Scholar Publishing Group
International Journal of Big Data Intelligent Technology
https://doi.org/10.38007/IJBDIT.2025.060214
ISSN 2790-0932 Vol. 6, Issue 2: 130-138

Distributed Architecture and Performance Optimization for Smart Device Management

Yajing Cai

Alexa Identity Service, Amazon.com Inc, Seattle 98121, Washington, USA

Keywords: Intelligent Device Management, Distributed Architecture, Performance Optimization, Equipment Scheduling, Task Assignment

Abstract: With the wide application of intelligent devices, the efficiency and performance requirements of intelligent device management system are increasing, and the traditional centralized architecture faces the problems of data processing bottleneck and system load inequality. Because of its high efficiency, flexibility and scalability, distributed architecture becomes an effective means to solve these problems. This paper discusses the design and application of distributed architecture in intelligent device management, and analyzes the performance optimization strategy. This paper introduces the basic concept of intelligent device management and features of distributed architecture, and analyzes its application in device scheduling, task allocation, security and reliability. Finally, some strategies such as data compression, transmission protocol optimization, efficient data retrieval and indexing, and asynchronous computing are proposed to improve the system performance and response speed. This paper provides a feasible architecture design and optimization scheme for intelligent device management system, which has important application value.

Introduction

Smart device management is the core component of the Internet of Things and smart city construction, with the rapid development of smart terminal and Internet of Things technology, the number of devices is growing exponentially, and the traditional centralized management can no longer meet its high efficiency, flexibility and scalability needs. The introduction of distributed architecture solves this problem by distributing data storage, processing, and task distribution to multiple nodes, reducing the impact of a single system bottleneck and providing better fault tolerance and system performance. In addition, as the complexity of device management tasks increases, how to improve the processing capacity and response speed of the system through performance optimization has become a key factor to improve the efficiency of intelligent device management. This paper will combine theory and practice to discuss the distributed architecture design and performance optimization strategy of intelligent device management, aiming at providing theoretical support and technical guidance for the optimization of intelligent device management system.

Copyright: © 2025 by the authors. This is an Open Access article distributed under the Creative Commons Attribution License (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited (https://creativecommons.org/licenses/by/4.0/).

1 Overview of Smart Device management

With the rapid development of Internet of Things (IoT) technology, the variety and number of smart devices have proliferated, and smart device management has become a key challenge. The intelligent device management system is designed to ensure the normal operation of equipment, condition monitoring, fault diagnosis and data collection through centralized management and control. However, in the context of the increasing number and complexity of devices, traditional centralized architectures face bottlenecks such as insufficient data processing capacity, low management efficiency and poor scalability. As a result, distributed architecture is becoming a mainstream solution.

The functions of intelligent device management include device registration, monitoring, remote control, data collection, fault detection and maintenance. With the diversification and expansion of device types and distribution, the management task is not limited to a single device, but how to efficiently manage a large number of devices in a distributed environment. By distributing data storage, task processing, and device scheduling to multiple nodes, the distributed architecture breaks the single-node bottleneck and improves the response speed, scalability, and reliability of the system. In this way, the distributed architecture effectively supports the efficient management of large-scale smart devices.

2 Application of distributed architecture in smart device management

2.1 Key technologies of distributed architecture

2.1.1 Microservice architecture and smart device management

Microservices architecture is an architectural pattern that breaks down an application into several small, independent, independently deployable services. In smart device management, the microservice architecture separates the various device management functions (such as device registration, condition monitoring, fault diagnosis, data storage, etc.) into different microservices, each of which is responsible for the processing of a specific function, and these services interact with each other through apis. The advantage of the microservices architecture is that it makes the device management system more flexible, efficient, and supports on-demand scaling.

Table 1. Advantages of microservices architecture

advantage	description	application example
Strong decoupling	Each microservice has an independent life cycle, reducing the interdependence between services and facilitating maintenance and upgrading.	Device status monitoring and device control services can be deployed independently for increased flexibility
high scalability	Service instances can be dynamically added according to device management requirements to improve system scalability.	Add device management service instances to handle the increase in the number of devices
Strong fault tolerance	The failure of microservices does not affect the running of other services, ensuring the high availability of the system.	When the device monitoring service fails, the fault diagnosis service can still run independently
Flexible technology stack	Different services can use different technology stacks to optimize their performance.	The device control service uses Java and the device data analysis uses Python

