

# Generative AI の ソフトウェア開発への 応用の試行

酒匂寛

[sako.hiroshi@gmail.com](mailto:sako.hiroshi@gmail.com)

Designers' Den Corp.

# 内容

- Generative AI の現状
- ソフトウェア開発に直接役立つ利用法
- 「伝えること」と「翻訳すること」
- より開発上流で役立つ利用法
- プロンプトエンジニアリング
- 考察とまとめ

# Generative AI の現状

- 新しい言語モデル
  - 昨年末に OpenAI が ChatGPT を発表、文脈を意識した高度な対話を可能にした。無料で提供されていることもあり一気にブレイク
  - API (ChatGPT API) も発表されて、さまざまなアプリケーションへの組み込みが始まった
- テック企業各社はそれぞれの取り組みを行っている
  - Microsoft：検索エンジン Bing への OpenAI の GPTの取り込み
  - Google：独自の言語モデル LaMDA (PaLM2) を組み込んだチャット Bard
  - DeepMind：チャットAI の Sparrow を開発中であると報じられる
  - Facebook：言語モデル LLaMA を研究者向けに提供

# Generative AI を使った様々なツール例（1）

- **ChatGPT**

- 単純にインタラクティブに使う以外に、ツールに組み込んで使うこともできる。**Generative AI**ブームの立役者。
- これまで**2021年9月**までの知識しか使えなかったが、新しいプラグイン機能によって**最新知識**にもアクセスできるようになった（しかし遅いし不安定）

- **Bing Chat**

- **Microsoft** が **Edge** ブラウザに組み込んだ **Generative AI** チャット。**ChatGPT**相当の能力にリアルタイムネット検索機能を持っている

- **Bard**

- **Google** が作成したチャットボット、動作は軽快。**ChatGPT**とほぼ同様の能力を持つ予定だが、公開されてまだ日が浅いためかいろいろと不都合がみうけられる

## Generative AI を使った様々なツール例（2）

- **GitHub Copilot**
  - VSCodeに組み込んで利用する。
  - 様々なコーディングアシスタント機能を提供する
- **NotionAI**
  - 多機能メモツール Notion に AI（OpenAI, Anthropic）を使った執筆、要約、翻訳機能などが付け加わったもの
- **Perplexity**
  - AI 付き検索ツール、検索結果をコンパクトにサマリーして答えてくれる。同時にその検索結果に関する証拠を示すためのリンクも示されるので事実を検索する際には ChatGPT よりも有用。
- **Etc ...** 雨後の筍のようにたくさんのツールが生まれている

# 直接的な利用法

ソフトウェア開発を例として

# コーディングアシスト (1/3)

ChatGPTにコード生成を依頼

【人間】 指定したURLにアクセスして、そのページのタイトルを出力する Python のコードを作成

----- (ここから下↓がGenerative AIの出力。以下同じ)

以下はPythonのrequestsとbeautifulsoup4ライブラリを使用して、指定したURLにアクセスし、そのページのタイトルを出力するコードです。

```
``python
import requests
from bs4 import BeautifulSoup

url = "https://www.example.com"
response = requests.get(url)
soup = BeautifulSoup(response.content, "html.parser")
title = soup.title.string

print(title)
``
```

まず、requestsライブラリをインポートしてURLにアクセスします。そして、beautifulsoup4ライブラリを使ってHTMLページから情報を抽出します。ここで、response変数にHTMLページの内容が格納され、BeautifulSoupオブジェクトのsoupを作成して、そのタイトルを取得し、最後に出力します。

コードが生成されるだけでなく、その説明も同時に生成されている

この「サービス精神の旺盛さ」が ChatGPT の特徴の一つ

# コーディングアシスト (2/3)

URLが固定だったので、引数として与えられるように依頼

【人間】 url をコマンドラインから取得できるようにして下さい

以下は、Python の requests ライブラリを使用して、指定された URL からページタイトルを取得し、出力するコード例です。また、sys ライブラリを使用して、コマンドラインから url を受け取ります。

```
``python
import sys
import requests
from bs4 import BeautifulSoup
```

```
# コマンドラインから url を受け取る
url = sys.argv[1]
```

```
# requests ライブラリを使用して、HTML ページを取得する
response = requests.get(url)
```

```
# BeautifulSoup ライブラリを使用して、HTML ページからタイトルを抽出する
soup = BeautifulSoup(response.text, 'html.parser')
title = soup.find('title').text
```

```
# タイトルを出力する
print(title)
``
```

このコードを `get_title.py` という名前で保存し、コマンドラインから次のように実行できます。

```
...
python get_title.py https://www.google.com/
```

引数の取り込みが追加された

# コーディングアシスト (3/3)

URLがないときはエラーで終了するように依頼

【人間】 指定した URL が存在しないときにはエラーメッセージを出して終了するようにして下さい。

以下に、指定した URL からタイトルを取得する Python コードを示します。

```
``python
import sys
import requests
from bs4 import BeautifulSoup

# コマンドライン引数から URL を取得する
url = sys.argv[1]

# URL にアクセスして HTML を取得する
response = requests.get(url)

# ステータスコードが 200 以外の場合はエラーメッセージを出力して終了する
if response.status_code != 200:
    print(f"Error: {response.status_code}")
    sys.exit()

# BeautifulSoup を使って HTML からタイトルを取得する
soup = BeautifulSoup(response.content, 'html.parser')
title = soup.title.string

# タイトルを出力する
print(title)
``
```

