Proceedings of CONF-MLA 2025 Symposium: Intelligent Systems and Automation: AI Models, IoT, and Robotic Algorithms
DOI: 10.54254/2755-2721/2025.LD27511

A Study of AI-Oriented Development for HarmonyOS Applications

Zihan Wei

Nanjing University of Posts and Telecommunications, Nanjing, China
b23080731@njupt.edu.cn

Abstract. IoT and AI technologies are now deeply combined. The age of ubiquitous intelligence needs smarter applications. Huawei's HarmonyOS is a distributed operating system for all scenarios. It provides a good platform for building smart IoT applications. But now, most studies about HarmonyOS application development only talk about common IoT tasks. There is no complete development method for adding AI functions. To solve this problem, this paper first looks at the current situation and difficulties of IoT and HarmonyOS application development. It points out the problems in supporting smart applications. Then, this paper puts forward an AI-based development method for HarmonyOS applications. It includes the whole process: setting up the environment, developing at the same time, distributed debugging, and putting applications into use. This method also explains the special benefits of combining HarmonyOS's own AI modules, such as computer vision and natural language processing, with distributed technologies. To test if this method works well, the paper uses two examples: a "Smart Temperature and Humidity Monitoring and Early Warning System" and an "AI Visual Cleaning Robot." These examples show how to quickly build smart applications with edge computing and cross-device cooperation on the HarmonyOS platform. The results show that the development method given in this paper is useful. It helps developers use HarmonyOS's AI functions fully. They can build high-performance smart applications for all kinds of IoT situations.

Keywords: OpenHarmony, IoT, AI applications, Software Development

1. Introduction

As the Internet of Things (IoT) develops into the "Internet of Everything," combining AI and IoT (AIoT) has become a big trend. Huawei's HarmonyOS has a distributed architecture. It brings new chances for AIoT development. But HarmonyOS development is still in the early stage. Developers do not have a standard method to make AI applications. This makes HarmonyOS's distributed AI potential not fully used.

To solve this problem, this paper studies AI-based development methods for HarmonyOS. Our main contributions are:

1. Developing a Paradigm: We put forward a standard, AI-oriented development process. It covers the whole life of an application.
2. Analyzing Key Technologies: We study how to combine HarmonyOS's built-in AI modules with its distributed technologies.

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2025.LD27511

3. Validating with Case Studies: We build a "Smart Temperature and Humidity Monitoring System" and an "AI Visual Cleaning Robot." We use them to test if our proposed paradigm works.

This research wants to give HarmonyOS developers a useful guide. It helps them build high-quality, smart native applications. In this way, it can help the "Internet of Everything" ecosystem grow better.

2. Background

2.1. IoT applications and the Internet of Everything (IoE)

The Internet of Things (IoT) is a network of physical objects. These objects have sensors and processors to connect and share data. At first, big hardware slowed its development. But with technologies like RFID, it developed faster. Now devices are smaller, quicker, and smarter.

The Internet of Everything (IoE) is an extension of IoT. It combines people, processes, and data to make more valuable connections. This change is making society move from simple information access to real "intelligentization". It is changing industries like manufacturing, healthcare, and transportation. It also helps the economy grow. From a technical view, IoT architecture has four layers: perception (sensors), network (data transmission), platform (data processing), and application (smart homes, industry).

2.2. HarmonyOS applications

HarmonyOS is a distributed operating system. It was developed and released by Huawei. From the view of distributed OS architecture, HarmonyOS has four parts: Distributed Soft Bus, Distributed Device Virtualization, Distributed Data Management, and Distributed Task Scheduling. This design makes it easier to develop distributed applications. From a layered architecture view, HarmonyOS has four layers: Kernel Layer, System Service Layer, Framework Layer, and Application Layer. Besides, the system can be customized. We can change subsystems or functional modules according to specific needs. The detailed system architecture is shown in Figure 1 [1].

