IPA

2023 年度 未踏 IT 人材発掘・育成事業 採択案件評価書

1. 担当 PM

竹迫 良範（株式会社リクルート データプロダクトユニット ユニット長）

2. クリエータ氏名

芝山 駿介（早稲田大学 先進理工学部 物理学科）

3. 委託金支払額

2,736,000 円

4. テーマ名

Python にトランスパイル可能な静的型付けプログラミング言語の開発

5. 関連 Web サイト

- 公式サイト：https://erg-lang.org
- パッケージレジストリサイト：https://package.erg-lang.org
- GitHub リポジトリ：https://github.com/erg-lang/erg
- 公式ドキュメント：https://erg-lang.org/the-erg-book/

6. テーマ概要

本プロジェクトでは、Python にトランスパイル可能な静的型付けプログラミング言語 Erg およびその開発ツール群の開発を行った。言語仕様は公式サイトおよび GitHub リポジトリで公開し、ほとんどの機能が実装済みとなった。コンパイラは Rust で実装されており、MIT および Apache 2.0 のデュアルライセンスのオープンソースソフトウェアとして公開した。OS は Linux、macOS、Windows に対応している。開発ツール群としては、Language Server、パッケージマネージャ、インストーラ、パッケージレジストリおよびレジストリの登録済みパッケージを確認できる Web サイトを開発した。

7. 採択理由

本プロジェクトは、Python へトランスパイル可能な静的型付けプログラミング言語とその周辺ツールを開発する提案である。Rust 風な Python というコンセプトは 2023 年 5 月初旬にも Mojo が発表されたりと業界内でも人気が高く、

ホットである。

動的型付けのプログラミング言語に対して静的型システムを持ち込むという試みは、過去では JavaScript における TypeScript だったり、最近では LaTeX の代替を目指す静的型付け組版システム SATySFi や、シェルスクリプトの代替を目指す静的型付けスクリプト言語 Cotowali などの先例があり、多くのプログラマーの生産性を高めることに成功している。

最近は高速な Python 処理系の実装も増えてきていることから、多様なバックエンドの選択肢も増えている。さらにネイティブコード実装をすることで高速化の他にライブラリのバージョン依存関係をコンパイル時に確定することでシングルバイナリとしての再現・再配布が用意となるメリットもある。過去の Python ライブラリの資産をうまく活用できるのであれば、エコシステムとしても良い。

Julia や Scala など過去の先人達の設計や実装もリスペクトしつつ、TypeScript の Python 版のようなものが生まれることを期待して本提案を採択した。

8. 開発目標

Python は世界中で広く使われているプログラミング言語であり、特に機械学習やデータ分析の分野での利用が増加している。しかし、Python が実行時にコードを検証する動的型付けプログラミング言語であるため、静的型付けプログラミング言語に比べてバグの発見が遅れる傾向にある。また、先行するプログラミング言語 Julia では、実行効率向上のために部分的に静的型付けを採用しているが、動的型付けのコードを混入させることも可能であり問題解決に至っていない。

これらの問題を解決するために、Python にトランスパイル可能な静的型付けプログラミング言語 Erg 及びその開発ツールを開発することを目標とする。Erg 言語は、Python のコード資産をそのまま再利用可能でありながら、高い静的検証能力を持つことを目指す。開発するツール群を通じて、低い環境構築コストと高い再現性を実現し、Python の問題点を解消しつつ、開発者の生産性を向上させることを目指す。

9. 進捗概要

以下の設計思想に基づいて Erg 言語の設計を行った。

- 強い静的型付け
- ミニマルな基礎文法
- 高い表現能力
- Python API との互換性

- 高い開発効率

上記の設計方針に基づき、Erg コンパイラ、Language Server、パッケージマネージャ、インストーラ、パッケージレジストリ、パッケージレジストリサイトなどの開発を行った。

① Erg コンパイラ

Rust 言語を用いて Erg コンパイラを実装した。Erg コンパイラは Erg 言語のソースコード（拡張子.er）を解析・検査し、Python バイトコード（拡張子.pyc）に変換する（図 1）。コンパイルの各ステージで発生したエラーは収集され、一度にユーザに報告するようにした。エラーは該当箇所のコードとともに提示され、エラーメッセージもユーザにとって理解しやすいものとなるようにした。

