KYB における AWS を活用した IoT-Platform の構築

～予知保全システムにおける Amazon SageMaker の活用～

内藤 孝昌
KYB株式会社

aws SUMMIT ONLINE

© KYB Corporation All rights reserved.

内藤 孝昌（Takayoshi NAITO）

- 所属

  KYB 株式会社　DX（Digital Transformation）推進部

- やっていること

  機械学習、クラウドなどのデジタル技術を活用した、
  IT システムの開発・運用

- 好きな AWS サービス

Amazon SageMaker　AWS Lambda　AWS Step Functions

aws certified Solutions Architect Associate

aws certified Machine Learning Specialty

© KYB Corporation All rights reserved.

話すこと	**① KYB がどんな IoT-Platform を構築中なのか** • 「予知保全」中心 **② AWS をどのように活用しているのか** • 「Amazon SageMaker」中心
話さないこと	• **予知保全の機械学習モデル**

Agenda

01 About KYB

- 会社紹介
- KYB の IT システム開発

02 IoT-Platform

- 概要と課題
- 予知保全システム

03 Amazon SageMaker の活用

- Amazon SageMaker を使う上での工夫
- ML Workflows 構築における工夫

04 Conclusion

01

About KYB

- 会社紹介
- KYB の IT システム開発

01 About KYB
- 会社紹介
- KYB の IT システム開発

02 IoT-Platform
- 概要と課題
- 予知保全システム

03 Amazon SageMaker の活用
- Amazon SageMaker を使う上での工夫
- ML Workflows 構築における工夫

04 Conclusion

About KYB

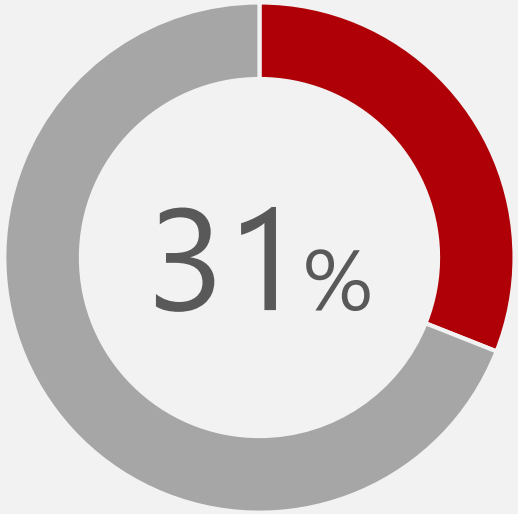
創立	創業者	代表取締役社長執行役員	売上高	従業員
1935年3月	萱場 資郎	大野 雅生	4,122億円 2018年度実績・連結	15,426名 2018年度・連結

人々の **暮らしを安全・快適** にする技術や製品の提供

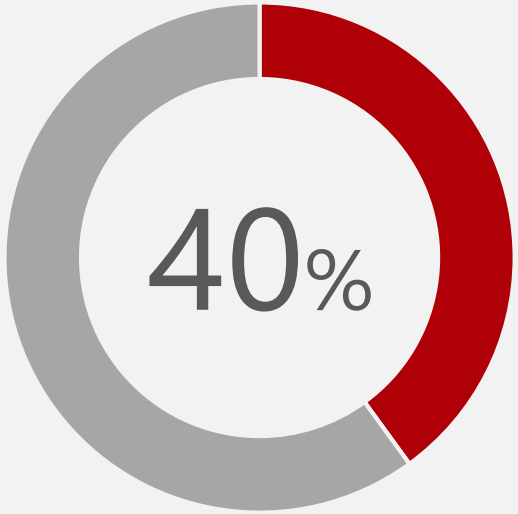
© KYB Corporation All rights reserved.

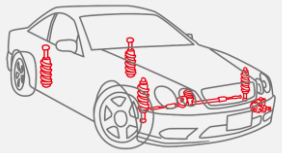
Main Product

建設機械用
油圧シリンダ(世界シェア)

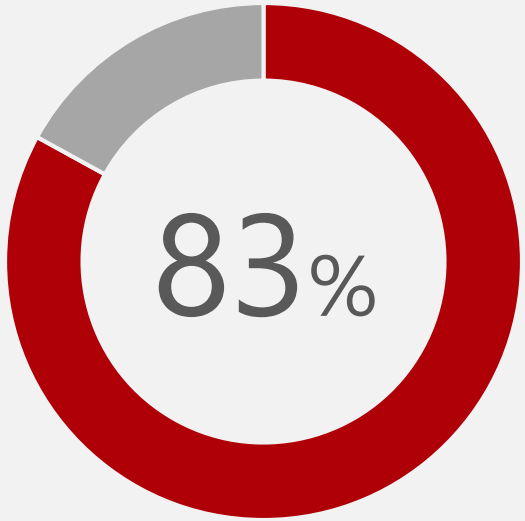


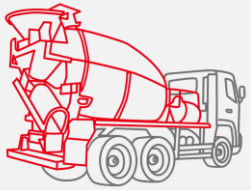
四輪車用
ショックアブソーバ（国内シェア）





コンクリートミキサー車
（国内シェア）





※当社調べ(2019年3月末現在)

© KYB Corporation All rights reserved.

創立80年以上の

老舗 **油圧技術メーカー** ですが、、、

<u>IT にも取り組んでいます</u>

© KYB Corporation All rights reserved.

KYB の IT システム開発の歴史 ～生産領域(一部)～

1996
POPICS

所謂 MES のシステムで、生産計画の提供や可動率などの KPI の自動計算を担っている中核システム

2000
品番チェックシステム

ハンディターミナルを用いた部品庫・製品庫などで使用するシステム

2002
トレサビシステム

製品のシリアル No と各種情報を紐付けた、トレーサビリティのシステム

2008
作業・検査支援システム

手順の提示、計測値の自動収集や判定を行う、組立作業や検査作業を支援するシステム

2013
QTS

鋳造工場における品質情報を全て収集し、各種分析を行えるようにしたシステム

2014
溶接不良検知

機械学習を活用し、時系列データのみで溶接品質を判定するシステム

2019
予知保全

クラウドと機械学習をフル活用した、IoT-Platformの一部機能
本日のお話の中心

▼ KYB 技報 No.60の特集記事にて近年の取り組みを紹介しています
https://www.kyb.co.jp/technical_report/data/no60j/all.pdf

© KYB Corporation All rights reserved.

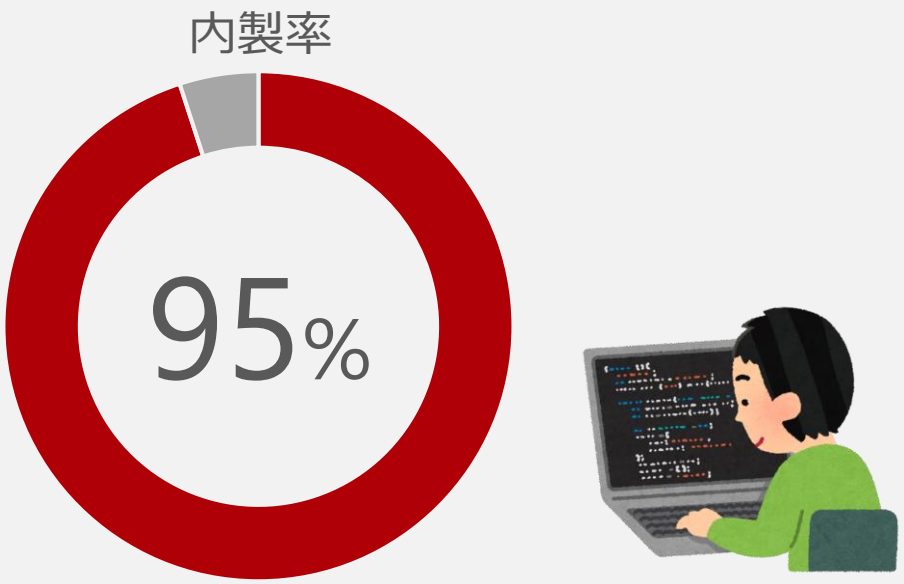
でも **油圧技術メーカー** だし、

ソフトウェア は **どうせ丸投げ** でしょ？

© KYB Corporation All rights reserved.

むしろほぼ内製

- 私のいるチームの実績
- チーム内で新規開発・運用しているシステムは数十個



※個人調べ。正確に出すともう少し高いはず。

© KYB Corporation All rights reserved.

