

なぜAI活用が進むほど、 エンジニアは考えなくなるのか？

～パイプライン構築で見た便利さの副作用と、手放してはいけない仕事～

2026/07/16 木曜日

NECソリューションイノベータ

ソリューションサービス事業ライン／PFSI事業部門／デジタルPF統括部

伊達 拓郎

AIで「業務がラクになった」
と感じている人

AIによって手放した仕事（作業）
ありませんか？

今日のテーマ

—— 手放した仕事と、それに気づいた話 ——

AIの活用が加速し、日々進化していく



漠然とした、不安

経験から、その不安への気づきと、解決のきっかけをお話しします。

自己紹介



伊達 拓郎 *だて たくろう*

NECソリューションイノベータ

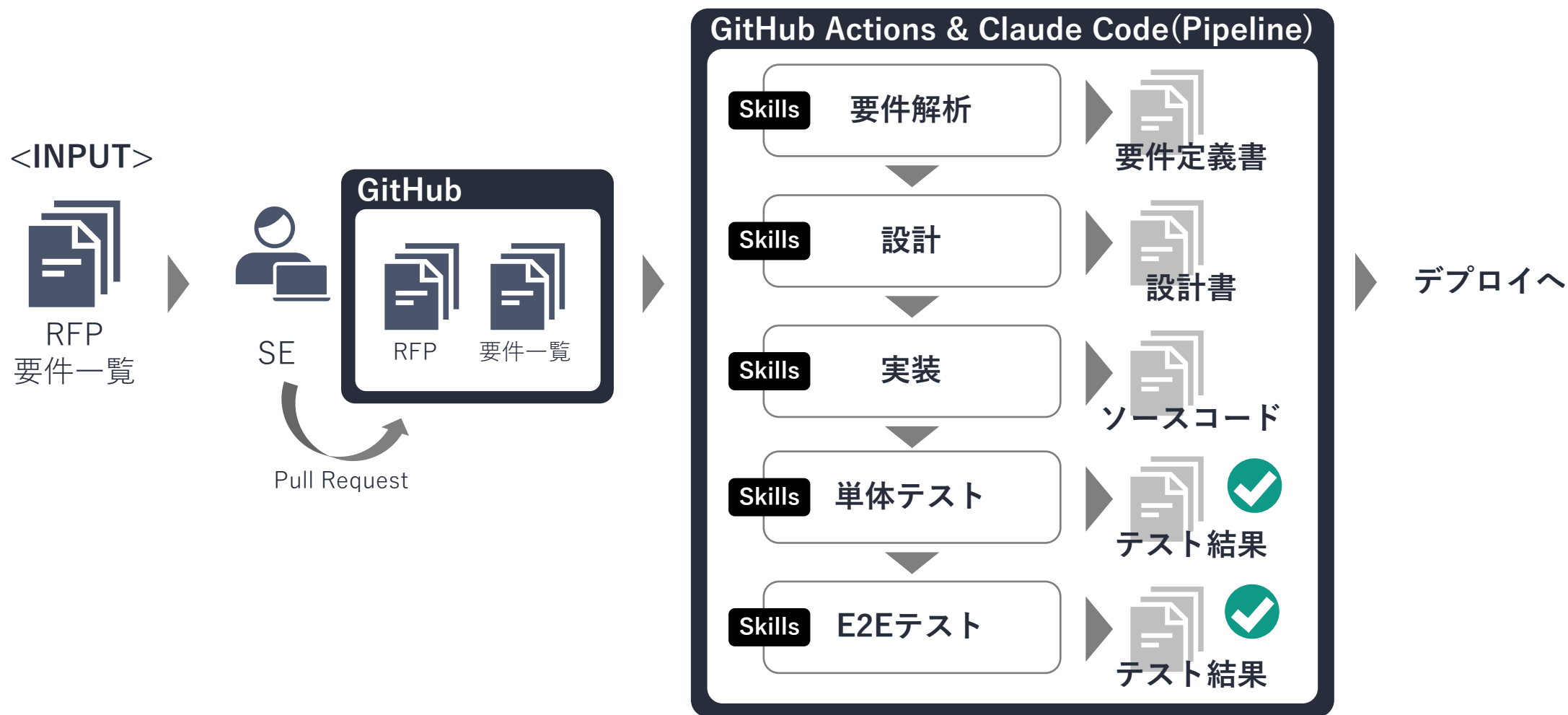
PFSI事業部 / デジタルPF統括部

2004年入社。マイグレーション → アプリ開発 → サービス企画・地域DX → AI
開発推進

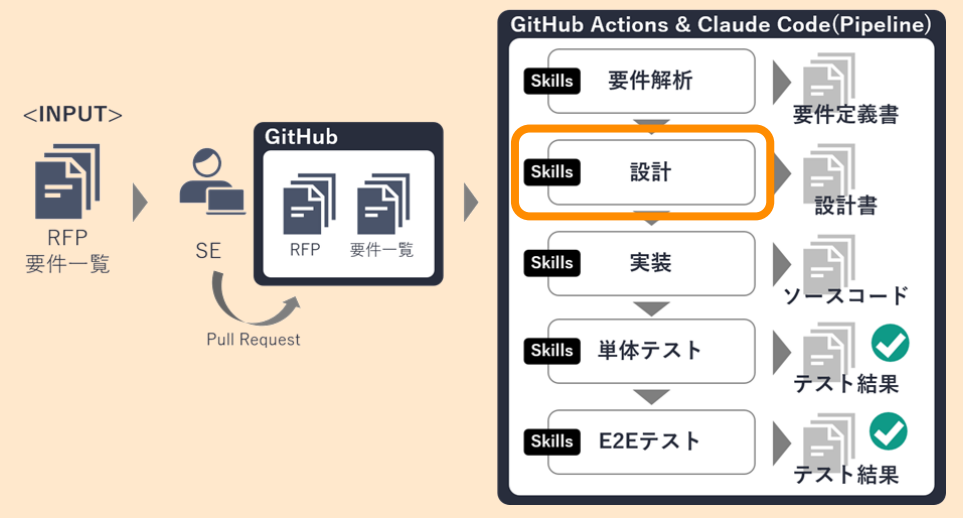
現在：AIネイティブな開発プロセスの検証・適用推進

実プロジェクトでAI駆動開発パイプラインを構築・運用

要件定義から検証までAIが担うパイプラインを構築



