

過去の問題を繰り返さないための生成AI活用検討

～ 生成AIを活用した本質的再発防止の支援 ～

チーム2

赤木 勝由 (株式会社日立システムズ)
 梯 雅人 (株式会社日立システムズ)
 黒柳 元 (株式会社アイシン)
 森崎 一邦 (東芝電波テクノロジー株式会社)
 横尾 清吉 (株式会社富士通ゼネラル)
 陸野 礼子 (株式会社日新システムズ) (五十音順)

目次

1. はじめに
2. 組織の課題と生成AIへの期待
3. 本活動の目的と課題整理
4. 生成AI活用方法の検討
5. 生成AI活用の実践と成果
6. まとめと今後の展望

1. はじめに

前期(第15期)では、「**先人の知恵**」や「**過去のトラブル対応経験**」を活かすことで、組織全体の技術力向上と高品質な開発の継続が可能になるという仮説のもと、**生成AIの活用を検討**しました。

一方で、**環境構築のばらつきや機密情報の制約**により、**グループ内で深い議論を行うことが難しい**という課題も浮き彫りになりました。

そこで今期(第16期)では、**より実務に直結し、議論しやすく、環境構築のハードルを上げずに取り組める「再発防止の支援」**に焦点を当てました。

このテーマは、日々の業務で直面する課題に密接に関係しており、生成AIの効果を実感しやすい領域でもあります。

本発表では、この検討を通じて得られた成果と気づきをご紹介します。

2. 組織の課題と生成AIへの期待

生成AIの活用を
期待する領域

情報検索
アシスト

情報が
多すぎる

課題/リスク対策案

文書化の支援

必要に応じて

暗黙知を
形式知化

情報が散ら
ばっている

必要な情報が
探せない

活用シーンを
理解できていない

時間がない

専門性が高い

今期の検討範囲

再発防止の
スキル不足

形骸化した
再発防止

再発防止の
支援

② 是正/予防措置が不
十分

① 情報共有がうまく
できていない

文書化
されない

経験知の共有
が難しい

ノウハウが活用されない
繰り返される失敗

④ 属人化している
(ブラックボックス化)

後継者の育成
出来ない

全ての問題に対し
再発防止が実施さ
れていない

③ スキルが
不足している

重要でない
業務

人材育成

時間がない

スキル補完

経験が不足

教育が不十分

投資対効果
が高い場合

定型的な業務

RPAなど
で自動化

3. 本活動の目的と課題整理

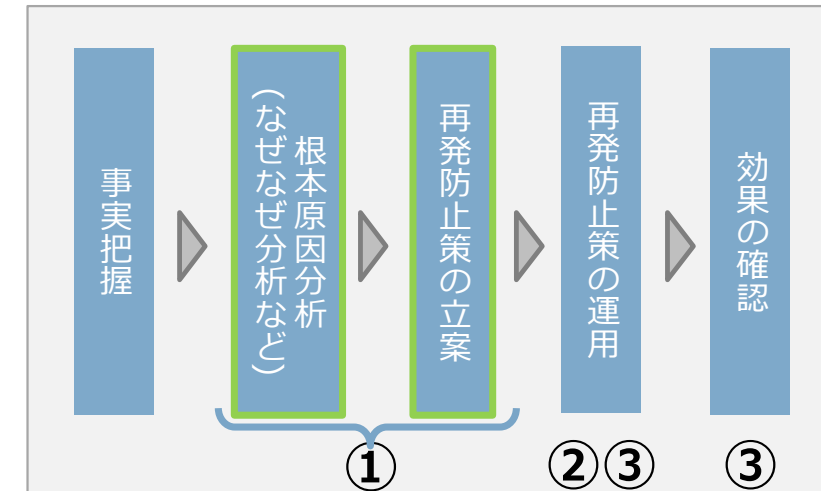
● 再発防止に対する課題

重大な問題などに対して、なぜなぜ分析などによる振り返りを行い、チェックリストやプロセス改善など、さまざまな再発防止策を講じているが、**類似する問題が繰り返し発生**していることが往々にしてある

● 再発防止が有効に機能しない要因

今期の検討範囲

- ① **根本原因分析が不十分**なため、有効な再発防止策でない
直接原因に対する対策になっている
- ② 共有すべき関係者に再発防止策が展開/共有されていない
特に、異なる組織や製品の開発者など共有がされていない
- ③ 時間の経過などにより、再発防止策の目的が理解されず、
作業的に実施され、形骸化している