02

IoT-Platform

- 概要と課題
- 予知保全システム

01 About KYB
- 会社紹介
- KYB の IT システム開発

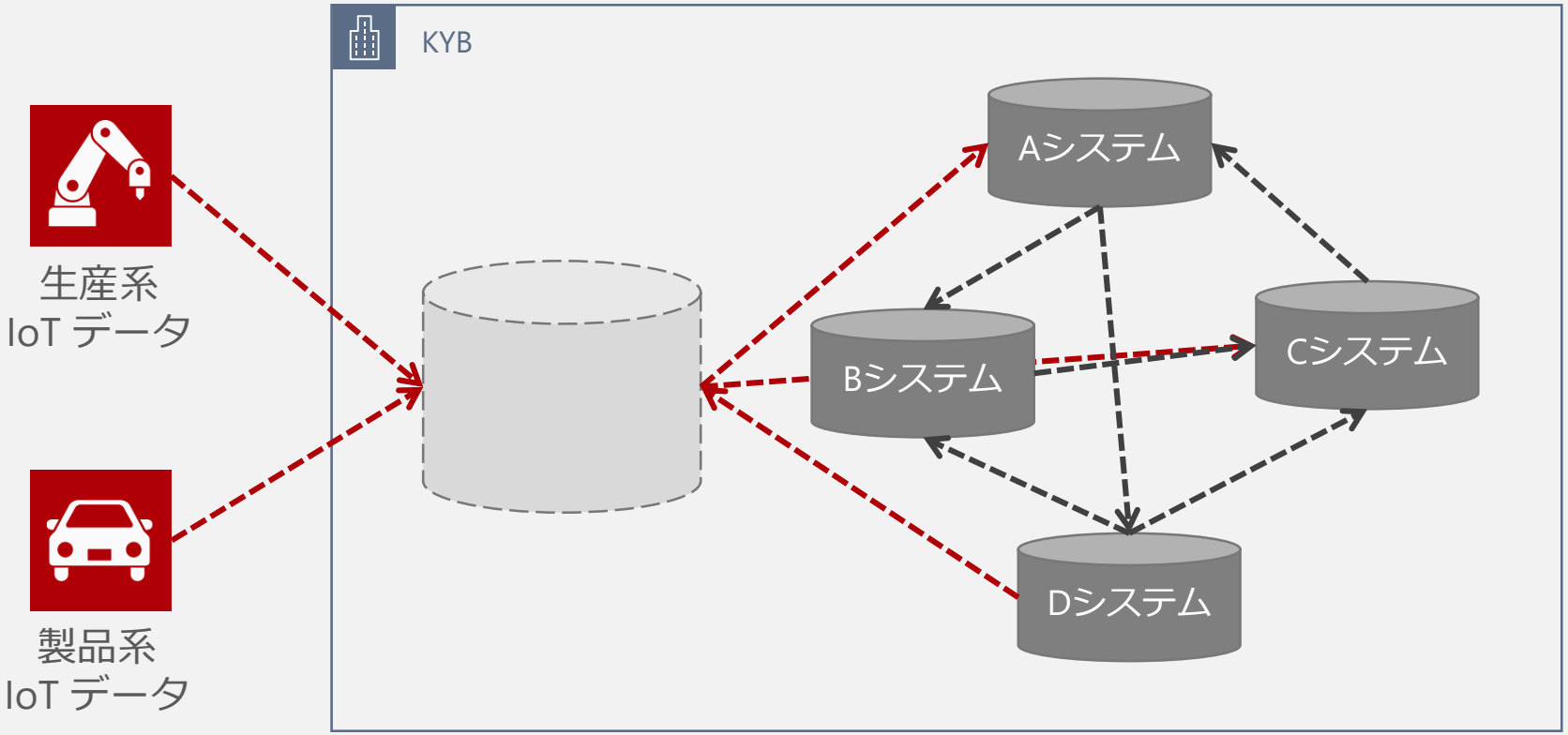
02 IoT-Platform
- 概要と課題
- 予知保全システム

03 Amazon SageMaker の活用
- Amazon SageMaker を使う上での工夫
- ML Workflows 構築における工夫

04 Conclusion

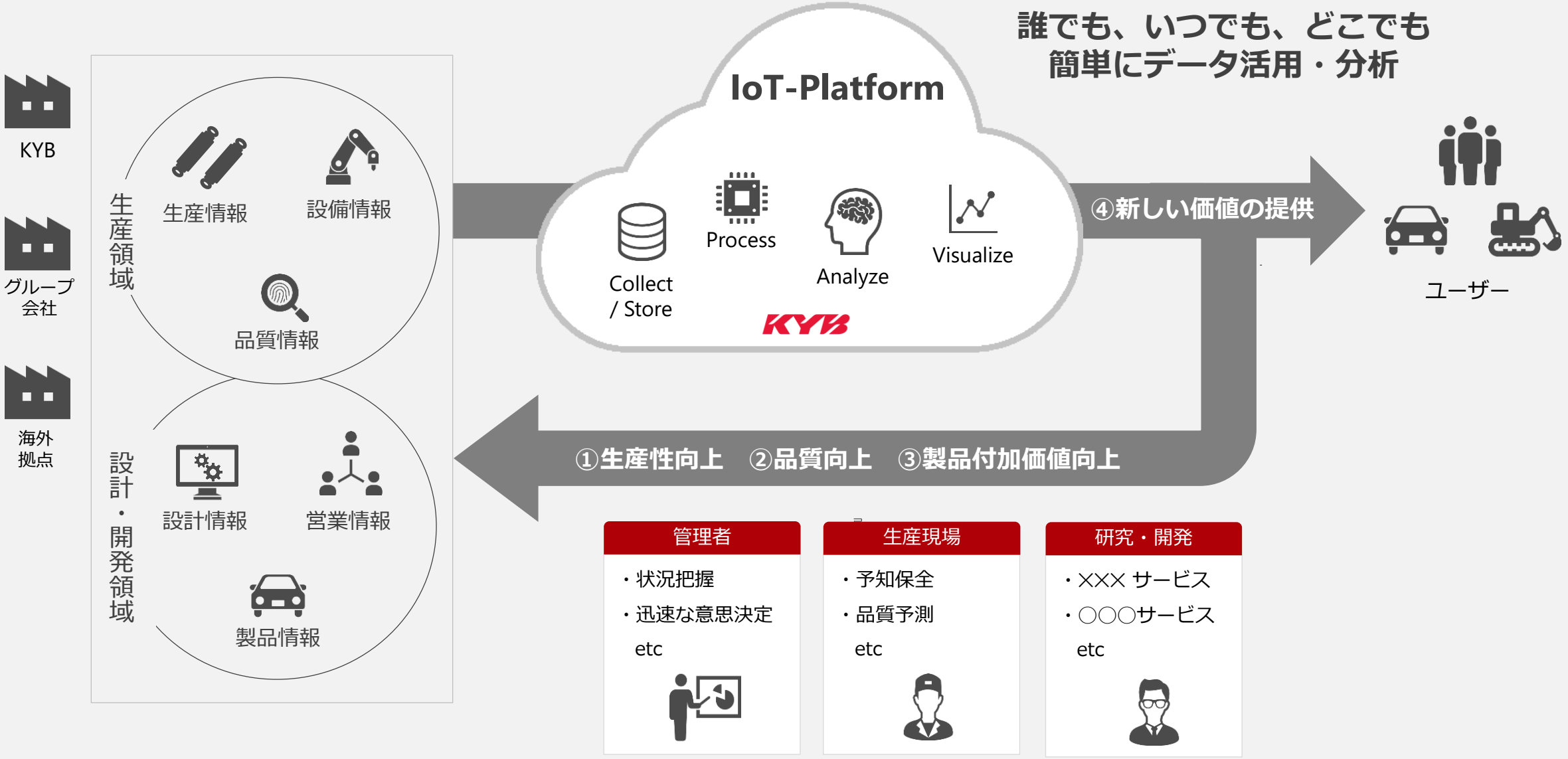
なぜ IoT-Platform か

- 既存システム自体がスパゲティ
- IoT データが増えると破綻



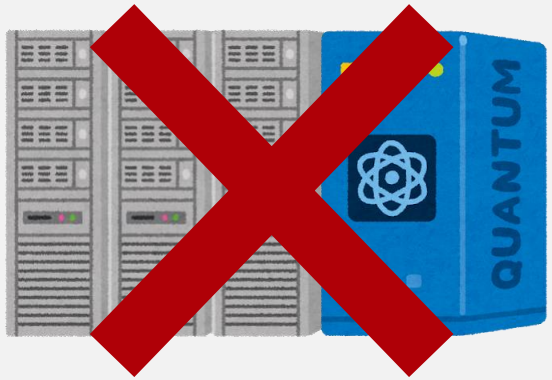
© KYB Corporation All rights reserved.

IoT-Platform の目指す姿



© KYB Corporation All rights reserved.

じゃあ Platform つくるぞー



仮想マシン
（オンプレ or クラウド）

© KYB Corporation All rights reserved.

オンプレや IaaS だと、、、

1 開発コスト ≦ 運用コスト　になってしまう

- **ビジネスに注力したい**
- クラスタ管理などに注力したくない

2 スモールスタートが できない / 難しい

- 最初から全てはうまくいかない
- 素早く構築し、繰り返し改善したい

© KYB Corporation All rights reserved.

サーバレス（FaaS）を中心に採用

IaaS	CaaS	PaaS	FaaS
Amazon EC2	Amazon ECS / Amazon EKS	AWS Elastic Beanstalk	AWS Lambda
Function	Function	Function	Function
Application	Application	Application	Application
Runtime	Runtime	Runtime	Runtime
Container	Container	Container	Container
OS	OS	OS	OS
Virtualization	Virtualization	Virtualization	Virtualization
Hardware	Hardware	Hardware	Hardware

User Management

User Management (Scalable unit)

Service Provider Management

© KYB Corporation All rights reserved.

02

IoT-Platform

- 概要と課題
- 予知保全システム

01 About KYB
- 会社紹介
- KYB の IT システム開発

02 IoT-Platform
- 概要と課題
- 予知保全システム

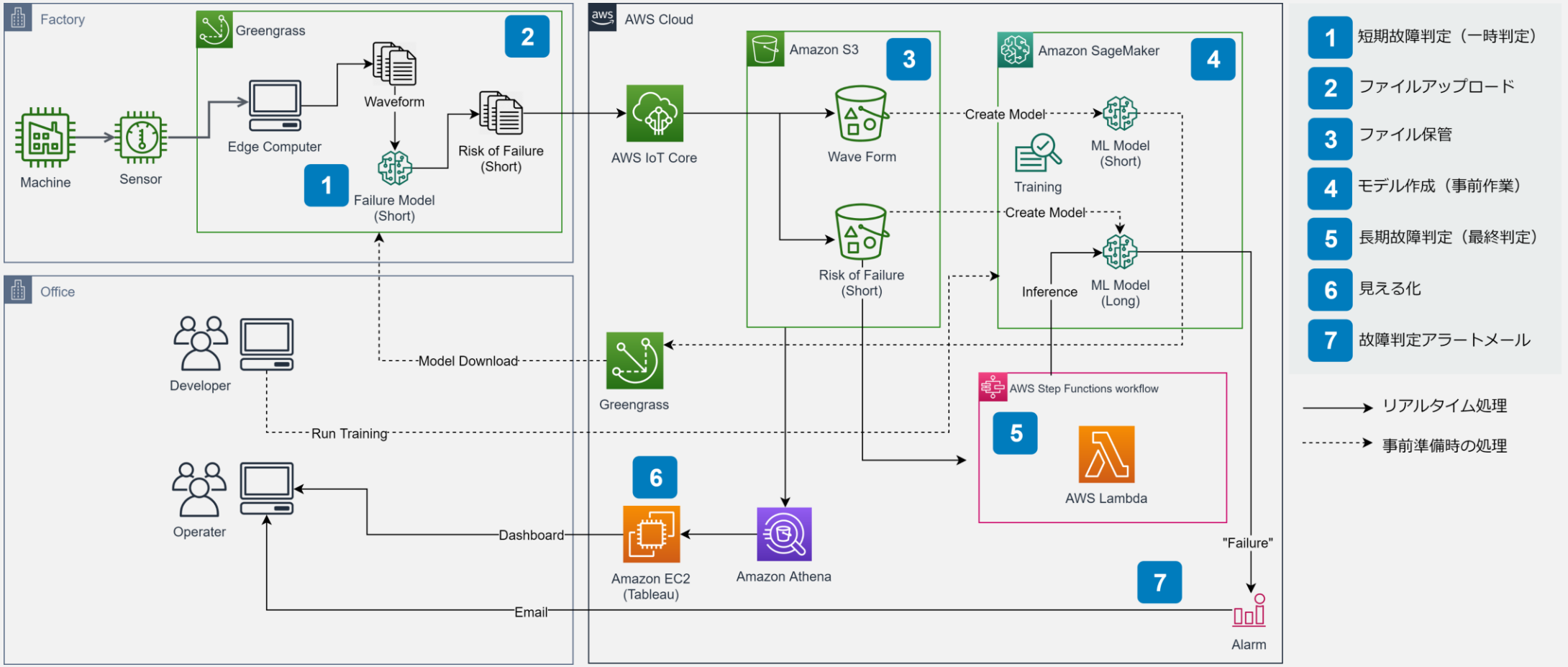
03 Amazon SageMaker の活用
- Amazon SageMaker を使う上での工夫
- ML Workflows 構築における工夫

04 Conclusion

保全の種類

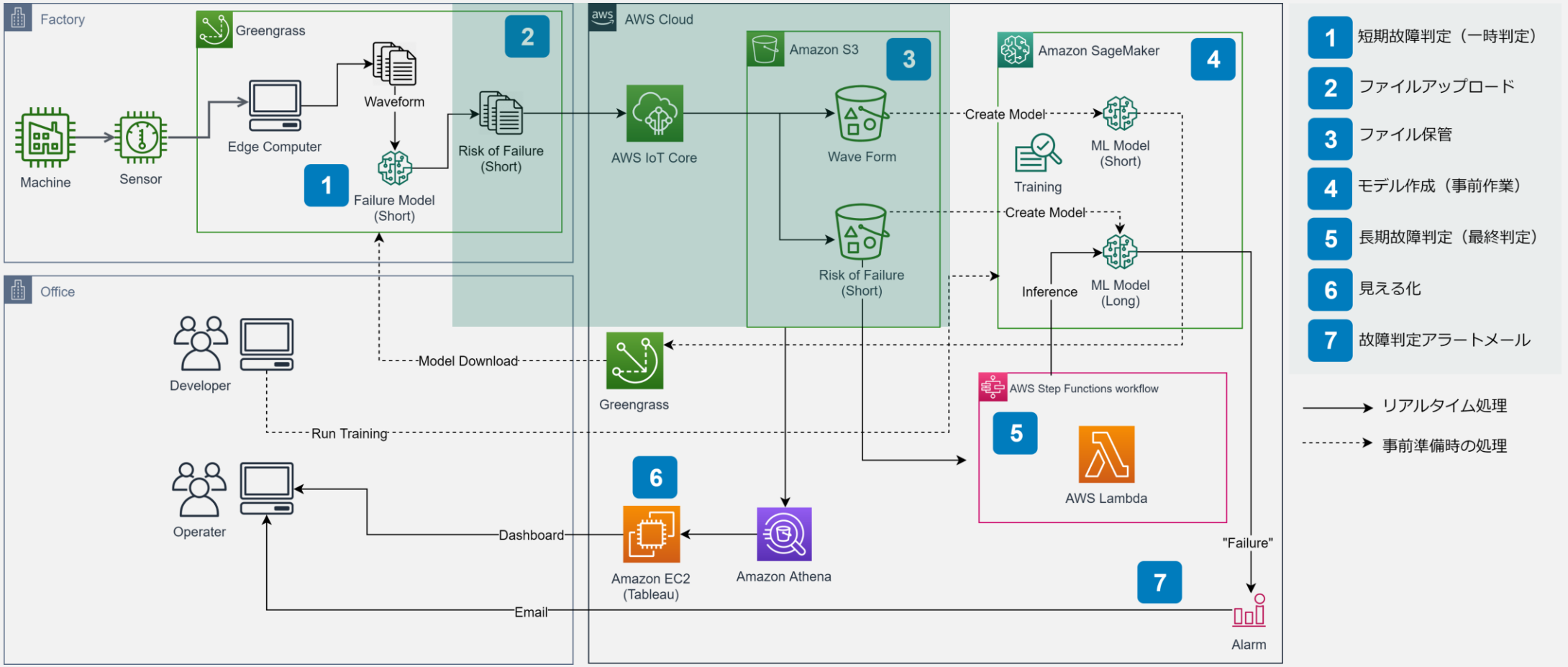
© KYB Corporation All rights reserved.

予知保全システムの全体像



© KYB Corporation All rights reserved.

データ収集



© KYB Corporation All rights reserved.

AWS IoT を用いた **デバイス管理** は魅力的ですよね？

AWS IoT Core

AWS IoT Device Defender

AWS IoT Device Management

© KYB Corporation All rights reserved.

