AISIN

次世代AI駆動型 COBOL → Java 標準化移行ソリューション

100%監査可能な標準化プロセスで、コンプライアンスリスクをゼロ化する次世代アプローチ

> 属人的で追跡不能なレガシー移行を、完全な監査証跡を持つ標準化プロセスへ。 AI駆動のガバナンス基盤で、ベンダーロックインなしの次世代Java移行を実現します。

2026年3月12日　アイシン株式会社

レガシー移行の4大リスク（現状の危機）

AISIN

現在のマイグレーションプロジェクトにおいて、従来のCOBOL移行手法は以下の致命的な課題を抱えています。これらはプロジェクトの遅延、品質低下、そして将来的なリスクの温床となります。

属人化と暗黙知のブラックボックス化

移行プロセスが熟練技術者の「経験と勘」に依存しており、ノウハウが体系化されていません。ベテラン退職による知識喪失が、企業にとって回復困難なリスクとなります。

膨大なドキュメント保守コスト

設計書やテスト仕様書の作成が手作業に依存しており、コードとの乖離が常態化。修正のたびに発生するドキュメント更新コストが、予算を圧迫し続けています。

コード変換品質のばらつき

従来の変換ツールは出力品質が一定せず、大量の「オフライン手修正」が発生。エンジニアのスキル差により品質が均一化されず、潜在バグが残存する原因となります。

最重要課題

監査証跡（エビデンス）の欠如

承認・指摘・修正のプロセスが可視化されていません。金融機関等で求められる厳格なトレーサビリティ要件を満たせず、コンプライアンス違反のリスクに直結する最大の課題です。

従来手法の致命的弱点 vs 本ソリューションの回答

AISIN

BEFORE: 従来手法の課題	AFTER: ソリューションの価値
独自基盤への依存（ロックイン） 生成コードが特定のベンダー基盤に依存。 **保守コスト高止まり、将来移行困難。**	**主要フレームワークに対応** SprintBoot、Sparkなどあらゆるフレームワークへ柔軟に対応し、 **100%の移植性とIT主権を回復**
手続き型ロジックのまま COBOL構文の直訳でスパゲッティ化。 **若手エンジニアが保守不能。**	**階層構造へ自動再構成** Controller/Service/Repositoryへ変換。 **現代的な3層アーキテクチャを実現。**
分散処理への自動変換不可 バッチ処理の並列化ができず、 **ハード増強でも処理時間が短縮されない。**	**分散バッチ最適化** データ依存を解析し並列コードを生成。 **リソース追加で線形スケールを実現。**
手修正により証跡喪失 手作業修正が記録されずブラックボックス化。 **コンプライアンス違反の重大リスク。**	**SOP内完結の監査ワークフロー** 「誰が・いつ・何を」修正したか、 **100%追跡可能な証跡を自動生成。**

なぜ今、解決すべきか？（ビジネスインパクト）

2025年の崖による技術的負債の顕在化、コスト圧力、そして厳格化するコンプライアンス要求。これらが同時に押し寄せる今、レガシー脱却はIT部門の課題を超え、経営の最優先アジェンダとなっています。

2025年の崖：技術空洞化

ベテラン技術者の大量退職により、COBOL資産を保守できる人材が枯渇。ブラックボックス化したシステムの維持が物理的に困難となる「タイムリミット」が迫っており、事業継続性への直接的な脅威となります。

メインフレーム保守費の高騰

ハードウェア保守期限の終了に伴う移行コストの増大や、ライセンス料の高騰がIT予算を圧迫。維持管理費の肥大化が、攻めのDX投資へ予算をシフトできない構造的な足かせとなっています。

規制強化：内部統制・監査要求

金融庁ガイドラインや内部統制報告制度など、システムに対する説明責任と監査要求が高度化。従来の「結果オーライ」なブラックボックス運用は許容されず、完全なトレーサビリティが必須条件です。

DX加速の前提条件

データ活用やクラウドネイティブなサービス連携において、硬直的なレガシーシステムがボトルネックに。市場競争力を維持・強化するためには、柔軟でオープンなデジタル基盤への刷新が不可欠です。