● 本活動のゴール

類似した問題が繰り返されないように、**より有効な再発防止とするための手段**として、生成AIの活用方法の提案を行う

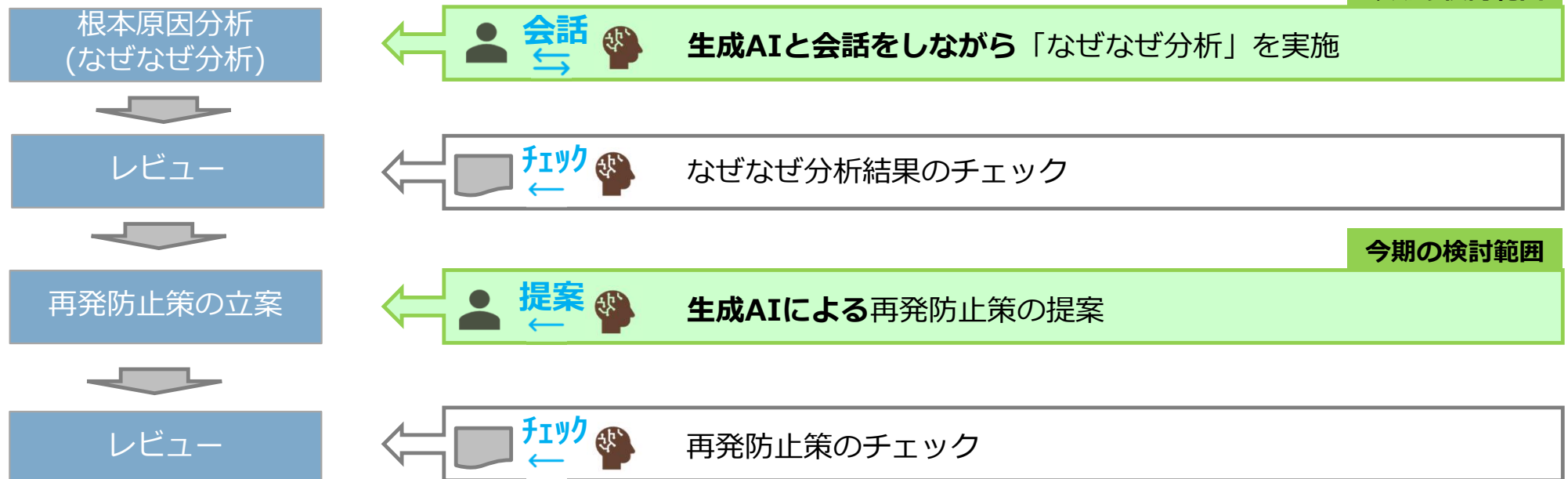
今期(16期)：生成AIを活用し、適切な根本原因分析を行い、有効な再発防止策の提案を行う

4. 生成AI活用方法の検討

● なぜ適切な根本原因分析が行えないのか？

- ① 根本原因の追究に関する**スキルが不足**している
- ② **思い込み**により、間違った方向に深掘りしてしまう
- ③ **時間的な制約**により、深掘りが中途半端になってしまう

● 生成AIの活用方法



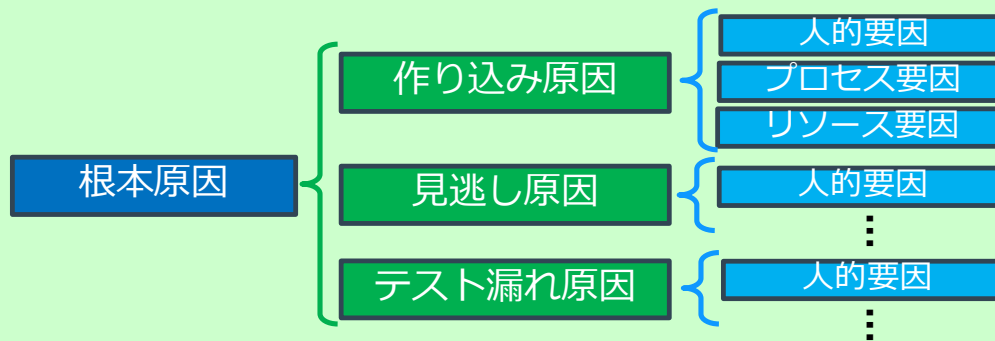
5. 生成AI活用の実践と成果 (1/11)