でも、**大容量データ** のアップロードは **苦手**

予知保全用
センサデータ

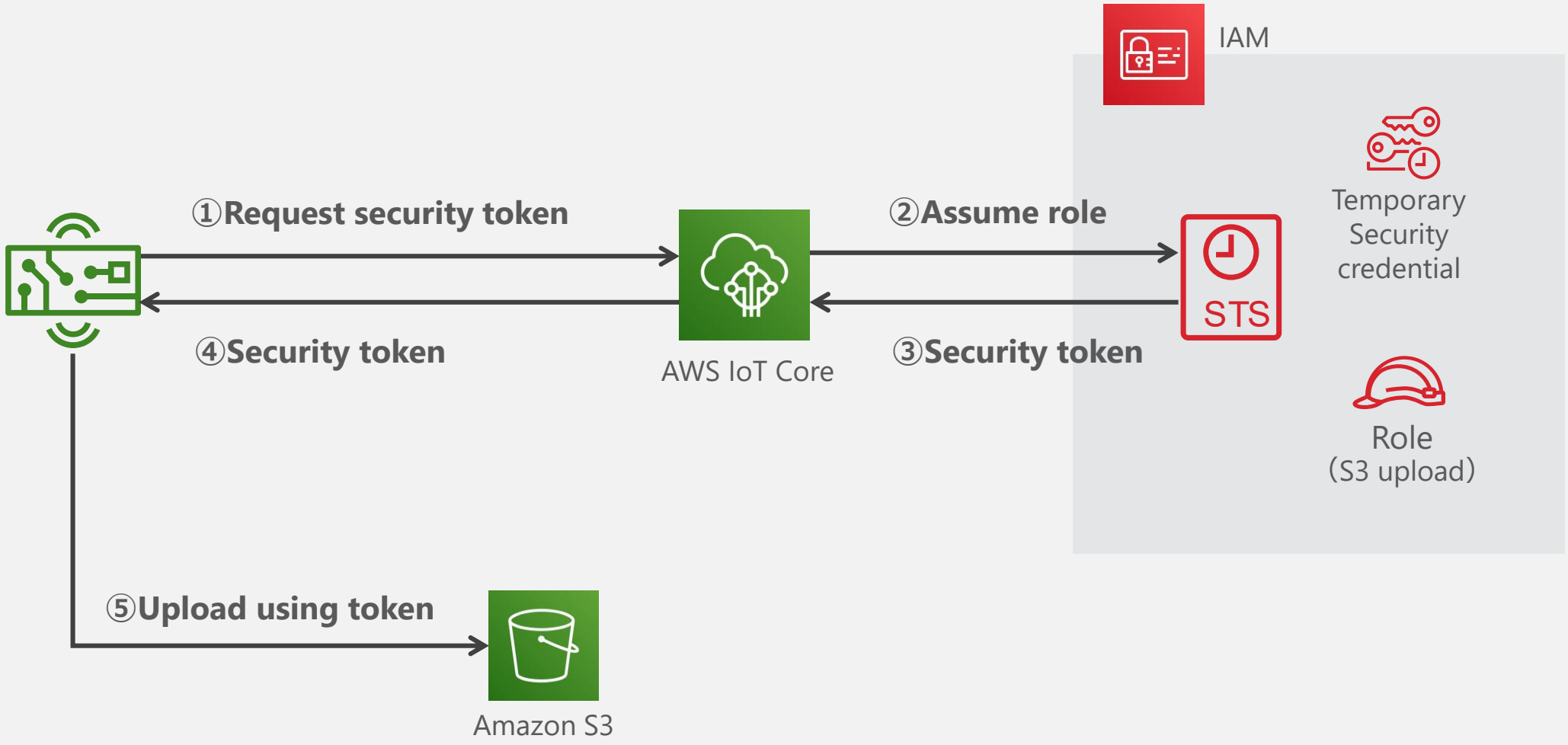
✓ 収集データ例

- 種類： 振動データ
- 収集頻度： 30分毎、10秒間
- **データ量： 40MB/ファイル**

MQTTでストリーミング
できる量ではない

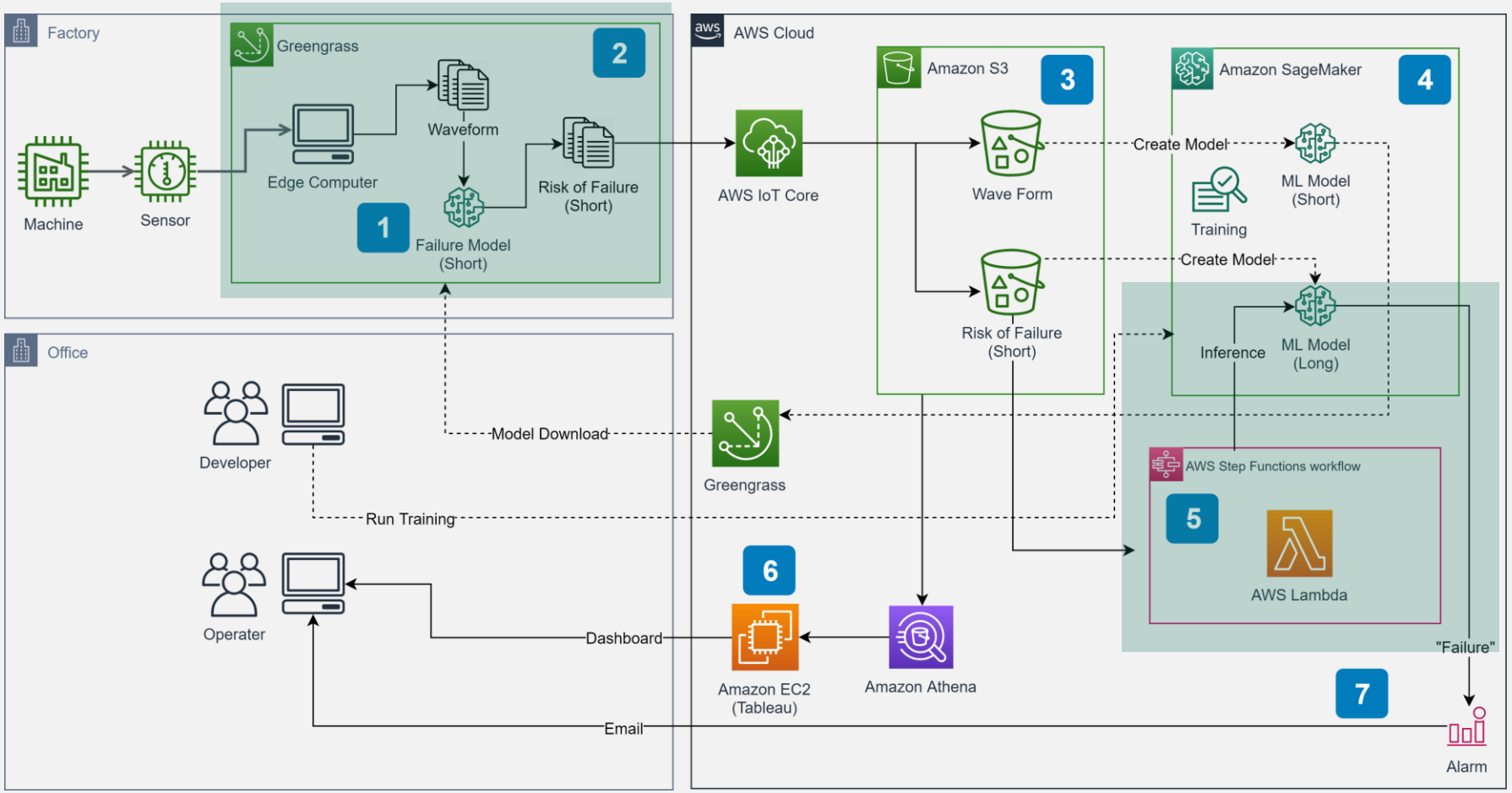
© KYB Corporation All rights reserved.

データ収集



© KYB Corporation All rights reserved.

故障判定



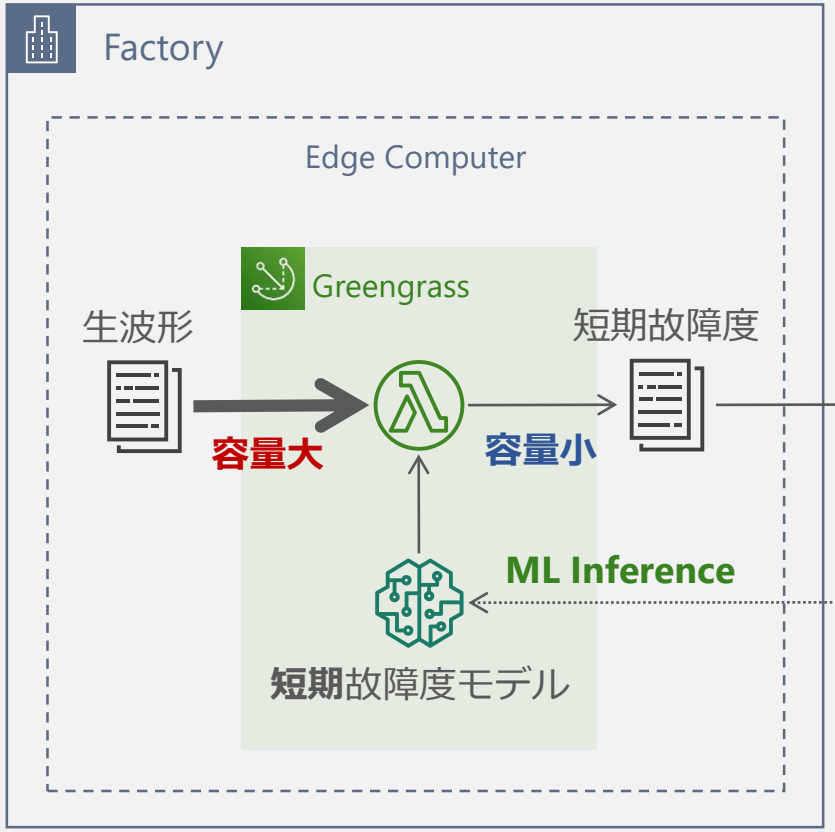
1 短期故障判定（一時判定）
2 ファイルアップロード
3 ファイル保管
4 モデル作成（事前作業）
5 長期故障判定（最終判定）
6 見える化
7 故障判定アラートメール

→ リアルタイム処理
⇢ 事前準備時の処理

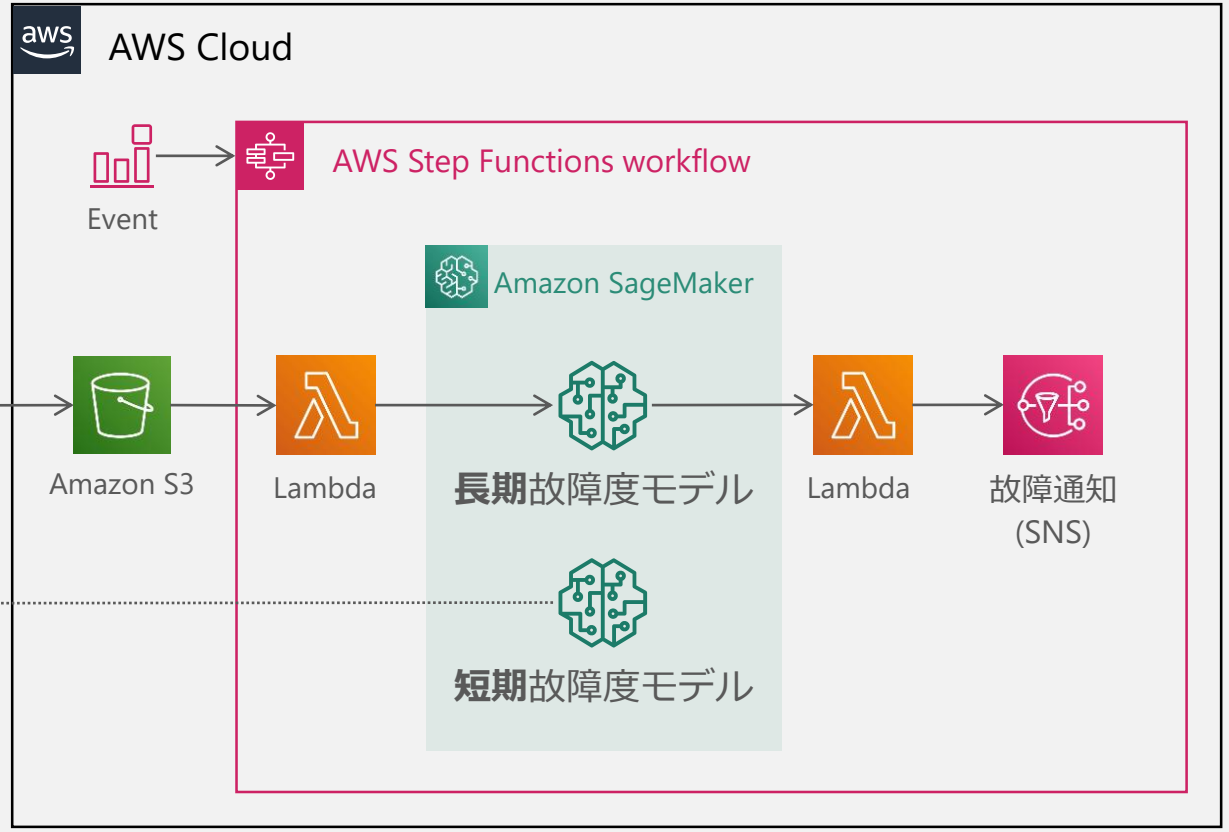
© KYB Corporation All rights reserved.

