freee会計から マイクロサービスを切り出すのに 4年かかりました

him0

2023年4月16日

freee会計 SMB会計開発 JM

him0

2017年新卒入社（Eng. 1期生）

入社以来 freee会計開発チームに所属

ワークフローを開発するチームのマネージャー兼プレイヤー、東京と関西の多拠点チームで開発しています

実は、freeeづくりはこんなにも面白い。
freee技術の日

ちょっとだけアプリケーションアーキテクチャと
freee の現状の話をします
ごちゃごちゃ言いますが
5枚目だけ分かれば大丈夫です

実は、freeeづくりはこんなにも面白い。
freee 技術の日

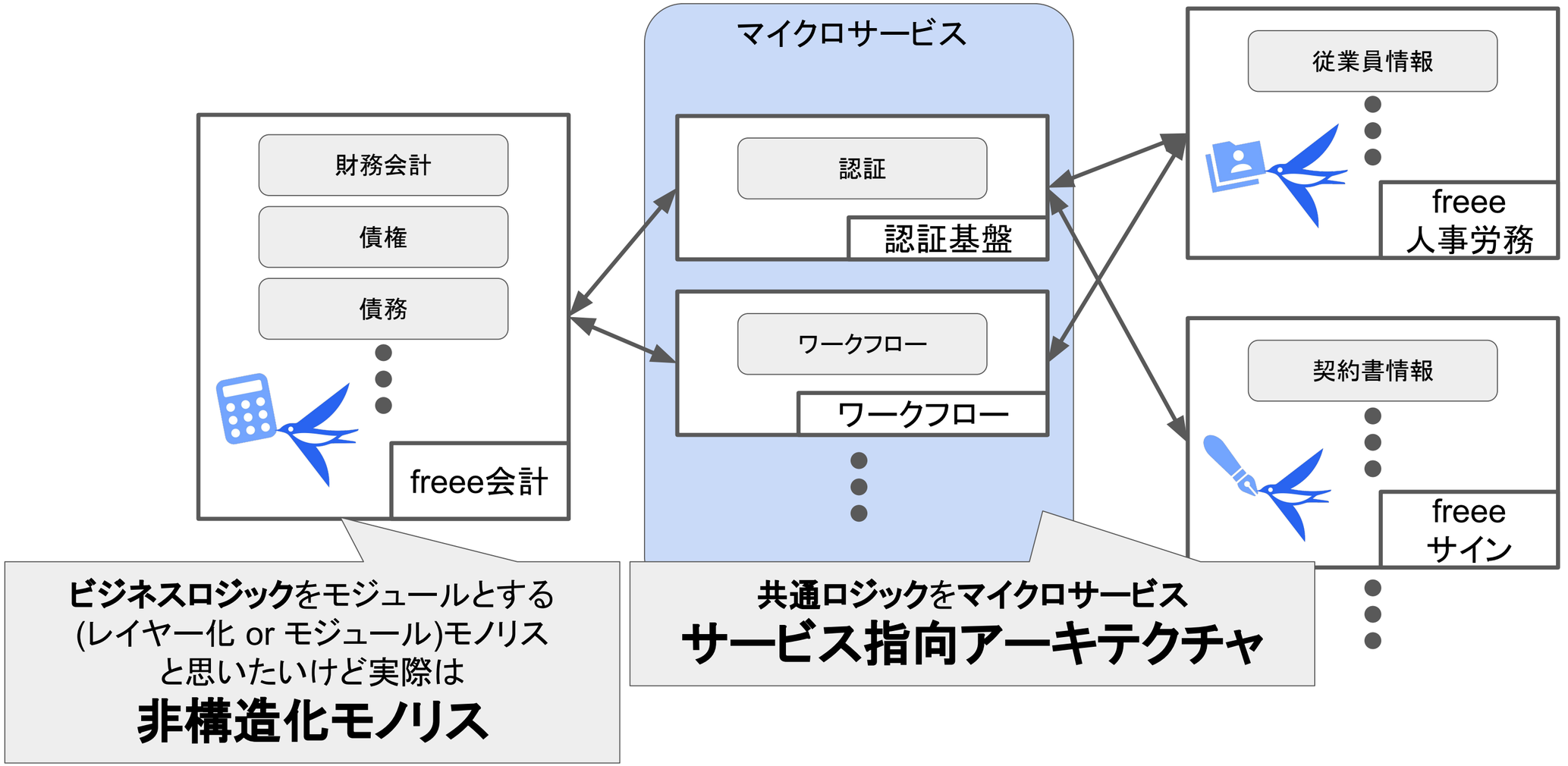
webアプリケーションのソフトウェアアーキテクチャ

- 大きな泥団子 (Big Ball Of Mud)
- モノリス (Monolith)
  - 非構造化モノリス
  - レイヤー化モノリス
  - モジュール式モノリス
- イベント駆動アーキテクチャ
- サービス指向アーキテクチャ
  - マイクロサービス (Micro Service)

一概にモノリスが悪でマイクロサービスが善であるわけではない
今後の**プロダクト戦略**や**組織構造**に合わせて**サービスの価値を最大化**できる
アーキテクチャを選択していくことが重要

