大学共同利用機関法人 情報・システム研究機構
国立情報学研究所
National Institute of Informatics

トップエスイー ソフトウェア開発実践演習

マイクロサービスによる仕様変更に強いシステム設計

株式会社NTTデータ・アイ 山田淳史
株式会社日本総合研究所 伊藤克也
富士通株式会社 桐山卓弥
株式会社デンソー
株式会社デンソー
近藤敬
三木康暉

開発における問題点

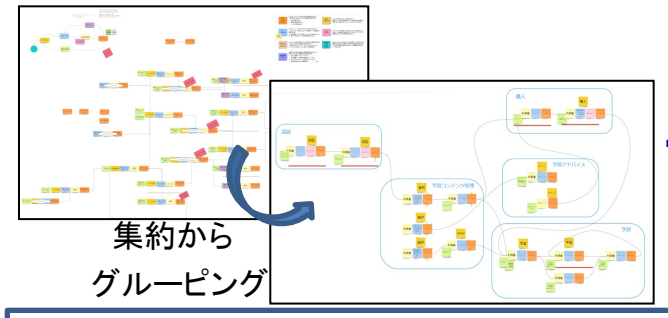
近年，顧客ニーズの多様化に対応するスピードが必要とされているが，従来のモノリスアーキテクチャでは変更容易性が低く，リリースに時間を要する．また一方で，サービスのあるべき姿を導き出すために，ドメインエキスパートの参画が求められているが，従来のモデリング技法では非エンジニアの参画が容易でないという課題がある．

手法・ツールの適用による解決

ChatGPTを取り込んだオンライン学習サービスを題材に，変更容易性およびユーザビリティを考慮した性能面を重視しマイクロサービスアーキテクチャの適用効果を検証した．併せて，検討にあたり，ドメイン設計有識者でなくとも直感的にモデリング可能なイベントストーミングを取り入れることで，初学者でもドメイン設計可能か確認した．

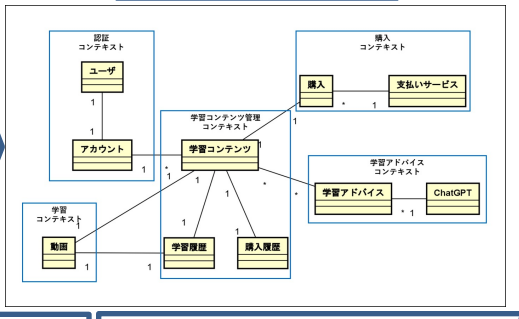
アプローチ

イベントストーミング結果

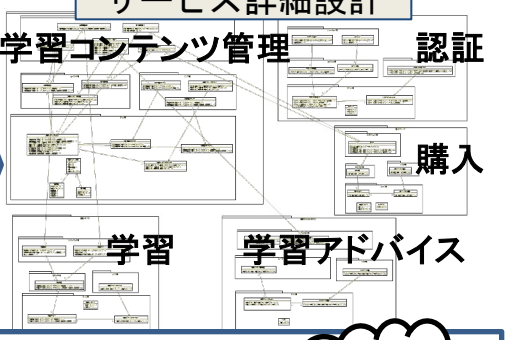


集約からグルーピング

ドメインモデル



サービス詳細設計



初学者でも機械的にコンテキスト分割が可能に

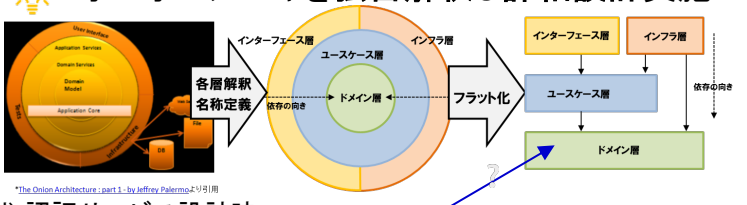
「学習コンテンツ管理」をコアサービスとして定義し，他のコンテキストを専門の機能に特化させる方がマイクロサービスとして有効と考えた

ユビキタス言語表

コミュニケーションツールとして有効

作成して終わりではなく適宜アップデート必要

オニオンアーキを独自解釈し詳細設計実施



*The Onion Architecture : part 1 - by Jeffrey Palermoより引用

例）認証サービス設計時…

【ユースケース】
アカウント作成
ログイン

主なロジック
・重複チェック
・検索
・アカウント規則チェック

重複チェックで全てのアカウントテーブルを展開して確認？
性能面で明らかに劣る・・・（**DDDトリレンマ**）
⇒ドメイン層ではなくインフラ層へ

純粋性
完全性
性能

DDDトリレンマの選択

ユーザビリティを考慮し，“完全性“ を犠牲に “性能“ 向上
すべてを満足させる設計は難しいことを実感

評価・考察

項目	設計結果に対する評価
変更容易性	ドメインモデルの分割及びオニオンアーキテクチャの採用により，修正箇所を限定できた DDDトリレンマへの対応のために完全性を犠牲にした
性能	コアービスへの柔軟なリソース配置により性能確保が可能な設計になった 処理の効率的な配分により性能確保できた（DDDトリレンマ対応） サービス間の連携がオーバーヘッドになる可能性がある

実プロジェクトへの適用に向けた課題

■概念モデルの検討プロセス
・ドメインを抽象的な概念でとらえきれず設計者視点で表現されてしまった
ドメインエキスパートが必要

■組織設計見直し（コンウェイ・逆コンウェイの法則）
・既存組織が設計成果に影響を与えるため、コンテキスト分割から組織設計の見直しを検討するべき
イベントストーミングが有用