故障判定

短期 故障判定（一時判定）

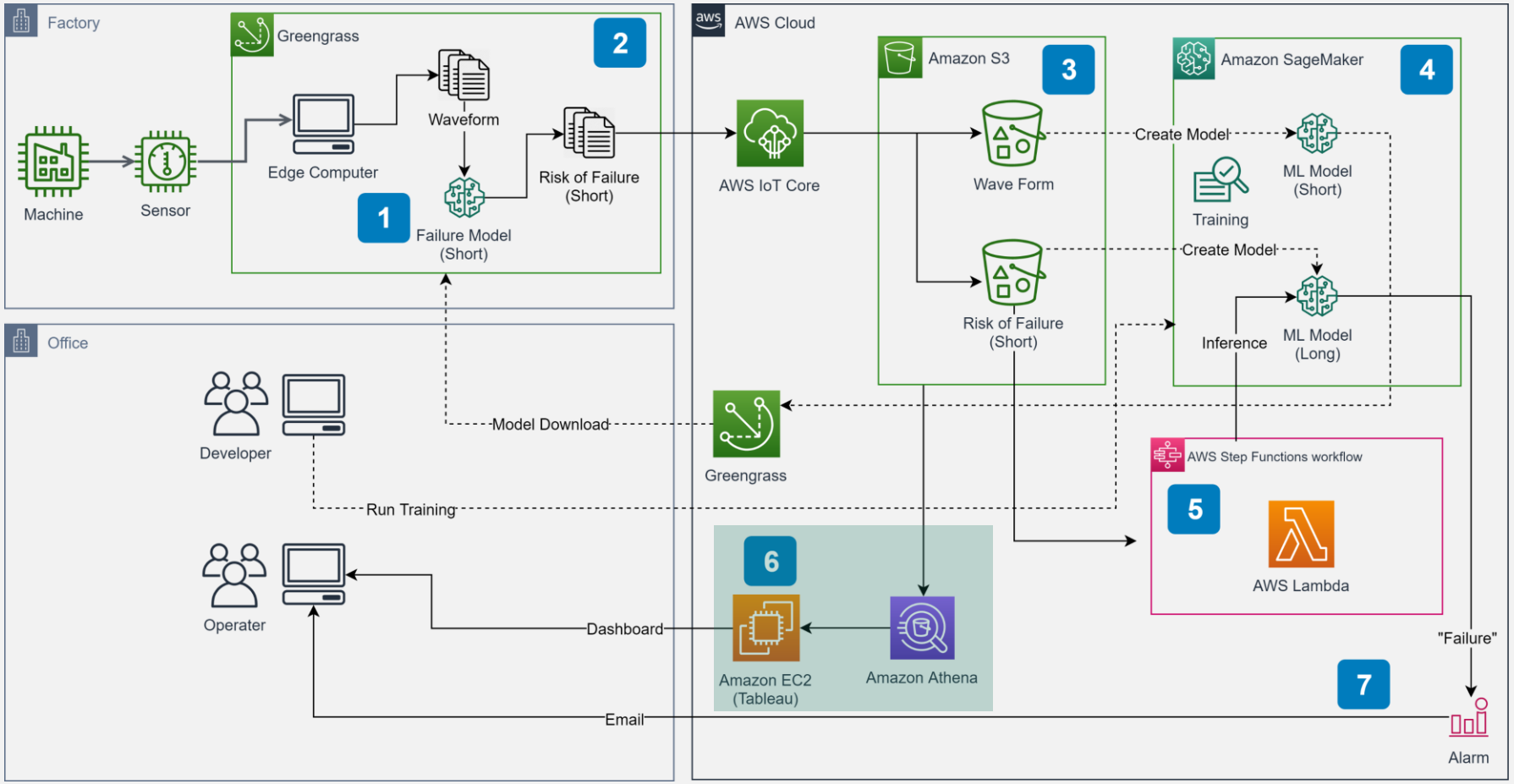


長期 故障判定（最終判定）



© KYB Corporation All rights reserved.

見える化



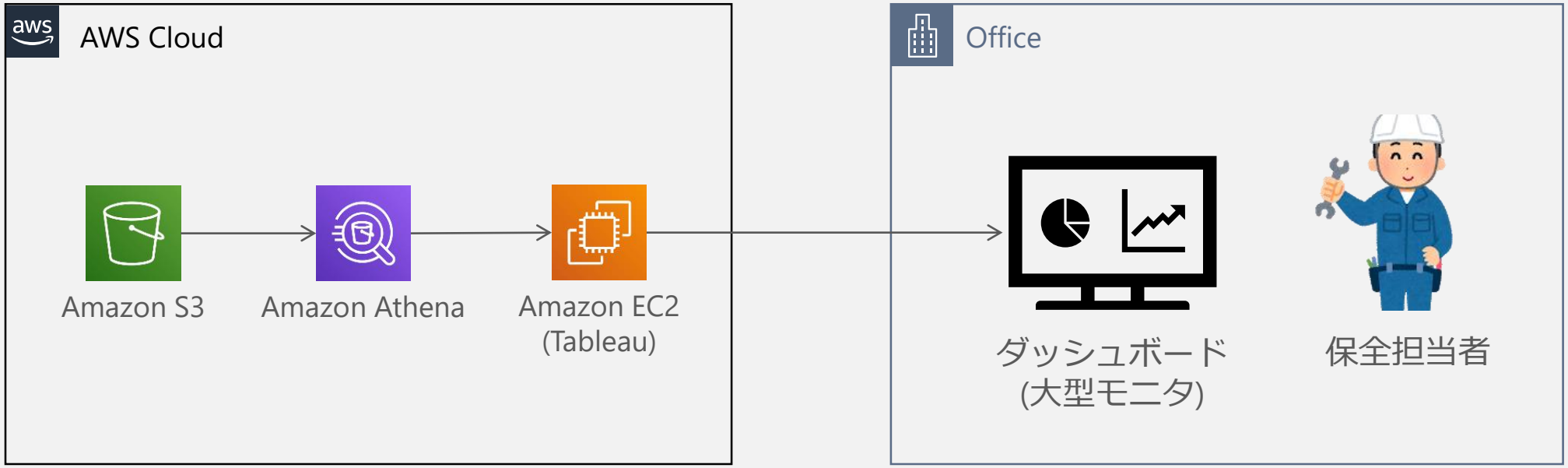
1 短期故障判定（一時判定）
2 ファイルアップロード
3 ファイル保管
4 モデル作成（事前作業）
5 長期故障判定（最終判定）
6 見える化
7 故障判定アラートメール

→ リアルタイム処理
⇢ 事前準備時の処理

© KYB Corporation All rights reserved.

見える化

使い方とコストを考慮し、**まずは Amazon Athena 採用**



※ダッシュボードの内容は重要なノウハウなので割愛

© KYB Corporation All rights reserved.

03

Amazon SageMaker の活用

- Amazon SageMaker を使う上での工夫
- ML Workflows 構築における工夫

01 About KYB
- 会社紹介
- KYB の IT システム開発

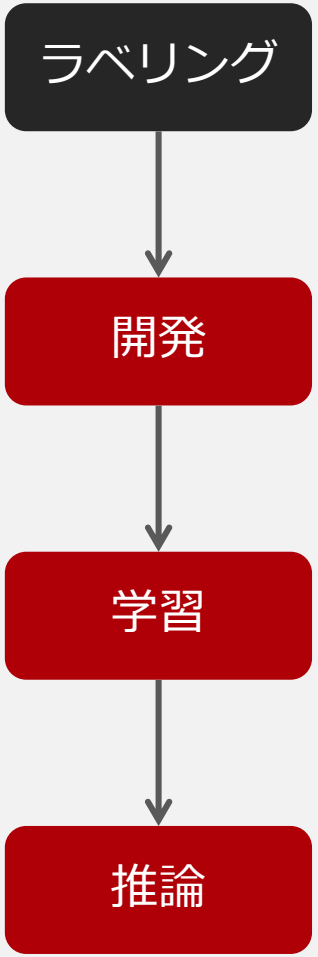
02 IoT-Platform
- 概要と課題
- 予知保全システム

03 Amazon SageMaker の活用
- Amazon SageMaker を使う上での工夫
- ML Workflows 構築における工夫

04 Conclusion

Amazon SageMaker とは

機械学習のあらゆるワークフローをサポート する、フルマネージド型サービス



- ラベリング支援

- マネージド Jupyter Notebook
- 主要フレームワークのサポート

- API で学習ジョブ
- 分散学習、チューニングに対応

- API で推論エンドポイントをデプロイ
- バッチ推論サポート

© KYB Corporation All rights reserved.

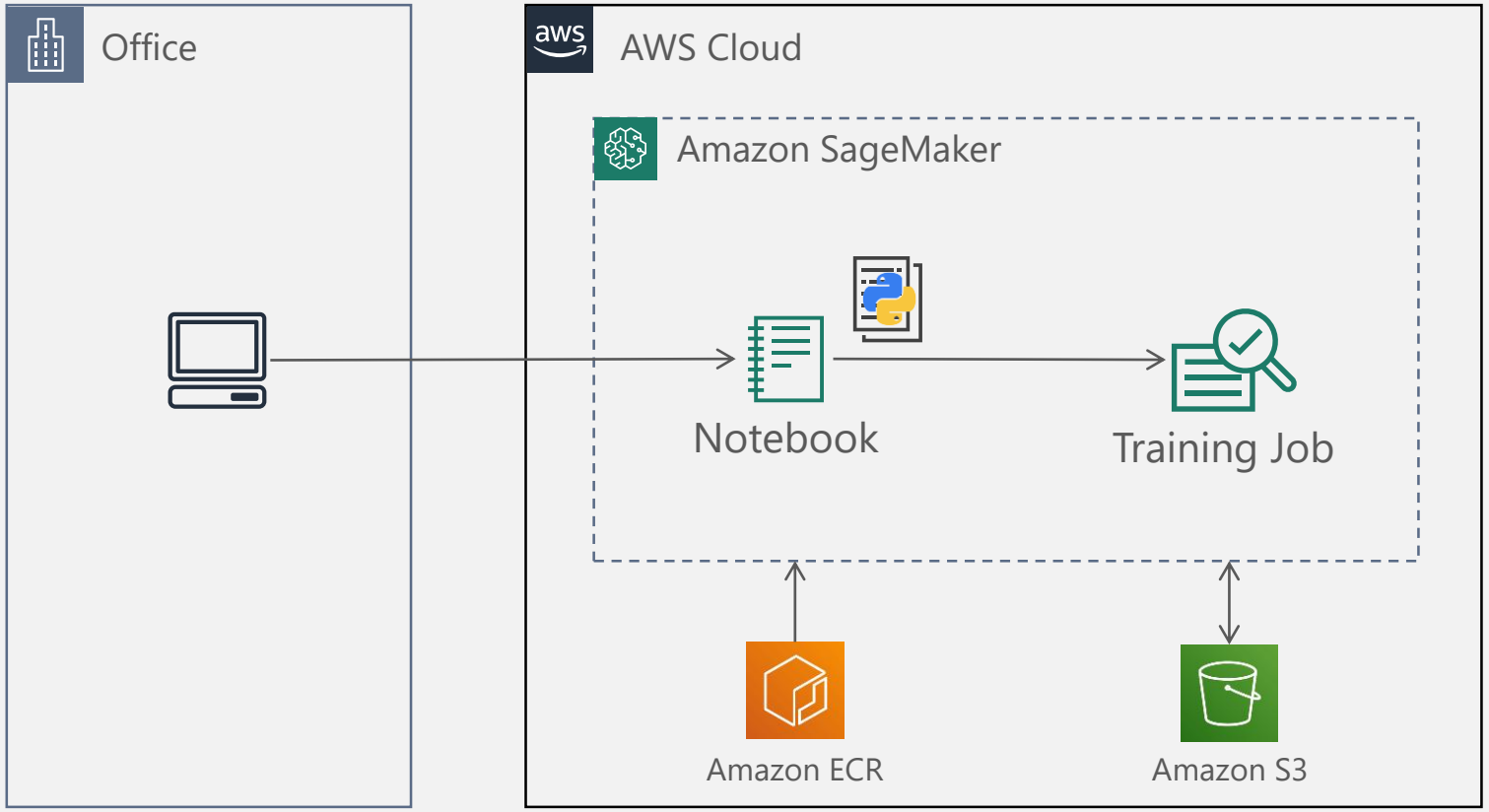
使ってみたらかなり **便利** だった

でも **最初はうまく行かなくて、色々と工夫** した

© KYB Corporation All rights reserved.

