4年間の取り組みで実現したコンテナ技術を活用したスケーラブルなインフラ構築とコスト削減

コンテナSummit 2023

2023.5.31

株式会社MIXI Vantageスタジオ みてねプロダクト開発部 基盤開発グループ
清水 勲

©MIXI

About Me

清水 勲 @isaoshimizu

2011年〜 株式会社ミクシィ（現MIXI）

- 2011年8月〜 SNS「mixi」運用エンジニア
- 2014年4月〜 モンスターストライク SRE
- 2018年2月〜 家族アルバム みてね SRE
- 2022年1月〜 SREグループ マネージャー
- 2023年4月〜 基盤開発グループ マネージャー

- 週末は社会人吹奏楽団での活動（楽団長、トロンボーン約30年、たまに指揮者）。
  キャンプとクラフトビールが好き。

©MIXI

MIXI GROUPの事業領域

エンタメ×テクノロジーの力で、
世界のコミュニケーションを豊かにする
サービスの創出を目指しています。

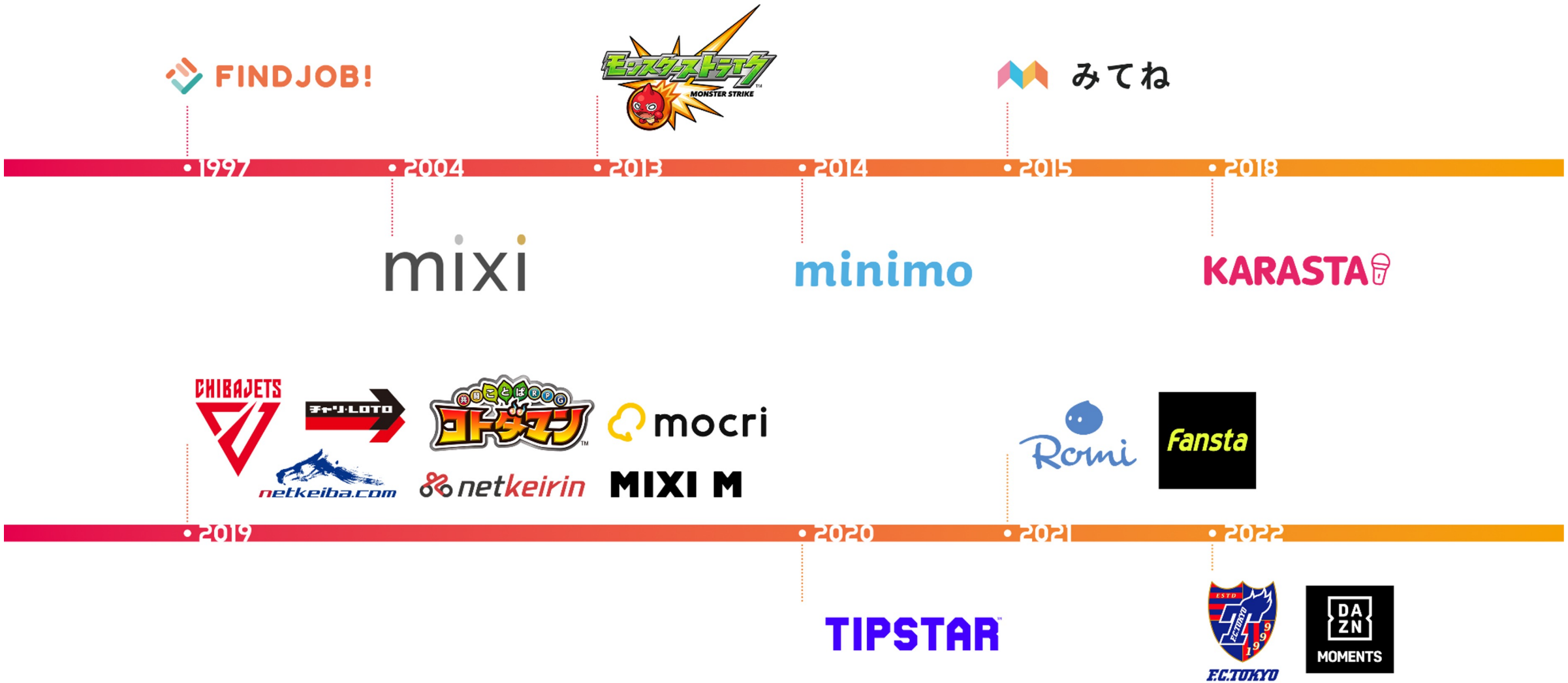
MIXI GROUPは、友だちや家族といった
親しい人々とのコミュニケーションを活性化する、
様々なサービスを開発・提供しています。

