Work-in-Progress: ESOps - An Agile Pipeline for Next-Generation Embedded Systems Development

Md Al Maruf, and Akramul Azim
Department of Electrical, Computer and Software Engineering, Ontario Tech University, Ontario, Canada

***Abstract*—Embedded Systems Operations (ESOps) introduces a feature-driven development pipeline for next-generation embedded system, leveraging domain-specific Large Language Models (LLMs) to expedite the development process. Targeted at industries such as automotive, aerospace, and healthcare, ESOps overcomes traditional development barriers by enabling a profound understanding of legacy codebase and integrating innovative AI-driven methodologies. This pipeline ensures systems predictability and reliability, meets stringent constraints, and shortens time-to-market by optimizing both the development and operational phases.**

***Index Terms*—Embedded Systems Development, ESOPs, Agile Pipeline, LLMs**

I. Introduction

Next-generation embedded systems involve complex interactions between software and hardware components, often within resource-constrained environments. The shift from sequential to distributed computing introduces unique challenges, particularly when safety-critical and less critical tasks must coexist within a multicore framework. System designers and developers frequently face significant delays as they strive to understand complex software requirements. Traditional C-based development methods, while reliable, typically lack the flexibility needed, leading to prolonged development cycles. Moreover, developers are hesitant to integrate and reuse legacy code due to potential safety compromises. ESOps addresses these challenges by leveraging agile, iterative pipelines enhanced by domain-specific LLMs [1]. This assists in analyzing the legacy codebase to extract software requirements, reuse existing features, generate code for new features, and identify potential software faults with solutions. Unlike MLOps and DevOps, which primarily focus on software development and operations for IT applications, ESOps specifically addresses the hardware-software co-design and the stringent reliability and predictability requirements.

II. Proposed Approach

This pipeline approach not only automates and refines requirements identification and code generation but also ensures that every system component adheres to defined feature relationships and constraints analyzing code, thereby enhancing predictability and system reliability.

a. *Legacy Code Analyzer*: Uses embedded LLMs (e.g., OpenAI) to analyze codebases, extracting key features and constraints.
b. *Feature Model Integrator*: Develops a feature model detailing the relationships and dependencies among system features.
c. *Code Synthesizer*: Generates code optimized for integration with existing legacy systems.
d. *Hardware-Software Integration Tool*: Ensures updated software is compatible with existing hardware, optimizing system performance.
e. *Predictive Task Scheduler*: Utilizes AI to optimize task scheduling based on criticality, WCET, and feature configurations.
f. *Human-AI Collaborative Editor*: Enables dynamic collaboration between developers and AI to refine code and models to meet project needs.
g. *Deployment and Automated Fault Detection*: Deploys systems with fault detection capabilities using LLMs to enhance reliability and minimize downtime.

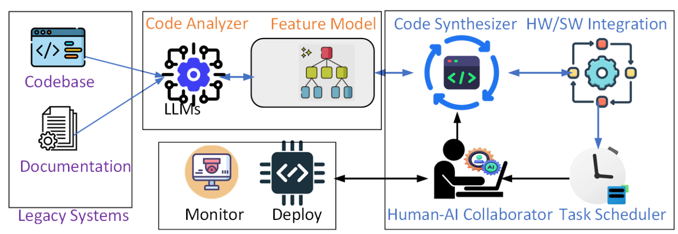

Figure 1: ESOps Workflow Diagram

The ESOps pipeline shown in Figure 1 begins with the *Legacy Code Analyzer*, which extracts software requirements from existing code to update the *Feature Model Integrator*. This updated model guides the *Code Synthesizer* in generating optimized code, which is then aligned with hardware configurations via the *Hardware-Software Integration Tool*. Task criticality is managed by the *Predictive Task Scheduler*, optimizing hardware-software interactions for performance. Throughout this process, the *Human-AI Collaborative Editor* refines outputs to ensure the software meets project standards. Concurrently, the *Automated Fault Detection system* after deployment identifies and resolves potential issues, ensuring system stability and efficiency.

III. Case Study and Conclusion

Using the ESOps pipeline, consider developing an advanced driver-assistance system (ADAS) for electric vehicles. ESOps streamlines the integration of legacy braking algorithms with new autonomous functionalities, employing domain-specific LLMs for code analysis and synthesis. This ensures compliance with safety standards and speeds up development.

References

[1] Z. Englhardt, R. Li, D. Nissanka, Z. Zhang, G. Narayanswamy, J. Breda, X. Liu, S. Patel, and V. Iyer. Exploring and characterizing large language models for embedded system development and debugging. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pages 1–9, 2024.

2771-571X/24/$31.00 ©2024 IEEE
DOI 10.1109/EMSOFT60242.2024.00008