The device control service uses Java and the device data analysis uses Python：

Assume that there are N microservice nodes in the device management system, the load of each service node S_i is Li, and the total load is L_{tota}. To achieve load balancing, the goal is to evenly distribute the total load L_{tota} among N service nodes. The load balancing of the system can be described by the following formula:

$$L_{total} = \sum_{i=1}^{N} L_i \tag{1}$$

In order to ensure optimal system performance, the load on each node should be balanced as much as possible. The optimization goal is to equalize the load of each service node:

$$Li = \frac{L_{totai}}{N} \tag{2}$$

This optimization goal ensures that each service node in the system distributes tasks evenly, improves resource utilization, and avoids overburdened nodes causing system bottlenecks.

2.1.2 Distributed database and data consistency guarantee

The intelligent device management system needs to process a large amount of device data, including device status, operation logs, and fault records. By storing data on multiple nodes, distributed database avoids the bottleneck problem of traditional single-node database in data processing, and ensures the high availability and scalability of the system. In order to maintain data consistency in a distributed environment, consistency protocols (such as Paxos and Raft) are used to ensure data synchronization among multiple nodes.

Table 2. Key technologies of distributed database

Technical name	describe	application example
Distributed data storage	Data is distributed across multiple nodes, increasing storage capacity and avoiding a single point of failure.	Use Cassandra to store device data for high availability and fast queries
Data fragmentation and replication	Data fragmentation allows data to be divided into multiple parts and stored on different nodes. The replication mechanism ensures data reliability.	The HBase fragmentation technology is used to store device information to increase the system processing capability
CAP theorem and consistency guarantee	According to the CAP theorem, distributed databases trade off between consistency, availability, and partition tolerance, using consistency protocols to ensure data consistency.	The Raft protocol ensures data consistency in distributed systems, especially when device data is synchronized
distributed transaction management	Use two-phase commit protocol (2PC) or three-phase commit protocol (3PC) to ensure consistency across nodes.	Manage device data updates across multiple database nodes to maintain transactional consistency

Consistency modeling analysis of distributed database:

Consider a distributed database system in which N nodes are responsible for storing device state

information. The write delay of node S_i is Di, and the total delay of the whole system is D_{total}, which is the largest delay among the nodes. In order to ensure data consistency, the delay will affect the data synchronization speed during write operations in the system. The total delay can be calculated by the following formula:

$$D_{total} = \max(D_1, D_2, ..., D_n) \tag{3}$$

In order to improve the performance and consistency of the system Di, the system needs to optimize the write delay and adopt the Raft protocol to ensure the atomicity and consistency of the write operation.

2.2 Security and reliability of distributed architecture

In distributed architecture, intelligent device management system is faced with the challenge of data security, privacy protection and system reliability. Distributed architecture allows data and tasks to be distributed among multiple nodes, which also increases the risk of a system being attacked or a node failing. In order to ensure the high availability and security of the system, it is necessary to adopt multi-layer protection measures and fault tolerant design.

2.2.1 Data security and privacy protection

*In the intelligent device management system, sensitive data such as device operation records, user information and control instructions need to be protected. Data encryption, authentication and access control are important measures to ensure data securi*ty.

Table 3. safety engineering

safety engineering	describe	application example
data encryption	Device data is encrypted using encryption algorithms to ensure the confidentiality of data transmission and storage.	The AES encryption algorithm is used to encrypt device status data for storage and transmission
Authentication and authorization	Use multi-factor authentication to ensure that the device management system allows only authorized users to access data.	Authenticate and authorize the device management system through OAuth 2.0 protocol
access control policy	Control access to device data based on user roles and permissions to prevent unauthorized users from accessing sensitive information.	Design access policy based on RBAC (Role Access Control) to ensure data security

2.2.2 Node Failure and System Fault tolerance Design

In the distributed architecture, node faults may cause some system functions to become unavailable, affecting the stability of device management. In order to ensure system reliability, distributed system usually adopts node redundancy, failover and automatic recovery mechanism.

