Multi-PaaS Oriented IoT Dual Middleware Model in 5G Era

Shuai Huang¹, Shejiao Hu¹, Zhengfeng Hou¹, Zhao Niu¹ and Lin Wu²
¹Department of Computer and information, Hefei university of technology, 485 Danxia road, Shushan district, Hefei city, Anhui province, China.
²Anhui Wanhong Electric company limited by shares, 39 Jincheng road, Linquan county, Fuyang city, Anhui province, China.
Email: shuai.huang.iot@mail.hfut.edu.cn

Abstract. The PaaS platform based on cloud computing technology provides a large number of professional and flexible services in the fields of intelligent voice and computer vision. The large-scale application of 5G technology enables the software and hardware of the Internet of Things to use the PaaS service to improve the intelligence level of the system and reduce development costs. However, the problem of service invocation and service selection in the process of combining the IoT system with multiple PaaS platforms directly affects the interaction between the IoT perception layer and the application layer. This paper proposes a 5G multi-PaaS-oriented IoT dual middleware model. The unified communication protocol DMCP is designed. DMCP implements unified parsing of messages based on variable-length data items, which reduces the protocol conversion between the perception layer terminal device and multiple PaaS platforms. Complexity and high communication efficiency. Designed the PaaS service selection algorithm DMPS, DMPS comprehensively according to the equipment demand preferences and the overall performance of PaaS services, to achieve accurate service selection in the multi-PaaS platform at the application layer. Finally, examples and comparative tests verify the feasibility and effectiveness of the proposed model, protocol and algorithm.

1. Introduction
Absolutely, the PaaS platform based on cloud computing technology is increasingly rich in functions, providing a large number of professional and flexible services in the fields of intelligent speech, computer vision, machine learning and so on. The development of 5G large connection and low latency technology makes it possible for IoT terminal devices to call PaaS services. PaaS service dating can increase the intelligence level of the IoT system and reduce system development and deployment costs, but it also brings new challenges to the interaction between the perception layer and the application layer.

In order to realize the interconnection between the perception layer and the application layer, IoT middleware has been widely studied and applied[1]. Based on the wearable device communication mechanism, Ngu et al. Designed an IoT middleware that connects smart watches and alcohol prediction systems[2]. White et al. Proposed middleware based on JavaEE architecture and EPL event processing technology, which has high processing performance when processing a large number of terminal device accesses[3]. Zheng Shuquan and others proposed middleware centered on network configuration, which realized the separation of receiving device configuration and processing logic through XML files, and improved the scalability of device receiving [4]. Deng Yihua et al. Proposed middleware embedded in RFID readers to implement data processing logic through ALE_Reporter, which enhanced the reader’s communication capabilities[5]. Wang Bing and others proposed middleware based on the message management mechanism, which improves the message processing
performance of the middleware system through the device reception standardization and multi-level cache processing mechanism[6]. Through the development of IoT as a service, some scholars have also conducted research on IoT middleware for service-oriented architecture (SOA). Chen Haiming et al. Designed a service-oriented IoT software architecture[7], and based on this, the service layer middleware is used to centrally manage the entity layer service and application layer service to achieve the collaborative work of the perception layer and the application layer[8]. Xu Y et al. Proposed an IoT cloud sensor architecture based on the Open Services Gateway Protocol (OSGI) and designed an extensible IoT service middleware atlas[9]. The atlas contains device drivers, gateway communication, and cloud connectivity. Can achieve the dynamic access of different sensor objects[10]. Aleksandar et al. Designed an IoT middleware based on the publish / subscribe mechanism in the crowd perception system, which can promote and manage sensor-aware services[11].

The above-mentioned middleware research is carried out in a transparent parallel connection of data transmission, that is, the terminal device of the perception layer sends data directly to the application layer, which mainly solves the problems of communication mechanism, device access, and device service management. However, in a 5G multi-PaaS IoT system, data transmission is no longer transparent, and device data needs to be processed by multiple PaaS platforms before being sent to the application layer. How the perception layer invokes multiple PaaS services and how the application layer makes service selection among multiple PaaS platforms has become a key issue that needs to be resolved when the IoT system is combined with multiple PaaS platforms.

