

投稿日 2024/03/29

## 新規事業開発のための技術選定

我々は、日々進化し多様化する技術の中から、新規事業開発に適した技術を選定する必要があります。技術は単なるツールにとどまらず、事業の成功を形作る基盤になるため、採用する技術は事業の価値と目的に基づいて決定されるべきです。

ここでは、新規事業のエンジニアリングにおける技術選定の考え方を、以下の3つの観点から示します。

- 利用する技術の数を最小限にすること
- 「作らないエンジニアリング」を支援すること
- インフラを自社で所有しないこと

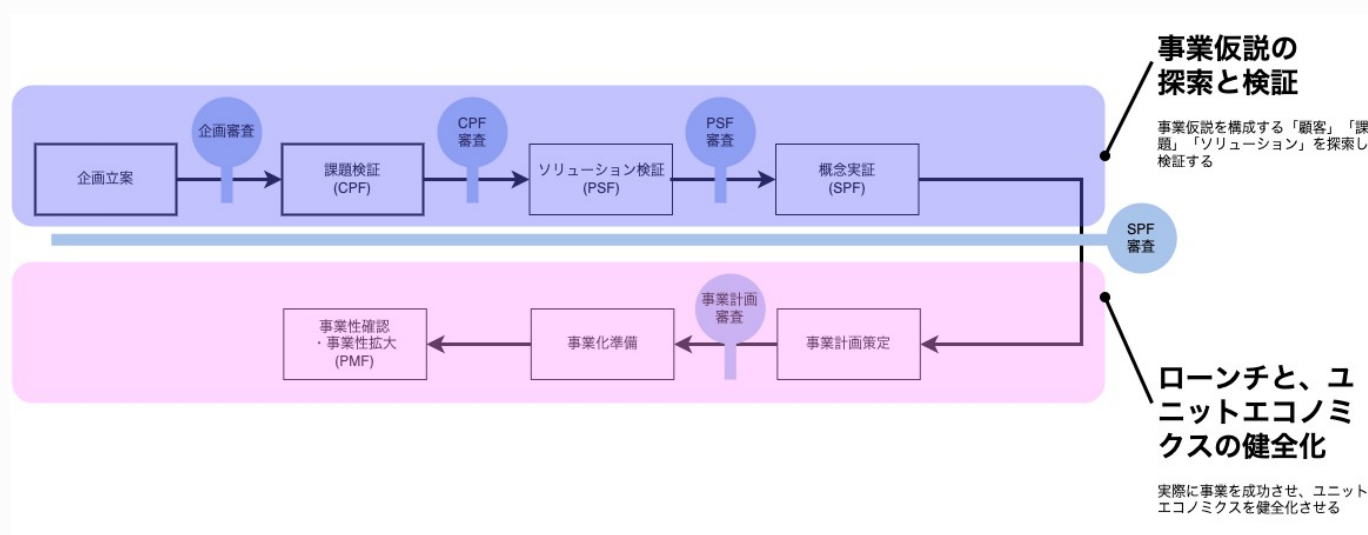
### 利用する技術の数

まず、扱う技術の数は少数に抑えるべきです。これにより、特定技術に関する知識を、組織の中で共有・応用できるようになります。

例えばプログラミング言語などはその良い例で、新規事業開発に取り組む組織の中でよく使われている言語を選択すると、組織から技術的なサポートも得られやすくなります。

また、新規事業開発には「事業仮説の探索と検証」と「ローンチとユニットエコノミクスの健全化」という2つのフェーズがあり、それぞれ求められるエンジニアリングが異なります。

その異なるエンジニアリングにおいても技術を「使いまわせる」方が学習効率も良いでしょう。



### 2種類のエンジニアリング

フェーズ	求められるエンジニアリング
事業仮説の探索と検証	事業仮説に対する高速・高頻度の検証のためのエンジニアリング

フェーズ	求められるエンジニアリング
ローンチとユニットエコノミクスの健全化	事業の高速な展開・成長のためのエンジニアリング

## 作らないエンジニアリングを支援できる

事業には、競争力のコアとなる箇所と、競争力のコアではないが事業を成立させるために必要な箇所があります。

「競争力のコアとなる箇所」は、市場競争にさらされながら生き残っていくために頻繁な改善が必要となる一方で、「競争力のコアとならない箇所」は変化することも少ないでしょう。そして、このような「コアとならない箇所」には、利用できる外部サービスやOSSプロダクトが多くあります。

例えば、モニタリングSaaSである[Datadog](#)や[New Relic](#)、カスタマーサポート用SaaSである[Zendesk](#)などはその一例でしょう。