参考　－設計－



設計書管理システム (MVC)		
項目	内容	
画面ID	DESIGN-001	
画面名	設計書管理画面	
対象ユーザ	設計者、レビュー担当者	
改題概要	設計書の作成、更新、削除、検索機能の実装。既存の設計書データの移行も行う。	
関連API	設計書の取得、設計書の作成、設計書の更新、設計書の削除	
作成日	2024-01-15	
バージョン	1.0.0	
HTMLレイアウト	設計書のリスト表示、詳細表示、検索フォーム、操作ボタン	

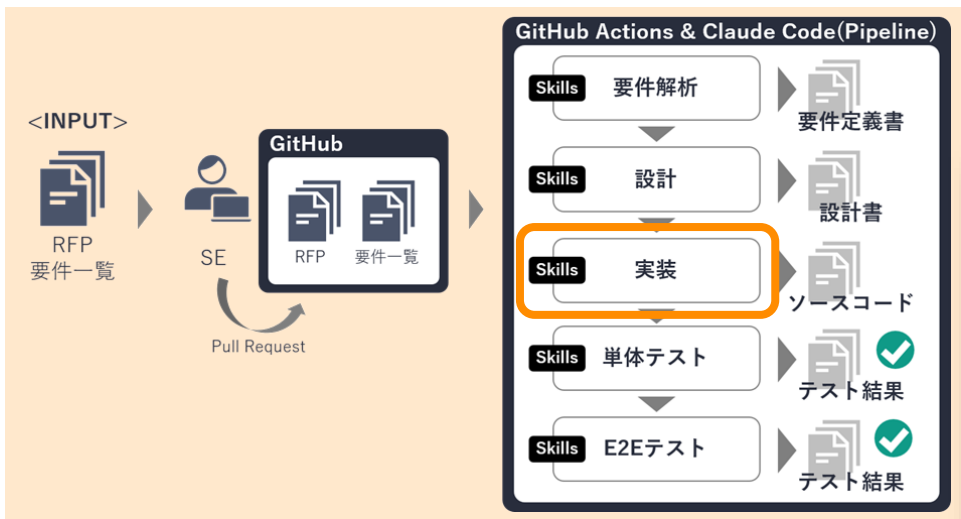
1. 改題の背景・現状の課題

#	課題	影響
1	設計書の作成、更新、削除機能の実装	設計書の管理が容易になる
2	設計書の検索機能の実装	設計書の検索が容易になる
3	設計書の移行機能の実装	既存の設計書データの移行が容易になる

2. 画面構成

```
設計書管理画面 ├── 設計書のリスト表示 │   ├── 検索フォーム │   ├── 設計書の検索結果 │   └── 設計書の操作ボタン ├── 設計書の詳細表示 │   ├── 設計書の検索結果 │   └── 設計書の操作ボタン └── 設計書の作成・更新・削除機能
```

