

ソフトウェア設計への 生成AI活用に向けた適用実験と パターンランゲージの開発

伊藤 弘毅

Mail: Ito.Hiroki@dr.MitsubishiElectric.co.jp

2024/9/13

三菱電機株式会社

- 伊藤 弘毅(いとう ひろき)

- 三菱電機株式会社
設計システム技術センター
ソフトウェア技術推進部



- 担当業務

- モデルベース開発の適用支援

- SQiPシンポジウム2022「パターンランゲージによるモデルベース開発
初学者に対する知識共有の試み」

- **IoT・AIシステムの設計支援**

- SQiPシンポジウム2023「ワークフローモデルの構築によるAI推論フ
ローの処理割当て手法の提案」
- SQiP研究会2023 研究コース5「人工知能とソフトウェア品質」に参加
→研究会の活動成果+αの内容を紹介します

三菱電機グループは、重電システム、産業メカトロニクス、情報通信システム、電子デバイス、家庭電器などの製造・販売・サービスを事業目的としています。



社会システム



電力システム



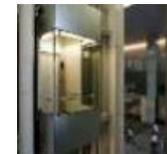
防衛・宇宙システム



FAシステム



自動車機器



ビルシステム



空調・家電



情報システム・サービス



電子デバイス

ミッション

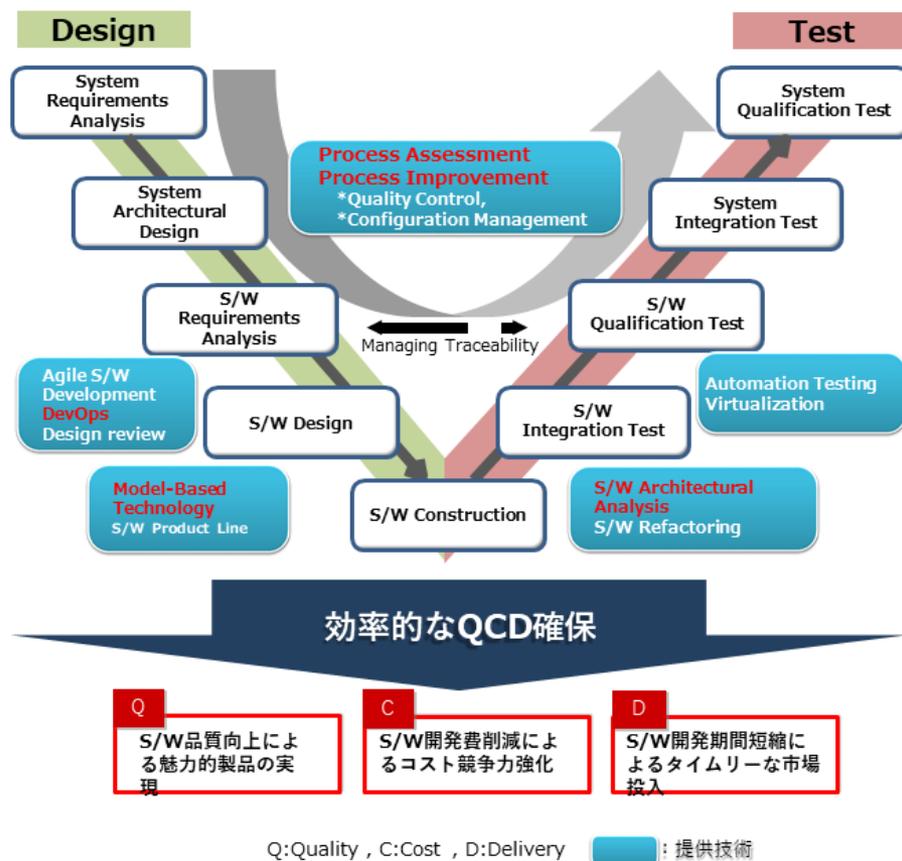
- ソフトウェア設計技術と開発プロセスの革新

ソフトウェア設計技術の開発と実用化とソフトウェア開発プロセスの改善とソフトウェア製品開発への適用支援を実施します。

活動実績

- 開発プロセス全体にわたる改善に貢献

ソフトウェア要求分析手法の構築、ソースコードの静的解析・構造分析ツールを活用した構造改善手法の構築及びシミュレーター開発による試験環境の整備など、ソフトウェア開発プロセス全体にわたる改善活動を推進しています。また、国際標準に準拠したプロセス改善、ガイドライン制定、設計上流でのデザインレビュー技法開発、セキュアな開発プロセス導入も推進し、製品に組み込まれるソフトウェアの品質と生産性の向上に貢献しています。



