AI活用時代における品質保証の進化

― AI活用開発プロセスにおけるQAの再設計 ―

■チーム5
・大坪正晴：㈱ NTTデータ
・風見玲衣：㈱ JCHunter
・菅原広行：ソニーセミコンダクタソリューションズ㈱
・柳原靖司：ブラザー工業㈱
・山本正平：パナソニック エレクトリックワークス㈱

(敬称略・50音順)

目次

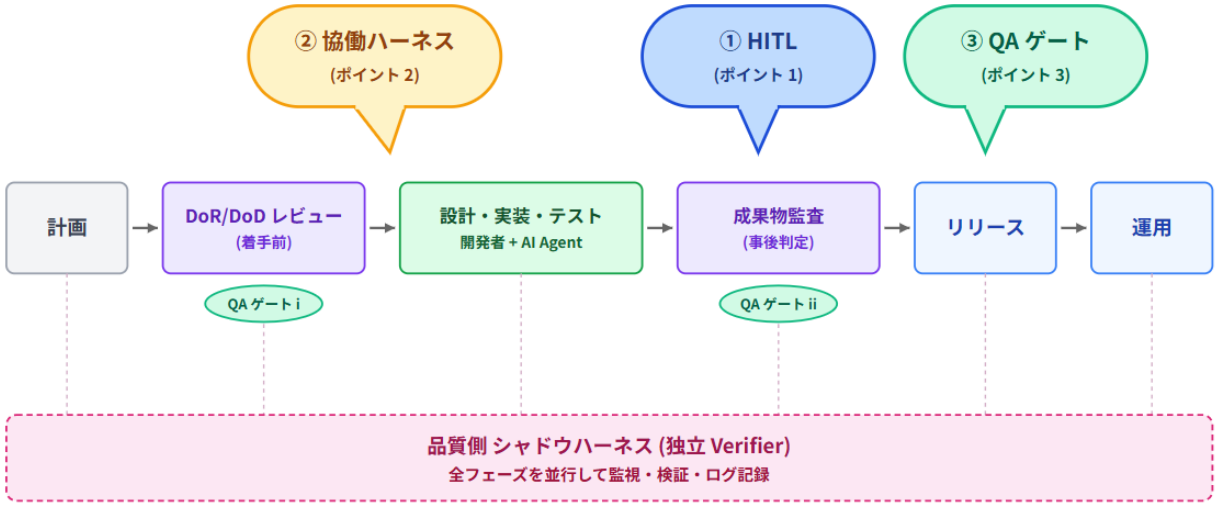
サマリ

本発表の問い
「AI活用開発において、QAはどう再設計されるべきか？」

注記：QEとはQuality Engineerの略。QA担当者と同義。

結論

- **QAは、AIの出力を後から検査する役割から、AIの使われ方そのものに品質を作り込む役割へ進化すべき**
- **QA再設計の３つのポイント**
  - **① HITL ― 残すべき判断点に人を置く**
  - **② 協働ハーネス ― AIの入力・出力・検証・差し戻しに品質機構を組み込む**
  - **③ QAゲート ― AIは判断支援まで、最終承認は人が担う**



