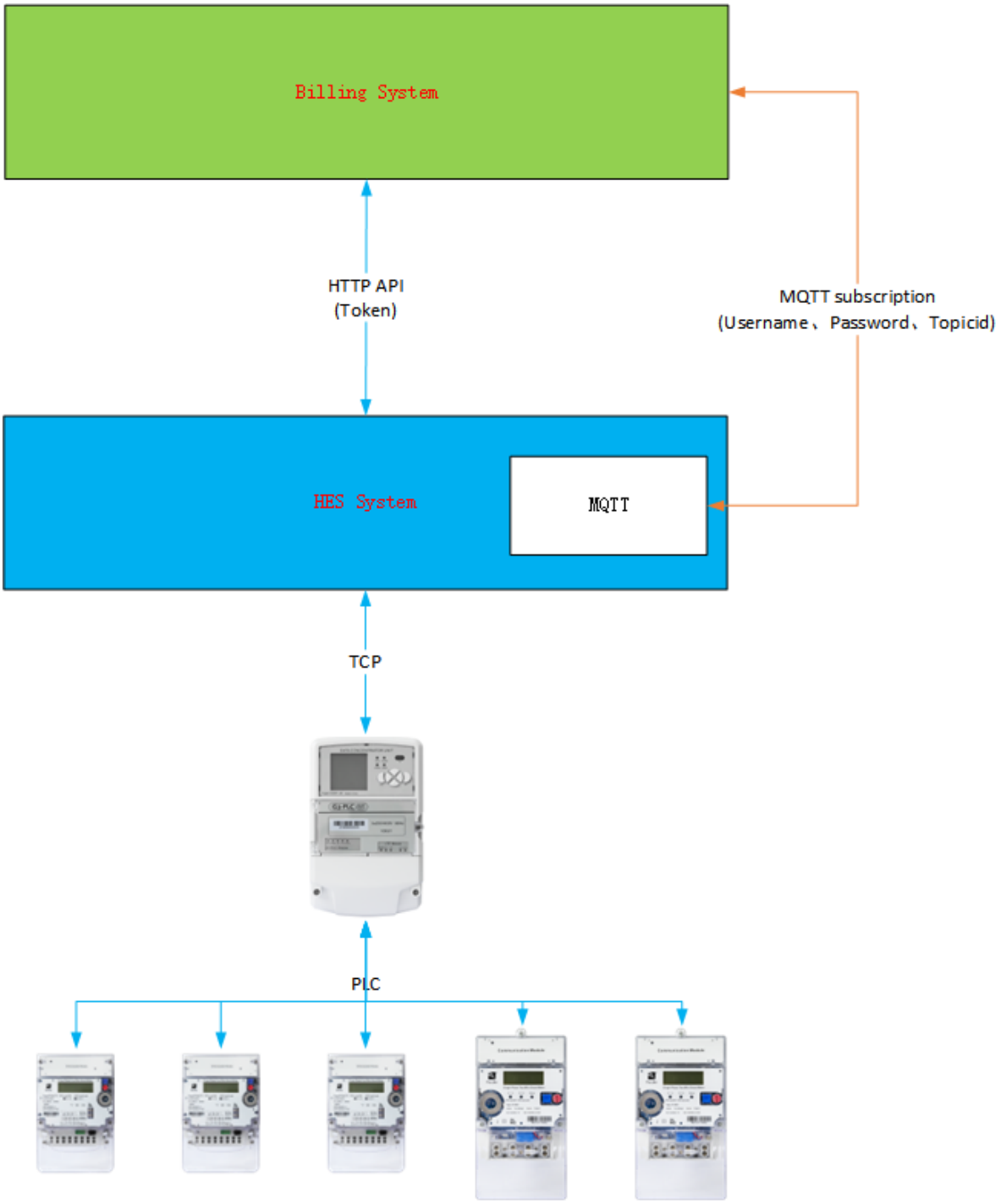
HES API Documentation

Date	Version	Describe	Author
2024-07-12	V1.0	Initial version	caoyangjin
2024-07-26	V1.0	1. The formatting of the document has been optimized. 2. Description explanations have been added to interface requests and returns. 3. Some interaction diagrams and access step flowcharts have been added. 4. Adjusted the token parameter in the API interface from the head header to the body parameters uniformly	caoyangjin hanwan

1. Overview

The document describes the specific usage of the interface.Both parties communicate through synchronous interface requests (HTTP+JSON) and asynchronous message subscriptions (MQTT)



2. Docking process

a). Refer to Figure 1 for the process of periodically obtaining frozen data reading from meters

b). Read from the meter, perform DCU file operations (add meter files, delete meter files, query meter files), and remotely control the meter by opening and closing the switch, as shown in Figure 2