Table 4. Node Failure and System Fault-tolerance Design

fault tolerant technique	describe	application example
Node redundancy and backup	Redundant data storage is implemented among multiple nodes to ensure that the system can continue to run when a node fails.	The master/slave replication mechanism is used to synchronize device data among multiple nodes to ensure high availability of the system
Failover and automatic recovery	When a node fails, the system automatically switches to the standby node and restores services, reducing system downtime.	In the device management system, the HA (High Availability) architecture is used to automatically switch over faulty nodes
Heartbeat detection and health check	Periodically check the health status of nodes and quickly replace faulty nodes to ensure the stable running of the system.	The heartbeat detection technology monitors the health status of each node in the device management system in real time

Fault tolerance and reliability modeling analysis:

Assume that the smart device management system adopts the master-slave replication mechanism, there is N node in the system, and the failure rate of each node is f_i . The reliability of the system R_{system} can be calculated by the following formula:

$$R_{system} = 1 - \prod_{i=1}^{N} (1 - f_i) \tag{4}$$

R_{system} indicates the reliability of the system. The formula shows that the reliability of the system is closely related to the node failure rate f_i , and the overall reliability of the system decreases with the increase of the node failure rate. The fault tolerance and availability of the system can be improved through the configuration of redundant nodes and fault tolerance design.

3 Performance optimization policy

3.1 Data compression and transmission protocol optimization

In the intelligent device management system, the network bandwidth of devices is often a limited resource, and the data generated by a large number of devices needs to be transmitted efficiently. The optimization of data compression and transmission protocol can effectively reduce network load, increase transmission speed and reduce latency. Common data compression techniques include symmetric cryptographic compression, lossless compression algorithms (such as gzip, LZ4, etc.), and semantically based compression methods (such as compression algorithms designed for the Internet of Things).

Data compression technology and optimization modeling:

Suppose that there are N devices in the system and the amount of data generated by device i is Di , then the total amount of data D_{totai} is:

$$D_{\text{totai}} = \sum_{i=1}^{N} Di \tag{5}$$

For compression in the process of data transmission, a lossless compression algorithm C is used to make the compression ratio of the data r, then the compressed data volume Dcompressed can be expressed as:

$$D_{compressed} = \frac{D_{total}}{\text{r}} \tag{6}$$

Assuming the compression ratio r is 2, the amount of data transmitted will be halved after data compression. The reduced data volume can effectively reduce the transmission network load, especially for the transmission of a large number of device data, and improve the response time of the system.

Transport protocol optimization:

In addition to data compression, choosing the right transport protocol is also the key to optimizing transmission performance. Although the traditional HTTP protocol is simple and easy to use, it is not suitable for highly concurrent device data transmission in terms of transmission efficiency and bandwidth consumption. Therefore, in smart device management, lightweight protocols such as MQTT and CoAP are widely adopted. Compared with the HTTP protocol, MQTT uses the publish/subscribe mechanism and is based on TCP/IP, which can reduce the redundant overhead of data transmission and improve the transmission efficiency.

By comparing the transmission delay and bandwidth usage of different protocols, the optimized transmission performance can be calculated by the following formula:

$$\text{Bandwidth Efficiency} = \frac{\text{Payload Size}}{\text{Total Data Sent}} \tag{7}$$

Payload Size is the size of valid Data, and Total Data Sent is the total amount of data transmitted. By optimizing the protocol, you can significantly improve bandwidth utilization and reduce invalid data transmission.

3.2 Distributed Storage System Optimization

With the explosion of device management data, the traditional centralized storage system cannot meet the requirements of high concurrency and massive data storage. The distributed storage system not only increases the storage capacity, but also improves the data access speed and fault tolerance by distributing data to multiple nodes. Distributed storage optimization is mainly reflected in the following aspects: data fragmentation, data redundancy, cache mechanism and data retrieval optimization.

Data fragmentation and redundancy

In a distributed storage system, data fragmentation divides a large data set into multiple small data blocks and distributes them to different nodes. Each node stores only part of the data. The data redundancy mechanism ensures that data is backed up on multiple nodes, providing higher fault tolerance and data recovery capability. Assume that there are N devices in the device management system, and the amount of data generated by each device is D_i. The fragmentation mechanism optimizes data storage by allocating data blocks to N_{shaed} nodes:

$$D_{\text{shard}} = \frac{D_{total}}{N_{shaed}} \tag{8}$$

The redundancy mechanism can ensure the backup of data through the number of copies R. When a node fails, the system can restore data from the backup node to avoid data loss.

Cache mechanism and data retrieval optimization:

The cache mechanism greatly speeds up data access by storing frequently accessed data in memory and reducing disk IO operations. Assuming that the data request volume in the system is Q, the cache hit rate is H, and the cache response time is T_{cache}, the cache acceleration effect can be expressed as:

$$T_{optimized} = T_{orginal} \times (1-H) + T_{cache} \times H \tag{9}$$

By optimizing the cache policy, the response speed can be significantly improved, especially for highly concurrent device management requests.