現在、スポーツ・ライフスタイル・デジタルエンターテインメント
の3つの領域で事業を展開しており、
それぞれの主な事業内容は右の通りです。

また、近年の投資活動の拡大と重要性を勘案し、
FY2023からはスタートアップやファンド出資等の投資活動を事業化しました。

©MIXI

主な事業・サービスのリリース年表

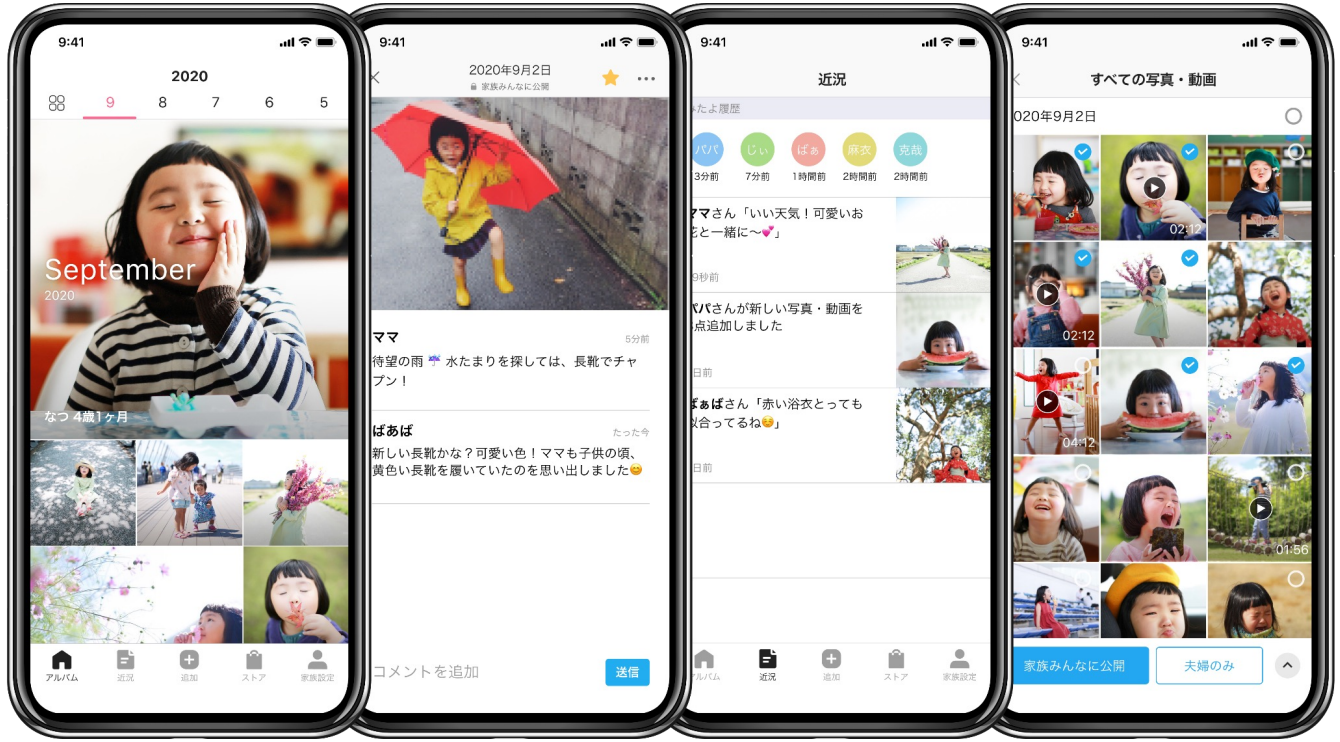


©MIXI

家族アルバム みてね

©MIXI

家族アルバム みてね



パパ・ママが撮った子どもの写真や動画を、祖父母や親戚など
招待した家族だけに簡単に共有できる写真・動画共有アプリ

商品の例

フォトブック

写真プリント

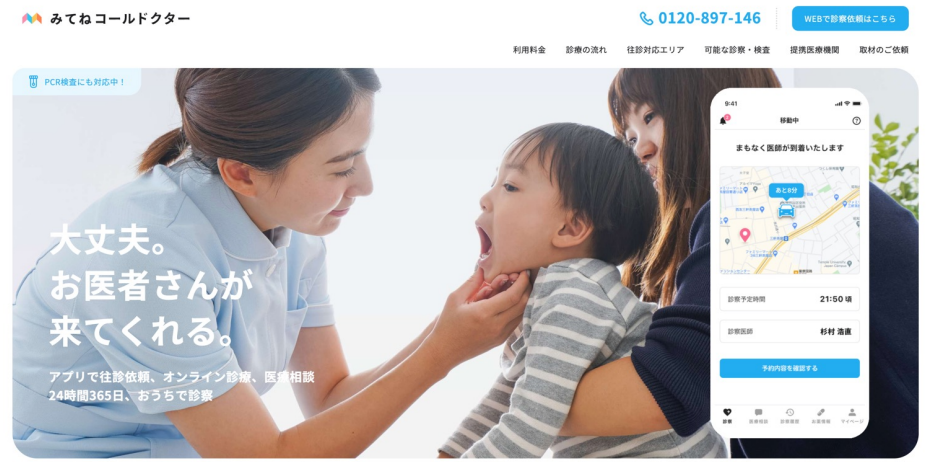
©MIXI

家族アルバム みてね

みてねみまもりGPS

みてね出張撮影

みてね年賀状

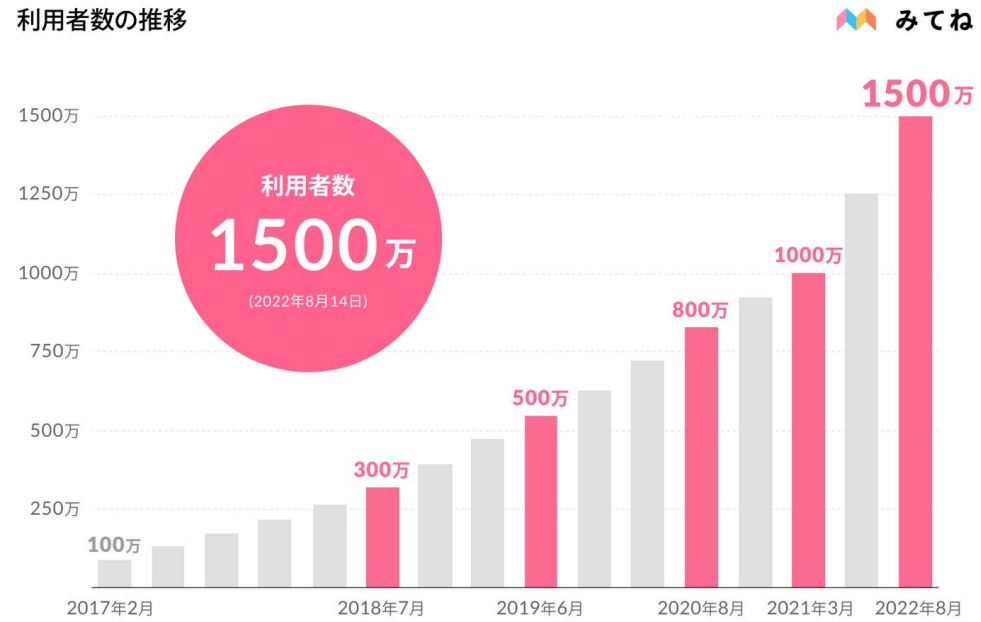


みてねコールドクター

©MIXI

家族アルバム みてね

- 2015年4月リリース
- 現在は7言語・175の国と地域でサービスを提供
- 海外では「FamilyAlbum」という名称で展開中
- 2022年8月14日に利用者数が1,500万人※1 を突破
- 日本国内ではママやパパの半数となる47.1％の方※2がご利用



※1 iOS・Android™ アプリ登録者数、ブラウザ版登録者数の合計
※2「みてね」登録時に入力されたお子さまの誕生日と厚生労働省発表「人口動態統計」から算出。2022年8月時点で47.1％

©MIXI

約４年間に渡るコンテナ移行の取り組みを紹介

（家族アルバム みてねにおける事例）

©MIXI

コンテナ以前のインフラにおける課題

©MIXI

コンテナ以前のインフラ

- VM のオーケストレーション（AWS OpsWorksによる）
- Chef、Cookbookによるインフラの構築・デプロイ自動化
- オートスケール（時刻ベース、ロードベースなど）
- VMのモニタリング機能の統合
- ユーザー管理、SSHキーの管理機能