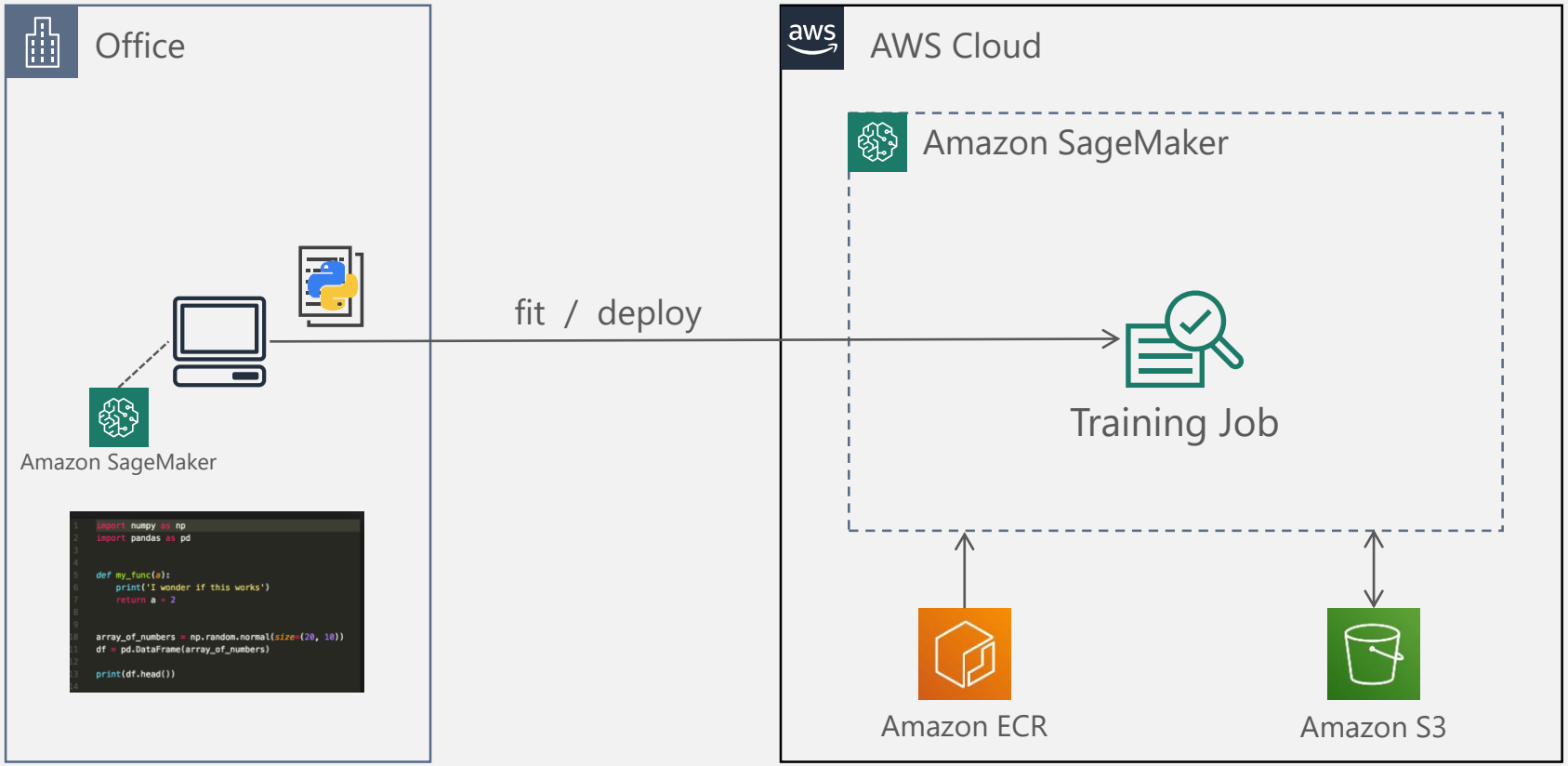
課題1: 使い慣れたエディタで開発したい

Notebook インスタンスが提供されているので、普通はそれを使う



© KYB Corporation All rights reserved.

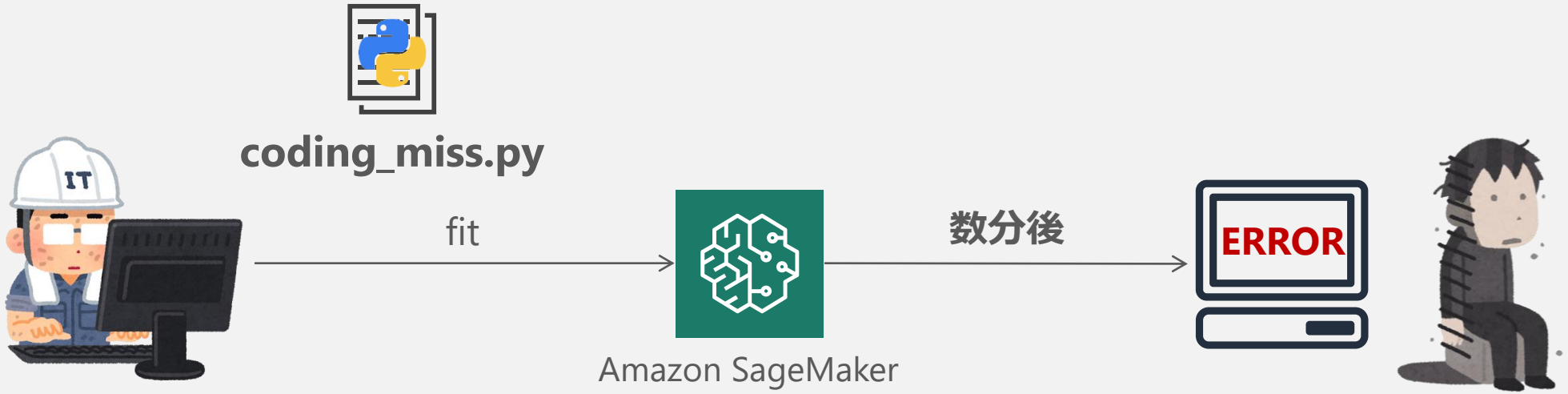
解決1: 開発マシンに SageMaker Python SDK 導入



© KYB Corporation All rights reserved.

課題2: 開発のイテレーションを高速化したい

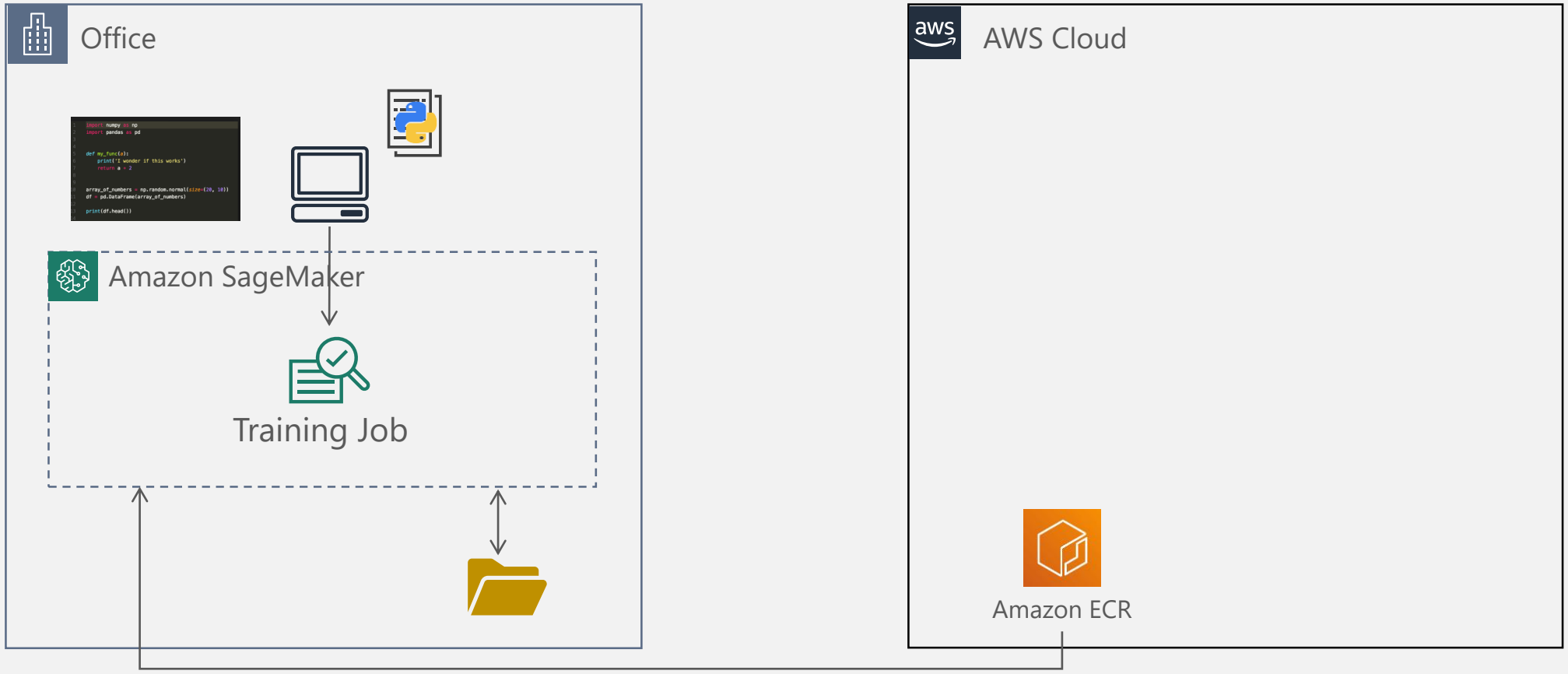
開発初期は**コンテナ起動時間がネック**



© KYB Corporation All rights reserved.

解決2: Local モードの活用

開発初期 は少量データで **ローカル開発**

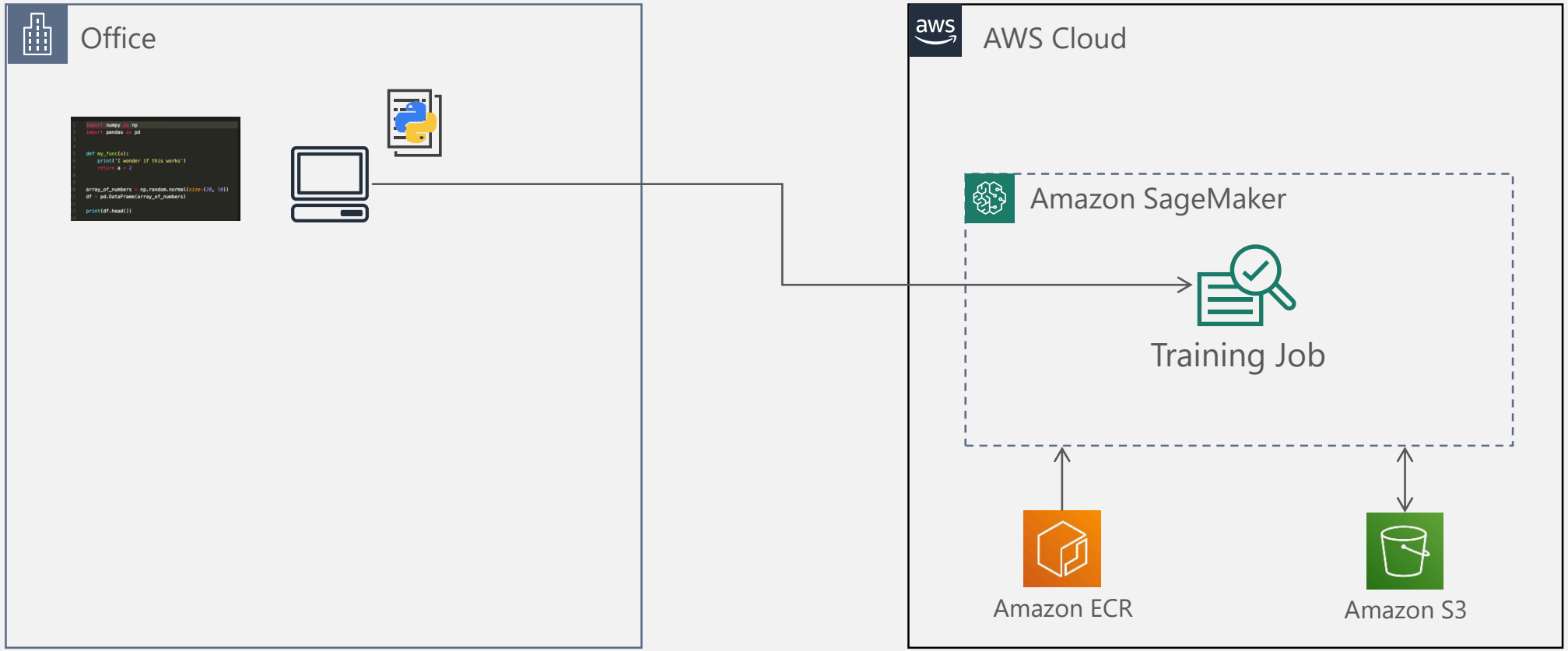


※ローカルのdockerイメージの利用や、逆にS3データを利用することも可能

© KYB Corporation All rights reserved.

解決2: Local モードの活用

後半 は大量データを使って **AWS 上で学習**



© KYB Corporation All rights reserved.

