Design of Data Synchronization Scheme based on theAI^{2}CS Cloud Platform

Yu-Ching Lin
National Center for High-Performance Computing, National Applied Research Laboratories, Taiwan

Abstract: *With the rapid development of artificial intelligence (AI) and Internet of Things (IoT) technologies, AIoT (Artificial Intelligence of Things) has developed into an important application technology in the fields of industry, people's livelihood, and medical treatment in recent years. This study uses the AI^{2}CS (AI and IoT Cloud Service) Platform of the National Center for High-Performance Computing as the AIoT application services such as data management, intelligent analysis, and data visualization. The AI^{2}CS platform is used to effectively manage equipment and data and improve the efficiency of the system development process. At the same time, the Kafka real-time data streaming protocol is also integrated into the AI^{2}CS platform to handle data transmission of various IoT devices, and a data synchronization transmission method based on the Kafka streaming protocol is proposed to solve the problems of time series data transmission delay and data asynchrony question. Finally, use the data visualization module of the AI^{2}CS platform to establish various AIoT application services.*

Keywords: *AIoT, AI^{2}CS, Apache Kafka, Data Synchronization*

INTRODUCTION

Over the past decade, the number of Internet of Things (IoT) devices has grown rapidly, from medical devices and homes to industrial automation. Devices such as wearables, sensors, appliances, and medical monitors are all connected and capable of collecting and sharing vast amounts of data. Artificial intelligence (AI) logically enables IoT to function better. IoT endpoints can have built-in intelligence, allowing them to not only collect and share data but also analyze, learn and make decisions.

AIoT combines AI and IoT to create intelligent devices that learn from the data generated and make autonomous decisions. The IoT technology is enabling edge intelligence and can significantly reduce the needs and costs associated with cloud-based analytics, and AI technology is expected to help the IoT realize its greater potential. AIoT can improve the operational efficiency of IoT and enhance data analysis capabilities[1,2].

TheAI^{2}CS platform of NCHC is applied as the AIoT service platform for system development. The AI^{2}CS platform is developed and integrated by NCHC (National Center for High-performance Computing Center, Taiwan) and Advantech's WISE-PaaS industrial IoT platform,[9] which is combined with TWCC (Taiwan Computing Cloud)[10] based on the high-speed computing resources of the cloud service platform. The AI^{2}CS is a cloud service platform that can be used in various industrial IoT applications, massive data analysis, and industry-academia research. The main goal of AI^{2}CS platform construction is to integrate edge computing and provide services of data collection, data analysis, and data visualization from edge perception and devices to the cloud.

In addition, with the development of cloud computing and AIoT, the amount of data has increased dramatically. How to efficiently transfer and analyze data to generate value has become a hot issue. Especially the transmission and analysis of time series data with immediacy requirements. Data communication is the most important step before data analysis and application. The AI^{2}CS platform supports many message communication protocols such as AMQP, MQTT, and CoAP. Apache Kafka is also integrated into the AI^{2}CS platform because of its high output and low latency characteristics [3,4]. The data streaming service developed based on Apache Kafka is used to process the transmission of high-frequency motor data and a data synchronization method is proposed to solve the asynchronous transmission problem in this intelligent analysis system.

THE ARCHITECTURE OF AI^{2}CS PLATFORM

The AI^{2}CS platform provides data collection, AI analysis, and visualization software services from edge perception and devices to the cloud. And it provides a complete development environment for maintaining and operating AIoT services efficiently. The AI^{2}CS platform is responsible for collecting various data of IoT devices and provides a high-performance distributed computing resource, which is responsible for executing the big data analysis and management uploaded by application services. Based on the distributed architecture, the work is performed in parallel to achieve the balance of the computing load, so as to obtain the effective performance of multi-sensor data processing and analysis.