アジャイルQAの枠組み × AI活用開発に固有の品質観点

「本提言は、メンバー所属企業における AI 活用プロジェクトの部分的運用実績と、 複数社で共有された現場知見を踏まえてまとめたものである」

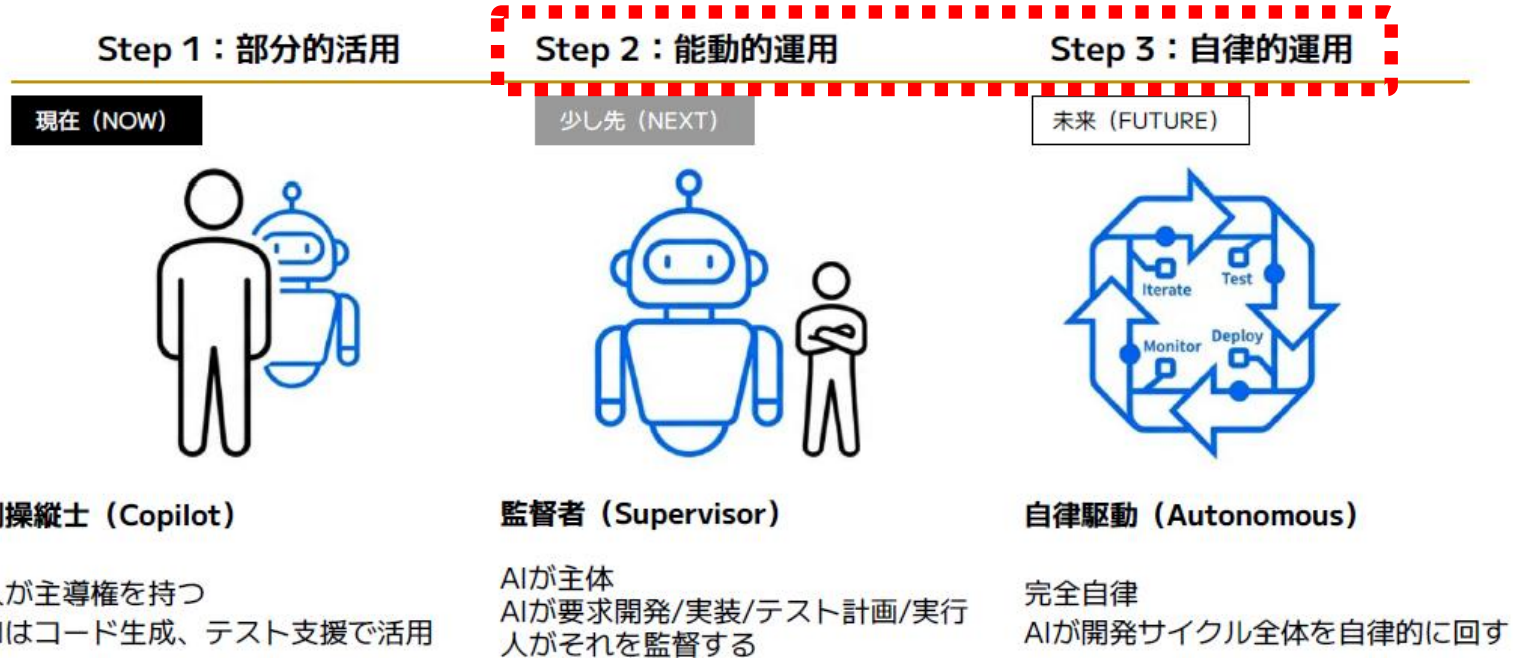
背景：開発スタイルの変化

生成AIの進化により、開発スタイルの変化が到来し、設計・実装・テストにAIが活用され始めている

開発生産性の向上が期待される一方、 AI成果物・AI利用プロセスに対する品質保証が課題となっている。

本発表では、**STEP2およびSTEP3**に焦点を当てる

時間軸での発展



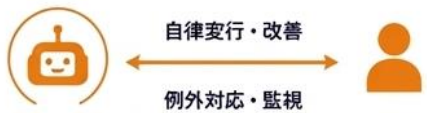
出典: Timee, AI時代にあわせたQA組織戦略, QA Tech Talk #7, Jan. 2026.

背景：人とAIの関与度

AI活用の進展を「人とAIの関与度」で3STEPに整理する。本発表では、人とAIが協働する STEP2(協働型) とSTEP3 (自律型) を対象とし、QAの役割設計の視点から論じる。

人とAIの関与度

人の関与：高い　　人とAIの協働　　人の関与：低い（例外対応中心）

	STEP1 AI駆動開発（ツール型） AIを“使う”開発	STEP2 協働型AI活用（協働型） AIと“働く”開発	STEP3 自律型AI活用 AIに“任せる”開発
特徴	AIは補助ツールとして 特定のタスクを支援 	上流工程からAIが提案・生成し 人間が監督・判断・承認 	AIが主体的に判断・実行し 人は例外時のみ介入 
AIの役割	個別タスクの支援（生成・補完・自動化）	提案・分析・生成・検証支援	自律的な計画・判断・実行・最進化
人の役割	指示、利用、最終判断	監督、評価、意思決定、承認	例外対応、ガバナンス、目標設定
意思決定の主体	人間	人間＋AI（協働）	AI（自律）
開発プロセスの特徴	従来の開発プロセスを維持 （AIは一部工程で活用）	上流工程からAIを活用し、 反復・フィードバックを重視	AIが全工程を挙動・自律的に最適化 継続的に進化
QAの主な対象	成果物（コード・仕様・テスト等）	AIの出力・判断プロセス・承認プロセス	システム全体の挙動・信頼性・ガバナンス
QAの役割・焦点	成果物の検証（テスト・レビュー）	判断の当当性保証・プロセス設計・監査	信頼性設計・リスク管理・継続監視
主なリスク	AIの出カミス、利用者依存、属人化	判断の形骸化、AI過信、基準の曖昧さ	暴走・予期せな最適化、 ブラックボックス化、制御不能

