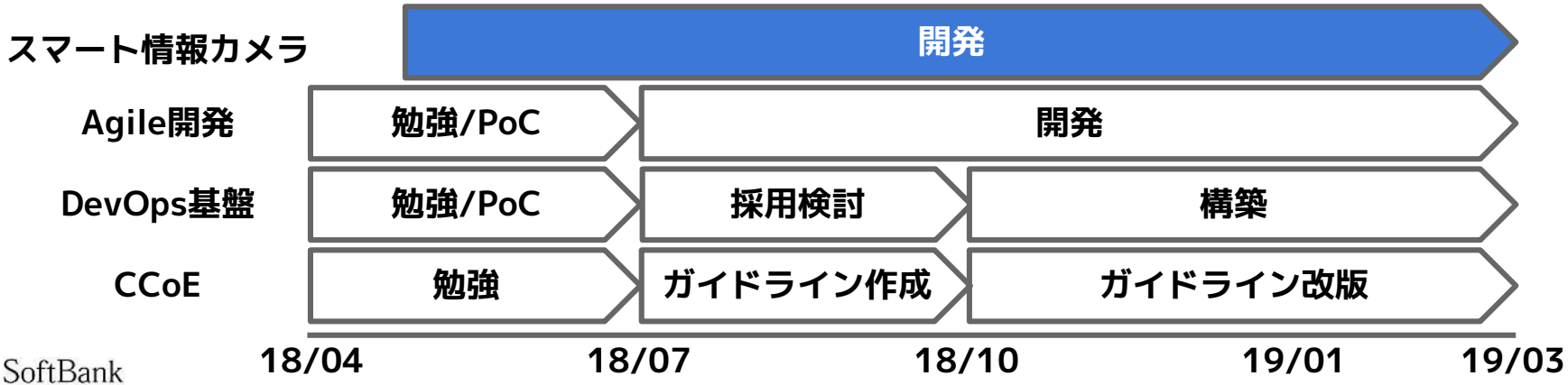
ソフトバンクの携帯基地局を活用した映像配信システムとAgile 開発基盤における AWS 活用事例