などなど。構築当初は効率よく開発、デプロイができていた。

©MIXI

数年運用していくうちにさまざまな課題に直面

©MIXI

コンテナ移行前の課題

デプロイ

- ChefのCookbookの管理、コードの依存関係が複雑に
- デプロイ速度の問題（Chefの仕組み上の問題）

パフォーマンス

- オートスケールが遅い（ピーク帯に問題が起きやすい）

開発・運用

- SSHしてホスト内で作業をしたくない（VM間の差分が生まれやすい）
- Docker使って開発したいという声（＝本番でもDocker使いたい）

コスト

- オンデマンドインスタンス前提でコスト高

©MIXI

コンテナへの期待

©MIXI

コンテナへの期待

- **OSに依存せず、イメージ化によるポータビリティ**
  - どの環境でも簡単にイメージを配置できる
- **オートスケール時の起動の速さ**
  - コンテナの配置の速さ、イメージのキャッシュ再利用
- **ディスポーザブルなインフラの実現**
  - SSHする機会を減らす
  - 中断に強い（スポットインスタンスを活用しやすい）
- **従来のVMよりも効率的なハードウェアリソースの活用**
  - ボトルネックが少ない、空きリソースを活用できる
- **さまざまなサービスがコンテナを前提としてきている**
  - AWSの例
    - Amazon EKS、Amazon ECS、AWS App Runner、AWS CodeBuild、AWS Batch、Amazon SageMaker

©MIXI

VM運用とコンテナ運用は何が違うのか

©MIXI

VM運用とコンテナ運用

	VM	コンテナ
デプロイ	VM内にコードを配置 プロセスの再起動	コンテナイメージの配置 コンテナの再起動
リソース管理	VMの台数、タイプを調整	コンテナの個数、リソースを調整
ソフトウェアアップデート	VMイメージの置き換え VM内で直接アップデート	コンテナイメージの修正・ビルド
インフラモニタリング	VM内にエージェントを配置	ホストにエージェントを配置
アプリケーションモニタリング	APMを利用	APMを利用