- 生成AIをソフトウェア実装に活用する事例・サービスが増えている
 - 事例：コード生成の試行、不具合の自動修正
 - サービス：Github Copilot、Amazon Q Developer
- ソフトウェア設計段階の生成AI活用の研究はソースコードと比較すると少ない
 - いきなりコードを書き始める・・・？
 - 設計段階から品質を作りこむと効果的

生成AIをどのように利用すれば、
設計段階の品質の作りこみに役立てられるか？

生成AIにどのようなプロンプトを入力すれば
設計品質の向上につながるのだろうか？

何か共通的に使えるノウハウ(=パターン)はあるだろうか？
もしあるのであれば、それを言語化すれば、
知識を周りの人と共有することができる！

設計品質の向上に役立つ
プロンプトパターンランゲージを作成した

- そもそも、生成AIはソフトウェア設計品質の確保に役立てられるのか
 - ソフトウェア品質を確保する方法：ソフトウェアパターン
 - 設計品質：デザインパターン、アーキテクチャパターン

生成AIでソフトウェアの設計書に アーキテクチャパターンの解決策を反映可能か実験

- 実験内容
 - アーキテクチャパターンの解決策を設計書の記述に対して反映可能か確認する
 - 【アーキテクチャパターン】
Repositoryパターン、Service Stubパターン、Cache Asideパターン
 - 【設計書】 航空便検索システムのモジュールの機能説明を記述したサンプルの文章(800字～1000字)

実験1：生成AIは設計書の記述に対し ソフトウェアパターンの適用ができるか

- 実験
 - 設計書に記述されたモジュールに、特定のアーキテクチャパターンの解決策を反映するようプロンプトで指示(ChatGPT GPT4)
- 結果
 - パターンの解決策を反映したモジュール構造の記述を出力できた

- データベースアクセスモジュール:

- リクエスト処理モジュールからのクエリは、まずキャッシュをチェックします。キャッシュにデータが存在する場合は、キャッシュからデータを取得して返します。
- キャッシュにデータが存在しない場合、データベースからデータを取得し、結果をキャッシュに保存します。その後、取得したデータをリクエスト処理モジュールに返します。

Cache Asideパターン

- 考察
 - 実際の設計書+他の複数のソフトウェアパターンで可能か検証要

実験2：特定の言語モデルによらずに ソフトウェアパターンを設計書に適用できるか

2024年2月の結果

- 実験
 - ChatGPT GPT3.5とGoogle Bard(PaLM 2)、Claude3(Opus, Sonnet)でアーキテクチャパターンの解決策が反映可能か確認
- 結果
 - いずれの言語モデルもアーキテクチャパターンを認識
- 考察
 - ただし、Claude3 Opus以外はGPT4と比べると設計書の反映の精度は劣る傾向

Repositoryパターン(ChatGPT GPT3.5)

- データ整形モジュール:

- 本モジュールは、データベースから取得したデータを処理し、ユーザーが理解しやすい形式に整形します。

- データベースアクセスをRepositoryパターンを使用して更新します。

具体的な更新内容は出力されず

●パターンランゲージ

- 元々は建築分野において町や建物を設計する上での知識を共有するために考案(*)
- 後にソフトウェア開発の分野に適用される(**)

ある「文脈」で繰り返し現れる
「問題」と「解決策」をパターンにし、
それを知識として共有するもの

今回は

ソフトウェアの設計品質を向上させる施策を
出力させる効果的なプロンプトの記述方法を
パターンランゲージとして整理

(*) A Pattern Language: Towns, Buildings, Construction, Oxford University Press, 1977