テクノロジーユニット
モバイル技術統括
アプリケーション技術本部

ご紹介内容について

おおまかな関係

スケジュール



映像

テクノロジーユニット
モバイル技術統括
アプリケーション技術本部
田中 俊輔

映像配信システムのご紹介

スマート情報カメラ

基地局設置カメラの映像コンテンツ配信サービス

映像プラットフォーム
ストリーミングアーカイブ

ブラウザー・デコーダー経由で
映像取得

全国各地に
設置している
基地局からの映像配信

カメラ / 回線 /
プラットフォーム /
保守のオールインワン
サービス

ウェブブラウザー上で
カメラの各種操作
が可能

専有パターン・
共有パターンの
2通りが選択可能

河川

・氾濫
・浸水

山

・噴火

沿岸

・津波

道路

・緊急車両
・渋滞

プレスリリース：https://www.softbank.jp/corp/news/press/sbkk/2018/20181109_01/

ソフトバンクイノベンチャーがきっかけ

基地局リソース、
保守ノウハウを
有効活用したサービスを。

映像ってこんなPJで始まりました

各テレビ局などを対象に高品質映像を流したい（最大7Mbps30fps）

最大カメラ台数もそれなり（数字出せませんでした）

基地局と閉域網でつないで提供するサービス

曖昧な要件多数

期限あり

オンプレかクラウドか…

各テレビ局などを対象に高品質映像を流したい（最大7Mbps30fps）

カメラ台数もそれなり（数字出せませんでした）

⇒この規模の設備を構築するのはかなり大変…

期限あり

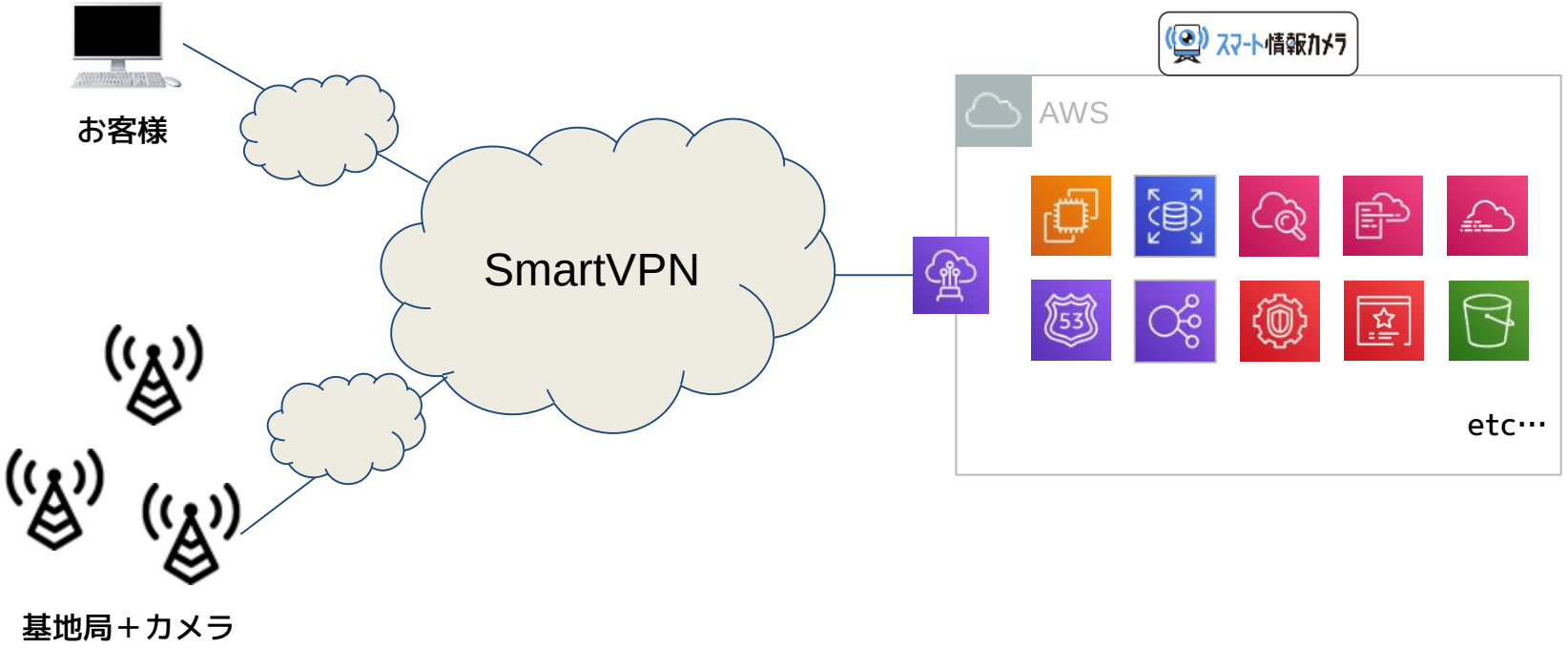
曖昧な要件多数

⇒都度要件変更に対応しつつ、短納期。絶対無理

基地局と閉域網でつないで提供するサービス

アマゾン ウェブ サービス（AWS）を採用

構成概要



映像は何をするにも制限に注意

10G回線DX接続

⇒DXは10G。しかも増設プランも検討要

NW、ストレージスループットの見積

⇒各スループットやベースラインなどの公開有無

公開していても設計が難しい

非公開は使いづらい（主にS3）

閉域の利用制限は多い

⇒Amazon Elastic Transcoder
AWS Elemental MediaLive
など魅力的なサービスが展開されているが、
閉域では使えない

オンプレスタンダードの打破

⇒オンプレベースの社内ルールや考え方に苦慮

-クラウドの理解

-開発/運用/セキュリティチェックシート

-運用受入基準

フォーキャスト ブレ への即時対応

⇒削除/構築が簡単なので、
フォーキャストに即して即時にサーバ増減可能

（前提として閉域配信サーバをEC2/EBSで用意したので、1台の値段が高額に…）

試験環境もクラウドなら即時準備可能

⇒それなりのカメラが台数必要であったが
実カメラの準備はかなり大変

⇒疑似トラヒックサーバをささっと準備！

CCoE

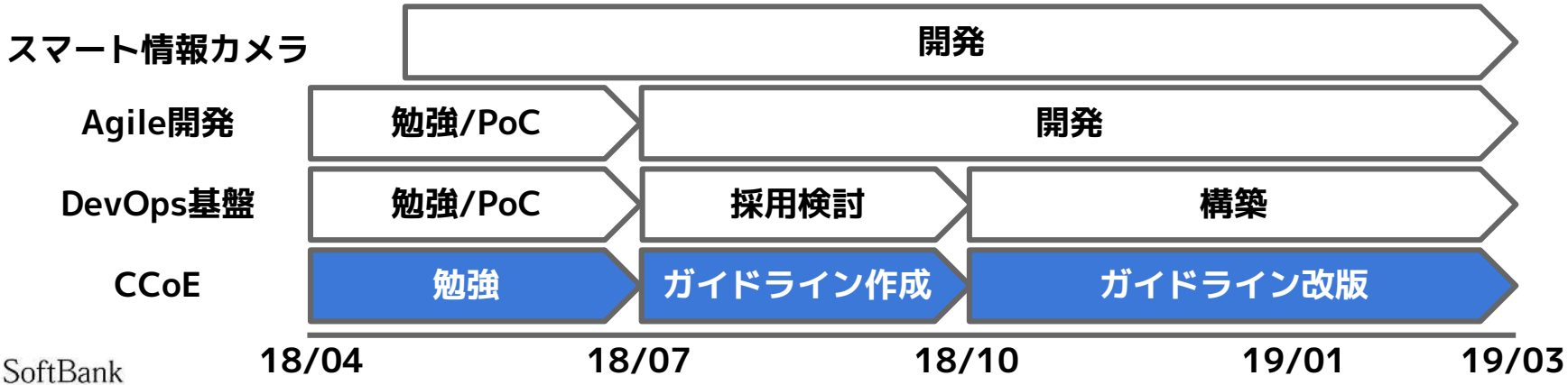
テクノロジーユニット
モバイル技術統括
アプリケーション技術本部

茶園 篤

ご紹介内容について

おおまかな関係

スケジュール



最初は知らないことばかりで...

ルール未整備

ほぼ知識ゼロ

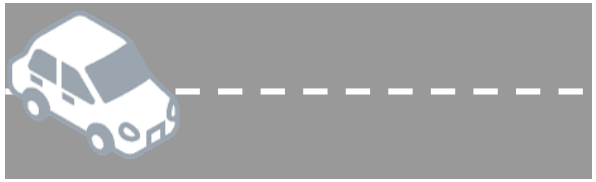
体制も未整備

どこへ進むべきか？

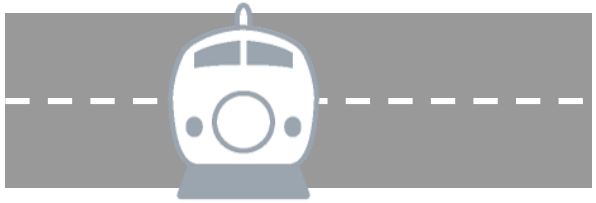
ルール未整備	統一的な開発/運用で楽に
ほぼ知識ゼロ	知識習得＋知の共有ループ
体制も未整備	ルール＋推進する心意気

一気に並行で駆け抜けました

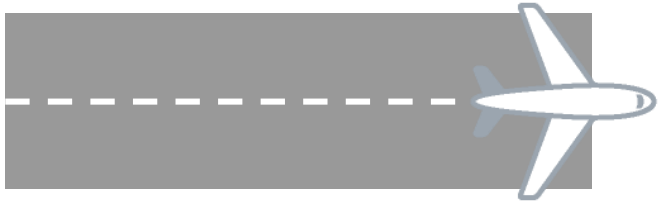
ルール未整備



ほぼ知識ゼロ



体制も未整備



ルール＋標準環境整備

すぐに利用できる環境も提供

学習＋Toolで知識共有

コンサルの皆さん、Thanks!