©MIXI

どうやってコンテナへ移行したか

©MIXI

コンテナオーケストレーションの選定

まずコンテナを運用するにはオーケストレーションサービスを選ぶところから

組織や課題に合ったサービスを選ぶのがオススメ

例： Amazon ECS、Amazon EKS、Google Cloud Run、Google Kubernetes Engine

コンテナオーケストレーションサービスが提供するもの（一例）

- デプロイメント
- スケジューリング
- 可用性
- トラフィックルーティング
- リソース管理

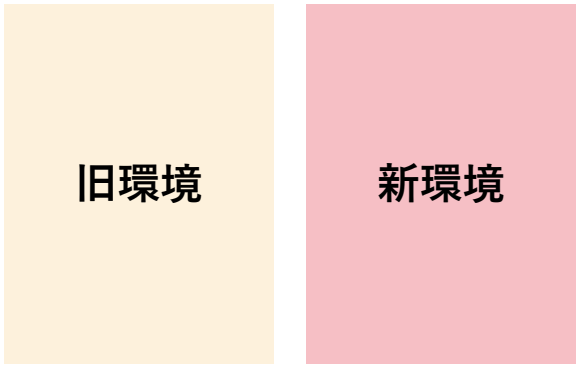
など

家族アルバム みてねでは「Amazon EKS」を採用

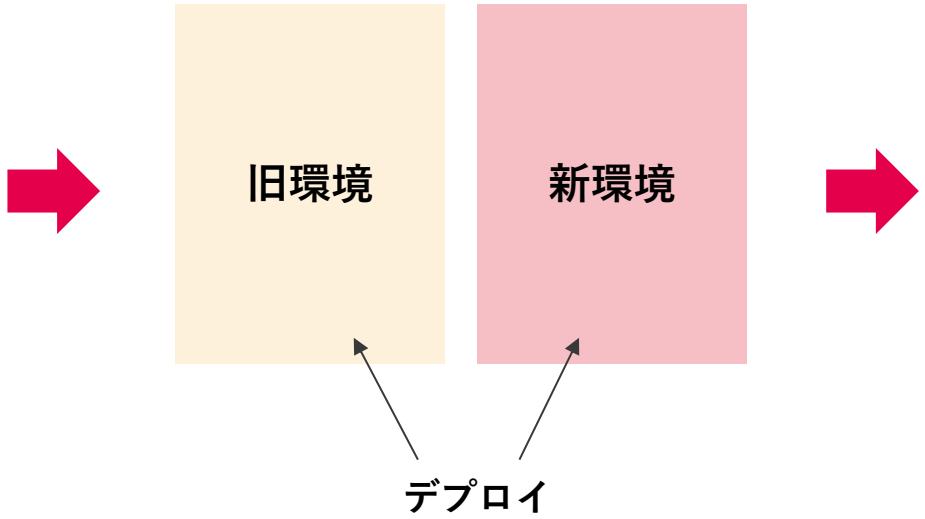
©MIXI

コンテナ移行の流れ

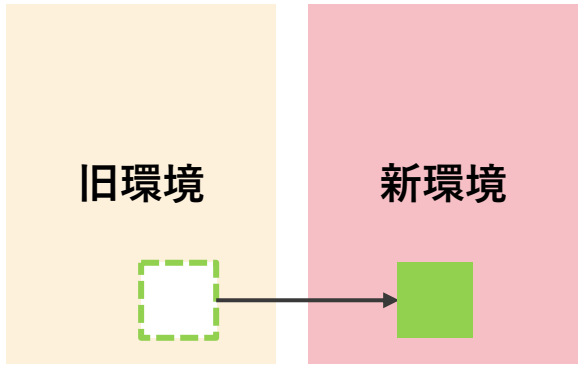
1. 新環境を構成



2. 両方の環境にデプロイできるように

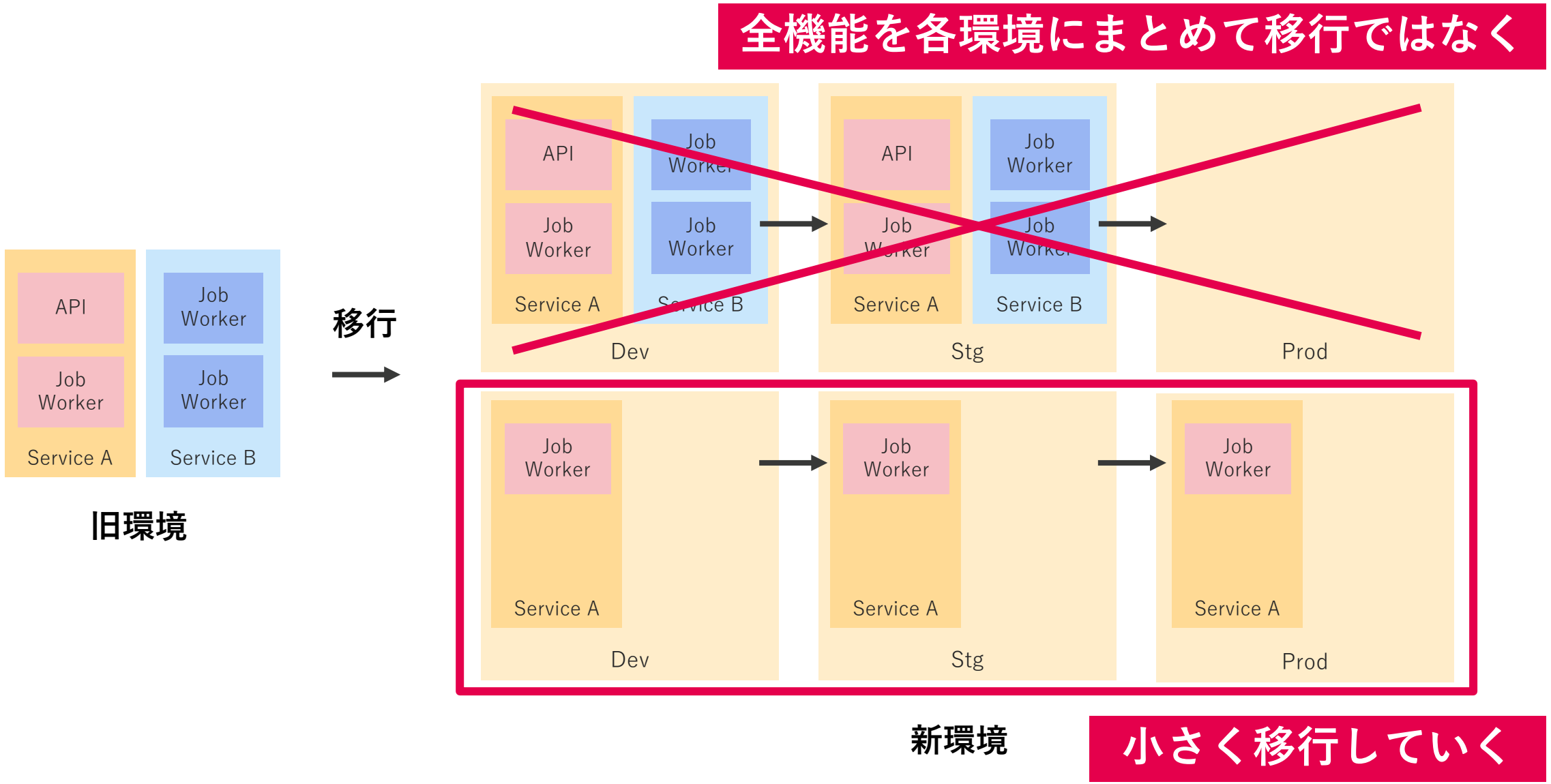


3. 機能ごとに徐々に移行
移行を終えたものは旧環境から削除



©MIXI

移行時のポイント



©MIXI

長年にわたる試行錯誤の結果、無事移行が完了！

©MIXI

コンテナ移行による効果はどうだったか

©MIXI

コンテナによる効果

- **オートスケールの高速化**
  - 急なトラフィックの変化に対してもスピーディにスケールできるように
- **スポットインスタンスのフル活用**
  - 9割以上のインスタンスがスポットになり**大幅なコスト削減**
  - 普段使っているスポットインスタンスはオンデマンド比較で**60〜70%割安**
- **構成がシンプルに**
  - 構成がDockerfileに集約された
- **さまざまな環境でも同じ構成を作りやすくなった**
  - イメージの再利用性

©MIXI

コンテナ移行時のトラブルに備えるために

©MIXI

移行時のトラブルに備える

新たな環境においてどんなトラブルが起きるかわからない

トラブルが起きた際に問題の原因調査をスムーズに行いたい

移行前にオブザーバビリティを整備しておく

©MIXI

移行時に採用したオブザーバビリティツール

- **アプリケーションモニタリング**
  - New Relic APM