本発表の対象範囲

本発表は、AIを組み込んだ製品そのものではなく、生成AI・Agentic AIを開発プロセスに活用する場合の品質保証を主対象とする。

主対象：AIを使った開発プロセス

- 要件化・設計支援
- コード／テスト生成
- レビュー・判定支援
- Agentic AIによる自律的な作業
- AI生成成果物のリリース判断

など

関連領域（直接の主対象外）

- AI搭載製品／サービスそのものの品質
- 学習データ・モデル精度・公平性
- AIモデルの安全性・ロバストネス

※ 相互に関連するが、本発表では開発工程のQAに焦点を置く

焦点：AIが開発工程に入り込んだとき、QAの役割・判断点・統制をどう再設計するか

本活動の目的

AI活用による開発の変化を踏まえ、品質を確保しながらAIを有効活用するためのQAの役割と仕組みを示す。
そのため、次の3点を明らかにする：

Q1　AI活用によって、開発の進め方と人に残る役割はどう変わるか？

Q2　その変化によって、従来QAのどの前提が機能しにくくなるか？

Q3　その課題に対して、QAはどのように再設計されるべきか？

課題① 従来QAの前提が崩れる

▲問題　従来の品質保証は、暗黙のうちに3つの前提に立っていた。

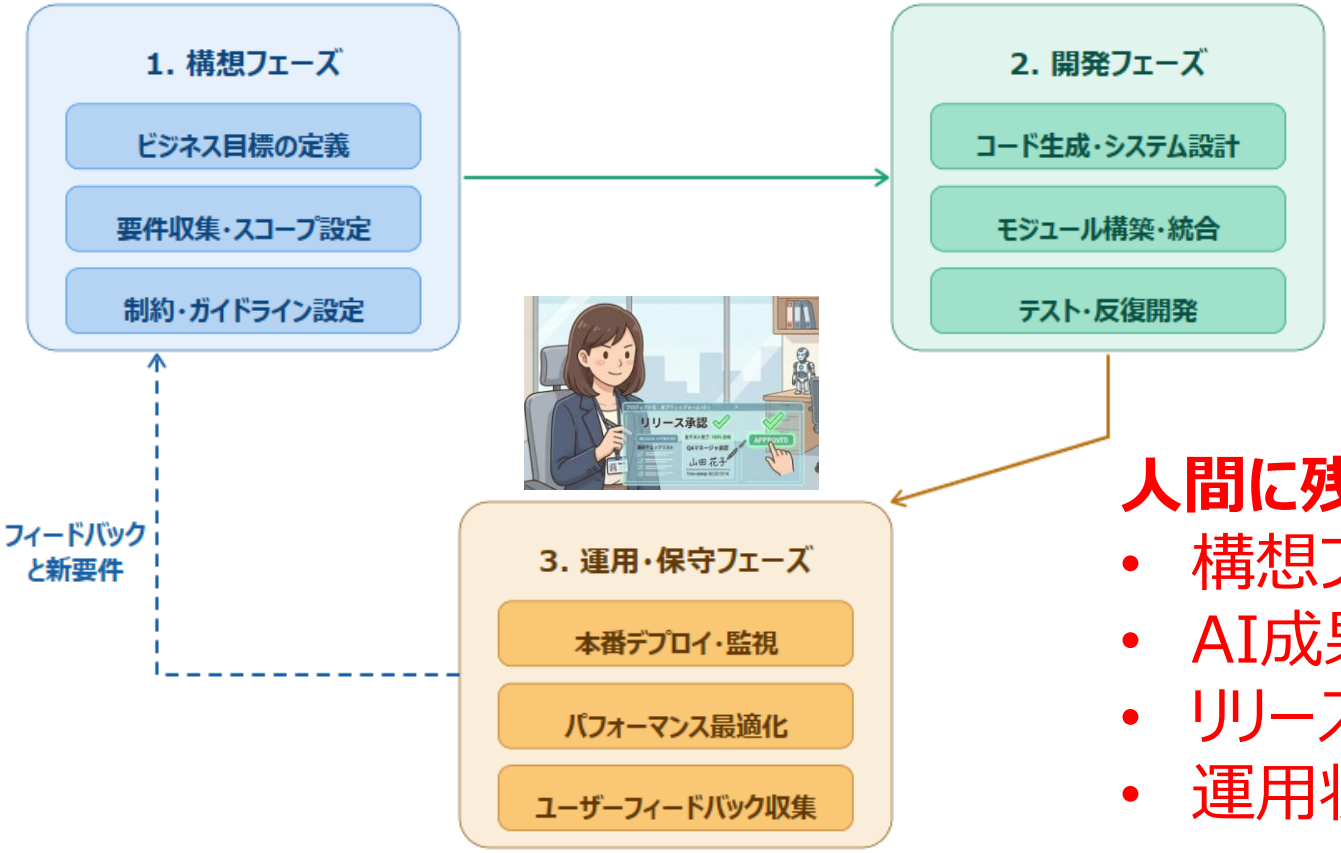
1. **人間が設計し、人間が実装する** ― 設計・実装等の成果物の意図を人が説明できる
2. **テストにより正しさを検証できる** ― Pass/Fail の二値判定で十分
3. **最終段階の検証で出荷可否を判断できる** ― 最終テストで品質を確認できる

従来の前提	AI活用後に起きること	**キーワード**
人間が設計・実装する → 意図は明示できる	AIが成果物を生成する際、要件の曖昧な部分を暗黙のうちに統計的に補完するため、意図・根拠のトレーサビリティを確保しにくい	**文脈理解の精度** **（意図の不可視化）**
Pass/Fail で検証できる	AI出力には揺らぎがあり、単発のPass/Failだけでは品質を十分に判断しにくい	**非決定性** **（非再現性）**
検証は後工程でよい	上流で生じた解釈のズレが、下流の実装・テストにそのまま流れ込み、最終工程で大きな手戻りとして顕在化する	**解釈のズレの伝播** **（上流依存性）**

課題② 人が担うべき判断と責任を、どこに残すか

AI活用開発の実態

— AIの自律度が高まるほど、人が判断・介入・責任を担うポイントを明示的に設計する必要がある



人間に残される役割

- 構想フェーズでの要件化支援
- AI成果物の監査
- リリース承認
- 運用状況の判断

課題③ QA判定ポイントが消失・変質する

文脈理解の精度

非決定性

解釈のズレの伝播

AI活用開発で起きる具体的な問題

― AIの自律度が高まるほど、従来のQA判定ポイントが消失・変質する

従来QAの前提	AI活用開発で起きること
開発と検証は時間的に分離	開発と検証が同時に高速回転
監査は結果ログを後追い確認	Agentic AIが複数工程を連続実行するため、後追い監査だけでは介入が間に合わない
検証対象は人が書いたコード	AI生成コードが大量・高頻度で流入
Pass/Failの二値判定	同じ入力でも結果が揺らぎ、判定の再現性を確保しにくい
監査者は外側に立つ	AI同士のレビューでは盲点が共有される(共謀的ハルシネーション)。 AI自身による自己採点では誤りが温存される(自己修正の盲点)。

課題④ 下流テストだけでは検出しにくい誤り

AI活用開発で起きる具体的な問題

文脈理解の精度

― AI由来の意味的な誤りは、下流テストだけでは検出しにくい

コーディングエージェントが
生成過程で混入する誤りの例
→ 厳密なチェックがないと見逃す

下流テストで検出しにくい4類型 (+ドメイン運用 1例) :

- 動いてしまうエラー　例：JavaScriptで findIndex() と find() の混同。型は通り、コードは動作する。
  TypeScriptのような型チェックでも捕捉されにくい。
- 環境差の見落とし　例：テスト環境(Node.js)では通るが、本番(ブラウザ)で挙動が変わる。
- 境界条件の混同　例：HTTPレスポンス(403 ⇔ 404)の誤認: 意味的には誤りだがテストが通過してしまう。
- 誤差の相殺　例：複数の誤った前提が打ち消し合い、結果としてテストが通ってしまう。
- ドメイン運用　例：FAQ自動応答で返金可否を誤って回答し、テストデータにも誤った期待値が混入する。