引数が足りない場合は考えていない

URLが存在しないときという指定を拡大解釈して対処している

このコードでは、まず `sys.argv[1]` を使ってコマンドラインから URL を取得し、`requests.get()` を使ってその URL にアクセスして HTML を取得します。次に、`response.status\_code` をチェックして、ステータスコードが 200 でない場合にはエラーメッセージを出力してプログラムを終了します。最後に、`BeautifulSoup` を使って HTML からタイトルを取得し、それを出力します。

# レビューアシスト

先のコードに問題がないかレビューを依頼、問題があれば指摘し修正案を出す

【人間】 Find problems with the following code

-----  
このコードにはいくつかの問題があります。エラー処理や例外処理が不足しています。  
以下は、修正したコードです。修正点についてはコメントで説明しています。

```
```python
import sys
import requests
from bs4 import BeautifulSoup

# コマンドライン引数から URL を取得する
try:
    url = sys.argv[1]
except IndexError:
    print("Error: URL not specified")
    sys.exit()

# URL にアクセスして HTML を取得する
try:
    response = requests.get(url)
except requests.exceptions.RequestException as e:
    print(f"Error: {e}")
    sys.exit()

# ステータスコードが 200 以外の場合はエラーメッセージを出力して終了する
if response.status_code != 200:
    print(f"Error: {response.status_code}")
    sys.exit()
```
```

# BeautifulSoup を使って HTML からタイトルを取得する

```
soup = BeautifulSoup(response.content, 'html.parser')
title = soup.title.string
```

# タイトルを出力する

```
print(title)
```
```

変更内容：

- \* `url` の取得部分で `IndexError` が発生する可能性があるため、`try-except` 文で例外処理を追加しています。
- \* `requests.get()` の例外処理を追加しています。
- \* ステータスコードが200以外の場合はエラーメッセージを表示し、`sys.exit()` を使用してプログラムを終了するようにしました。

# ここまでの手順

- 仕事を依頼
  - ここではプログラムの生成を依頼
- 仕事の仕様変更を依頼
  - 引数の追加
- 仕事の仕様変更を依頼
  - 例外処理の追加
- 結果のレビューを依頼
  - 問題点を指摘しさらに処理を追加

生成

ChatGPTの優れている（面白い）点は、対話を重ねて答を生成していただけること

内省

生成した結果を評価させることもできる

# 注意

- ここまで調子よく進んできたが ...
- ここで作られたプログラム（成果物）は正しいかどうかは保証されない
- なので最終成果物（中間成果物も）の正しさは人間が確認しなければならない

これは常に気をつけておかなければならないこと

# 解析アシスト (1)

形式仕様記述言語VDM-SL のコードの解析を依頼

【人間】以下の VDM-SL のコードを説明して下さい。

types

金額型 = int;

商品券型 ::

商品券媒体 : 商品券媒体型

商品券額面 : 金額型;

商品券媒体型 = <電子> | <紙>;

商品券セット型 ::

商品券媒体 : 商品券媒体型

商品券構成 : set of 商品券型

inv ts ==

ts.商品券媒体 = <電子> => forall t in set ts.商品券構成 & t.商品券媒体 = <電子>

and

ts.商品券媒体 = <紙> => forall t in set ts.商品券構成 & t.商品券媒体 = <紙>;

# 解析アシスト (2)

【人間】プログラミングに関する知識がない業務専門家にわかるように、説明して下さい。

このコードは、商品券に関する情報を整理するためのルールを定義しています。以下の4つの要素が含まれています。

- **金額:** これは、お金の金額を表します。
- **商品券:** これは、商品券に関する情報を表すための要素で、2つの部分から構成されています。
  - **商品券のタイプ:** 電子タイプの商品券か、紙のタイプの商品券かを示します。
  - **商品券の金額:** 商品券の価値（お金の金額）を示します。
- **商品券のタイプ:** これは、商品券が電子タイプか紙タイプかを示す要素です。
- **商品券のグループ:** これは、同じタイプの商品券をまとめたもので、2つの部分から構成されています。
  - **商品券のタイプ:** グループ内のすべての商品券のタイプを示します（電子か紙か）。
  - **商品券のリスト:** グループに含まれる商品券の一覧を示します。

このコードには、**商品券のグループに関する特別なルール**が定義されています。このルールは、商品券のグループ内のすべての商品券が同じタイプであることを保証します。例えば、商品券のグループが電子タイプの場合、そのグループ内のすべての商品券は電子タイプでなければならないということです。同様に、商品券のグループが紙タイプの場合、そのグループ内のすべての商品券は紙タイプでなければならないということです。

# 解析アシスト (3)

【人間】 この VDM-SL を形式仕様として受け取って実際のプログラミングを行う **エンジニア向け** の解説を行って下さい。

この VDM-SL 形式仕様は、商品券に関するデータ構造を定義しています。プログラマーとして、この仕様を実際のプログラムに変換する際に、以下のデータ構造と制約を考慮する必要があります。

- **金額型:** これは整数型で、金額を表します。プログラムで金額を扱う際に、整数型を使用してください。
- **商品券型:** これは構造体で、商品券の情報を表します。プログラムでは、クラスや構造体を使用して、以下の2つのプロパティを持つオブジェクトを定義します。
  - **商品券媒体:** 商品券が電子媒体か紙媒体かを表す列挙型を定義します。例えば、C++ではenum class、Javaではenum、PythonではEnumを使用します。
  - **商品券額面:** 金額型（整数型）を使用して、商品券の額面金額を表します。
- **商品券セット型:** これは構造体で、同じ媒体の商品券のセットを格納します。プログラムでは、クラスや構造体を使用して、以下の2つのプロパティを持つオブジェクトを定義します。
  - **商品券媒体:** 商品券媒体型を使用して、商品券セット内のすべての商品券の媒体を表します。
  - **商品券構成:** 商品券型のコレクション（リスト、セット、配列など）を使用して、商品券セットに含まれる商品券のリストを表します。

前のスライドでは「**商品券のグループに関する特別なルール**」と書かれていた

この仕様には、**商品券セット型に対する制約（不変条件）**が定義されています。この制約を遵守するために、商品券セットに商品券を追加する際には、追加しようとする商品券の媒体がセットの媒体と一致することをチェックするメソッドや関数を実装してください。例えば、クラスのメソッドやコンストラクタ内でチェックを行い、不一致があれば例外を投げるようにします。

実際にプログラムを実装する際は、使用するプログラミング言語の機能や慣習に従って、これらのデータ構造と制約をコードに反映させてください。

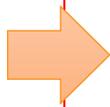
# 解析アシスト (4)

