまらず、ダイキン社内規定を参照したプラクティスも存在**

重要度	ダイキンセキュリティガイドライン上の定義	本ベストプラクティスでの定義
レベル1	• 社外に開示されることにより当社が極めて重大な損失又は不利益を受ける・受ける恐れのあるもの • 漏洩させないことを優先し、厳選し、かつ厳格に管理する	• データが1つでも持ち出された場合、当社に大きな損失を与えるもの ◦ M&Aなどの高度な企業活動上の秘密、コア技術に関する文書、工場内の生産管理情報、社内での限定的な利用を条件に提供を受けた営業秘密など

- **定期的なアップデートを実施**

ドキュメントをコードとして管理する試み

- AsciiDocとGithubを活用し、ドキュメントをコードのように（テキストベースで）管理している

- コードとして管理し自動化することのメリットは大きいと考えている
  - textlintなどの日本語に対する静的チェック
  - PRを利用したドキュメントのレビュー
  - Web/PDFなどの複数の出力形式の管理
  - 多言語対応 など…

- レビュー資料などの公式ドキュメントなどにもこのような取り組みを広げていきたい
  - レビュー議事録の自動作成
  - 生成AIを活用した（半）自動レビュー

Github上でのドキュメントのレビュー

doc/reference_architecture/iot/index.adoc Outdated

```
77 + **(1) メッセージのバッファリング** +
78 + Lambdaでのデータ処理の前にKinesisを経由することで、メッセージブローカで受け付けたメッセージをバッファリン
     グできるため、La
     処理できる。 +
79 + Kinesisを経由せて
     ファリングされて
     メッセージデータ
80 + Kinesisを経由し
     欠損が許容される
```

[redacted] Mar 30, 2022

[IMO] こちら（後段の記載も
ッファして複数レコードを後
量データの処理を効率的に行
多くないなどの場合に」など
」

例えば、
・Lambdaに複数レコードま
BatchWriteItemなどの複数レ
ーバーヘッドが高い）
・OpenSearchに渡すような
OpenSearchの負荷を減らす

1

doc/iot/index.adoc Outdated

```
27 + また、``AWS IoT Core`` 自体も複数のサービスによって成り立っており、クラウド上でIoTサービスを構築するにあた
     り、それぞれ基本的なサービスを提供する。
28 +
29 + .AWS IoT Core概要
30 + image::https://docs.aws.amazon.com/ja_jp/iot/latest/developerguide/images/aws_iot_data_services.png[]
```

dk-maekawahir on Mar 1,

[IMO]これ以上画像に要素
いう印象。この画像のどの

dk-yanagiko on Mar 4, 202

上記の画像にそれぞれの
た

Reply...

Resolve conversation

doc/external_security/index.adoc Outdated

```
15 +
16 + === IDS/IPS
17 +
18 + === プロクシサーバ
```

github-actions bot on Jan 18, 2022

[textlint] <eslint.rules.spellcheck-tech-word> reported by reviewdog

プロクシ => プロキシ (spellcheck-tech-word)

Reply...