Aiming at the above problems, this paper proposes a dual middleware model of the Internet of Things for 5G multi-PaaS. The terminal-side middleware design a unified communication protocol based on variable-length data items to implement the protocol conversion between terminal equipment and multiple PaaS platforms Methods. An application-side middleware has designed a PaaS service selection algorithm, which comprehensively considers the selection of equipment requirements and the overall performance of PaaS services to achieve the correct selection of multiple PaaS services.

2. Multi-PaaS Oriented Dual Middleware Model for IoT

2.1. Main Architecture

With the large-scale application of 5G technology in IoT systems, IoT systems can invoke a variety of specialized services provided by the PaaS platform, such as data visualization, video analysis, speech synthesis, and semantic analysis. Compared with the traditional IoT system, the 5G multi-PaaS IoT system is much more complicated in terms of information interaction:

- There are multiple PaaS platforms between the sensing device and the application layer. Application layer processing, but after the PaaS processing, and then transmitted to the application layer;
- terminal device protocols are diverse, and different PaaS platform access protocols are different, when terminal devices call PaaS services, they face a variety of Protocol conversion issues;
- There are a large number of PaaS services with similar functions. These services may be distributed across multiple PaaS platforms. The system needs to consider the application scenario requirements comprehensively and select appropriate PaaS services for terminal devices.

This article designs a MPO-DM, a dual middleware model for the Internet of Things, as shown in figure 1. The model includes two parts, the terminal-side middleware HSM and the application-side middleware ASM.
2.2. Device-Side Middleware
Because the terminal equipment of the perception layer does not have a unified data format, and there are huge differences in the protocols of different PaaS platforms, the terminal equipment cannot directly call the services provided by the PaaS platform.

The HSM communicates with the terminal equipment using a unified communication protocol. After receiving various field data uploaded from the terminal equipment, the HSM first parses the field data packets and then performs protocol conversion according to different PaaS platforms to implement the terminal equipment's call to different PaaS services.

2.3. Application-Side Middleware
Since there are a large number of PaaS services distributed in different platforms in the 5G IoT system, the system needs to comprehensively evaluate these PaaS services according to the service requests of the terminal devices, and select appropriate services for the terminal devices. This paper designs an application-side middleware ASM between the PaaS platform and the application layer to achieve service selection in multiple PaaS platforms.

The realization of service selection through ASM has the following advantages:
- ASM runs on the server side, and can comprehensively consider the service performance of each PaaS platform to find a suitable PaaS service for the terminal device;
- ASM uses service selection and service providers. The caller's separation reduces the coupling between the terminal device and the PaaS platform, which facilitates system expansion.

2.4. Multi-PaaS Service Interaction Mechanism
The multi-PaaS service interaction mechanism of MPO-DM is shown in figure 2. It mainly includes service selection and service invocation.

Figure 1. MPO-DM Architecture

2.2. Device-Side Middleware
Because the terminal equipment of the perception layer does not have a unified data format, and there are huge differences in the protocols of different PaaS platforms, the terminal equipment cannot directly call the services provided by the PaaS platform.

The HSM communicates with the terminal equipment using a unified communication protocol. After receiving various field data uploaded from the terminal equipment, the HSM first parses the field data packets and then performs protocol conversion according to different PaaS platforms to implement the terminal equipment's call to different PaaS services.

2.3. Application-Side Middleware
Since there are a large number of PaaS services distributed in different platforms in the 5G IoT system, the system needs to comprehensively evaluate these PaaS services according to the service requests of the terminal devices, and select appropriate services for the terminal devices. This paper designs an application-side middleware ASM between the PaaS platform and the application layer to achieve service selection in multiple PaaS platforms.

The realization of service selection through ASM has the following advantages:
- ASM runs on the server side, and can comprehensively consider the service performance of each PaaS platform to find a suitable PaaS service for the terminal device;
- ASM uses service selection and service providers. The caller's separation reduces the coupling between the terminal device and the PaaS platform, which facilitates system expansion.

2.4. Multi-PaaS Service Interaction Mechanism
The multi-PaaS service interaction mechanism of MPO-DM is shown in figure 2. It mainly includes service selection and service invocation.
The process of selecting services.
1. The terminal device sends the service requirements defined based on its own preferences to HSM;
2. HSM sends the service requirements directly to ASM;
3. After ASM receives the service requirements, it selects the appropriate service in the PaaS service set according to the service selection algorithm;
4. ASM will select the PaaS service information and return it to the HSM;
5. The HSM sends the received service information to the terminal device, and the terminal device uses the service information to make a service call.