(**) プログラムデザインのためのパターン言語, ソフトバンクパブリッシング, 2001

●パターン一覧

① 専門用語

施策検討

② スコープの限定

③ 候補の例示

④ ズームアウト

評価

⑤ 観点の設定

⑥ メリットとデメリット

⑦ 別解

ソフトウェアの設計ドキュメントを作成している。

その状況において

設計の品質が向上するように漠然とドキュメントの修正を指示しても、一向に期待通りの出力が得られない。

そこで

ソフトウェア開発の専門用語や設計品質の向上に役立つ施策を知っているようであれば、その単語をプロンプトに入力して回答を誘導させよう。

- ○○の保守性を向上させるアーキテクチャパターンの例を挙げてください
- ○○にRepositoryパターンを適用してください



その結果

具体的な専門用語をプロンプトで指定することにより、意図通りの出力が得られるようになる。

ソフトウェアの設計ドキュメントを作成している。

その状況において

プロンプトの内容が抽象的過ぎて出力される結果も抽象的となり、一向に設計品質の向上に繋がる修正が達成できない。

そこで

意図したとおりの出力を得るために、漠然とした指示ではなく、スコープを絞った質問をしよう。

- データベースアクセスモジュールにCache Asideパターンを適用してください
- データベースの時間効率性を向上する施策の例を挙げてください



その結果

意図したとおりに設計ドキュメントを修正することができ、効率よく品質を向上させる施策を反映させることができる。

ソフトウェアの設計ドキュメントを作成している。

その状況において

具体的な施策をプロンプトで指示しないと、誤った修正をドキュメントに施してしまう可能性がある。

そこで

初めに、品質を向上させる手段の候補をいくつか出力させ、その中から適切な手段を成果物に反映させよう。

- ○○の保守性を向上させるアーキテクチャパターンの例を挙げてください
- データベースの時間効率性を向上する施策の例を挙げてください



その結果

提案された候補の中から有用な施策を選択することで、品質を向上させる設計をドキュメントに反映することができる。

スコープの限定したプロンプトを入力することで、設計品質を高めたソフトウェアの設計ドキュメントを作成した。

その状況において

作成した設計ドキュメントについて、本来必要な検討が漏れてしまっており、最終的に品質の悪い製品を開発してしまう可能性がある。

そこで

さらなる品質向上のために、観点や対策の抜け漏れがないか、あえて抽象的にプロンプトで質問して確認しよう。

- 保守性を向上させる方法の例を挙げてください
- 〇〇パターンの他に、〇〇の保守性を向上させる方法の例を挙げてください

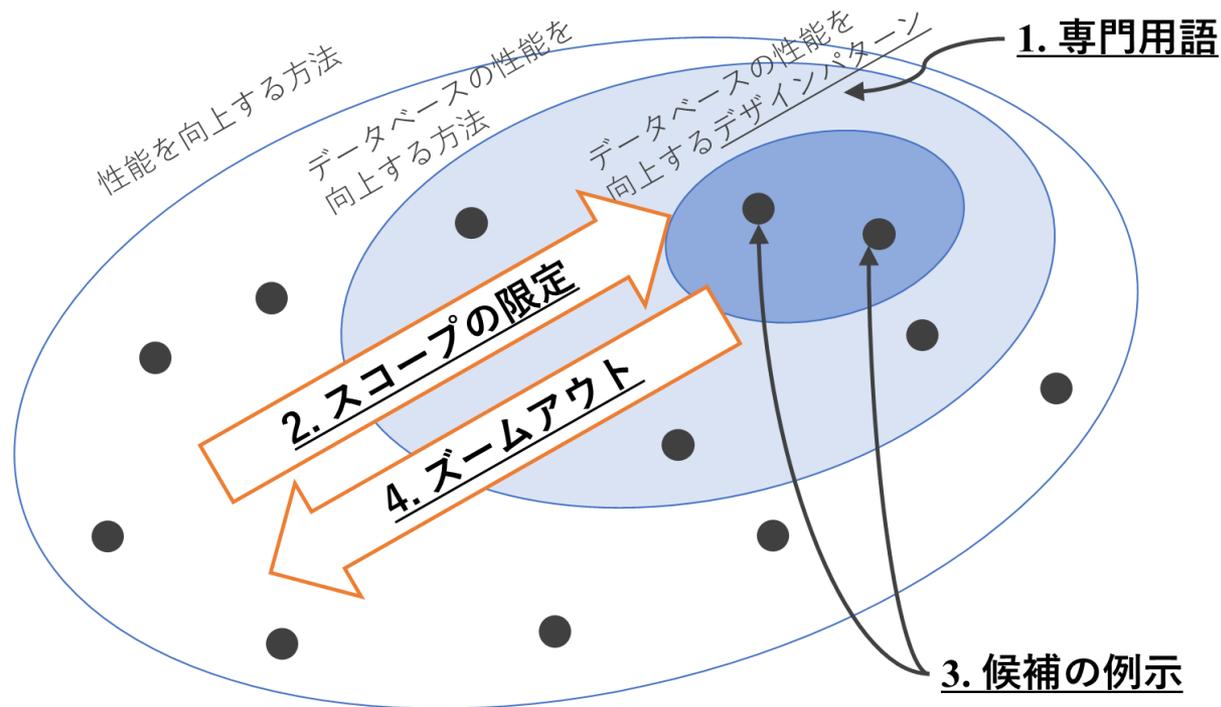
その結果



考慮が漏れていた設計観点を知ることができ、さらなる設計品質の向上が期待できる。

● 全体の流れ

1. 専門用語(ex.デザインパターン)を使用して**スコープの限定**をする
2. 候補の例示を指示して、選択肢を出力させる
3. **ズームアウト**で検討から漏れた施策がないか確認する



スコープの限定



データベースに対するアクセスにアーキテクチャパターンのRepositoryパターンを適用し、記述を更新してもらえますか？

専門用語

--

リクエスト処理モジュール:本モジュールは、フロントエンドから送信されたフライト検索のリクエストを処理します。(略)