The AI^{2}CS platform is composed of multiple REST ful API function modules, and its architecture is shown in Figure 1. Its structure from bottom to top includes a three-layer structure of infrastructure management, service and data platform, and IoT industrial application software. The edge service of the AI^{2}CS platform can centrally manage ground-end and edge-end devices, provide fast device access, and actively push and broadcast alarms when abnormal accidents occur, minimizing unexpected downtime or abnormal risks. Through the equipment asset management service to establish equipment templates and management configurations, it can be quickly expanded to multiple different production lines or factories in industrial applications, and event alarms and reports can be set for each piece of equipment. Each production line and each factory area can help users quickly grasp and optimize the performance of a large number of equipment.

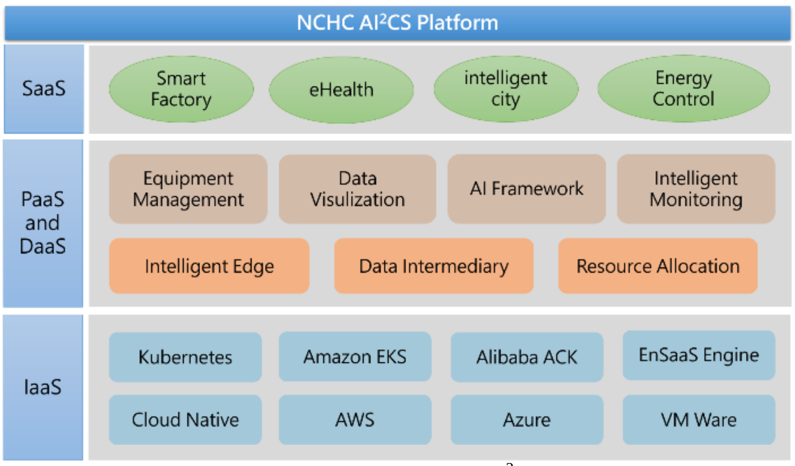

Figure 1: The architecture of the AI^{2}CS platform

The data processing service of the AI^{2}CS platform provides data collection, data aggregation, data monitoring, and abnormal alarm functions. It can continuously collect, process, and store a large amount of time-series data generated by various devices and systems. Users can access real-time data or historical data by RESTful APIs. The data processing service supports data aggregation functions to configure rules flexibly according to the scenario and provides a rule engine for data extraction, transformation, and filtering through SQL-like syntax. And it supports million-level tag management and millisecond-level data collection frequency to support various IoT applications [5,6].

The AI^{2}CS platform provides database storage services such as PostgreSQL, MySQL, InfluxDB, and MongoDB, focusing on scalability and compliance with standards. It does not require self-deployment and maintenance and has functions such as elastic expansion, data monitoring, backup, and restoration. The data storage service is closely integrated with the AI^{2}CS platform, providing a convenient operation interface through the service management system and quickly integrating with the applications on the AI^{2}CS platform.

The visualization service of the AI^{2}CS platform includes various dashboard components developed based on Grafana's open source data visualization technology. The display information can be dynamically switched according to variables such as data sources, tags, and Time-Range, and users can define alert rules and notification conditions. It can present a variety of visualization services for business intelligence categories. Through the dashboard monitoring interface, managers can clearly view the usage status of various devices. The 3D visualization tool can convert 2D static graphics into 3D models and flow charts, which are close to the real application scene. And at the same time keep in touch with the edge data source, which is convenient for users to monitor the parameter changes of various equipment in the actual field, and display real-time information and remote control dynamically. Another set of 3D visualization tools is also provided on the AI^{2}CS platform, which displays real-time dynamic information and remote control. Important scenes can be easily redrawn through a highly intuitive and user-friendly interface. This 3D visualization tool is based on HTML5 Canvas technology, supports standard JavaScript language, and performs logical analysis and processing on object properties, helping to enhance data value and operational efficiency.

The AI framework service of the AI^{2}CS platform is an artificial intelligence development service that provides resource management to pool computing resources and offers services for maximizing, managing, and monitoring computing resource utilization to improve the operating efficiency of AI platforms. It also integrates a variety of machine learning function libraries and provides a complete AI life cycle development service, which simplifies the workflow of AI modeling and offers tools to facilitate data processing, batch modeling, and remote batch deployment. The AI framework service supports various AI applications such as object detection, Image classification, abnormal detection, and equipment diagnosis. Through the AI framework service, online AI model development and training can be done, and the model can be deployed to the edge device with the inference engine. At the same time, combined with the model life cycle management service, retraining can be automatically started when necessary, and the model efficiency can be continuously improved.