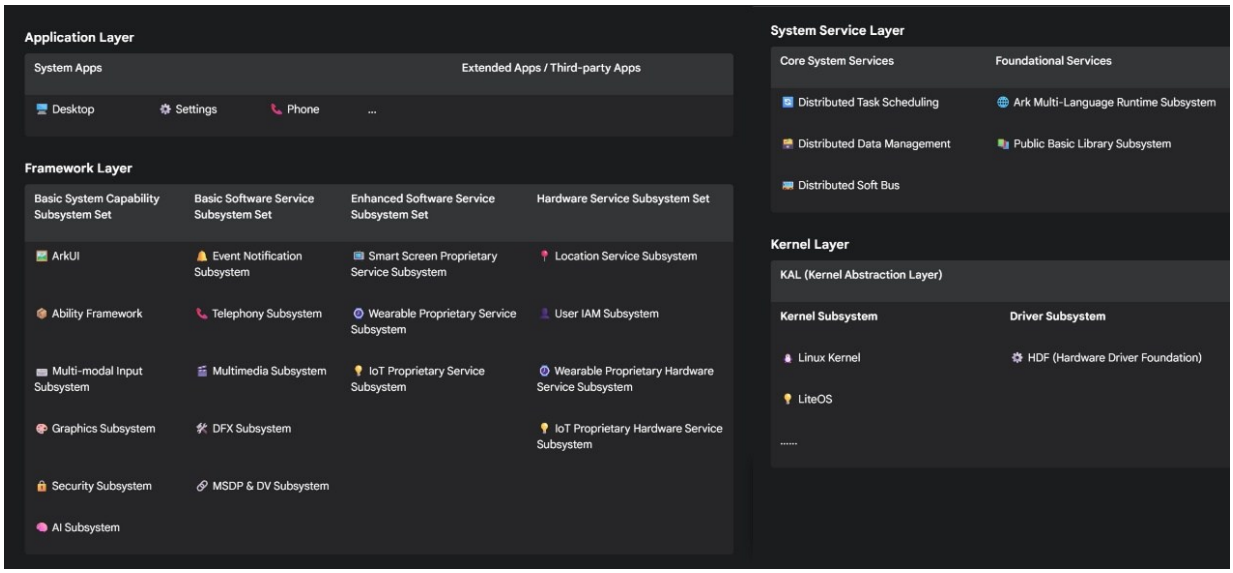

Figure 1. Layered architecture of HarmonyOS

The core of HarmonyOS is a microkernel. Compared with a monolithic kernel, it only keeps the most important core functions. These functions include memory management, task scheduling, and inter-process communication. All other functions are moved outside the kernel to keep it simple.

DOI: 10.54254/2755-2721/2025.LD27511

Because of this simple design, the kernel has fewer tasks and less code. This reduces system problems and makes the system more secure, stable, and easy to maintain. But the microkernel is slower in performance. Tasks that could be done in a monolithic kernel need to be coordinated across multiple microkernel parts. Using HarmonyOS's development methods, the system is highly reliable. It works on different platforms, manages memory well, and supports many application scenarios.

2.3. AI-based application software

AI applications are now very common. They can be divided into four main types. General applications are intelligent assistants for office work (for example, DingTalk AI). Creative tools make content like art or code based on what users ask (for example, Wenxin Yige). Industry-specific applications are for fields like finance, education, and healthcare. But their AI functions are often not very mature. Finally, AI hardware applications combine AI with smart devices. They make voice assistants or digital agents (for example, Xiao Ai). These advanced tools are more and more used in daily life. They bring users great convenience. They also help users make decisions and plans better.

