



# 製造業におけるAWSを用いた リファレンスアーキテクチャの構築と 実プロジェクトへの適応

ダイキン工業株式会社  
テクノロジー・イノベーションセンター  
幸浦 祐作

1. はじめに（自己&会社紹介）
2. 研究開発の課題と実現したいこと
3. リファレンスアーキテクチャの取り組みと課題
4. 実プロジェクトへの適応
5. その先にあるもの



大阪在住（妻、3歳の息子）

趣味：

ゴルフ、筋トレ、ポケモン

Twitter（現X）

[@yuuuuuuuu427](https://twitter.com/yuuuuuuuu427)

（ほぼROM専です）

- ✓ 幸浦 祐作（こうら ゆうさく）
- ✓ テクノロジー・イノベーションセンター所属
- 2016年（新卒入社）～2018年
  - 中国向け空調機IoTシステムの設計開発
- 2019年～2021年
  - THINKLETのモバイルアプリ設計開発
- 2022年
  - 人材交流で他社に出向
  - 営業業務を経験
- 2023年～現在
  - リファレンスアーキテクチャ開発

## 空調事業



住宅用



業務用



サービス

その他

2%

化学

7%



2022年度  
連結売上高  
3兆9,816億円

空調  
91%

## 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



# 研究開発の課題と 実現したいこと

# 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



## 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開発を行い、仮説検証サイクルを高速で回すために、AWSを始めとしたクラウドの活用が重要

## 現在の課題

- クラウドソリューション開発の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
  - 特定の開発者への属人化、ベンダーへの過度な依存
  - 類似ソリューションの地域、拠点ごとの性能・品質ブレ
  - トラブルシューティング、技術知見などの共有の難化

### 重点戦略テーマ

成長戦略テーマ3テーマは不変。個別に新たな強化観点を追加。重点テーマ全体では、将来を見据えて、今から取り組むべき強化地域/事業テーマとして、2テーマを追加【赤字が該当】

### 成長戦略

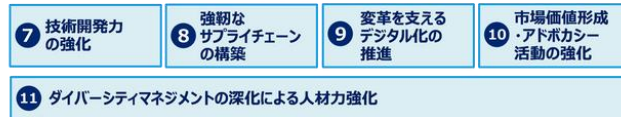


### 当社の成長を支える各事業

### 強化地域/事業



### 経営基盤強化



## 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
- スピード感を仮説検証サイ 始めとしたク

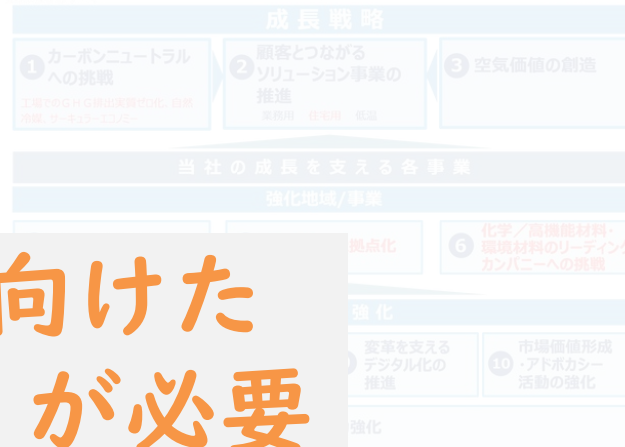
## ダイキン社内に向けた「設計の標準化」が必要

## 現在の課題

- クラウドソリューション開発の 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、過去トラのチーム間共有も不十分で同様の課題を再生産  
⇒ 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

### 重点戦略テーマ

成長戦略テーマ3テーマは不変。個別に新たな強化観点を追加。重点テーマ全体では、将来を見据えて、今から取り組むべき強化地域/事業テーマとして、2テーマを追加【赤字が該当】







# リファレンスアーキテクチャ の取り組みと課題

プラット  
フォーム

本当に必要な業務ロジックだけ記載すれば、デプロイまで実施できるような環境。アプリケーションの構築に専念できる。工数効果は高いが、用途は限定されるので、色々なプラットフォームが必要かも？

共通  
モジュール

アプリケーションの部品として使えるようなモジュール。組み合わせるだけでアプリが形作れる。  
共通化例: SDK的なもの / Okta認証モジュール 等

リファレンス  
アーキテクチャ

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は利用者側にある。

ベストプラクティス

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。

プラット  
フォーム

本当に必要な業務ロジックだけ記載すれば、デプロイまで実施できるような環境。アプリケーションの構築に専念できる。工数効果は高いが、用途は限定されるので、色々なプラットフォームが必要かも？