参考 — 実装 —



— 実装フロー完了 ✓

ai-implementation フローの全ステップを完了し、 `claude/issue-44-20260603-1100` に push しました。

- ✓ 前提条件の確認 (roadmap 実装ガード) — 前提なし
- ✓ 設計成果物 (`claude/issue-44` から取り込み) ・ 設計書MD ・ 既存コード調査
- ✓ `/test-driven-development` — テスト先行 (RED → GREEN)
- ✓ `/speckit-implement` — 実装
- ✓ `/verification-before-completion` — lint / test / build 全PASS
- ✓ `/requesting-code-review` — `specs/044-new-shop-management/review.md` 出力
- ✓ コミット → push

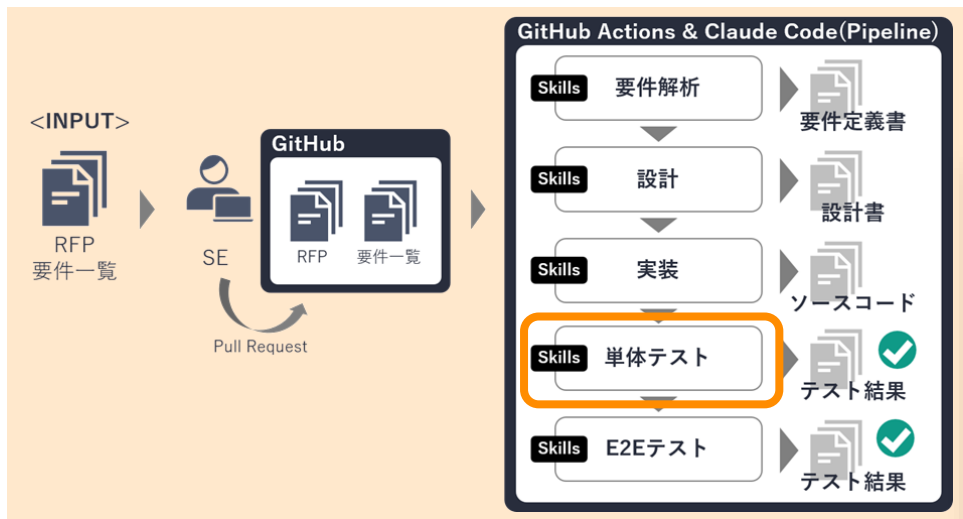
実装内容

BO API (既存ファイルへ追加のみ・破壊的変更なし)

- 新規6API: `/api/v1/shop-management` / `/api/v1/shop-management/{id}` / `/api/v1/shop-management/{id}/items` / `/api/v1/shop-management/{id}/items/{id}` / `/api/v1/shop-management/{id}/items/{id}/items` / `/api/v1/shop-management/{id}/items/{id}/items/{id}`
- 全APIで KBN=1 権限チェック (E0003) + 入力バリデーション (E0001)
- CSV解析 (Shift_JIS→UTF-8 / iconv-lite) ・ Cognito連携 ・ 部分失敗継続 (failedRows)
- `ShopService` / `ShopController` / `ShopDb` / `routes/Index.js` / `package.json` (iconv-lite追加)

BO Web (新規5ページ + 子コンポーネント)