```
import sys
import requests
from bs4 import BeautifulSoup
from urllib.parse import urlparse

def is_valid_url(url):
    try:
        result = urlparse(url)
        return all([result.scheme, result.netloc])
    except ValueError:
        return False

def get_page_title(url):
    try:
        response = requests.get(url)
        if response.status_code == 200:
            soup = BeautifulSoup(response.content, "html.parser")
            title = soup.title.string if soup.title else "No title found"
            return title
        else:
            return f"Error: {response.status_code}"
    except requests.RequestException as e:
        return f"Error: {str(e)}"

if __name__ == "__main__":
    if len(sys.argv) > 1:
        url = sys.argv[1]
        if is_valid_url(url):
            title = get_page_title(url)
            print(title)
        else:
            print("Error: Invalid URL")
    else:
        print("Usage: python get_page_title.py <URL>")
```



This Python script is used to fetch and display the title of a web page given its URL. The script checks if the provided URL is valid, retrieves the HTML content of the page, and then extracts the title using the BeautifulSoup library. If the script is run without providing a URL or if the provided URL is invalid, it will display an error message.

Here's a step-by-step explanation of the code:

1. Import the necessary libraries: `sys`, `requests`, `BeautifulSoup`, and `urllib.parse`.
2. Define a function `is_valid_url(url)` to check whether a given URL is valid or not.
3. Define a function `get_page_title(url)` to fetch the web page title for a given URL.
4. Check if the script is being run as the main module. If it is, proceed to the next steps.
5. Check if a command-line argument (URL) is provided. If not, print the correct usage message.
6. If a URL is provided, check whether it's valid using `is_valid_url()`. If the URL is invalid, print an error message.
7. If the URL is valid, get the web page title using `get_page_title()` and print it.

Usage:

To use this script, run the following command in your terminal:

```
mathematica
```

Copy code

```
python get_page_title.py <URL>
```

Replace `<URL>` with the web page URL you want to fetch the title for. For example:

```
arduino
```

Copy code

```
python get_page_title.py https://www.example.com
```

The script will then output the title of the web page, or an error message if there's an issue with the URL or fetching the content.

「伝えること」  
と  
「翻訳すること」

# 別の視点：何のための「翻訳」？

やりたいのは「**翻訳**」？それとも「**伝達**」？

もしやりたい事が「伝達」ならChatGPTに「翻訳」を頼むのではなく「伝えたい事」を教えて作文してもらおうという戦略も。

→ **Generative** の特性を最大に活かす方向性へ

以下の内容を伝える文章を英語で作って下さい。  
出力形式はマークダウン形式とします。  
多くの応募を促す文言も入れてください。

-----

青空キャンプ場予約開始

受付は4月21日10時から

応募資格は18歳以上なら誰でも

ウェブページ ([aozora-camp.com](http://aozora-camp.com)) のみで受付

第三希望まで入力して、応募書多数の場合は抽選

お問い合わせはメールアドレス [xxx@yyy](mailto:xxx@yyy) まで

もちろん最終チェックは必要  
だが、日本語の原稿を「翻  
訳」するよりは自然な文章が  
生成される可能性が大きい？

## Aozora Campground Reservations Now Open!

Get ready for an unforgettable experience! Aozora Campground is excited to announce that **reservations will open on April 21st at 10 AM**. Don't miss this fantastic opportunity – we encourage everyone to apply and secure their spot!

**Eligibility:** If you're 18 years old or older, you're welcome to join us!

**How to apply:** Reservations will only be accepted through our website: [aozora-camp.com](http://aozora-camp.com)

**Preference:** You can enter up to three preferences. In case of a large number of applications, spots will be allocated via a lottery system.

**Contact us:** If you have any questions or concerns, please feel free to reach out to us at our email address: [xxx@yyy](mailto:xxx@yyy)

We can't wait to welcome you to Aozora Campground! ✨

もちろん日本語でも ...

以下の内容を伝える文章を英語で作って下さい。  
出力形式はマークダウン形式とします。  
多くの応募を促す文言も入れてください。

-----

青空キャンプ場予約開始  
受付は4月21日10時から  
応募資格は18歳以上なら誰でも  
ウェブページ ([aozora-camp.com](http://aozora-camp.com)) のみで受付  
第三希望まで入力して、応募書多数の場合は  
お問い合わせはメールアドレス [xxx@yyy](mailto:xxx@yyy) まで

## 青空キャンプ場予約開始！🏕️大募集中！

みなさん、待ちに待った青空キャンプ場の予約受付が4月21日10時から開始されます！🎉  
あなたもこの機会に、忘れられないキャンプ体験をしませんか？多くの方々のご参加をお待ちしています！

### 応募資格

18歳以上であれば、どなたでもご応募いただけます。