The above interaction process needs to solve two key issues:
- How to design a unified communication protocol to achieve efficient conversion of multiple device data and multiple PaaS protocols;
- How to comprehensively consider the needs of terminal equipment and PaaS service performance, and design a Precise PaaS service selection algorithm.

3. Unified Communications Protocol
3.1. Software Requirement Analysis
In the traditional IoT device communication protocol is designed based on fixed-length data items. The protocol message is composed of multiple fixed-length data items. The byte length and offset position of each data item are fixed. When communicating, it is necessary to strictly follow the position and length. Parse the message [12].

Actual IoT systems have a large amount of field data, which may be hundreds or even thousands, and only a small amount of data needs to be transmitted for each communication. If the message format contains all the field data, it will lead to problems such as data redundancy, large communication overhead, and slow protocol parsing speed. To avoid the above problems, the Internet of Things system is usually designed into a variety of different messages according to the function
combination, likes \( \{P_1, P_2, \ldots, P_m\} \). This kind of message transmits a kind of field data. Each message \( P_i \) requires a specific parser, and \( M \) messages require \( M \) parsers.

In a multi-PaaS IoT system, the protocols of different PaaS platforms are very different, and specific protocol conversions need to be performed for each platform. If access to \( N \) PaaS platforms, a total of \( M \times N \) protocol conversions are required. In practical applications, \( M \) and \( N \) are often large, and a large number of protocol conversions make HSM difficult to implement and poor robustness. When a certain PaaS protocol is changed, the corresponding \( M \) types of parsing programs need to be modified.

3.2. Unified Communications Protocol

Aiming at the problem of “M×N protocol conversion” existing when the traditional IoT device communication protocol calls multi-PaaS services, this paper designs a unified communication protocol DMCP between HSM and devices based on “variable-length data items”. As shown in Figure 3, the DMCP protocol message is composed of multiple “variable-length data items”. Each variable-length data item represents a field data, which is defined by a triple (Length, Type, Data), where Length Refers to the length of the data item, Type refers to the data type number agreed by the sending and receiving parties, and Data refers to the value of field data. The number and position of data items in a message are dynamically variable, so the message format is always the same regardless of how many field data are transmitted.

| Data Item 1 | Data Item 2 | Data Item n |
|------------|------------|------------|
| Length     | Type       | Data       |
| Length     | Type       | Data       |
|           |            | ……          |
| Length     | Type       | Data       |

Figure 3. Example of DMCP Protocol messages

The DMCP protocol is parsed sequentially according to the triples of the variable-length data items. The analysis process is shown in algorithm 1. Initialize the field data type number table and the data to be parsed, then parse the length, type, and value of the data items in order, and finally output all field data.

Algorithm 1 DMCP Protocol Analysis Algorithm

Input: \( \text{MsgArrays} = \{b_1, b_2, \ldots, b_n\} \) // Unparsed message
Output: ItemList // Parsing results

1. INIT TypeMap = \{ \{type : params\} \}
2. for \( i = 0 \) in \([0,N]\)
3. \( \text{item_length} = \text{convetBytesToInt}(\text{MsgArrays}[i]) \)
4. \( \text{itemArrays} = \text{MsgArrays}[i,i + \text{item}(\text{length})] \)
5. \( \text{item_type} = \text{convetBytesToInt}(\text{itemArrays}[1:1 + 2]) \)
6. \( \text{item_data} = \text{itemArrays}[3:] \)
7. \( \text{item_params} = \text{TypeMap}[	ext{item(type)}] \)
8. Add item to ItemList
9. \( i = i + \text{item_length} \)
10. endfor
11. return ItemList

The DMCP protocol uses a unified parsing program for any variable-length data item. Therefore, HSM uses the DMCP protocol to communicate with terminal devices with the following advantages:

- The complexity of protocol conversion is low. The DMCP protocol uses one message type to dynamically transmit multiple data items. If access to \( N \) PaaS platforms, HSM only needs \( 1 \times N \) protocol conversions, which is earlier than traditional device communication protocols and simplifies \( M \times N \) protocols conversion. It is \( 1 \times N \), which greatly reduces the complexity of
protocol conversion when HSM invokes multiple PaaS platforms.