構造的問題： 要件の曖昧さ ＋ ドメインの暗黙知欠落 → ハルシネーションが入り込みやすい状況

QA再設計の基本方針

AI活用時代のQAは、AIの出力を後から検査するだけではなく、AIの使われ方そのものに品質を作り込む役割へ進化する必要がある

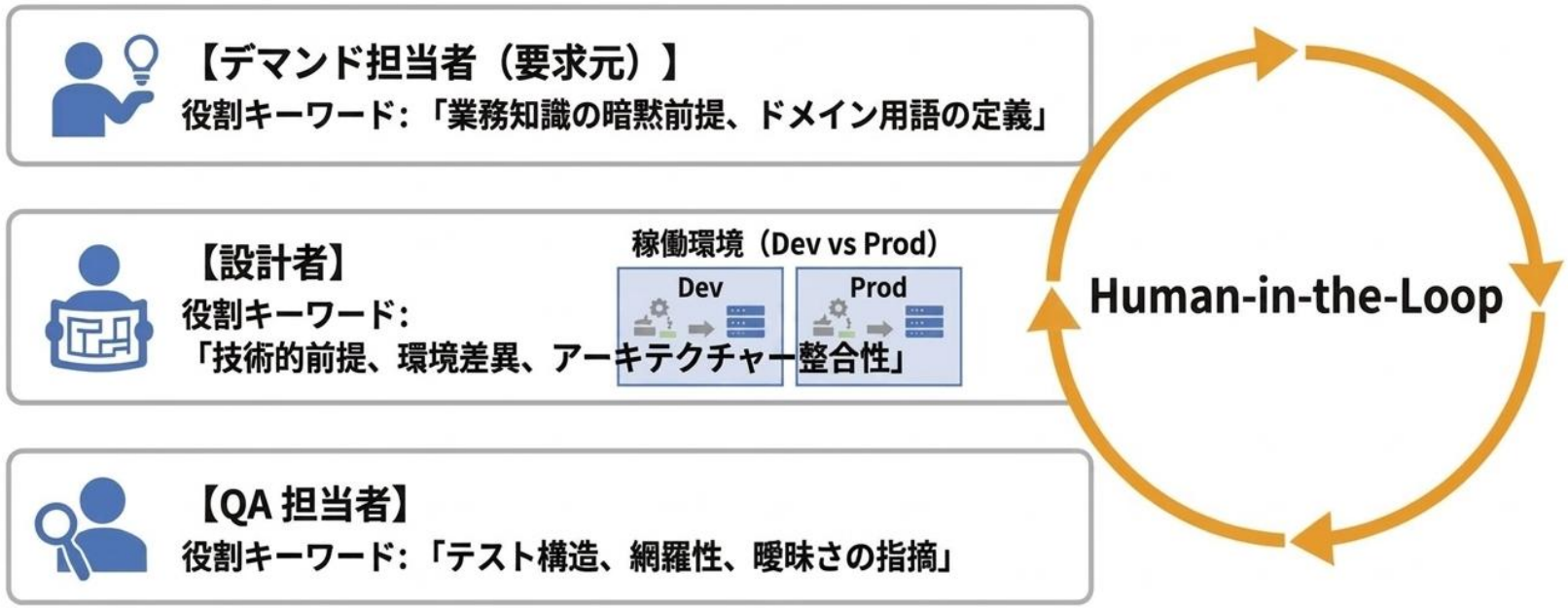
従来のQA	AI活用時代のQA
成果物を後工程で検査する	AI利用の入力・出力・判断点に品質確認を組み込む
人が作った成果物をレビューする	AI生成物、プロンプト、利用ログ、判断証跡を確認する
Pass/Failで判定する	揺らぎ・不確実性・残存リスクを含めて判断する
不具合を止める	誤った方向へ進まない仕組みを設計する
リリース前に品質を最終確認する	上流からHITL・ハーネス・QAゲートを組み込む

QAは、AIの利用を抑制するのではなく、AIを安全に活用するための「判断点・制御・証跡・承認」の仕組みを設計する

QA再設計のポイント①：HITL（Human-in-the-Loop）

HITL (Human-in-the-Loop)
— 人をAIの自律的開発の作業ループに組み込む。全工程に人を挟むのではなく、**残すべき判断点に人を置く**

◎ハルシネーションは、ドメイン知識の暗黙前提と本番環境の前提条件に集中する。
① AI 同士のレビューは共謀的ハルシネーションを起こす ② AI の自己修正には盲点がある