### 予約方法

ウェブページ [aozora-camp.com](http://aozora-camp.com) にアクセスし、専用フォームからお申し込みください。受付はウェブページのみとなります。

### 第三希望まで入力

予約時には、第三希望までの希望日を入力してください。もし応募が多数の場合、抽選によって予約が決定されます。

### お問い合わせ

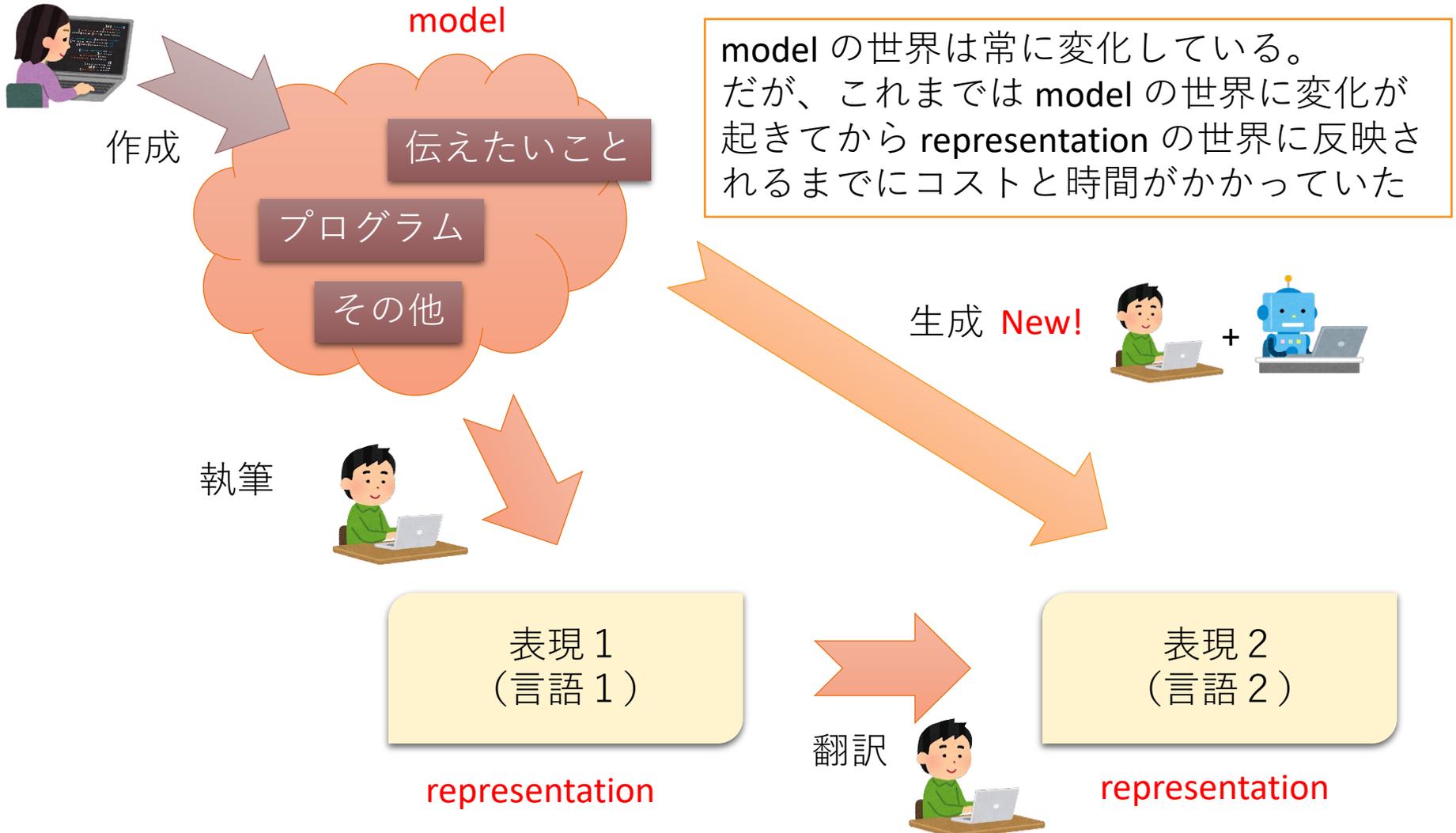
お問い合わせは、メールアドレス [xxx@yyy](mailto:xxx@yyy) までお願いいたします。

ぜひお早めにお申し込みください！大自然の中で素晴らしいキャンプ体験が待っています！🌲🔥

# 翻訳 vs 生成

- 「翻訳」は何も英語から日本語への変換だけを指しているわけではない
- ある記述を別の記述へ、目的に応じて書き換える行為も「翻訳」の一種と考えると良い
- たとえば
  - 要約もパラフレーズも翻訳の一種
  - 箇条書きを文章にするのも翻訳の一種
  - 常識で行間を埋めるのも翻訳の一種 ...
  - Etc ...
- これらの作業に **Generative AI** は大きく関わってくる

# Model から Representation へ



# より上流工程への応用

プログラミングにたどり着く前の段階での利用

# 業務手順の策定 (1)

これは全部人間の入力

思いついたことを五月雨式に書いている

あなたは業務定義を考える専門家です。以下の記述を読んで**販売員のための業務手順書**を作成して下さい。

- こがね弁当は持ち帰り弁当屋です
- お客さんは店頭で販売員に口頭で注文します
- 販売員はお客様の注文をタブレットに入力します
- タブレットに入力する時点でリアルタイムに合計額が計算されます
- 店頭に窓口があるだけで客席はありません
- 注文はキッチンに伝達され調理員が作ります
- 弁当ができたら販売員がお客様に弁当を手渡し代金を受け取ります
- 代金の決済は、現金、クレジットカード、ICカード、バーコード決済が利用できます
- お客様がクーポンを持参していたら会計前にクーポンを受け取って情報を入力し割引を行います

# 業務手順の策定 (2)

これは全部 ChatGPT の出力

販売員の業務を「常識」から推論している

## こがね弁当 販売員業務手順書

- 1.お客様の出迎え** 店頭にお客様が来られたら、丁寧に挨拶をして注文を受け付けます。
- 2.注文の受付** お客様から口頭で注文を受け、タブレットに注文内容を入力します。
- 3.合計額の確認** タブレットに入力した時点でリアルタイムに合計額が計算されるので、お客様に合計金額をお伝えします。
- 4.クーポンの確認** お客様がクーポンを持参している場合は、会計前にクーポンを受け取り、タブレットに情報を入力して割引を適用します。
- 5.注文内容のキッチンへの伝達** タブレットを使って、注文内容をキッチンに伝達します。調理員が弁当の調理を開始します。
- 6.弁当の受け取りとお客様への手渡し** 弁当が調理されたら、キッチンから弁当を受け取り、お客様に手渡します。
- 7.代金の受取** お客様から代金を受け取ります。現金、クレジットカード、ICカード、バーコード決済が利用できます。お客様の希望に応じた決済方法を選択してください。
- 8.お客様への感謝の言葉** 代金の受取が完了したら、お客様に感謝の言葉を述べ、丁寧にお見送りをします。
- 9.次のお客様の対応** お見送りが終わったら、次のお客様の対応に移ります。

以上がこがね弁当の販売員業務の手順です。注意深く対応し、お客様に満足いただけるサービスを心がけてください。

# 業務手順の策定 (3)

ここで想定された業務に利用するアプリのプロトタイプ作成を依頼

【人間】販売員が使うタブレット端末上のウェブアプリのプロトタイプを作成します。クライアント側はHTML、埋め込みCSS、埋め込みJavaScript、サーバー側はPythonで生成してください。

- Python の Flask を使ったプロトタイプを生成してきた。
- 上のプロンプトのままでは動作しなかったが、追加の指示を2回行った結果動作するプロトタイプが生成された。

数量を変えると合計金額が計算される

注文を確定すると、サーバー側に注文内容が python の Dict 形式で出力される