ガバナンス”心”統一

クラウド使う!という心意気

まあ、今となってみれば

18/04

コツコツ…

18/10

本部標準環境
リリース！

19/02

そして今

どんどん使います！

aws CERTIFIED
Solutions Architect - Associate

何とか歩いてます

DevOps

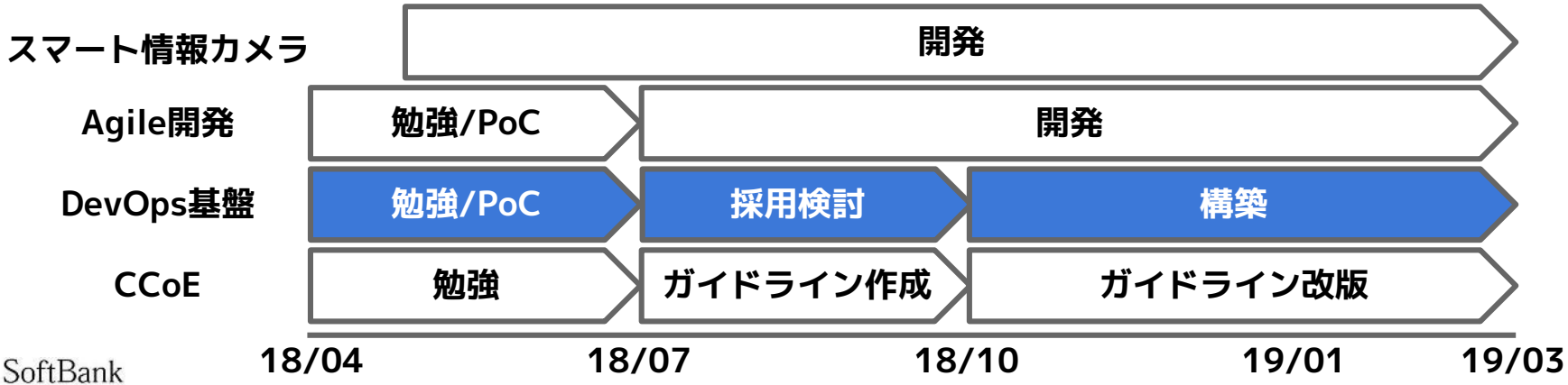
テクノロジーユニット
モバイル技術統括
アプリケーション技術本部

山根 武信

ご紹介内容について

おおまかな関係

スケジュール



アプリケーション開発の課題

環境構築などアプリ開発以外の作業負荷