データアクセスリポジトリモジュール(新規追加):

この新しいモジュールは、フライト情報の取得や検索条件に基づくクエリの実行など、データベースアクセスのロジックを担当します。リポジトリは、データベースからフライト情報を取得し、それをビジネスオブジェクトにマッピングする責任を持ちます。この抽象化により、アプリケーションの他の部分はデータアクセスの実装詳細から隔離され、データソースの変更やテストの容易性が向上します。

スコープの限定



データベースに対するアクセスの保守性を向上させるアーキテクチャパターンの候補を挙げてもらえますか？

専門用語

候補の例示

--

(以下、モジュール構成)



この航空便予約システムのフライト検索機能のモジュール構成において、データベースに対するアクセスの保守性を向上させるためのアーキテクチャパターンを提案します。

1. リポジトリパターン: リポジトリパターンでは、データアクセスロジックとビジネスロジックを分離します。リポジトリ層がデータベースとの直接のやり取りを担当し、アプリケーション層はリポジトリ層を通してデータにアクセスします。これにより、データベースの変更がアプリケーションコードに影響を与えることなく行えるようになります。
2. データアクセスオブジェクト (DAO) パターン:
3. サービス層パターン:
4. CQRS (Command Query Responsibility Segregation) パターン:



以下は、航空便予約システムのシステム仕様書における、フライト検索機能のモジュール構成の記述です。このモジュール構成に対して、性能効率性を向上させる方法の例を挙げてもらえますか？

ズームアウト

--

候補の例示

(以下、モジュール構成)



航空便予約システムのフライト検索機能のモジュール構成において、性能効率性を向上させるための方法をいくつか提案します。

リクエスト処理モジュール

- リクエストキャッシング: よくある検索リクエストの結果をサーバー側でキャッシュし、同一リクエストに対して迅速に応答する。

データベースアクセスモジュール

- 効率的なクエリの最適化: SQLクエリの最適化を行い、データベースの応答時間を短縮する。
- インデックスの利用: データベースに適切なインデックスを設定し、検索速度を高める。

生成AIにプロンプトを入力し、設計品質を高める解決策を知ることができた。

その状況において

生成AIが出力した方法が、本当に効果的なのか確かめたいと思っている。回答結果を見ても、なにを根拠に回答の結論に至ったのか、よく分からないことがある。

そこで

生成AIに回答のよりどころとなる観点をプロンプトに与えよう。参照する観点は、人が与えても構わないし、生成AIに候補の例示をさせても構わない。書籍などで先人の知恵を知っているようであれば、その観点を設定すると信頼性が増すだろう。

- #観点で示した項目について5点満点で評価をしてください

その結果



生成AIが出力した方法が、特定の観点において適切かどうか評価することができる。

生成AIにプロンプトを入力し、設計品質を高める解決策を知ることができた。

その状況において

提案された解決策の負の側面が分からず、本当に適用してもよいか分からない。

そこで

出力された候補について、適用した際のメリットとデメリットを出力させよう。
専門用語で観点の設定をしたプロンプトを入力して、その観点がプラスになるのかマイナスになるのか聞いてみてもよい。