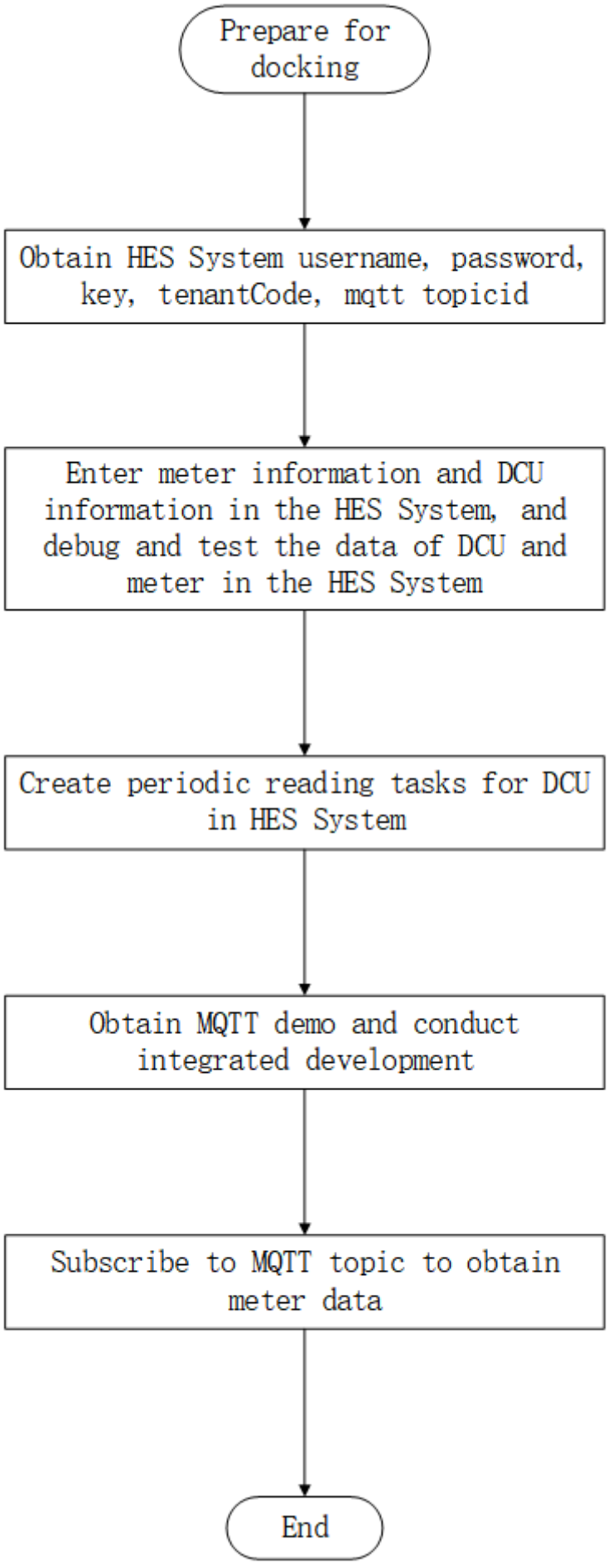

Prepare for docking
Obtain HES System username, password, key, tenantCode, mqtt topicid
Enter meter information and DCU information in the HES System, and debug and test the data of DCU and meter in the HES System
Create periodic reading tasks for DCU in HES System
Obtain MQTT demo and conduct integrated development
Subscribe to MQTT topic to obtain meter data
End

Figure 1

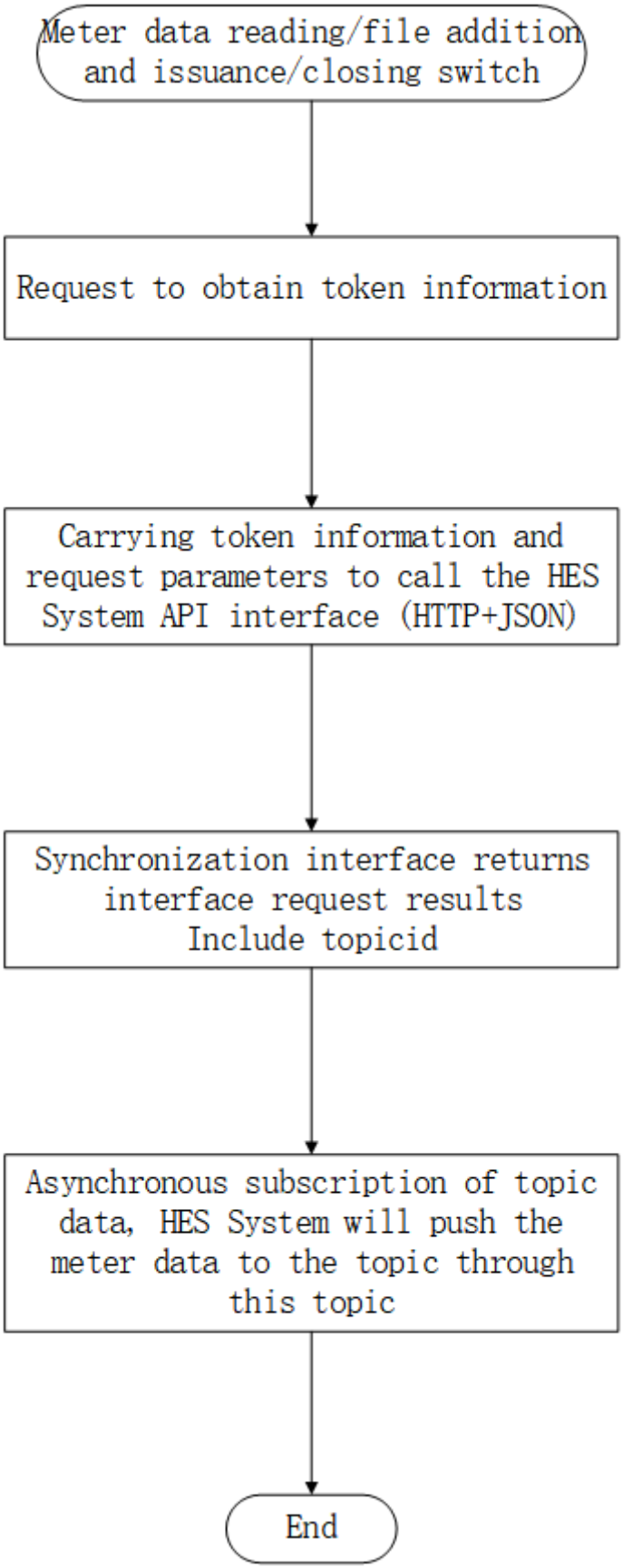

Meter data reading/file addition and issuance/closing switch
Request to obtain token information
Carrying token information and request parameters to call the HES System API interface (HTTP+JSON)
Synchronization interface returns interface request results Include topicid
Asynchronous subscription of topic data, HES System will push the meter data to the topic through this topic
End

Figure 2

3. Business interface

3.1. Obtain TOKEN

To obtain the API credential.