```
{'items': [{'name': '弁当A', 'price': 500, 'quantity': 1}, {'name': '弁当B', 'price': 600, 'quantity': 1}, {'name': '弁当C', 'price': 700, 'quantity': 2}], 'coupon': None, 'paymentMethod': 'cash'}
```

## こがね弁当 - 注文管理

商品名 価格 数量

弁当A 500

弁当B 600

弁当C 700

クーポンコード:

合計金額: 2500 円

## 決済方法の選択

現金

# 業務手順の改善

【人間】 この業務手順を改善するアイデアをいくつか出して下さい

手順とは直接関係ない話も出てきている

- 1. オンライン注文システムの導入** お客様が事前にオンラインで注文できるシステムを導入し、来店時には待ち時間を短縮させることができます。
- 2. デジタルメニューの導入** デジタルメニューを導入し、お客様が注文する際に直接タブレットに注文を入力できるようにすることで、注文ミスを減らし、効率を向上させます。
- 3. セルフレジの設置** セルフレジを設置し、お客様自身が決済を行えるようにすることで、販売員の負担を軽減し、効率を向上させます。
- 4. 弁当の受け渡し方法の改善** 弁当の受け渡し時に、専用のトレイや袋を使用することで、お客様が持ち帰りやすくなります。
- 5. クーポンのデジタル化** デジタルクーポンを導入し、お客様がスマートフォンやタブレットでクーポンを提示できるようにすることで、紙のクーポンの管理が不要になり、環境にも優しいです。
- 6. メニューの変更やキャンペーンの告知** 店頭にメニューの変更やキャンペーン情報を掲示し、お客様に情報提供を行い、リピーターを増やす努力をしましょう。
- 7. 顧客満足度調査の実施** お客様からフィードバックを受け取ることができる顧客満足度調査を定期的の実施し、改善点を見つけてサービスの質を向上させましょう。
- 8. 販売員の研修・教育** 販売員が円滑なコミュニケーションを行い、お客様に適切なサービスを提供できるように、定期的な研修や教育を行いましょう。

# プロンプト エンジニアリング

# プロンプトを書く際の 2 原則

- 原則 1

- 明瞭で具体的な指示を書くこと
  - 何が欲しいのかをはっきりさせる

- 原則 2

- モデルに「考える」余裕を与えること
  - 複数のことをいっぺんに考えさせると混乱する

# (現在) 有名な手法

- **ゼロショット (Zero-shot)**
- **フューショット (Few-shot)**
- **チェインオブソート (Chain of thought)**
- **プロンプトプライミング (Prompt priming)**
- **インストラクション (Instruction)**
- **アンカリング (Anchoring)**
- **ネガティブプロンプト (Negative Prompt)**
- **... etc ...**

細かいテクニックはいろいろあるが一所懸命覚えてもあまり意味はない

# プロンプトエンジニアリング (0)

## プロンプト構成要素

- 文脈（ルール）の設定
- 行ってほしい内容（目的、制約、手段）
- 出力形式
- 詳細化への促し

# プロンプトエンジニアリング（1）

あなたは要件定義アシスタントです。相手の書く文章を解析して、アクター、アクターが最終的に行うこと、そのために満たされているべき前提条件に分解し、それぞれを箇条書きにしてください。もし不明な点ある場合は何でも質問して下さい。

文章「唐津市は景気浮揚策として商品券を販売しようとしています。販売対象は市民です。販売される商品券は額面5000円、販売価格4000円で、プレミアが1000円分乗せられています。商品券の販売数には限りがあり上限10000枚が予定されています。人気が予想されるため、事前申請と抽選を行い当選者に商品券を販売することにします。事前申請で申し込めるのは最大4枚までです。この業務を行うために必要な手続きを挙げて下さい」

## プロンプトの工夫の基本

- (1) AIにロールを割り当てる
- (2) 行って欲しいこと