Data retrieval and index optimization:

The efficiency of data retrieval and query is very important in large-scale distributed storage system. Using efficient indexing mechanisms (such as B+ trees, inverted indexing, etc.) can significantly improve the retrieval speed. Assume that the system has N devices, and each device has M log records. By using indexes, the query time can be reduced from $O(M)$ to $O(\log M)$, which greatly improves the query efficiency.

3.3 Asynchronous computing and parallel processing technology

In the intelligent equipment management system, asynchronous computing and parallel processing technology can effectively improve the system's computing power and processing efficiency. By dividing computing tasks into small tasks and assigning them to multiple computing nodes, cluster resources can be fully utilized to improve computing efficiency. Especially in big data analysis, equipment status monitoring and other scenarios, asynchronous computing and parallel processing technology has a wide range of application prospects.

Asynchronous computing technology:

Asynchronous computing avoids the blocking of the main thread by delegating time-consuming computing tasks to the background thread, thus improving the response speed of the system. In the device management system, asynchronous computing is usually used to deal with device status change, fault detection and other tasks. Assuming that the device management system has N device status monitoring tasks, and the processing time of each task is t_1, the optimization effect of asynchronous computing can be analyzed by the following formula:

$$T_{async} = \max(t_1, t_2, \ldots t_N) \tag{10}$$

Compared with synchronous computing, its maximum processing time is no longer the sum of all tasks, but the most time-consuming one among all tasks, which significantly improves the concurrent processing capability of the system.

Parallel processing technique:

Parallel processing technology greatly improves the efficiency of data processing by dividing the computing task into several small tasks and assigning them to different computing nodes for simultaneous computation. For big data analysis tasks in the device management system, the data can be divided into multiple small blocks, calculated separately, and the results are combined. Assuming that there are N computing nodes in the system, and the amount of data processed by

each node is D_i, the total processing time $T_{parallel}$ after parallel processing can be expressed by the following formula:

$$T_{parallel} = \frac{T_{total}}{N} \tag{11}$$

Parallel processing significantly reduces overall computing time by maximizing the utilization of system resources, especially in the case of large amounts of data, providing timely feedback and responses.

Practical modeling of parallel computing:

Suppose that the device management system needs to process M device log data, the data amount of each device is D_i, and the processing capacity of each node is C, then the total time required for parallel computation $T_{parallel}$ can be expressed as:

$$T_{parallel} = \frac{M \times D_i}{C \times N} \tag{12}$$

Where M is the number of devices, D_i is the log data amount of each device, C is the computing capability of each node, and N is the number of parallel computing nodes. By increasing the number of compute nodes N, the total data processing time can be greatly reduced and the response efficiency of the system can be improved.

Conclusion: With the continuous progress of technology, new problems and challenges facing smart device management are also emerging. For example, how to achieve more efficient data consistency assurance in large-scale distributed systems and how to achieve finer performance optimization in complex equipment environments still need further research. Future research directions will focus on how to combine emerging technologies, such as artificial intelligence, edge computing and 5G technology, to further improve the intelligence level and performance of intelligent device management systems. Overall, the combination of distributed architecture and performance optimization technology will provide a more flexible, scalable and efficient solution for smart device management, and promote the development of smart device management systems in a more efficient and intelligent direction.

References

[1] *Gupta S, Patel N, Kumar A, et al. Intelligent resource optimization for scalable and energy-efficient heterogeneous IoT devices [J]. Multimedia Tools & Applications, 2024, 83(35):85-88.*

[2] *Hernandez N, Almeida F, Blanco V. Optimizing convolutional neural networks for IoT devices: performance and energy efficiency of quantization techniques [J]. Journal of supercomputing, 2024(9):80:88-92.*

[3] *Abu Khurma R, Braik M, Alzaqebah A, et al. Advanced RIME architecture for global optimization and feature selection [J]. Journal of Big Data, 2024, 11(1):858-866.*

[4] *Su H, Luo W, Mehdad Y, et al. Llm-friendly knowledge representation for customer support [C]//Proceedings of the 31st International Conference on Computational Linguistics: Industry Track. 2025: 496-504.*

[5] *K. Zhang, "Optimization and Performance Analysis of Personalized Sequence Recommendation Algorithm Based on Knowledge Graph and Long Short Term Memory Network," 2025 2nd International Conference on Intelligent Algorithms for Computational*

Intelligence Systems (IACIS), Hassan, India, 2025, pp. 1-6, doi: 10.1109/IACIS65746.2025.11211298.