共通  
モジュール

アプリケーションの部品として使えるようなモジュール。組み合わせるだけでアプリが形作れる。  
共通化例: SDK的なもの / Okta認証モジュール 等

リファレンス  
アーキテクチャ

AWSリファレンスアーキテクチャ

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は利用者側にある。

ベストプラクティス

AWSベストプラクティス

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。



第一の矢

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

### 1. 改訂履歴

### 2. はじめに

#### 2.1. 本書の読み進め方

#### 2.2. 本書の凡例

#### 2.3. 問い合わせ

### 3. サービス選択

#### 3.1. 要約

#### 3.2. コンピュートサービス

#### 3.3. データストアサービス

### 4. ネットワーク設計

#### 4.1. 要約

#### 4.2. 前提条件

#### 4.3. VPC内設計

#### 4.4. VPC外部とのネットワーク設計

#### 4.5. 外部システム連携

#### 4.6. その他

### 5. データ保護、管理

#### 5.1. 要約

#### 5.2. セキュリティレベルと保護の手法

#### 5.3. 暗号化

### 6. ログ管理

#### 6.1. 要約

#### 6.2. 設計の概要

#### 6.3. 監査ログの集約

#### 6.4. アーキテクチャごとのログ管理方式

#### 6.5. S3でのログ保管

### 7. 認証、認可

#### 7.1. 要約

#### 7.2. 認証と認可

#### 7.3. AWS Cognito

#### 7.4. AWSにおける認証/認可の構成パターン

#### 7.5. Cognitoを利用する場合のユースケース例

#### 7.6. Cognitoユーザプール認証におけるセキュリティ

### 8. 運用、監視

#### 8.1. 要約

#### 8.2. 監視

#### 8.3. バックアップ

#### 8.4. 構成管理

### 9. 外部接続のセキュリティ

#### 9.1. 要約

#### 9.2. AWSにおけるセキュリティ設計方針

#### 9.3. 通信の暗号化

#### 9.4. 通信経路の防御/検知

#### 9.5. 侵害の検知

### 10. コスト最適化

#### 10.1. 要約

#### 10.2. コスト管理の基本的なポリシー

#### 10.3. 利用するサービス

#### 10.4. コスト最適化のベストプラクティス

# ベストプラクティス例① 計算リソースの選択



EC2



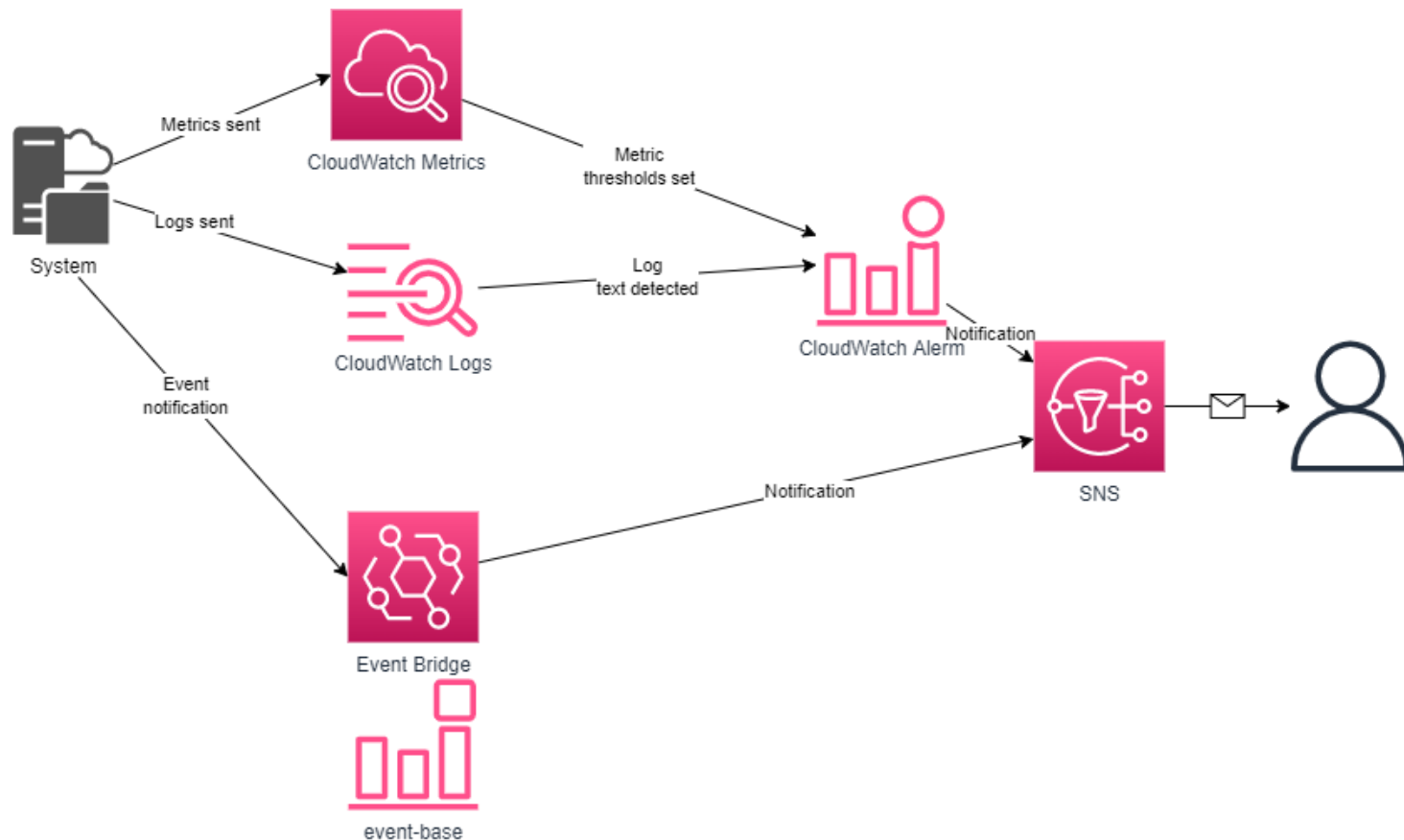
(ECS Fargate)



Lambda

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	<ul style="list-style-type: none"><li>・ 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</li><li>・ 好みのOSを利用できる</li><li>・ OS以上の全てを自分でコントロールできる</li></ul>	<ul style="list-style-type: none"><li>・ サーバを自分で構成して実行できる</li><li>・ アプリケーションの構成を制御できる</li><li>・ スケールを自分でコントロールできる</li></ul>	<ul style="list-style-type: none"><li>・ 必要なときだけコードを実行できる</li><li>・ インフラの構成・管理を行う必要がない</li></ul>

# ベストプラクティス例② ログ監視アーキテクチャ



- 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
  - 更新作業やレビュー作業を効率化、最新情報への更新を継続的に実施
- コードとして管理し自動化することのメリット
  - 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…
- 今後他プロジェクトや議事録などのドキュメントに横展開



# 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	<ul style="list-style-type: none"><li>● 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</li><li>● 漏洩させないことを優先し、厳選し、かつ厳格に管理する</li></ul>	<ul style="list-style-type: none"><li>● 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</li><li>○ M&amp;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</li></ul>

