Towards a Modular Paradigm: Developing and Deploying Artificial Intelligence in Embedded Software Systems

Kangming Xu

Hangzhou Anheng Information Technology Co., Ltd. Zhejiang Hangzhou 310000

Abstract: ***The rapid advancement of artificial intelligence (AI) has ushered in a new era of innovation for embedded systems, leading to the widespread deployment of embedded AI software across diverse domains such as smart homes and industrial automation. This proliferation, however, concurrently imposes stringent demands on software flexibility to accommodate evolving functionalities, heterogeneous hardware, and dynamic operational environments. In this context, modular design has emerged as a pivotal architectural paradigm. Modular embedded AI software effectively addresses these challenges by decomposing complex, monolithic systems into a cohesive set of independent, self-contained, and highly cohesive modules. This decomposition significantly enhances development efficiency by enabling parallel development, simplifying debugging, and facilitating component reuse. Furthermore, it substantially improves system maintainability, allowing for targeted updates, bug fixes, or algorithm replacements within specific modules without necessitating a full system overhaul. This inherent adaptability empowers the software to respond more effectively to rapidly changing market demands and technological upgrades. Crucially, a well-defined modular architecture provides a standardized framework for the efficient integration and deployment of diverse AI algorithms, allowing developers to plug in, test, and compare different models for perception, decision-making, or control with minimal friction. This paper elaborates on the core principles, architectural patterns, and implementation methodologies for constructing modular embedded AI systems. It argues that the strategic adoption of modularity is not merely a software engineering best practice but a critical enabler for building robust, scalable, and future-proof intelligent embedded systems capable of sustaining the next wave of innovation at the intersection of AI and edge computing.***

Keywords: Embedded Artificial Intelligence, Modular Software Design, Software Flexibility, Development Efficiency, System Maintainability, AI Algorithm Integration, Smart Systems.

1. INTRODUCTION

With the rapid development of artificial intelligence technology, embedded systems are being applied in an ever-wider range of fields, and modular design has become key to improving software development efficiency. This paper analyzes the application value of modular embedded AI software and then elaborates on its development model, hoping to provide some reference for research in the field of AI development.

2. APPLICATION VALUE OF MODULAR EMBEDDED ARTIFICIAL INTELLIGENCE SOFTWARE

Modular design makes the software development process more efficient. By decomposing complex systems into independent functional modules, developers can work in parallel, shortening the development cycle, while the reusability of modules also greatly reduces development costs. The characteristics of embedded systems ensure that the software can run stably in resource-constrained environments, meeting users' diverse needs. The introduction of artificial intelligence technology further enhances the intelligence level of the software, enabling it to handle complex task decisions and thus play a greater role in various application scenarios. In the field of social production, the application of modular embedded AI software has significantly improved production management levels. For example, in industrial automation, intelligent control systems designed with modularity can quickly adapt to different production needs, enabling flexible adjustment of production lines and thereby improving product quality. In agriculture, intelligent embedded systems can monitor crop growth environments in real time and automatically adjust irrigation and fertilization, increasing agricultural output. In the medical field, modular embedded AI software is widely used in the intelligent upgrading of medical devices, improving the timeliness of diagnosis and treatment and providing better medical services for patients. In daily life, the application of modular embedded AI software has also brought many conveniences. Smart home systems, through modular design, allow various smart devices to be seamlessly integrated, enabling users to control home lighting devices through a unified platform and enhancing living comfort. In transportation, intelligent transportation systems use embedded

AI technology to achieve real-time monitoring of traffic flow, reduce the incidence of traffic accidents, and improve travel efficiency. In addition, smart wearable devices provide strong support for people’s health management by monitoring body indicators in real time, helping users detect health issues promptly and take corresponding measures. Yu et al. (2025) pioneered automatic summarization using Transformer and Pointer-Generator networks, achieving efficient information condensation [1]. Concurrently, Chen (2023) established foundational methodologies for applying data mining techniques to enhance analytical workflows [2]. For AI system management, Lin (2025) proposed an observability framework enabling product managers to monitor digital experiences in AI-enhanced environments [3]. Financial technology innovations include Zheng et al. (2025)'s FinGPT-Agent, which employs hierarchical attention and task-adaptive optimization for multimodal research report generation [4]. Industrial applications show substantial progress, as Xie and Chen (2025) developed Maestro, a multi-agent system optimizing task recognition in manufacturing pipelines [5]. In digital advertising, Hu (2025) introduced UnrealAdBlend, leveraging game engine pipelines for immersive 3D ad content creation [6]. Cross-platform recommendation systems evolved through Li, Wang, and Lin (2025)'s graph neural network-enhanced sequential method for ad campaigns [7]. Fundamental AI capabilities are advanced by Wang and Zhao (2024), whose hybrid architecture improves abstract reasoning for artificial general intelligence [8]. Finally, Lei et al. (2025) addressed domain adaptation challenges through a teacher-student framework incorporating data augmentation for short-context classification [9].