Resolve conversation

DAIKIN

第二の矢

AWS リファレンス アーキテクチャ

とはいえ、、、

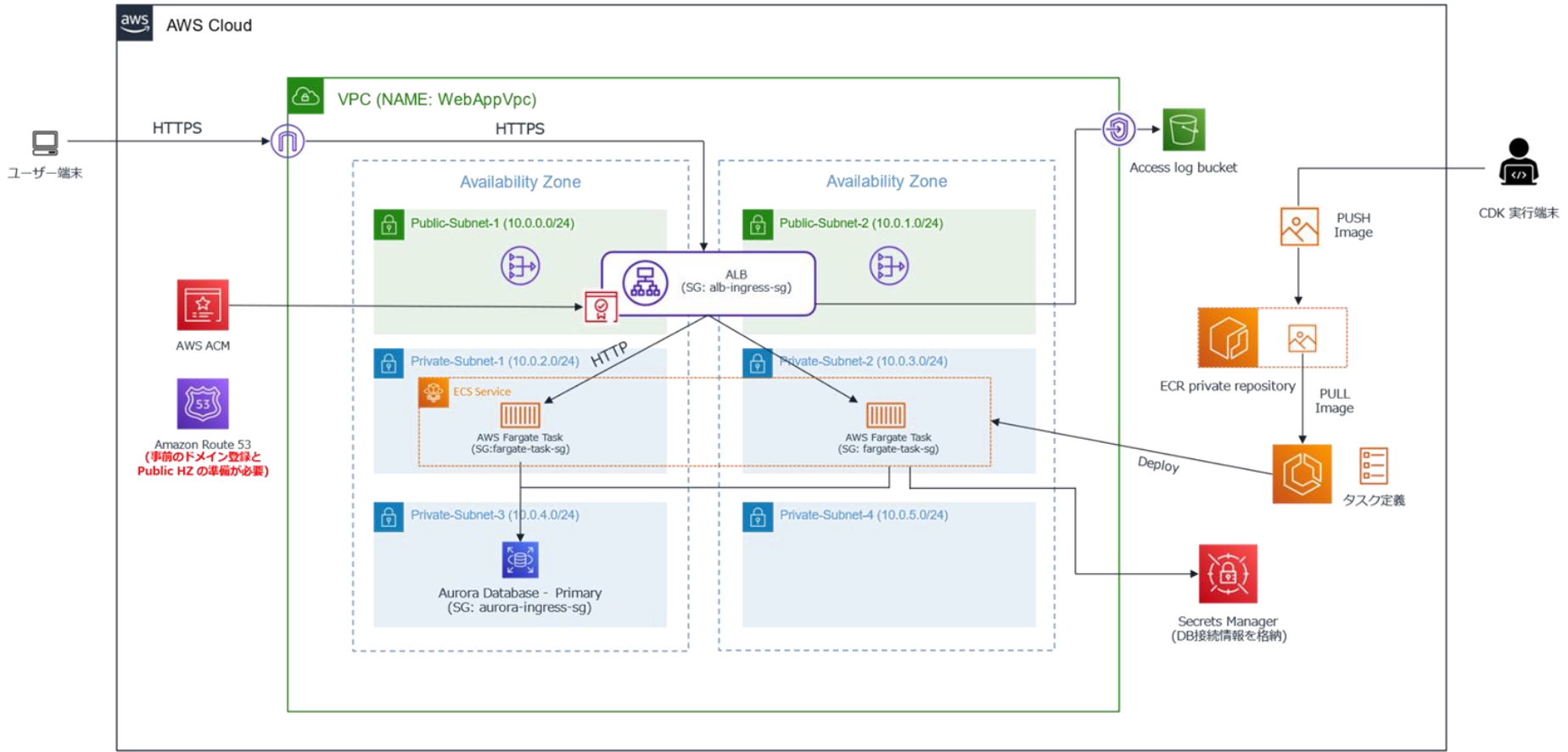
- AWSの知識が一定以上あれば、ベストプラクティスからある程度最適な設計を導き出せるが、**そもそも200P以上のドキュメントを通読することを全技術者に求めるのは無謀**
- ベストプラクティスも正解が一つ書いてあるわけではなく、実装者によってブレは生じる。ブレ自体はプロジェクトに合わせた設計を行う上で必要だが、**どの程度までブレを許容するのかを判断するには、そこそこの経験が必要**

- ベストプラクティスに従った参考実装=**リファレンスアーキテクチャ**を提供することで、さらなる底上げを図る
- リファレンスアーキテクチャでは、設計例だけではなく**その設計をAWS上に再現可能なコードも提供**し（IaC）、実装者によるブレが基礎設計レベルでは極力発生しないようにしている
- 可観測性など新しい概念に対しては、**実際の動作を確認できるワークショップ**も作成し、導入イメージが持てるようにした