- The protocol is robust. When adding a PaaS platform, HSM only needs to add one conversion program, without modifying the analysis program.
- High communication efficiency. Communication efficiency is the proportion of effective information in the communication scale. When the terminal equipment adopts the DMCP protocol, field data is transmitted on demand, and it is not necessary to transmit placeholder data for a fixed message format.

4. PaaS Service Selection Algorithm

In the IoT service middleware, the sensing and control functions provided by the sensing layer devices are abstracted as IoT services, and the selection of IoT services has been a research hotspot in academia. Qiao Xiuquan et al. Proposed an IoT service selection method based on event-driven model, and completed the dynamic response of the service through device sensing information events[13]; Li Lingyu et al. Proposed a service selection method based on semantic model, which semantically annotated device services. Later selection[14]; Chen et al. Proposed establishing a regression model from historical data of device service calls for service selection[15]; Jia et al. Proposed a multi-stage centralized service selection algorithm in semantic matching, using hierarchical filtering Service selection[16]; Guo et al. Proposed a service selection method that can achieve load balancing of service providers[17].

The above service selection methods are based on device-side sensing and control services, rather than PaaS services. The attributes of the sensing and control services are relatively stable. Static modeling is often performed with fixed values. Service selection is often based on the functional attributes of service providers. In a multi-PaaS IoT system, when ASM chooses a PaaS service, it cannot just focus on its functional attributes. It also needs to comprehensively consider various non-functional attributes (ie, QoS), such as delay, accuracy, and cost of the PaaS service, and the terminal. The application scenario requirements of the device. Therefore, this paper designs a PaaS service selection algorithm based on device QoS preferences.

4.1. PaaS Service Request with QoS Preference

When a terminal device submits a PaaS service request to ASM, it must specify its specific requirements for the service, including functional requirements and non-functional requirements. Among them, functional requirements refer to the basic requirements that PaaS services need to meet, such as service type, operation type, and invocation method; non-functional requirements are expectations of PaaS service QoS, such as invocation delay time, function execution time, response time, Cost of use, security, availability, and interoperability.

In a 5G multi-PaaS IoT system, terminal devices have a preference for the QoS attributes of PaaS services according to different application environments. For example, the cost attributes of image processing services for civil security video surveillance equipment are higher than the delay attributes, while fire video The monitoring device's preference for the delay attribute of the image processing service is much higher than the cost attribute. In the service selection, ASM should consider how to better meet the non-functional requirements of the device preference attributes while ensuring the functional requirements of the device.

The PaaS service request model \( HSM_{\text{request}} \) of the terminal device includes three parts of functional requirements, non-functional requirements and QoS preference attributes, which are defined as:

\[
HSM_{\text{request}} = \{BM_{\text{req}}, QM_{\text{req}}, QM_{\text{prefer}}\}
\]  

Among them, \( BM_{\text{req}} = \{b_1, b_2, \ldots, b_m\} \) refers to the requirements of the functional attributes of the target PaaS service; \( QM_{\text{req}} = \{r_1, r_2, \ldots, r_n\} \) refers to the expectations of the QoS attributes of the target PaaS service; \( QM_{\text{prefer}} \subseteq QM_{\text{req}} \) refers to the terminal device's preference attribute set, which can include multiple preference attributes.
4.2. Multi-PaaS Service Normalization

Multi-PaaS service normalization realizes unified quantification of different attributes of a PaaS service set by abstractly modeling a single PaaS service. Service normalization is the premise of ASM for service selection.

4.2.1. PaaS service model. PaaS services are distributed in different PaaS platforms. Their functional attributes are relatively stable, but non-functional attributes (that is, QoS attributes) are affected by network conditions, loads, and so on. Considering that the QoS attribute is closely related to the time factor, a PaaS service model \( S_i \) is defined as:

\[
S_i = \begin{cases} 
\text{BaseModel} = (BP_1, ..., BP_i, ..., BP_n) \\
\text{QoSModel} = (DT, QM(t))
\end{cases}
\]

\( DT = [T_s, T_r + T] \)

\( QM(t) = \{f_{q_1}(t), ..., f_{q_j}(t), ..., f_{q_m}(t)\} \)

\( t \in [T_s + kT, T_s + (k+1)T], k \in N \)

Among them, BaseModel refers to the functional attributes of PaaS services, including multiple parameters, such as service type, operation type, and invocation method. QoSModel refers to the non-functional attributes of the PaaS service, \( DT = [T_s, T_r + T] \) is the effective time period of the QoS attributes of the PaaS service, \( T_s \) is the start time, and \( T \) is the period for updating the QoS attributes. QM(t) refers to the QoS attribute set, and \( f_{q_j}(t) \) refers to the QoS attribute function that changes with time in DT.