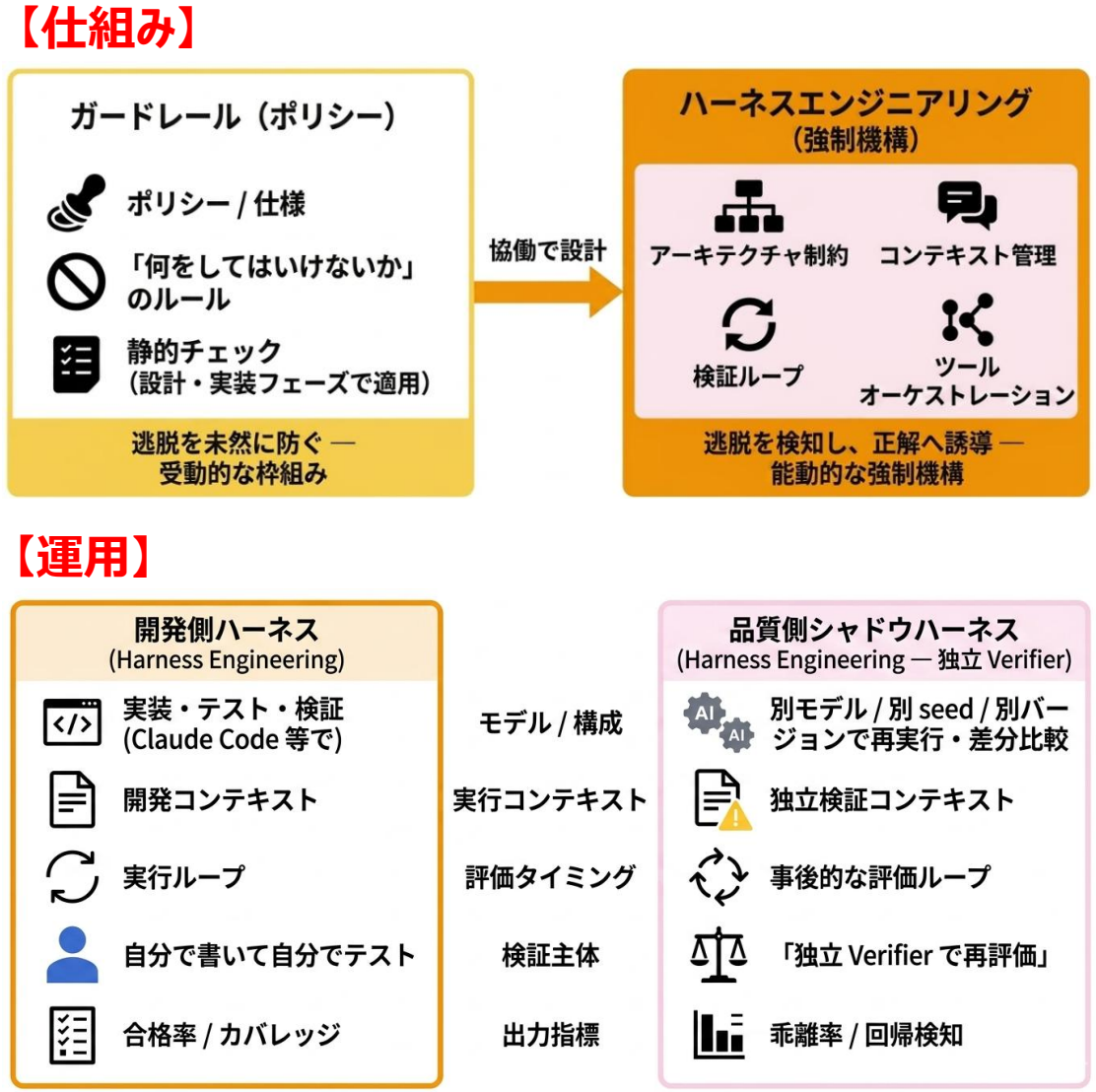
QA再設計のポイント②：協働ハーネス

協働ハーネス

ガードレール(ポリシー)＋ハーネスエンジニアリング(強制機構)・実行・検証・制御機構

AIの入力・出力・レビュー・差し戻し・ログを
開発者とQA担当者が協働で設計する

◎従来の「止めるQA」から
「正しい方向へ戻す能動的QA」へ

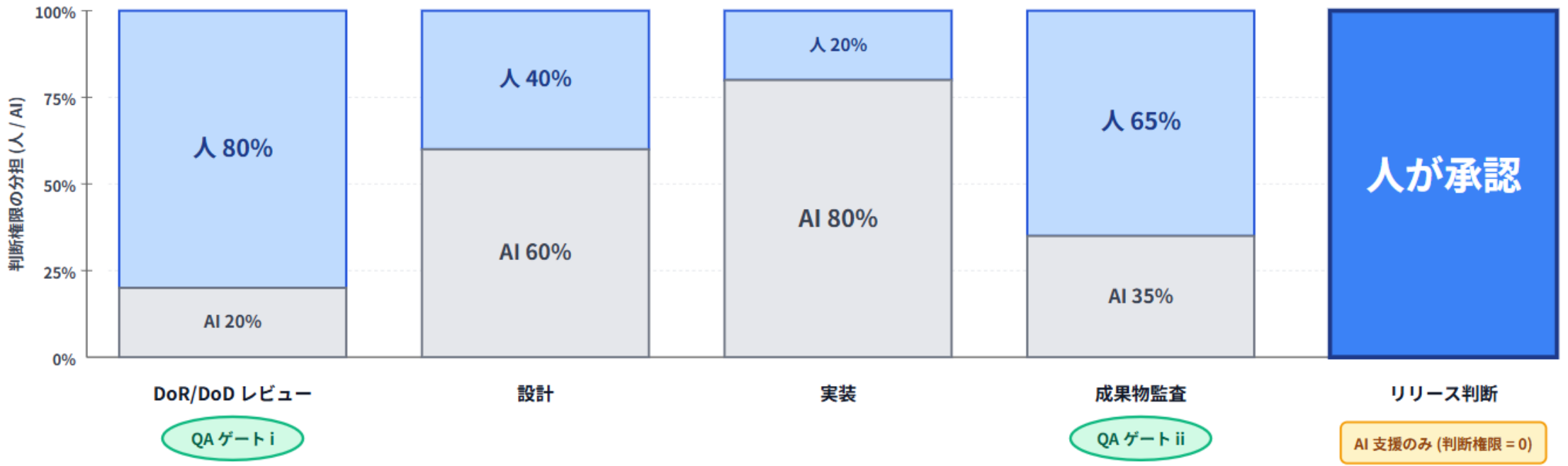


QA再設計のポイント③：QAゲート

QAゲート

完了条件の定義やリリース承認は AI に任せない

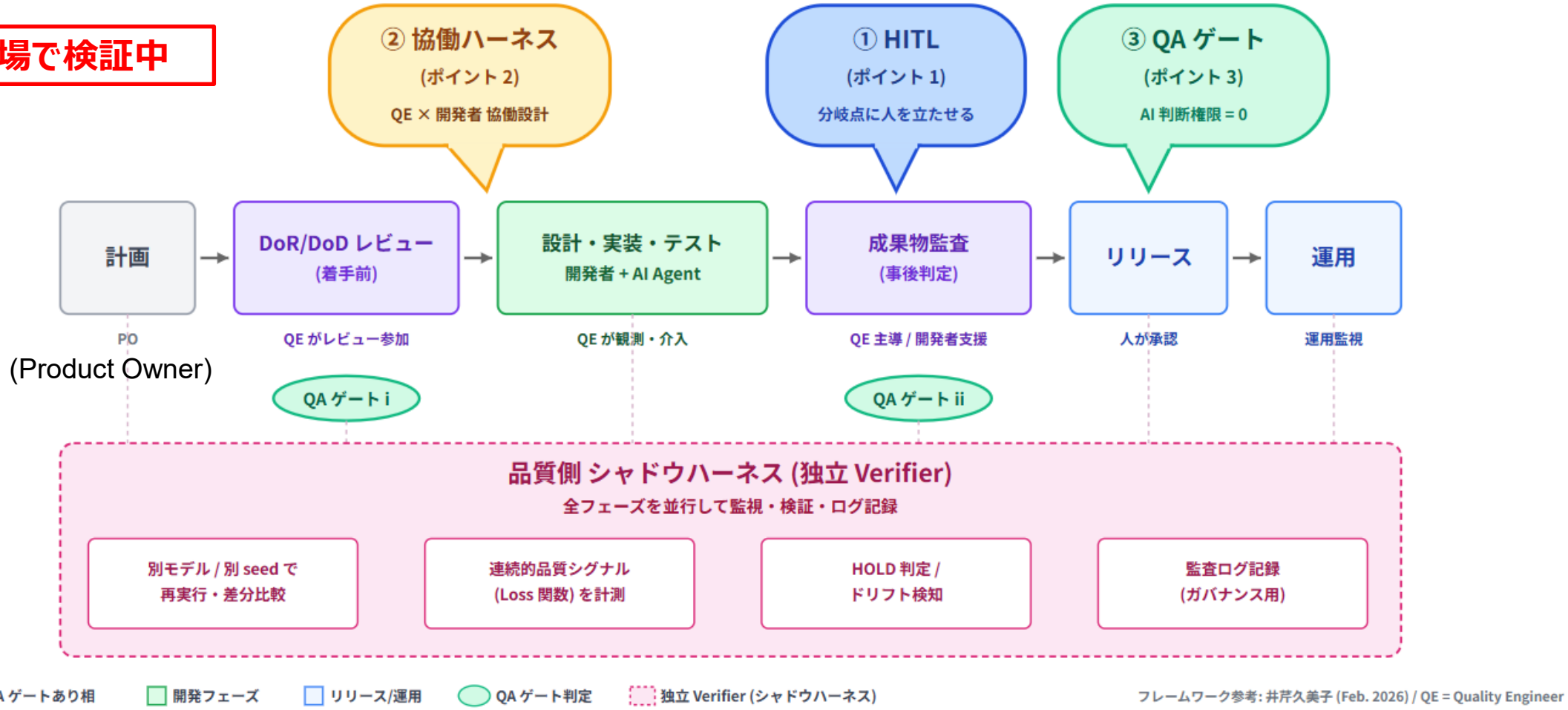
AI は品質エビデンスを集約・提示する (Decision Support) が、
承認責任は人が担う



提案モデルの全体像

アジャイルの枠組みに、AI活用開発における品質の勘所を組み込んだ融合フレームワーク

現場で検証中



AI活用現場でまず始めること

高自律なAI活用(STEP3) へ一気に進むのではなく、まずは、**協働型AI活用(STEP2) において、人の判断品質を維持する仕組みから整備**する。

観点	最初に実施すべきこと	目的	対応するポイント
AI利用管理	利用モデル・プロンプト・入出力ログ記録・追跡性・再現性の向上	再現性確保	ポイント② 協働ハーネス
品質監査	AI利用箇所・生成経緯の記録	監査可能性向上	ポイント③ QAゲート
HITL設計	人間承認ポイント定義	AI過信防止	ポイント① HITL
出力品質	複数案比較レビュー	判断品質向上	ポイント① HITL
モデル管理	利用モデル・バージョン管理	出力揺らぎ抑制	ポイント② 協働ハーネス
品質監視	利用モデル・サービス変更と出力傾向の監視	継続品質確保	ポイント② 協働ハーネス

QA再設計の適用方針

ー リスクに応じてQAの関与レベルを変える

すべてのAI利用に同じ重さで適用するのではなく、品質への影響とAIの自律度に応じてQA関与を段階化する。

AI利用区分	利用例	主な品質リスク	QAの関与レベル
個人利用	調査・学習・壁打ち	誤情報・理解違い	原則関与なし