> Instructions
>
> When using the token, please note the following:
>
> - The token is valid for 7200 seconds (2 hours). If obtained repeatedly within the validity period, the same result will be returned and automatically renewed. After expiration, obtaining it again will return a new token.
> - Developers need to cache the token for subsequent interface calls.

Request Method: POST

3.1.1 Body Parameters

Name	Type	Required	Description
tenantCode	String	Yes	Tenant code
username	String	Yes	Username
password	String	Yes	Password
secretKey	String	Yes	Signature verification key information

3.1.2 Return Parameters

Name	Type	Description
code	Integer	Return status code 200=Success 500=Fail
msg	String	Return status description
data	Object	Return data structure
token	String	Generated Token
expireIn	Long	Expiry time in seconds

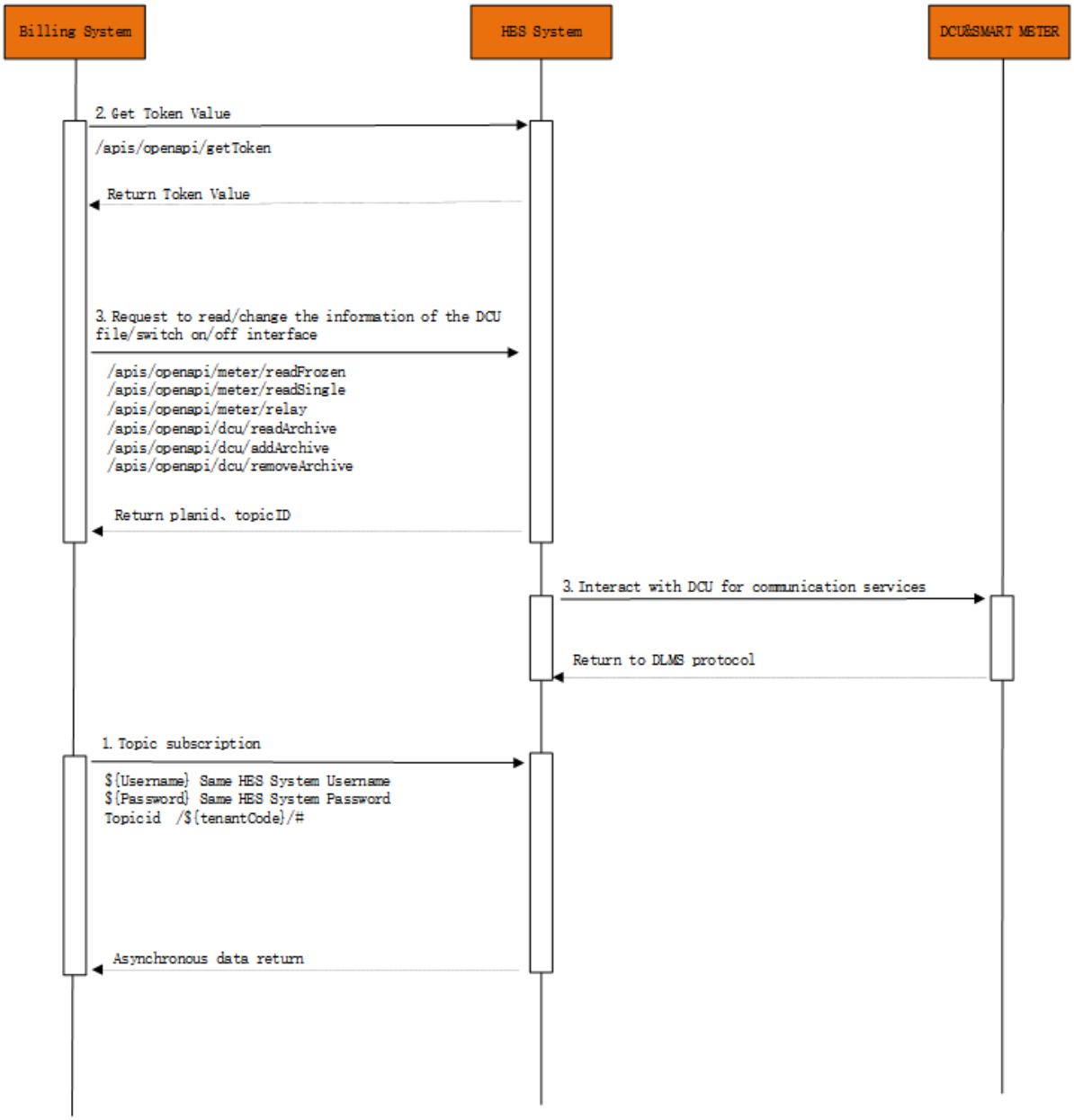
3.1.3 Request Example

```
POST /apis/openapi/getToken HTTP/1.1
Content-Type: application/json;charset=UTF-8
Host: keami.mol-electron.com

{
  "tenantCode" : "1000000067", //Provided by HES
  "username" : "kuangda",  //Provided by HES
  "password" : "kd123456",  //Provided by HES
  "secretKey" : "55be0dc7ac554610165183db43991c50"  //Provided by HES
}
```

3.1.4 Return Example

```
HTTP/1.1 200
Content-Type: application/json;charset=UTF-8

{
    "code": 200,
    "msg": "success",
    "data": {
        "token": "eyJleHAiOjE3MjA1NzUyMTcsInVzZXJuYW1lIjoiYWRtaW5sb3dhbiJ9",
        "expireIn": 7200
    }
}
```

3.2. Real-time Electricity Meter Reading Profile Data(Data Category: Composite)



Initiate a reading from the electricity meter to obtain real-time data. Supported data items are detailed in the **Reading Data Range**. The API returns the result of operation plan creating, meter data will push to MQ with specific topic.