3. HarmonyOS application development study

Research on the HarmonyOS IoT application ecosystem shows it is becoming more and more advanced. Important studies have focused on several areas:

1. Comparison with Android/iOS: Li and his team [2] studied the software engineering differences between HarmonyOS and traditional mobile applications in a systematic way.
2. Architecture and Distributed Capabilities: Researchers have shown that HarmonyOS is valuable in systems for wind tunnel data collection [3] and wearable IoT health monitoring [4].
3. Compatibility Solutions: Ma and his team [5] studied API changes to solve compatibility problems caused by old versions in open-source programs.
4. Open-Source Value: Niu and his team [6] analyzed patch information. They found that as patches increased, error rates in HarmonyOS became much lower. This shows HarmonyOS is easy to use and has low deployment cost.
5. Industrial Applications: Tian and his team [7] built a cloud-edge-end system to monitor industrial environmental data.
6. OpenHarmony Ecosystem: Studies have proved that it is possible to port third-party libraries [8]. They also explored its potential in Vehicle Infrastructure Cooperative Systems (VICS) [9] and real-time data monitoring [10].

Although these studies have new ideas, the study points out that current HarmonyOS application development mainly focuses on common tasks. It has not systematically solved intelligent application scenarios. This limits the practical use and scope of the applications made so far.

4. AI-based HarmonyOS application development

Based on the survey in Section 3, we can clearly see that AI-based task scenarios take up a large part of practical applications. However, there are no special design solutions for these AI applications. This is why we do the research in this paper. Focusing on the features of HarmonyOS development, this study proposes an AI-oriented development paradigm under HarmonyOS and proves its effectiveness through practical examples.

4.1. Overall workflow for intelligent HarmonyOS application development

To systematically guide the development of HarmonyOS applications with AI functions, we built a standardized development workflow. This workflow covers the full process from environment setup

DOI: 10.54254/2755-2721/2025.LD27511

to application release. It pays attention to both AI function integration and distributed scenario needs. The overall workflow for intelligent HarmonyOS application development is divided into four core stages: Preparation, Development, Testing and Debugging, and Release and Iteration. The development workflow for HarmonyOS applications is illustrated in Figure 2:

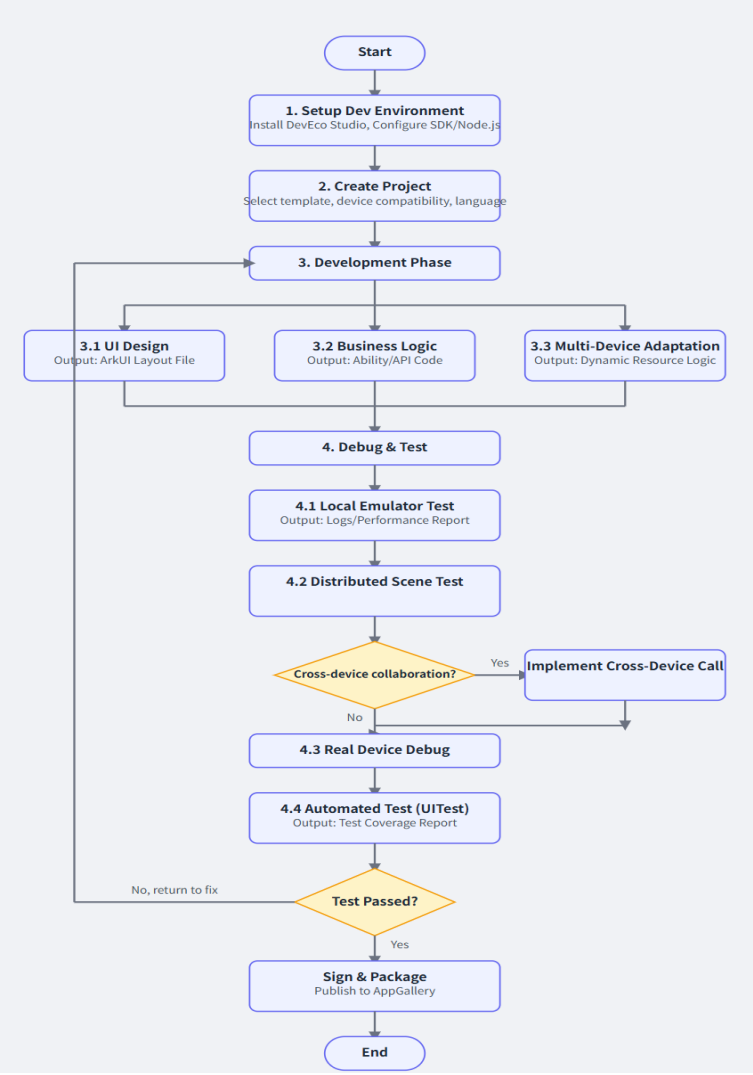

Figure 2. Overall workflow for intelligent HarmonyOS application development

The workflow has the following key characteristics:

Modular Parallel Development: In the core Development Phase of intelligent IoT applications, UI design, intelligent business logic implementation, and multi-device adaptation are parallel submodules. Developers can work on these modules at the same time and put them together at the end. This improves development efficiency.

(1) UI Design: Outputs ArkUI layout files.

(2) ntelligent Business Logic Implementation: Outputs Ability code and AI-related distributed API code.

(3) Multi-Device Adaptation: Outputs dynamic resource loading logic.

Intelligent Input and Output Management: Each stage of the workflow has clear inputs and outputs.

(1) Input: The intelligent development process starts with initial information such as requirement documents, design drafts, target device selection, and developer account credentials.

DOI: 10.54254/2755-2721/2025.LD27511

(2) Output: Final products include installable HAP/APP packages, successfully published applications in smart marketplaces, and user feedback data for future updates.

Intelligent Iterative Updates: Application release is not the final step. The workflow emphasizes continuous maintenance. User feedback starts a new round of intelligent analysis and version updates. This forms a complete closed loop from development to market and back to development. It ensures continuous optimization and long-term vitality of the application.

Embedded Intelligence: The biggest difference between this paradigm and traditional development workflows is that AI capabilities are fully integrated in the whole process. From choosing AI-related SDKs when creating a project, adding AI models as core parts of business logic during development, to testing AI algorithm accuracy and performance specially, intelligence is not a later added function. It is an inherent and deeply integrated part of the whole application development process.

4.2. Detailed module overview

HarmonyOS has a layered architecture. It embeds rich AI functions in the System Service Layer and Framework Layer. These AI functions are not separate. They are deeply combined with the system's underlying distributed architecture, microkernel, and low-latency features. This forms its unique AI technological advantages.

4.2.1. Computer vision module

HarmonyOS's computer vision functions not only provide a full set of APIs but also have distributed collaborative vision. Using distributed soft bus technology, multiple device cameras can be combined into a logical "super camera" [9]. For example, an application can use the front camera of device A and the rear wide-angle camera of device B at the same time. It can do multi-angle facial recognition or environmental perception. This helps finish complex visual tasks that a single device cannot do. This function lets developers create new application scenarios, such as multi-camera live streaming and collaborative security monitoring.

4.2.2. Natural Language Processing (NLP) module

The NLP module of HarmonyOS is good at understanding user intent based on distributed user profiles. Through distributed data management, the system can safely combine user behaviors and data from multiple devices (with user permission). This creates a complete user profile. Based on this, the NLP module can not only understand single commands but also predict user intent by considering context and user habits. For example, if a user searches for a recipe on a tablet and then goes to a smart refrigerator and says, "I need help," the system can guess that the user wants to cook. It will actively show the needed ingredients on the refrigerator screen. This shows proactive intelligence across devices.

4.2.3. Speech recognition and synthesis module

This module's advantages are low latency and highly reliable edge processing. Thanks to HarmonyOS's microkernel architecture and deterministic latency engine, the speech processing process is much shorter. This makes response times faster. Moreover, most speech recognition and synthesis tasks can be done at the edge, without relying on the cloud. This not only protects user privacy but also ensures that core functions like voice control work even when the network is poor or offline. This is very important for scenarios that need high reliability, such as smart homes and in-vehicle systems.

