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

田村 来希（京都大学 大学院情報学研究科 社会情報学コース 修士課程）

3. 委託金支払額

2,880,000 円

4. テーマ名

ライブマイグレーション可能な WebAssembly ランタイムと WASI をインタフェースとするライブラリ OS の開発

5. 関連 Web サイト

- Kafu SDK のリポジトリ：https://github.com/tamaroning/kafu/
- バイナリ変換ツールのリポジトリ：https://github.com/tamaroning/binaryen/tree/snapify
- デモのリポジトリ：https://github.com/tamaroning/kafu-demo
- ドキュメンテーション：https://tamaroning.github.io/kafu

6. テーマ概要

本プロジェクトでは、クラウドとエッジを透過的に連携する分散環境で WebAssembly（Wasm）モジュールを実行し、関数呼び出し境界でノード間マイグレーションを行うフレームワーク「Kafu」を開発した。近年、ロボットやドローンなどのデータをデバイス上で即時に分析したいニーズや、クラウド上で高負荷な計算を行いたいニーズが高まり、クラウドとエッジの連携が重要になっている。しかし、従来のクラウド・エッジ連携サービスの開発は、ネットワーク通信や RPC、状態保存処理など複雑な実装が必要であり、開発コストが高い。

Kafu は、アスペクト指向プログラミングの概念を導入し、C/C++のソースコードに関数属性（アノテーション）を追加するだけで実行ノードを動的に指定できる仕組みを実現した。開発者は単一プロセスのアプリケーションを書く感覚で分散サービスを記述でき、ネットワーク通信や状態保存を意識する必要がな

い。分散実行の基盤として WebAssembly を採用した理由は、その高い移植性と軽量性にある。

さらに、WebAssembly ランタイムのライブマイグレーション技術を開発し、プログラムを特定のランタイム実装に依存せずに中断・再開可能にする独自のバイナリ変換（Snapify）を実現した。メモリの差分転送や圧縮による通信の高速化を行い、リアルタイム画像処理のプログラムにおいて 100ms 程度でのライブマイグレーションが可能であることを実証した。成果物は SDK としてインストーラとともに配布し、ソースコードはオープンソースとして GitHub 上で公開した。

7. 採択理由

本提案は、WebAssembly（Wasm）とライブラリ OS を組み合わせ、OS や CPU アーキテクチャを超えてプログラムを中断・再開できるライブマイグレーション機能を備えた実行環境を開発するものである。従来の技術では不可能だったクロスプラットフォームでのライブマイグレーションを、Wasm のポータビリティとユーザ空間で動作するライブラリ OS により実現し、安全性と性能の両立を図ろうとしている。特に、JIT コンパイラとスタックマップによる低オーバーヘッドな移行処理、OS 資源（ソケットやファイル等）の状態保存、セキュアな設計を同時に備えようとしている点を評価した。既に PoC としてライブラリ OS や Wasm コンパイラを実装済みであり、移行速度やオーバーヘッドの観点で従来技術を上回る結果を得ている点も評価し採択とした。

8. 開発目標

本プロジェクトの開発目標は、開発者が通常のプログラム（単一プロセスのアプリケーション）を書くのと同じように、クラウド・エッジ上で分散実行されるサービスを記述できるフレームワークを作成することであった。具体的には、以下の技術を用いて開発を行った。

第一に、アスペクト指向プログラミングの導入である。C/C++のソースコードに関数属性（アノテーション）を追加するだけで、開発者がプログラムの実行ノードを動的に指定できるようにする。第二に、WebAssembly ランタイムのライブマイグレーションである。WebAssembly の軽量性と移植性を活かし、プログラム実行中の状態（チェックポイント）を保存・復元することで、ネットワーク越しに実行中のノードを高速に切り替える仕組みを実現する。

当初はライブラリ OS の開発も計画していたが、プロジェクト中盤でユースケースを再検討した結果、分散配置されるサービスの開発・実行のためのツール開発に方針を変更した。

9. 進捗概要

本プロジェクトでは、当初掲げていた開発目標に対して、以下の成果物を完成させた。SDK に含まれる主要なコンポーネントは、C/C++向けの Kafu ヘッダー、統合 CLI ツール（Kafu CLI）、C/C++から WebAssembly へのコンパイラ（kafu clang）、分散クラスタ内の各ノードとして動作する gRPC サーバ（kafu serve）、単一マシン上でクラスタの挙動をエミュレートする開発用環境（kafu singlenode）、Kubernetes マニフェストを Kafu コンフィグから自動生成するツール（kafu kustomize）、そして WebAssembly モジュールを中断・再開可能にするバイナリ変換ツール（Snapify）である。

プロジェクト開始当初の実施計画から変更した点として、ライブラリ OS の開発は行わず、分散配置されるサービスの開発・実行のためのツール開発に注力した。また、低レベルコンテナランタイムの開発も、分散サービス開発フレームワークへと方向転換した。

リアルタイム画像処理の実証では、差分転送と圧縮による最適化により 100ms 程度での高速なライブマイグレーションが可能であることを確認し、期待通りに安定動作することを実証した。

10. プロジェクト評価

本プロジェクトは、WebAssembly のライブマイグレーションという技術的に新規性の高いテーマに取り組み、コンパイラからデプロイツールまで一貫した SDK を田村氏一人で設計・実装し完成させた点において、高い技術力と実装力が発揮されたプロジェクトである。

特に、Binaryen をフォークして独自に開発したバイナリ変換ツール Snapify は、特定のランタイム実装に依存せずに WebAssembly モジュールに中断・再開機能を付与するという独創的なアプローチであり、従来のランタイム改造方式とは異なる新しい手法を提案している。Asyncify の技術を流用・拡張し、コールスタックのアンワインド・リワインドやグローバル変数の保存・復元をバイナリレベルで実現した技術的完成度は高い。

アスペクト指向プログラミングによる分散処理の抽象化は、開発者がネットワーク通信を意識せずに分散サービスを記述できるという点で、開発効率を向上させる可能性がある。デモアプリケーションとして、基本的なマイグレーション動作、機械学習を用いた画像分類、YOLO によるリアルタイム物体検出の 3 段階を用意し、フレームワークの実用性を段階的に示した点も適切であった。

11. 今後の課題

プロジェクトの前半において、提案するプロダクトの有効性を示すことができるユースケースの探索に多くの時間を費やした。ブロックチェーンやコントラクトなど様々な応用を検討したものの、決定的なユースケースを見出すこと

はできず、最終的に開発内容がエッジコンピューティング向けのサービス開発フレームワークに落ち着くまでに紆余曲折があった。プロジェクト期間中は時間切れになってしまったユースケースの検討は改めて考える必要がある。今後のプロジェクトの方針を決める上でも、ユースケースから逆算した機能開発が必要である。

具体的な今後の課題としては大きく以下の 2 つがある。第一に、Wasmtime 以外のランタイムの選択である。現在は WebAssembly ランタイムとして Wasmtime のみを利用しているが、本プロジェクトでは WebAssembly レベルのバイナリ変換を用いており、ランタイム実装に依存せずにライブマイグレーションが実現できる。組み込み向けの WAMR やエッジ向けの WasmEdge などを選択肢として提供できれば、ユースケースが広がることが期待される。第二に、外部状態の保存・復元のサポートである。現在はファイルやネットワークソケットなどの外部状態の保存・復元をサポートしておらず、移行元で開いたファイルディスクリプタは移行先で使用することができない。プロジェクトの目標である透過性を高めるには、OS 状態のキャプチャとライブマイグレーション時のスナップショットへの含め方を検討する必要がある。