9つの標準化移行パイプライン（SOP）

AISIN

分析・仕様化

PHASE 1

STEP 01
逆向仕様書生成
Spec v1

STEP 02
COBOLテスト仕様
Test Cases

STEP 03
移行分析
Analysis

設計・自動変換・テスト

PHASE 2

STEP 04
Java設計書
Class Design

STEP 05
実施計画書
Plan

STEP 06
Java変換
Source Code

STEP 07
テスト検証
Report

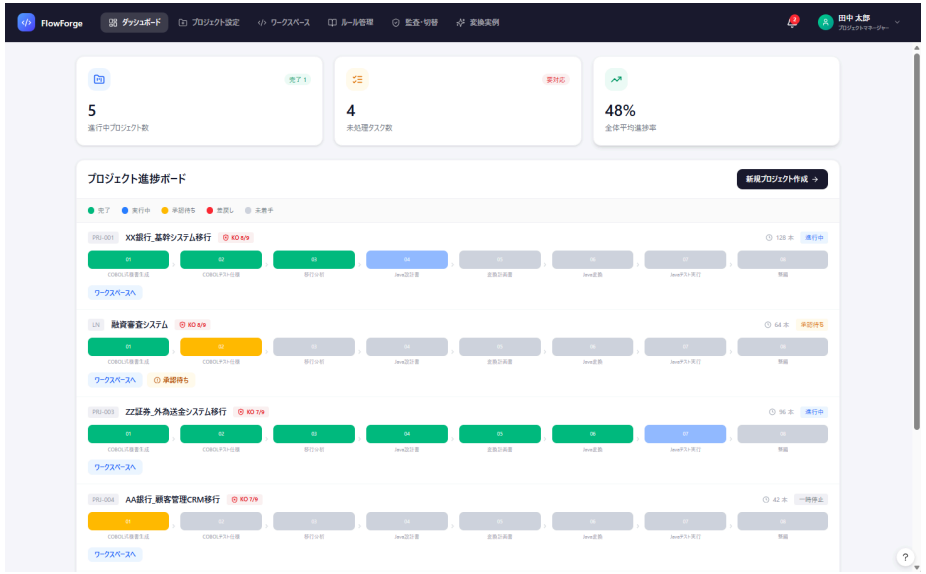
検証・納品

PHASE 3

STEP 08
整編・パッケージ
Delivery

STEP 09
最終切替
Approval

Live Dashboard View



AI駆動型ガバナンス

標準化
SOPによる
抜け漏れ防止

承認ゲート
各Step完了時の
システム承認

可視化
遅延リスクの
即時検知

ソリューション全体像：SOP駆動 + Chatbot型Agent UX

AISIN

AI駆動型移行エンジン

LAYER 3 **Chatbot型 Agent UX**

対話・実行・レビューが統合された一気通貫UI。指示から承認までシームレスな体験を提供。

Experience

LAYER 2 **SKILL.md ガバナンス統制**

変換ルールをコードから分離し資産化。独自のLint機能により、100%の規約遵守を自動的に実現。

Governance

LAYER 1 **SOP駆動の標準化プロセス**

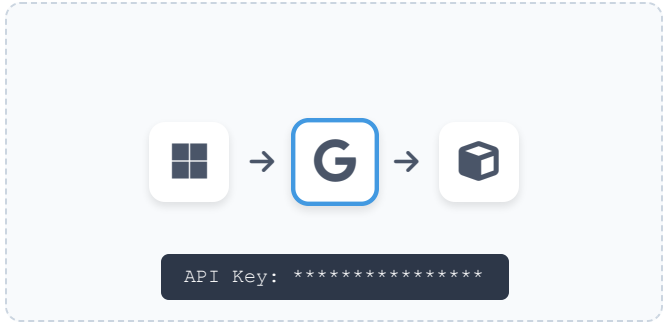