- Repositoryパターン適用によるメリットとデメリットを教えてください
- サービス指向アーキテクチャは、スキルレベルの観点でデメリットはありますか

その結果



提案された解決策を適用することのデメリットも確認することで、意図せぬ品質の低下を防ぐことができる。

生成AIにプロンプトを入力し、設計品質を高める解決策を知ることができた。

その状況において

提案された解決策が本当に一番優れた回答であるか分からない。また、生成AIが出力した方法以外に、全く解決策がないかどうか分からない。

そこで

生成AIに、出力した回答の代替案があるかを聞いてみよう。ある条件下では、別の方法を採用した方が望ましい場合があるかもしれない。

- X以外に適切な方法があれば教えてください
- 条件によって回答が変わる場合は、その条件と修正後の回答を出力してください

その結果



出力された解決策以外に可能性のある代替案を確認することができる。もし、代替案の方が開発の状況に合っている場合は、それを採用することでより良い設計にできる。



マイクロサービスアーキテクチャの適用は、チームのスキルレベルの観点で
デメリットはありますか？

メリットとデメリット

観点の指定



マイクロサービスアーキテクチャの適用におけるチームのスキルレベルの観点でのデメリットは以下のような点が考えられます：

1. マイクロサービスの設計と開発：各サービスを独立して設計・開発する必要があるため、ソフトウェア設計の知識が要求されます。特に、サービスの境界を明確にし、疎結合を保つための設計スキルが必要です。
 2. サービス間通信とデータの一貫性：マイクロサービスは独立して動作するため、サービス間の通信が増えます。RESTful API、gRPC、メッセージングキューなどの通信手法の知識が必要です。
 3. デプロイメントと運用の複雑さ：各マイクロサービスを独立してデプロイするため、CI/CDパイプラインの構築が不可欠です。コンテナ技術（Dockerなど）やコンテナオーケストレーション（Kubernetesなど）の知識が求められます。
- …(略)…

これらのデメリットを克服するためには、チームがこれらの技術やスキルを持っているか、または学習する意欲があるかどうかを評価することが重要です。必要に応じてトレーニングや外部の専門家の支援を受けることも検討すると良いでしょう。



コーヒーチェーンのモバイルオーダーサービスの#状況は次のとおりです。#状況を踏まえ、モバイルオーダーサービスの候補のアーキテクチャのうち、どれが望ましいか評価してください。

--

#状況 ...



適切なアーキテクチャ: マイクロサービスアーキテクチャ
選定理由: ...



マイクロサービスアーキテクチャ以外に適切なアーキテクチャがあれば教えてください。

別解



適切なアーキテクチャ: サービス指向アーキテクチャ(SOA)
選定理由: マイクロサービスアーキテクチャに次いで、サービス指向アーキテクチャ(SOA)も適切です。SOAは各サービスが独立しており、高いモジュール性を提供します。

- まとめ

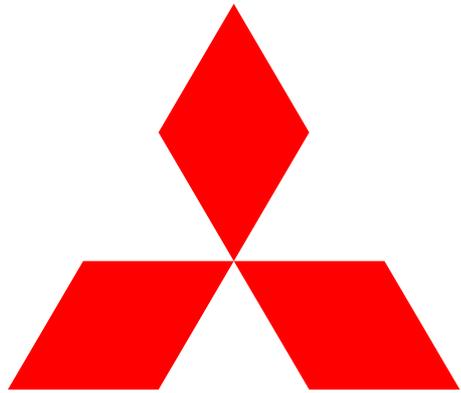
- ソフトウェア設計品質向上のためのプロンプトパターンランゲージを作成
 - 施策検討と評価に関する全7パターン

- 今後の展望

- プロンプトパターンランゲージの拡充
- もし、こんなプロンプトを入力したら「うまくいきそう」「おもしろそう」などアイデアがありましたら、投稿いただけるとうれしいです！

- 謝辞

- 本研究において、活動の機会をご提供くださりました日本科学技術連盟様、貴重なご意見を頂いたSQiP研究会の石川さん、栗田さん、徳本さん、参加者の皆さまに厚く御礼申し上げます。



**MITSUBISHI
ELECTRIC**

Changes for the Better