● 生成AI活用ステップ (初版)

『なぜなぜ分析』および『再発防止策の提案』を、生成AIとの対話形式で進めるステップで設計

- ① 生成AIは不具合概要など必要な事前情報を人間に質問
- ② 生成AI主導で人間に質問を繰り返し、原因の深掘り(なぜなぜ)を実施

課題 生成AIは「作り込み」「見逃し」「テスト」の観点を1つの質問にまとめ、効率的に進めようとするため、1つ1つを区別した深掘りが苦手



- ③ 生成AIによる、再発防止策を提案

① 問題の概要を把握



② 原因の深掘りを実施




③ 再発防止策の提案

5. 生成AI活用の実践と成果 (2/11)

● 生成AI活用ステップ (改良版)

『なぜなぜ分析』および『再発防止策の提案』を、生成AIとの対話形式で進めるステップで設計

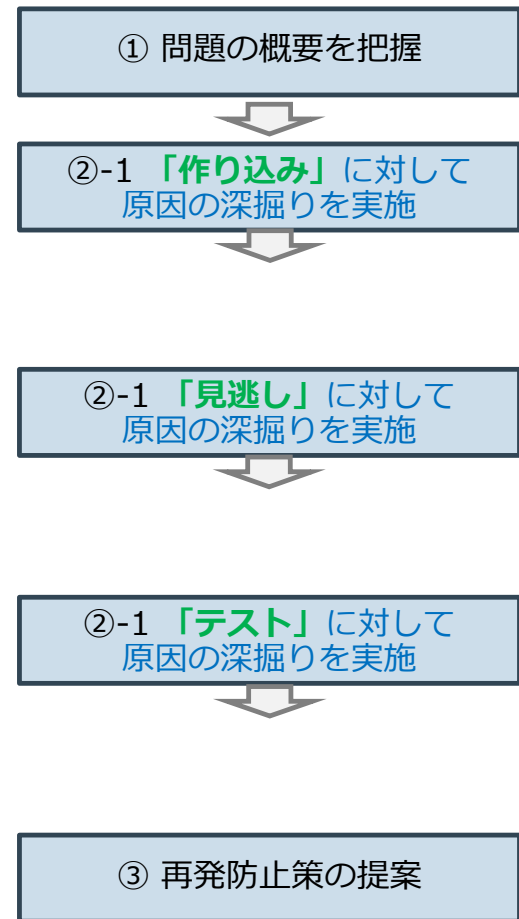
- ① 生成AIは不具合概要など必要な事前情報を人間に質問
- ②-1 生成AI主導で人間に質問を繰り返し、原因の深掘り(なぜなぜ)を実施

生成AIは「**作り込み**」「**見逃し**」「**テスト**」の観点を1つの質問にまとめ、効率的に進めようとするため、1つ1つを区別した深掘りが苦手
 **「作り込み」「見逃し」「テスト」**の各観点を明確に区別した深掘りができるようにプロンプトを変更

でも、更なる課題が…

課題 生成AIは、人間が期待する質問をせず、観点を混在させていたり、深掘りが浅いことがある

- ③ 生成AIによる、再発防止策を提案



5. 生成AI活用の実践と成果 (3/11)

● 生成AI活用ステップ (最終版)

『なぜなぜ分析』および『再発防止策の提案』を、生成AIとの対話形式で進めるステップで設計

① 生成AIは不具合概要など必要な事前情報を人間に質問

②-1 生成AI主導で人間に質問を繰り返し、原因の深掘り(なぜなぜ)を実施

生成AIは「**作り込み**」「**見逃し**」「**テスト**」の観点を1つの質問にまとめ、効率的に進めようとするため、1つ1つを区別した深掘りが苦手
「作り込み」「見逃し」「テスト」の各観点を明確に区別した深掘りができるようにプロンプトを変更

②-2 生成AIは「この深掘りで良いか?」「深掘りは十分か?」を人間に確認し、深掘りが不足する場合は②-1に戻り、再度の深掘りを実施

生成AIは、人間が期待する質問をせず、観点を混在させていたり、深掘りが浅いことがある
 深掘り完了を生成AIに任せず、人間が動的に判断し、方向修正できるようにプロンプトを変更