DOI: 10.54254/2755-2721/2025.LD27511

4.2.4. AI framework and tool support

The core AI framework of the HarmonyOS ecosystem is the cloud-edge collaborative MindSpore [11]. Its lightweight edge inference framework, MindSpore Lite, is optimized for IoT devices with limited resources. It supports efficient AI model execution at the edge. The cloud-edge collaboration function lets developers train and optimize models on the cloud using the full MindSpore framework. Then they can deploy the models to various HarmonyOS devices with one click. Additionally, techniques like federated learning let devices work together to optimize models while protecting data privacy. This forms a continuously evolving intelligent loop.

4.2.5. Distributed AI capability

This is the core feature of HarmonyOS AI. It frees AI computation from individual devices and lets AI functions flow and be shared freely. Through distributed task scheduling, a complex AI task (such as real-time style transfer for high-definition video) can be split automatically. The computational work is dynamically assigned to the most suitable devices in the network (like a PC or smart display). The device that starts the task (such as a smartphone) only shows the results. This compute-sharing model lets lightweight devices use the AI power of more powerful devices. It greatly expands the scope of AI application scenarios.

4.3. Case study

To clearly show how the proposed development paradigm and AI module features are used in real projects, this section gives two development examples. They are used to test the general applicability of the workflow and show the practical effectiveness of HarmonyOS AI capabilities.

4.3.1. AI-based intelligent temperature and humidity monitoring with anomaly warning system

This case aims to develop an intelligent monitoring system. It can collect environmental temperature and humidity in real time. It uses an edge AI model to predict risks such as fire danger.

Verification of Development Workflow

The development of this case strictly follows the workflow defined in Section 4.1, as detailed below:

(1) Preparation: We clarified the requirements including temperature and humidity monitoring with AI-based anomaly warning. We selected hardware including Hi3861 development board, SHT30 sensor, and HarmonyOS smartphone. We set up the development environment using DevEco Studio.

(2) Development Phase: This stage shows modular parallel development. The team was divided into two groups. One group was responsible for business logic on the Hi3861 device such as sensor data collection and AI inference. The other group was responsible for UI design on the smartphone including monitoring curve interface and alert pop-ups. The two groups worked at the same time and connected through predefined distributed communication APIs.

(3) Testing and Debugging: At first, we tested sensor data collection accuracy on a local emulator. Then we did distributed scenario testing to check whether an anomaly detected by the Hi3861 could successfully trigger alerts on the smartphone. This step corresponds to the “Cross-Device Collaboration?” decision branch in the workflow diagram. It needs special testing of distributed communication stability and latency.

(4) Release and Iteration: After testing, we packaged the firmware for the data collection device and the smartphone HAP package separately. Future updates can use user feedback such as false alarm rate and power consumption to optimize the AI model or communication logic.

DOI: 10.54254/2755-2721/2025.LD27511

AI Capability Application and Effectiveness

(1) Core AI Capability: This case mainly used AI framework and tool support. We trained a lightweight decision tree model in the cloud and deployed it to the resource-limited Hi3861 development board using MindSpore Lite [11]. It achieved real-time edge intelligence. This shows HarmonyOS AI can support low-power, low-computation devices.

(2) System Effectiveness: The final system allows the smartphone UI to clearly show historical temperature and humidity curves. When the edge AI model finds a fire risk, the smartphone receives the alert within three seconds. It shows a high-priority notification and marks the abnormal data points on the interface.