4.2.2. Multi-PaaS service normalization. PaaS service attribute parameter change range and utility direction are inconsistent, simple methods such as calculating the average value are meaningless. Among service sets with similar functions, non-functional attributes (QoS) have become the standard for differentiating different services[18], so this paper normalizes the non-functional attributes of PaaS services.

There is a service set \( S = \{S_1, S_2, ..., S_n\} \) composed of n PaaS services, and all QoS attributes of \( S \) at a time are expressed as:

\[
S = \begin{bmatrix} 
q_{11} & \cdots & q_{1m} \\
\vdots & \ddots & \vdots \\
q_{n1} & \cdots & q_{nm}
\end{bmatrix}
\]

(3)

\( q_{ij} \) is the j-th QoS attribute value of \( S_i \).

The QoS attributes of the PaaS service can be divided into positive correlation attributes and negative correlation attributes according to the direction of action. Positive correlation attributes refer to larger values, such as security attributes; negative correlation attributes refer to smaller values, such as usage cost attributes. If \( q_{ij} \) is a positive correlation property, calculate the normalized value \( q'_{ij} \) according to formula (4):

\[
q'_{ij} = \begin{cases} 
\frac{q_{ij} - \text{min}(q_{ij})}{\text{range}(q_{ij})}, \text{range}(q_{ij}) \neq 0 \\
1, \text{range}(q_{ij}) = 0
\end{cases}
\]

(4)

If \( q_{ij} \) is a negative correlation property, it is calculated according to formula (5):
Among them, \( \text{range}(q_i) = \max(q_i) - \min(q_i) \).

After normalizing all the services in the set according to the above calculation method, \( S \) is obtained, and any QoS attribute \( q_{ij} \) in \( S \) is distributed between 0 and 1 and has a positive correlation.

4.3. PaaS Service Selection Algorithm Based on Device Preference

When ASM selects PaaS services, it needs to comprehensively consider the functional and non-functional attributes of the service. Among them, the functional attributes must meet the equipment requirements, and the service selection is relatively simple. Non-functional attributes are relatively difficult to choose because they meet the needs of the device as much as possible, and the device has preferences for different QoS attribute requirements. This paper first defines the comprehensive utility value of PaaS service selection. Based on this, a dual middleware PaaS service selection algorithm (DMPS) based on device preference is designed.

4.3.1. PaaS service selection comprehensive utility value.

The PaaS service selection comprehensive utility value is a quantification of the degree of matching of PaaS service selection results. It is determined by the QoS preference attribute utility value and the QoS overall attribute utility value.

The utility value of the QoS preference attribute is defined as \( U_{\text{prefer}}(i) \), and \( U_{\text{prefer}}(i) \) represents the degree to which \( S_i \) satisfies the \( \text{HSM}_{\text{req}} \) specified preference attribute in \( \text{QM}_{\text{req}} \), and \( U_{\text{prefer}}(i) \) is calculated using formula (7). Among them, \( r_j \) is the normalized value of the \( \text{QM}_{\text{req}} \) demand value for the QoS attribute \( j \). The requirement for the preference attribute can be satisfied, so the maximum value of the utility value of the preference attribute is 1.

\[
U_{\text{prefer}}(i) = \begin{cases} 
q_{ij} / r_j & q_{ij} < r_j \\
1 & q_{ij} \geq r_j 
\end{cases}
\]

The utility value of the overall QoS attribute is defined as \( U_{\text{entire}} \), \( U_{\text{entire}} \) represents the comprehensive satisfaction degree of \( S_i \) for all QoS attribute requirements of \( \text{QM}_{\text{req}} \) in \( \text{HSM}_{\text{req}} \), and is calculated using formula (8).

\[
U_{\text{entire}} = \frac{\text{S}_i / \text{QM}_{\text{req}}}{\sum_{j=1}^{k} (q_{ij} / r_j)}
\]

The comprehensive utility value of service selection is defined as \( AA \). AA comprehensively considers the effect of preference attribute utility value and overall attribute utility value on the evaluation of service choice results, and is calculated using formula (9).

\[
U(i) = \sum (\alpha_i \times U_{\text{prefer}}(i)) + \beta \times U_{\text{entire}}, \quad \sum \alpha_i + \beta = 1
\]
preference attribute utility values and overall attribute utility values on the comprehensive utility value.