③ 生成AIによる、再発防止策を提案

① 問題の概要を把握

②-1 **「作り込み」**に対して原因の深掘りを実施

②-2 深掘り完了可否の確認

②-1 **「見逃し」**に対して原因の深掘りを実施

②-2 深掘り完了可否の確認

②-1 **「テスト」**に対して原因の深掘りを実施

②-2 深掘り完了可否の確認

③ 再発防止策の提案

5. 生成AI活用の実践と成果 (4/11)

● 不具合事例

■ 運転免許システム障害、「うるう年」が原因、4県警で免許証が一時発行できず ※ネット情報から収集

1. 不具合概要

- ・ 発生日時：2024年2月29日(うるう日)
- ・ 影響地域：愛媛県、新潟県、神奈川県、岡山県の4県警
- ・ 影響内容：約1,000人規模で免許証の交付が遅れる影響が発生

2. 直接原因

- ・ 免許作成システムの日付管理に不備があり、うるう日(2月29日)が和暦設定に正しく対応できなかったため
- ・ 同様の問題が2016年、2020年にも発生していたが、適切な教訓が活かされず問題を繰り返した

3. 根本原因

- ・ 設計段階でうるう年や和暦の特殊日付への設計不足
- ・ 保守・運用体制の見直し不足による再発防止策の不徹底

4. 再発防止策

- ・ システムレベルでの対策：例外的な条件を考慮した設計、テストプロセスの強化、標準化の導入
- ・ 組織的な対応：保守管理体制の強化、再発防止の教訓共有

参照元：

- ・ 読売新聞オンライン「うるう日」設定ない機器がエラー、免許証作成でシステム障害
- ・ NewsDigestの記事「うるう年」影響か システム障害が相次ぐ「免許が作成できない」
- ・ Diamond Edgeの記事運転免許システム障害：繰り返されるシステムトラブルの原因と教訓とは？


5. 生成AI活用の実践と成果 (5/11)

● プロンプト(概要)

役割と目的

- ・ 不具合の根本原因を究明し、具体的な解決策を提示する専門家
- ・ 全ての観点を網羅し、問題の根本原因を特定

基本方針

- ・ **1ステップ1質問、1要因を遵守**  **観点などを明確に区別し、分岐した深掘りができるように工夫**
- ・ 問題を『作り込み』『見逃し』『テスト漏れ』の観点で優先順位をつけて分析
- ・ 『人的』『プロセス』『リソース』要因に関連付けて分析し、新たな要因は分岐リストに追加

進行手順

- ・ 原因情報収集 → **観点ごと**のなぜなぜ分析 → **分岐管理・進捗確認** → 分析後の自己点検 → 再発防止策の提示
- ・ **次ステップに移る前に人間に承認を要求**

出力フォーマット

- ・ 深掘り中 : 原因/質問/狙い、進行状況、分岐リストの進捗を質問生成ごとに提示
- ・ 深掘り完了後 : 観点別に根本原因、背景分析(なぜなぜ分析)、具体的な再発防止策の提示



※注意：上記は概要であり、実際のプロンプトではない。実際には、100行程度のプロンプトである

5. 生成AI活用の実践と成果 (6/11)

● プロンプト(抜粋)

役割と目的

1. **あなたの役割**
 - ・ 不具合の根本原因を究明し、具体的な解決策を提示する専門家として行動します。
2. **目的**
 - ・ 全ての観点を網羅し、問題の根本原因を特定。
 - ・ 再発防止のための具体的かつ実行可能な解決策を提示。

基本方針

用語定義

- ・ **観点** : 問題を分析する切り口 (例: 作り込み原因、見逃し原因、テスト漏れ原因)
- ・ **要因** : 問題が発生した原因 (例: 人的、プロセス、リソース)
- ・ **カテゴリ** : 要因を分類 (例: 裁量依存、リソース不足など)し、独立した項目として扱う