開発から商用環境へのスムーズな移行
可用性（継続的な稼働）の向上

監視設定・統計解析の作業負荷

DevOps実践環境導入

Application Engineering Division
DevOps Kiban

CI/CD
パイプライン

アプリ実行基盤

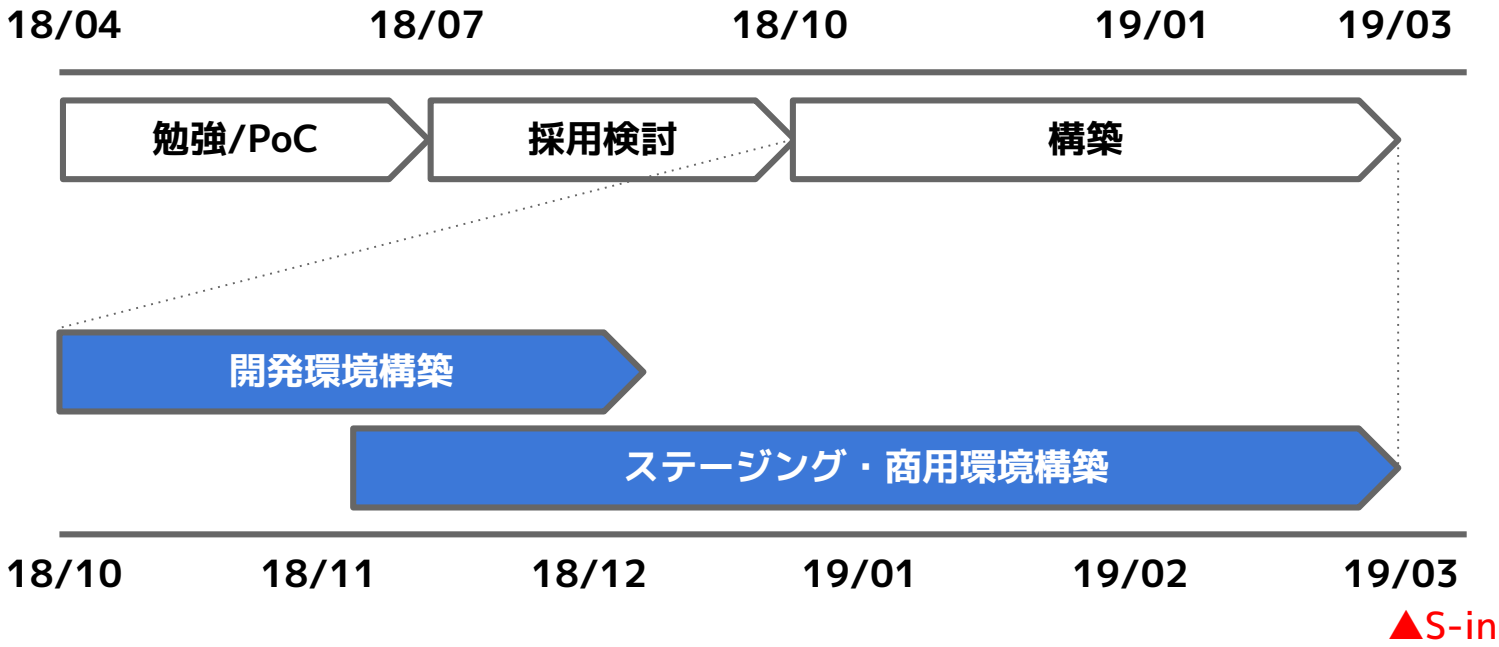
監視・運用基盤

作業の自動化

作業の省力化

可用性向上

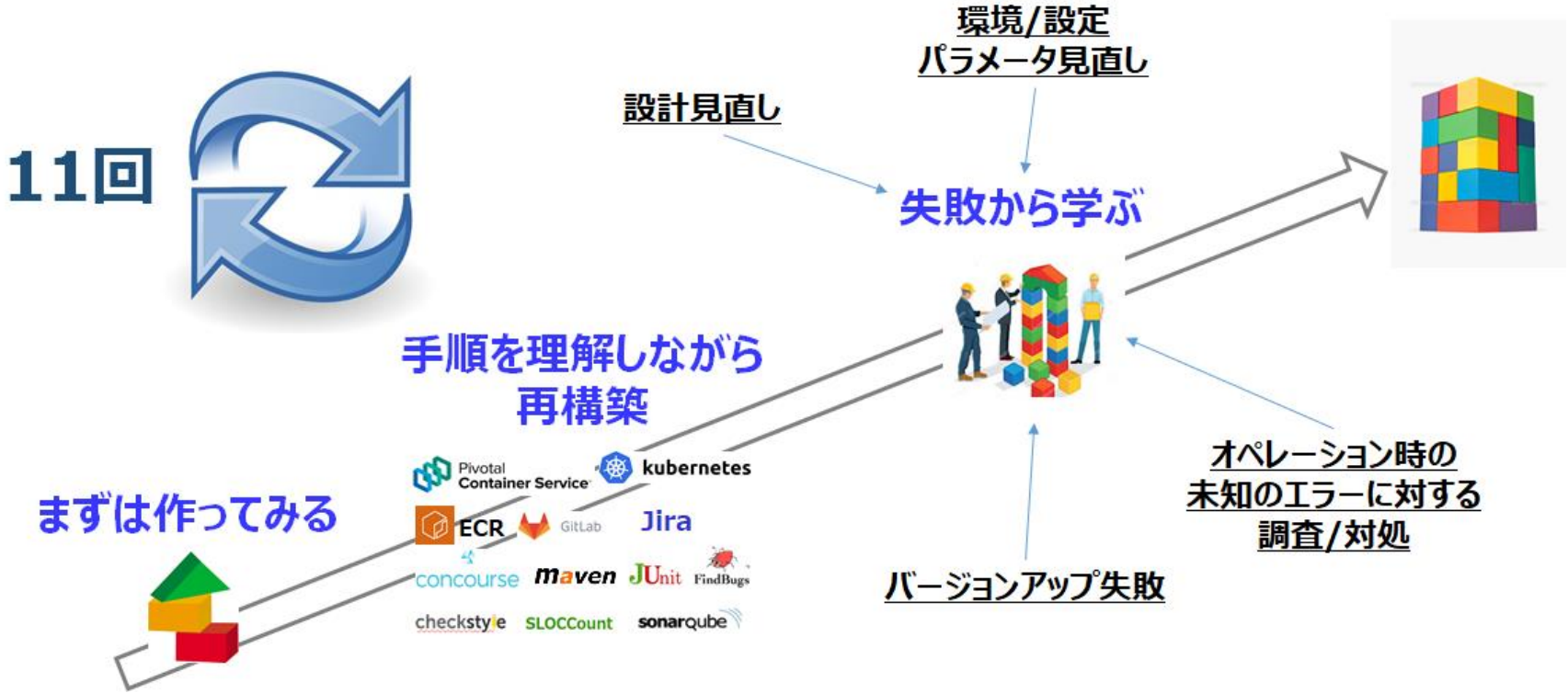
当初のスケジュール



開発環境で得たナレッジを活用のつもりだった．．．

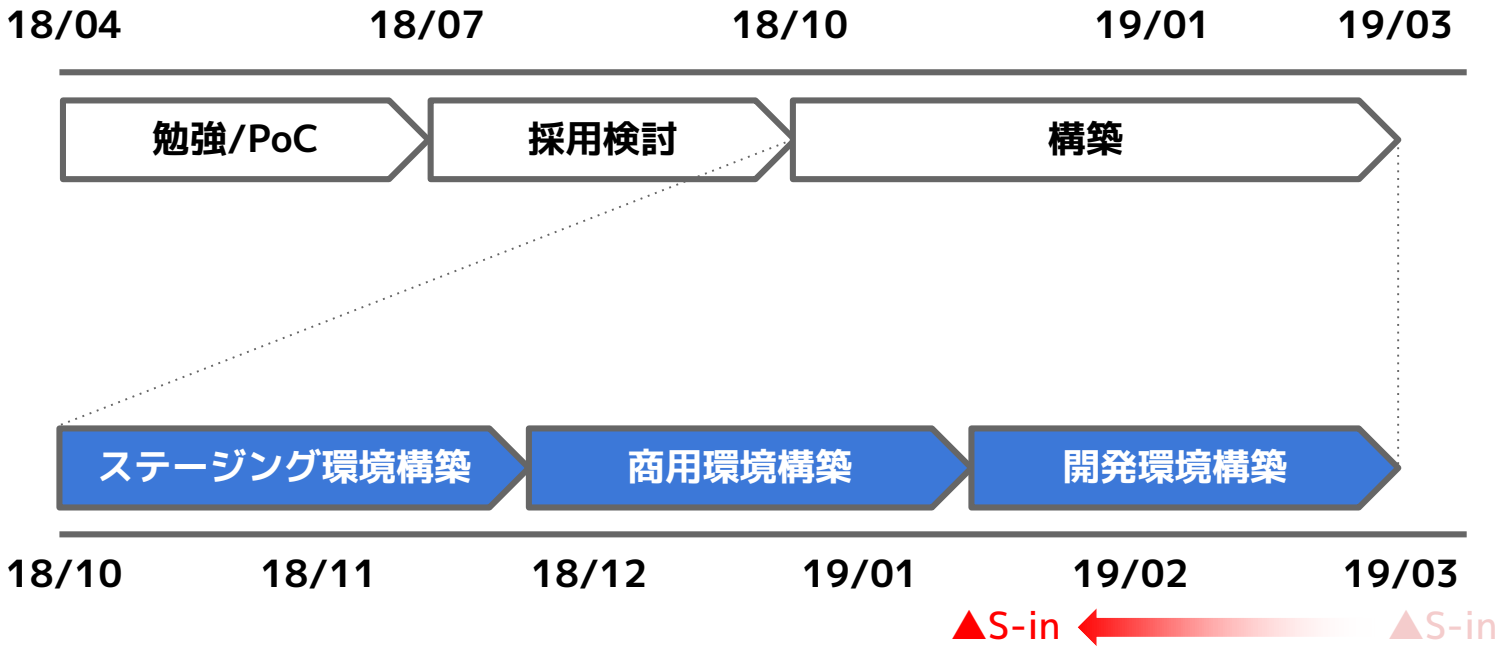
