IPA

2025 年度 未踏 IT 人材発掘・育成事業 採択案件評価書

1. 担当 PM

曾川 景介（newmo 株式会社 CTO）

2. クリエータ氏名

戸田 朋花（株式会社サイバーエージェント／株式会社 AbemaTV）

3. 委託金支払額

1,822,000 円

4. テーマ名

ソフトウェア設計のコード化と数学的検証を実現する Go フレームワークの開発

5. 関連 Web サイト

- goat：https://github.com/goatx/goat
- goat-cli：https://github.com/goatx/goat-cli

6. テーマ概要

本プロジェクトでは、Design as Code を実務で実現するためのソフトウェアとして goat と YAGI を開発した。goat は Go 言語でシステムの仕様を記述できるライブラリで、コードを書くような設計体験、仕様の自動検証、設計と実装の一貫性の維持を可能にする。YAGI は goat を利用した仕様駆動開発の AI エージェントであり、AI との対話を通じて自然言語による仕様書と goat によるコードの仕様を作成できる。

日本のソフトウェア産業では、大手 SIer が設計を担当し下請け企業が実装を行う「IT ゼネコン」構造が存在し、設計と実装の間でフィードバックループが機能しにくい課題がある。また、多くの開発現場では論理設計が軽視される傾向にあり、設計手法の非標準化、設計と実装の統合の困難さ、設計の検証プロセスの不在といった問題が存在する。本プロジェクトは「Design as Code」、すなわちソフトウェア設計のコード化を一般的な開発手法として定着させることでこれらの問題を解決することを目指した。

7. 採択理由

本提案は、Go 言語による設計記述とモデル検証を可能にするソフトウェアを開発し、設計と実装の乖離や属人的な品質管理といったソフトウェア開発の構造的課題に対する実践的な解決策を目指すものである。設計をコードとして記述し、ソフトウェア的に検証可能にする「Design as Code」の思想により、設計の標準化・再利用性・検証性を同時に実現しようとしている。従来の形式手法の高い学習コストや特殊環境依存を、Go による実装を用いて回避することで、現場に適用しやすくなると提案者は述べている。PoC ではすでに非決定性や競合状態の検出も可能であることが示されていた。

初期的な構想自体は Go に特化しているが、将来的には他の言語による実装を用いることで、より広範な開発環境に適用が可能になると考えられる。本プロジェクトはいかにソフトウェア開発現場に浸透させるかという点も課題であるが、高い熱量をもつ提案者なら解決可能であると評価し採択とした。

8. 開発目標

本プロジェクトの開発目標は、Design as Code を実現するために必要なソフトウェアを開発し、既存のソフトウェア開発ライフサイクルに組み込めるライブラリを構築することであった。具体的には、ソフトウェア開発に馴染みがあるがモデル検査などの数学的知識を持たないエンジニアでも、ソフトウェア設計をコードとして記述することでモデル検査の恩恵を受けられ、かつ Design as Code を既存の開発サイクルに組み込むことができるライブラリの開発を目指した。

この目標を実現するライブラリ goat のコアバリューは、コードを書くような設計体験を実現すること、記述した設計の正しさが検証可能なこと、設計と実装の一貫性を維持しやすくすることの 3 点である。さらに、コードによる仕様を可視化するツール goat-cli と、goat を応用して仕様駆動開発を支援する AI エージェントアプリケーション YAGI を開発することを目指した。

9. 進捗概要

本プロジェクトでは、goat の MVP をリリースしてその改善を繰り返すことを計画しており、実際に goat は v0.1.0 から開発を開始して v0.5.0 まで複数回のリリースを行った。goat のコア API として、ステートマシンとイベントによるシステムのモデリング、condition と rule によるルール定義、ハンドラによる振る舞いの記述、そしてモデル検査の実行機能を実装した。さらに、仕様から Protocol Buffers のスキーマ生成、OpenAPI Schema 生成、gRPC サーバの E2E テストコードの生成機能も実装した。

プロジェクト開始当初の実施計画からの変更点として、仕様からアプリケーションのインタフェースとなるコードを直接生成する方式ではなく、IDL

（Interface Definition Language）を生成する方式に変更したことが挙げられる。また、goat-cli と YAGI の実装は実施計画では挙げられていなかった。goat-cli は goat による設計を Mermaid 記法のシーケンス図として可視化する CLI ツールであり、YAGI は生成 AI を活用した開発スタイルの急速な広がりを踏まえ、プロジェクト終盤で新たに開発したアプリケーションである。

10. プロジェクト評価

本プロジェクトは、日本のソフトウェア産業が抱える論理設計の軽視という構造的な課題に対して、「Design as Code」という新しい概念を提唱し、それを実現するためのライブラリを開発した意欲的なプロジェクトである。

goat は、Go 言語の型システムとジェネリクスを活用して型安全な API を提供しており、ソフトウェア開発に馴染みのあるエンジニアがモデル検査の数学的前提知識なしに設計の検証を行える点で、既存の形式手法ツール（TLA+や Alloy など）とは異なるアプローチを取っている。Go 言語の実行環境があれば動作するため環境構築が不要であり、CI/CD パイプラインへの組み込みも容易である。さらに、仕様から IDL や E2E テストコードを生成する機能により、設計と実装の一貫性を維持する仕組みが実現されている。

YAGI については、当初計画にはなかった新たな取り組みであるが、生成 AI の急速な発展という 2025 年の時代背景を的確に捉え、仕様駆動開発（Spec Driven Development）と Design as Code を組み合わせた開発支援ツールとして開発された。自然言語による仕様だけでなくコードによる仕様も併用することで、より正確な意思疎通と仕様の検証を可能にするという設計思想は理にかなっている。Google ADK を基盤としたマルチエージェント構成の実装も、技術的に適切な選択であった。

11. 今後の課題

goat および YAGI は、実務で利用されることで価値を発揮するプロダクトであるが、現時点ではまだユーザを獲得できていない。今後は goat の v1.0.0 のリリースと YAGI のデモアプリケーションの公開に取り組み、実際にユーザに使ってもらうことでフィードバックを受けて開発を進めていって欲しい。

goat では、モデル検査の表現力と実用性を高めるため、公平性制約の指定機能の追加や、大規模な状態空間を扱う際の実行コスト制御のための状態数上限設定機能の導入が計画されている。IDL 生成機能の改善や IDL 自体のバージョン管理機能も必要である。また、イベント処理の表現力を高めるため、未登録イベントに対する ignore や defer の仕組みを導入するなど、仕様の記述力を高めることも必要である。

YAGI では、本プロジェクトの途中から急遽取り組んだことなので PoC の状態に留まっている。デモアプリケーションとして公開できる水準まで完成度を

さらに高めることが必要になる。Human-in-the-Loop などのガードレールの実装や、セキュリティ面の改善も必要である。