実は、freeeづくりはこんなにも面白い。
freee 技術の日

現状の feee会計 のアーキテクチャ



会計がモノリスな構造を選択してきた理由と問題

freee会計(アプリケーション) のビジネス要件

- 売上を担うプロダクト機能追加やビジネスのピボットを素早く行えること

モノリスのメリット

- 開発者がコードにアクセスしやすく理解しやすい
- ドメインをまたいだデータ連携が容易

モノリスのデメリット

- ドメイン依存関係の複雑化、仕組みで防げない
- スケーラビリティ、パフォーマンス対応の複雑化

ばりばりアプリケーション作るぞ

↓

モノリスで作る

実は、freeeづくりはこんなにも面白い。
freee技術の日

共通ロジックをマイクロサービスとしてきた理由

共通ロジックに対するビジネス要件

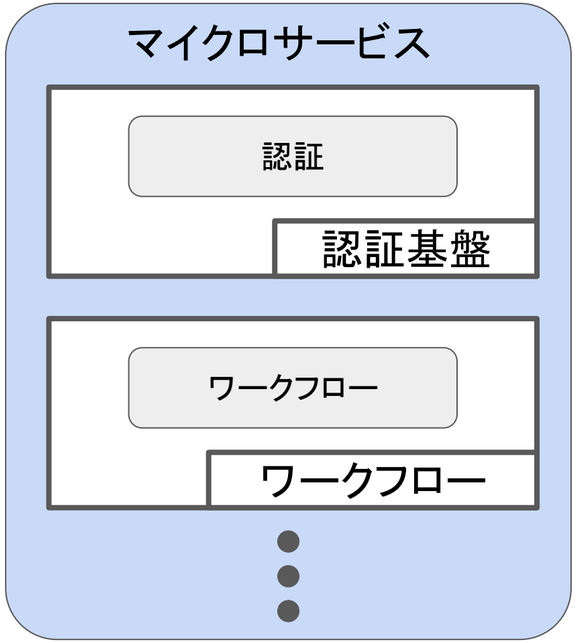
- 柔軟性よりも堅牢性が大事
- 小さく作ってコードはあまり動かなさない

マイクロサービスのメリット

- アプリとの連携により機能追加が実現可能
- ドメインを明確に切って開発に集中できる

マイクロサービスのデメリット

- 複数のアプリから依存されるのでロジックを改修に対するコストが高い

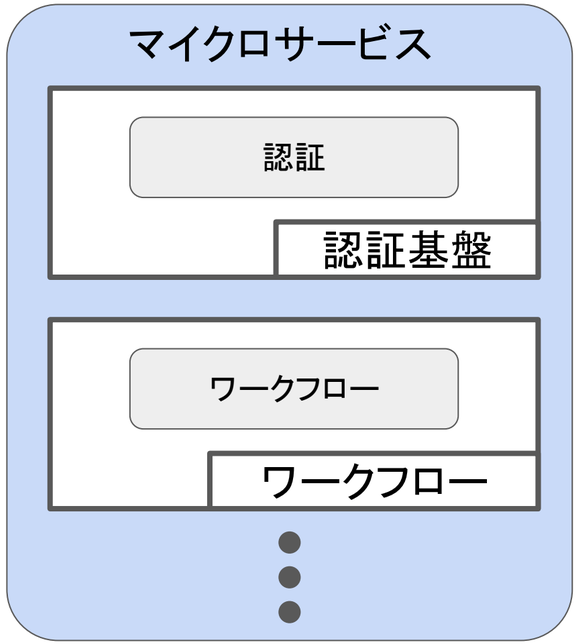


複数アプリケーションでの利用

マイクロサービス作る

実は、freeeづくりはこんなにも面白い。
freee 技術の日

freee におけるアーキテクチャ採用理由をざっくり



ばりばりアプリケーション作るぞ

⬇

モノリスで作る

複数アプリケーションでの利用

⬇

マイクロサービス作る

freeeづくりはこんなにも面白い。
freee技術の日

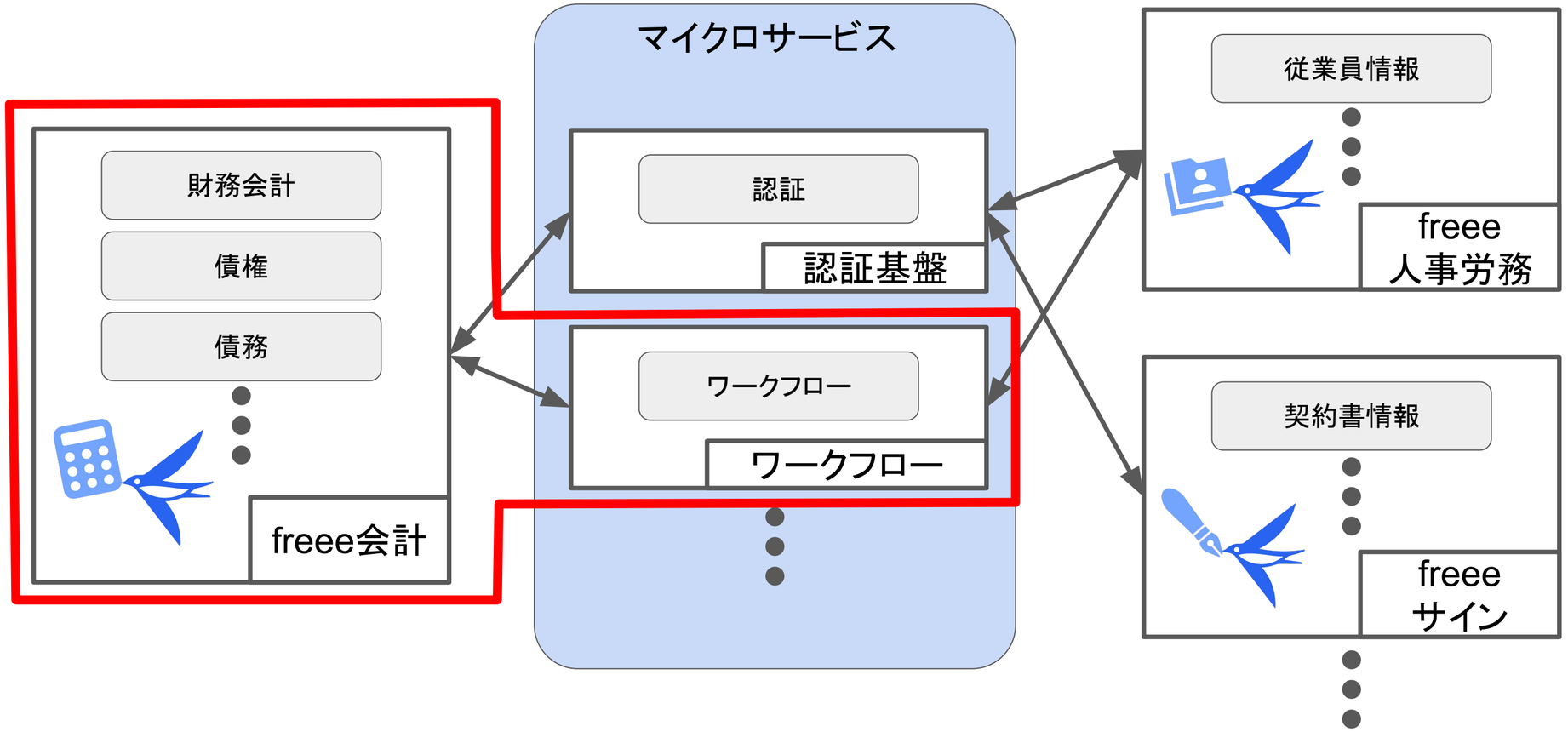
本題入ります

