

自己紹介スライド

京都大学大学院 工学研究科 修士1回生

山口輝樹

目次

サマリ

開発経験詳細

- スマホアプリ開発 まごとも
- アルバイト 株式会社StudioRadish
- Documentation-AI
- (新歓データベース)

補足 (大学・研究等)

1. 自己紹介スライド

- 氏名: 山口 輝樹 (やまぐち てるき)
- 学年: 修士1回生
- 学部: 京大院 工学研究科 (学部は京大工学部地球工学科)
- 出身: 愛知県

•エンジニアを志したきっかけ:

**事業をつくるうえでの価値創出や課題解決の最前線に関われる
と感じたこと**

- ・プログラミングとの出会いは日常的な課題解決
- ・その後の経験から、単なる課題解決手段のひとつに留まらず、技術が事業そのものの駆動力であると実感
- ・自分が面白いと思う領域=技術を強みに、ビジネスに携わりたい

2. 今後のキャリアビジョン

- ・志望・興味分野:

ソフトウェアエンジニア=>プロダクトオーナー、プロダクトマネージャー

- ・目指す像: 事業を作れる、意思決定できるビジネスマン

- ・ 幅広い知識・経験をもち、市場やリソースの制約も加味して、最善の方針を決定できるエンジニア (技術選定、設計、UIデザイン etc.)

- ・ プロジェクト内外の問題を技術者の視点をまじえて解決できるマネージャー

- ・就職先としての企業に対する期待:

キャリアを自分で作るための環境と支援がある

- ・ 多様な規模・内容のプロジェクトを経験できる
- ・ 手を挙げたら挑戦できる

- ・インターン探しの軸: ???

- ・ Bizを含む多くの関係者とやり取りしながら、機能開発?
- ・ インフラ・k8s・SREへの挑戦?
- ・ データ分析基盤など、事業推進・意思決定のための基盤の設計?

3. スキル・技術スタック

- 言語:

TypeScript, Python, Dart, HTML/CSS, SQL

- フレームワーク:

アプリ・システム開発: Express.js, React, Flutter, Laravel

機械学習: pandas/matplotlib/scikit-learn

- 開発ツール:

Git, Docker, OpenAPI, Firebase/Google Cloud,
GitHub Actions, AWS, Linux

- 開発経験年数:

約3年 (2022年3月から)

DEV まごとも

(whicker.info)

「若い友達」を作りたいお年寄りと、
「スキマ時間」で働きながら社会貢献したい学生をマッチング
(アプリの形態としてはクラウドソーシングサイトのイメージ)

アプリ開発だけではなく、ビジネスを間近で見たい！
CTOとして参画。先行していたアプリ開発を引き継ぐ

ビジネスモデル

日本初！学生とシニアのマッチングアプリ ※自社調べ

「利用」ではなく、「投資」として使ってもらえる社会へ！

仕事と介護の両立支援に
役立ちアプリで効率的に！

2,500円/h



個人情報登録&依頼

レポート

依頼者

ご家族@見守り/親孝行サービス



WEB&アプリ

1,200~1,500円/h



研修&仕事選択

レポート

学生

スキマ時間に
(空きコマ)社会貢献

高齢化問題に関心ある
Z世代は全体の13.2%
(集客コストかけず、
毎月集まっている。)



シニア

サービス提供

シニアが自分で払いたいと言い、
財布はシニアからの場合が多い。
=家族親が喜び自身の出費もないと
リピート率は高くなる(月約2万円)

DEV まごとも

アプリ: Flutter + Firebase

Riverpod x MVVM, Atomic Design, WidgetBook(Flutter版Storybook)
クライアントサイドでビジネスロジックを実行

管理画面: Next.js

バックエンド: Cloud Functions & Cloud Tasks

モジュラーモノリス/OpenAPI

イベント駆動な処理はFirestoreのトリガー機能で対応し、Cloud Pub/Subは使わない

時刻に紐づいた処理をCloud Tasksタスクキューでスケジューリング

技術: ロジックのクライアント→サーバーサイド移行

前提:

もともと、ビジネスロジックはFlutter**クライアントサイドでFirebaseを直接利用**

しかし、**WEB対応**の案が浮上。

→各クライアントデバイスではなく、**APIサーバーにビジネスロジックを集約**する必要性

アーキテクチャ選定:

- モノリスのメリットは大きくない (既にFirebase Authなどマイクロサービス向きのクラウド利用)
- マイクロサービスのインフラ管理コストも考慮

→**モジュラーモノリス&サーバーレスデプロイ**なら、中間的な使用感でちょうど良い。
モジュール間をまたぐトランザクションは、イベント駆動の結果整合で対応して密結合化を防ぐ

技術選定:

publish-subscribeメッセージングのために、Cloud Pub/Subの使用を検討したが中止。
∵ クライアントサイドに残っているビジネスロジックとの協調で相性が悪いから。

代わりに、**Transactional Outbox**によるイベント書き込み & ハンドリング

→クライアントサイドからもダブルコミットなしでイベントをトリガー(publish)。
テスト構築が容易に、subscriberの存在がわかりやすくなった

技術: ドメイン駆動設計の実践

業務フロー分析の必要性:

アプリ導入前までは、人力で仲介業者的なオペレーション。

アプリ導入でどう変わるか、中でも注力すべき**競争優位性**は何かを探索。

実践:

コアドメイン・サブドメインを分析

「現在はどうなのか？」イベントストーリーミングする一方

「アプリならどうか？」イベントストーリーミングを行い、集約やコンテキスト境界を明らかに
ユビキタス言語 (共通語彙) によるコミュニケーション

効果:

プラットフォームとしての事業方針への示唆

ワークショップ的に行うことで、関係者間での共通認識ができた

業務ルール(仕様)そのもの変更に伴う、変更の影響範囲を特定しやすくなった

コードが書きやすくなった

プロジェクト管理・経営企画会議

開発プロジェクト

学生エンジニア3名、デザイナー1名、社長

タスク管理・進捗管理:

期待する成果を明らかにしてタスクを定義し、伝える (一方、伝えなくていいことは伝えない)
実装タスクだけでなく「考える」タスクをお願いすることも
手が止まった時のサポート

経営・企画会議:

市場ポジショニング・ターゲティング戦略策定 (資金調達のために！)
サービス設計の改良 (ex: 依頼の有効期限) ・利用規約の作成
事業提携の計画 (福利厚生を導入やフランチャイズ化)

DEV WhiteBoard

アルバイト @株式会社StudioRadish 2022年5月

新歓データベースから2ヶ月！

フロントエンドフレームワーク開発:

- Miroのようなもの？
- but あるWBに別のWBやガジェットを貼付 (ツリー構造) → グラフィカルなNotion
- サーバーサイドレンダリング。サーバーサイドでも仮想DOMを保持する

RPC用ミドルウェア プロトタイプ開発:

- OORPC (Object-Oriented RPC):
ステートフル通信を前提としたRPCフレームワーク。旧来のGUIのMVC1のような通信スタイルを実現
- d-OORPC (distributed-OORPC):
ステートフルなKubernetes。しかし永続化の必要なステートはインメモリではなくDBに永続化。
すべてのリクエストはタスクキューに入れられ、pull型でワーカーノードが処理。

DEV WhiteBoard

プロジェクトリーダー

コミュニケーションが少なく、情報共有が難しい

しかし、未知のモノをつくるので全員のアイデアの凝集がより重要
→仕組みの整備

issue・PRの活用、全員見れる進捗報告、定期ゆるLT&勉強会

知識を補完し合えるようになった。

他のサブシステムに求めるモノが可視化され、要件が固まった。

DEV Documentation-AI

GitHub: github.com/HLHHS11/Documentation-AI

YouTube: [技育展プレゼン動画](#)

ソースコードから、LLMを用いて自動ドキュメント生成

シンボル間の**依存関係**を解決し、

シンボルの**ソースコード**と、依存先シンボルの生成済み**ドキュメント**

必要なコンテキスト情報を補完しながらプロンプトを投げる

Python, LangChain, dependency-injector, AST(標準ライブラリ)
クリーンアーキテクチャ(軽量DDD)、有向非巡回グラフ(DAG)、並列処理

DEV Documentation-AI

技育展2023 決勝大会進出

企業賞 (GMOインターネットグループ様)

精度向上と客観的指標 – プロンプト追求、コンテキスト追加

多言語対応 – 抽象クラスと依存性注入による疎結合化

変更の監視 – 現状、変更が無関係でも全部ドキュメント生成

VSCode拡張機能 → Cursorに組み込む??