In addition, the AI^{2}CS platform also provides development tools based on digital twin technology. Through simple assembly and configuration operations, edge devices can be connected to the cloud for unified management, and the digital twin models of devices can be established and displayed. The digital twin model adopts the configuration operation to perform component combination, model mounting, twin model design, and device topology management. It also can quickly bind the template to the upper-layer application to realize rapid field development or system integration.

THE DATA SYNCHRONIZAYION METHODDESIGN

The AI^{2}CS platform supports many message communication protocols such as AMQP, MQTT, CoAP, and Apache Kafka. The data streaming service is developed based on Apache Kafka's architecture to transfer data on this intelligent analysis system. Apache Kafka is an open-source distributed messaging platform designed for processing real-time streaming data for distributed stream process. Its architecture is essentially a large-scale publish/subscribe message queue according to the distributed transaction log file architecture. The data streaming service is a broker-based solution that operates by keeping streams of data as records in a cluster of servers. Data can thus be assigned to different partitions and topics. Within a partition, these messages are indexed and stored together with timestamps. Other itineraries called consumers can query information from the partition [7,8]. The data streaming service executes on a cluster of one or more servers, and partitions can be distributed across cluster nodes.

The data streaming service is a distributed message system based on Zookeeper coordination and management. The data streaming service can be executed on a data streaming cluster of one or more servers, and partitions can be distributed across cluster nodes. The producers can be deployed on any data streaming server to send messages on one or more topics to the cluster. The data can be distributed to different partitions under the self-defined topic. Consumers can be deployed on a data streaming server or other devices to receive messages for one or more topics from a partition. A consumer can be a group, and messages on the same topic can only be read by one consumer in the same consumer group, but multiple consumer groups can read messages on the same topic at the same time. Each broker of the data streaming server receives messages from producers and writes them to disk while responding to data requests from consumers. Multiple brokers can be connected together to form a cluster. There is a dynamically established controller in the cluster, which is responsible for allocating partitions and monitoring the status of the brokers [9.10].

Most of the current solutions use the message queue protocol to synchronize the data of the remote data server for the purpose of synchronizing data servers or backup data [11.12,13].There is a lack of good solutions tothe immediacy of the receiver and the synchronization of the data source time [14]. Therefore, this study proposes a data synchronization method based on the architecture of Apache Kafka to solve the above problems.

The architecture of the data streaming service is shown in Figure 2. A synchronous control module is established in the data streaming broker to synchronize the time of the data transmitted by each producer and provide time-consistent data to consumers. The synchronous control module creates a Synchronous Topic in the data streaming broker, and stores the acquired real-time synchronization data obtained in each Topic and Partition, which provides the data for synchronizing storage and analysis to each Consumer.

When the data transmission frequency is too fast, there may be a problem that the data time of the transmission and the receiving end are not synchronized due to factors such as network delay or disk I/O. As a result, when the Producer sends the N_{th} data, the Consumer is only receiving the M_{th} data, where $M<N$. When the amount of data is larger or the time is longer, the situation that the data on both sides are out of synchronization will be more serious, where $M<<N$. When there are multiple device sources and the data transfer times are not synchronized, there may be a device without data at the same point in time. As a result, the receiver cannot store the data in the database synchronously, and cannot analyze the asynchronous data.