業務支援	文書作成・要約・コード補助	成果物への誤り混入	利用ルール・自己確認
業務判断支援	レビュー・品質判定の支援	誤判断・説明責任不足	ルール・教育・監査
製品・サービス品質影響	**AI生成コードの自動反映 AIによるテスト結果の自動判定 AIによるリリース判断 AIによる自律的な変更・実行**	**不具合・顧客影響・誤リリース**	**QAレビュー・証跡・承認ゲート**

3つのポイントの適用範囲と強度を、品質影響とAIの自律度に応じて調整する

提案モデルにより期待される効果

アジャイルの枠組みに、AI活用開発における品質の勘所を組み込んだ融合フレームワーク

このモデルのベネフィット：

HITL: 人の判断を支点とすることで誤った自動判断を防ぐ

協働ハーネス: AIの入力・出力・検証・差し戻しに品質を作り込む

QAゲート: 品質エビデンスに基づき、人が最終承認する

今後の課題

- **シャドウハーネスの合理性** ― 品質担当部門のQA担当者への教育(能力開発)、開発側と品質側の役割分担・検証手段
- **品質シグナルの設計** ― 現場の技術的知見の蓄積(統計データの組み込み等)

  例：ドリフト検知率／差し戻し率／本番障害件数など、何を主指標にすべきか？
- **AIによる検証バイアスの管理** ― AIモデルの検証能力をどのように評価・監視するか？
- **QA人材・組織能力** ― AI利用設計・データ分析・監査能力の強化

まとめ

Q1　開発の進め方と人に残る役割はどう変わるか？

AIが設計・実装・テストを高速に反復する一方、人には、意図の明確化、重要判断、監督、リリース承認が残る。

Q2　従来QAのどの前提が機能しにくくなるか？

「人が作る」「Pass／Failで判定する」「後工程で検証する」という前提は、意図の不可視化・非決定性・解釈のズレの伝播によって、それだけでは十分に機能しにくくなる。

Q3　QAはどのように再設計されるべきか？

QAは、AIの出力を後から検査する役割から、AIの使われ方そのものに品質を作り込む役割へ進化する。

HITL　— 残すべき判断点に人を置く
協働ハーネス — AIの入力・出力・検証・差し戻しに品質機構を組み込む
QAゲート　— AIは判断支援まで、最終承認は人が担う