変化することが少ない「競争力のコアとならない箇所」まで自分達で開発してしまうと、莫大な費用が必要になります。そのような不要な開発コストを避け「専門家」の構築したサービス品質を得るためには、それらを自分達で「作らず」、既存のサービスを「つなぐ」ことを選びましょう。

## インフラを所有しない

新規事業開発では仮説検証を繰り返すことになります。

多くの場合においてはサーバーやネットワークが必要になりますが、それらを所有すると初期投資が大きくなるとともに、その維持管理にも多大なコストが必要になります。

このため、いかにして「所有せずに済む」エンジニアリングを完徹できるかという点も重要になります。

その代表的な例が[Amazon Web Services \(AWS\)](#)や[Microsoft Azure \(Azure\)](#)といったクラウドサービスの利用です。クラウドサービスは、必要な時に必要なだけリソースを作成でき、破棄することも簡単です。

特にサーバーレスアーキテクチャでは、実際にサーバーを用意する必要はなく、クラウド事業者が用意した環境上にユーザーがソースコードを配置することでアプリケーションを運用できます。

アプリケーションの規模にもよりますが、ユーザー数の読めない段階では多くの場合メリットを享受できるでしょう。常時起動が必要なサーバーと比べて費用がかからない上、サーバーOSの管理といった作業からも解放されます。

## 新規事業における効果的な技術選定の事例

ここでは、効果的な技術選定の基準に基づいて我々が取り組む事業で選定した技術スタックを紹介します。

ここで紹介する技術スタックはあくまで参考例です。現場のエンジニアや事業オーナーと相談し、事業の価値と目的に合わせて適用してください。

## クラウドインフラ

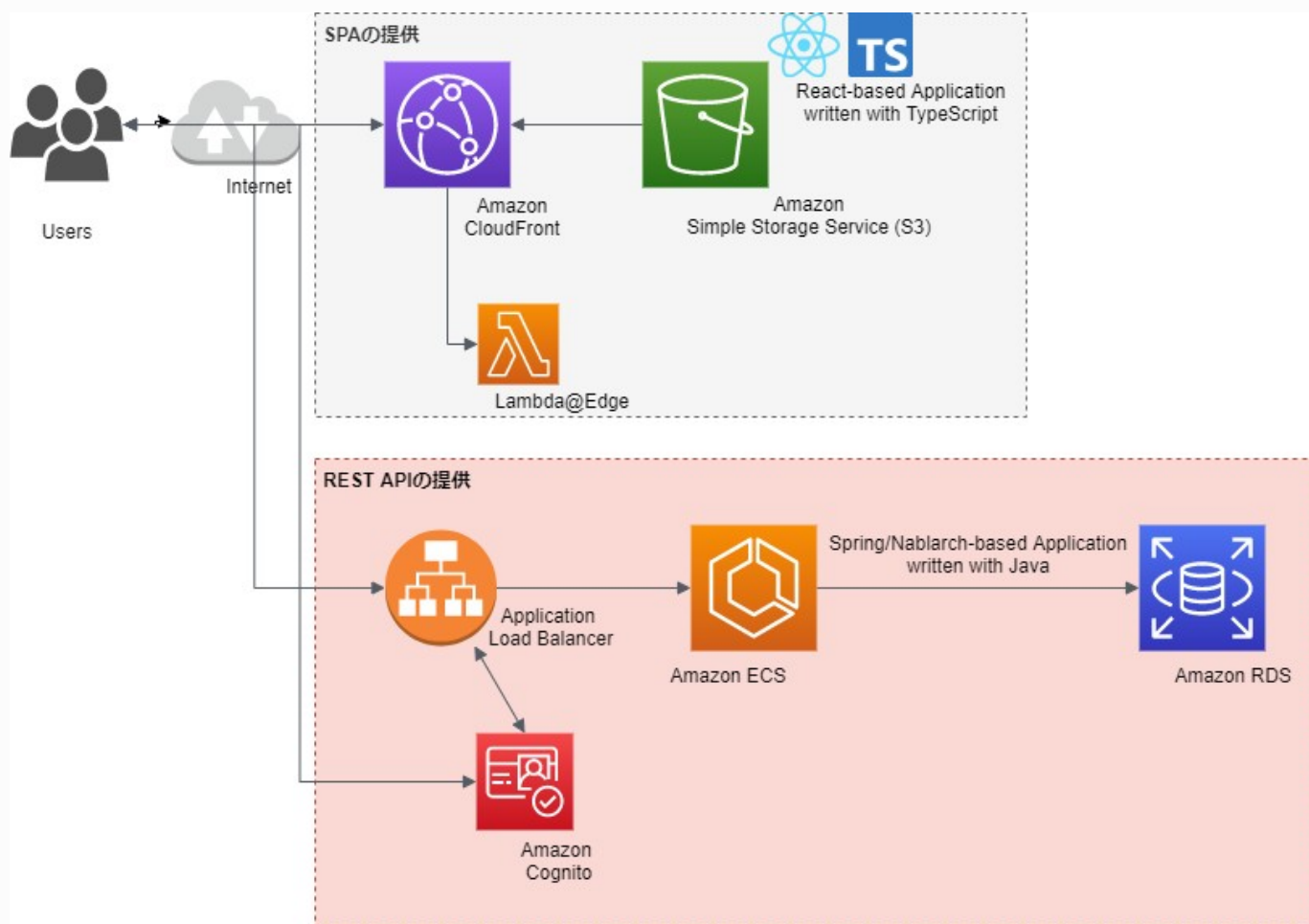
上記のアーキテクチャ図からもわかるように、我々が携わる多くの新規事業開発ではクラウドサービスとしてAWSを、Infrastructure as Code (IaC)のツールとしては[Terraform](#)を使っています。

