

活動のキッカケ

効率化

- AIで楽できるかな？
- 要求仕様やテストドキュメントでAIをどう使えば効率化できる？
- 中身はあるが、うまく表現できない→日本語を直してくれる？

みんなが楽できる
プロンプトガイドラインを
思考中

対話

- AFFORDD Bot
- FAQの自動化
- 思考を鍛えたい

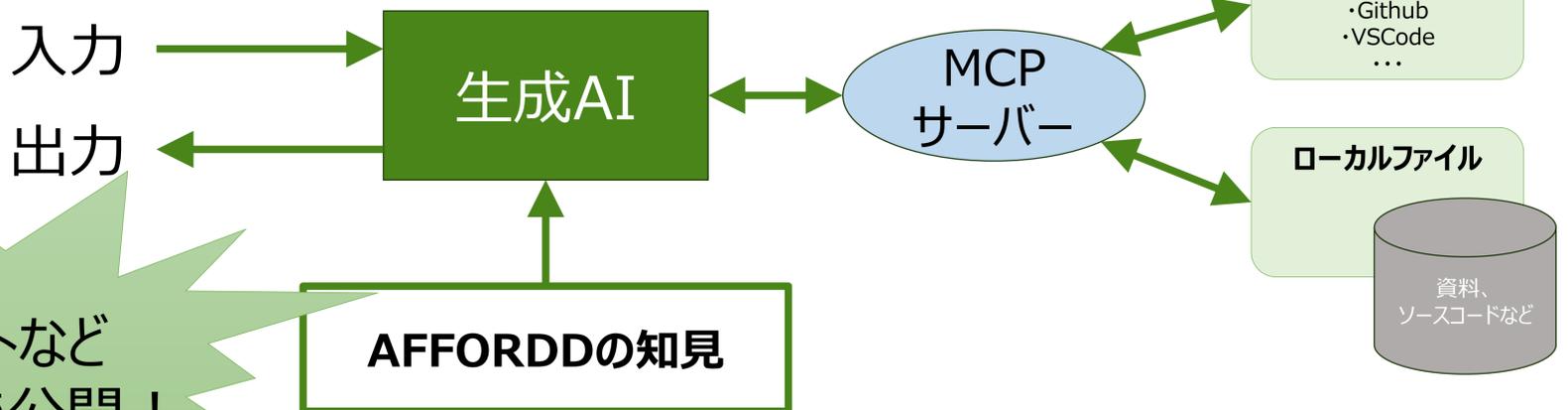
Fine-tuningやRAGは
環境や維持の面で
難しい(様子見)

AI搭載製品

- AIを使ったシステムに対する要求仕様ってどう書くの？
- **品質はどこまで保証するの？**

ここまで活動に**至っていない**

AFFORDDの知見を活用し、効率化を試みた



プロンプトなど
会員限定で公開！

試行した生成AI

メインは以下の2つ

 **ChatGPT** OpenAI

- GPT-4o
- GTP-o3

 **Claude** Anthropic

- 3.7 sonnet
- Extended thinking mode
- MCPサーバー

一部でやってみた

 **Gemini**
Advanced 2.5Pro

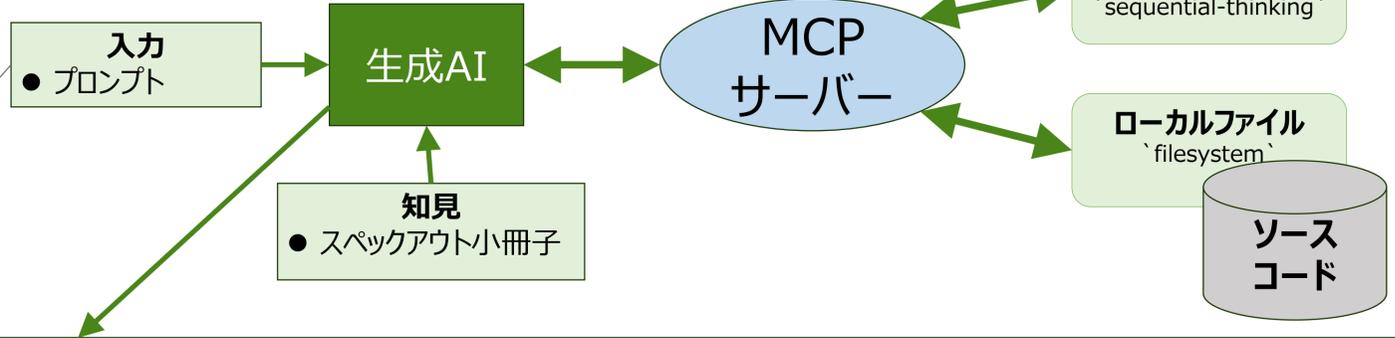
今後は、生成AI時代の派生開発プロセスを探求していく

スペックアウト生成 T19研究会の知見を活用!