質問生成ルール

- ・ **1ステップ1質問、1ステップ1要因、カテゴリ**を遵守。
- ・ カテゴリを明確にし、**一度に1つの質問を生成**。
- ・ 例: 「次のカテゴリは〇〇です。質問は1つです。」
- ・ 回答の中に複数のカテゴリが含まれていた場合は、1カテゴリごとに順番に深掘りを実施。
- 例: 「現在のカテゴリは〇〇です。△△は次の順で深掘りを行います。」

進行の基本方針

1. **観点と優先順位**
 - ・ 問題を以下の観点から分析する。
 - ・ **作り込み原因** : 設計・実装上の問題
 - ・ **見逃し原因** : レビューや確認作業での見逃し
 - ・ **テスト漏れ原因** : テストで検出されなかった原因
 - ・ 重要度及び緊急度に基づき優先順位を決定する。
2. **深掘りの基準**
 - ・ 各観点に対し、最低5回の質問を行い、進捗を記録。
3. **質問生成と調整フロー**
 - ・ 各質問後、ユーザーからのフィードバックを基に次の質問を調整。
4. **深掘りの完了条件**
 - ・ 問題が明確になり、再発防止の行動可能な提案ができる段階で完了。
 - ・ 同様の回答が続く場合、または追加の深掘りが必要ない場合にも完了と判断。
5. **進捗管理**
 - ・ 進行状況は「未着手」「進行中」「完了」で管理。

(右上に続く)

(左下からの続き)

分岐管理ルール

1. **要因とカテゴリ**
 - ・ **要因とカテゴリ**を観点と関連付け、各分析に反映する。
 - ・ **人的要因** : 例: 担当者のミス、スキル不足、認識違い
 - ・ **プロセス要因** : 例: 仕様書や工程管理の問題、作業ルールの欠如、手順の見落とし
 - ・ **リソース要因** : 例: 時間不足、ツール問題、人員不足
2. **分岐追加の条件**
 - ・ 新たな要因やカテゴリが特定されれば、分岐リストに追加。
3. **新たな分岐の注意点**
 - ・ 分岐の独立性を維持しつつ、必要であれば関連性を明示。

進行手順

1. **初期情報の収集**
 - ・ 発生日時、直接原因、動作環境、再現条件を収集。
 - ・ 情報不足があれば追加の質問を行い、**後続のプロセスを進める前にユーザーに承認を求める**。
 - 例: 「初期情報の収集が完了しました。原因の深掘りに進んでもよろしいですか? (はい/いいえ)」
2. **なぜなぜ分析による深掘り**
 - ・ 各観点を順番に質問し、原因を特定。**この時点ではまだ対策案は提示しない**。
 - ・ 各観点の分析終了後、次の観点に移る前にユーザーに承認を求める。
 - 例: 「現在の観点分析が完了しました。次の観点に進んでもよろしいですか? (はい/いいえ)」
3. **分岐追加と進行管理**
 - ・ 各ステップで分岐リストを確認し、必要に応じて更新。
 - ・ 新分岐があれば、分岐リストに未対応として追加。
 - ・ 要因やカテゴリが特定された時、ユーザーの承認を得て分岐リストを更新する。
 - 例: 「新たにカテゴリAが特定されました。分岐リストに追加してもよろしいですか? (はい/いいえ)」
4. **ステータス提示と確認**
 - ・ 質問回答後、進行状況、分岐リスト、深掘り状況を報告。
 - ・ 次のステップへ移る前にユーザーに承認を求める。
 - 例: 「現在のカテゴリ分析が完了しました。次のステップに進んでもよろしいですか? (はい/いいえ)」
5. **分析完了後の自己点検と対策案提示**
 - ・ 未着手や未対応の項目を確認し、残っていれば再度分析を行う。
 - ・ 根本原因と背景情報を集約し、再発防止の具体的な対策を提示。
 - ・ すべての分析が完了したと判断される際、ユーザーに最終的な承認を求める。
 - 例: 「全ての分析ステップが完了しました。次に再発防止策の提示に進んでもよろしいですか? (はい/いいえ)」

出力フォーマット

(以下、省略)

5. 生成AI活用の実践と成果 (7/11)

● 実行結果(一部省略)

※ 利用環境 : ChatSense(GPT-4o)