これは、習熟している開発者が多いことと、[DevOps環境構築キット Epona](#)で両者をサポートしているためです。

## アーキテクチャ

事業アイデアが技術的な制約を課さない場合、我々はSPA+REST APIでのサーバーレスアーキテクチャを基本に考えています。

もちろん事業ごとに細部は異なるでしょうが、AWS上で実現するときは以下のようなアーキテクチャ<sup>1</sup>になるでしょう。



基本アーキテクチャ

## プロトタイピング

プロトタイピングツールについては[Figma](#)を利用します。

デザイナーと事業オーナーがスムーズにコミュニケーションを取れるとともに、デザインの意図や進行過程を残せることがメリットです。

## 開発言語

言語としては、型の存在による型安全性の確保と生産性の向上のため、JavaとTypeScriptを選択しました。

言語	理由
<a href="#">Java</a>	<ul style="list-style-type: none"> <li>・習熟している開発者が多い言語である</li> <li>・エコシステムが充実しており、フレームワーク・ライブラリ等も充実している</li> <li>・静的型付けのため、開発効率が上がりやすい</li> </ul>
<a href="#">TypeScript</a>	<ul style="list-style-type: none"> <li>・フロントエンドアプリケーションはJavaScriptが必須となるが、それに対して静的型付けを提供してくれ、開発効率が上がりやすい</li> </ul>

## フレームワーク

フレームワーク	主な利用場所	理由
<a href="#">Nablarch</a>	バックエンド	<ul style="list-style-type: none"> <li>・後方互換性が保証されるとともにセキュリティも強固であり、事業価値とは必ずしも関係しないアップデートに関する負荷が少ない</li> <li>※ただし、Nablarchが事業に必要なクラウドサービス等とのインテグレーションや処理方式に対応していない場合は、<a href="#">Spring Framework</a>も検討する</li> </ul>
<a href="#">React</a>	フロントエンド	<ul style="list-style-type: none"> <li>・自社内で経験者が多い</li> <li>・対応するUIライブラリが豊富にある</li> </ul>
<a href="#">React Native</a>	モバイル	<ul style="list-style-type: none"> <li>・Reactの知見を転用できる</li> <li>・クロスプラットフォーム対応のため、<a href="#">Android</a>、<a href="#">iOS</a>の双方に対応できる</li> </ul>

## VCS Provider

VCS Providerとしては、[GitLab](#)、[GitHub](#)のいずれかを利用することが多いです。

## 最後に

新規事業開発において、適切な技術選定は事業の成功を形作る基盤になります。

この記事では、利用する技術の数を最小限にすること、「作らないエンジニアリング」を支援すること、インフラを自社で所有しないことという3つの観点から、最適な技術選定の基準を示しました。



/\* Recommend \*/


## 「新規事業開発」 のおすすめ記事はこちら

この記事に関連する記事もお読みください。


 新規事業開発


サービス開発の進め方

2024/03/29  4


 新規事業開発

プロトタイプ作成のプロセス

2024/03/29  2

 新規事業開発

新規事業開発のステージ・ゲート  
プロセスとその評価基準

2024/03/29  21

最近投稿された記事も用意しました。

 新規事業開発


サービス開発の進め方

2024/03/29  4

 新規事業開発

プロダクト機能の見積


2024/03/29  2

 新規事業開発


プロダクトの言語化

2024/03/29  3

「新規事業開発」 で最も読まれている記事を以下にまとめています。

 新規事業開発


新規事業開発のステージ・ゲート  
プロセスとその評価基準

2024/03/29  21

 新規事業開発

新規事業スタートガイド

2024/03/29  48

 新規事業開発

仮説検証とMVPへの向き合い方

2024/03/29  2