01〜09の全工程をパイプライン化。属人性を排除し、誰が実行しても同じ品質を保証する基盤層。

Process

LLM柔軟性とベンダーロックイン排除

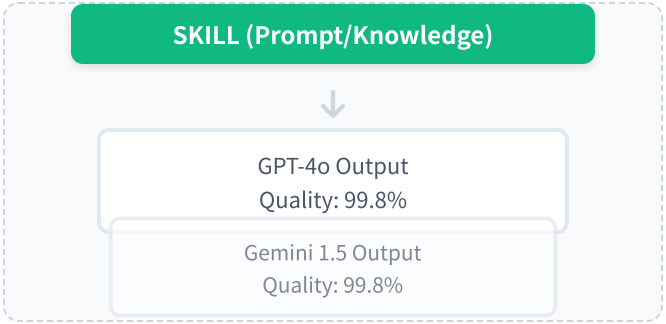
AISIN

APIキー差し替えによるLLM即座切替



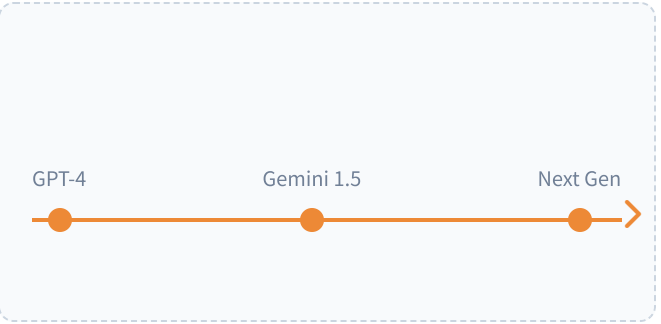
OpenAI、Google Gemini、Azure等の主要LLMプロバイダーに対応。**APIキーを差し替えるだけ**で、モデルを即座に切り替え可能。特定ベンダーへの依存リスクをゼロ化します。

SKILL活用による一貫品質保証



モデルが変更されても、最適化された**「SKILL（ナレッジ資産）」**が緩衝材となり、出力品質を均一に保ちます。エンジニアはモデルの違いを意識せず開発に専念できます。

最新LLM技術への継続的アクセス



AI技術の進化に合わせて、常に**最新・最強のモデル**を選択・利用可能。システムの陳腐化を防ぎ、将来にわたって技術的優位性を維持し続けます。

4つの絶対的競争優位性

AISIN

独自基盤からの脱却

Spring Boot、Spring Batch、Sparkをはじめとする、すべてのフレームワークに対応可能。特定技術に依存せず、ベンダーロックインを排除します。

真のJavaへ自動再構成

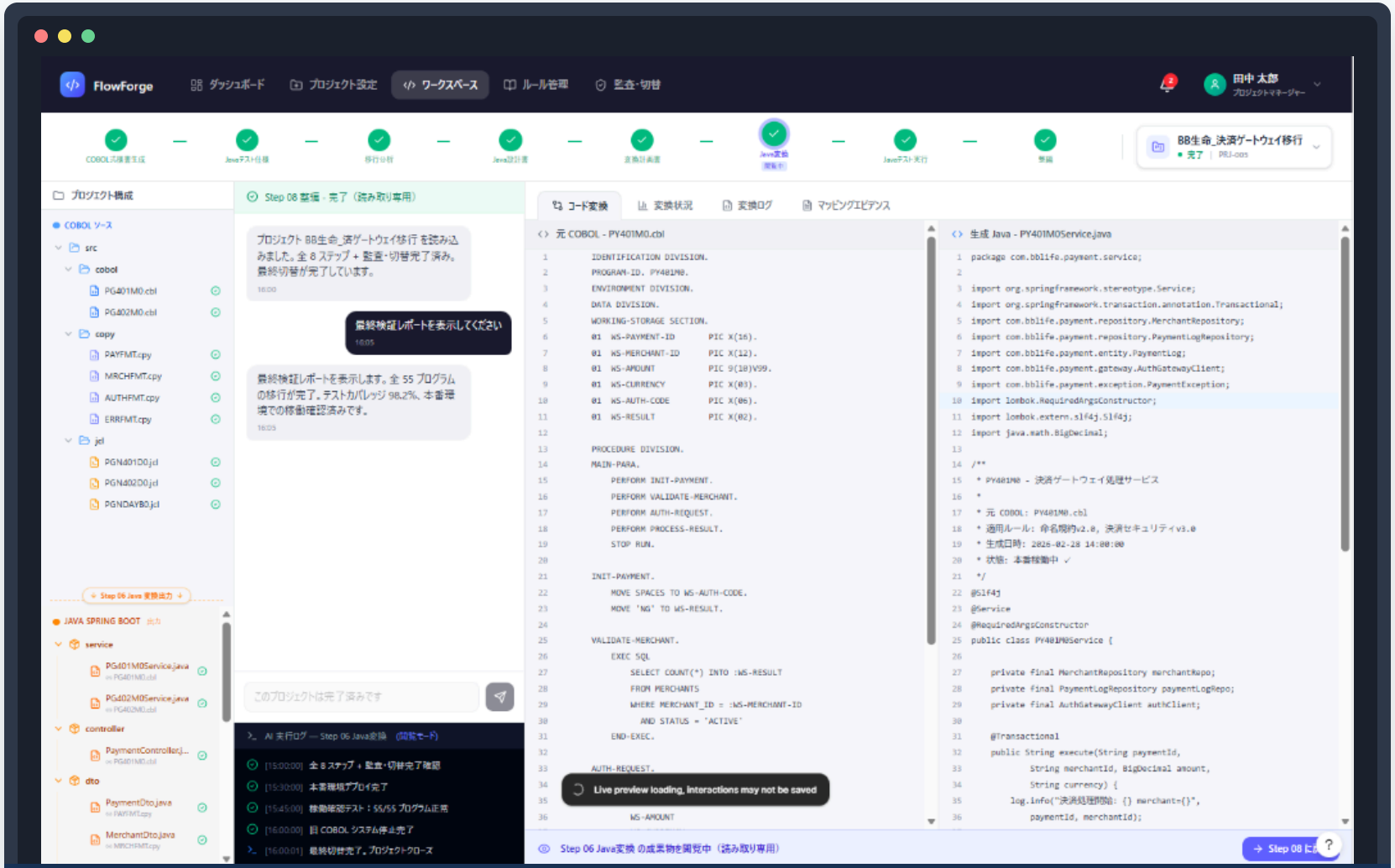
COBOL特有の手続き型ロジックを、若手エンジニアも保守可能なController/Service階層へ自動変換。