Infrastructure as Code : IaC

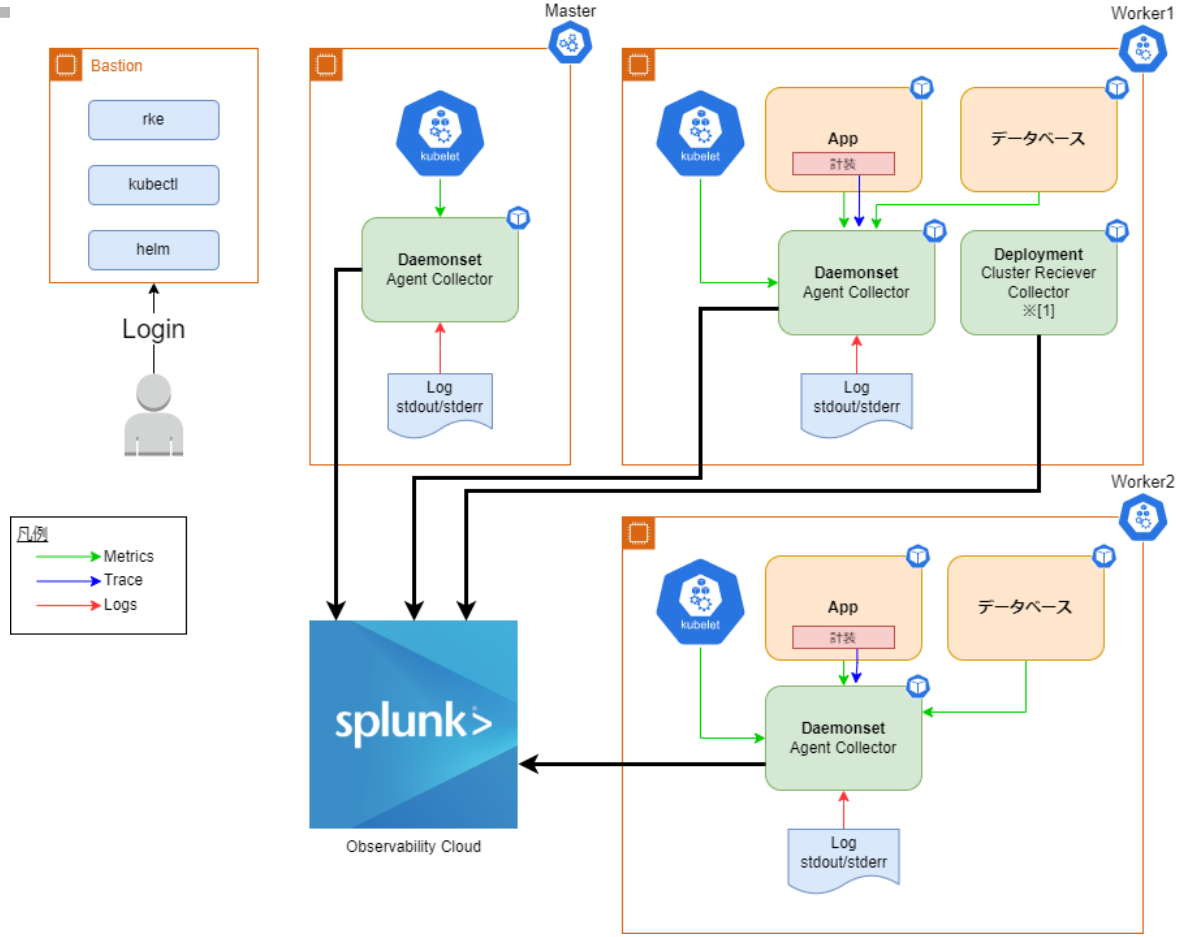
- AWS上のサービスや、サーバ上のリソースなどを対象に、プログラミングコードによってインフラの構成を自動で構築すること
- 重要な特性: 何度実行しても同じ結果となること = **冪等性**
- 作成したいインフラリソースの種別に応じて、様々なツールやサービスが存在する
  - クラウド系
    - AWS Cloudformation / CDK
    - Azure Resource Manager / Azure Automation
    - Google Cloud Deployment manager
  - オープン系
    - Hashicorp Terraform
    - Pulumi
  - サーバ内リソース系
    - Ansible / Chef / Puppet
- （2023年になって、Infrastructure from Codeという言葉もよく聞くように）

リファレンスアーキテクチャ例 Web3層



リファレンスアーキテクチャ例: 可観測性ワークショップ

- 可観測性が議論になるコンテナベース（Kubernetes）環境を構築
- ワークショップ環境自体もIaC化
- Kubernetesワークショップとしても活用