- AWS社と共同で定期的なアップデートを実施



第二の矢

AWSリファレンス

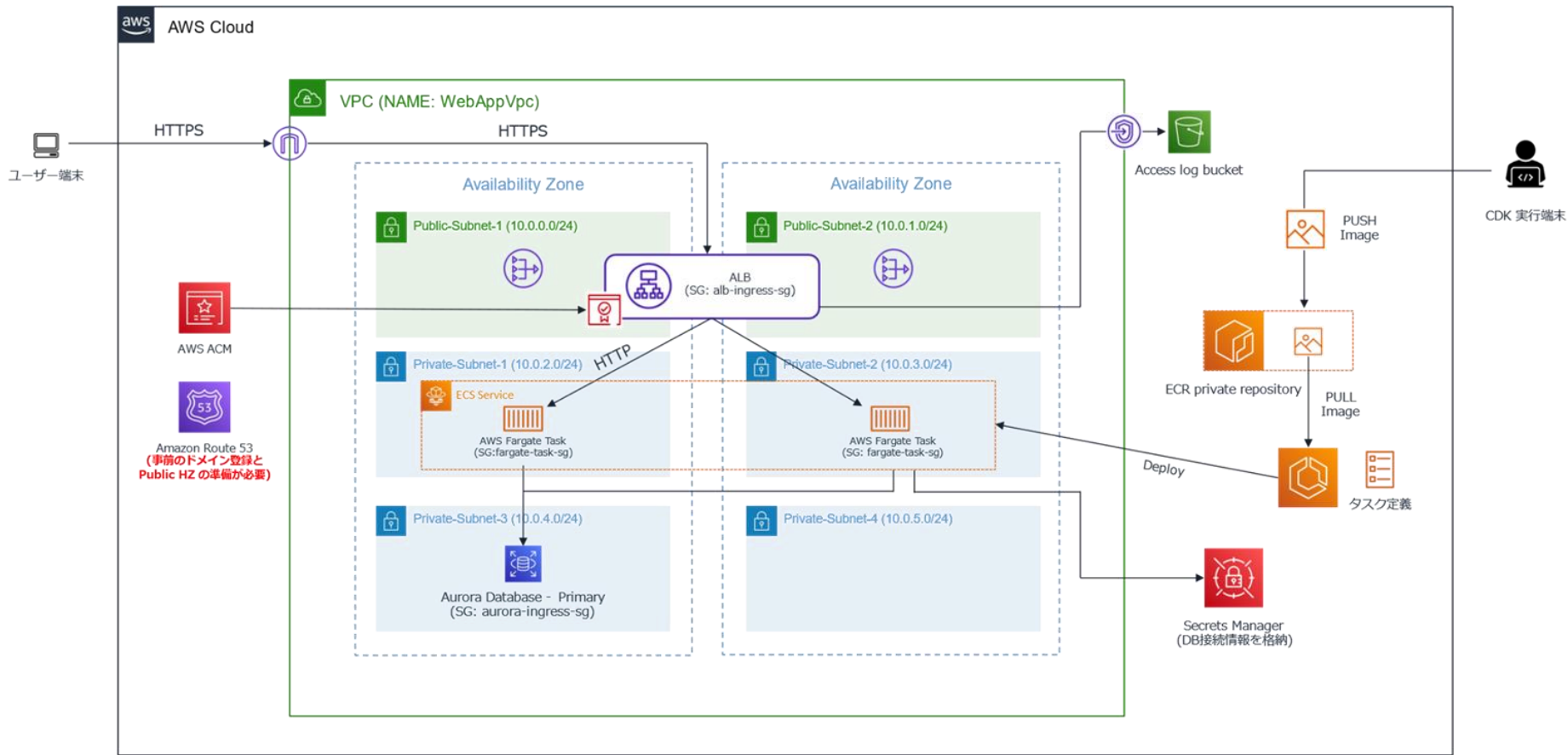
アーキテクチャ

- 内容の質は高いが、初学者には難解なところも多々あり、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
- 正解が一つ書いてあるわけではなく、プロジェクトの特色やシステムの要件に応じてカスタマイズが必要であり、そのためには、最低限の技術力と経験が必要

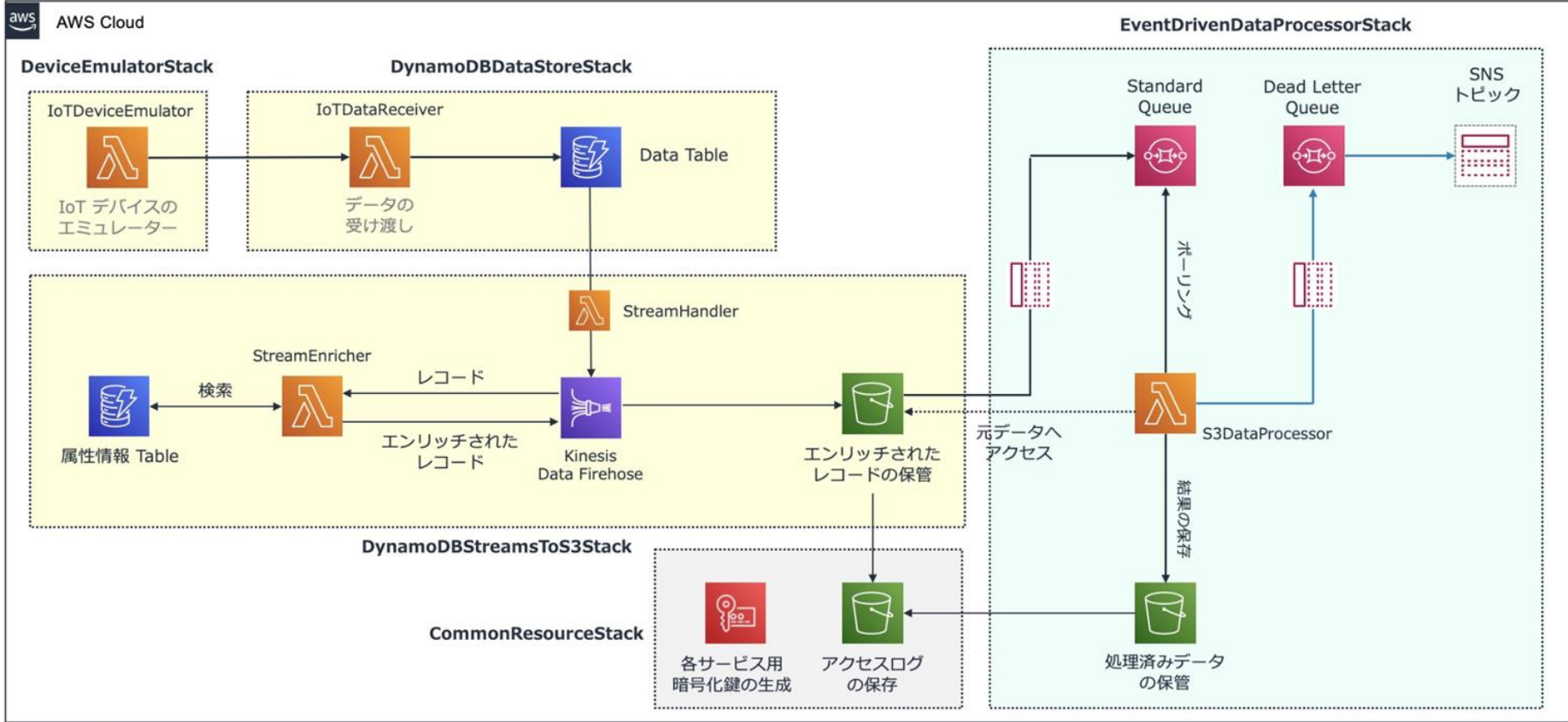


- ベストプラクティスに従った参考実装=リファレンスアーキテクチャを提供し、実装の簡易化と理解の底上げを図る
- 設計例に加えてその設計をAWS上に再現可能なコード(IaC)も提供し、実装者によるブレが極力発生しないように

# リファレンスアーキテクチャ例①：Web3層



# リファレンスアーキテクチャ例②：データストリーム



凡例：  ユースケース①に対応  ユースケース②に対応  共通リソース



作ったはいいものの、、、、

- 何人か読んでくれる人はいるが、浸透しているとは言えない
- いきなり使ってください！はハードルが高すぎる
- 全体に周知する仕組みがないため存在すら知らない人が多数。

## なかなか使ってもらえない

- ・ 何人か読んでくれる人はいるが、浸透しているとは言えない
- ・ いきなり使ってください！はハードルが高すぎる
- ・ 全体に周知する仕組みがないため存在すら知らない人が多数。

「どうやったらみんなにつかってもらえるだろう」



「直接入り込んで推進していくしかないのか？」



# プロジェクト参画まで

---



別事業部のプロジェクトチームから個別相談依頼が来る



別事業部のプロジェクトチームから個別相談依頼が来る



- ・ 支援内容に作成したリファレンスアーキテクチャがある程度  
適応できそう
- ・ 直接利用者からのフィードバックを得られるチャンス

別事業部のプロジェクトチームから個別相談依頼が来る



- ・ 支援内容に作成したリファレンスアーキテクチャがある程度  
適応できそう
- ・ 直接利用者からのフィードバックを得られるチャンス



当プロジェクトの周知活動の第一歩として支援を実施



実プロジェクトへの適用



# 支援先紹介

## 空調事業



住宅用



業務用



サービス

その他

2%

化学

7%

DAIKIN

2022年度  
連結売上高

3兆9,816億円

空調  
91%

## 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



住宅用



業務用



サービス

その他

2%

化学

7%



2022年度  
連結売上高

3兆9,816億円

空調  
91%

## 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



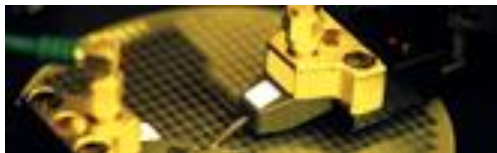
自動車用途

- ・フッ素化学分野で世界シェアNo.2
- ・「フッ素化学製品」の グローバルトップメーカーとして、豊富な製品ラインアップときめ細かなテクニカルサービス体制
- ・冷媒と空調を扱う唯一のメーカーとして、低温暖化係数冷媒の研究開発にも注力