分散処理の自動最適化

データ依存関係を解析し、スケールアウト可能なコードを生成。ハードウェア追加で線形に性能向上。

完全な監査証跡

すべての変換と修正履歴をSOPシステム内に記録。属人的手修正を排除し、100%のトレーサビリティを実現。



左側の手続き型COBOLが、右側のクリーンなSpring Bootコードへ自動変換されています

SKILL.md ガバナンス：ルール資産化とトレーサビリティ

ルールとコードの分離

変換ロジックをブラックボックス化せず、外出しの「SKILL.md」として資産管理。プロジェクト固有の規約変更にも柔軟に対応可能です。

Lint/Pre-check 自動検証

生成されたコードが規約に準拠しているかをAIが自動チェック。人間が見落としがちな命名規則や例外処理の統一を強制します。

生成物とルールの紐付け

「どのバージョンのルール」で「どのソースコード」が生成されたかを厳密に記録。将来の監査時にも、生成根拠を100%説明可能です。

移行プロセスの実例 ：移行前COBOLソースコード

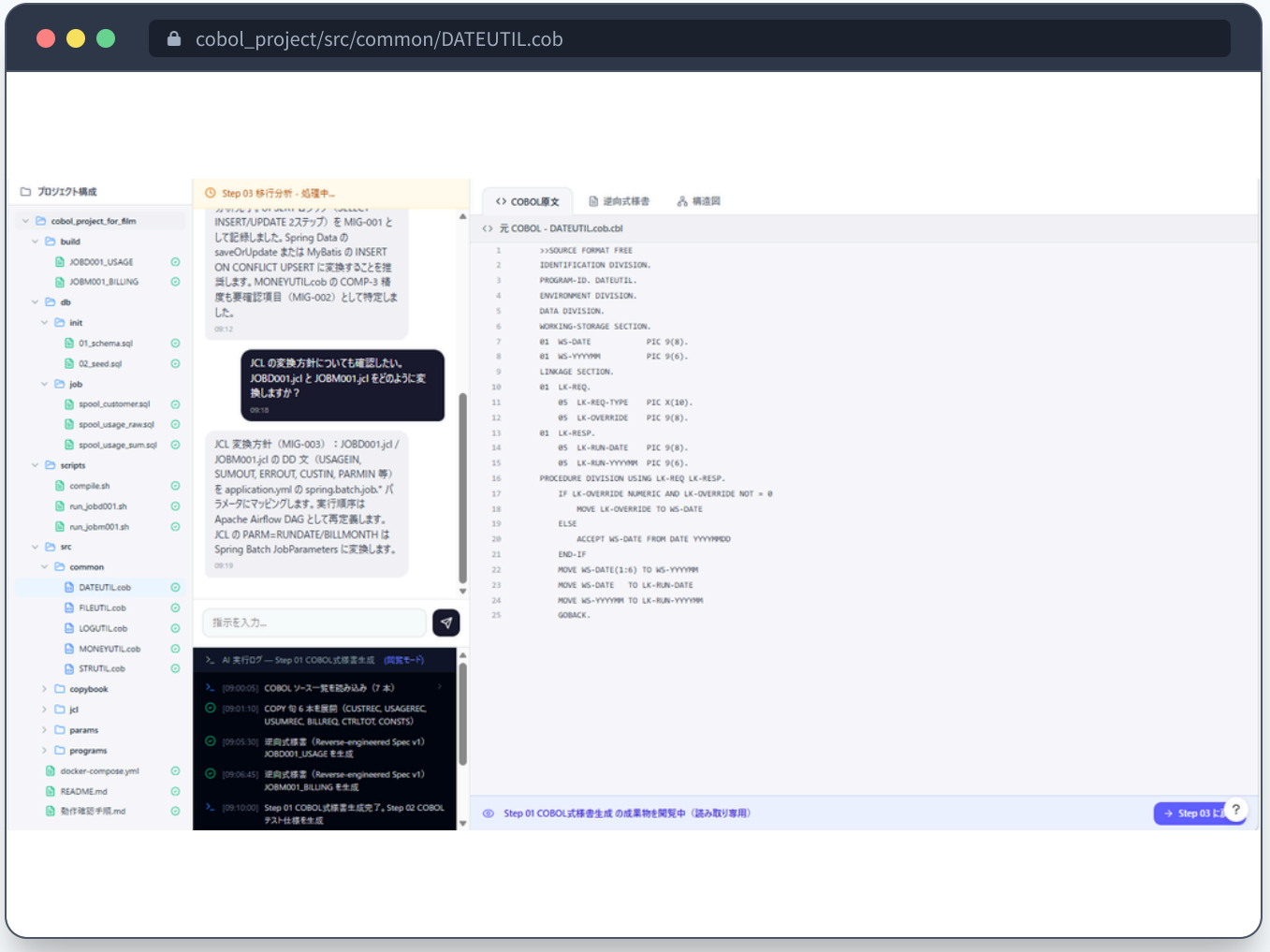
AISIN

1

既存COBOL資産の解析

現状分析・可視化フェーズ

既存のCOBOL資産（DATEUTIL.cob等）の実際の画面です。 IDENTIFICATION 、 ENVIRONMENT 、 DATA 、 PROCEDURE DIVISIONを含む標準的なCOBOL構造をAIが読み込みます。