- (3) 出力形式
- (4) 詳細化への促し

# プロンプトエンジニアリング（2）

要求に従った出力が得られている

分解：

**アクター：**

- 唐津市
- 市民

**アクターが最終的に行うこと：**

- 商品券の販売（唐津市）
- 商品券の購入（市民）
- 満たされているべき前提条件：
- 販売される商品券は額面5000円、販売価格4000円
- 商品券のプレミアは1000円分
- 商品券の販売上限は10000枚
- 事前申請と抽選を行う
- 当選者に商品券を販売する
- 事前申請で申し込めるのは最大4枚まで

**この業務を行うために必要な手続き：**

- 商品券の販売計画を立案（唐津市）
- 事前申請方法と抽選方法を決定（唐津市）
- 事前申請期間を設定（唐津市）
- 市民に事前申請方法と抽選方法を周知（唐津市）
- 市民が事前申請を行う（市民）
- 抽選を実施（唐津市）
- 当選者に連絡（唐津市）
- 商品券の販売と配布（唐津市）
- 市民が商品券を利用（市民）
- 不明な点：
- 事前申請の方法は何ですか？
- 抽選方法にはどのような基準がありますか？

**不明な点：**

- 事前申請の方法は何ですか？
- 抽選方法にはどのような基準がありますか？

# プロンプトエンジニアリング（3）

【人間】事前申請はウェブを使った申込みとします。申込時に市民は自分のアカウントを作成します。抽選方法は申請からランダムに申込商品券数が10000枚になるまで選び続けます。

## • アクター:

1. 唐津市
2. 市民

## • アクターが最終的に行うこと:

1. 商品券の販売（唐津市）
2. 商品券の購入（市民）

## • 満たされているべき前提条件:

1. 販売される商品券は額面5000円、販売価格4000円
2. 商品券のプレミアは1000円分
3. 商品券の販売上限は10000枚
4. 事前申請と抽選を行う
5. 当選者に商品券を販売する
6. 事前申請で申し込めるのは最大4枚まで
7. 事前申請はウェブを使った申込み
8. 申込時に市民は自分のアカウントを作成
9. 抽選方法は申請からランダムに申込商品券数が10000枚になるまで選び続ける

## • この業務を行うために必要な手続き:

1. 商品券の販売計画を立案（唐津市）
2. 事前申請方法と抽選方法を決定（唐津市）
3. 事前申請期間を設定（唐津市）
4. ウェブ申請システムを開発・準備（唐津市）
5. 市民に事前申請方法と抽選方法を周知（唐津市）
6. 市民がウェブ上でアカウントを作成（市民）
7. 市民がウェブ上で事前申請を行う（市民）
8. 抽選を実施（唐津市）
9. 当選者に連絡（唐津市）
10. 商品券の販売と配布（唐津市）
11. 市民が商品券を利用（市民）

質問に答えた結果

# プロンプトエンジニアリング

<https://www.promptingguide.ai/jp> (英語版の翻訳)

細かいテクニックはこの辺を読んで試してみるのも良い



The screenshot shows the 'Prompt Engineering Guide' website. The main heading is 'Prompt Engineering Guide'. The page is divided into sections: 'Introduction', 'LLM設定', '基本的なプロンプティング', 'プロンプトの要素', 'プロンプトをデザインする一般的なTips', 'プロンプトの例', and 'Techniques'. The 'Techniques' section includes 'Zero-shotプロンプティング', 'Few-shotプロンプティング', 'Chain-of-Thoughtプロンプティング', '自己整合性 (Self-Consistency)', '知識生成プロンプティング', '自動プロンプトエンジニア', 'アクティブプロンプト', and '方向性刺激プロンプティング'. The main content area contains the title 'Prompt Engineering Guide' and introductory text about LLMs and prompt engineering.