## 【フッ素化学製品が活躍する幅広い分野】



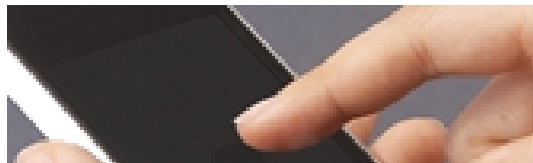
自動車分野



半導体分野



住宅・生活分野



情報通信分野



環境・エネルギー分野



データ収集

処理

可視化・解析



実験装置を使って  
物質の化学反応  
データを取得

生データを解析可能な  
形式に整形・結合

BIツールなどで  
解析・分析



# 支援先での課題

## データ収集

### 格納場所検討



### DB・UI検討



### 転送方法検討



## 処理

### 処理ツール検討

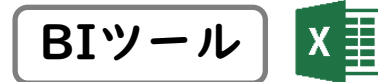


### 実装方法検討



## 可視化・解析

### 可視化ツール検討



### モデル実装方法 検討



### 解析結果の 共有方法検討



現状：チーム毎にツールを検討、個人でコーディング、データ所持している

・ 格納場所検討



・ 処理ツール検討



・ 可視化ツール検討



・ DB・UI検討



・ 実装方法検討



・ モデル実装方法  
検討



・ 転送方法検討



・ 解析結果の  
共有方法検討



現状：チーム毎にツールを検討、個人でコーディング、データ所持している

- 課題：
- ✓ 実装方法が統一されておらず、類似の実装が各個人で実施される  
【車輪の再開発】
  - ✓ データ項目、出力フォーマットが統一されておらず、比較・議論が困難  
【データの分散】

・ 転送方法検討

・ 解析結果の  
共有方法検討

# 現状と課題とあるべき姿

現状：チーム毎にツールを検討、個人でコーディング、データ所持している



- 課題：
- ✓ 実装方法が統一されておらず、類似の実装が各個人で実施される  
【車輪の再開発】
  - ✓ データ項目、出力フォーマットが統一されておらず、比較・議論が困難  
【データの分散】



あるべき姿：

取得したデータに対して統一された適切な処理を実行し、それによって生成されたデータが一意的な場所に保存され、そこに全開発者がアクセスできる



現状：チーム毎にツールを検討、個人でコーディング、データ所持している



- 課題：
- ✓ 実装方法が統一されておらず、類似の実装が各個人で実施される  
【車輪の再開発】
  - ✓ データ項目、出力フォーマットが統一されておらず、比較・議論が困難  
【データの分散】