解決2: Local モードの活用

- **role と train_instance_type** を変更するだけ
- **データもローカル** を利用可能

通常モード

```
role = get_execution_role()
train_input = "s3://foo-bar/train_aws.csv"

estimator = PyTorch(entry_point='mnist.py',
                    role=role,
                    train_instance_count=1,
                    train_instance_type='ml.p2.xlarge',
                    framework_version='1.4.0')

estimator.fit({'train': train_input})
```

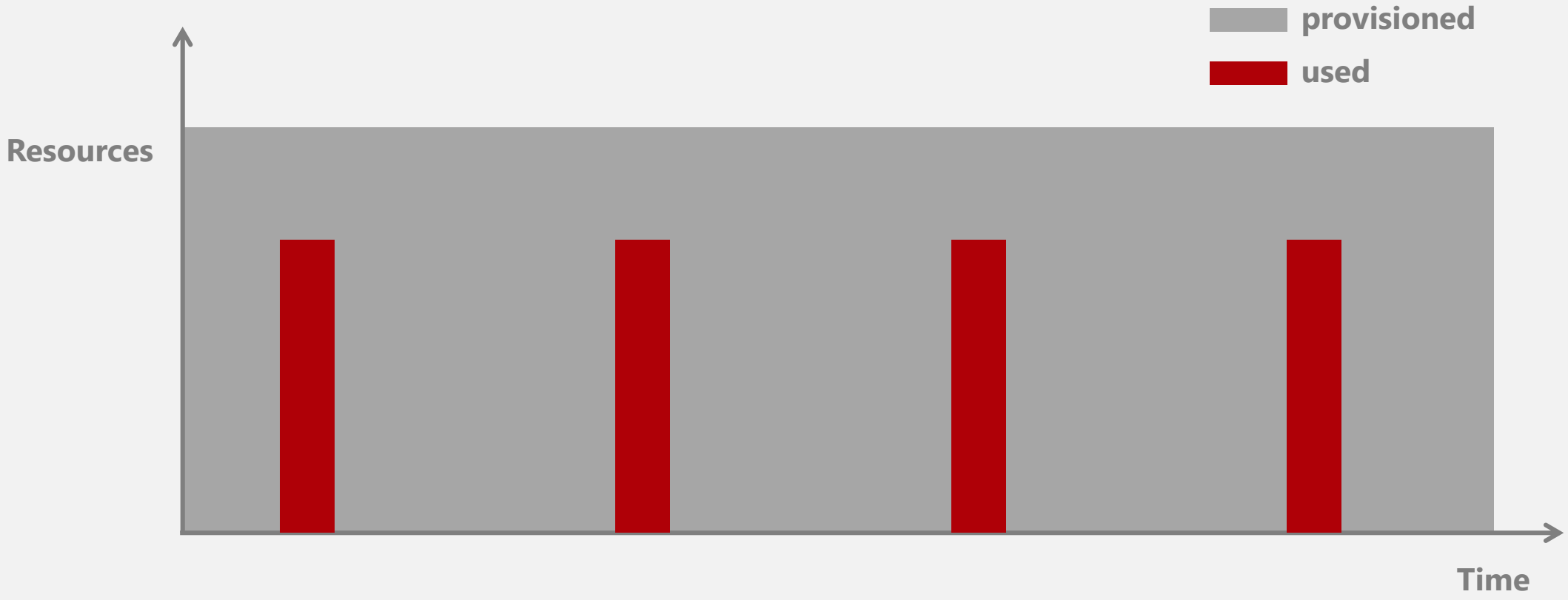
Localモード

```
role = 'arn:aws:iam::xxxxx:role/SageMakerExecutionRole'
train_input = "file://foo-bar/train_local.csv"

estimator = PyTorch(entry_point='mnist.py',
                    role=role,
                    train_instance_count=1,
                    train_instance_type='local_gpu',
                    framework_version='1.4.0')

estimator.fit({'train': train_input})
```

© KYB Corporation All rights reserved.

課題3: 低頻度の推論を低コストで実現したい

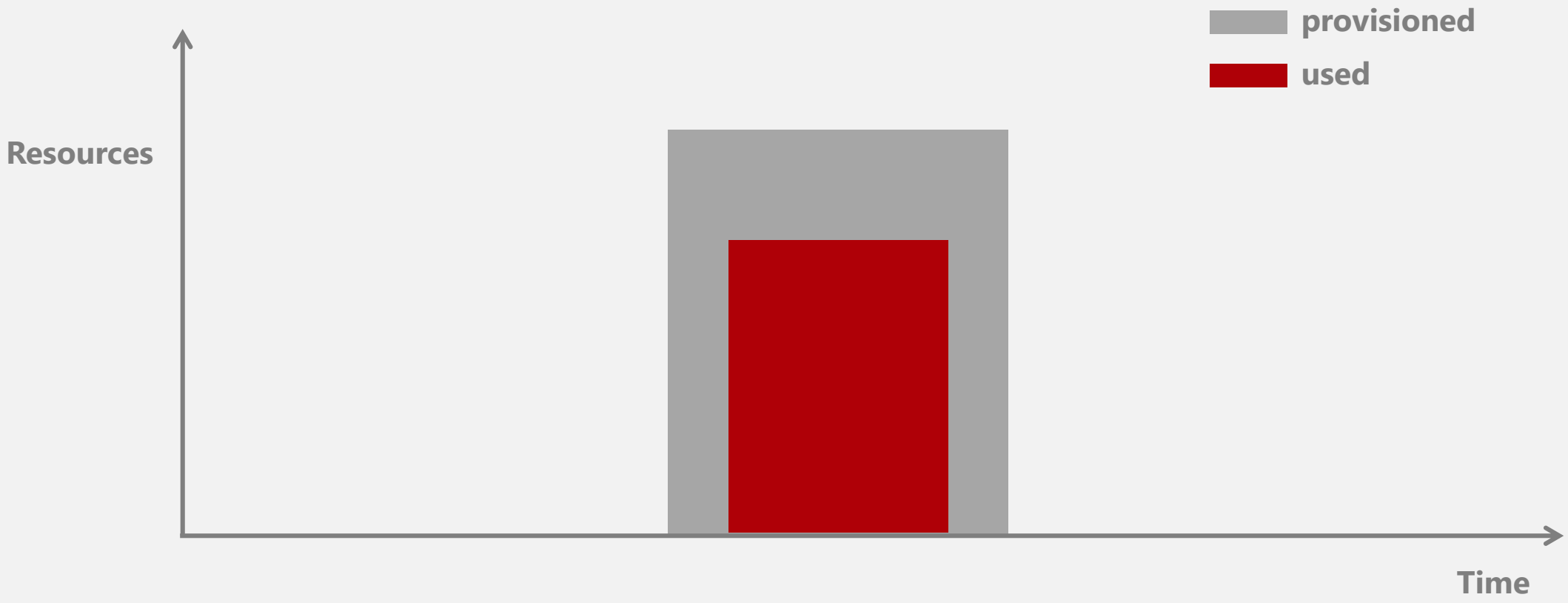
低頻度の利用では、**エンドポイントはリソースの無駄が多い**



© KYB Corporation All rights reserved.

解決3: Batch Transform の活用

低頻度かつ非リアルタイムで良いなら Batch Transform



© KYB Corporation All rights reserved.

その他: Managed Spot Training の活用

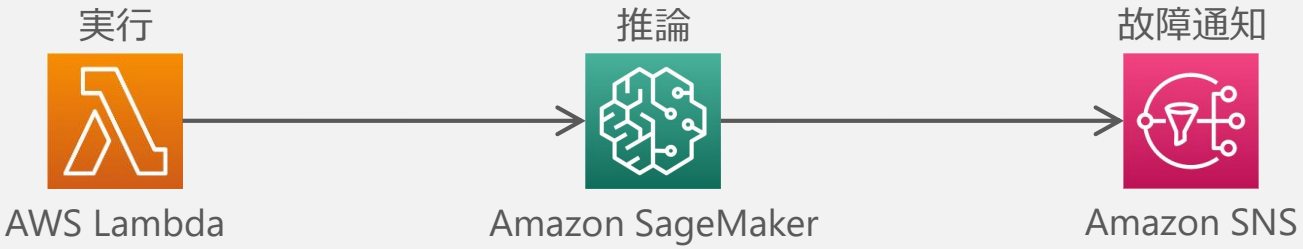
- **スポットの使用フラグと時間を設定** するだけ
- **中断対策**として **checkpoint** を利用可

```
estimator = PyTorch(entry_point='mnist.py',
                    role=role,
                    train_instance_count=1,
                    train_instance_type='ml.p2.xlarge',
                    train_max_run=6000,
                    train_use_spot_instances=True,
                    train_max_wait=7200,
                    checkpoint_s3_uri='s3://example-bucket/checkpoints',
                    checkpoint_local_path='/opt/ml/checkpoints/',
                    framework_version='1.4.0')
```

© KYB Corporation All rights reserved.

モデルの学習は出来たし、推論の確認もした。

あとは実行して故障通知するだけとか余裕じゃん。



© KYB Corporation All rights reserved.

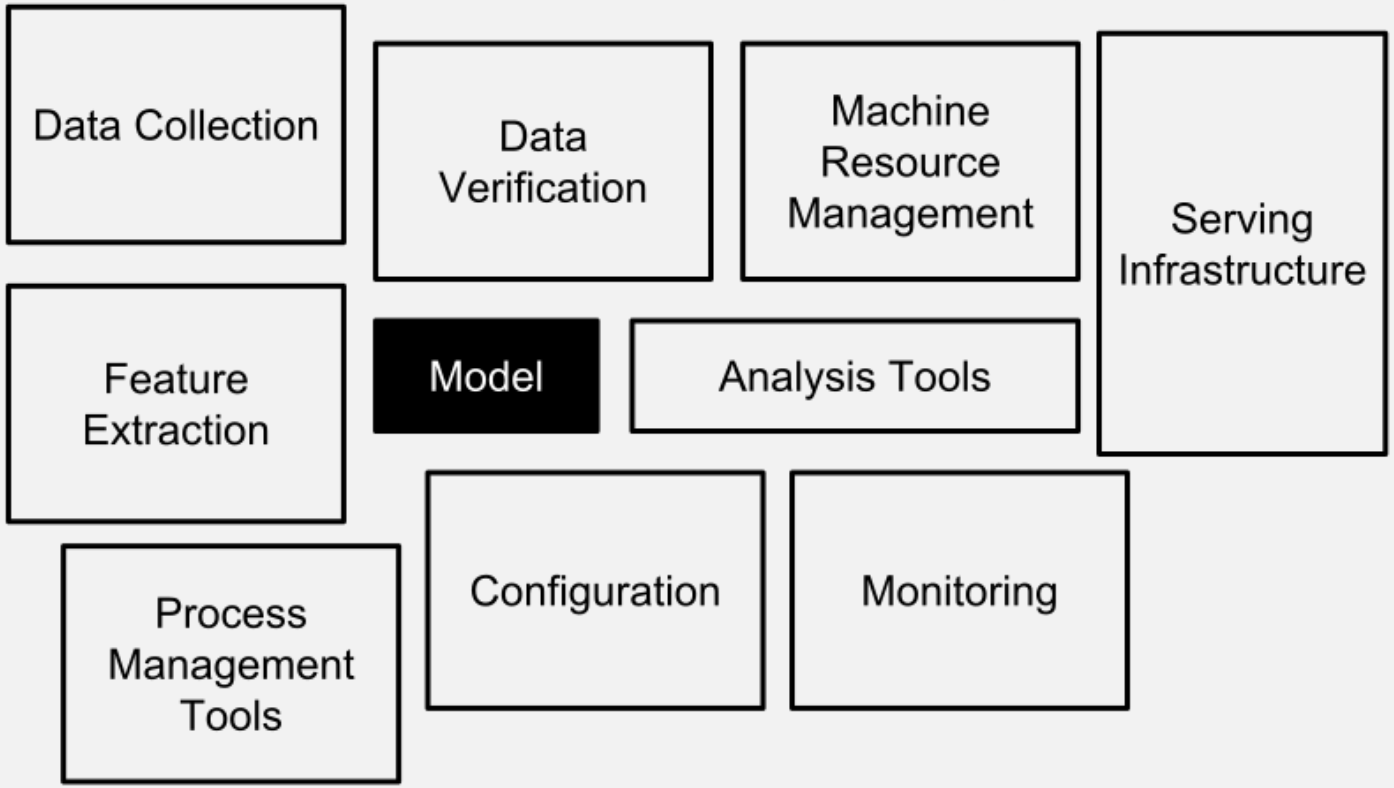
もちろんそんな簡単なわけがない

むしろここからが重要

© KYB Corporation All rights reserved.

ML Workflows の実現

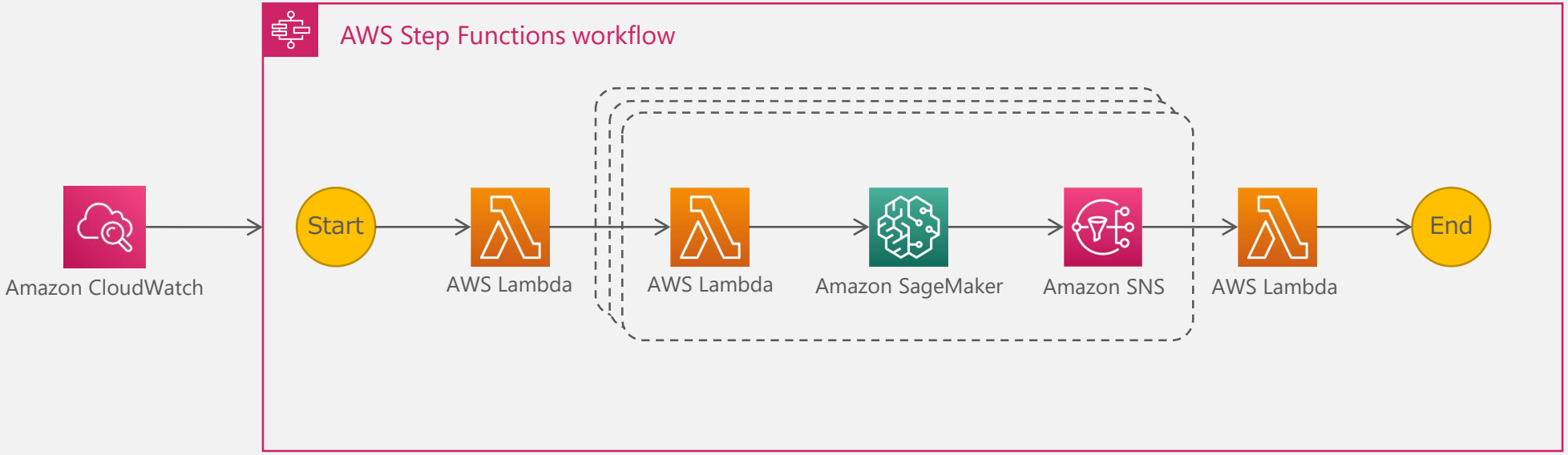
どうやって全体を
管理するの？



https://developers.google.com/machine-learning/testing-debugging/pipeline/overview?hl=zh-tw

© KYB Corporation All rights reserved.