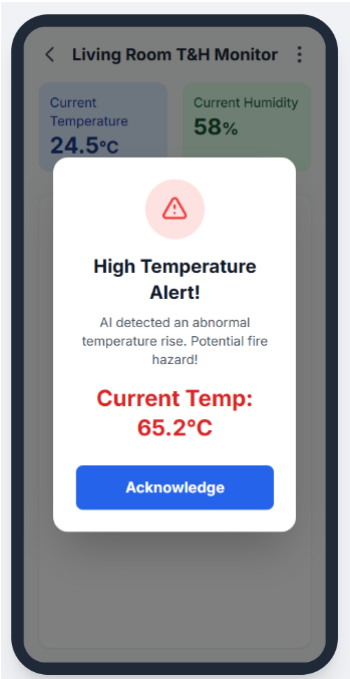

Figure 3. Smartphone UI of the intelligent monitoring system

4.3.2. AI smart cleaning robot with integrated visual recognition

This case shows the development of an intelligent cleaning robot. It has AI-based obstacle avoidance and trash recognition functions. It tests more complex AI functions and distributed collaboration.

Verification of Development Workflow

(1) Preparation: We defined the project goal including vision-guided cleaning robot. We selected hardware platforms including HarmonyOS-enabled smart car with camera and HarmonyOS tablet. We set up a DevEco Studio environment that supports C++ and ArkTS hybrid development.

(2) Development Phase: This stage also shows modular parallel development.Business Logic Group: Responsible for the robot-side C++ low-level logic including motor control, camera data collection, and integration of the YOLOv5-lite [12] model for computer vision processing.

UI Design Group: Responsible for developing the ArkUI interface of the tablet-side control application including real-time video display area and virtual joystick.

Multi-Device Adaptation Group: Responsible for implementing distributed communication logic between the robot and tablet including video stream transmission and remote procedure call (RPC) [13] for control commands.

DOI: 10.54254/2755-2721/2025.LD27511

(3) Testing and Debugging: We first tested the visual algorithm on the robot on a PC using datasets. Then we did real-device testing to see obstacle avoidance in a real environment. Finally, distributed scenario testing made sure that remote control commands from the tablet were correctly executed by the robot and that the video stream stayed smooth. This phase included iterative cycles of testing, returning to development, and solving problems.

AI Capability Application and Effectiveness

(1) Core AI Capabilities: This case widely used the computer vision module and distributed AI capability.

On the robot, the edge-deployed YOLO model can recognize obstacles and trash in real time. It is typical edge AI computation.

More importantly, it shows distributed AI collaboration. The robot does AI-based perception and decision-making. The tablet acts as an "enhanced display and interaction center." User operations on the tablet such as guiding cleaning paths combine human intelligence with the robot's AI intelligence through distributed communication to finish tasks together.

(2) System Effectiveness: The cleaning robot can plan paths by itself. It can avoid obstacles such as slippers and cables well. It can recognize common trash such as paper balls and clean them. On the tablet, users can see real-time high-definition video sent from the robot. They can guide it to specified areas by touching the screen. It achieves easy and efficient human–machine interaction.

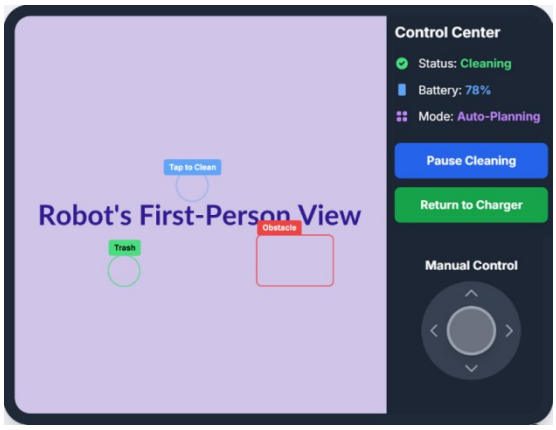

Figure 4. Tablet-side control UI of the AI smart cleaning robot

Through these two typical case studies, we have proved the rationality and general applicability of the proposed development workflow. We also clearly showed the powerful functions of HarmonyOS in edge-side AI and distributed AI collaboration. These cases show that HarmonyOS provides developers with a full-stack AI solution. It ranges from simple sensor intelligence to complex visual recognition. It effectively supports diverse, high-performance AI application development needs in future Internet of Everything scenarios.