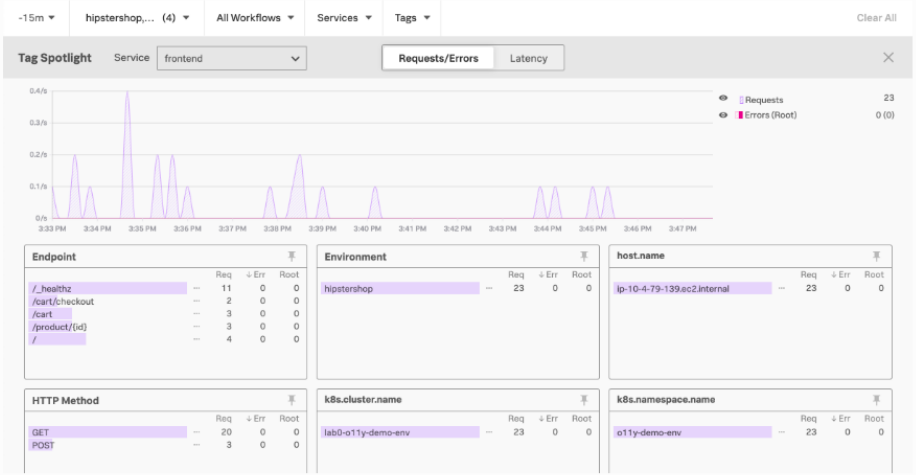


リファレンスアーキテクチャ例: 可観測性ワークショップ

2.5.1.3. サーバリソース等では気づけない問題の洞察

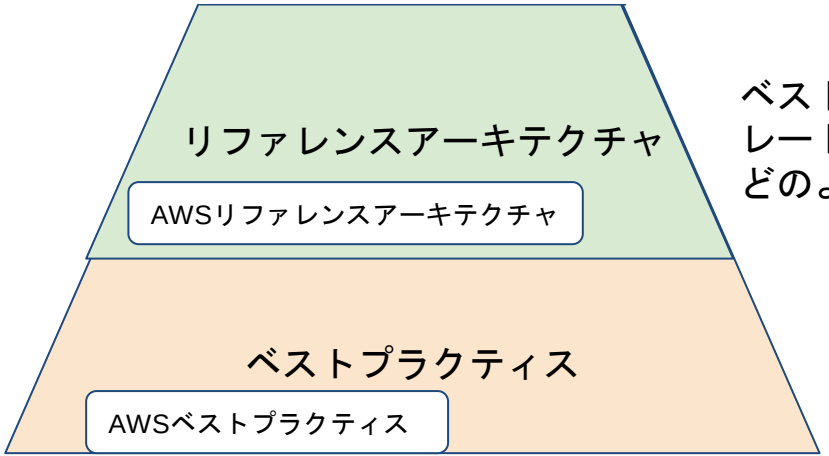
APMでは、より詳細な洞察のために「Tag Spotlight」機能を提供します。
Tag Spotlightでは全てのサービスについて、リクエスト、エラー、経過時間（RED）のメトリクス時系列グラフを提供します。
上記のメトリクスを分析することで、高いレイテンシやエラー率の原因となる傾向を発見することが可能となります。

次の画像は、frontend サービスにおけるリクエストとエラーを表示したものです。



次の画像は、サービスマップの例を示しています。
paymentservice にエラー(赤い点)が発生しており、checkoutservice と paymentservice との依存関係の矢印が赤くなり影響があることを発見します。