あなたはソフトウェア開発においてソースコードを理解する専門家です。"スペックアウト小冊子_00.pdf"を熟知しています。以下の特定の箇所についてスペックアウト資料を作成します。

特定の箇所: 計算機を実行すると、数字を入力すると値に応じてUnit_131:手(👏)から、Unit_107:ジャンボジェット(✈️)に切り替わります。

- ...
- ...
- ...



出力

ChatGPT ※ChatGPTはMCPサーバなし

処理構造のスペックアウト

単位切り替えは UnitConverter クラスが担当し、IConverterDataLoader インターフェースに基づきます。

Mermaid クラス図

```

classDiagram
    class UnitConverter {
        +UnitConverter(dataLoader, currencyDataLoader)
        +LoadData()
        +Convert(value, fromUnit, toUnit)
    }
    class IConverterDataLoader {
        +LoadUnits()
        +LoadCategories()
    }
    class Unit {
        +int id
        +string name
    }
    UnitConverter --> IConverterDataLoader
    UnitConverter --> Unit
    
```

データ構造のスペックアウト

Unit 構造体は単位ID、名前、略称、変換フラグ、定義されます。

Mermaid ER図

```

erDiagram
    UNIT ||--o{ UNIT : "has"
    UNIT {
        INT id
        STRING name
        STRING abbreviation
        STRING isConversionSource
        STRING isConversionTarget
        BOOLEAN isWhimsical
    }
    
```

制御構造のスペックアウト

状態遷移は、数値入力後に該当する単位を判別し、条件に基づいて行われます。

Mermaid 状態遷移図

```

stateDiagram-v2
    [*] --> WaitingForInput
    WaitingForInput --> ProcessingInput : numberEntered
    ProcessingInput --> SelectingUnit : checkValue
    SelectingUnit --> Unit_131 : valueInRangeForHand
    SelectingUnit --> Unit_107 : valueInRangeForJet
    
```

処理のシーケンス図

数字入力から結果表示までの流れを示します。

Mermaid シーケンス図

```

sequenceDiagram
    participant User
    participant CalculatorUI
    participant UnitConverter
    participant UnitData
    User->>CalculatorUI: 入力値を入力
    CalculatorUI->>UnitConverter: Convert(value)
    UnitConverter->>UnitData: 対象Unitを判定  
Unit_131 または Unit_107
    UnitData-->>CalculatorUI: 単位情報を返却
    CalculatorUI-->>User: 結果表示 (👏 または ジャンボジェット)
    
```

Claude

計算機アプリの単位変換時の「奇妙な単位」表示機能のスペックアウト

概要

計算機アプリの単位変換機能では、数値を入力すると通常の変換結果に加えて「補足的な変換結果 (Supplementary Results)」が表示されます。このなかには、「手(👏)」や「ジャンボジェット(✈️)」といった「奇妙な (whimsical)」単位も含まれており、入力値の大きさに応じて表示される単位が自動的に切り替わります。

処理構造のスペックアウト

まず、単位変換の基本的な処理構造を図式化します。

Mermaid シーケンス図

```

sequenceDiagram
    participant UI as ユーザーインターフェース
    participant VM as UnitConverterViewModel
    participant UC as UnitConverter
    UI->>VM: 数値入力
    VM->>UC: SendCommand()
    UC->>VM: Calculate()
    UC->>VM: CalculateSuggested()
    VM->>UC: SuggestedValueCallback(suggestedValues)
    UC->>VM: UpdateSupplementaryResults()
    VM->>UC: RefreshSupplementaryResults()
    UC->>VM: 通常の単位と奇妙な単位を分離
    UC->>VM: 奇妙な単位は最大1つだけ追加
    VM->>UI: 補足的な変換結果の表示更新
    
```

全体構造図

単位変換機能と「奇妙な単位」表示の全体的な構造を图示します。

1. 数値入力
2. 単位選択
3. 単位変換
4. 奇妙な単位を分離
5. 奇妙な単位を選択
6. 補足的な変換結果の表示更新
7. 補足的な変換結果の表示更新