※各ポイントの適用範囲と強度は、品質影響とAIの自律度に応じて調整する

品質を確保しながらAIを有効活用できる仕組みを設計することが、AI活用時代のQAの役割

ご清聴ありがとうございました！

用語集

用語	階層・分類	意味	特徴・例
AIネイティブ開発	パラダイム (設計思想)	AIをプロダクトの中核として前提設計する開発。AIなしでは成立しない。	ChatGPT、Copilot系プロダクト、自然言語で操作する業務アプリ。 **AIが主役**
AI駆動開発	手法 (プロセス)	生成AIで開発プロセス(設計・実装・テスト等)を加速する手法。プロダクト自体はAI非依存でよい。	GitHub Copilotによるコード補完、テストケースの自動生成。 **AIは相棒・道具**
AI-DLC	フレームワーク (ライフサイクル管理)	AI版SDLC。データ品質・モデルドリフト等、AI特有の不確実性を前提とした管理モデル。	要求分析→実装→評価→デプロイ→運用・監視。運用後の改善ループが従来SDLCとの違い。
ハーネスエンジニアリング	技術領域 (制御設計)	AIを安全・安定・再現可能に使うための制御構造を設計する技術。「AIを野放しにしない」工学。	入力制御(プロンプト・コンテキスト)、出力検証、ガードレール、ログ・監査。 **AI品質保証の中核技術**
協働ハーネス	運用形態 (協働設計)	ガードレールとハーネスエンジニアリングを、開発者とQA担当者が協働で設計・運用する仕組み。	開発側ハーネスと品質側シャドウハーネスの二層で運用。 **「止めるQA」から「正しい方向へ戻す能動的QA」へ**
共謀的ハルシネーション	失敗様式	AI同士のレビューで、共通の学習基盤に由来する盲点が共有され、誤りが検出されないまま伝播する現象。	同系統モデルの相互レビュー → 同じ誤りを見逃す。独立Verifier(Shadow Harness)が必要な根拠。
シャドウハーネス	検証機構	品質部門が開発側とは独立に保有する検証ハーネス。別モデル・別seedで再実行し差分を比較する。	共謀的ハルシネーションへの構造的対策。 **協働ハーネスの「品質側」を担う**
連続的品質シグナル	内部信号 (AI内部制御層)	AIが自分の出力品質を数値化し、次の挙動を自律的に修正するための連続値スコア。	Loss = w_1·AT + w_2·AR + w_3·HD + w_4·VC の合成指標。**協働ハーネスを動かす内部信号**

参考文献

[1] A. Bandi et al., "The Rise of Agentic AI: A Review of Definitions, Frameworks, Architectures, Applications, Evaluation Metrics, and Challenges," Future Internet, vol. 17, no. 9, Art. 404, Sep. 2025. https://www.mdpi.com/1999-5903/17/9/404

[2] D. Robbins et al., "AI Assurance: A Repeatable Process for Assuring AI-Enabled Systems," MITRE Technical Report PR-24-1768, Jun. 2024. https://www.mitre.org/sites/default/files/2024-06/PR-24-1768-AI-Assurance-A-Repeatable-Process-Assuring-AI-Enabled-Systems.pdf

[3] Z. Zhang et al., "LLM Hallucinations in Practical Code Generation: Phenomena, Mechanism, and Mitigation," Proc. ACM Softw. Eng., Jan. 2025. arXiv:2409.20550

[4] R. Arike et al., "Technical Report: Evaluating Goal Drift in Language Model Agents," arXiv:2505.02709, May 2025.

[5] S. Rabanser et al., "Towards a Science of AI Agent Reliability," arXiv:2602.16666, Feb. 2026.

[6] S. Mehta et al., "Beyond Accuracy: A Multi-Dimensional Framework for Evaluating Enterprise Agentic AI Systems," arXiv:2511.14136, Nov. 2025.

AI活用時代の品質保証の留意点

従来の「人がプロセスで品質を作り込む」やり方と、「AIが工程内で品質を作り込む」やり方には、類似点と本質的な差異がある。
従来手法をそのまま適用すると機能不全に陥る。