[6] *Y. Zhao, "Design and Financial Risk Control Application of Credit Scoring Card Model Based on XGBoost and CatBoost," 2025 International Conference on Intelligent Communication Networks and Computational Techniques (ICICNCT), Bidar, India, 2025, pp. 1-5, doi: 10.1109/ICICNCT66124.2025.11233033.*

[7] *B. Li, "Research on the Spatial Durbin Model Based on Big Data and Machine Learning for Predicting and Evaluating the Carbon Reduction Potential of Clean Energy," 2025 International Conference on Intelligent Communication Networks and Computational Techniques (ICICNCT), Bidar, India, 2025, pp. 1-5, doi: 10.1109/ICICNCT66124.2025.11232698.*

[8] *Q. Xu, "Implementation of Intelligent Chatbot Model for Social Media Based on the Combination of Retrieval and Generation," 2025 2nd International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS), Hassan, India, 2025, pp. 1-7, doi: 10.1109/IACIS65746.2025.11210989.*

[9] *Y. Zou, "Research on the Construction and Optimization Algorithm of Cybersecurity Knowledge Graphs Combining Open Information Extraction with Graph Convolutional Networks," 2025 2nd International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS), Hassan, India, 2025, pp. 1-5, doi: 10.1109/IACIS65746.2025.11211353.*

[10] *M. Zhang, "Research on Joint Optimization Algorithm for Image Enhancement and Denoising Based on the Combination of Deep Learning and Variational Models," 2025 International Conference on Intelligent Communication Networks and Computational Techniques (ICICNCT), Bidar, India, 2025, pp. 1-5, doi: 10.1109/ICICNCT66124.2025.11232800.*

[11] *W. Han, "Using Spark Streaming Technology to Drive the Real-Time Construction and Improvement of the Credit Rating System for Financial Customers," 2025 International Conference on Intelligent Communication Networks and Computational Techniques (ICICNCT), Bidar, India, 2025, pp. 1-6, doi: 10.1109/ICICNCT66124.2025.11232932.*

[12] *J. Huang, "Research on Multi-Model Fusion Machine Learning Demand Intelligent Forecasting System in Cloud Computing Environment," 2025 2nd International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS), Hassan, India, 2025, pp. 1-7, doi: 10.1109/IACIS65746.2025.11210946.*

[13] *J. Huang, "Performance Evaluation Index System and Engineering Best Practice of Production-Level Time Series Machine Learning System," 2025 International Conference on Intelligent Communication Networks and Computational Techniques (ICICNCT), Bidar, India,*

[14] *X. Liu, "Research on User Preference Modeling and Dynamic Evolution Based on Multimodal Sequence Data," 2025 2nd International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS), Hassan, India, 2025, pp. 1-7, doi: 10.1109/IACIS65746.2025.11211273.*

[15] *F. Liu, "Architecture and Algorithm Optimization of Realtime User Behavior Analysis System for Ecommerce Based on Distributed Stream Computing," 2025 International Conference on Intelligent Communication Networks and Computational Techniques (ICICNCT), Bidar, India, 2025, pp. 1-8, doi: 10.1109/ICICNCT66124.2025.11232744.*

[16] *Wang, Y. (2025). Intervention Research and Optimization Strategies for Neuromuscular Function Degeneration in the Context of Aging. Journal of Computer, Signal, and System Research, 2(7), 14-24.*

[17] *Shen, D. (2025). Construction and Optimization Of AI-Based Real-Time Clinical Decision Support System. Journal of Computer, Signal, and System Research, 2(7), 7-13.*

[18] Hu, Q. (2025). The Practice and Challenges of Tax Technology Optimization in the Government Tax System. Financial Economics Insights, 2(1), 118-124.

[19] Sheng, C. (2025). Analysis of the Application of Fintech in Corporate Financial Decision-Making and Its Development Prospects. Financial Economics Insights, 2(1), 125-130.

[20] Wei, X. (2025). Deployment of Natural Language Processing Technology as a Service and Front-End Visualization. International Journal of Engineering Advances, 2(3), 117-123.