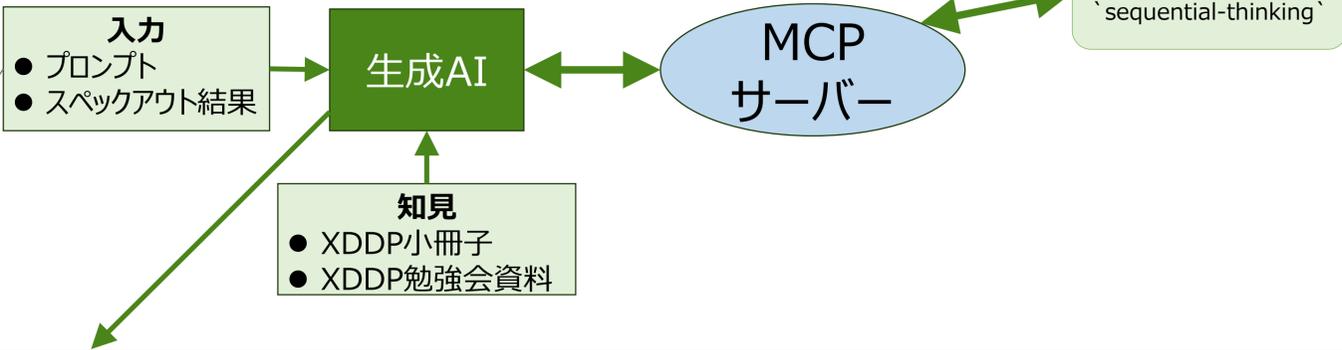
LLMに与える「知見」によって生成結果が大きく異なる

XDDP3点セット生成 T3研究会の知見を活用!

あなたはソフトウェアの変更を行う専門家です。「XDDP小冊子.pdf」を熟知しています。以下の変更要求をXDDPのプロセスに従って3点セットの成果物を作成します。

変更要求: 新たな単位として「東京タワー」を追加し、電卓の数字に応じて「ジャンボジェット機、何個分」から「東京タワー、何個分」切り替わる。

- 変更要求仕様の作成
 - 1-1. スペックアウト結果を元に、「変更要求」について、「変更要求仕様_TM付き記述例.md」を参考に変更要求仕様を1つの表で作成してください。
 - 1-2. ...



出力

ChatGPT ※ChatGPTはMCPサーバなし

要求仕様: LENGTH-001 - 長さ単位の追加 (東京タワー)

- 要求: システムは、長さの単位に「東京タワー (333メートル)」を追加し、単位変換およびUI表示で扱えるようにする。
- 理由: ユーザーに親しみやすく、視覚的にイメージしやすい遊び単位を提供することで操作体験を向上させる。
- 説明: 「東京タワー」は既存の遊び単位 (手やジャンボジェット) と同様に、長さカテゴリに属し、換算・表示対象とする。

Before / After

観点	Before (変更前)	After (変更後)
単位定義構造	Unit構造体にはid=131, 107など既存の遊び単位のみ	Unit構造体にはid=999 (東京タワー) が追加
初期化処理	InitializeUnits()に東京タワーの登録なし	InitializeUnits()で東京タワーUnitが構築・登録される
変換処理	Convert()は東京タワー換算を扱わない	Convert()は333mを東京タワー換算 (およびその逆) で処理
UI表示	単位選択画面に東京タワーが表示されない	単位選択リストに東京タワーが表示され、選択・表示可能となる
定義ファイル	単位定義ファイルに東京タワーの登録なし	UnitData.json等にid名称/換算係数/

変更要求TM (東京タワー単位追加)

変更要求仕様 (行)	Unit.h	UnitConverter.cpp	UnitConverter.h	UnitData.json	UnitSelectionView.cpp
1 LENGTH-001.001 単位定義構造に「東京タワー (id=999)」を追加する	●				
2 LENGTH-001.002 初期化処理で東京タワー単位を構築・登録する		●			
3 LENGTH-001.003 単位変換処理で333m換算を可能にする		●	●		
4 LENGTH-001.004 UIに「東京タワー」選択肢を追加する					●
5 LENGTH-001.005 単位定義ファイルに情報追加				●	

Claude

変更要求仕様書 (TM付き)

ID	内容	UnitConverterDataConstants.h	UnitConverterDataLoader.cpp	UnitConverterDataLoader.h	UnitConverter.cpp	UnitConverter.h
UNIT001	長さ単位として「東京タワー」を追加する。単位変換・表示処理も追加する。					
UNIT002	長さ単位として「東京タワー」を追加する。単位変換・表示処理も追加する。					
UNIT003	長さ単位として「東京タワー」を追加する。単位変換・表示処理も追加する。					
UNIT004	長さ単位として「東京タワー」を追加する。単位変換・表示処理も追加する。					
UNIT005	長さ単位として「東京タワー」を追加する。単位変換・表示処理も追加する。					