実は、freeeづくりはこんなにも面白い。
freee技術の日

アプリケーションエンジニアが
4年かけて取り組んだ
WFマイクロサービスの切り出しの話

実は、freeeづくりはこんなにも面白い。
freee技術の日

実際に切り出した部分



もともとfreee会計の中にあった**ワークフロードメイン**を
マイクロサービスとして**切り出した**

マイクロサービスの切り出し専任ではない
メインプロジェクトがあるエンジニア
25%~50% ぐらいの工数

アプリケーションエンジニアが
4年かけて取り組んだ
WFマイクロサービスの切り出しの話

実は、freeeづくりはこんなにも面白い。
freee技術の日

アプリケーションエンジニアが
4年かけて取り組んだ
WFマイクロサービスの切り出しの話

実は、freeeづくりはこんなにも面白い。
freee技術の日

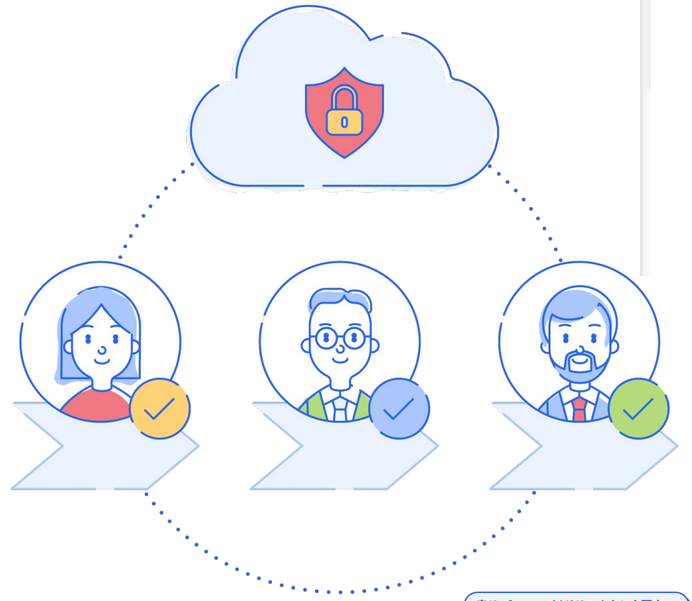
ワークフロー(WF)とは

- 特定の申請を承認するプロセス
- 会社のお金を動かす際などに、
  金額や用途が正しいのかを検証

freee のWFロジック

- 承認経路の管理
- 承認状態の管理
- 部門・役職から承認者を選出

などの機能が備わっている



実は、freeeづくりはこんなにも面白い。
freee 技術の日

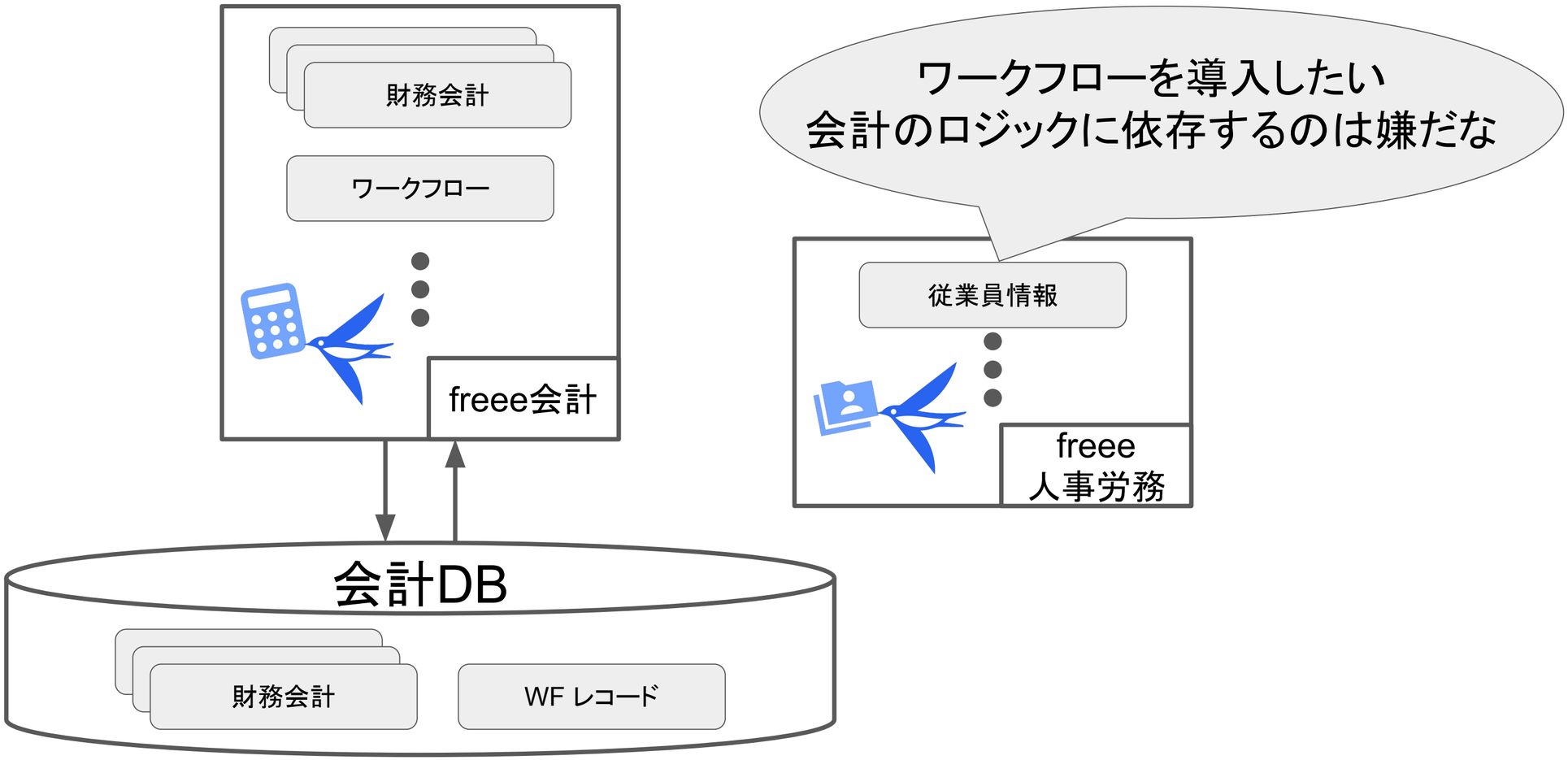
アプリケーションエンジニアが
4年かけて取り組んだ
WFマイクロサービスの切り出しの話

時系列

- 2018年 プロジェクト発足
- 2019年 マイクロサービス完成
- 2020年 検索パフォーマンスの問題が顕著化
- 2021年 Elasticsearch のお世話でてんやわんや
- 2022年 DB負荷対策、DB分離が急務に
- 2022年末 ついに DB分離 完全なマイクロサービスへ