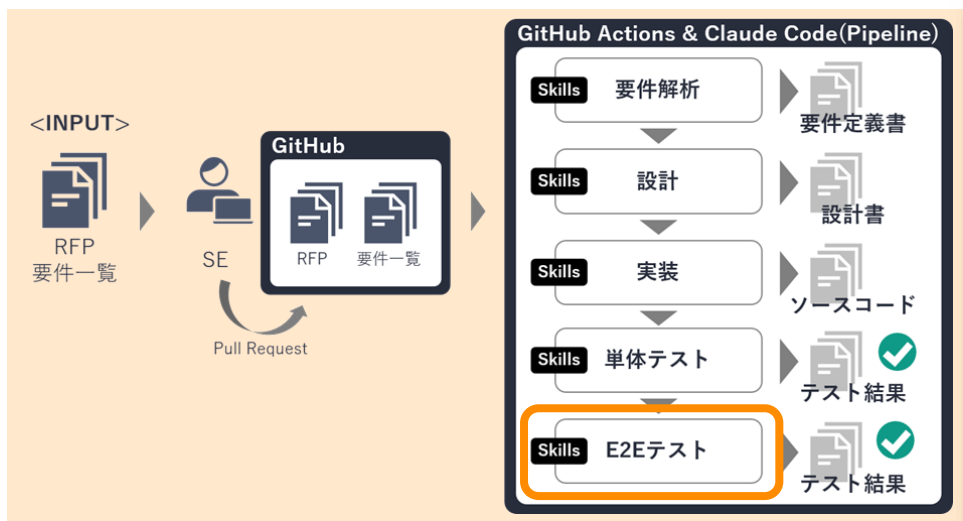
参考 – 単体テスト –



Lint, ビルド, テスト, 脆弱性監査を実施

The screenshot shows a GitHub Actions workflow run for a repository named 'dev-conet-iitokodori'. The workflow is triggered by a pull request. The status is 'Success'. The total duration is 3m 49s. The workflow file is 'ci.yml' and the event is 'pull_request'. The workflow steps are: 'build' (1m 31s) and 'internal-e2e' (2m 8s). The 'internal-e2e' step is expanded, showing a list of jobs: 'lint-web' (34s), 'lint-api' (36s), 'test-api' (49s), 'test-web' (38s), and 'audit' (14s). All jobs are marked as successful with blue checkmarks.

参考 – E2Eテスト –



Playwrightでの実施

The screenshot displays the Playwright test results for the project "e2e". The test suite "new-school-management.spec.js" is shown with a list of tests. The first four tests are marked as failed (red X), while the remaining tests are marked as passed (green checkmark). The test results are as follows:

Test Name	Duration
- 運用管理者 (KBN=1) > SC-010 一致しない学校名で検索すると0件メッセージを表示する (e2e)	13.4s
- 運用管理者 (KBN=1) > SC-001 未利用シード校を更新 (利用中化) し成功トーストを表示する (e2e)	2.0m
- 運用管理者 (KBN=1) > SC-004 学年クラスを編集して更新 (フルリプレース保存) する (e2e)	2.0m
- 運用管理者 (KBN=1) > SC-005 販売店検索ダイアログから紐付けを追加して更新する (e2e)	2.0m
- 運用管理者 (KBN=1) > Fill "1" locator("grade-row").nth(1).locator("input").first() — new-school-management.spec.js:229	63ms
- 運用管理者 (KBN=1) > Fill "B" locator("grade-row").nth(1).locator("chip-input") — new-school-management.spec.js:230	45ms
- 運用管理者 (KBN=1) > Press "Enter" locator("grade-row").nth(1).locator("chip-input") — new-school-management.spec.js:231	27ms
- 運用管理者 (KBN=1) > Click getByRole("button", { name: "更新", exact: true }) — new-school-management.spec.js:232	41ms
- 運用管理者 (KBN=1) > Expect "toBeVisible" getByText("同じ学年が複数あります") — new-school-management.spec.js:233	13ms
- 運用管理者 (KBN=1) > After Hooks	68ms

Below the test results, a "Screenshots" section shows a screenshot of the application under test. The screenshot displays a web interface with a sidebar menu, a header, and a main content area. The main content area shows a form for "学年クラス設定" (Grade/Class Setting) with fields for "学年" (Grade) and "クラス" (Class). The form includes a "検索" (Search) button and a "更新" (Update) button. The interface is in Japanese.

動かすほどに、見えてきたもの

日々の開発は
確かに、回っていた



けれど、現場から
課題が、顔を出し始めた

便利なはずだった。でも——3つの声が、引っかかった

お客様

[伝わる]

「何から確認すれば
？ 読み方を教えて」

→ お客様目線を、忘れて
いた

部下

[判断]

「AIから、できない
と返ってきました」

→ AIの判定を、うのみに
していないか

同期

[やりがい]