この先に目指すもの



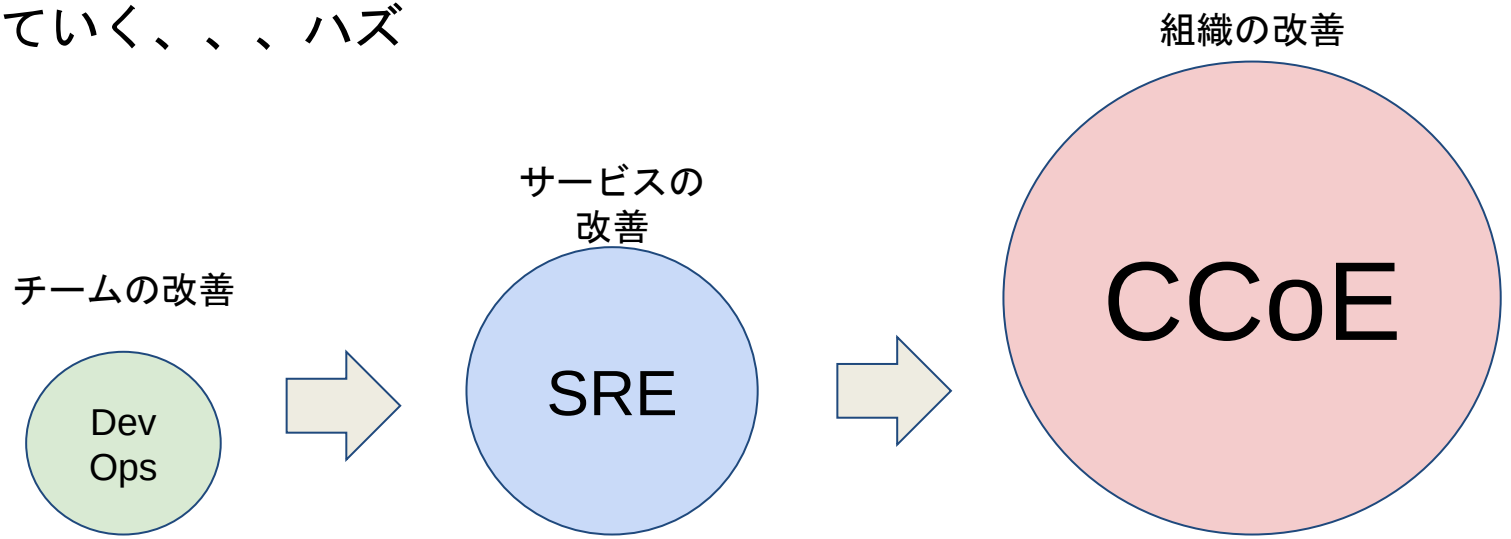
ベストプラクティスをベースに、実装例として明確に定義しテンプレート化したもの。
どのようなアーキテクチャを選択するかの選定は利用者側にある。

あるべきシステム設計をドキュメント化したもの。
自分たちがどういう状態にあるのかを見定め、さらにプラクティスを読み解いて実装する必要がある。

最後に…
～標準化はしたけれど～

使ってもらってナンボ、使いやすくてナンボ、巻き込んでナンボ

- 標準を作ったり、リファレンスを作ったりしてはいるものの、広げていくのは大変だということを実感中
- まず使ってもらうことが大事
- 使ってもらって、フィードバックを受けて改善しづつけることも大事
- 巻き込む人を増やしていくと、それは自然とCCoEという活動につながっていく、、、ハズ



というわけでコミュニティも立ち上げました

社内コミュニティ:

AWS Community for Daikin Co-workers

- 立ち上げ勉強会に80名弱参加
- 今後定期的な小勉強会を起こして組織化していく予定

社外コミュニティ:

CCoE実践者コミュニティ関西

- 関西のCCoE実践企業（ダイキン工業、オージス総研、KINTO Technologies）有志で立ち上げ
- 立ち上げに50名弱参加し、活発に議論

標準化で終わらせないために

- 標準化を「負債」にしない

  「標準化」はともすれば面倒なルールに成り下がってしまう。常に情報鮮度を保ち、標準を更新し続ける動きをしていく

  ⇒ **Githubベースの軽いドキュメント**

- 標準を守れる仕組みを作る

  リファレンスアーキテクチャなど、標準をどう実現・実装するかを分断しない仕組みを作り、広げ、フィードバックを受け改善していく

  ⇒ **リファレンスアーキテクチャをフックとした実プロジェクトの支援**

- 標準はみんなで作っていく

  たたき台としてベストプラクティスやリファレンスは作成しているが、それを絶対とせず、どう運用するかをワイガヤしながら決めていく

  ⇒ **社内・社外のクラウドコミュニティづくり**


DAIKIN