実は、freeeづくりはこんなにも面白い。
freee 技術の日

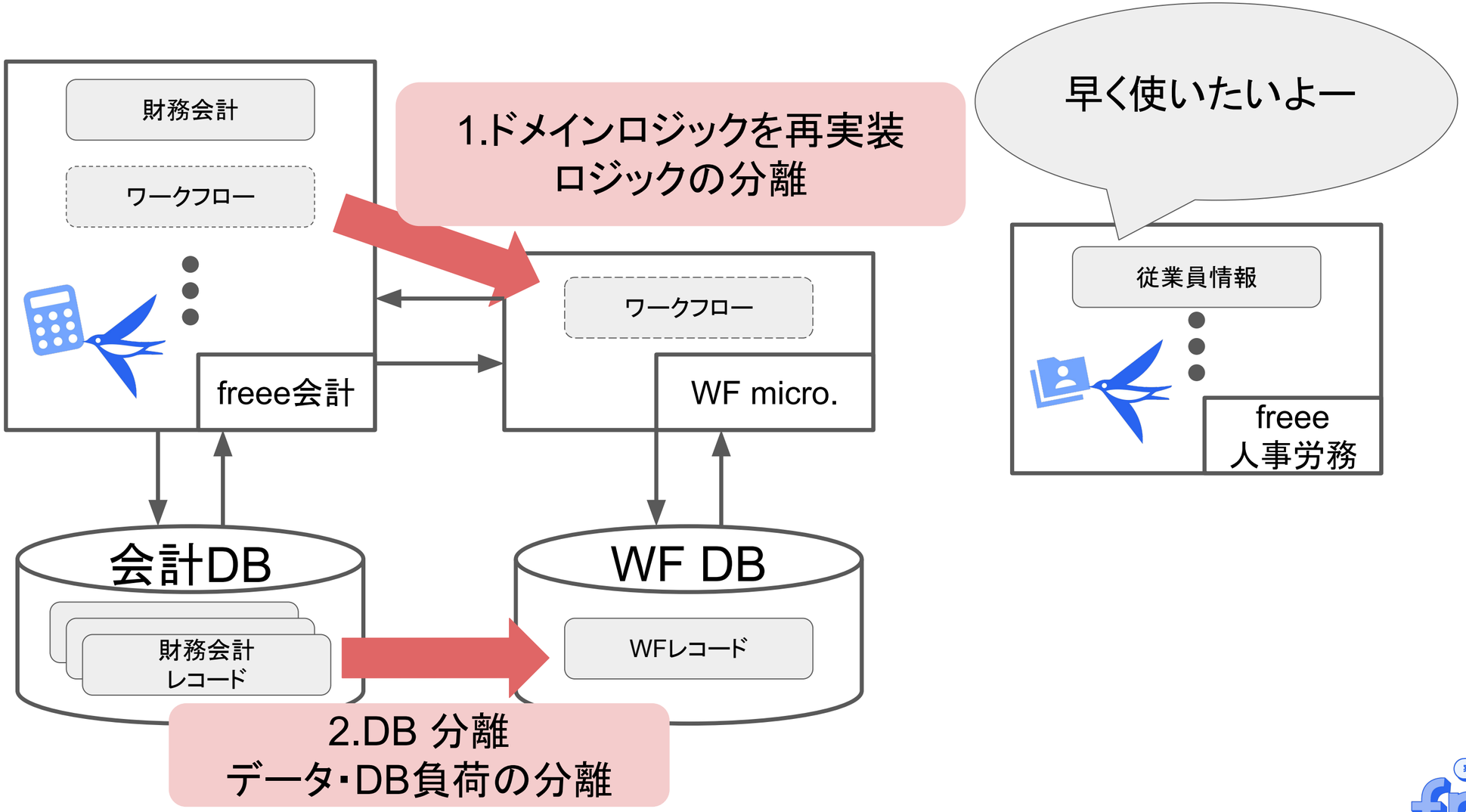
2018年 ワークフローの汎用利用の要求が出てきた



WFマイクロサービス化の機運が高まりました

実は、freeeづくりはこんなにも面白い。
freee技術の日

マイクロサービスの切り出し要件は2つ



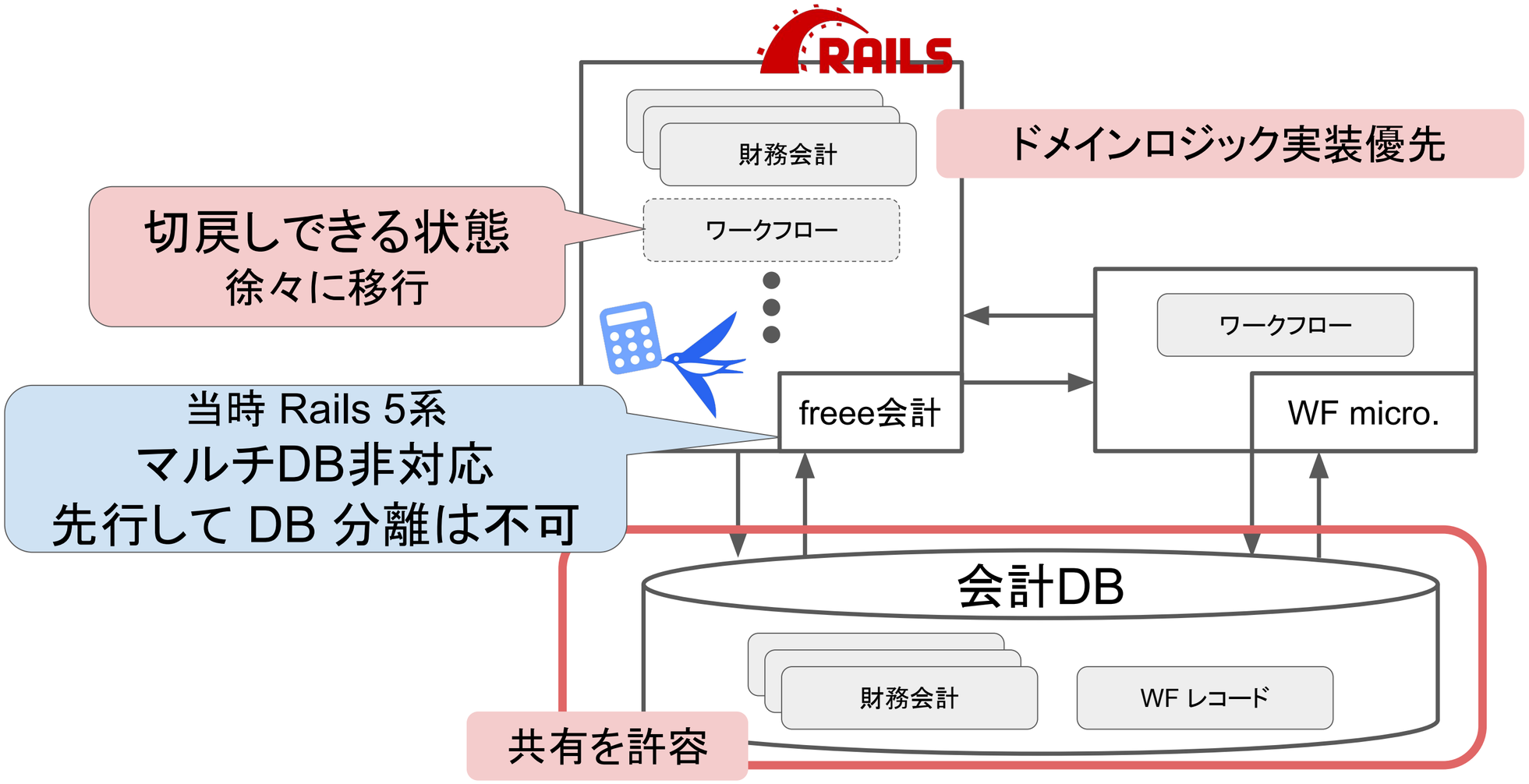
2つの要件を満たせば WF はマイクロサービスとして完全に独立

実は、freeeづくりはこんなにも面白い。
freee技術の日

どう進めていくべきか？ 

実は、freeeづくりはこんなにも面白い。
freee技術の日

マイクロサービスの作成を優先



安全にマイクロサービスに移行するために
ドメインロジック2重にして徐々に切り替えていく戦略をとった

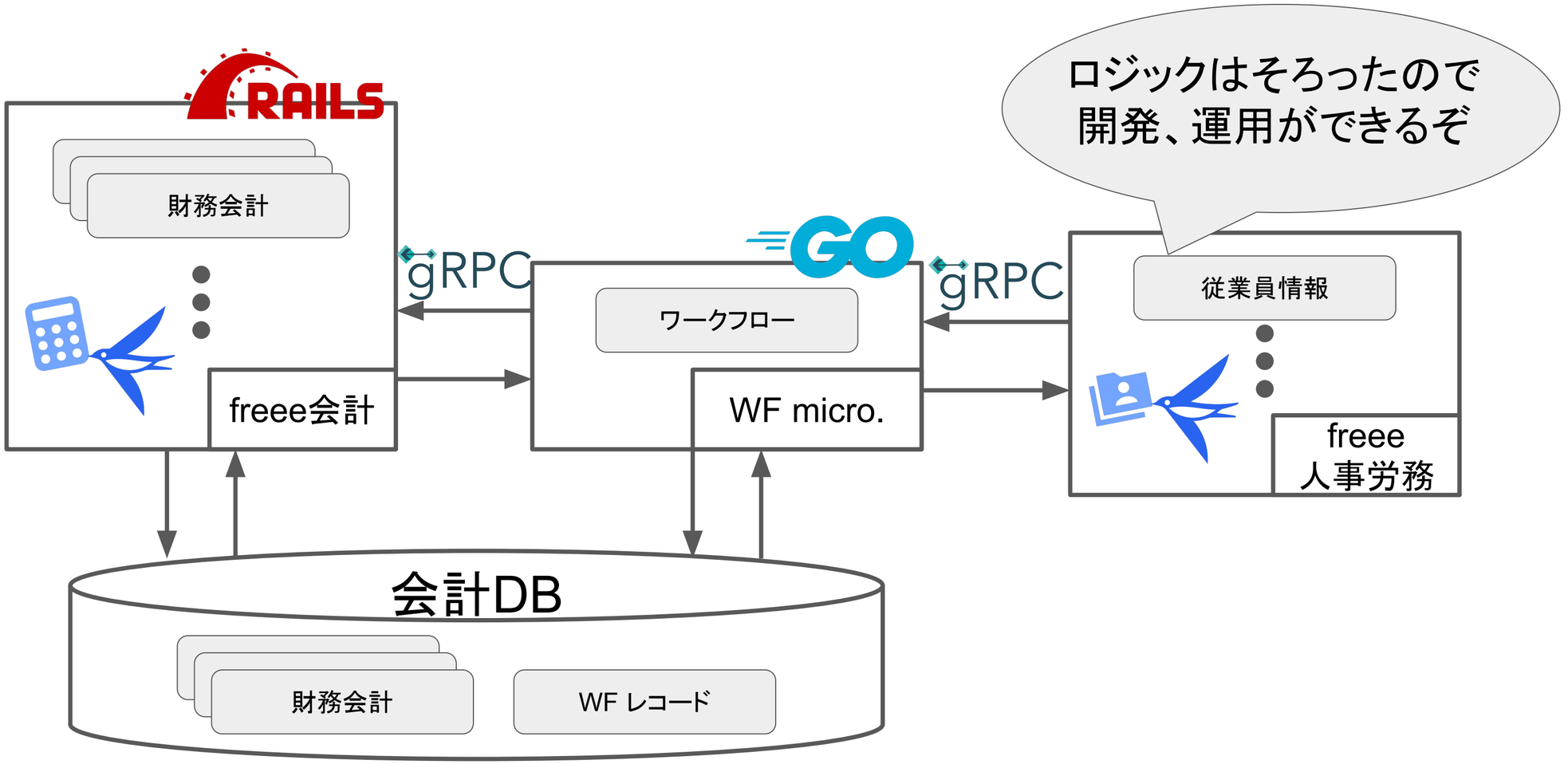
実は、freeeづくりはこんなにも面白い。
freee技術の日

時系列

- 2018年 プロジェクト発足
- 2019年 マイクロサービス完成
- 2020年 検索パフォーマンスの問題が顕著化
- 2021年 Elasticsearch のお世話でてんやわんや
- 2022年 DB負荷対策、DB分離が急務に
- 2022年末 ついに DB分離 完全なマイクロサービスへ

実は、freeeづくりはこんなにも面白い。
freee 技術の日

2019年 ロジックの実装が完了



人事労務チームは爆速で既存機能に WF 機能を導入できた
データストアを共有する半マイクロサービス状態

実は、freeeづくりはこんなにも面白い。
freee 技術の日

時系列

- 2018年 プロジェクト発足
- 2019年 マイクロサービス完成
- 2020年 検索パフォーマンスの問題が顕著化
- 2021年 Elasticsearch のお世話でてんやわんや
- 2022年 DB負荷対策、DB分離が急務に
- 2022年末 ついに DB分離 完全なマイクロサービスへ