変更設計書

UnitConverterDataConstants.hの変更設計書

変更設計書: 計算機アプリ 作成日: 2025年4月14日
 一人名/ UnitConverterDataConstants.h 作成者: c修正
 スク名 修正者: c修正
 変更要求仕様: 単位列挙型定義に「東京タワー」の 確認者: c確認
 識別子を追加する
 NIT01.1 変更行数: 1 見積り時間: 5分
 作業時間: 5分

UnitConverterDataLoader.cpp - GetUnits関数の変更設計書

変更設計書: 計算機アプリ 作成日: 2025年4月14日
 一人名/ UnitConverterDataLoader.cpp 作成者: c修正

人が間違いやすい「トレース」を正確に生成してくれる

USDM生成 T2研究会の知見を活用！

あなたは要求仕様定義の専門家です。USDMの要求仕様の書き方「afford-t2-usdmtext-basic_1.3.pdf」を熟知しています。

以下のUSDMの例を参考にUSDMを作成します。

売上分析 (USAGE)	要求	URIAGE-01	理由	説明
			画面のドロップダウンリストで店舗を指定するところから、店舗コードを取得し、取得した店舗コードを右側の売上テーブルを検索し、検索した結果を編集し、編集したデータを右側の売上テーブルに書き戻すことを行う。	
			店舗ごとの売上と利益を知りたい。それで売上が少ない店舗を対策するために立てたい。売れ筋の商品を知りたい。	
			説明	実施時期は、10月ごろ。
			<ドロップダウンリストの	

入力

- プロンプト
- ユースケース記述

生成AI

MCPサーバー

外部サービス
単なる生成の場合は利用無し

外部サービス
Notion

知見

- USDM小冊子
- USDMマナー一覧 (佐藤さん作)

出力

ChatGPT ※ChatGPTはMCPサーバなし

要求仕様: VISUAL-001 - 数値に応じた遊び単位の自動切り替え

要求

- ID: VISUAL-001
- 要求: システムは、入力された数値に応じて表示単位を「手 (Unit_131)」または「ジャンボジェット (Unit_107)」で自動的に切り替えて表示する。
- 理由: 数値に基づいて視覚的なスケール感を動的に変化させることで、ユーザーの直感的理解と体験価値を高める。
- 説明: 本要求は、スペックアウト工程において明らかになった既存実装の仕様から導出された要求であり、新たな機能追加ではなく既存機能の明示的な要求化である。

要求と仕様のトレーサビリティ (階層表)

レベル	ID	内容	対象ファイル / クラス / 関数	備考
要求	VISUAL-001	数値に応じて遊び単位を切り替えて表示する	-	スペックアウトにより顕在化した要求
L 仕様	VISUAL-001.001	数値入力に基づいて単位IDを選択する処理	UnitConverter.cpp / UnitConverter.Convert()	条件分岐によるID切り替え
L 仕様	VISUAL-001.002	切り替えた単位IDに基づくUI表示更新	CalculatorUI.cpp / UpdateResultDisplay() / UnitSelectionView.cpp / RenderUnits()	表示切り替え
L 仕様	VISUAL-001.003	isWhimsical属性による単位分類	Unit.h / Unit構造体	遊び単位 (例: 手、ジャンボジェット) に分類

Claude

markdown形式で生成

歩数計 USDM要求仕様書

カテゴリ名 (記号)	要求ID	要求理由/仕様	内容
歩数計 (WALK)	WALK-01	要求	システムは加速度センサーを使用してユーザーの動きを検知し、歩行動作を識別して歩数をカウントし、リアルタイムでデータを更新する。
		理由	ユーザーの健康管理のために正確な歩数データが必要であるため。また、日常の運動量を可視化することで健康意識の向上につながるため。
		説明	歩数計の基本機能。バックグラウンドでも継続的に動作する。
		<動きの検知>	
	WALK-01.001		加速度センサーからのデータを100msごとにサンプリングする。
	WALK-01.002		センサーデータから上下方向の加速度変化を抽出する。
		<歩行の判定>	
	WALK-01.010		上下方向の加速度変化が歩行パターンに一致した場合、歩行と判定する。
	WALK-01.011		歩行パターンの判定基準は以下の通りとする。 - 動きの周波数: 0.80~1.50Hzの間 - 動きの強度: 0.80~3.00の間
	WALK-01.012		明らかに歩行でない動き (振り物での揺れなど) を検出した場合、カウントしない。
		<歩数のカウント>	
	WALK-01.020		歩行と判定された場合、歩数カウンタを1増加させる。
	WALK-01.021		歩数データは30分に1回システムストレージに保存する。
	WALK-01.022		日付が変わった時点 (午前0時) で、日次の歩数集計を行い、履歴データとして保存する。
		<バッテリー消費の最適化>	
	WALK-01.030		バッテリー残量が20%未満になった場合、サンプリング頻度を200msに変更する。
	WALK-01.031		バッテリー残量が10%未満になった場合、サンプリング頻度を500msに変更する。
歩数計 (WALK)	WALK-02	要求	システムは計測した歩数をユーザーが確認できるように画面に表示する。
		理由	ユーザーが自分の活動レベルをリアルタイムで確認できるようにするため。アプリのメイン画面で最も目立つ位置に表示する情報。
		説明	
		<歩数の表示>	
	WALK-02.001		画面中央に今日の累計歩数を大きなフォントで表示する。
	WALK-02.002		- 3時ごとにカンマ区切り (例: 12,345歩) - 「歩」の単位を追加
	WALK-02.003		歩数表示は1秒ごとに更新する。