あるべき姿：

取得したデータに対して統一された適切な処理を実行し、それによって生成されたデータが一意的な場所に保存され、そこに全開発者がアクセスできる

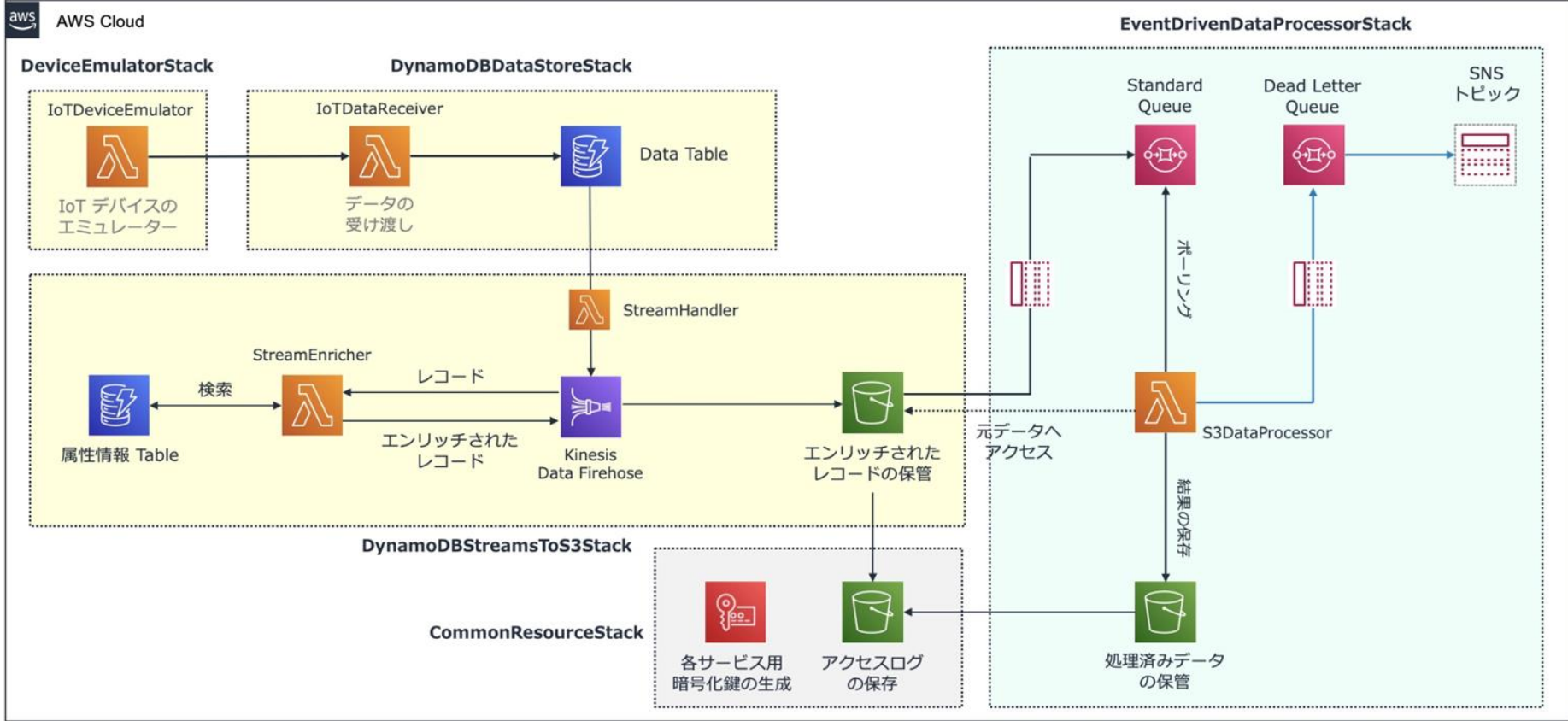
**AWSとGitHubを利用した、データ解析パイプラインを構築**



# 実際に行った支援と問題点

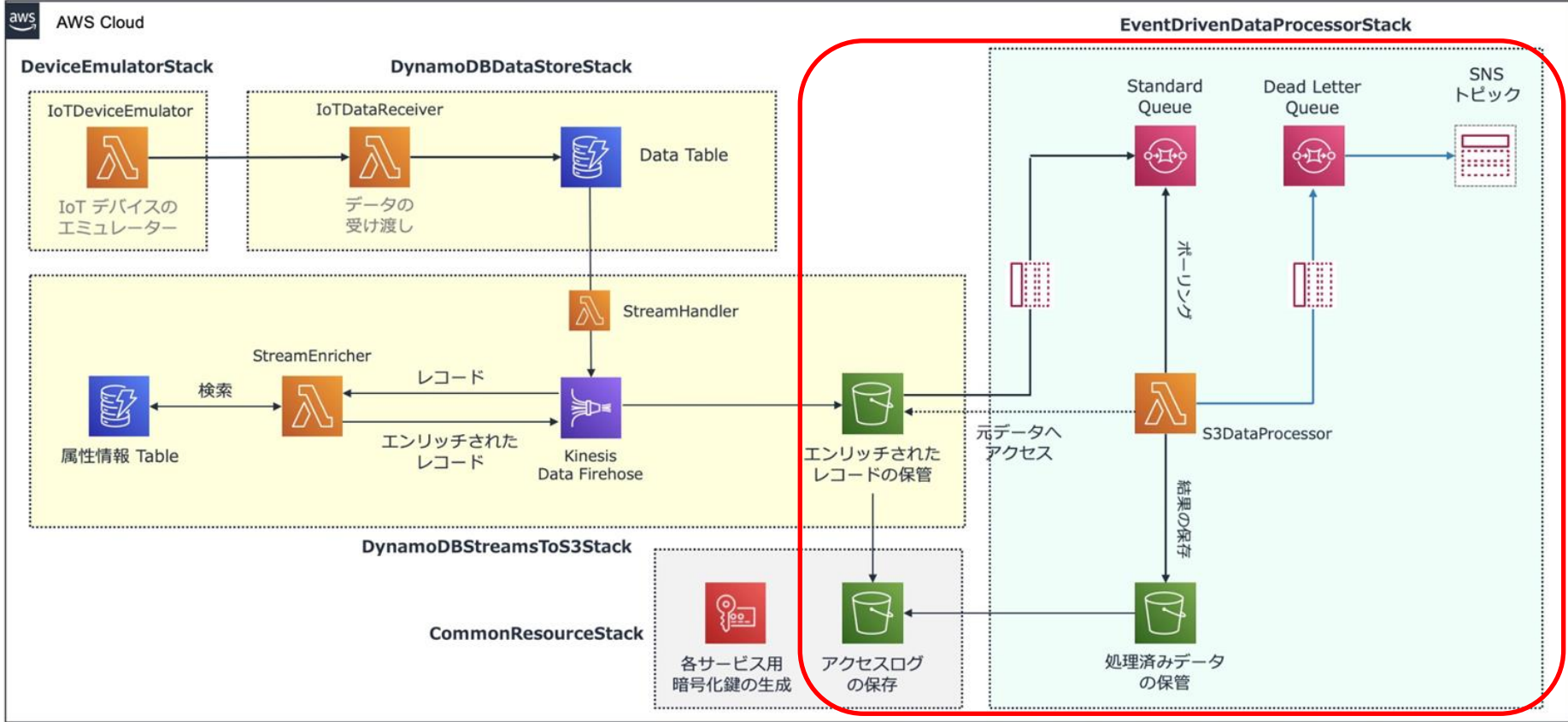


# リファレンスアーキテクチャ例②：データストリーム



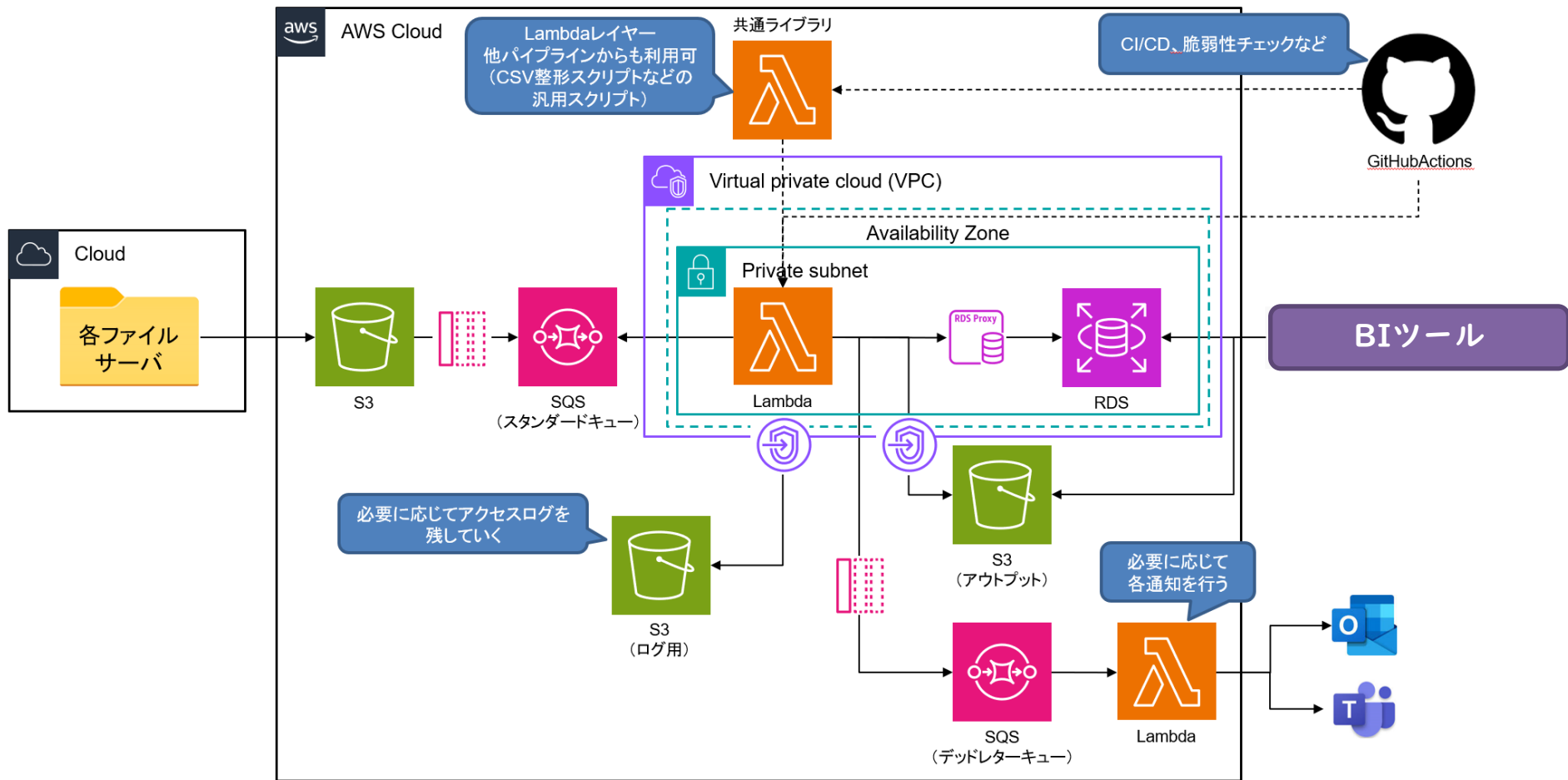
凡例：  ユースケース①に対応  ユースケース②に対応  共通リソース

# 今回適応できそうな範囲



凡例 :   ユースケース①に対応   ユースケース②に対応   共通リソース

# 開発当初のアーキテクチャ





- **ドメインに対する理解不足**
- **セキュリティ要件への対応**
- **利用&運用難易度の容易化**

- **ドメインに対する理解不足**
- セキュリティ要件への対応
- 利用&運用難易度の容易化

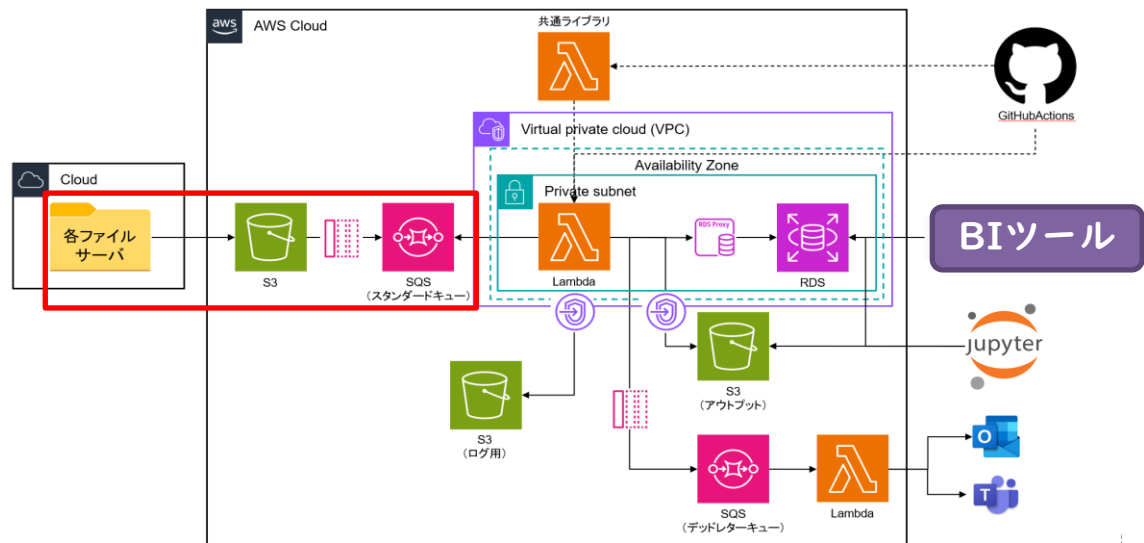
開発者側には化学の知識は全く無く、作成したパイプラインが実際どのように使われるのか想像するのが難しい

➡ 仕様・実装に齟齬が出るケースも

# ドメインに対する理解不足

開発者側には化学の知識は全く無く、作成したパイプラインが実際どのように使われるのか想像するのが難しい

➡ 仕様・実装に齟齬が出るケースも



S3に置かれるデータ1つにつき  
キューを1つ発行する想定



メタデータや実験計画など、  
キューを発行しないデータが  
存在することが発覚