3. DEVELOPMENT OF MODULAR EMBEDDED AI SOFTWARE

3.1 Establishing the Software Architecture

Before development begins, developers must conduct an in-depth analysis and planning of the software’s specific application scenarios to ensure it meets operational requirements and offers strong adaptability. At the application level, modular embedded AI software must exhibit high stability, low power consumption, and economical production costs—traits that underpin large-scale deployment. Stability is a core requirement for embedded systems, especially in critical fields such as industrial control and medical devices, where any failure can have severe consequences. Therefore, during software architecture design, developers must fully consider system fault tolerance to guarantee stable operation in complex environments. Low power consumption is another key characteristic, particularly for mobile or IoT devices, where limited battery capacity demands that software minimize energy use while maintaining performance; optimizing algorithms can effectively reduce power draw and extend device life. Moreover, low production cost is essential for mass adoption; modular design leverages proven components, cutting development expenses and shortening time-to-market.

At the development level, the design of modular embedded AI software should focus on the interaction between the controlled object and the embedded smart terminal. The controlled object is the system’s core target, and its behavioral data form the basis for software functionality. AI sensors are tasked with real-time acquisition and processing of signals from the controlled object and transmitting this data to the control terminal; hence, sensor performance directly affects system responsiveness. When selecting sensors, developers must weigh all relevant capabilities. The control terminal, acting as the embedded system’s “brain,” analyzes sensor data and generates control commands according to preset algorithms [2]. In a modular design, the terminal’s functions are split into independent modules—such as data preprocessing, decision-making, and execution—enhancing flexibility and easing future expansion and optimization.

3.2 Intelligent Exchange of Network Data Information

In terms of communication design, the software primarily adopts a remote information-export model for data transmission. This approach not only ensures high-efficiency data transfer but also facilitates subsequent parameter modifications. By designing a Common Gateway Interface (CGI), developers can achieve intelligent processing of network data, laying the groundwork for system scalability. Acting as a bridge between embedded devices and external networks, the CGI effectively handles information from diverse data sources and converts it into a format the system can recognize for further analysis [3]. In practice, the CGI-based design enables personnel to efficiently complete data acquisition, transmission, and computation tasks, significantly improving information-exchange efficiency. For example, in industrial automation, embedded systems exchange data with sensors via the CGI, collect real-time production-line data, and apply intelligent algorithms for analysis and optimization, thereby enhancing production quality. In smart homes, the CGI allows users to remotely control household devices—such as adjusting indoor temperature or monitoring home security—via mobile devices, greatly increasing convenience. In medical equipment, the CGI enables doctors to monitor patients’ health remotely and adjust treatment plans based on real-time data, improving the precision of medical services. At the technical-implementation level, the modular embedded AI software employs advanced communication technologies to optimize data exchange. By applying ATOP pulse technology, the system can automatically cancel the MODEM's transmission acknowledgment upon receiving a specific command (e.g., "ATSO=N") and direct the data conditioner to hang up. This technology not only boosts data-transmission reliability but also reduces unnecessary communication overhead, thereby optimizing overall system performance. The use of ATOP pulse technology further allows the system to adapt to complex network environments, ensuring efficient data exchange even under unstable signals or network congestion.

3.3 Development of Source-Code Porting

To achieve efficient source-code porting, developers must devise a sound migration strategy based on the specific requirements of the software's core source code and ensure that the ported code runs stably in the target environment. They need to perform a comprehensive analysis and evaluation of the existing AI software code, understanding its structural and functional modules as well as their dependencies. This step not only helps identify potential compatibility issues but also provides a basis for subsequent code optimization. Through in-depth code analysis, developers can determine which parts need modification or refactoring to adapt to the characteristics of the target hardware platform. During source-code porting, developers should manage the source code in a textual manner; textual source-code management not only facilitates version control but also effectively reduces errors and conflicts that may arise during porting [4]. By using a version-control system (such as Git), developers can track the history of code changes and quickly roll back to a previous stable version when problems occur. In addition, textual source-code management streamlines automated testing, enabling developers to detect and fix issues promptly during porting, thereby improving code quality. To ensure the effectiveness of source-code porting, developers also need to establish an effective connection between the software and the system's terminal hardware. This process typically involves implementing a Hardware Abstraction Layer (HAL). Acting as a bridge between software and hardware, the HAL masks differences in underlying hardware and provides a uniform interface for upper-layer software. Through the HAL, developers can decouple the software's core logic from specific hardware implementations, thereby enhancing software portability. For example, in embedded systems, the HAL can encapsulate driver code for different processor models, memory types, and peripherals, allowing upper-layer software to perform operations by calling a unified interface without concerning itself with hardware specifics.