「この“確認だけ”の作
業、誰が楽しいの？」

→ 効率化から、抜けてい
た観点

——どれも、最初は小さな違和感だった。

この3つの違和感に、1つずつ、向き合うことにした。

3つの違和感

1

伝わる

2

判断

3

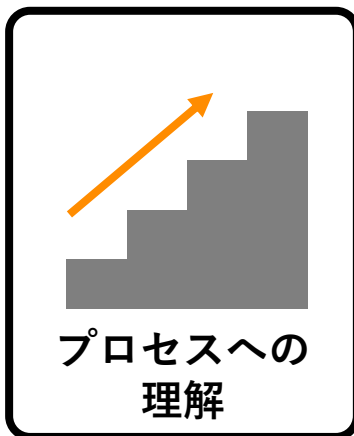
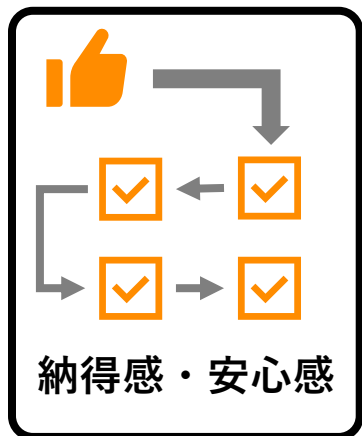
やりがい