Notionのタスクとして生成

USDM要求仕様

VIEW-01

<動き検知>

要求と仕様の関係が正確。理由の生成がうれしい

PFD生成 勉強会の知見を活用！

#PFD作成プロンプト
あなたはプロセス設計の専門家です。PFD (プロセスフローダイアグラム) を十分に理解し、PFDの3点セット (プロセスフローダイアグラム、プロセス定義書、成果物定義書) を作成します。添付の問題を理解し、...

#PFDのWebシミュレーションプロンプト
1. 添付したファイルはプロセスフローダイアグラム (PFD) で、成果物とプロセスの連鎖を表現しています。ダイアグラムの成果物は成果物定義書に、プロセスの定義はプロセス定義書に記述しています。このPFDに対し、...

入力

- プロンプト
- プロセスの問題

生成AI

MCPサーバー

外部サービス
利用無し

知見

- PFD勉強会資料

出力

ChatGPT

プロセスフローダイアグラム

成果物間の関係

成果物定義

ID	成果物名 / ファイル名	目的	編集方針	構成 (Cardinality)
D1	会議開催案内 (mail/PDF)	会議情報の事前共有	発行後は変更不可	会議名、開催日時・場所・オーナー・参加者
D2	議事録テンプレート (.docx)	体裁統一	管理責任者のみ改変	開催概要・決定事項・アクションアイテム・議事詳細・承認欄
D3	会議動画 (mp4)	内容の客観記録	録画後一括記録・編集不可	音声・映像
D4	会議中メモ (txt/xls)	発言メモ	会議中のみ追加	1..N (発言者、発言内容)
D5	議事録 (ドラフト)	会議内容の周知	承認前は修正可	開催概要・決定事項・0..Nアクションアイテム・議事詳細
D6	議事録承認リスト	レビューコメント受付	全箇所がCloseされるまで更新可	1..N (制作者ID、箇所ID、対応状況)
D7	議事録 (最終版)	正式記録	承認後は修正不可	D5 + 承認欄

プロセス定義

No	プロセス名	詳細作業	入力	出力	実施条件 / 実行	担当者	完了判定
P1	開催概要を決定	開催概要を決定	D1, D2	D3 (結果入力)	D1が実行済	事務局	D5開催概要が決定
P2	文字起こし	動画・メモを聞き取り、発言内容を詳細録へ追記	D3, D4	D5更新	D4更新	事務局	全発言が追記
P3	議事録作成	決定事項を抽出しD5決定事項を完成	D4, D5更新	D6更新	D5更新	事務局	決定事項欄が完成
P4	アクション確認	次回までの決定事項・担当者を抽出しD5アクションアイテムへ反映	D4, D5更新	D6更新	D5更新	事務局	アクション欄が完成
P5	議事録承認	発言者・発言内容を格納しD5議事録を完成	D4, D5更新	D6更新	D5更新	事務局	詳細欄が完成
P6	承認依頼	D5の承認・内容・体裁チェック、依頼を一覧化	D5, D6更新	D7更新	D6更新	事務局	D6が実行
P7	承認確認	D6を全てクローズするまで修正	D5, D6更新	D7更新	D6更新	事務局	D6が完了
P8	承認完了	決定事項・承認済み会議録を出力	D5, D7更新	D7更新	D7更新	事務局	D7承認
P9	保存						

Claude

プロセスフローダイアグラム

シミュレーション生成

成果物定義書: D1 議事録

プロセス定義書: P1 会議開催情報を記入する

成果物間の関係生成

ドラフトを一瞬で作れる
PFDから業務に効果的なアウトプット生成が可能