4.3.2. Utility evaluation strategy. According to the QoS preference of the terminal device, setting the values of $\alpha_i$ and $\beta$ forms different utility evaluation strategies ES. First define the degree of preference as $PreferLevel(i)$, calculated as follows:

$$PreferLevel(i) = \begin{cases} 
    \text{High}, & r_j \in S_{q_j}, [0, 20\%] \\
    \text{Medium}, & r_j \in S_{q_j}, [20\%, 80\%] \\
    \text{Low}, & r_j \in S_{q_j}, [80\%, 100\%] 
\end{cases}$$

(10)

Among them, $r_j \in S_{q_j}, [0, 20\%]$ indicates that the value of $r_j$ is within the maximum 20\% of the value set of the QoS attribute $j$ in $S$.

- If $PreferLevel(i)$ is High, the attribute needs to be satisfied first when service selection, and the $\beta$ in the utility evaluation strategy is set to 0.2;
- If $PreferLevel(i)$ is Medium, the preference attribute utility value and the overall attribute utility value are considered uniformly when the service is selected, and the $\beta$ in the utility evaluation strategy is set to 0.5;
- If $PreferLevel(i)$ is Low, the service selection takes into account the preference attributes, and also considers the overall attribute matching degree, and sets the $\beta$ in the utility evaluation strategy to 0.8.

The AA in the above utility evaluation strategy has (11) and (12) two calculation methods, which need to be calculated and set according to different stages in service selection.

$$\begin{align*}
    a_i &= 1 - \beta \\
    a_{j(p(i))} &= 0 \\
    \sum a_i &= 1 - \beta \\
    a_i(\text{High}) : a_i(\text{Medium}) : a_i(\text{Low}) &= 3 : 2 : 1
\end{align*}$$

(11) (12)

4.3.3. PaaS service selection algorithm for dual-middleware. The basic idea of the DMPS algorithm is to select the corresponding utility evaluation strategy according to the QoS preference of the equipment on the premise that the functional requirements of the equipment are met as much as possible, and then select the service according to the utility evaluation strategy. In the selection process, if a suitable service cannot be found for the current attribute preference level, then the attribute preference level is reduced and searched again. When multiple services are found according to different preference attributes, the comprehensive utility value is selected for evaluation according to the PaaS service, and finally the most suitable PaaS service is selected. The service selection algorithm is as follows:

Algorithm 2 DMPS algorithm
ASM performs service discovery on the connected PaaS platform, and builds PaaS services combined with $S_{all}$.

1. Filter $S_{all}$ according to $BM_{req}$ to obtain a service set $S$ that meets $BM_{req}$, and establish a service selection result set $S_{res}$.

2. Normalize the $S$ and $QM_{req}$ of $HSM_{req}$ to get $S'$ and $QM_{req}'$.

3. Get $QM_{prefer}'$ from $QM_{prefer}$ of $HSM_{req}$ and $QM_{req}'$.

4. for $i$ in $QM_{prefer}$:
   6. Calculate $PreferLevel(i)$ according to Equation(10).
   7. if $PreferLevel(i) = High$ then $S_{H} = S(q_{i} \geq r_{i})$.
   8. if $S_{H} \neq \emptyset$ then
   9. Set $PreferLevel(i) = Medium$
   10. if $S_{H} \neq \emptyset$ then
       11. Set $\beta$ from $ES_{High}$, Set $\alpha$ from Equation(11)
       12. Add $S_{i}(U_{i} \text{ is maximum})$ to $S_{res}$
   13. if $PreferLevel(i) = Medium$ then
       14. Set $\beta$ from $ES_{Medium}$, Set $\alpha$ from Equation(11)
       15. Add $S_{i}(U_{i} \text{ is maximum})$ to $S_{res}$
   16. if $PreferLevel(i) = Low$ then
       17. Set $\beta$ from $ES_{Low}$, Set $\alpha$ from Equation(11)
       18. Add $S_{i}(U_{i} \text{ is maximum})$ to $S_{res}$
   19. endfor
   20. Set $\beta$ from $ES_{High}$, Set $\alpha$ from Equation(12)
   21. $S_{result} = U_{result} \text{ is maximum in } S_{res}$
   22. return $S_{result}$

5. Experiments and Key Performance Tests
The purpose of the experiment is to verify the feasibility of the MPO-DM model, test the DMCP protocol, and test the accuracy of the PaaS service selection algorithm based on device QoS preferences.