実は、freeeづくりはこんなにも面白い。
freee 技術の日

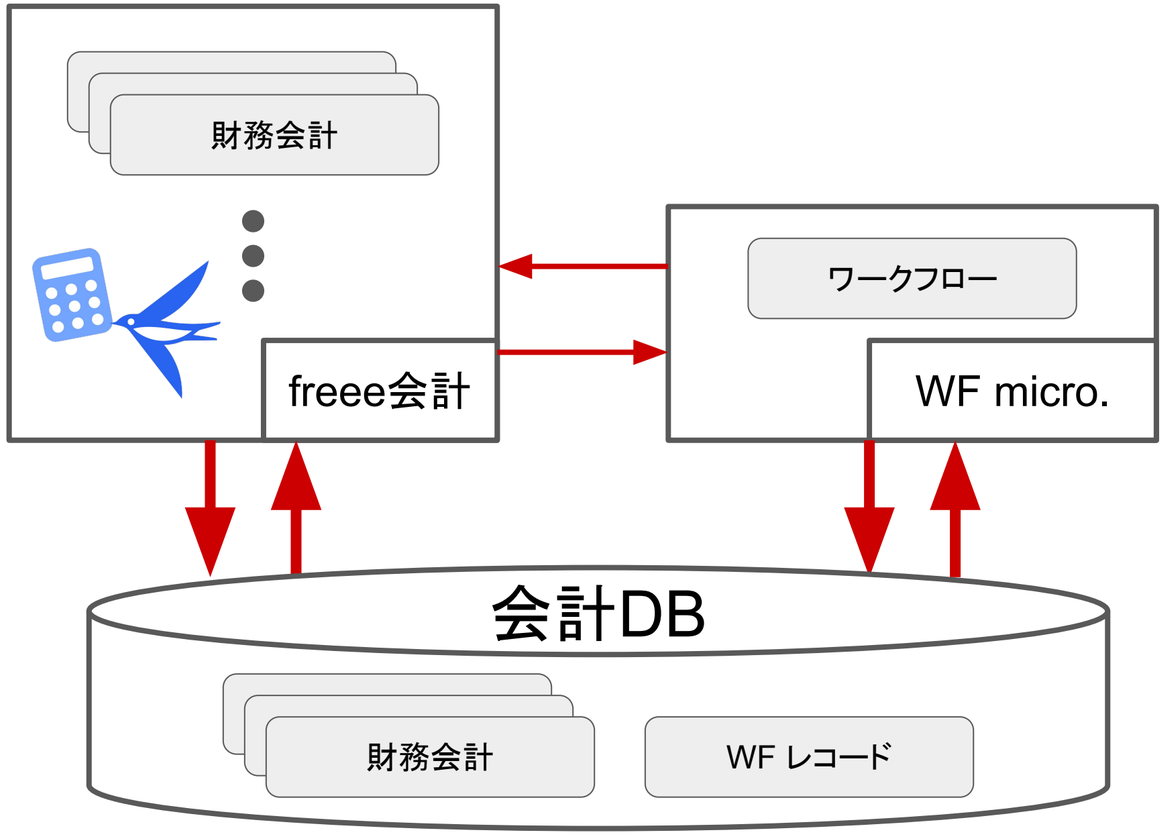
2020年 検索パフォーマンスの問題が顕著化

マイクロサービスとアプリでデータの重複を避けてしまった（マイクロサービスアンチパターン）

会計とWF micro. のデータを結合する際のパフォーマンスが気になり始める

データの取得流れ

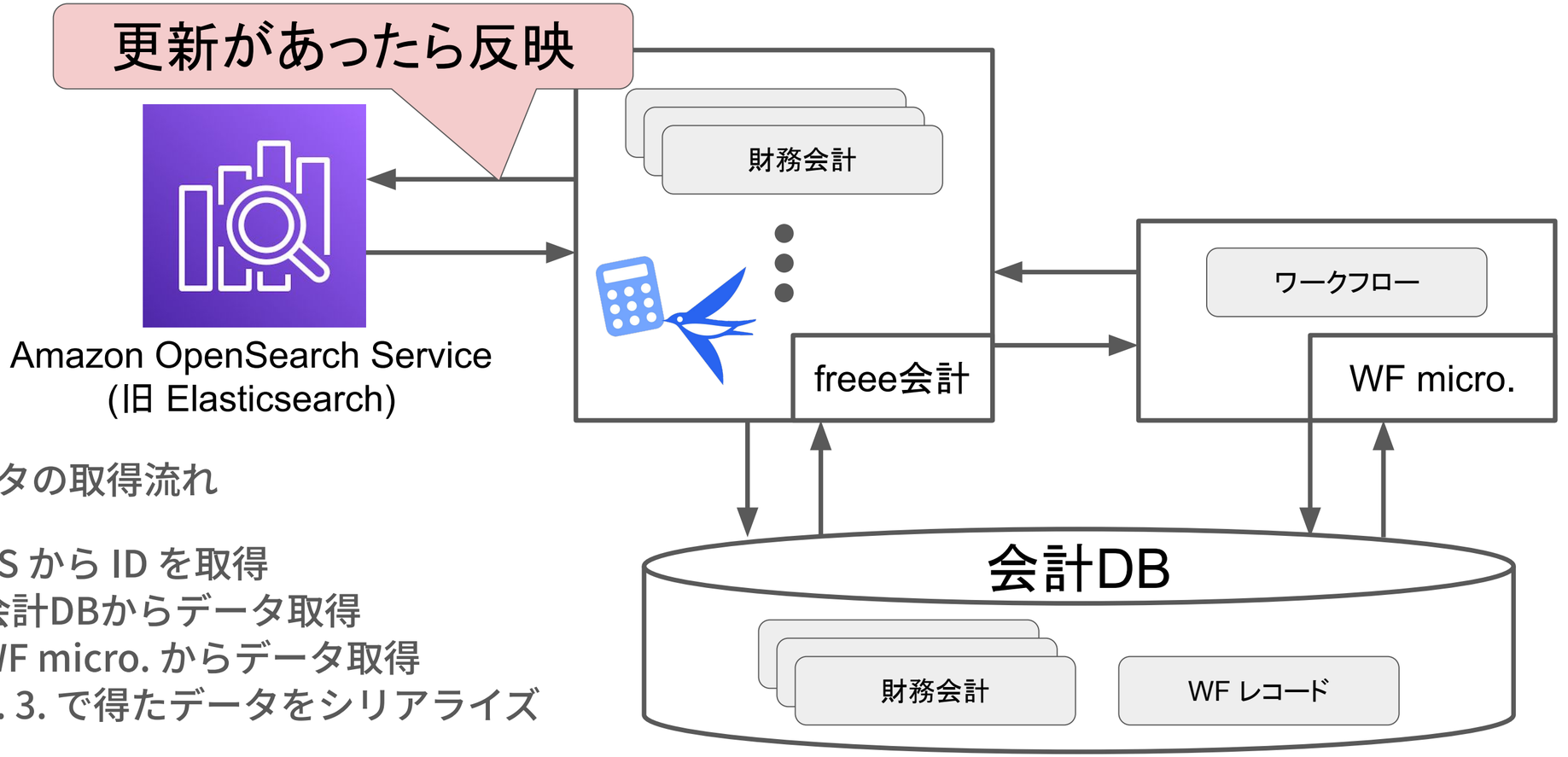
1. WF micro. からアクセス可能なデータの ID を全件取得
2. 1. で手に入れた ID を in 句に入れつつ会計がDB から検索対象になりそうなデータを取得
3. 2.で得たデータをもとに WF micro. から WF のステータスを取得
4. 2. 3. で得たデータをシリアライズ