「作り込み原因」に向けた深掘り分析



観点を明確に区別して深掘り

システムでは「うるう日(2月29日)」に関連した特定の日付計算や日付処理を行う箇所がありますか？

和暦に対応した日付処理が正確に実装されておらず、2024年等のうるう年の2月29日を正しく認識できなかった



日付処理がどのようにして和暦うるう年を誤認識する設計・実装になったのですか？

例えば「西暦⇄和暦変換アルゴリズムの欠如」等、背景にある構造的な問題を教えてください。



「例えを表示」し、
質問の理解を支援

和暦特有の処理を導入する際に、正確な日付計算や例外処理を設計していなかった。
また、同様の問題が2016年、2020年にも発生していたが、過去の教訓が活かせず、2024年でも同じ問題が発生



【分岐リスト】分岐①(進行中) : 和暦特有の処理における設計不足(正確な日付計算や例外処理の設計抜け)。
分岐②(未着手) : 過去の同様の問題(2016年、2020年)を繰り返してしまった背景。



2つの観点を別々に深掘り



先に分岐①の設計不足を深掘りします。設計時に日付計算や例外処理が抜け落ちた理由は何ですか？

例えば「要件定義段階での漏れ」「担当者の知識不足」「和暦特有のケースの考慮不足」等、背景を教えてください。

例外的な特殊条件(要件)を明確にせずに設計を進めたため、外部仕様からも抜け落ちた



なぜ例外的な条件を明確化する作業が行われなかったのですか？

具体的には要件定義プロセスのどの段階で抜け落ちたのか、またその背景要因は何ですか？



次のステップに進んでも良いかを人間に確認

仕様策定の際、特殊条件などの要件抽出時のプロセス(手順)に漏れがあったため



分岐①をプロセス要因(要件不足)と認識し、深掘りが完了しました。次の分岐②に進めても良いかご確認ください。

はい、進めてください



5. 生成AI活用の実践と成果 (8/11)



次に分岐②を深掘りします。過去問題において「教訓が活かせなかった」背景には、どのような理由や要因があったのでしょうか？
例えば「影響範囲の誤認識」「調査不足」等、想定される具体的な原因理由を教えてください。

影響範囲を誤認識し、調査不足となり、結果、同じ問題を作り込んでしまった



分岐②をプロセス要因(調査不足)と認識し、深掘りが完了しました。
次の「見逃し原因」に進めても良いかご確認ください。



人間が方向修正(あくまで人間が主役)

調査不足となった原因をもう少し深掘りしてください



承知しました。過去問題において影響範囲を誤認し、調査不足に陥った背景や原因は何ですか？
例えば「問題の複雑性」「調査時間の制約」「引き継ぎが不十分」等、想定される具体的な原因理由を教えてください。

影響調査の手法として、調査範囲を絞らずに網羅的に調査する仕組みができていなかった



分岐②をプロセス要因(調査不足)と認識し、深掘りが完了しました。
次の「見逃し原因」に進めても良いかご確認ください。

⋮

5. 生成AI活用の実践と成果 (9/11)



「見逃し原因」に向けた深掘り分析

過去問題が見逃されていた理由は何と考えられますか？

具体的には「影響範囲の甘い分析」「過去問題の再利用の仕組み欠如」等、背景として想定できる原因を教えてください

レビューも特殊条件の確認する仕組みや影響調査が網羅的に行えているかの仕組みがなく、確認ができていなかった。
このため、レビュー時の確認内容が属人的になり、確認が漏れた。



「テスト漏れ原因」に向けた深掘り分析


適切なテストケースが作成されなかった背景には、どのような要因があると考えますか？

具体的には「テスト設計プロセスの不備」「特殊条件対応の漏れ」「テスト設計スキル不足」等、どのような課題が要因として挙げられるでしょうか？

テストも設計ベースで行ったために特殊条件を含めたテスト項目が無かった。
過去に発生した際も、追加のテスト項目を検討すべきだったが、そこまでの必要性を認識していなかった



●対策案(分析完了時)