AI活用時代の品質保証の留意点

— AIネイティブ開発を前提とした場合、「最終監査者」としての品質部門の役割は機能しない。

I. 従来手法の継承可能な部分を保ちつつ、
II. AIに固有の「非決定性・自己修正・共謀的ハルシネーション・常時ドリフト」に対応する

従来QAを否定するのではなく、「どこを継承し、どこを変えるか」を整理する必要がある。

連続的品質シグナル

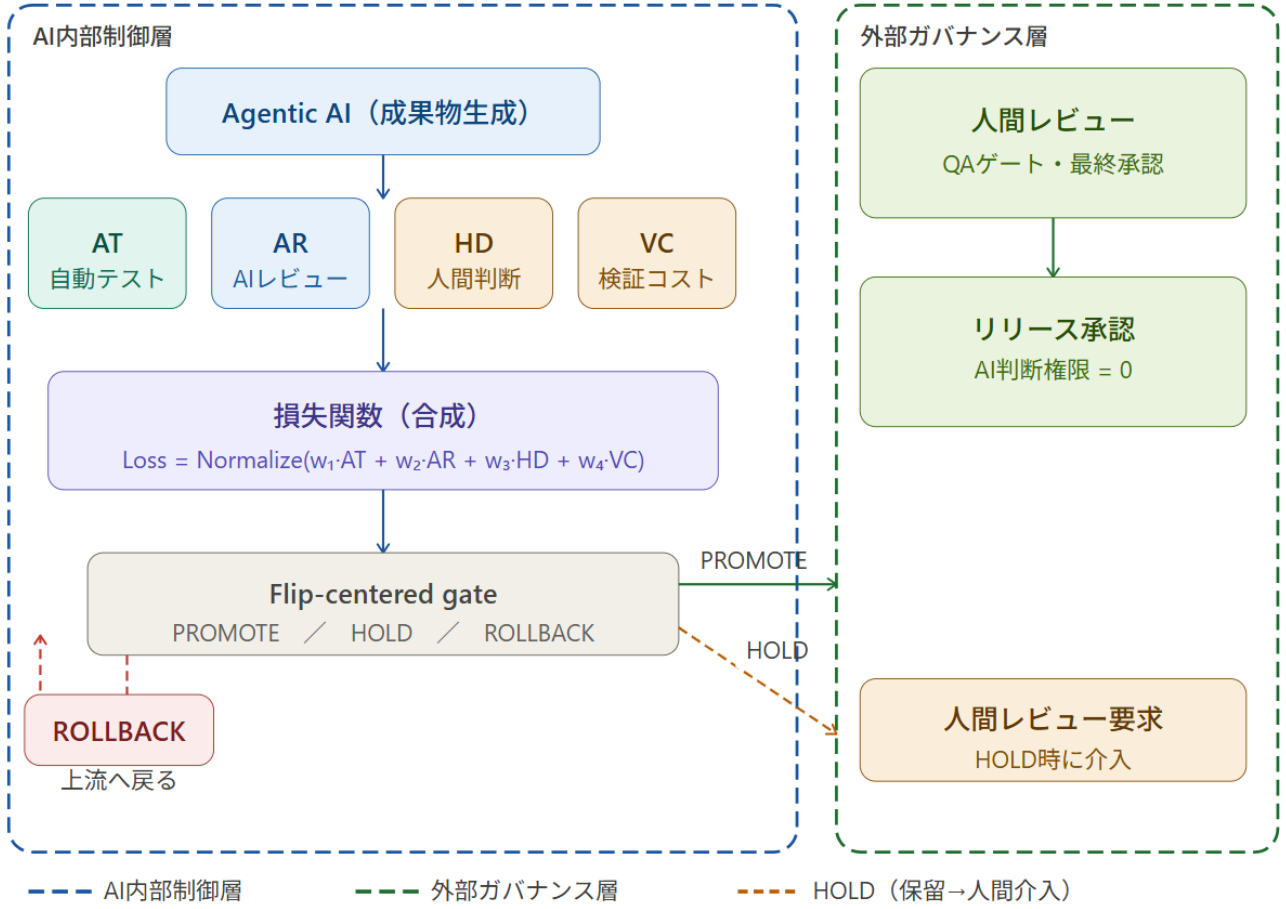
1. 全体像

AI-DLCにおける連続的品質シグナルとは、Agentic AIが生成したソフトウェア開発成果物について、客観的テスト結果（AutoTest）・独立AIレビュー（AIReview）・人間の意味的判断（HumanDecision）・検証コスト（VerificationCost）を、開発フェーズに応じた動的重みで合成し、かつ因子間の条件付き依存関係と時間減衰を考慮した乖離量スコアである。このスコアはN回の反復実行による統計量として算出され、個別ケースのflip方向を監視するPROMOTE/HOLD/ROLLBACKの三値ゲートと組み合わせて、AIの自律的な前進・保留・差し戻しを判定する。ただし、本信号は計測可能な品質次元における最善のナビゲーション信号であり、サイレント品質劣化やコンテキストドリフトに対しては、出力特性分布の異常検知層を並行して運用する必要がある。リリース判断フェーズでは、AIの判断権限はゼロとするが、品質エビデンスの集約・提示（Decision Support）は維持する。

【補足】アーキテクチャ上の位置づけ

連続的品質シグナルは、AI-DLC全体の品質保証アーキテクチャにおける「AI内部制御層」に属するナビゲーション信号である。すなわち、Agentic AI自身が自分の出力品質を定量化し、次の振る舞いを自律的に修正するための内部フィードバック機構である。一方、ガードレール（ポリシー）・QAゲート（DoDレビュー・成果物監査）・品質側シャドウハーネス（独立Verifier）・HITL（Human-in-the-Loop）は、いずれも「外部ガバナンス層」に属し、AIの出力を外側から監督・承認・差し戻す責務を負う。この二層構造を混同してはならない。連続的品質シグナルは従来のPass/Failゲートを置き換えるものではなく、AIが自己改善ループを高速回転させるための内部信号として機能する。最終的な合否判定・リリース承認・ガバナンス上の説明責任は、引き続き外部ガバナンス層（明示的ルール＋人間の判断）が担う。この層分離により、AIは連続的シグナルを用いた高速な自己最適化を行いつつ、人間とガバナンス機構は「なぜその判断に至ったか」という説明責任と監査証跡を確実に確保できる。

付録



以下の合成指標 **Loss**値により、
AIの振る舞いをAIに自律的に判断させる

AT: 自動テストの結果
（例：単体テスト、コードカバレッジ）

AR: AIレビューの結果
（例：要件カバレッジ、テストコード品質）

HD: 人間の判断結果
（例：ドメイン暗黙知の検証）

VC: 検証コスト
（例：カスケード障害の波及コスト）

付録

2．連続的品質シグナルの補足

2-1. HOLD判定後の運用ルール

HOLDは「判定の保留」を意味し、放置すべき状態ではない。

※HOLD発動時の標準運用ルールは以下のとおりとする

- HOLD発動時: 自動再試行を1回実施し、揺らぎ由来の判定変動かを確認する。
- 再試行後もHOLDが継続する場合: 人間（QA担当者または設計者）にレビューを依頼する。
- 人間レビューの結果: PROMOTE承認、ROLLBACK指示、または再プロンプトによるAIへの差し戻しのいずれかを選択する。
- 連続HOLD回数の上限: 3回を超えた場合は自動リトライを停止し、人間介入に強制切り替えする。　※無限ループ防止

2-2.因子の性質に応じた統計処理の分離

N回統計量の算出は、因子の性質に応じて以下のように分離する。

- AutoTest (決定論的): 単回実行で十分。同じ入力に対し同じ結果が返るため、統計集約は不要。
- AIReview (非決定論的): N回実行の中央値・信頼区間を用いる。同一成果物に対し評価AIの出力が揺らぐため、統計集約により信頼性を確保する。
- HumanDecision (揺らぎあり): 複数レビュアー間の合意度を考慮する設計が望ましい（必要に応じて）。
- VerificationCost (計測値): 単回計測で十分。コストデータは決定論的に算出可能。