〔 伝わる 〕
お客様

正確なのに、なぜ「伝わらない」のか？

AI駆動開発における「顧客との対話減少」と「成果物のギャップ」

「対話」による共創



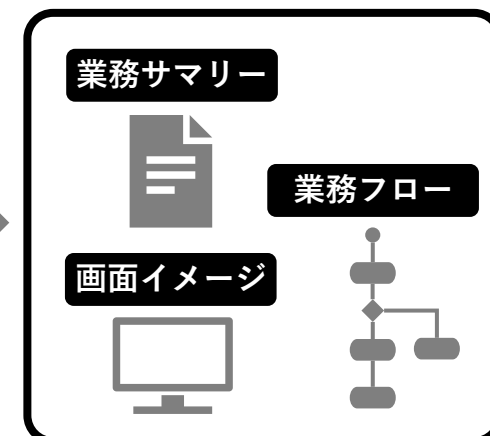
「一気通貫」による課題



AIが出力する情報



お客様が見たい情報



何を見ればよいか分からない

技術的には、完璧だった。それでも——

すべて揃っていた

クラス名・メソッド名・ソース根拠
——自信を持って、お渡しした



お客様

「どこを見れば
いいのか、わからない」

なぜAIの成果物は「伝わらない」のか？

関心領域のミスマッチ

コードやDB設計の整合性そのものに、お客様の関心はない

お客様が見たいのは
「自社の業務」

判断基準の不在

クラス名・メソッド名
(How) を判断したいわけではない

お客様が判断したいのは
「業務が成立」

変数名の壁

sys_val_dt のような変数名を、
読みたいわけではない

お客様が読みたいのは
「自分たちの言葉」

正確さの問題ではなく、「読む人」が違っていた。

問題は、正確さではなかった

技術的精度
正確だった

クラス名・記述内容・ソース根拠

≠

お客様の理解

伝わらなかった

「どこを確認すればよいか、わからない」

正確さは、伝わることを保証しない。

同じ仕様書に、まったく異なる読者がいた

SE（設計者）
実装の正確さを見る

クラス名・メソッド名
ソース根拠・整合性

≠

お客様
業務が正しく反映されているか

業務サマリー・業務フロー
表示項目（変数名なし）

同じ文書でも、読者が変われば「必要な情報」は違う。

読み手に合わせて、書き分ける

人

誰に向けて書くかを決める

+

AI

各読者版を書き分ける

人手なら冗長な作業も、AIなら一度で済む。

3つの違和感

1

伝わる

2

判断

3

やりがい

[判断]
部下

AIの「できない」は、本当に終わりなのか？

AIの「×（できない）」で、考えるのをやめていた

AIが「×（不可能）」と言うと
そこで、手が止まる



それ以上
考えなくなる

例：ソースからのリバース。AIが「無理」と判定した瞬間、探索が終わる——工夫の余地ごと消えた。

深く考える——その習慣が、抜けていた。

AIの「×（できない）」を起点に変える3つの視点



前提を疑う

「本当に無理？」 制約・前提・プロンプトを変えてみる



別の切り口を 探る

一度の回答で満足しない。別の角度から問い直す



探索と判断は人 が担う

最終決定のハンドルは、AIに明け渡さない

AIの「できない」を突破する「人の経験・知見」の力



AIが「できない・難しい」と言った

「ソースのみでは〈業務分析〉〈ユーザ設計〉の作成が難しそうです」



人の経験・知見が、突破する

業務・ユーザ視点の情報を、人が補って問い直す