Request Method: POST

3.2.1 Body Parameters

Name	Type	Required	Description
devList	String	Yes	Execution meter numbers, separated by commas
cgcode	String	Yes	Meter reading item code, details in **Reading Data Range**
start	String	Yes	Data start time range, YYYY-MM-DD HH:mm:ss
end	String	Yes	Data end time range, YYYY-MM-DD HH:mm:ss
token	String	Yes	HES interface request signature verification

3.2.2 Return Parameters

Name	Type	Description
code	Integer	Return status code 200=Success 500=Fail
msg	String	Return status description
data	Object	Return data structure
planId	String	Created operation plan ID
topicId	String	MQTT subscription operation result data Topic：/${tenantCode}/planId/#

3.2.3 MQ Push Data

Name	Type	Description
planId	String	Operation plan ID
data	Object	Return data structure
status	String	Reading result status 00=Success 01=Partial Success 02=Failure
list	Array	Reading data list
devAddr	String	Meter number
timestamp	Long	Data timestamp (Instant reading takes the task return time; Composite reading takes the freeze time)
channelData	Array	Channel data array
unit	String	Channel data unit
value	String	Channel data value
cgcode	String	Channel data meter reading item code, details in **Reading Data Range**

3.2.4 Request Example

```
POST /apis/openapi/meter/readFrozen HTTP/1.1
Content-Type: application/json;charset=UTF-8
Host: keami.mol-electron.com

{
  "devList" : "216088078,214670998,216769554,216088076,216088077", //smart meter
devaddrs
  "cgcode" : "2", //Meter reading item code, for example 2===Monthly Frozen Data
  "start" : "2024-06-01 00:00:00",  //Data start time range
  "end" : "2024-07-01 00:00:00", //Data end time range
  "token": "eyJleHAiOjE3MjA1NzUyMTcsInVzZXJuYW1lIjoiYWRtaW5sb3dhbiJ9"
}
```

3.2.5 Return Example

```
HTTP/1.1 200
Content-Type: application/json;charset=UTF-8

{
    "code": 200,
    "msg": "success",
    "data": {
        "planId": "P003_20240711162001113070",
        "topicId": "/1000000067/P003_20240711162001113070"
    }
}
```

3.2.6 MQ Push Example

```
{
    "planId": "P003_20240711162001113070",
    "data": {
        "status" : "00",
        "list" : [
            {
                "devAddr" : "202407100001",
                "timestamp" : 1720540800000,
                "channelData" : [
                    {
                        "unit": "kwh",
                        "value": "7340.740",
                        "cgcode": "5069"
                    },
                    {
                        "unit": "kwh",
                        "value": "0.330",
                        "cgcode": "5074"
                    },
                    {
                        "unit": "kvarh",
                        "value": "1092.290",
                        "cgcode": "5084"
                    },
                    {
```

```
                    "unit": "kvarh",
                    "value": "0.230",
                    "cgcode": "5089"
                },
                {
                    "unit": "kw",
                    "value": "0.619",
                    "cgcode": "5099"
                },
                {
                    "unit": "kw",
                    "value": "0.619",
                    "cgcode": "5100"
                },
                {
                    "unit": "kw",
                    "value": "0.000",
                    "cgcode": "5101"
                },
                {
                    "unit": "kw",
                    "value": "0.000",
                    "cgcode": "5102"
                }
            ]
        },
        {
            "devAddr" : "202407100002",
            "timestamp" : 1720627200000,
            "channelData" : [
                {
                    "unit": "kwh",
                    "value": "7340.740",
                    "cgcode": "5069"
                },
                {
                    "unit": "kwh",
                    "value": "0.330",
                    "cgcode": "5074"
                },
                {
                    "unit": "kvarh",
                    "value": "1092.290",
                    "cgcode": "5084"
                },
                {
                    "unit": "kvarh",
                    "value": "0.230",
                    "cgcode": "5089"
                },
                {
                    "unit": "kw",
                    "value": "0.619",
                    "cgcode": "5099"
                },
                {
                    "unit": "kw",
                    "value": "0.619",
```

```
                        "cgcode": "5100"
                    },
                    {
                        "unit": "kw",
                        "value": "0.000",
                        "cgcode": "5101"
                    },
                    {
                        "unit": "kw",
                        "value": "0.000",
                        "cgcode": "5102"
                    }
                ]
            }
        ]
    }
}
```


3.3. Real-time Electricity Meter Reading Instantaneous Data(Data Category: Single)

Initiate a reading from the electricity meter to obtain real-time data. Supported data items are detailed in the **Reading Data Range**. The API returns the result of operation plan creating, meter data will push to MQ with specific topic.