- 複雑な日付処理ロジックを解析
- データ構造の依存関係を特定
- ブラックボックス仕様を可視化

この複雑なレガシーコードを解析し、次のステップで設計書を自動生成します。

2

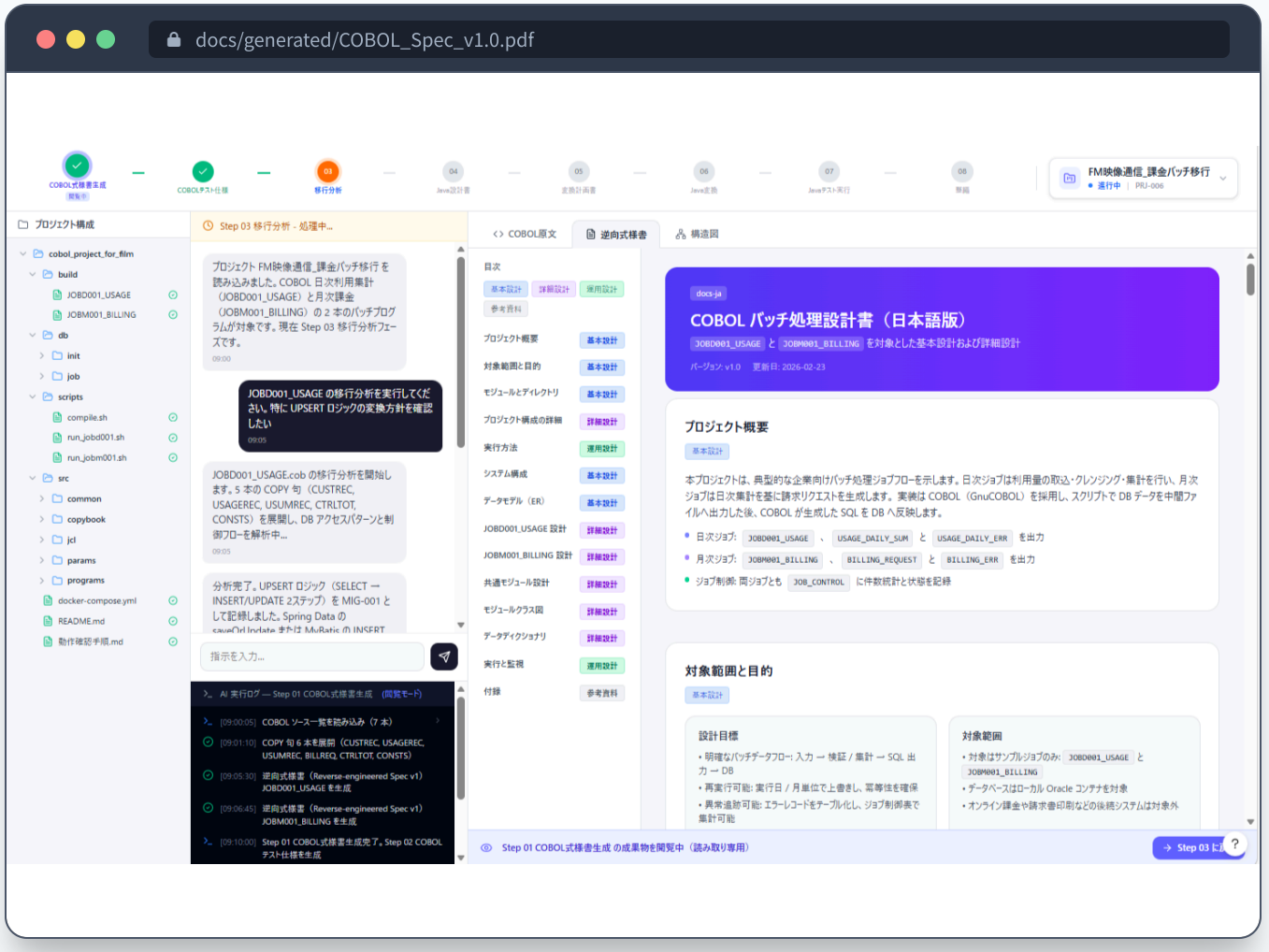
日本語設計書の自動生成

ドキュメント化フェーズ

解析結果に基づき、詳細な日本語設計書を自動生成します。 プロジェクト概要、処理フロー、データ定義、テーブル・カラム情報を人間が読める形式でドキュメント化します。

- プロジェクト概要と処理フローの可視化
- データ定義書（DD）の自動生成
- CRUD図・フローチャートの出力

手作業ゼロで標準化された仕様書を作成し、属人化を排除します。

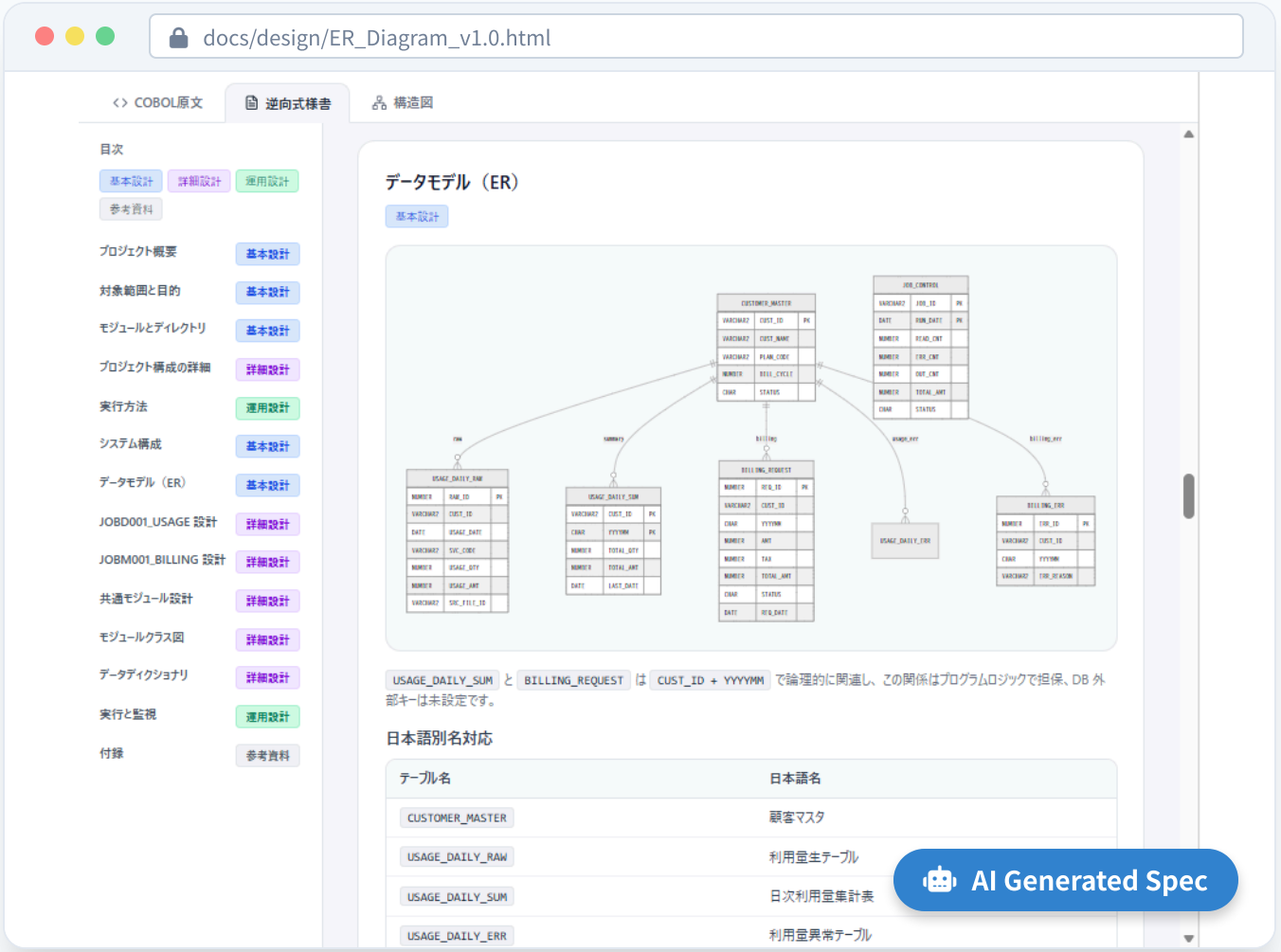


データモデルの完全可視化

自動生成されたER図と定義書

COBOLソースコードからデータベース構造を解析し、テーブル間のリレーションシップ（ER図）を自動描画します。 さらに、物理名と論理名（日本語名）の対照表も自動生成され、システムのデータ構造が一目瞭然になります。