「あのSkillsでも、期待するアウトプットが出てなかった？」

「何が足りないか」を、人が問いを立てて補う



「同じプロンプトでも、出力内容にばらつきがある」

個別の観点・前提を、人が加えて絞り込む

「できない」の多くは、人の問いと情報で、突破できた。

ゴールとその背景は、人が握る

AI

結果は出す

でも「なぜ・どうやって」は見えない

だから

人

ゴールと背景を握る

なぜこの仕様か・修正か——次の指示を出す

結果について考えることは、手放してはいけない。

3つの違和感

1

伝わる

2

判断

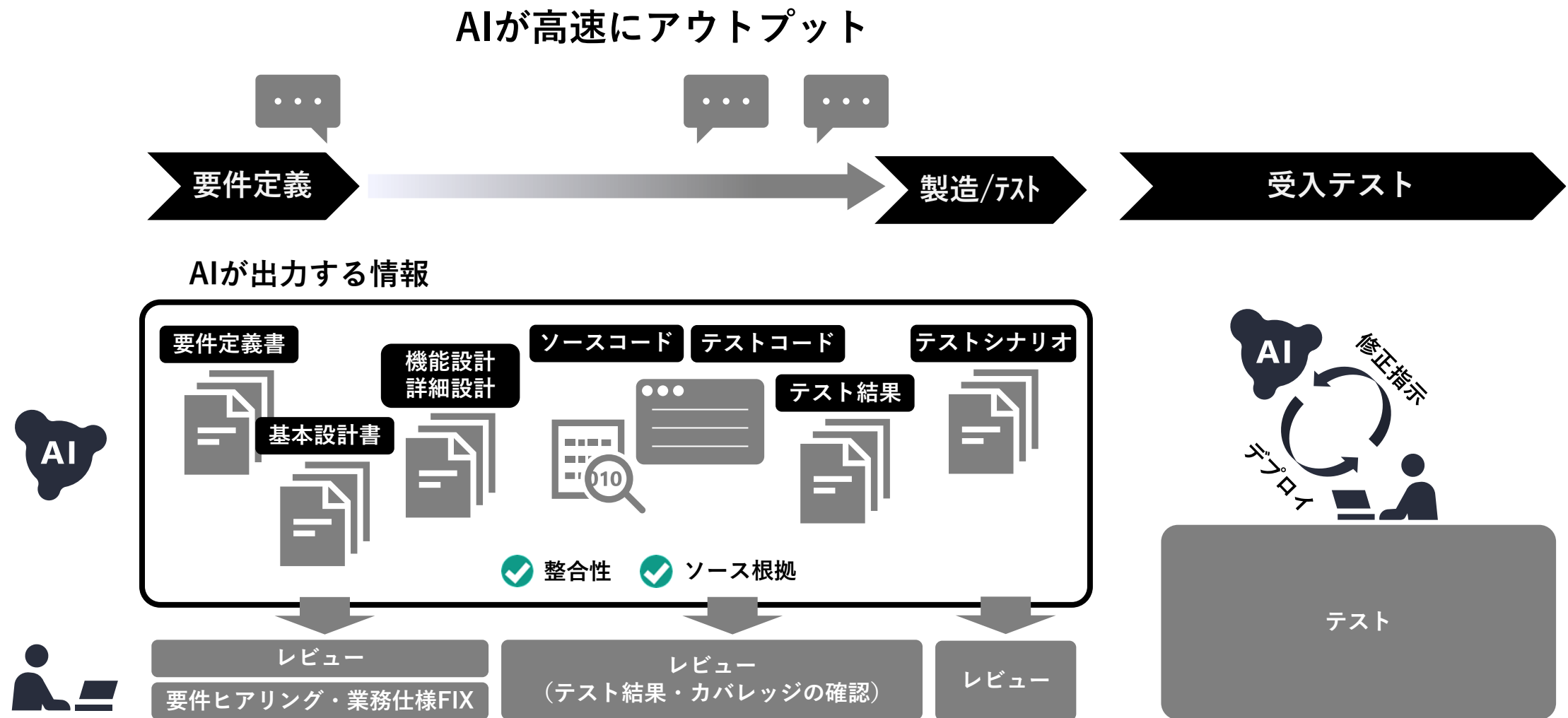
3

やりがい

[やりがい]
同期

効率化したのに、なぜ「心が動かない」のか？

どこをAIに任せ、どこに人を充てるか



効率化の陰で、心が動かなくなっていた

「AIのアウトプットを“確認するだけ”の作業って、
誰が楽しいん？」

—— 同期（AI駆動開発の検討会で）

効率を追って、「考える余白」まで奪っていた。

確認だけが残り、心が動かなくなっていた。

人に「判断」を残す、役割設計

これまでの役割分担
AI = 作業 / 人 = 確認



これからの役割設計
人に、判断・探索・意思決定
を残す

判断する仕事に戻れば、やりがいも戻る。

「探索する喜び」を設計する

問いは、人が投げる

AIに骨組みを作らせ人が直す、
を逆転。人が「問い」を立て、
AIが支える。

探索を、楽しむ

効率第一から脱却。試行錯誤
やアイデアを練る過程に、人
を置く。創造性を取り戻す。

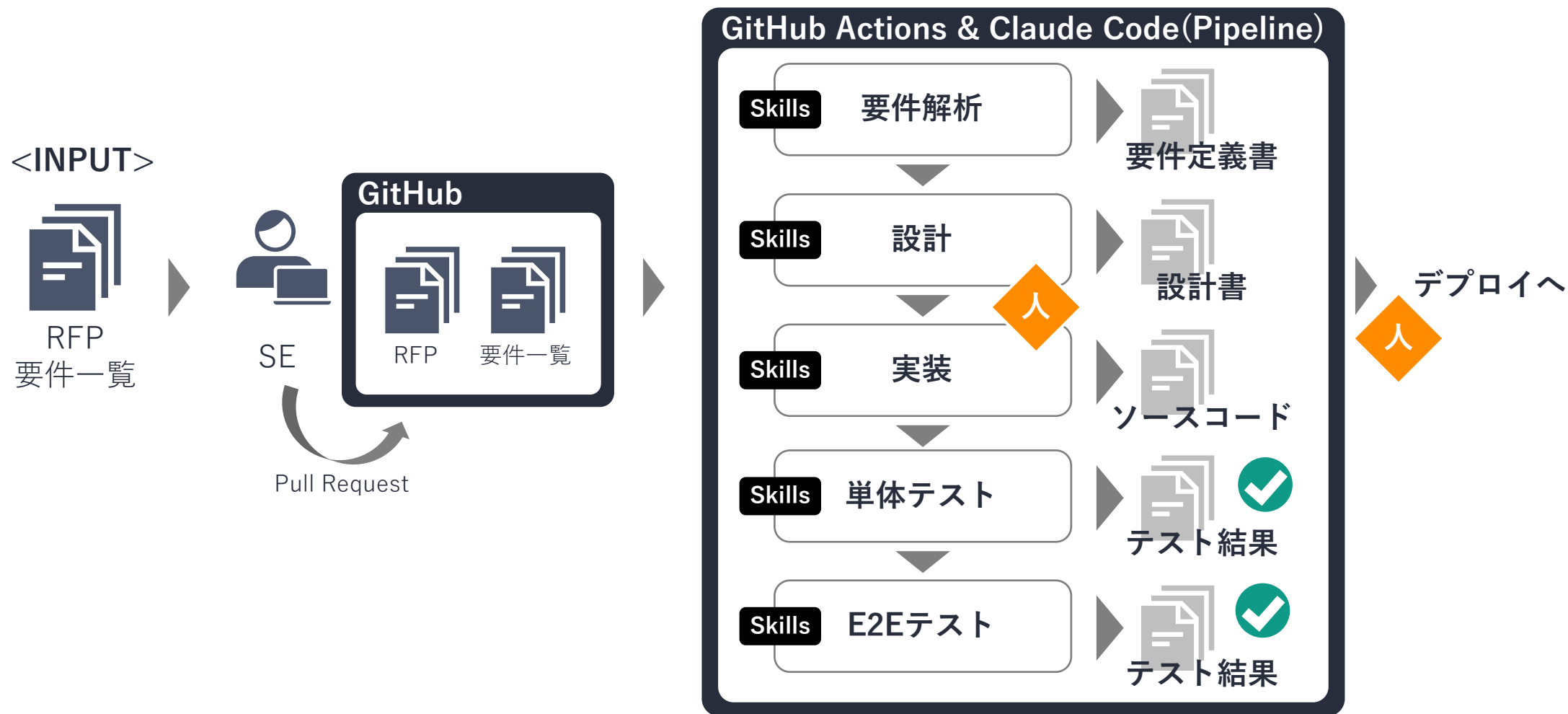
責任が、手応えになる

判断には責任が伴う。それが
自己効力感と、やりがいを生
む。

効率のための分担から、「やりがい」のための設計へ。

3つの違和感は、一つの問いだった

その問いとは——「自分は、深く考えるのをやめていないか」
だから、全自動の流れに人の「ゲート」を設けた



読者設計を直すだけでは、足りなかった

3つの違和感の正体は、“確認だけ”になっていた、私自身だった

お客様	→	伝わらなくなった
部下	→	探索しなくなった
同期	→	作業になっていった

——三つは、別々の問題じゃなかった。全部、“深く考える”のをやめた、一つの自分だった。

[伝わる]

正確に書くことと、伝わるように書くことは、別の仕事だった

正確さ

機械のための品質

≠

伝わること

人のための品質

お客様が確かめたいのは、正誤ではなく「伝わったか」

〔判断〕
AIの「×」は、答えではなく、問いの起点だった

AIの「×」
判定

≠

人が出す
答え

結果の「なぜ」まで考えることこそ、人に残る仕事。

[やりがい]
心が動く仕事が、戻ってきた

確認だけの連続
心は動かない

≠

探索する仕事