Request Method POST

3.3.1 Body Parameters

Name	Type	Required	Description
devList	String	Yes	Execution meter numbers, separated by commas
cgcode	String	Yes	Meter reading item code, details in **Reading Data Range**
token	String	Yes	HES interface request signature verification

3.3.2 Return Parameters

Name	Type	Description
code	Integer	Return status code 200=Success 500=Fail
msg	String	Return status description
data	Object	Return data structure
planId	String	Created operation plan ID
topicId	String	MQTT subscription operation result data Topic：/${tenantCode}/planId/#

3.3.3 MQ Push Data

Name	Type	Description
planId	String	Operation plan ID
data	Object	Return data structure
status	String	Reading result status 00=Success 01=Partial Success 02=Failure
list	Array	Reading data list
devAddr	String	Meter number
timestamp	Long	Data timestamp (Instant reading takes the task return time; Composite reading takes the freeze time)
cgcode	String	Channel data meter reading item code, details in **Reading Data Range**
unit	String	Unit
value	String	Data value

3.3.4 Request Example

```
POST /apis/openapi/meter/readSingle HTTP/1.1
Content-Type: application/json;charset=UTF-8
Host: keami.mol-electron.com

{
  "devList" : "202407100001,202407100002",
  "cgcode" : "5069",
  "token": "eyJleHAiOjE3MjA1NzUyMTcsInVzZXJuYW1lIjoiYWRtaW5sb3dhbiJ9"
}
```

3.3.5 Return Example

```
HTTP/1.1 200
Content-Type: application/json;charset=UTF-8

{
    "code": 200,
    "msg": "success",
    "data": {
        "planId": "P003_20240711162001113070",
        "topicId": "/1000000067/P003_20240711162001113070"
    }
}
```

3.3.6 MQ Push Example

```
{
    "planId": "P003_20240711162001113070"
    "data": {
        "status" : "00",
        "list" : [
```

```
            {
                "devAddr" : "202407100001",
                "timestamp" : 1720540800000,
                "cgcode" : "5069",
                "unit": "kwh",
                 "value": "7340.740"
            },
            {
                "devAddr" : "202407100002",
                "timestamp" : 1720540800000,
                "cgcode" : "5069",
                "unit": "kwh",
                 "value": "7340.740"
            }
        ]
    }
}
```

3.4. Obtain Electricity Meter Historical Profile Data(Data Category: Composite)

Retrieve historical data from the system. Supported data items are detailed in the **Reading Data Range**.

Request Method: POST

3.4.1 Body Parameters:

Name	Type	Required	Description
devList	String	Yes	Query meter numbers, separated by commas
cgcode	String	Yes	Meter reading item code, details in **Reading Data Range**
start	String	Yes	Data start time range, YYYY-MM-DD HH:mm:ss (within half a year)
end	String	Yes	Data end time range, YYYY-MM-DD HH:mm:ss (within half a year)
curPage	Integer	Yes	Paging query parameter, current page
pageSize	Integer	Yes	Paging query parameter, amount of data per page
token	String	Yes	HES interface request signature verification

3.4.2 Return Parameters:

Name	Type	Description
code	Integer	Return status code 200=Success 500=Fail
msg	String	Return status description
data	Object	Return data structure
total	Integer	Total amount of data that meets the conditions
curPage	Integer	Paging query parameter, current page
pageSize	Integer	Paging query parameter, amount of data per page
list	Array	Historical data list
devAddr	String	Meter number
timestamp	Long	Data timestamp (Instant reading takes the task return time; Composite reading takes the freeze time)
channelData	Array	Channel data array
unit	String	Channel data unit
value	String	Channel data value
cgcode	String	Channel data meter reading item code, details in **Reading Data Range**

3.4.3 Request Example

```
POST /apis/openapi/meter/historyFrozen HTTP/1.1
Content-Type: application/json;charset=UTF-8
Host: keami.mol-electron.com

{
  "devList" : "202407100001,202407100002",
  "cgcode" : "1",
  "start" : "2024-07-10 00:00:00",
  "end" : "2024-07-10 00:15:00",
  "curPage" : 1,
  "pageSize" : 20,
  "token": "eyJleHAiOjE3MjA1NzUyMTcsInVzZXJuYW1lIjoiYWRtaW5sb3dhbiJ9"
}
```

3.4.4 Return Example

```
HTTP/1.1 200
Content-Type: application/json;charset=UTF-8

{
    "code": 200,
    "msg": "success",
    "data": {
        "total" : 4,
        "curPage" : 1,
        "pageSize" : 20,
```

```
        "list" : [
            {
                "devAddr" : "202407100001",
                "timestamp" : 1720540800000,
                "channelData" : [
                    {
                        "unit": "kWh",
                        "value": "7340.740",
                        "cgcode": "5069"
                    },
                    {
                        "unit": "kWh",
                        "value": "0.330",
                        "cgcode": "5074"
                    },
                    {
                        "unit": "kvarh",
                        "value": "1092.290",
                        "cgcode": "5084"
                    },
                    {
                        "unit": "kvarh",
                        "value": "0.230",
                        "cgcode": "5089"
                    },
                    {
                        "unit": "kW",
                        "value": "0.619",
                        "cgcode": "5099"
                    },
                    {
                        "unit": "kW",
                        "value": "0.619",
                        "cgcode": "5100"
                    },
                    {
                        "unit": "kW",
                        "value": "0.000",
                        "cgcode": "5101"
                    },
                    {
                        "unit": "kW",
                        "value": "0.000",
                        "cgcode": "5102"
                    }
                ]
            },
            {
                "devAddr" : "202407100002",
                "timestamp" : 1720627200000,
                "channelData" : [
                    {
                        "unit": "kWh",
                        "value": "7340.740",
                        "cgcode": "5069"
                    },
                    {
                        "unit": "kWh",
```

```
                        "value": "0.330",
                        "cgcode": "5074"
                    },
                    {
                        "unit": "kvarh",
                        "value": "1092.290",
                        "cgcode": "5084"
                    },
                    {
                        "unit": "kvarh",
                        "value": "0.230",
                        "cgcode": "5089"
                    },
                    {
                        "unit": "kW",
                        "value": "0.619",
                        "cgcode": "5099"
                    },
                    {
                        "unit": "kW",
                        "value": "0.619",
                        "cgcode": "5100"
                    },
                    {
                        "unit": "kW",
                        "value": "0.000",
                        "cgcode": "5101"
                    },
                    {
                        "unit": "kW",
                        "value": "0.000",
                        "cgcode": "5102"
                    }
                ]
            }
        ]
    }
}
```


3.5. Obtain Electricity Meter Historical Instantaneous Data(Data Category: Single)

Retrieve historical data from the system. Supported data items are detailed in the **Reading Data Range**.