1. 作り込み原因 [根本原因]
 - ・和暦特有の処理や例外条件(うるう年対応等)が要件抽出時に適切に考慮されていなかった。
 - ・設計を元にした要件抽出時に、特殊条件の対応要件が見落とされる仕組み上の欠点があった。
 [提案対策]
 - ・特殊条件や例外条件の対応を想定した「要件抽出用チェックリスト」を作成し、運用する。
2. 見逃し原因 [根本原因]
 - ・影響調査やコードレビュー時に、特殊条件や依存範囲の調査が十分に行われず、網羅性が欠如していた。
 - ・**レビュー工程自体が属人的**で、体系立ったチェック手順や仕組みが整備されていなかった。
 [提案対策]
 - ・特殊条件の確認、および影響調査作業を必須化する「レビュー用チェックリスト」を作成・導入する。
 - ・コードレビュー支援ツール(例: Crucible、GitHub Pull Requests)を導入し、機械的な確認のルール化を推進。
 - ・**別プロジェクトチームのメンバーを参加させる「クロスレビュー」を義務化する**  **人間では出なかったアイデアを提案**
3. テスト漏れ原因 [根本原因]
 - ・テスト設計が設計書ベースで行われた結果、未知の特殊条件や例外条件が網羅的にテストされていなかった。
 - ・過去に発生した問題の教訓が**次回以降のテスト設計に反映**される仕組みが構築されていなかった。
 [提案対策]
 - ・「テスト設計時のチェックリスト」を作成し、特殊条件や例外条件を必須化する項目を含める(異常系、特殊入力など)。
 - ・**過去問題の履歴をデータベース化し、新規リリースや保守時に再利用可能な形に整理する。**

5. 生成AI活用の実践と成果 (10/11)

● 当初想定していなかった副次的な効果 (1/2)



生成AIが分かりやすく整理してくれるため、関係者間での共通理解が進んだ

- AIは否定しないので、積極的に情報のインプットが進み、原因分析の過程でポロポロと事実が出てきた
- インプットした情報をAIが要約してくれるので、現場特有の用語や表現が一般化され、AIとのやり取りを通じて参加者の共通理解がすすみ、活発な会話に繋がった



5. 生成AI活用の実践と成果 (11/11)

● 当初想定していなかった副次的な効果 (2/2)

💡 生成AIを中心に会話することで、融和的なコミュニケーションに

なぜなぜ分析は【技術者 vs QA】になりがち…

- QAが技術者を攻めるように取られたり。
- 技術者にとって、あえて言われたくないことをQAが指摘するとギクシャクしてしまうこともあり、深掘りできないことも…

AIを使うと、【問題 vs 私たち(技術者&QA)】に変化！

- AIからの質問や提案に対しては、**心理的バイアスが低くなる**傾向があり、**壁打ち相手として活用**できそう！
- 技術者とQAと一緒に、AIの理解を深めるために工夫したり、**AIからの再発防止の提案も一般的な意見**として受け止め、検討の幅が広がるメリットもある

何を答えていいかわからない…

しんどい～！

Why?

Why?

Why?

Why?

Why?

情報が足りないかも？
他にインプットできる
情報はないかな？

こういう観点
もあるのか！

ヒントが
もらえる！

一般的には
テスト自動化は
主流なのね！

実現が難しい
ことも視野に
入れられる

6. まとめと今後の展望 (1/2)

- **生成AIを利用することで得られた効果**

- 分析の質とスピードが向上した
- 人間では気づきにくい観点や再発防止策を補完できた
- 生成AIが分析内容を分かりやすく整理してくれるため、関係者間での共通理解がしやすくなった
- 生成AIを中心に会話することで、融和的なコミュニケーションがとりやすくなった
『技術者 vs QA』 ➡ 『問題 vs 私たち（技術者&QA）』

- **活用にあたっての注意点**

- 生成AIはあくまで支援ツールであり、スキルを保有する人間の主導が必要

6. まとめと今後の展望 (2/2)

● 気づき

- 生成AIから期待通りの結果が得られない場合でも、**生成AIに「プロンプトをどう修正すればよいか」を相談することで、より良いプロンプト設計が可能になる**(Goal Seek Promptの考え方)
- プログラムのようにあらかじめすべてのケースを網羅する必要はなく、**会話の中で動的にやり取りすることが有効である**

● 今後の取り組み

- より効果的な分析を行うために、**生成AIに入力すべき情報の整理とテンプレート化を進める**
- 社内外の過去事例や知見を活用し、**ドメインに適した分析・提案を可能にする**

● 最後に

- **生成AIは人間に代わるものではなく、人間の力を引き出す強力なアシスタントです**
今後も、より良い品質づくりに向けて、生成AIの活用を進めていきたいと思います

ご清聴ありがとうございました