手応えが戻る

「考える・探す」が戻れば、やりがいも戻る。

あの問いへの答え： ゲートの正体は、「深く考える場」だった

伝わるか・できないか・任せられているか——3つとも、突き詰めれば「深く考える」ことだった。

[伝わる]

伝わるように書けているか

[判断]

結果の「なぜ」まで考えているか

[やりがい]

人に判断を残せているか

AIが担う「正確さ」 + 人が担う「深く考えること」 = AI時代の分業

AIの時代の、エンジニアの仕事

コードを書く「作業者」から、技術・データを“価値”へ翻訳し、意思決定する「アーキテクト」へ。
AI活用を前提に、人間ならではの“深い検証力”と“問題設定力”が問われる。

技術実装が自動化されるほど、際立つ仕事がある

AIが担う領域

実装・設計・検証

仕様書の生成
コードの品質
タスクの分解

人と人のある領域

何を作るかの合意

お客様の真意を引き出す対話
成果物にお客様が納得する瞬間
組織間で優先順位を決めきる調整

自動化が進むほど、人と人の間の仕事がかっきり見えてくる

ずっとそこにあった仕事が、ようやく見えてきた

「AIに仕事が奪われる」のではない。

伝える・結果を考える——その先に、やりがいがある。

これらは、ずっとそこにあった仕事——技術の作業に隠れて、見えなかっただけ
そして、やりがいも、そこに戻る。

あなたの仕事に、
人と人の仕事はいくつありますか？

伝える

読者に合わせて書く
「伝わったか」を確認する

結果を考える

AIの答えを疑う
なぜ・ゴールまで問い直す

やりがい

何を作るかを決める
判断を取り戻す

あなたの仕事には、いくつありますか？ _____ 個

手放してはいけない、3つの仕事

相手に伝える

正確さではなく、
「伝わったか」を問う

結果について考える

答えを疑い、
ゴールと背景の「なぜ」を考える

やりがいのある仕事

判断が戻れば、
働く手応えも戻る

AIが賢くなるほど、この3つを手放さないことが、人の仕事になる。

NEC120年以上の歴史は、挑戦の歴史。

「より良いものを追求する」という思いで、

テクノロジーにできることをひとつずつ増やしてきた。

国家の安全を守ること。社会のすみずみまで便利にすること。

一人ひとりの今日を笑顔にすること。未来の当たり前を創ること。

それでもまだ、ゴールじゃない。

社会の課題が、どんどん複雑になっている今こそ、

ダイナミックなイノベーションが必要だ。

個性豊かな仲間が必要だ。

だから、私たちは共創する。多様性を掛け合わせて、

テクノロジーの可能性を広げ、

社会の可能性も、あなた自身の可能性も広げていく。

この世界を、もっと頼もしく、誰もが輝く場所にするために。

この先の未来が、もっと楽しみになるように。進もう。

あなたと未来、 高鳴るほうへ。



NEC

\Orchestrating a brighter world