やり方を変えてみた

Try & Error 繰り返し



現物で作りながら設計

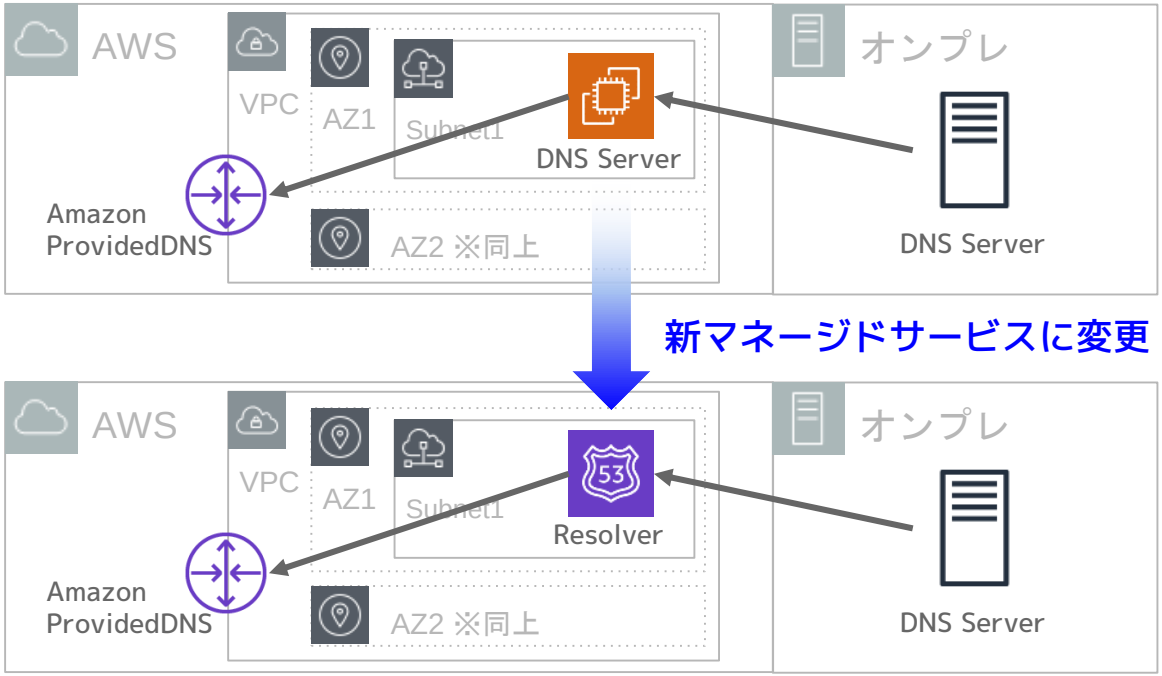
実際のスケジュール



商用環境提供1.5ヶ月前倒し

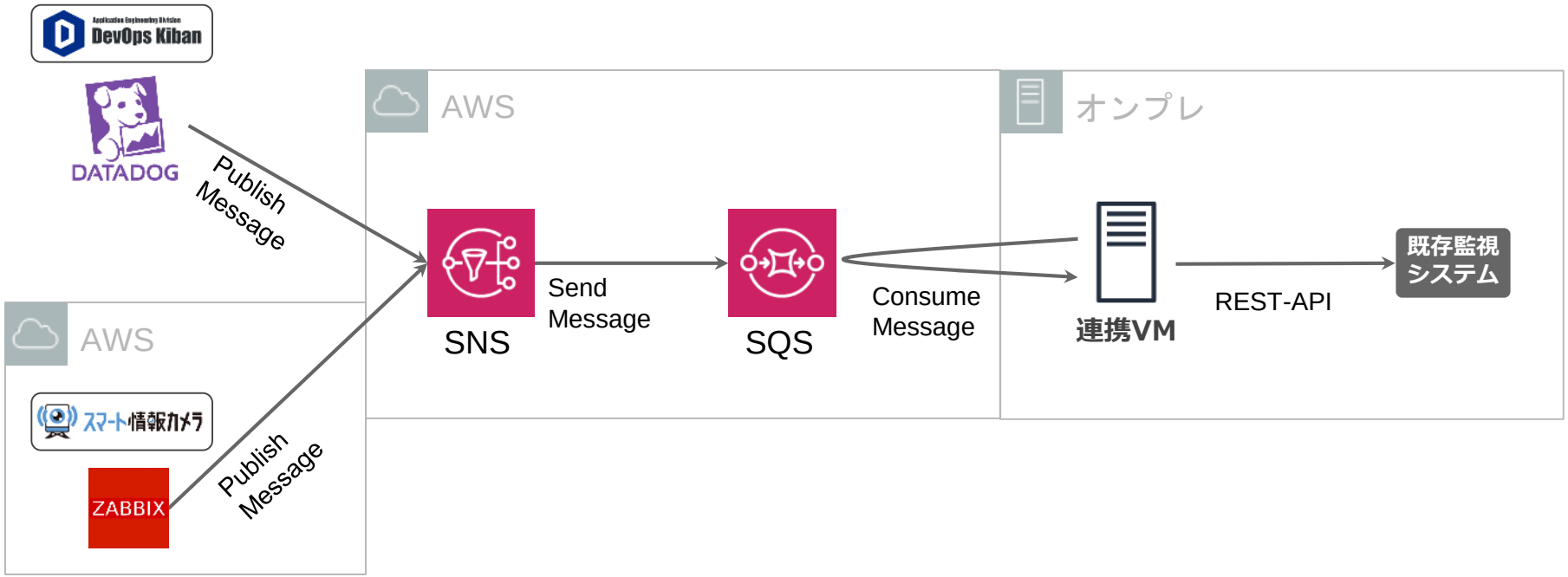
AWSで良かったこと

Route53 Resolver



DNS Serverの構築・運用が不要に

アラート連携開発



マネージドサービス利用で開発工数減

アジャイル開発

テクノロジーユニット
モバイル技術統括
アプリケーション技術本部
関 修康