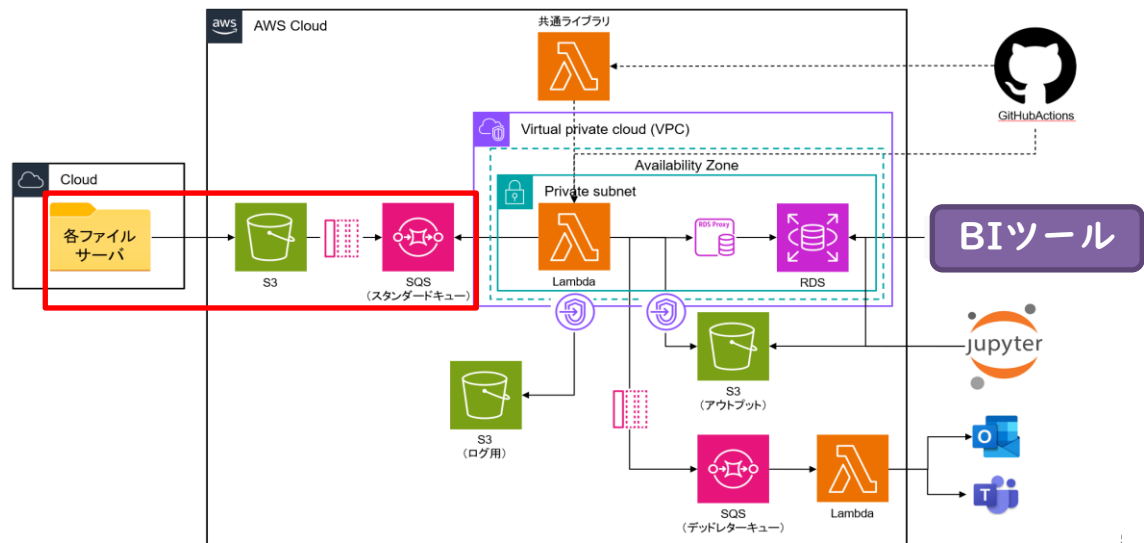
一部再設計が必要となり  
手戻り発生



# ドメインに対する理解不足

開発者側には化学の知識は全く無く、作成したパイプラインが実際どのように使われるのか想像するのが難しい

➡ 仕様・実装に齟齬が出るケースも



S3に置かれるデータ1つにつき  
キューを1つ発行する想定



メタデータや実験計画など、  
キューを発行しないデータが  
存在することが発覚



一部再設計が必要となり  
手戻り発生

ドメインエキスパートと密に連携を取るのは当然だが、  
あとから仕様追加されるケースも多々あるので、「柔軟な設計・実装」を心がける

- ・ ドメインに対する理解不足
- ・ **セキュリティ要件への対応**
- ・ 利用&運用難易度の容易化

実験データには企業名などの機密情報が含まれているケースや、まとまった統計・分析済みデータは機密データ扱いとなる場合がある

➡ クラウド上に無対策で置くことは難しく、管理職への説明・承認が必要

# セキュリティ要件への対応

実験データには企業名などの機密情報が含まれているケースや、まとまった統計・分析済みデータは機密データ扱いとなる場合がある

➡ クラウド上に無対策で置くことは難しく、管理職への説明・承認が必要



要件の熟読・理解



対応可能かどうかの検討  
(難しい場合は代替案提示)



管理職への説明・承認

# セキュリティ要件への対応

実験データには企業名などの機密情報が含まれているケースや、まとまった統計・分析済みデータは機密データ扱いとなる場合がある

➡ クラウド上に無対策で置くことは難しく、管理職への説明・承認が必要



要件の熟読・理解



対応可能かどうかの検討  
(難しい場合は代替案提示)