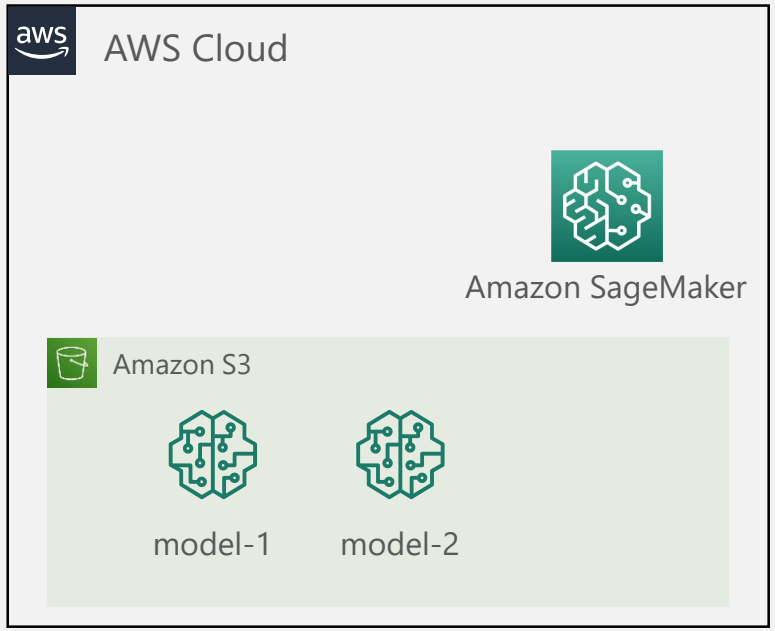
AWS Step Functions でお手軽 Workflow 化



© KYB Corporation All rights reserved.

学習 Workflow

未実装（あえて人間のオペレーションを挟む）



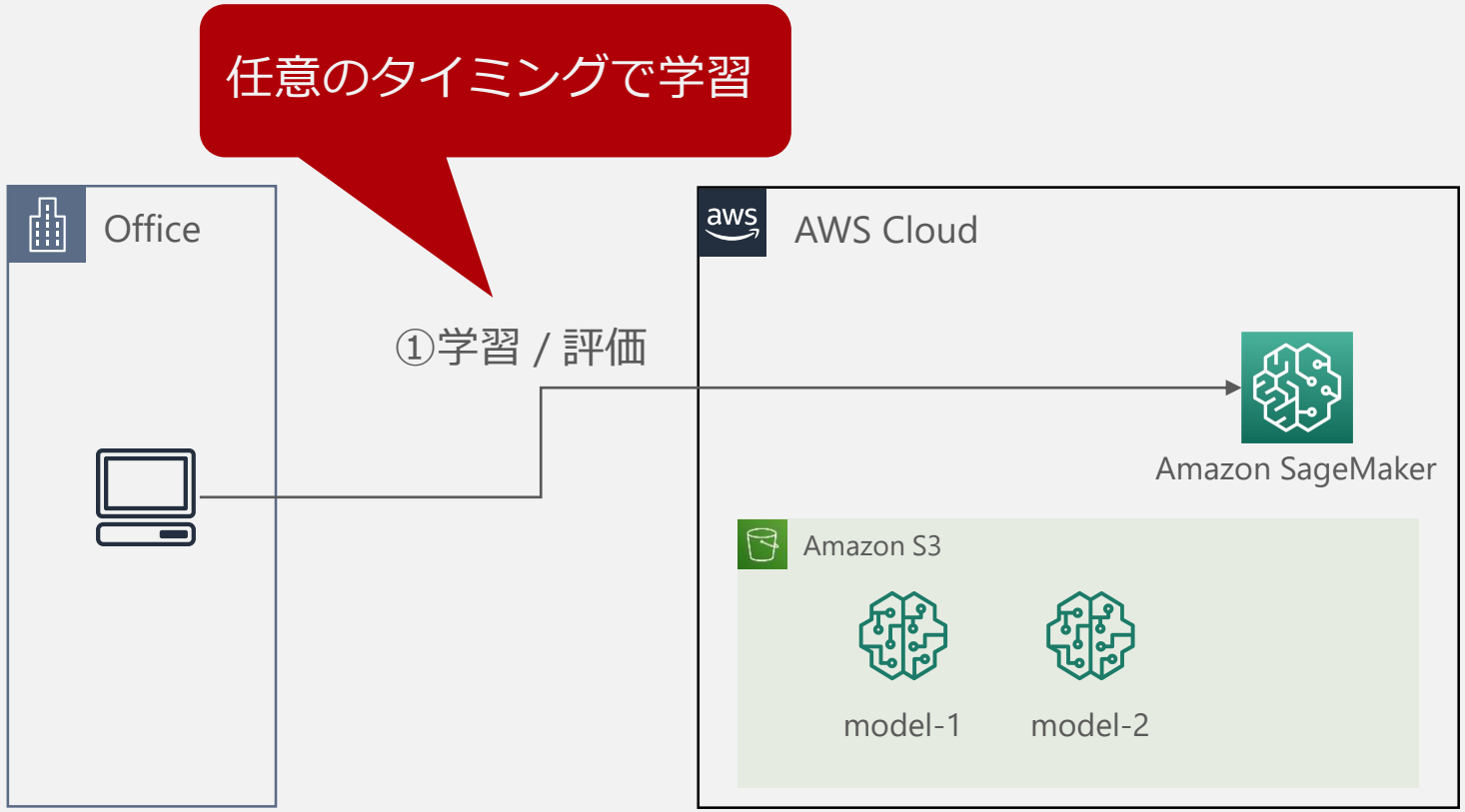
モデル名	作成日	タグ
model-1	2020-5-12	prod
model-2	2020-5-13	dev

※ 実際にはトレーニングジョブにタグ付け

© KYB Corporation All rights reserved.

学習 Workflow

未実装（あえて人間のオペレーションを挟む）



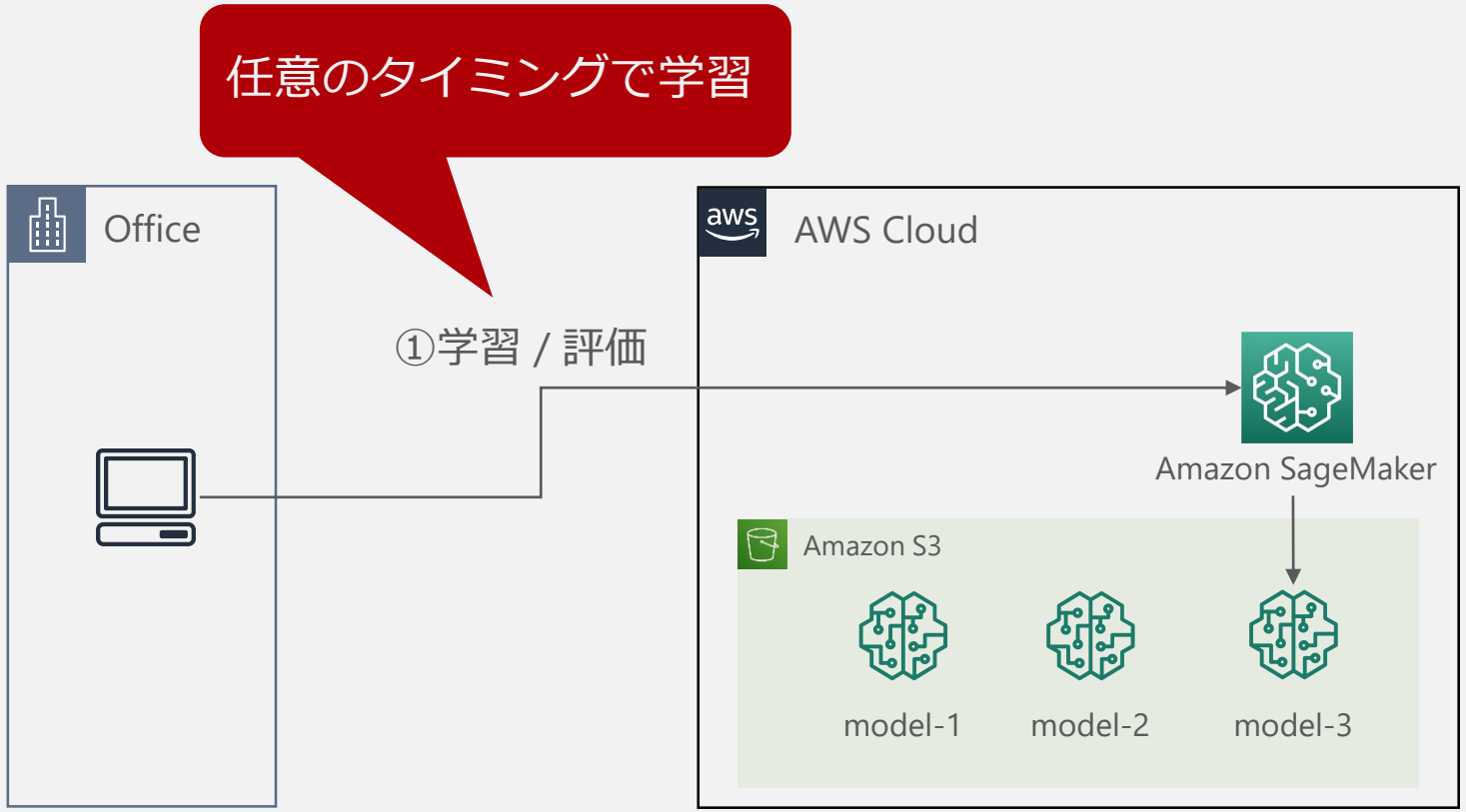
モデル名	作成日	タグ
model-1	2020-5-12	prod
model-2	2020-5-13	dev

※ 実際にはトレーニングジョブにタグ付け

© KYB Corporation All rights reserved.

学習 Workflow

未実装（あえて人間のオペレーションを挟む）



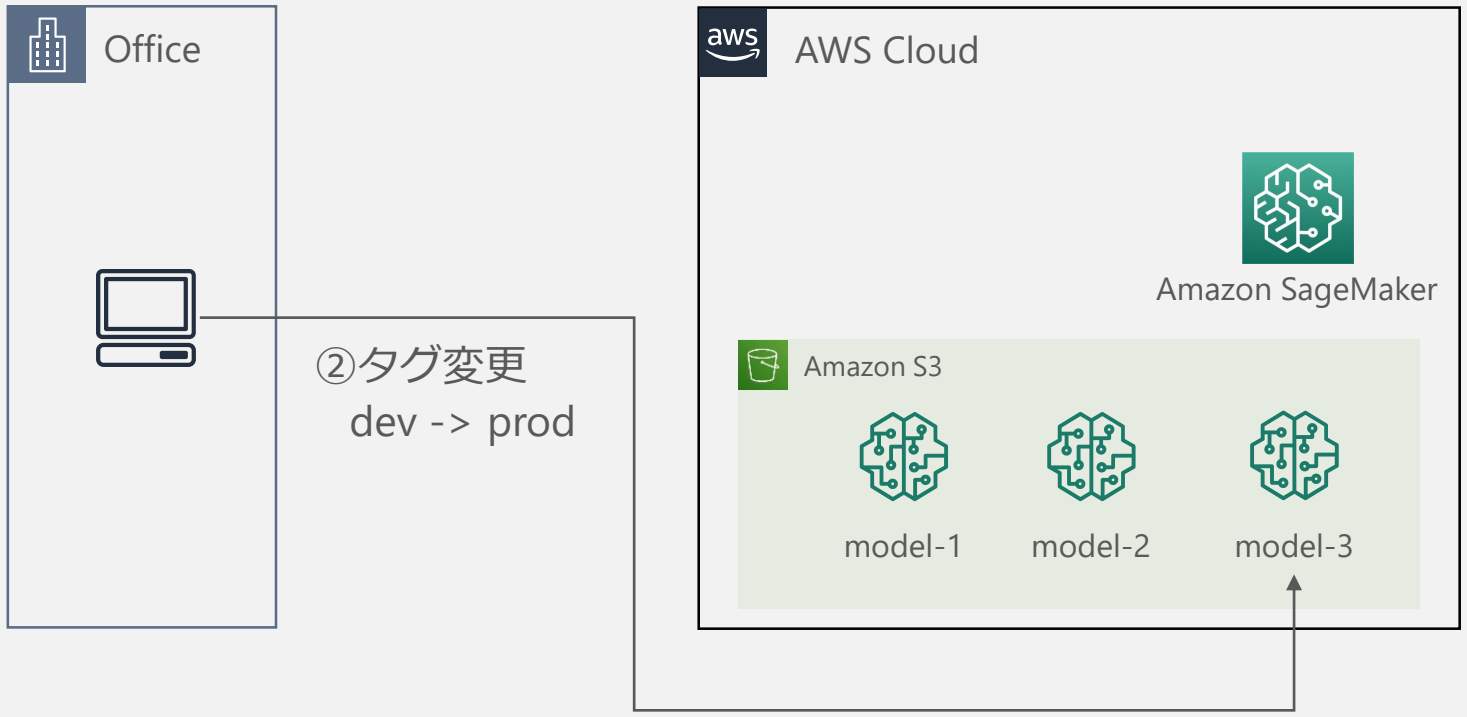
モデル名	作成日	タグ
model-1	2020-5-12	prod
model-2	2020-5-13	dev
model-3	2020-5-14	dev

※ 実際にはトレーニングジョブにタグ付け

© KYB Corporation All rights reserved.