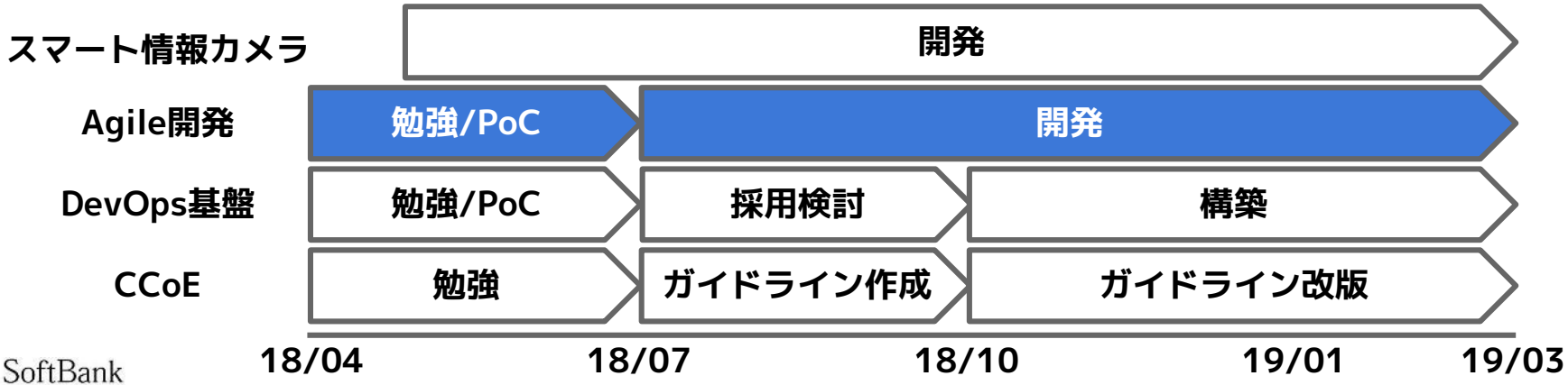
お伝えすること

- アジャイル開発の何が良かったか？
- AWSをどの様に活用したのか？

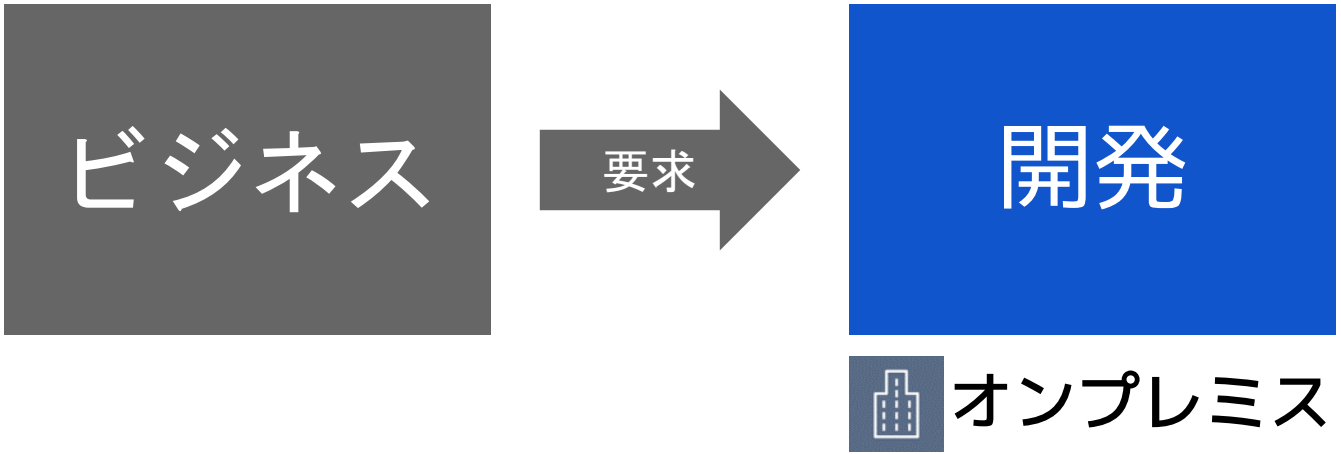
ご紹介内容について

おおまかな関係

スケジュール



ビジネス側から要件をヒアリングして開発する立場

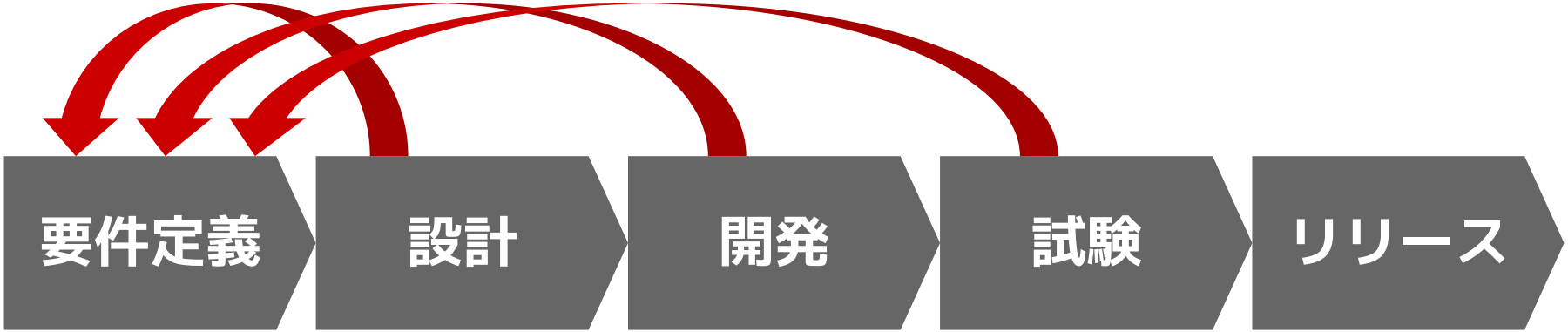


今までのやり方はオンプレ＋ウォーターフォール

リリースまで最短６ヶ月

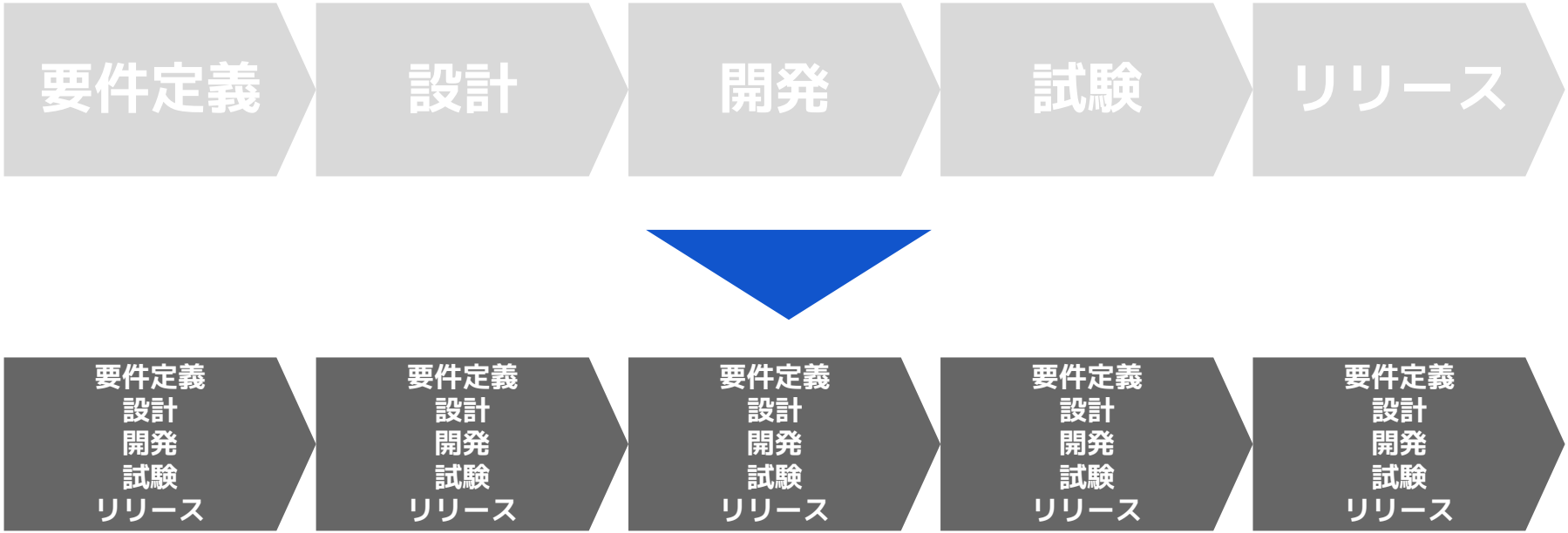