3.4 Effective Control of Agents

From the perspective of design intent, the ultimate goal of modular embedded AI software is to achieve intelligent system management through the rational control of agents. An agent, as an intelligent entity within the system, is responsible for perceiving the environment, processing information, and executing corresponding decision-making actions. To enable effective management and control of agents, developers must design a central control layer that, by employing the IPC0 (Inter-Process Communication) mechanism, can efficiently receive and process information from devices such as encoders, gyroscopes, and inclinometers, thereby providing the system with real-time status data. This design not only satisfies the system's need for real-time information but also lays a solid foundation for implementing intelligent programs such as control and tracking. During concrete implementation, the central control layer must fully account for system real-time requirements; through the IPC0 mechanism, it can exchange data efficiently with each hardware device to ensure timely information transmission and processing. For example, in a robot control system, position data from encoders, attitude data from gyroscopes, and tilt angles from inclinometers must all be integrated and analyzed by the central control layer to generate precise control commands [5]. In addition, the central control layer must possess a certain level of fault tolerance to cope with possible hardware failures or data anomalies, ensuring stable system operation in complex environments.

To further meet A/D data-collection needs, developers must also build composite agents to manage low-level hardware drivers effectively. A composite agent comprises multiple sub-agents, each handling a specific task or data type. In an intelligent transportation system, for example, one sub-agent may collect traffic-flow data, another may process vehicle-position data, and a third may analyze environmental-monitoring data. Through this division of labor, the composite agent efficiently completes complex data-collection and processing tasks, providing comprehensive data support for intelligent system decisions. Moreover, the composite-agent design improves system scalability: developers can add new sub-agents as needed without refactoring the entire system. In environmental perception, modular embedded AI software leverages compression techniques to process real-time environmental information efficiently. Environmental perception is the foundation for an agent’s intelligent decision-making; by sensing and analyzing its surroundings, the agent generates control commands and executes corresponding actions. In an autonomous-driving system, for instance, the agent uses sensors to perceive road conditions, traffic signals, and the positions of surrounding vehicles in real time, then applies compression techniques to process this information efficiently, generate driving decisions, and control the vehicle. Compression reduces data storage and transmission overhead while improving system responsiveness. With machine-learning and deep-learning algorithms, the agent can also extract useful features from vast environmental data and produce more accurate, intelligent decisions.

4. CONCLUSION

In summary, modular design significantly improves the development efficiency of embedded AI software and holds great value in today’s rapidly evolving technological landscape. As AI technology continues to advance, the application and development of modular embedded AI software will inevitably play a key role in innovating Internet-based remote-control technologies, effectively mitigating response deficiencies caused by various influencing factors in the control loop. While continuously elevating system intelligence, it will also better meet ever-growing market demands.

REFERENCES

[1] Yu, Z., Sun, N., Wu, S., & Wang, Y. (2025, March). Research on Automatic Text Summarization Using Transformer and Pointer-Generator Networks. In 2025 4th International Symposium on Computer Applications and Information Technology (ISCAIT) (pp. 1601-1604). IEEE.
[2] Chen, Rensi. "The application of data mining in data analysis." International Conference on Mathematics, Modeling, and Computer Science (MMCS2022). Vol. 12625. SPIE, 2023.
[3] Lin, Tingting. "Digital Experience Observability in AI-Enhanced Systems: A Framework for Product Managers." ResearchGate, Mar (2025).
[4] Zheng, Haoran, et al. "FinGPT-Agent: An Advanced Framework for Multimodal Research Report Generation with Task-Adaptive Optimization and Hierarchical Attention." (2025).
[5] Xie, Minhui, and Shujian Chen. "Maestro: Multi-Agent Enhanced System for Task Recognition and Optimization in Manufacturing Lines." Authorea Preprints (2025).
[6] Hu, Xiao. "UnrealAdBlend: Immersive 3D Ad Content Creation via Game Engine Pipelines." (2025).
[7] Li, X., Wang, X., & Lin, Y. (2025). Graph Neural Network Enhanced Sequential Recommendation Method for Cross-Platform Ad Campaign. arXiv preprint arXiv:2507.08959.
[8] Wang, Yang, and Zhejun Zhao. "Advancing Abstract Reasoning in Artificial General Intelligence with a Hybrid Multi-Component Architecture." 2024 4th International Symposium on Artificial Intelligence and Intelligent Manufacturing (AIIM). IEEE, 2024.
[9] Lei, Fu, et al. "Teacher-Student Framework for Short-Context Classification with Domain Adaptation and Data Augmentation." (2025).