Request Method: POST

3.5.1 Body Parameters:

Name	Type	Required	Description
devList	String	Yes	Query meter numbers, separated by commas
cgcode	String	Yes	Meter reading item code, details in **Reading Data Range**
start	String	Yes	Data start time range, YYYY-MM-DD HH:mm:ss (within half a year)
end	String	Yes	Data end time range, YYYY-MM-DD HH:mm:ss (within half a year)
curPage	Integer	Yes	Paging query parameter, current page
pageSize	Integer	Yes	Paging query parameter, amount of data per page
token	String	Yes	HES interface request signature verification

3.5.2 Return Parameters:

Name	Type	Description
code	Integer	Return status code 200=Success 500=Fail
msg	String	Return status description
data	Object	Return data structure
total	Integer	Total amount of data that meets the conditions
curPage	Integer	Paging query parameter, current page
pageSize	Integer	Paging query parameter, amount of data per page
list	Array	Historical data list
devAddr	String	Meter number
timestamp	Long	Data timestamp (Instant reading takes the task return time; Composite reading takes the freeze time)
unit	String	Unit
value	String	Data value

3.5.3 Request Example

```
POST /apis/openapi/meter/historySingle HTTP/1.1
Content-Type: application/json;charset=UTF-8
Host: keami.mol-electron.com

{
  "devList" : "202407100001,202407100002",
  "cgcode" : "5069",
  "start" : "2024-07-10 00:00:00",
  "end" : "2024-07-11 00:00:00",
  "curPage" : 1,
  "pageSize" : 20,
  "token": "eyJleHAiOjE3MjA1NzUyMTcsInVzZXJuYW1lIjoiYWRtaW5sb3dhbiJ9"
}
```

3.5.4 Return Example

```
HTTP/1.1 200
Content-Type: application/json;charset=UTF-8

{
    "code": 200,
    "msg": "success",
    "data": {
        "total" : 2,
        "curPage" : 1,
        "pageSize" : 20,
        "list" : [
            {
                "devAddr" : "202407100001",
                "timestamp" : 1720540800000,
                "unit": "kWh",
                 "value": "7340.740"
            },
            {
                "devAddr" : "202407100001",
                "timestamp" : 1720627200000,
                "unit": "kWh",
                 "value": "7340.740"
            },
            {
                "devAddr" : "202407100002",
                "timestamp" : 1720540800000,
                "unit": "kWh",
                 "value": "7340.740"
            },
            {
                "devAddr" : "202407100002",
                "timestamp" : 1720627200000,
                "unit": "kWh",
                 "value": "7340.740"
            }
        ]
    }
}
```

3.6. Perform Meter Circuit Breaker Operations

Perform circuit breaker operations (open or close) on specified electricity meters. The API returns the result of operation plan creating, operation result data will push to MQ with specific topic.

Request Method: POST

3.6.1 Body Parameters:

Name	Type	Required	Description
devList	String	Yes	Meter numbers to operate, separated by commas
controlMode	Integer	Yes	Execution parameter 1=open circuit breaker, 2=close circuit breaker
token	String	Yes	HES interface request signature verification

3.6.2 Return Parameters:

Name	Type	Description
code	Integer	Return status code 200=Success 500=Fail
msg	String	Return status description
data	Object	Return data structure
planId	String	Created operation plan ID
topic	String	MQTT subscription operation result data Topic： /tenantCode/planId/

3.6.3 MQ Push Data:

Name	Type	Description	Is Returned?
planId	String	Operation plan ID	Yes
data	Object	Return data structure	Yes
status	String	Execution status 00=Success 01=Partial Success 02=Failure	Yes
list	Array	Operation result list	Yes
devAddr	String	Meter number	Yes
successful	Boolean	Execution result true=success false=failure	Yes

3.6.4 Request Example:

```
POST /apis/openapi/meter/relay HTTP/1.1
Content-Type: application/json;charset=UTF-8
Host: keami.mol-electron.com

{
  "devList" : "202407100001,202407100002",
  "controlMode" : 1,
  "token": "eyJleHAiOjE3MjA1NzUyMTcsInVzZXJuYW1lIjoiYWRtaW5sb3dhbiJ9"
}
```

3.6.5 Return Example:

```
HTTP/1.1 200
Content-Type: application/json;charset=UTF-8

{
    "code": 200,
    "msg": "success",
    "data": {
        "planId": "P003_20240711162001113070",
        "topic": "/20240701/P003_20240711162001113070"
    }
}
```

3.6.6 MQ Push Example:

```
{
    "planId": "P003_20240711162001113070",
    "data": {
        "status" : "00",
        "list" : [
            {
                "devAddr" : "202407100001",
                "successful" : true
            },
            {
                "devAddr" : "202407100002",
                "successful" : true
            }
        ]
    }
}
```

3.7. Retrieve Meter Information

Retrieve the meter archive information from the DCU. The API returns the result of operation plan creating, meter information will push to MQ with specific topic.