管理職への説明・承認

**説明作業もリファレンスアーキテクチャを社内に  
浸透させていくために大切な業務**

## 例) RDSのデータに対する大量ダウンロード禁止

- ・クエリの内容を毎回チェックする？
- ・正常動作との区別は？
- ・データ一覧画面のキャプチャは防げる？
- ・何件からが大量ダウンロード？

例) RDSのデータに対する大量ダウンロード禁止

- ・クエリの内容を毎回チェックする？
- ・正常動作との区別は？
- ・データ一覧画面のキャプチャは防げる？
- ・何件からが大量ダウンロード？



**要件をそのまま満たすことは難しい**

例) RDSのデータに対する大量ダウンロード禁止

- ・クエリの内容を毎回チェックする？
- ・正常動作との区別は？
- ・データ一覧画面のキャプチャは防げる？
- ・何件からが大量ダウンロード？



要件をそのまま満たすことは難しい



代替策を検討



## この要件の意図

→ 悪意ある従業員のデータ持ち出しと外部漏洩を防ぐため

## この要件の意図

➡ 悪意ある従業員のデータ持ち出しと外部漏洩を防ぐため

## 代替案



アクセス権をシステム管理者とBIツールに絞り、ユーザが直接データを参照しなくてよいアーキテクチャを構築

- ✓ DB認証情報が格納されているSecretsManagerのアクセスをポリシーでIAMロールを制限、パスワードも定期更新
- ✓ IAMロール操作はTrailで監視、IAMユーザ操作はSCPで制限
- ✓ DBの権限操作(ユーザ追加など)をCloudWatchで監視&通知
- ✓ データ閲覧はBIツールからのみ、そちらでダウンロード制限

- ・ ドメインに対する理解不足
- ・ セキュリティ要件への対応
- ・ **利用 & 運用難易度の容易化**

当然ながら事業部にはITに詳しくない人が多数を占める上に、若手（2、3年目）のIT初心者が利用するケースも多々ある

➡ 開発・テストがやりやすい様々なツール・環境を別途開発

- GitHubActionsを利用したCI/CDパイプライン

➡ デプロイをActionsからに制限することで、環境の無法地帯化防止

- DockerやLocalStackを用いたローカル開発環境

➡ デプロイ前にLambda関数のテスト実施可能にすることで、開発速度向上&手戻り防止

- Lintor + git hookでのコード自動整形

➡ チーム内でのルールを強制的に遵守

- 横展開用の対話型スクリプト

➡ 手順の簡易化によって、他チームへ広げやすく

```
ubuntu@ $ node init_component.mjs
? Select components to add. team
? Enter team name in Camel Case. (example. TeamTestX) TeamTestXYZ
? Enter RDS Database name, This Database must already be created. |
```



# 支援の振り返り

- **ドメインに対する理解不足**
- **セキュリティ要件への対応**
- **利用&運用難易度の容易化**

- **ドメインに対する理解不足**
  - 認識できていなかった新たな仕様や要求
- **セキュリティ要件への対応**
  - 事業部独自の要件を満たすためのリソース
- **利用&運用難易度の容易化**
  - 継続して利用するために必要なツール群