- テーブル関連図（ER）の自動描画
- CUSTOMER_MASTER等の依存関係特定
- 日本語別名対応表の自動生成



複雑なバッチ処理のデータフローを視覚的に理解しやすくし、設計レビューの効率を劇的に向上させます。

設計書特写：ジョブ設計書詳細

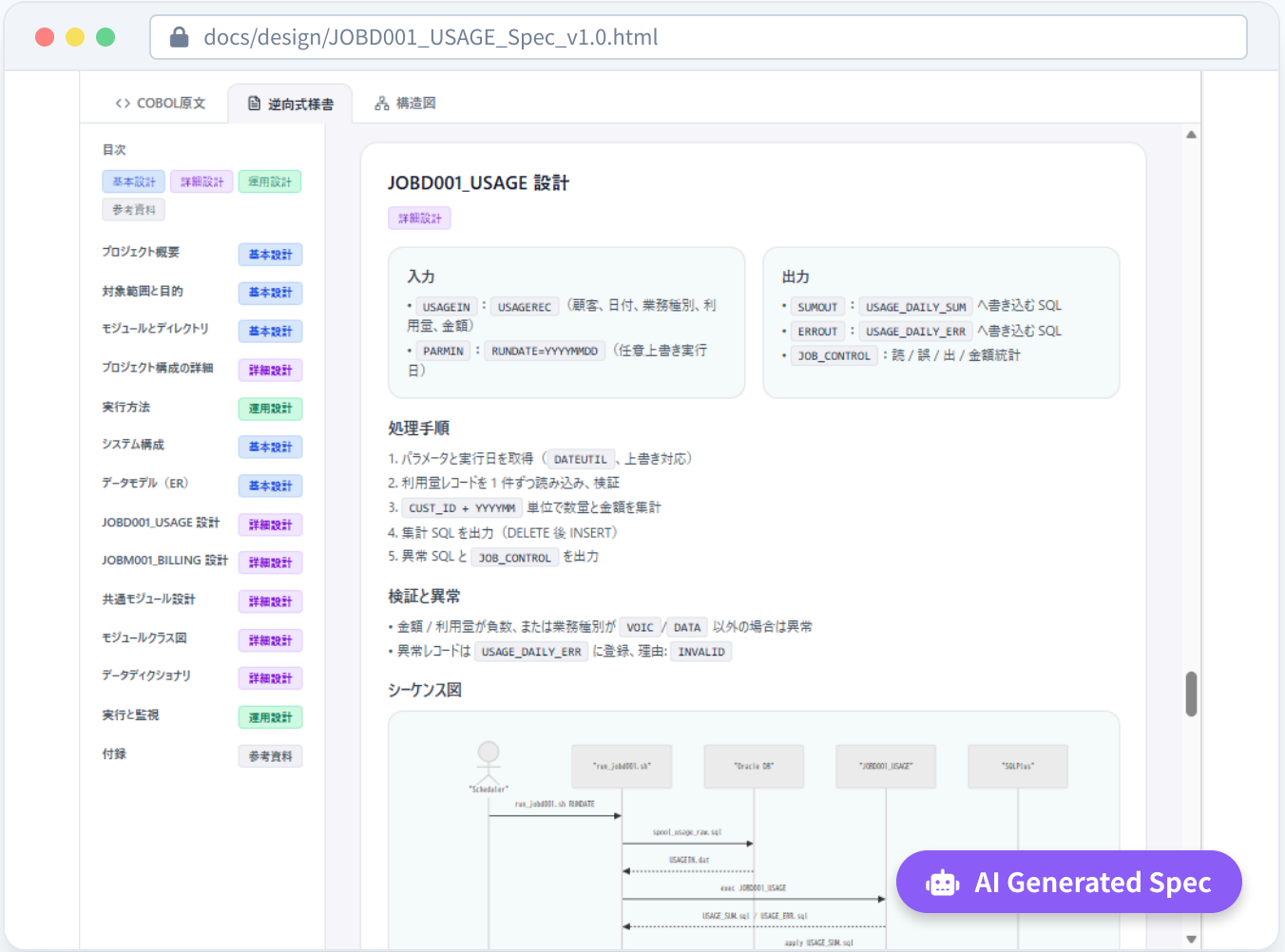
AISIN

バッチジョブの完全定義

JOBD001_USAGE 詳細設計

ソースコードから処理ロジックを読み解き、入力・出力、具体的な処理手順、例外処理を自然言語で再構築します。 シーケンス図も自動生成され、処理の流れを直感的に把握できます。

- 入出力定義（USAGEIN, SUMOUT等）
- 処理手順と検証ロジックの明文化
- シーケンス図によるフロー可視化



ブラックボックス化していた処理ロジックが「誰でも読める仕様書」として蘇り、保守性が飛躍的に向上します。

移行プロセスの実例 ：変換後Javaソースコード

AISIN

3

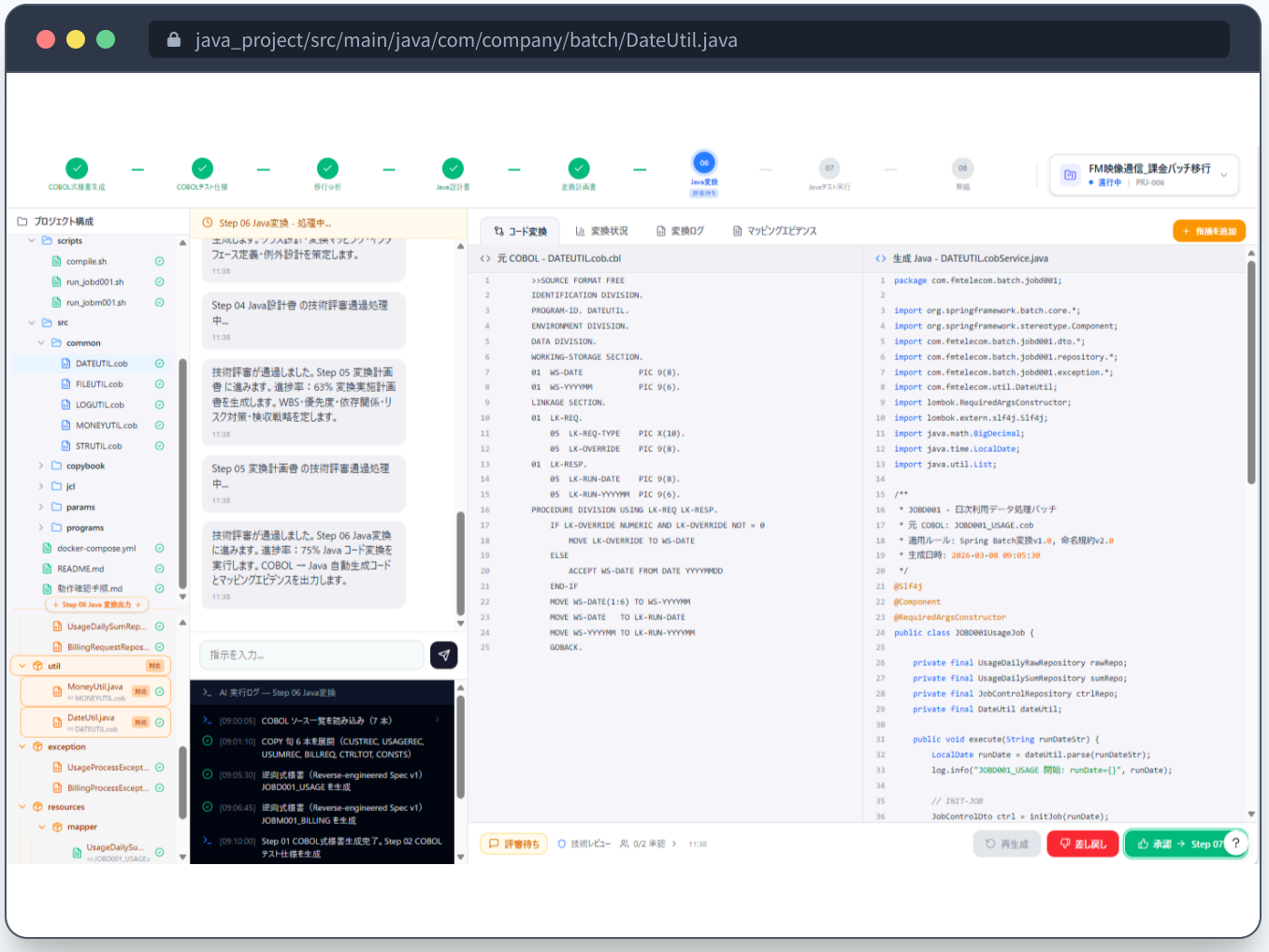
モダンJavaコード生成

高品質コード生成フェーズ

設計書に基づき、Spring Boot、Spring Batch、Sparkなど、あらゆるフレームワークに対応可能なJavaコードへ変換。 可読性が高く、保守容易なオブジェクト指向コードを出力します。