5.1. Experimental Environment
In the context of "Internet + charging infrastructure", the team developed a set of new energy vehicle charging pile management systems[19] to provide users and managers with intelligent management and control services for charging piles. To put into use. This experimental environment is based on this system. The charging pile equipment is a self-developed 5G communication 120KW DC charging pile, as shown in figure 4.
The hardware-side middleware HSM in the dual middleware runs in the embedded Debian 6.3 system, with 1GB of memory, and communicates with the charging pile device using RS232. The application-side middleware ASM runs on Alibaba Cloud server, the operating system is CentOS 6.8, the memory is 4GB, and the charging pile application layer software uses HTTP to communicate.

5.2. MPO-DM Feasibility Verification
In the original charging pile management system, the charging pile equipment directly sends various field data to the application layer for processing, and the application layer provides data management functions for system management. Figure 5 is the charging pile status management interface, and figure 6 is the charging record management interface. It can be found that the data visualization effect of the original system is not good.

Figure 4. 5G DC charging pile

Figure 5. Charging pile status management interface of the original system
In order to further improve the level of intelligent management and control of the system, the original system called data visualization services of different PaaS platforms through MPO-DM. The charging pile device first converts the operating status data through HSM and then sends it to the PaaS platform. The data visualization service of the PaaS platform processes and visualizes the charging pile operation data, and then sends the rendering result to the application layer via ASM.

**Figure 6.** Display part of the data in a curvilinear way

**Figure 7.** Charging pile status management interface after calling PaaS service
There are two types of data visualization PaaS services called by the system: monitoring services and information summary services. The visual effects after the system calls the PaaS service are shown in figure 7 and figure 8, respectively. Comparing figures 5 and 6, it is found that the charging pile management system greatly improves the level of data visualization after calling the PaaS service through MPO-DM.

5.3. DMCP Protocol Test
In the new energy vehicle charging pile management system, there are 48 types of communication data between the charging pile equipment and the background, which are mainly divided into six categories: charging control, status information, charging information, automotive BMS information, and charging history, record type and billing information type, a total of 1580 field data. Based on the above-mentioned field data, this section tests the conversion complexity and communication efficiency of the DMCP protocol.

5.3.1. Protocol conversion complexity. For the six major functions of the charging pile, three PaaS platforms were selected for the experiment: OneNet, ThingJS, and Alibaba Cloud IoT. The DMCP protocol and the TDC protocol in [12] were used for testing. HSM uses the protocols required by different protocols The number of conversions is shown in figure 9. Obviously, the number of protocol conversions required by the DMCP protocol is much smaller than that of traditional device communication protocols, and the protocol conversion complexity is low.
5.3.2. Communication efficiency. The experiment takes the charging pile status information in the status information class as an example, and compares the communication efficiency between the DMCP protocol and the traditional device communication protocol.

In the actual application environment, the number of valid parameters of the charging pile status information transmission usually does not exceed 30% of the total. Therefore, within 25 parameters, the experiment is based on the number of different parameters during the transmission of state information. In the second test, the data transmission efficiency results of the two protocols are shown in figure 10. The data transmission efficiency of the DMCP protocol is basically stable and always higher than that of the traditional device communication protocol.

![Figure 9](image)

**Figure 9.** Numbers of protocol conversions required by HSM to use different protocols

![Figure 10](image)

**Figure 10.** Numbers of protocol conversions required by HSM to use different protocols
5.4. PaaS Service Selection Algorithm Test

The accuracy of the DMPS algorithm was tested in experiments, and the R-S algorithm[20] and M-S algorithm[16] were selected for comparison. Referring to PaaS platforms such as OneNet and ThingJS, a set S with 150 PaaS services is established. HSM issues service selection requests to ASM in the numbers of 200, 400, 600, and 800. After receiving service selection requests, ASM uses different Service selection algorithm. Table 1 shows the accuracy test results of the three selection algorithms.