**当初想定のアーキテクチャから数倍に膨張**

### パイプライン開発の観点では…

- いくつか問題はあったものの、適切に対応できた
- 現状問題なく動作している



## パイプライン開発の観点では…

- いくつか問題はあったものの、適切に対応できた
- 現状問題なく動作している

## リファレンスアーキテクチャの観点では…

- × 結果チームの成果物とは別物に
- × 今回の事業部に限った話ではない
- × このまま利用者側に委ねるのは厳しい

パイプライン開発の観点では…

○ いくつか問題はあったものの、適切に対応できた

○ **支援自体は成功と言えるが、  
リファレンスアーキテクチャに関して  
今後の活動の見直し・議論が必要**

× 今回の事業部に限った話ではない

× このまま利用者側に委ねるのは厳しい



この先にあるもの

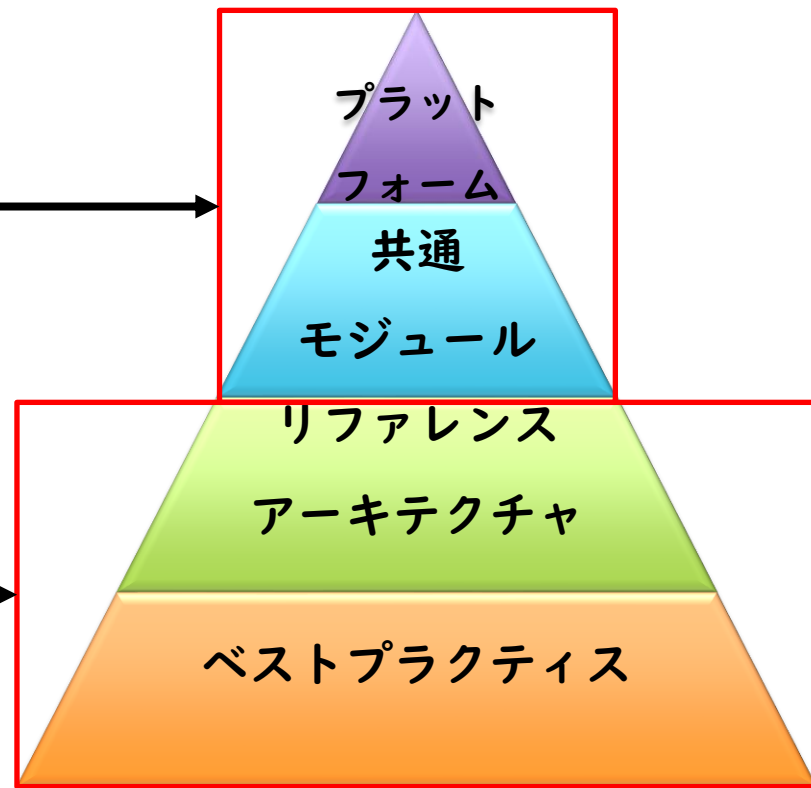
- ✓ やはり、リファレンスアーキテクチャを利用して  
もらうハードルは高い

- ✓ やはり、リファレンスアーキテクチャを利用してもらうハードルは高い

このゾーンを早急に整備していきたい



現在の成果物をベースに支援活動を行いつつ



- ✓ 困っていることに気づいていない部署に対するアプローチ

適応先の優先順位をどうするか？

- ✓ 対応部署（我々）の工数とスキルセット

リソースの振り分け、チーム内教育

- ✓ 既存成果物の運用

① リファレンスアーキテクチャ

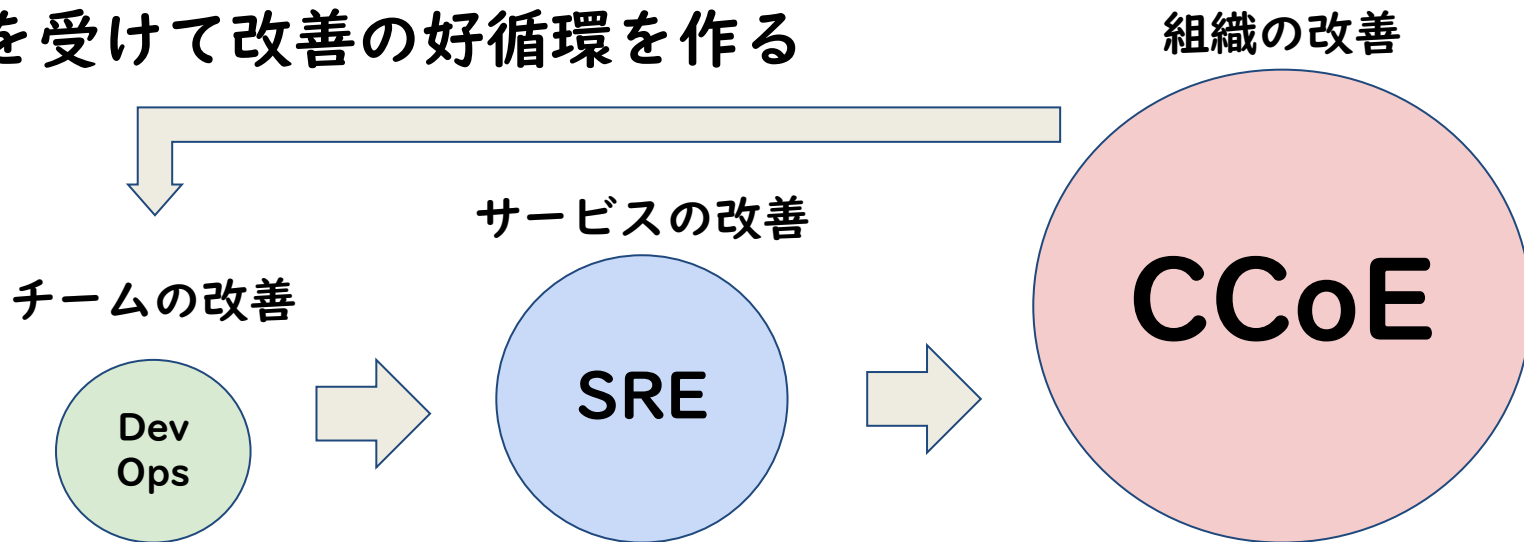
周知・提供方法、ラインナップの拡充・分類

② ベストプラクティス

各アップデート対応、AmazonQなどの生成AIとの棲み分け

## ✓ 作ったものを広げていくのは大変だ

- いきなり全体にばらまくことは難しい
- 地道に知ってもらう・使ってもらうための活動を続けていく
- 徐々に巻き込む人を増やして輪を広げていく→フィードバックを受けて改善の好循環を作る



というわけで社内コミュニティも立ち上げました



# AWS Community for Daikin Co-workers

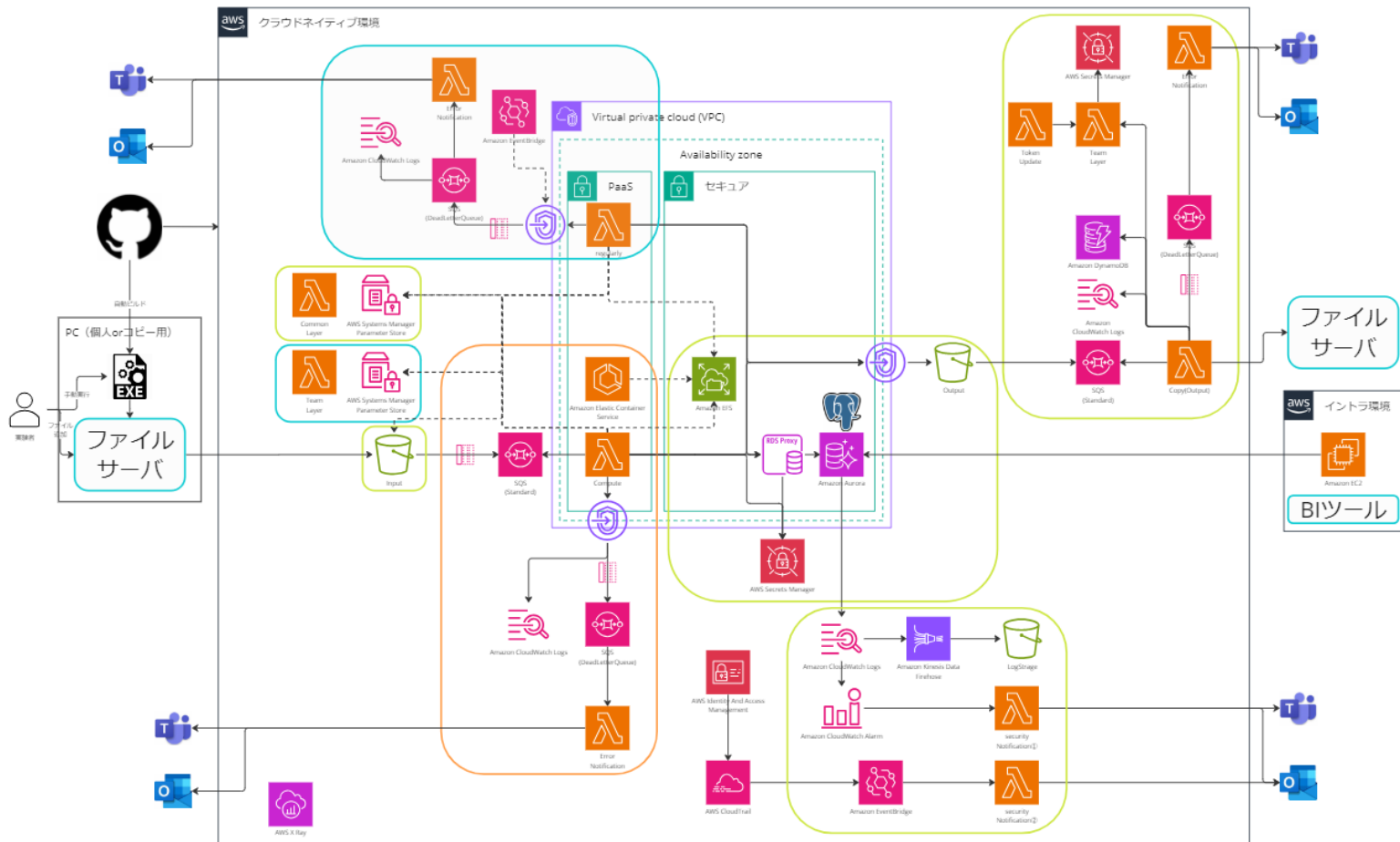


- ・ 立ち上げ勉強会に80名弱参加
- ・ 今後定期的な小勉強会を起こして組織化していく予定
  - 他社SREとの交流会
  - re;Inventの報告会
  - AWS主催のワークショップ





# 最終アーキテクチャ (再掲)



- ✓ ダイキンではAWSを用いたクラウド開発を進めており、高速化/高度化のためにリファレンスアーキテクチャやベストプラクティスの制定・運用を開始している
- ✓ 実プロジェクトの第一弾として、化学事業部の分析プラットフォームを設計・開発し、PoCとして運用を開始している
- ✓ 分析プラットフォームとしてはある程度求めるものが完成したが、リファレンスアーキテクチャとしては利用難易度という大きな課題が見えた
- ✓ これからもリファレンスアーキテクチャの拡充と合わせて、さらなる適応先部門と連携を広げていきながら、社内コミュニティを立ち上げ、定期的に勉強会や講演を実施する予定