一覧を表示するだけでも SQL を複数回発行する必要がある状態に
また深刻な部分は会計とWFレコードの結合でパフォーマンス対応が必須

実は、freeeづくりはこんなにも面白い。
freee技術の日

検索パフォーマンスのカイゼン、検索基盤の導入



データの取得流れ

1. ES から ID を取得
2. 会計DBからデータ取得
3. WF micro. からデータ取得
4. 2. 3. で得たデータをシリアライズ

会計と WF micro. のデータを結合した状態でインデキシング
検索負荷の軽減が可能になった

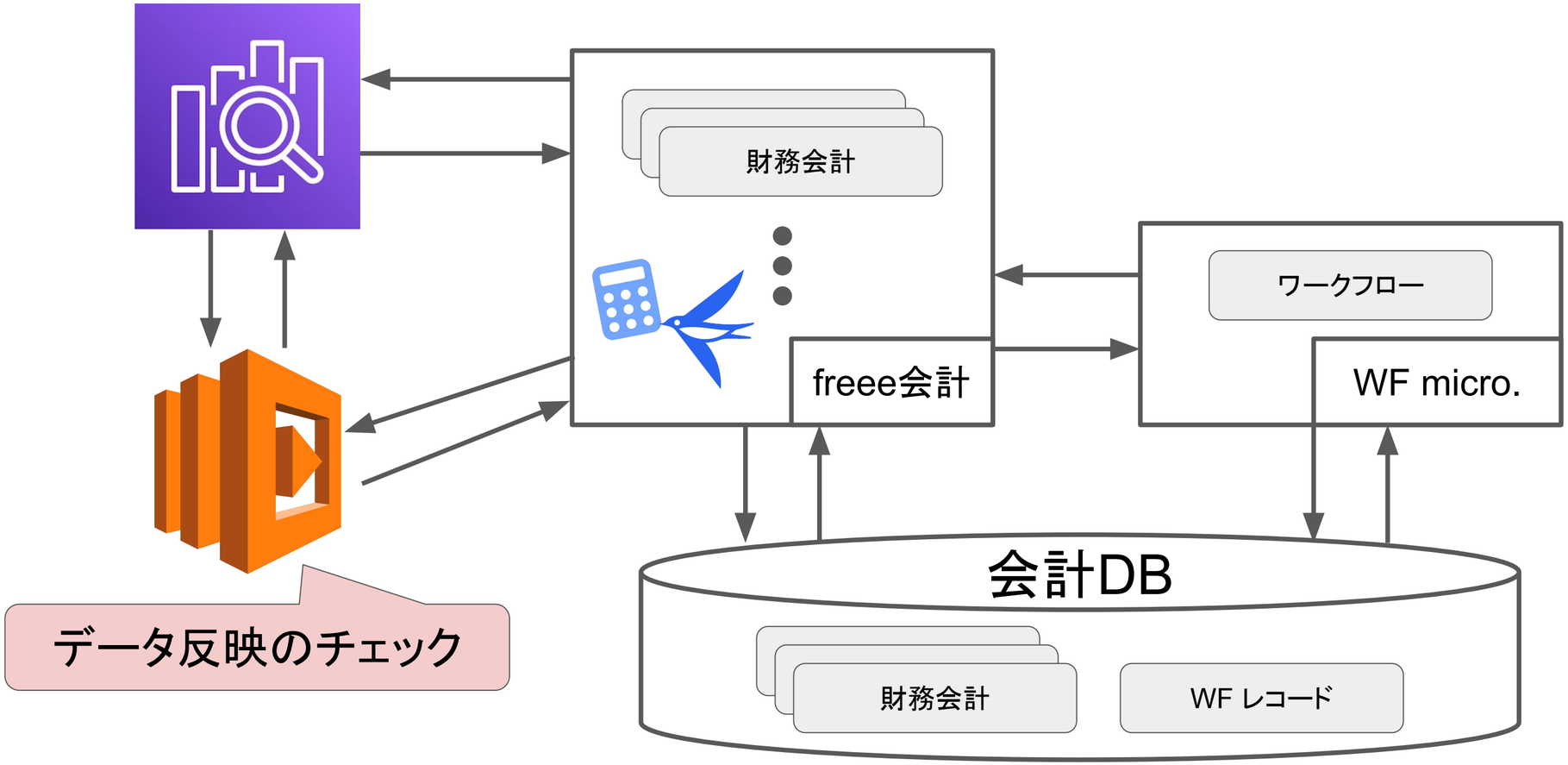
実は、freeeづくりはこんなにも面白い。
freee技術の日

時系列

- 2018年 プロジェクト発足
- 2019年 マイクロサービス完成
- 2020年 検索パフォーマンスの問題が顕著化
- 2021年 Elasticsearch のお世話でてんやわんや
- 2022年 DB負荷対策、DB分離が急務に
- 2022年末 ついに DB分離 完全なマイクロサービスへ

実は、freeeづくりはこんなにも面白い。
freee 技術の日

2021年 Elasticsearch のお世話でてんやわんや



キャパシティプランニングやデータの復旧ロジックの導入、監視の強化
てんやわんやしていたら1年終わりました

実は、freeeづくりはこんなにも面白い。
freee 技術の日

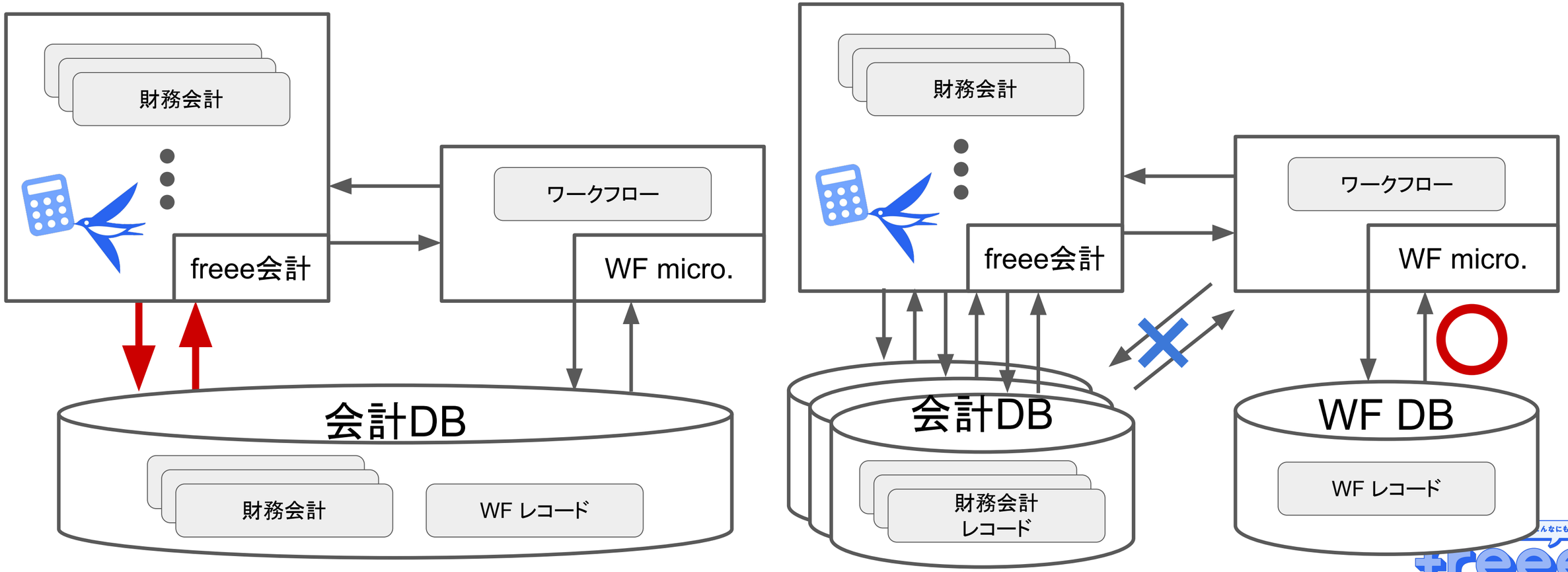
時系列

- 2018年 プロジェクト発足
- 2019年 マイクロサービス完成
- 2020年 検索パフォーマンスの問題が顕著化
- 2021年 Elasticsearch のお世話でてんやわんや
- 2022年 会計DB負荷対策が始まった
- 2022年末 ついに DB分離 完全なマイクロサービスへ