Request Method: POST

3.7.1 Body Parameters:

Name	Type	Required	Description
dcuNo	String	Yes	DCU SN
token	String	Yes	HES interface request signature verification

3.7.2 Return Parameters:

Name	Type	Description
code	Integer	Return status code 200=Success 500=Fail
msg	String	Return status description
data	Object	Return data structure
planId	String	Created operation plan ID
topic	String	MQTT subscription operation result data Topic： /tenantCode/planId/

3.7.3 MQ Push Data:

Name	Type	Description
planId	String	Operation plan ID
data	Object	Return data structure
status	String	Reading result status 00=Success 02=Failure
list	Array	Meter archive data list
devAddr	String	Meter number
status	String	Archive enable status 0=Disabled 1=Enabled
commType	String	Communication type
moduleNext	String	Upper-level route meter number

3.7.4 Request Example:

```
POST /apis/openapi/dcu/readArchive HTTP/1.1
Content-Type: application/json;charset=UTF-8
Host: keami.mol-electron.com

{
  "dcuNo" : "2211000051",
  "token": "eyJleHAiOjE3MjA1NzUyMTcsInVzZXJuYW1lIjoiYWRtaW5sb3dhbiJ9"
}
```

3.7.5 Return Example:

```
HTTP/1.1 200
Content-Type: application/json;charset=UTF-8

{
    "code": 200,
    "msg": "success",
    "data": {
        "planId": "P003_20240711162001113070",
        "topic": "/20240701/P003_20240711162001113070"
    }
}
```

3.7.6 MQ Push Example:

```
{
    "planId": "P003_20240711162001113070",
    "data": {
        "status": "00",
        "list": [
            {
                "devAddr" : "202407100001",
                "status" : "1",
                "commType" : "1",
                "moduleNext" : "202407100001"
            },
            {
                "devAddr" : "202407100002",
                "status" : "1",
                "commType" : "1",
                "moduleNext" : "202407100001"
            }
        ]
    }
}
```

3.8. Install Meter

Add meter archive information to the DCU. The API returns the result of operation plan creating, operation result data will push to MQ with specific topic.

Request Method: POST

3.8.1 Body Parameters:

Name	Type	Required	Description
dcuNo	String	Yes	Concentrator SN
token	String	Yes	HES interface request signature verification
archives	Array	Yes	Meter archive array
devAddr	String	Yes	Meter number
status	String	Yes	Archive status 0=Disabled 1=Enabled

3.8.2 Return Parameters:

Name	Type	Description
code	Integer	Return status code 200=Success 500=Fail
msg	String	Return status description
data	Object	Return data structure
planId	String	Created operation plan ID
topicId	String	MQTT subscription operation result data Topic：/tenantCode/planId/#

3.8.3 MQ Push Data:

Name	Type	Description
planId	String	Operation plan ID
data	Object	Return data structure
status	String	Execution result status 00=Success 02=Failure

3.8.4 Request Example:

```
POST /apis/openapi/dcu/addArchive HTTP/1.1
Content-Type: application/json;charset=UTF-8
Host: keami.mol-electron.com

{
  "dcuNo" : "2211000051",
  "token": "eyJleHAiOjE3MjA1NzUyMTcsInVzZXJuYW1lIjoiYWRtaW5sb3dhbiJ9",
  "archives" : [
    {
      "devAddr" : "202407100001",
      "status" : "1"
    },
    {
      "devAddr" : "202407100002",
      "status" : "1"
    }
  ]
}
```

3.8.5 Return Example:

```
HTTP/1.1 200
Content-Type: application/json;charset=UTF-8

{
    "code": 200,
    "msg": "success",
    "data": {
        "planId": "P003_20240711162001113070",
        "topic": "/20240701/P003_20240711162001113070"
    }
}
```

3.8.6 MQ Push Example:

```
{
    "planId": "P003_20240711162001113070",
    "data": {
        "status": "00"
    }
}
```

3.9. Remove Meter

Remove meter archive information from the DCU. The API returns the result of operation plan creating, operation result data will push to MQ with specific topic.

Request Method: POST

3.9.1 Body Parameters:

Name	Type	Required	Description
dcuNo	String	Yes	Concentrator SN
token	String	Yes	HES interface request signature verification
archives	Array	Yes	Meter number array
devAddr	String	Yes	Meter number

3.9.2 Return Parameters:

Name	Type	Description
code	Integer	Return status code 200=Success 500=Fail
msg	String	Return status description
data	Object	Return data structure
planId	String	Created operation plan ID
topic	String	MQTT subscription operation result data Topic： /tenantCode/planId/

3.9.3 MQ Push Data:

Name	Type	Description
planId	String	Operation plan ID
data	Object	Return data structure
status	String	Execution result status 00=Success 02=Failure

3.9.4 Request Example:

```
POST /apis/openapi/dcu/removeArchive HTTP/1.1
Content-Type: application/json;charset=UTF-8
Host: keami.mol-electron.com
HES_Access_Token: <Your Access Token Here>

{
  "dcuNo" : "String",
  "archives" : [
      "202407100001",
      "202407100002"
  ]
}
```

3.9.5 Return Example:

```
HTTP/1.1 200
Content-Type: application/json;charset=UTF-8

{
    "code": 200,
    "msg": "success",
    "data": {
        "planId": "P003_20240711162001113070",
        "topic": "/20240701/P003_20240711162001113070"
    }
}
```