- **インフラモニタリング**
  - New Relic Infrastructure
  - Prometheus
  - Amazon CloudWatch
  - Grafana

©MIXI

アプリケーションのモニタリング

- 従来から**New Relic APM**（Application Performance Monitoring）を使ってきた
- APMではアプリケーションのパフォーマンス、エラーの状況が詳細にわかる
- **VMでもコンテナでも動くアプリケーションは同じ**
  - APMはどちらの環境でも漏れなく動作して、問題があれば原因となる情報を提供する

©MIXI

インフラのモニタリング

- インフラのモニタリングはVMとコンテナで構成やモニタリング対象が異なる
- VMよりコンテナの方が構成するコンポーネントの階層が多くなり複雑性が高くなる
- New Relic Infrastructureのように網羅性の高いツールの活用は効果的
- 移行時においてはロードバランサーやデータベースのモニタリングも重要
  - 前段でのトラフィックやエラーレートの変化に気づく
- OSSも併用してコスト最適化するのも一つの選択肢

©MIXI

オブザーバビリティの重要性

- **未知の問題**に対して原因の調査ができる
- システムの移行など大きな変更を伴う場合、あらかじめ狙ったメトリクスのモニタリングだと問題に気づけないことがある
- 日々の開発、運用においても役に立つことが多い
  - 開発段階からAPMを通じてパフォーマンスの問題に気づくことができる
- システムに対する熟練度に関わらず、より多くのメンバーが分析できるようになる

©MIXI

コンテナ移行時のオブザーバビリティはとても大事

©MIXI

まとめ

©MIXI

まとめ

- 移行によってコンテナのメリットは十分に享受することができた
  - 運用コストもインフラコストも大きく削減することができた
- システムの移行作業は小さく行い、早くフィードバックを得られるようなサイクルを回すことが大事

- オブザーバビリティを整えておくと、移行時のトラブル解決が早くなる＝ユーザーへの影響を最小限にできる
  - コンテナ環境においてもNew Relic APMを重宝した
  - オブザーバビリティはトラブル解決だけでなく日々の開発や運用にも役立つ、熟練度に関わらずより多くのメンバーが分析できるようになる

©MIXI