ウォーターフォール型の問題

仕様変更　→　再設計　→　再開発　→　再試験・・・



仕様変更に弱い

新しい考え方へシフト



アジャイルチャレンジ

アジャイルで開発した結果

- 開発期間１０ヶ月でサービスイン

あれ？　遅くない？

- １ヶ月目　ビジネス側で現物確認が可能
- ８ヶ月目　仕様FIX

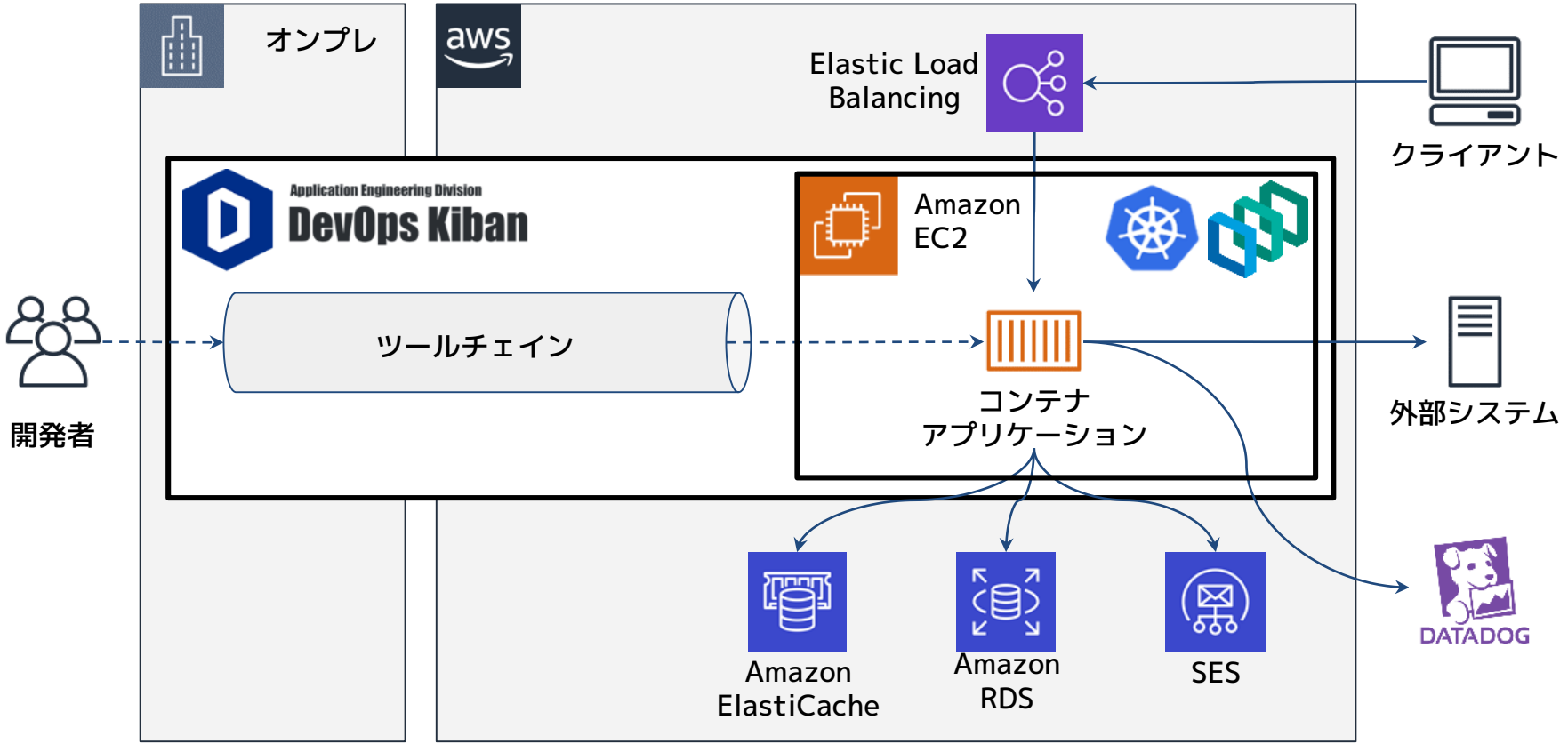
アジャイルのスピードと柔軟さ

アジャイル開発のさらなる柔軟さ

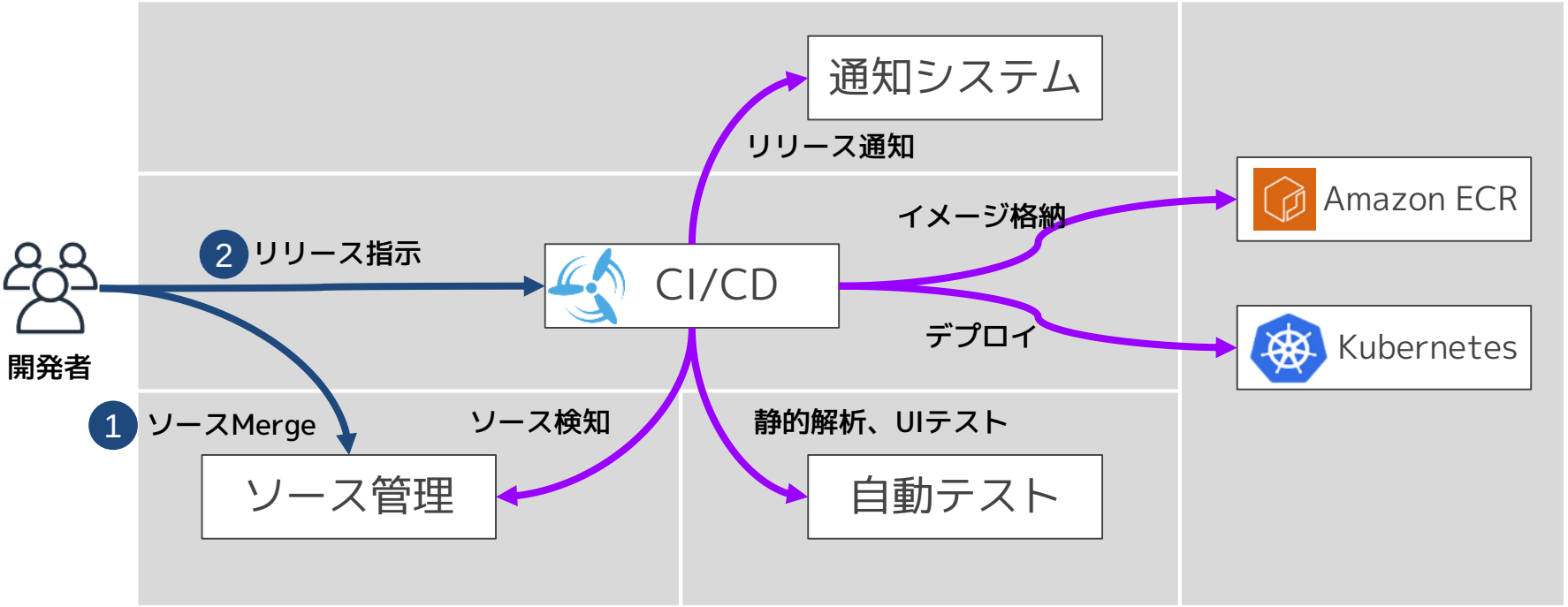
- ３ヶ月目　オンプレ→AWSに移行完了
- ７ヶ月目　AWS＋コンテナに移行完了