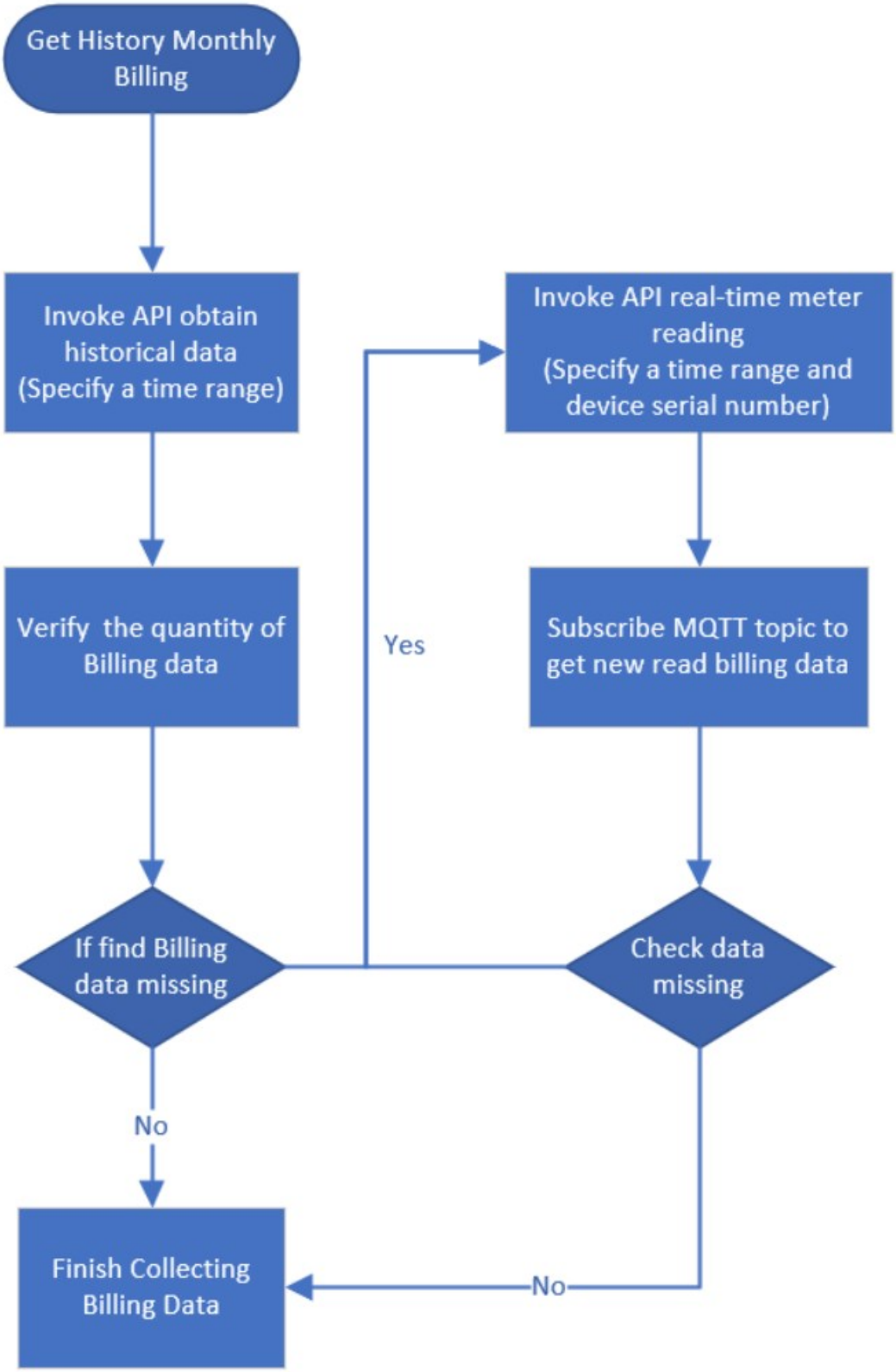
3.9.6 MQ Push Example

```
{
    "planId": "P003_20240711162001113070",
    "data": {
        "status": "00"
    }
}
```

4. Enclosure

4.1 API invoke example

Let's take Monthly Billing data as an example to describe how to invoke APIs. For Monthly Billing data collecting, first invoke API （obtain historical data） to get historical data. Save the return data and check the quantity. If find out data missing invoke API (real-time reading) to get data from meter. Subscribe the MQTT topic to get new read billing data and check data missing. Repeat the operation to make sure all data is collected.



4.2 Reading Data Range

Supported data items:

Data Category	cgcode	Description	Data Name
Composite	1	Daily Frozen Data	dailyBilling
Composite	2	Monthly Frozen Data	monthlyBilling
Composite	3	Load Curve	loadProfile
Single	5053	Phase A Current	currentA
Single	5054	Phase B Current	currentB
Single	5055	Phase C Current	currentC
Single	5050	Phase A Voltage	voltageA
Single	5051	Phase B Voltage	voltageB
Single	5052	Phase C Voltage	voltageC
Single	5022	Total Active Power	activePowerTotal
Single	5023	Phase A Active Power	activePowerA
Single	5024	Phase B Active Power	activePowerB
Single	5025	Phase C Active Power	activePowerC
Single	5030	Total Reactive Power	reactivePowerTotal
Single	5031	Phase A Reactive Power	reactivePowerA
Single	5032	Phase B Reactive Power	reactivePowerB
Single	5033	Phase C Reactive Power	reactivePowerC
Single	5038	Total Apparent Power	apparentPowerTotal
Single	5039	Phase A Apparent Power	apparentPowerA
Single	5040	Phase B Apparent Power	apparentPowerB
Single	5041	Phase C Apparent Power	apparentPowerC
Single	5046	Total Power Factor	powerFactorTotal
Single	5047	Phase A Power Factor	powerFactorA
Single	5048	Phase B Power Factor	powerFactorB
Single	5049	Phase C Power Factor	powerFactorC
Single	5069	Total Forward Active Energy	importActiveEnergy
Single	5074	Total Reverse Active Energy	exportActiveEnergy
Single	5084	Total Forward Reactive Energy	importReactiveEnergy
Single	5089	Total Reverse Reactive Energy	exportReactiveEnergy
Single	5099	Maximum Active Demand	activeMaxDemand

Data Category	cgcode	Description	Data Name
Single	5104	Maximum Reactive Demand	reactiveMaxDemand