学習 Workflow

未実装（あえて人間のオペレーションを挟む）



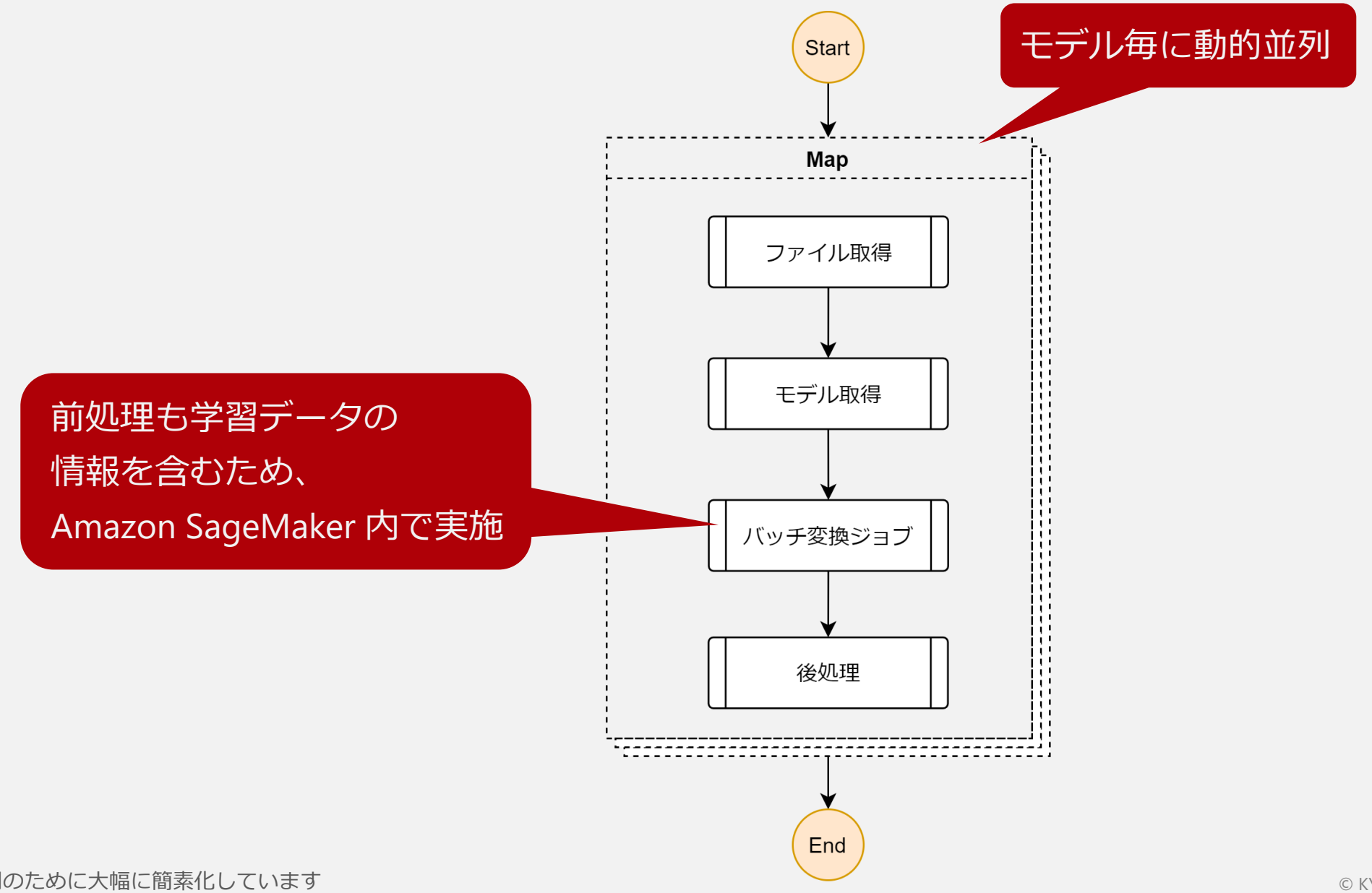
タグ変更でモデル差し替え

モデル名	作成日	タグ
model-1	2020-5-12	prod
model-2	2020-5-13	dev
model-3	2020-5-14	prod

※ 実際にはトレーニングジョブにタグ付け

© KYB Corporation All rights reserved.

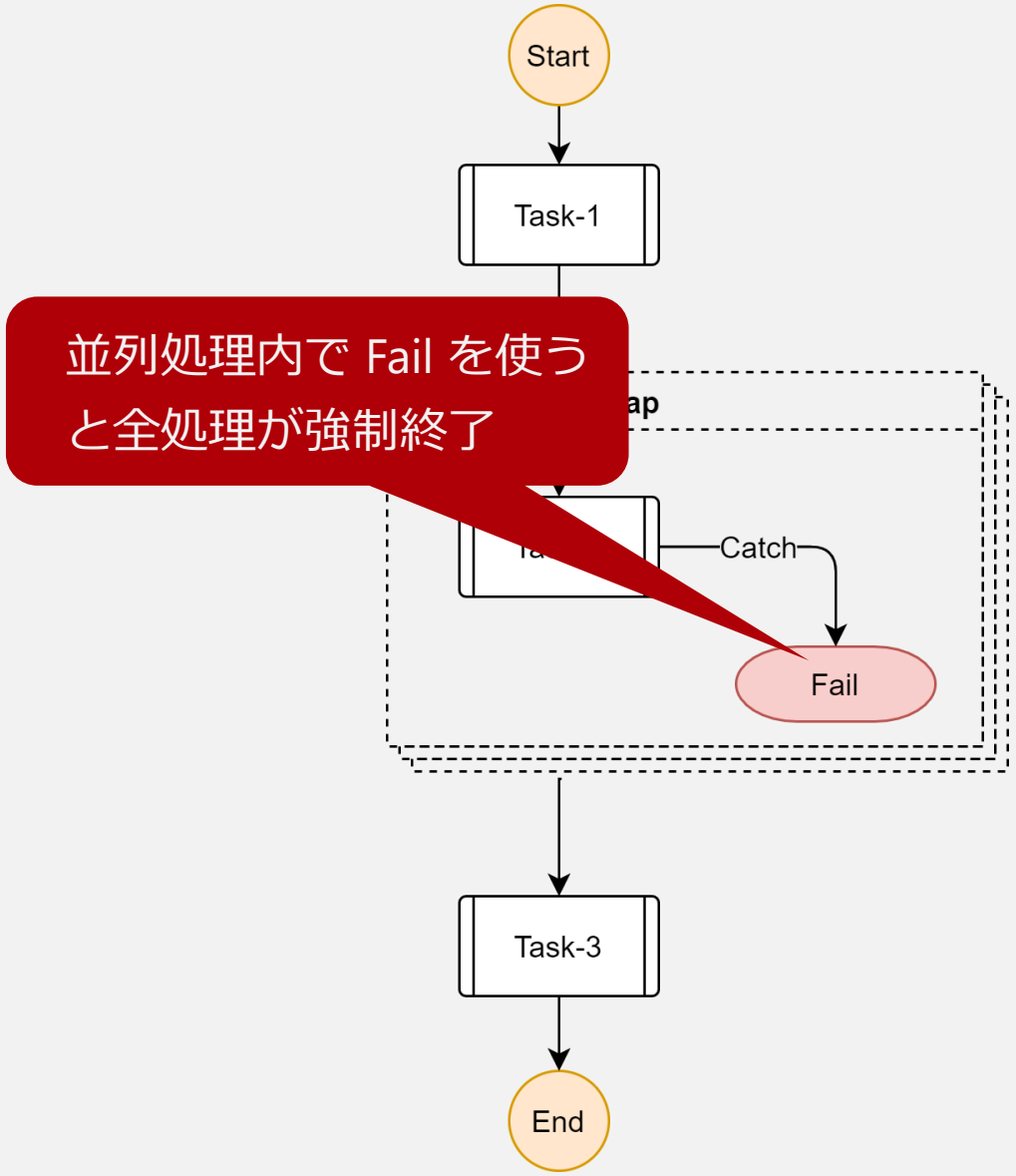
推論 Workflow

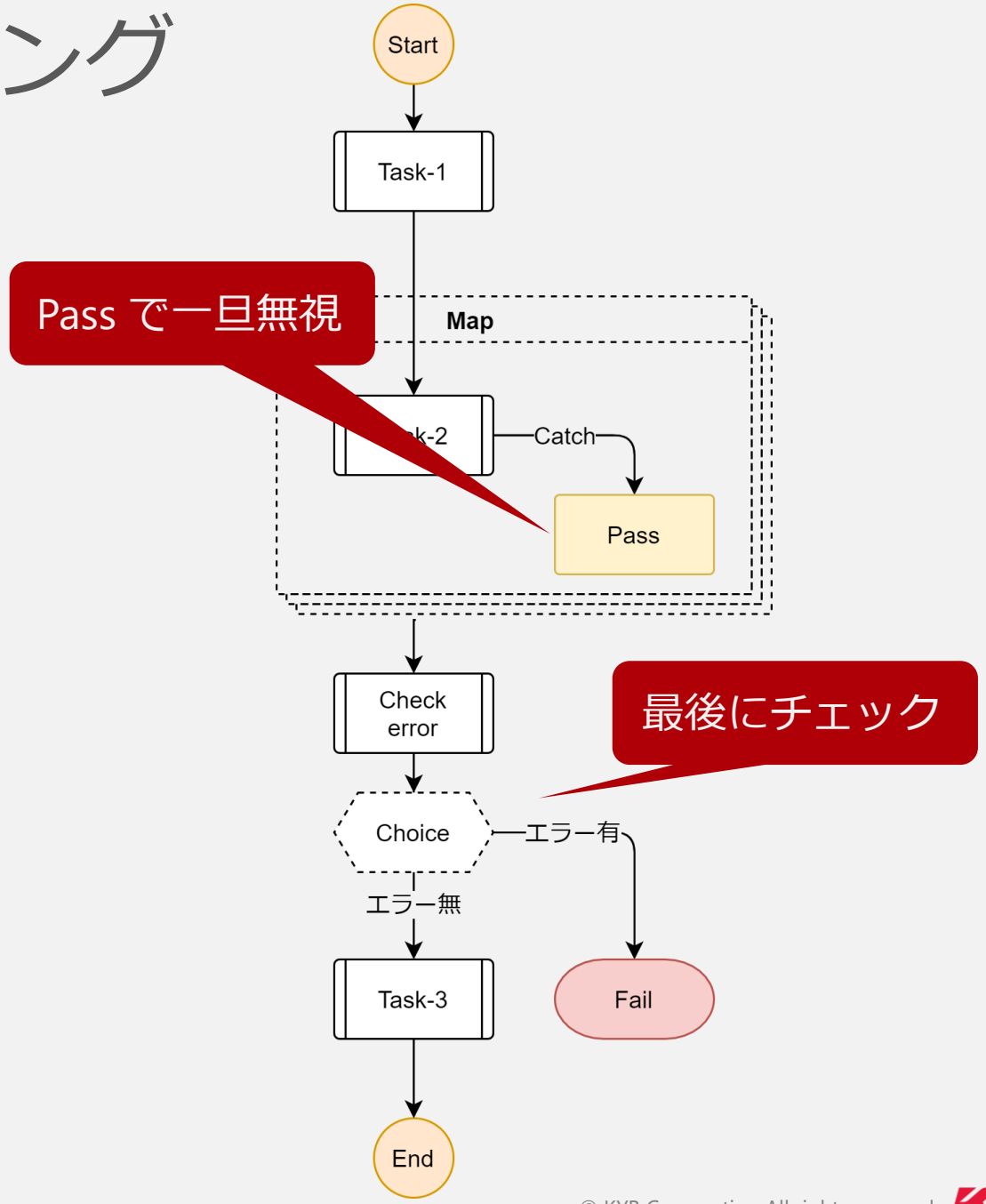


※ 説明のために大幅に簡素化しています

© KYB Corporation All rights reserved.

並列処理のエラーハンドリング





© KYB Corporation All rights reserved.

04

Conclusion

01 About KYB
- 会社紹介
- KYB の IT システム開発

02 IoT-Platform
- 概要と課題
- 予知保全システム

03 Amazon SageMaker の活用
- Amazon SageMaker を使う上での工夫
- ML Workflows 構築における工夫

04 Conclusion

Amazon SageMaker まとめ

- 既に **我々にとって必須** のサービス
  - 今後は **他部署にも展開** していく計画

- まだまだ進化しており、**今後も非常に楽しみ**

Amazon SageMaker

© KYB Corporation All rights reserved.

KYB の取り組みまとめ

- KYBでは **IoT-Platform の構築** を進めています
  - 外注丸投げではなく、**内製を中心** に取り組んでいます

- **製品領域** への適用も進めている最中です

© KYB Corporation All rights reserved.

We’re hiring!

© KYB Corporation All rights reserved.

Thank you!

内藤 孝昌

naitou-tak@kyb.co.jp

aws SUMMIT

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

KYB