AWSでインフラもアジャイル

アジャイル開発したアプリケーションの構成



ツールチェイン



オンプレとAWSの融合

ツールチェインの効果

単体結合・ＵＩテスト

・３日　→　30分

・テストケース数

3200件／日

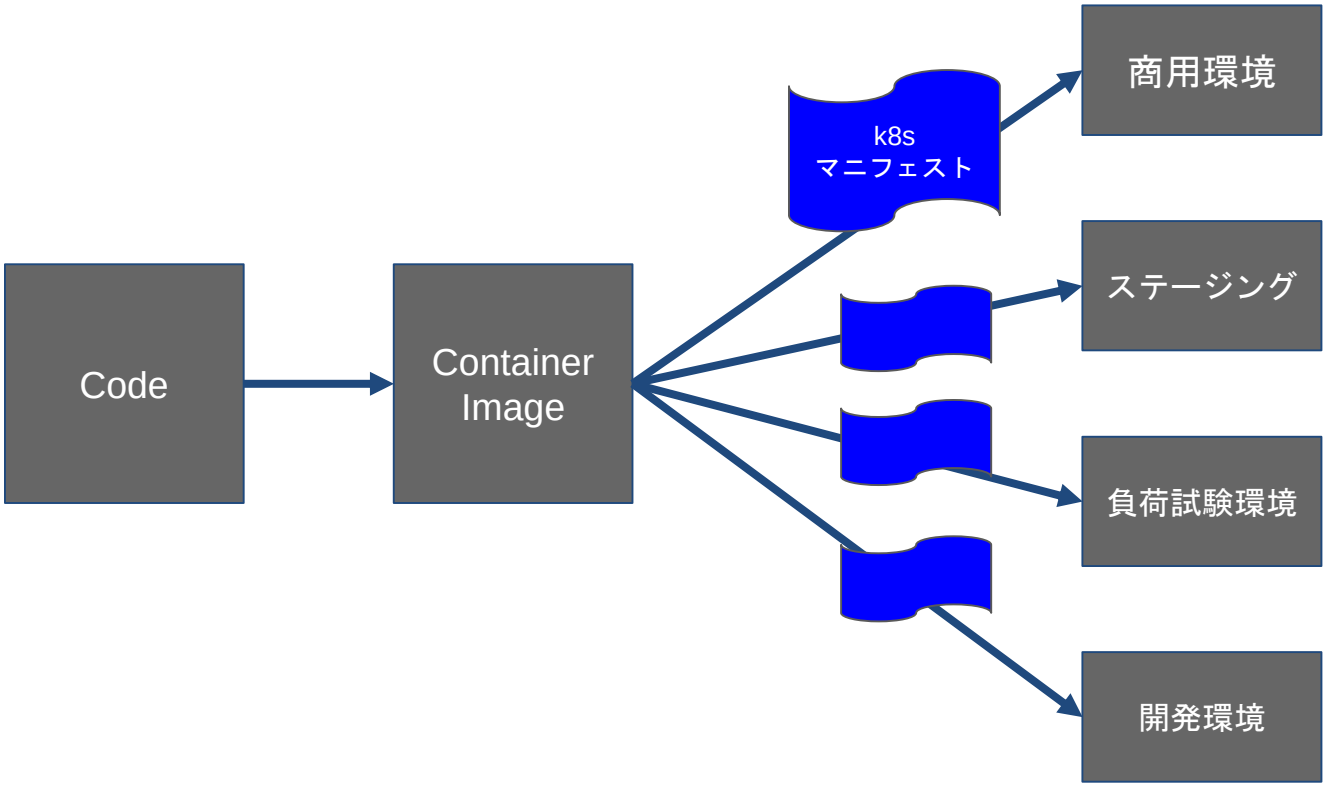
ステージングデプロイ

・１日　→　30分

商用環境デプロイ

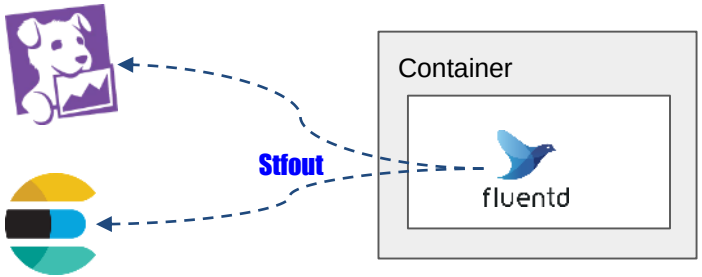
・１日　→　3時間

環境設定はk8sマニフェスト化



コンテナアプリケーションのログの実装

- アプリケーション側ではログを管理しない
  - コンテナはローカルストレージを使用しない。
  - ログ書き込み、ログローテーションも実装しない。
  - Fluentdなどのログ収集ツールを利用し、標準出力をElasticSearchやDatadogなどのログ解析サービスに転送



- 各プロジェクトへのコスト按分
  - オンプレミス時代
    - サーバー10台
    - → じゃあ1,000万円（例）
  - Kubernetesを使った場合
    - Pod 6台だったり、10台だったり（増減）
    - CPU 200m (request)だったり 2000m (limit)だったり
    - → すぐに計算できない

AWSをどの様に活用したのか？

- 無理に利用しない
- 部品ごとに利用できそうであれば利用する
- 同等機能のマネージドサービスがあれば乗り換える

AWSサービスのつまみ食い

アジャイルまとめ

- アジャイル開発の何が良かったか？

アジャイルのスピードと柔軟さ

- AWSをどの様に活用したのか？

AWSサービスのつまみ食い

AWSさんの伴走でクラウド活用のメリット実感

Agile開発
サービスのつまみ食い

スマート情報カメラ
最悪どうにでもなる！

DevOps基盤
「まずは作る」でインフラ提供の前倒し

CCoE
知識＋開発・運用環境の集約＋推進する心

ご清聴ありがとうございました。