- あらゆるフレームワークに 完全対応
- 可読性の高いオブジェクト指向設計
- 型安全性と例外処理の適正化

独自の独自ライブラリを含まない純粋なJavaコードであり、ベンダーロックインを排除します。



設計書カスタマイズとマルチフォーマット対応

AISIN

SKILLへのサンプル学習による統一

既存の設計書を「正解データ」としてAIに学習

SKILLリポジトリに貴社の既存設計書（サンプル）をアップロードするだけで、AIがその記述ルールやフォーマットを学習（Few-Shot Learning）。**独自の命名規則やドキュメント様式に完全準拠**した設計書を自動生成し、手作業による修正コストをゼロにします。

用途に合わせたマルチフォーマット出力

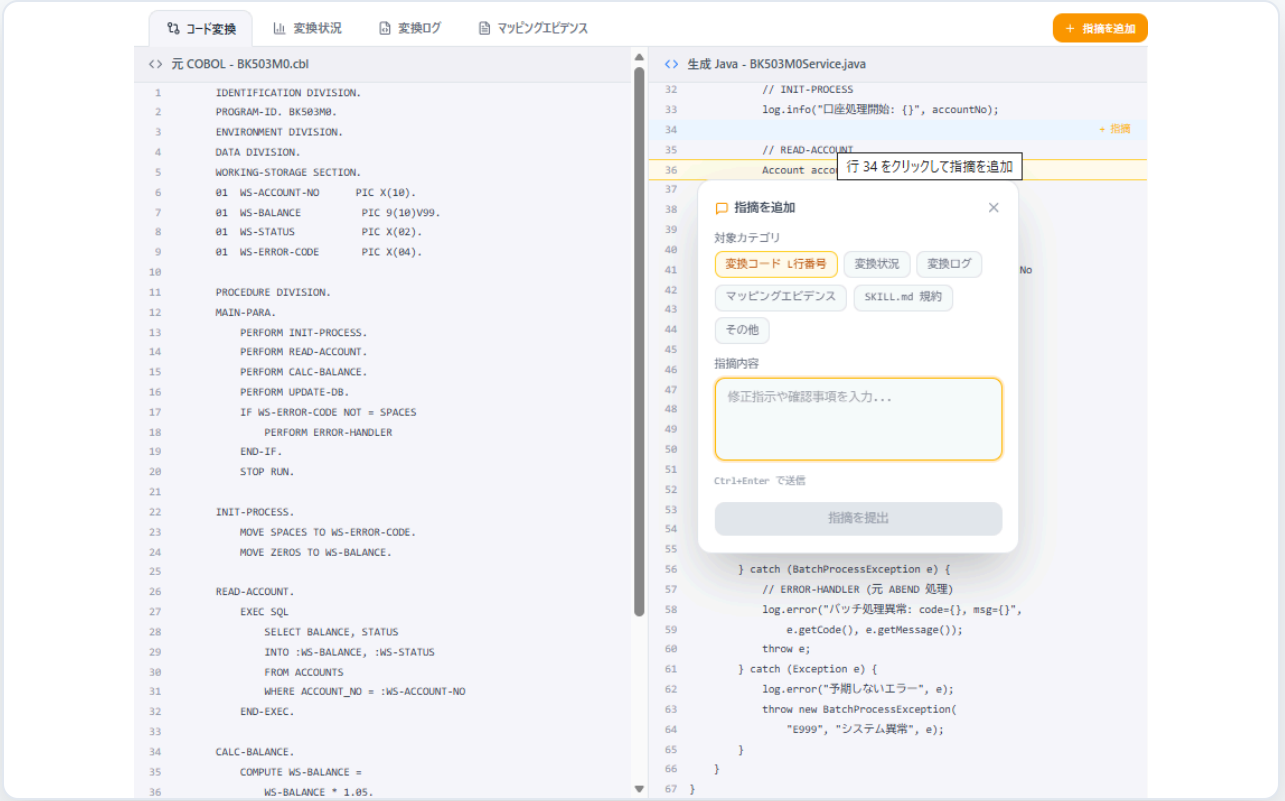
ワンクリックで最適な形式へ変換・出力

生成されたドキュメントは、用途に応じて**PDF、Markdown、HTML**など複数の形式で即座に出力可能。経営層への報告用資料から、開発チームがGitでバージョン管理するための技術資料まで、あらゆるシーンに対応します。

レビュー＆再生成：Human-in-the-loop品質保証

完全なトレーサビリティ

誰がいつ、なぜ修正したか。
すべての履歴を改ざん不可能な状態で保存。



ステータス管理ワークフロー

- 承認待ち (Waiting)
- 差戻し・修正中 (Remanded)
- レビュー完了 (Reviewed)
- 最終承認 (Final Approved)

全社規模のコードスタイル統一と柔軟な特例対応

AISIN

統合SKILLリポジトリによる派生管理

全社共通ルールとプロジェクト固有ルールの階層管理

共通SKILLを継承しつつ、プロジェクトごとの差異を「派生（Variant）」として管理。複数ベンダーが参画しても、システム全体で統一されたコード品質を強制的に担保します。

手作業ゼロのチャット型例外対応

ルール化できない特殊処理も自然言語で即時修正

ブラックボックスな手修正（ハンドコーディング）を完全に排除。会話形式の指示でAIがコードを修正するため、意図が明確な履歴として残ります。

エンタープライズ水準のLLMセキュリティと「揺らぎのない」品質

AISIN

CIOの2大懸念を解消

セキュリティ境界モデル

法人契約LLM (BYO-LLM) による完全遮断

機密コードが外部の学習データとして再利用されるリスクを、システム的に完全に遮断。メガクラウドのエンタープライズ基準で保護。

品質フィルタリング・ファンネル

「出力のゆらぎ」を排除する4層構造

生成AIの「生」出力 (ゆらぎ有り)

1. SKILL制約 (Constraint) — 厳格なルール定義 → ✕ 自由解釈
2. Demo参照 (Few-Shot) — 手本コード参照 → ✕ スタイル違反
3. Self-Correction — 自己反省ループ → ✕ 論理エラー
4. 全自動テスト — 網羅的検証 → ✕ バグ・退行

品質保証されたコード (再現性100%)

4層のフィルターを通すことで、「幻覚」や「不規則な出力」を構造的に排除。人間が書くコード以上の均一性を担保。

エンタープライズ要件への完全対応

AISIN

CI/CD エコシステム
開発インフラとシームレスに連携し、
自動化パイプラインを確立

LAYER 4

完全な監査証跡
「いつ・誰が・何を」変更したか
100%追跡可能 (WORM管理)