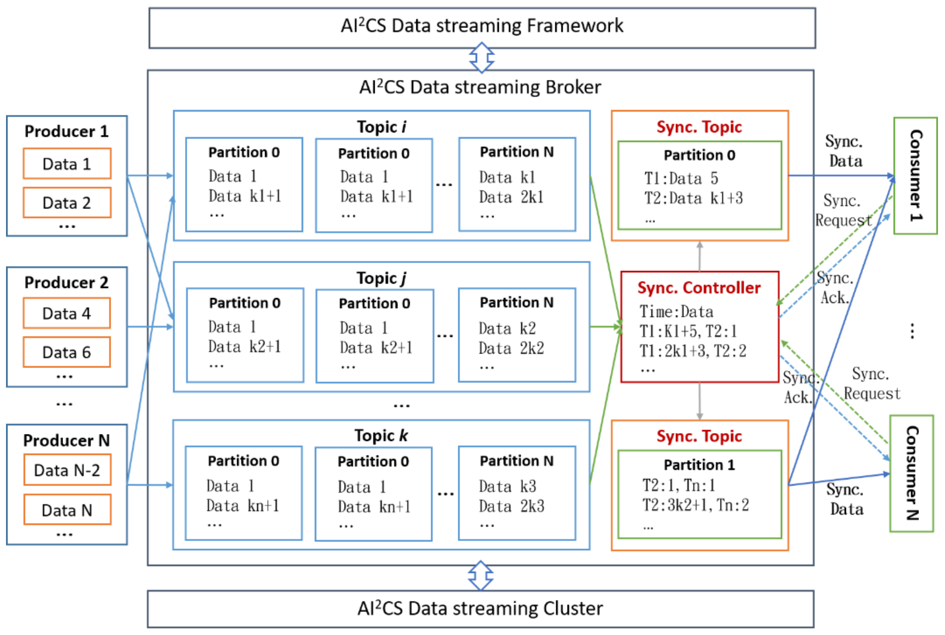

Figure 2: The architecture of data streaming service

The data synchronization method utilizes the feature that subscription topics in the data streaming broker have multiple partitions to classify time series data according to the time-frequency or characteristics of transmission and assign them to different topics and partitions for transmission. And then create a synchronous control module and a message synchronization Topic, and obtain time synchronization data based on time units to ensure data synchronization. The receiving end then aggregates the data according to the characteristics of the data to facilitate the storage and analysis of the data.

The workflow of data synchronization for the data streaming service is shown in Figure 3 and the processes are described as follows:

1. Classify the data by category and time frequency and distribute it to the designated Topic. If there exist multiple data sources, send these data to the specified Topic through multiple Producers according to the category or characteristics of the data. For example, the data of temperature sensors with 0.1-second transmission frequency are sent to Topic 1, the data of vibration sensors with 0.01-second transmission frequency is sent to Topic 2, etc.
2. In order to reduce the time delay caused by the accumulation of a large amount of data in a single Partition, the transmitted data is grouped and transmitted to multiple Partitions, and multiple data can be transmitted in parallel, increasing the overall Throughput and reducing the overall data delay time. For example, each Producer sets a hash value k according to thenetwork latency degree of data transmission and creates k Partitions in the Topic to be transmitted.
3. Set the sequence offset value d of the data to be transmitted by each Topic, which represents the number and order of the data. And take the remainder value x after hashing the k value, where $x=d\%k$. And then distribute the data to the corresponding Partition x. The timestamp, offset ID, and value of data are recorded in each Partition, and the data is written to each Partition in order by time and offset ID.
4. Create a synchronous control module and a synchronous Topic in Data Streaming Broker, and record the transmission timestamp T and the offset value d of data.
5. Consumer sends a message request to the synchronous control moduleto read the synchronous data. The synchronous control module gets the newest data from each Partition of all Topics and compares the timestamp of each piece of data according to the data with a common smallest time unit. And then group and store these data in the corresponding Partition of Synchronous Topic.
6. The synchronous control module responds to the synchronous Ack message and informs the Consumer that it can get data from the Synchronous Topic.
7. Each Consumer starts to get the time-synchronized data corresponding to the Synchronous Topic according to the data characteristics and timestamps.
8. The Consumer synchronously stores the acquired data according to the category of data or analysis requirements in the corresponding database or program in the AI^{2}CS platform.