The source code of figure3 and figure4 would be available at https://www.dropbox.com/scl/fi/qp1e37nzrwv7zf9sfvh1u/HarmonyOS-AI-App-UI-Mockups.paper?rlkey=r6666qu9htye658u5m3hdr3yw&dl=0

5. Conclusion

The study aims to solve the current lack of an intelligent application development method in HarmonyOS. First, we review the current state of HarmonyOS applications in the IoT field and find the limitations of existing development models. Then, we build a complete AI-oriented HarmonyOS application development method. Furthermore, the paper deeply analyzes how HarmonyOS deeply combines AI functions with distributed technologies. It shows its core advantages in collaborative

DOI: 10.54254/2755-2721/2025.LD27511

perception, intent prediction, and computing power sharing. Through the development and practice of two representative cases—an intelligent temperature and humidity monitoring system and an AI cleaning robot—the proposed method is successfully proved to be effective and widely usable. The cases show that using this method can greatly help developers deploy AI models on resource-limited edge devices and efficiently achieve cross-device intelligent collaboration. Finally, it can create high-performance AI applications that meet the needs of complex scenarios. Future work may focus on deploying and optimizing more complex AI models in the HarmonyOS distributed environment. It may also research AI development tools for specific industries such as intelligent healthcare and industrial IoT. The goal is to further lower development difficulties and expand the application scope of HarmonyOS in the AI field.

References

[1] OpenHarmony. (2025). OpenHarmony operating system overview. https://docs.openharmony.cn/pages/v4.1/zh-cn/OpenHarmony-Overview_zh.md
[2] Li, L., Gao, X., Sun, H., et al. (2023). Software engineering for OpenHarmony: A research roadmap. arXiv preprint arXiv:2311.01311.
[3] Zhang, C., Li, X., & Wu, H. (2024). Application of HarmonyOS in wind tunnel data collection system. In Third International Conference on Electronic Information Engineering and Data Processing (EIEDP 2024) (Vol. 13184, pp. 932–936). SPIE.
[4] Liu, S., Rong, X., Li, S., et al. (2022). Design of HarmonyOS-based IoT system for wearable health monitoring. In International Conference on Internet of Things and Machine Learning (IoTML 2021) (Vol. 12174, pp. 66–71). SPIE.
[5] Ma, T., Zhao, Y., Li, L., et al. (2023). CiD4HMOS: A solution to HarmonyOS compatibility issues. In 2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE) (pp. 2006–2017). IEEE.
[6] Niu, Y., Zhou, L., & Liu, Z. (2022). A deep look at the code changes in OpenHarmony. In Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data (pp. 426–434). Cham: Springer Nature Switzerland.
[7] Tian, H., & Ye, K. (2023). CEESys: A cloud-edge-end system for data acquisition, transmission and processing based on HiSilicon and OpenHarmony. In International Conference on Metaverse (pp. 3–14). Cham: Springer Nature Switzerland.
[8] Xu, Y., Yang, J., Liang, D., et al. (2024). MQTT third-party library porting method and application based on the OpenHarmony standard system. In 2024 International Conference on Networking and Network Applications (NaNA) (pp. 305–310). IEEE.
[9] Chen, J., Zhuang, H., & Yang, M. (2023). A distributed vehicle-infrastructure cooperation system based on OpenHarmony. In International Conference on Cognitive Computation and Systems (pp. 258–271). Singapore: Springer Nature Singapore.
[10] Li, H., Zhang, B., Wu, Z., et al. (2024). A study on real-time monitoring of meter-driven data in power systems based on OpenHarmony. In 2024 International Conference on Artificial Intelligence and Power Systems (AIPS) (pp. 519–523). IEEE.
[11] Huawei. (2025). MindSpore: A new-generation all-scenario AI framework. https://www.mindspore.cn/
[12] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 779–788).
[13] OpenHarmony. (2025). Distributed RPC. https://docs.openharmony.cn/pages/v4.1/zh-cn/application-dev/reference/apis-ability-kit/js-apis-inner-ability-wantAgent.md/