実は、freeeづくりはこんなにも面白い。
freee 技術の日

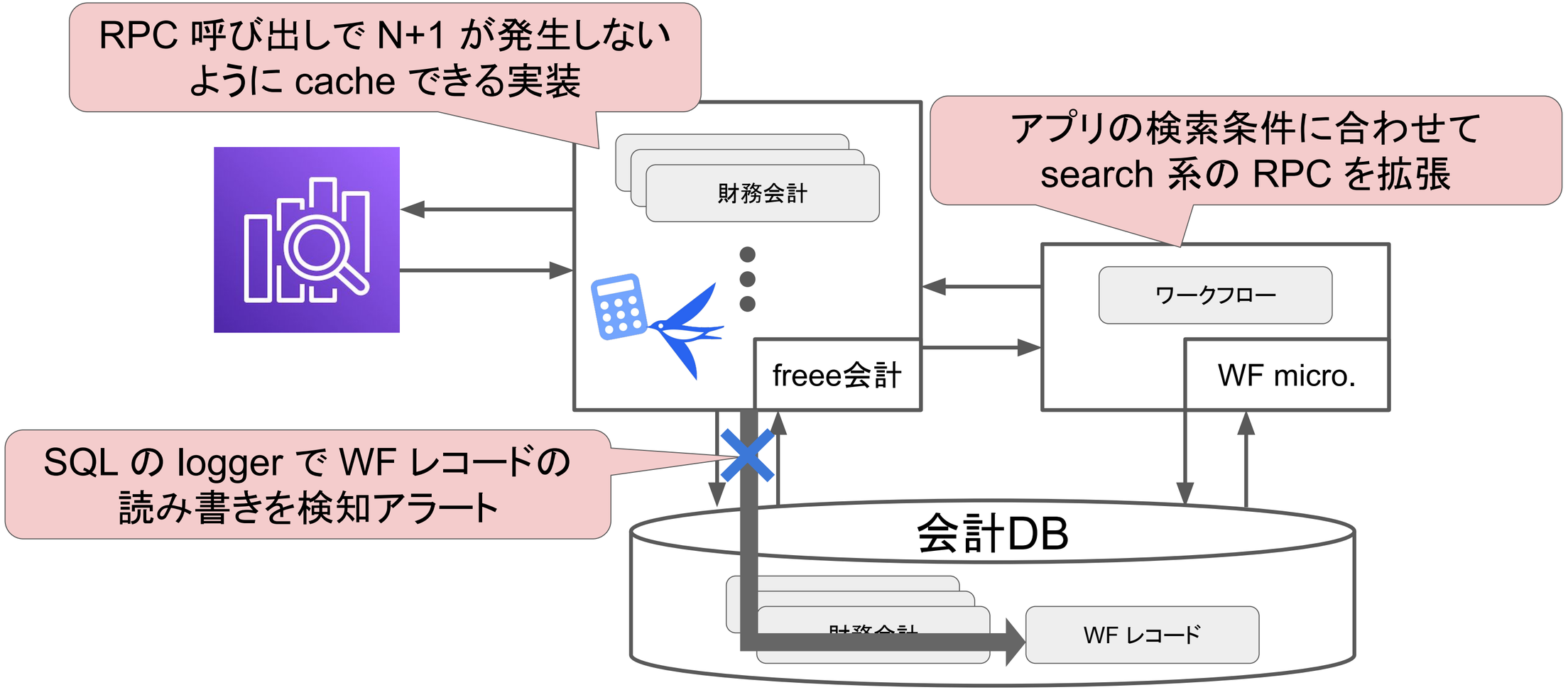
2022年 会計DB負荷対策が始まった

- 会計DBの負荷対策としてシャーディングの検討が開始
- WF レコードはシャーディング対象とせず切り出しして欲しいという流れになった



WF DBの分離は急務となった

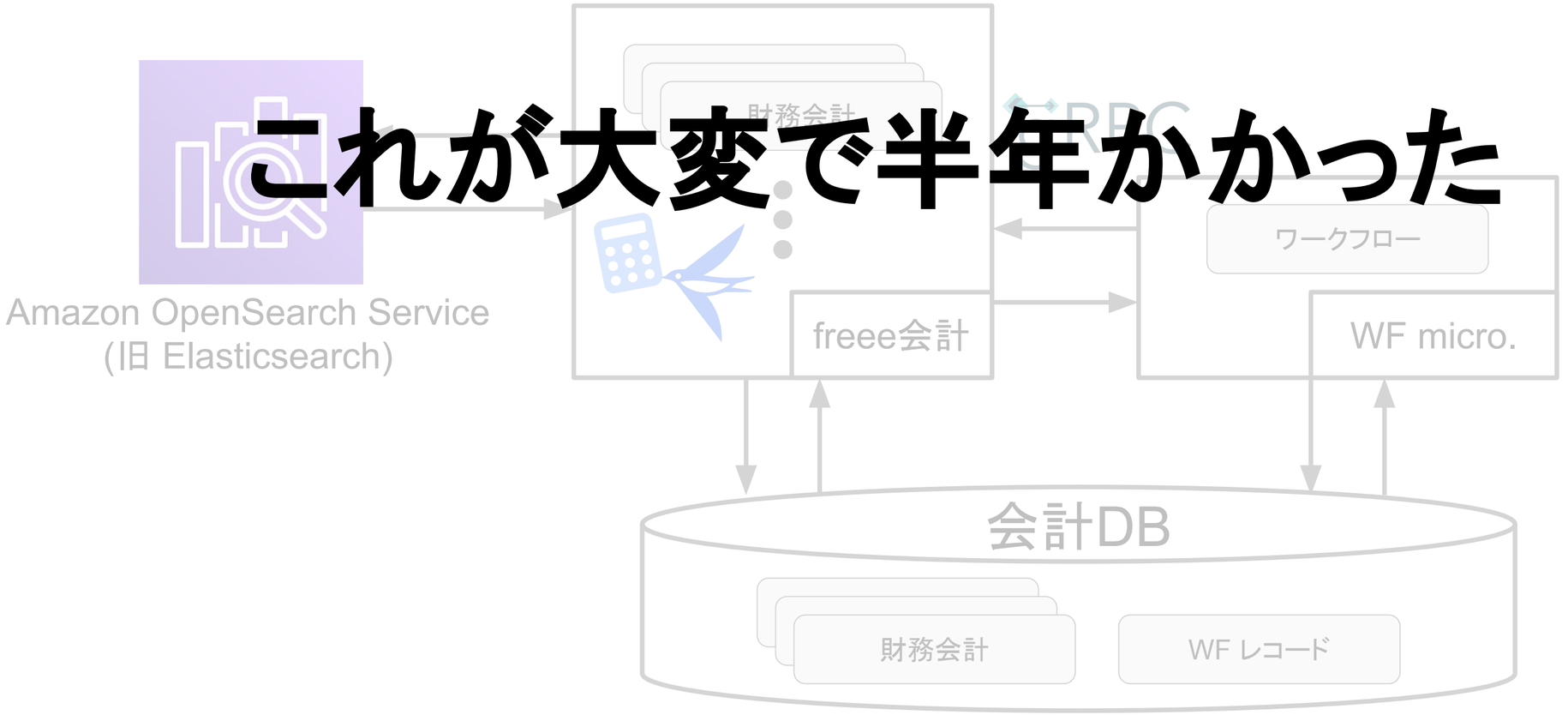
切り出すためにひたすら JOIN 外し



Rails の ActiveRecord により JOIN はコード検索で見つけるのが難しい
ひたすら利用箇所を調べて RPC 呼び出しに置き換える作業に勤しんだ