まだ始まったばかり

競争相手が少ない

今始めると第一人者？



しかし細かいテクニックそのものを一所懸命覚えても意味がない。原理原則から組み立てることが大切

# 考察とまとめ

# Generative AI のメリットとデメリット

- メリット

- 豊富なバリエーションの中から検討できる
- クリエイティブな提案ができる
- 上手く使えばコストの削減につながる

- デメリット

- 倫理的な問題や法的な問題が生じる可能性がある
- 生成されたデータの品質や信頼性を保証できない場合がある
- 人間の感性や価値観を置き換えることはできない

これは実は AI 生成。一般的な問いかけには一般的な答えしか返ってこない

# 効果的プロンプト作成のために（1）

- 漠然とした質問に対しては漠然とした答えしか返ってこない
- どのような前提条件の下でどのような結果が（どのような形式で）欲しいのかをある程度方向付けてやると役に立つ出力が生まれやすい
- 対話的 AI に対して役割（ロール、文脈）を与えることも重要
- 一度生成した出力に対して制約条件を付け加えて改善案を問いかけるという使い方も有用

## 効果的プロンプト作成のために（２）

- きちんとした成果物があるならそれを前提に、欲しい解析結果を指定して生成を依頼することが可能
- ソースコードや形式的な記述は対象が十分に整っているので、そこに対する問い合わせや変換は実りが多い（model から representation へ）
- レビューなどの目的には有用なことが多い（別視点を提供）

## 効果的プロンプト作成のために（3）

- プロンプトライブラリの蓄積
- 自分の仕事に必要な前提条件、制約条件、出力形式を蓄積しておくと便利
- 画像生成系のAIのためのプロンプトなどもストックできる
- 要求定義、仕様策定、設計、テストケース生成、テストデータ生成などにふさわしいプロンプトを工夫して、ライブラリ化しておけば様々な局面で作業が楽にできるかもしれない

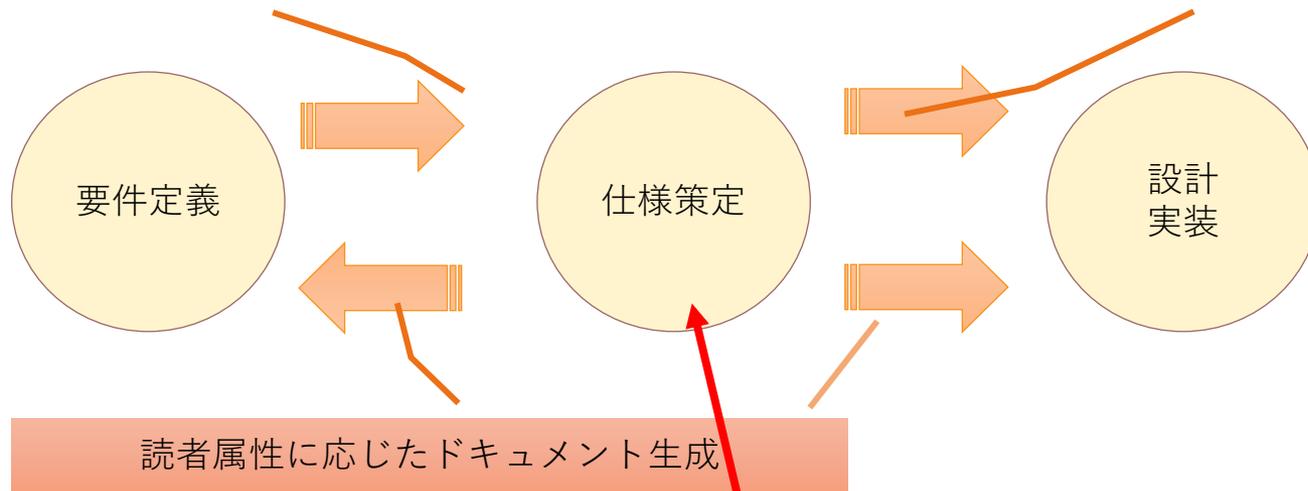
## 効果的プロンプト作成のために（４）

- レビューや記述のテストは必須
- たとえば形式仕様記述を中核にして、生成した記述は仕様レベルでテストできるようにすれば大規模な適用も可能か？
- 設計・実装の生成に向けても、テストケースをうまく生成できるようにすれば信頼性が担保できる？
- 事前、事後、不変条件式などの解析をうまく使えば強力

# 大きなソフトウェア開発 ローと Generative AI

プロンプトエンジニアリングを使った要件抽出  
インタビュー案生成、疑問点抽出 etc ...

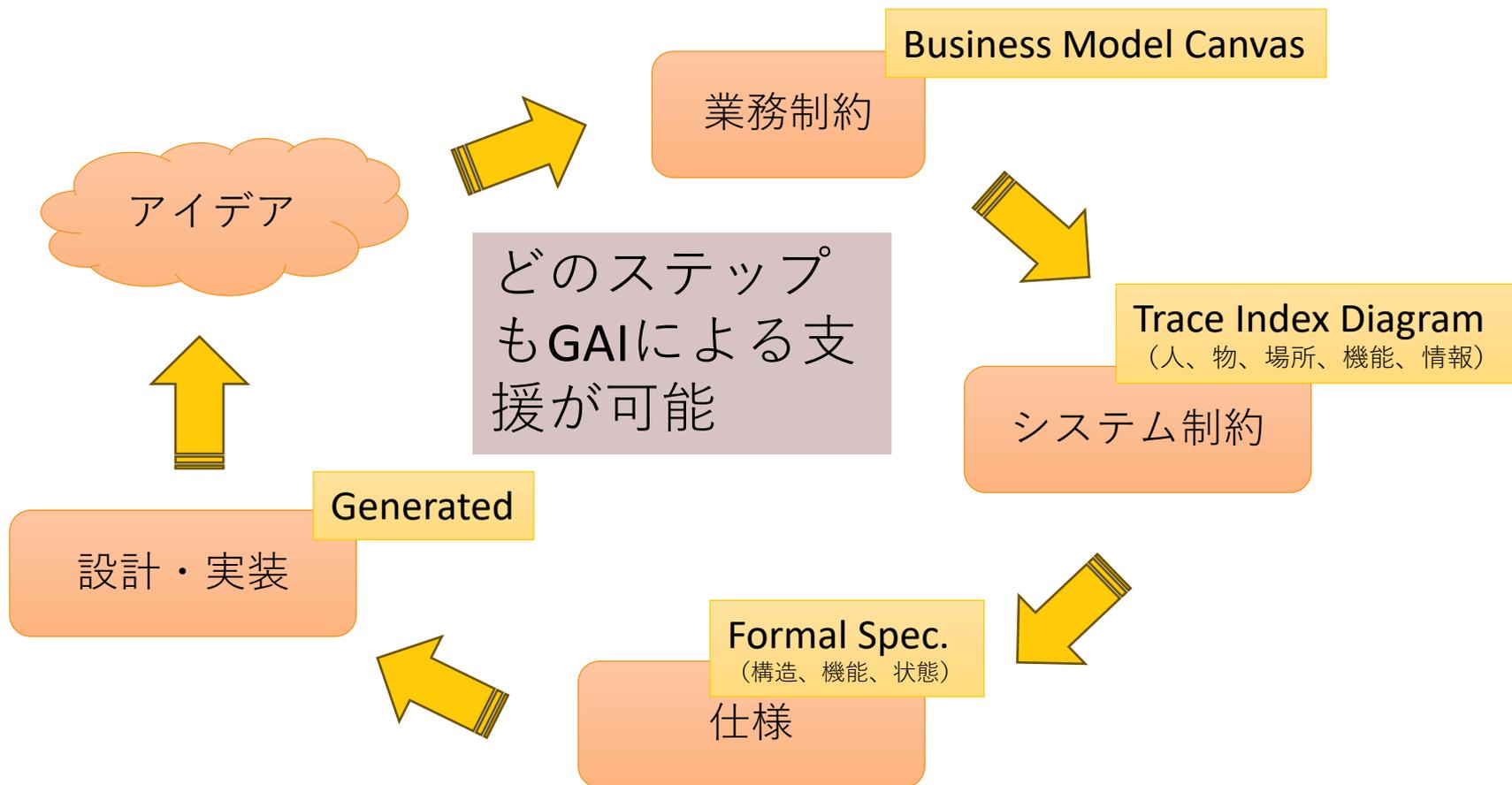
インタラクティブな設計・実装生成  
テストケース、テストデータ生成 etc ...



形式記述を核に知識を効果的に蓄積できるか？

個人的テーマ

# 典型的開発フローとGAI（予告編）



# おまけ

- **Generative AI は強力で危険**
  - 使い方を間違えるとゴミのような出力を大量に生成する
  - AI の役割、求めている出力（基準）、求めている形式などをうまく整理してガイドしてやると有用な出力を得られる → プロンプトエンジニアリングの重要性、有用なプロンプトのライブラリ化と共有
- ちょっとした作業には極めて便利
  - 簡単な環境設定、ちょっとしたシェルスクリプト、簡単な正規表現、ちょっとしたユーティリティ、Excel のマクロ、基準にあったデータ生成 etc ... 決して難しくはないが、それでも調べながら自分でやると 30分とか1時間とかの時間を使ってしまうものは、あっという間に終わるようになる。
- ノーコード、ローコードにも影響
  - クローズドなノーコード、ローコードを使うくらいなら、オープンなライブラリを組合せてアプリケーションを生成できる **Generative AI** の方が良いかも？（生成したコードの解析や改善もできる）
- **What と 実践を優先**
  - 細かいプロンプトテクニックを覚えようとするよりも（細かいテクニックは無効になりがち）、実際に使って、できることできないことを自分で体感して行くことがおすすめ。あくまでも「問題」に注目したい