開発補助AIエージェント

→チャットボット、コード自動生成・自動実行

DEV 新歓データベース


2022年3月。2年9ヶ月前。

新入生（200人超）の情報を効率的に管理・共有
Googleスプレッドシート と GoogleAppsScript

タイムライン機能 → 情報更新に気づける！
色分け機能、検索機能、日程管理機能

1年後、ついでにWebアプリもつくってみた

DEV 新歓データベース



9:26

	A	B	C	D	E	F	G	H
1			名前	学科	経験		出身	連担
4								
5								
6								
7								
8								
9								
10								



9:23

ああ hlhs11.github.io

作成 検索 クリア

山田太郎 やまだ

経 3 年 理

入会意思 藤島 しゅんすけ

参加日程 編集

しゅんすけの後輩

更新 削除

山口輝樹 やま

軟フレ 6 年 地球工

強 愛知・メ はるっぺ

参加日程 編集

体育会と迷ってる

更新 削除

< > 共有 ブックマーク コピー

UNIVERSITY

土木工学

- 都市計画: 数理最適化、交通網モデル、景観設計…
- 土質力学: 透水性、地盤沈下、建物基礎、地震学、液状化現象…
- 構造力学: 構造物、ひずみ、荷重、材料強度、フレーム構造…
- 水理学: 流体力学、ベルヌーイの定理、層流・乱流、降雨浸透…
- 環境工学、測量学、材料学、コンクリート工学、公共経済学…

RESEARCH

MLを用いて河川流量データの欠損値補完を行うフレームワークの提案

先行研究:

観測所のデータを注意深く選び、トライアル&エラーを経て補完の成果とする

新規性:

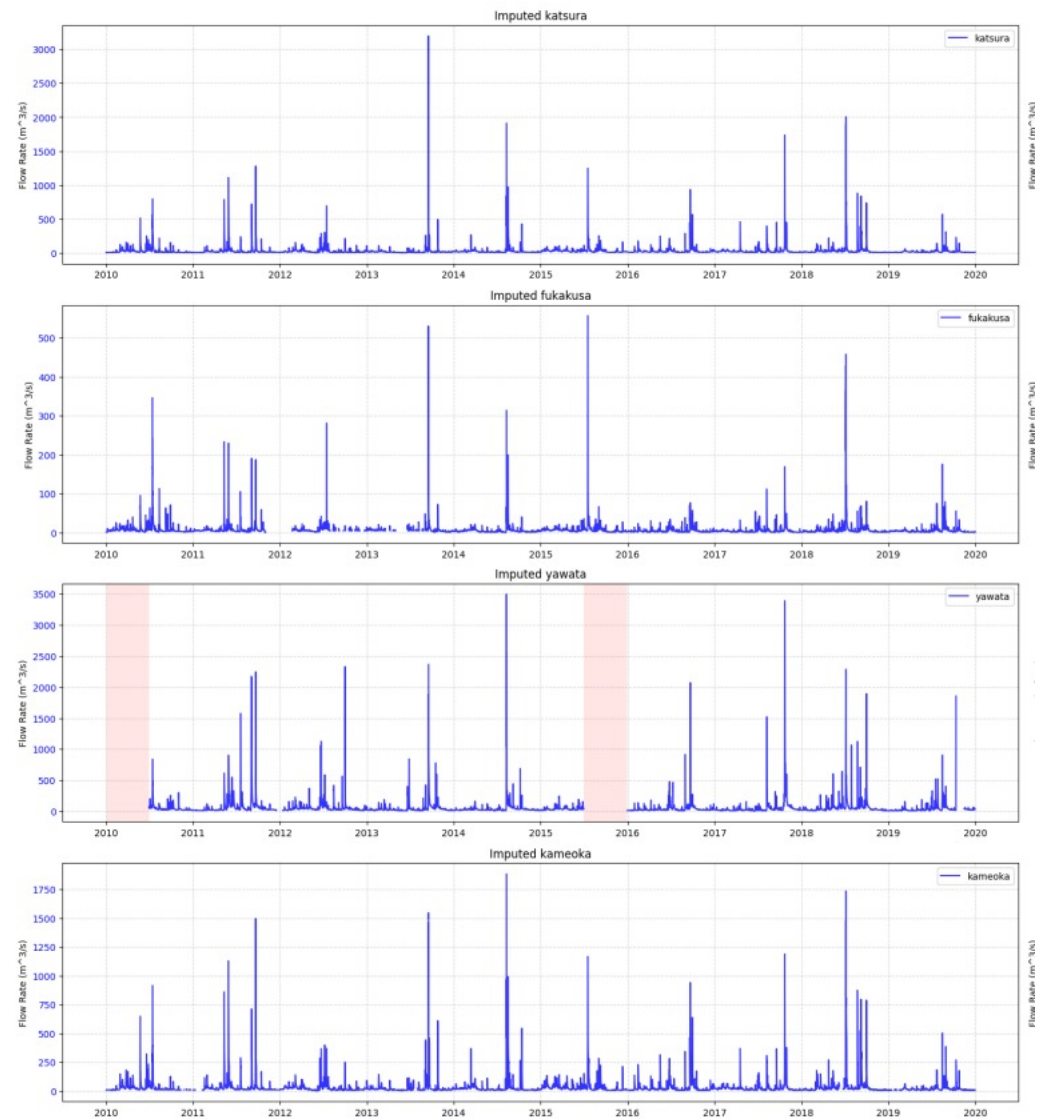
人力かつ実施者のもつ知識に依存するような手法ではなく、
機械的に一括で対応できる手法を模索する点