実は、freeeづくりはこんなにも面白い。
freee 技術の日

ひたすら JOIN 外し



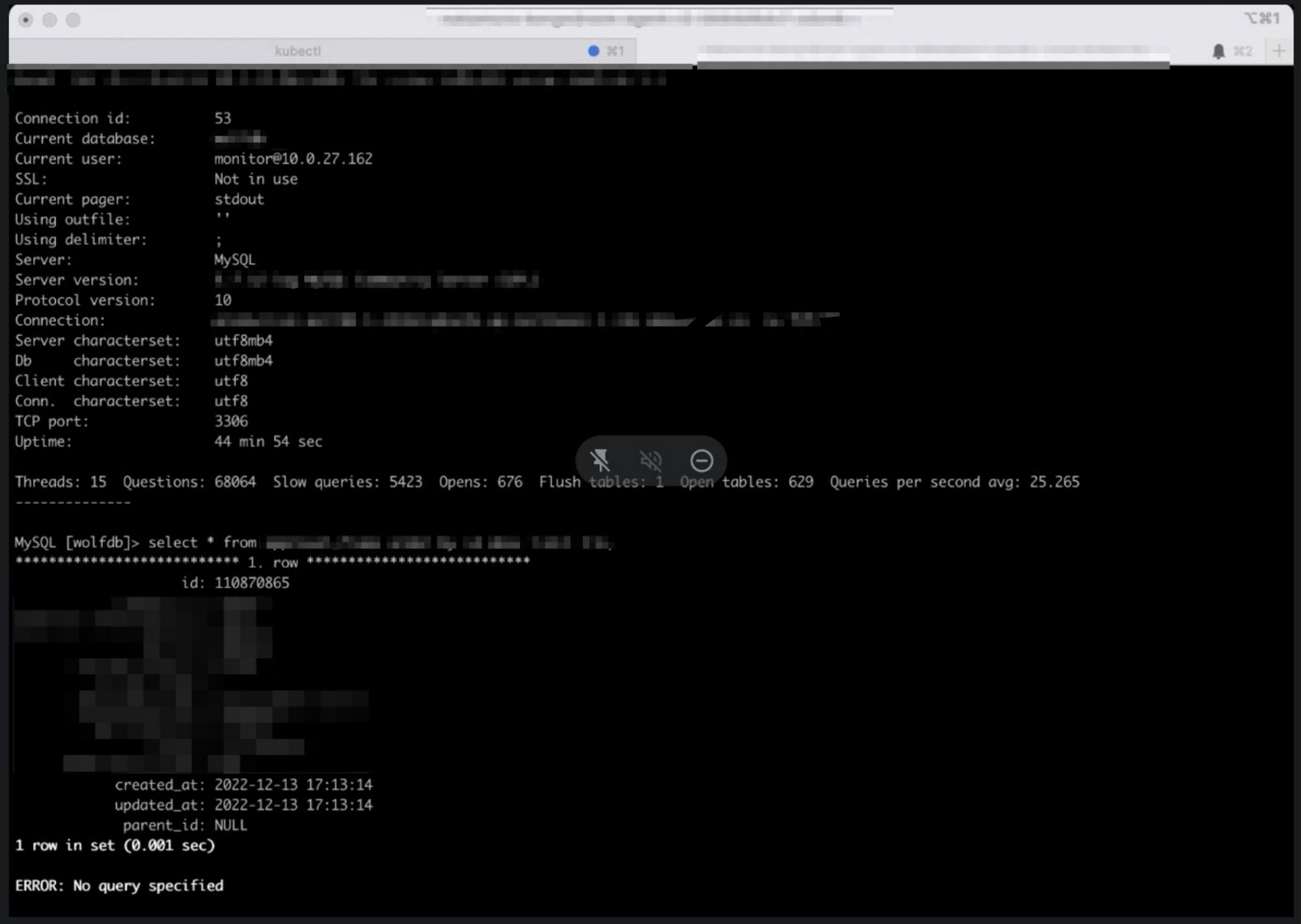
これが大変で半年かかった

JOIN 剥がしが可能になっていたが全然進んでいなかった
ひたすら利用箇所を調べて RPC 呼び出しに置き換える作業に勤しんだ

そしてついに

実は、freeeづくりはこんなにも面白い。
freee技術の日

[redacted] 画面が共有されています



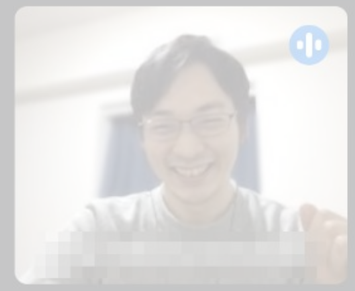
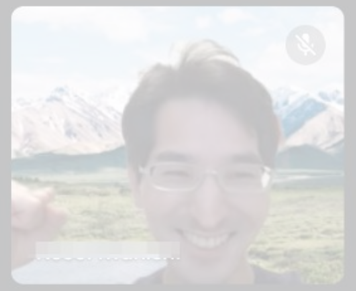
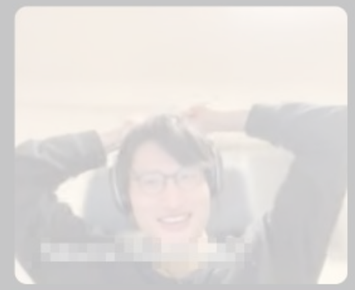
2:30 | 12/14 定期メンテナンス

8

画面が共有されています

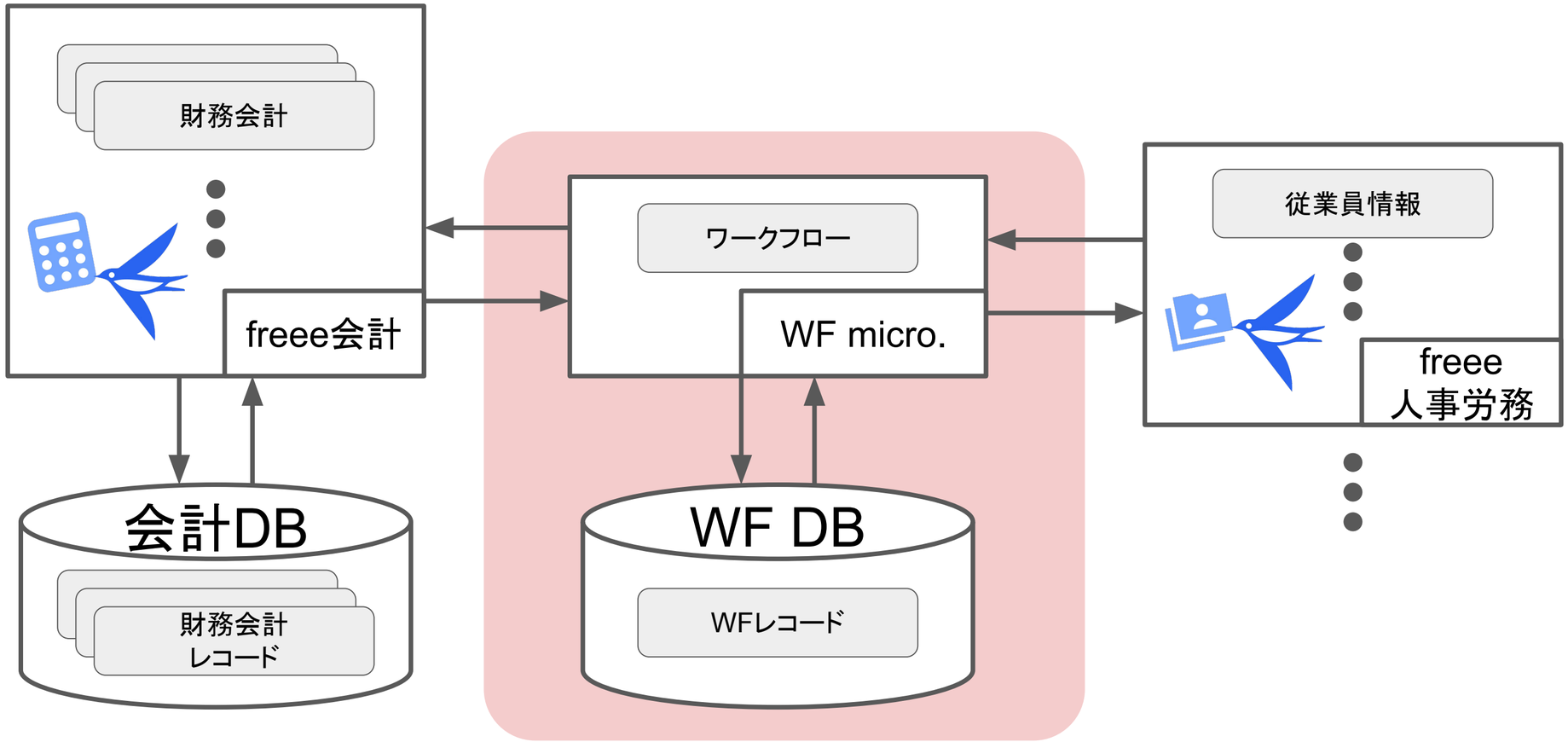
本番 DB を分離成功

```
Connection id:          53
Current database:
Current user:           monitor@10.0.27.162
SSL:                    Not in use
Current pager:          stdout
Using outfile:          ''
Using delimiter:        ;
Server:                 MySQL
Server version:
Protocol version:       10
Connection:
Server characterset:    utf8mb4
Db     characterset:    utf8mb4
Client characterset:    utf8
Conn.  characterset:    utf8
TCP port:               3306
Uptime:                 44 min 54 sec

Threads: 15  Questions: 68064  Slow queries: 5423  Opens: 676  Flush tables: 1  Open tables: 629  Queries per second avg: 25.265
--------------

MySQL [wolfdb]> select * from
*************************** 1. row ***************************
                    id: 110870865

           created_at: 2022-12-13 17:13:14
           updated_at: 2022-12-13 17:13:14
            parent_id: NULL
1 row in set (0.001 sec)

ERROR: No query specified
```

2:30 | 12/14 定期メンテナンス

2022年末 ついに理想状態



ワークフローのロジックが会計と独立

実は、freeeづくりはこんなにも面白い。
freee技術の日

時系列

- 2018年 プロジェクト発足
- 2019年 マイクロサービス完成
- 2020年 検索パフォーマンスの問題が顕著化
- 2021年 Elasticsearch のお世話でてんやわんや
- 2022年 DB負荷対策、DB分離が急務に
- 2022年末 ついに DB分離 完全なマイクロサービスへ

実は、freeeづくりはこんなにも面白い。
freee 技術の日

まとめ

- モノリスからマイクロサービスを切り出していくことは可能
  - 一時的に DB を共有した状態を取り段階的な移行ができる
- マイクロサービス化には投資として取り組める組織づくりが大切
  - ユーザへの恩恵が出るのがちょっと先
  - エンジニアがメリットを説明できる必要がある
- マイクロサービスアーキテクチャにおける
  - データの重複は許容したほうが対処しやすい
  - ズレ起こるもの、リカバリーするもの
- 4年もかかったプロジェクトやりきった感すごい

実は、freeeづくりはこんなにも面白い。
freee 技術の日

今後

- アプリケーションごとに同様の実装が必要になっている
  - 実装を抽象化してマイクロサービスの方で吸収する必要
- WF の体験を freee 内で統一していきたい
  - フロントエンドのコンポーネントの統一も必要
- etc.

課題は山積み、ユーザ価値が最大化できる方法を探っていく必要がある

感謝

- WF の分離に関わったアプリケーションのエンジニアは30名以上
- インフラチームのメンバーも10名以上

ほんとうに、ありがとうございました

利用くださる皆様、興味を持ってくださる皆様

これからの開発にご期待ください

実は、freeeづくりはこんなにも面白い。
freee 技術の日

実は、freeeづくりはこんなにも面白い。

freee技術の日