Table 1. Different algorithm selection accuracy

| Number of requests | Algorithm | 200 | 400 | 600 | 800 |
|-------------------|----------|-----|-----|-----|-----|
|                   | DMPS     | 93.2% | 92.6% | 93.4% | 92.9% |
|                   | M-S      | 88.9% | 87.2% | 86.8% | 86.5% |
|                   | R-S      | 83.3% | 82.1% | 80.9% | 79.4% |

The experimental results show that the DMPS algorithm has the highest accuracy rate, the M-S algorithm takes second place, and the R-S algorithm has the lowest accuracy rate. The MS algorithm starts from the overall QoS attributes of the PaaS service and does not consider the requirements attributes of some devices individually, so the accuracy rate is lower than that of the DMPS algorithm. The RS algorithm only selects based on the single QoS attribute of the PaaS service when selecting services. Regardless of the pros and cons of other attributes, some selection results are not suitable for the terminal device, so the accuracy is the lowest.

6. Conclusions

This article focuses on the research of interaction problems when 5G IoT systems are combined with multiple PaaS platforms. A dual middleware model, MPO-DM, is constructed. The DMCP protocol is designed based on variable-length data items, and the DMPS algorithm is designed according to the preferences of terminal equipment needs. The results of experiments show that:

- The actual IoT charging pile management system uses the MPO-DM model to implement the protocol conversion between the charging pile equipment and different PaaS platforms, complete the call to different data visualization PaaS services, and improve the intelligent management of the system Level;
- Compared with the traditional device communication protocol, HSM uses DMCP protocol to communicate with the device with low protocol conversion complexity and high communication efficiency;
- MPO-DM can perform comprehensive PaaS service selection when the DMPS algorithm can be integrated Considering the terminal equipment demand preferences and the overall performance of PaaS services, the algorithm has a high accuracy rate.

In the subsequent research, the interaction mechanism of the dual middleware model of the Internet of Things will be further improved by combining cloud computing PaaS services and edge computing micro-cloud services.

7. References

[1] Samin A C and Mohamed A. El-Zawawy. 2017 J. Iaasc. 24 pp 1-9
[2] Anne N, Mario G, Vangelis M, Surya N and Quan Z S. 2017 J. IEEE.Iotj. 4 pp 1-20
[3] White S, Alexandre A and David R. 2008 C. Int. conf. on Distributed Event-based Systems
[4] Shuquan Z, Qian W and Zhigang D. 2013 J. Joca. 33 pp 2022-25
[5] Yihua D and Shengli X. 2008 J. Cead. 7 pp 1716-8
[6] Bing W and Tinggui C. 2017 J. Cepap. 16 pp 89-97
[7] Haiming C and Li C. 2016 J. Cjoc. 39 pp 853-71
[8] Haiming C, Hailong S, Meng L and Li C. 2017 J. Cjoc. 40 pp 1725-49
[9] Yi X and Sumi H. 2014 C. Int. Conf. 6th on Cloud Computing Technology and Science
[10] Yi X and Sumi H. 2015 J. IEEE.Iotj. p 1
[11] Aleksandar A, Martina M and Krešimir P. 2016 J. Fgcs. 56 pp 607-22
[12] Kaicheng Z, He H and Liangtu S. J. Ceaa.
[13] Xiuquan, Yang Z, Budan W, Bo C, Shuai Z, Huadong M and Junliang C. 2013 J. Si. 10
[14] Lingyu L, Ning L and Guanyu L. 2016 J. Aroc. 3 pp 802-5
[15] Chen C, Fang W. 2018 C. Int. Conf. 9th. on Software Engineering and Service Science. pp 72-7
[16] Bing J, Wuyungerile L and Tao Z. 2017 C. Int. Conf. on Computational Science and Engineering and IEEE International Conference on Embedded and Ubiquitous Computing
[17] Mengying Guo and Xudong Y. 2018 C. Int. Conf. on Algorithms and Architectures for Parallel Processing. pp 17-27
[18] Zhichun J, Yuan L and Xiang Li. 2019 J. Ceaa. 55 pp 74-8
[19] Qing L and Shejiao H. 2019 J. Mact. 38 pp 61-5
[20] Liangzhao Z, Boualem B, A.H.H and Ngu. 2004 J. IEEE.Tose. 30 pp 311-27