結果:

MissForest アルゴリズムを複数の人工欠損シナリオや入力変数選定シナリオで適用。
怪しいデータを入力から削ったときより、まとめて含めたほうが良い成果
→特徴量選択 的なことをせずとも、一括での変数投入・補完が可能であることを示唆

今後:

既存の河川流量のデータベースを置き換えるプロダクト・サービスとして提案??



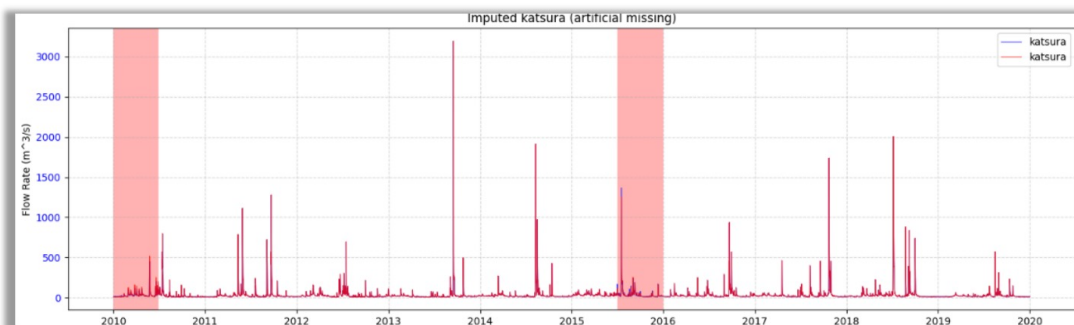
Train Regressor
Predictors:
Target:

River Streamflow Data

	STATION							
	1	2	p-1	p	
1	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	
...	0	0	0	0	0	0	0	
...	0	0	0	0	0	0	0	
...	0	0	0	0	0	0	0	
...	0	0	0	0	0	0	0	
...	0	0	0	0	0	0	0	
...	0	0	0	0	0	0	0	
...	0	0	0	0	0	0	0	
...	0	0	0	0	0	0	0	
...	0	0	0	0	0	0	0	
...	0	0	0	0	0	0	0	
n-1	0	0	0	0	0	0	0	
n	0	0	0	0	0	0	0	

NOTE:
No distinction for each predictor
whether it's observed or predicted

Example Result of MF Application



RMSE: 11.7
R²: 0.97