Through the data synchronization method, when a large amount of high-frequency data is transmitted, the data receiving end can receive more real-time information from the transmitting end. If the time of the data source is not synchronized, it can also obtain synchronized data for storage and analysis by applying the data synchronization method.

CONCLUSION

This study uses the AI^{2}CS cloud platform to provide AIoT data management, smart analysis, data visualization and other application services. It uses the AI^{2}CS platform to effectively manage equipment and data and improve the efficiency of the system development process. The AI^{2}CS platform provides real-time processing, analysis and visualization services of data. At the same time, this research also proposes a data synchronization method based on the Apache Kafka architecture to solve the problems of data transmission delay and time asynchrony. Using the AI^{2}CS platform, various IoT data and analysis results can be displayed in real time on mobile devices, and digital twin models can be established to provide users with real and virtual application services, improve system development efficiency and provide a more convenient management mechanism.

ACKNOWLEDGEMENTS

This research was funded by National Center for High-Performance Computing, National Applied Research Laboratories and the Ministry of Science and Technology in Taiwan.

REFERENCES

[1] P. Zhang, R. P. Chen, et al. An AIoT-based system for real-time monitoring of tunnel construction. Tunnelling and Underground Space Technology 2020; 109: 1-12.

[2] C. S. Lee, Y. L. Tsai, et al. AI-FML agent for robotic game of Go and AIoT real-world co-learning applications. World Congress on Computational Intelligence (IEEE WCCI) 2020; 19-24.

[3] Djenouri, Y., Belhadi, A., Srivastava, G., Houssein, E. H., & Lin, J. C. W. "Sensor data fusion for the industrial artificial intelligence of things," Expert Systems, 39(5), e12875, 2022.

[4] T. P. Raptis, A. Passarella, and M. Conti, "Distributed data access in industrial edge networks," IEEE Journal on Selected Areas in Communications, vol. 38, no. 5, pp. 915–927, 2020.

[5] B. Leang, et al. Improvement of Kafka Streaming Using Partition and Multi-Threading in Big Data Environment. In: Sensors 2019; 134.

[6] B. R. Hiraman, C. V. M., et al. A Study of Apache Kafka in Big Data Stream Processing. ICICET 2018, pp.1-3.

[7] X.-C. Chai et al., "Research on a distributed processing model based on kafka for large-scale seismic waveform data," IEEE Access, vol. 8, pp. 39 971–39 981, 2020.

[8] H. Kim et al., "Message latency-based load shedding mechanism in apache kafka," in Euro-Par 2019: Parallel Processing Workshops, U. Schwardmann et al., Eds. Cham: Springer International Publishing, 2020, pp. 731–736.

[9] J. Xu, J. Yin, H. Zhu, and L. Xiao, "Modeling and verifying producer consumer communication in kafka using csp," in 7th Conference on the Engineering of Computer Based Systems, ser. ECBS 2021. New York, NY, USA: Association for Computing Machinery, 2021.

[10] M. Tsenos, N. Zacheilas, and V. Kalogeraki, "Dynamic rate control in the kafka system," in 24th Pan-Hellenic Conference on Informatics, ser. PCI 2020, 2020, pp. 96–98.

[11] Bennett, T.; Gans, N.; Jafari, R. Data-Driven Synchronization for Internet-of-Things Systems. ACM Trans. Embed. Comput. Syst. 2017, 16, 69.

[12] González-Zapata, A.M., et al. Synchronization of chaotic artificial neurons and its application to secure image transmission under MQTT for IoT protocol. Journal of Nonlinear Dynamics 2021, pp.4581-4600.

[13] J. Chen, Q. Wu and H. Wu, Key Technologies of Multi-active data synchronization for Dispatch and Control System. Asia-Pacific Conference on Image Processing, Electronics and Computers (IPEC) 2020, pp.177-183.

[14] Lin, Z.; Zhang, L. Data synchronization algorithm for IoT gateway and platform. In Proceedings of he 2nd IEEE International Conference on Computer and Communications (ICCC), Chengdu, China,4–17 October 2016; pp. 114–119.