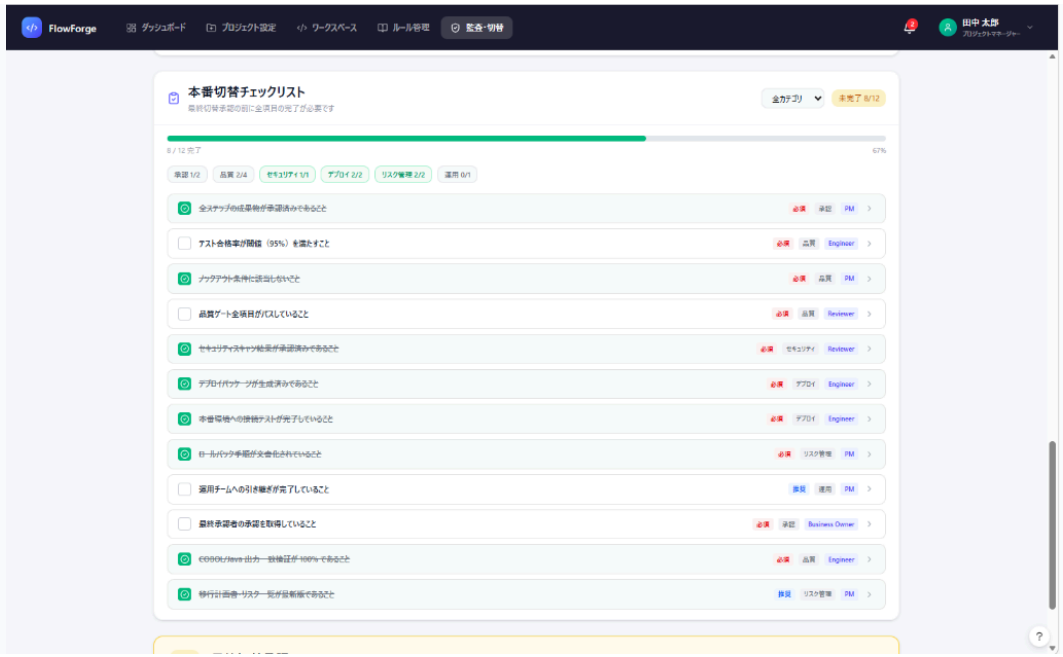
LAYER 3 (CORE)

セキュリティ保護
金融機関レベルのデータ保護基準に準拠 (TLS 1.3 / SSO)

LAYER 2

RBAC 権限管理 (基盤)
職務分掌に基づいた厳格なアクセス制御基盤

LAYER 1



エンタープライズ対応の3つの柱

コンプライアンス準拠
金融機関レベルの監査証跡要件を満たし、規制対応コストを大幅に削減します。

セキュアバイデザイン
多層防御アーキテクチャにより、データ漏洩リスクと不正アクセスを未然に防ぎます。

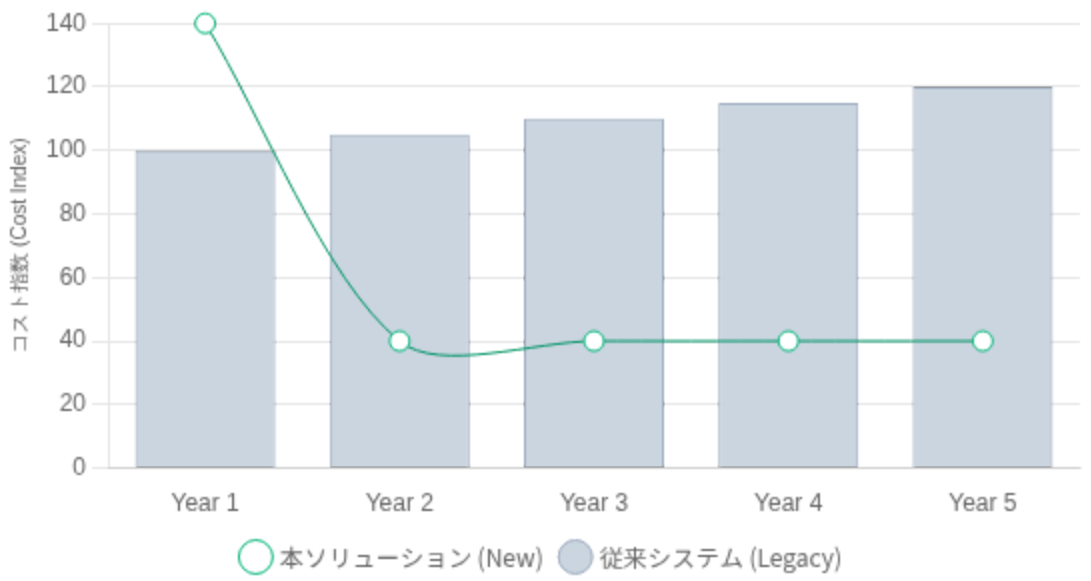
開発生産性と統制
CI/CDパイプラインとの完全統合により、スピードとガバナンスの両立を実現します。

導入効果の枠組み：TCO最適化とリスク低減

AISIN

TCO（総保有コスト）の大幅削減

5〜10年の長期視点で見た劇的なコスト圧縮効果



保守・運用コスト (5年累計)

約 60% 削減 ↓ Down

※レガシーハードウェア撤廃による効果

ライセンス費用

ゼロ (Zero) ✓ Free

※OSS採用による効果

属人化リスクの恒久的解消

「特定の技術者しか触れない」状態から脱却。標準Java技術への移行で、市場の豊富な若手人材による保守が可能になります。

ベンダーロックインの完全排除

独自基盤やランタイムに依存しない純粋なOSS技術を採用。システムの主導権（ITガバナンス）を取り戻します。

監査適合・コンプライアンス強化

変更を100%追跡可能な証跡を自動生成。金融機関レベルの厳格な監査要求にも、追加工数なしで即座に対応可能です。

定量的評価フレームワーク（POC予定）

3-5倍	100%	0円
移行スピード	監査カバレッジ	基盤ライセンス費

※PoCにて実測値を算出後に正式試算

AISIN

ご清聴ありがとうございました

次世代AI駆動型：COBOL → Java 標準化移行ソリューション