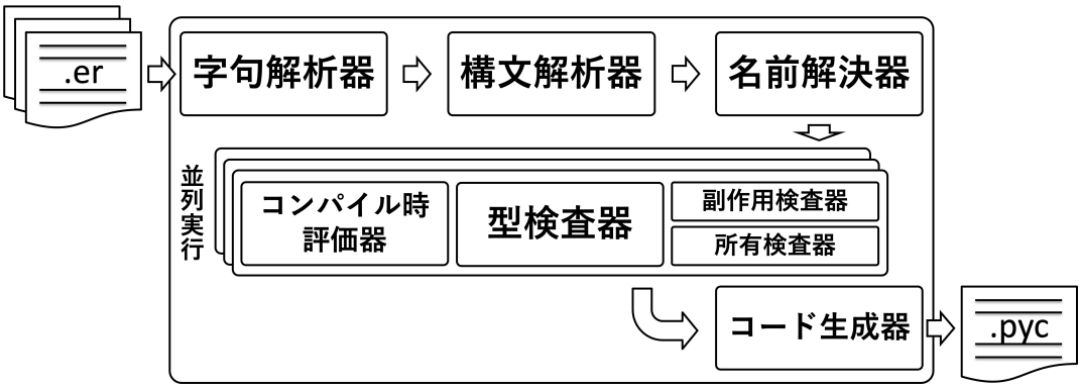

図 1：Erg コンパイラの概説図

② Python パッケージとしての Erg コンパイラ

Erg コンパイラは Python パッケージとしても機能し、Python インタプリタから動的に呼び出すことが可能になった。これにより、Erg モジュールを実行時に読み込んだり、Erg コードの AST を直接操作したりすることができるようになった。

③ Language Server

Erg Language Server も Rust で実装した。Language Server Protocol（LSP）に対応し、エラー表示や補完機能、定義ジャンプなどのコーディング支援機能をエディタに提供することができた。

④ パッケージマネージャ

パッケージマネージャを Erg 自身で実装した。パッケージの作成、ビルド、テスト、公開、インストールなどパッケージ開発のライフサイクル全般を管理することができた。

⑤ インストーラ

Erg 言語のコンパイラ、パッケージマネージャを一括でインストールするためのスクリプトを開発した。インストール先コンピュータのアーキテクチャに適合するコンパイラのバイナリと標準ライブラリを GitHub release から取得して自動でインストールする機能を持つ。

⑥ パッケージレジストリ

パッケージレジストリは GitHub を使用して構築した。パッケージマネージャがパッケージレジストリのリポジトリに対して Pull Request として送信し、メンテナの確認後に merge することでパッケージの登録を行う仕組みを作った。

⑦ パッケージレジストリサイト

パッケージレジストリに登録されたパッケージの情報を閲覧するための Web サイト（https://package.erg-lang.org）を構築した。この Web サイトにアクセスすることで、パッケージの検索やメタデータの閲覧が可能になった。

⑧ 著名な Python パッケージの型定義パッケージ群

Erg コンパイラは Python の組み込み API や標準ライブラリの型定義を同梱しているが、著名なサードパーティの Python パッケージの型定義のパッケージもレジストリに登録した。

10. プロジェクト評価

開発したプログラミング言語 Erg は、Python にトランスパイル可能でありながら完全に静的型付けを実現し、Python API の型定義を利用することで既存ライブラリの資産を簡単に呼び出すことができるようになった。世の中の先行する自作言語と比較して、Erg は実用プログラミング言語をかなり意識して開発され、コンパイラ、Language Server、パッケージマネージャ、インストーラ、パッケージレジストリ、パッケージレジストリサイトなどの周辺ツールもしっかり開発整備された。Erg 言語が単なる概念実証にとどまらず、実際の開発プロセスで利用可能な実用性を備えていると言える。

当初プロジェクトの計画ではネイティブコードバックエンドと Python インタプリタとのバインディング機構の実装に重点を置く予定であった。しかし、パッケージマネージャの開発途中で pyc バックエンドや Language Server に多くのバグや改善点が見つかり、これらの修正にプロジェクトの重心を移し品質を高めた。結果として、実用プログラミング言語としてのクオリティが向上した。

11. 今後の課題

完成度が高いものに仕上がっているが、今後の課題として以下のようなことが考えられる。

① マクロ機構の実装

Erg の基礎文法が Python とは表面的に乖離していることがユーザビリティの面で問題になりうる。これを解決するため、今後の拡張としてマクロ機構を導入することが考えられる。

② ネイティブコードバックエンドの実装

現在は Python バイトコードを生成するバックエンドが主流であるが、今後コンパイルターゲットを増やしていくために、ネイティブコードバックエンドの開発が必要である。

③ コンパイラの最適化

名前解決や型変数の除去処理がホットスポットであり、これらのコード最適化を行うことで、さらなるコンパイル速度の向上が期待できる。

④ 型定義パッケージの拡充

著名なサードパーティ Python パッケージの型定義パッケージをレジストリに登録したが、今後はより多くのライブラリの API に対応していく必要がある。

⑤ 広報活動の実施

Erg 言語を広く普及するため、共同開発者やユーザ獲得へ向けて広報活動も必要である。