ICS 黑体 5 号
中国标准文献分类号

CES

团　　体　　标　　准

T/CES XXX-XXXX

面向电动汽车充电设施服务平台的电网级互动管控支持系统接入规范

Access Specification for Grid-Level Interactive Control and Management Support System of Electric Vehicle Charging Facility Service Platform

XXXX-XX-XX 发布　　XXXX-XX-XX 实施

中国电工技术学会　发布

目　　次

前　言

本文件依据GB/T 1.1-2020《标准化工作导则　第 1 部分：标准的结构和编写》给出的规则起草。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

本文件由中国电工技术学会提出。

本文件起草单位：国网信息通信产业集团有限公司、国网天津市电力公司、天津联联睿科智慧能源科技有限公司、华瑞快充（天津）科技有限公司、云南电网能源公司、成都城投能源投资集团管理有限公司。

本文件主要起草人：李思维、袁帅、郭书麟、项冬南、裴小璐、张剑、祖国强、刘晓楠、张帅、曾欢、李俊达、胡朝伟等人。

本文件为首次制定。

面向电动汽车充电设施服务平台的电网级互动管控支持系统

接入规范

1 范围

1.1 业务背景

随着新能源汽车产业的快速发展，车网互动（Vehicle-to-Grid, V2G）技术已成为智能电网与新能源汽车协同发展的关键方向。电网级车网互动管控支持系统是指根据接入运营商的负荷调控潜力进行调控需求分解，向运营商级平台下发调控指令，采集各运营商负荷调控执行情况并进行监测的车网互动管控系统，电网级车网互动管控支持系统不直接控制充电设施。

当前，车网互动系统面临充电设施多运营商接入、电网负荷实时调控难度大、数据安全要求高的挑战。本规范旨在建立统一的接口接入标准，解决不同运营商平台与电网侧系统间的互联互通问题，支撑充电设施的电网侧调度管理，提升电网对新能源汽车负荷的接纳能力和调控效率。

1.2 技术范围

本文件规定了电网级互动管控支持系统的数据传输和安全防护的一般原则，包含充电桩等信息交换的数据传输体系、平台认证要求、密钥的管理和使用要求。

本文件适用于电网级互动管控支持系统与运营商服务平台或运营商级互动管控系统的充电设施服务信息交换。

2 规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件；不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 15148 - 2024 电力负荷管理系统技术规范

NB/T 33028 - 2018 电动汽车充放电设施术语

T/CEC 102.1 - 2021 电动汽车充换电服务信息交换 第 1 部分：总则

T/CEC 102.2 - 2021 电动汽车充换电服务信息交换 第 2 部分：公共信息交换规范

T/CEC 102.3 - 2021 电动汽车充换电服务信息交换 第 3 部分：业务信息交换规范

3 术语和定义

车网互动（V2G）：电动汽车通过充放电装置与电网相连，实现电能双向流动，以及车辆与电网之间信息交互的技术。电动汽车可作为储能单元向电网供电，也可从电网获取电能，同时电网可对电动汽车充电进行控制和管理。

负荷运营商：通过整合需求侧资源参与需求响应等负荷管理工作的需求侧资源协调优化服务机构。

电网级互动管控支持系统：根据接入运营商的负荷调控潜力进行调控需求分解，向运营商级平台下发调控指令，采集各运营商负荷调控执行情况并进行监测的车网互动管控系统，电网级车网互动管控支持系统不直接控制充电设施。

运营商级互动管控支持系统：接收电网级系统下发的调控指令，结合自身调控潜力，进行站桩维度的指令分解与下达，实现充电站/桩维度的高并发实时精准聚合调控。

运营商平台：泛指运营商级互动管控支持系统和运营商充电设施运营管理平台。

4 符号和缩略语

无

5 数据传输与安全

5.1 数据传输体系

5.1.1 接口格式

所有数据传输均采用HTTPS 接口，全部数据采用UTF-8 编码方式。

应使用如下格式：https://[域名]/evcs/v[版本号]/接口名称。

5.1.2 架构设计

数据传输采用三层架构设计：

接入层：支持运营商平台通过 HTTPS 接口接入，采用负载均衡技术处理并发请求，单接口并发处理能力≥1000 次 / 秒。

传输层：建立基于软件定义网络（SDN）架构的智能广域网通道，通过应用层协议优化与动态路由策略，确保骨干网传输延时≤50ms，丢包率≤0.1%。

应用层：基于消息队列（MQ）实现异步数据交互，支持每秒 10 万条消息的吞吐量，保障海量数据传输的可靠性。

5.2 接口调用方式

所有接口均使用 https 的 post 方式提交参数，request 和 response 均使用JSON 格式传输据。网络延时不超过 200ms,丢包率不高于 1%。双方相互开通防火墙白名单，保障接口访问安全性。

5.3 消息主体规范

5.3.1 服务申请

一般由运营商平台标识（OperatorID）、参数内容（Data）、时间戳（TimeStamp）、自增序列（Seq）和数字签名（Sig）组成，消息主体内容见下表：

参数名	说明	举例
OperatorID	运营商平台标识	接口调用方的运营商平台ID，以组织机构代码作为

		运营商平台ID,组织机构代码为统一社会信用码的9-17位，例如统一社会信用码为911101904000072O1Z，组织机构代码为：400007201
Data	各接口具体参数信息	
TimeStamp	时间戳	接口请求时的时间戳信息，格式为yyyyMMddHHmmss
Sig	参数签名	
Seq	自增序列	4位自增序列取自时间戳，同一秒内按序列自增长，新秒重计。如0001

5.3.2 参数返回

数据传输接口的返回参数一般由返回值（Ret）、返回信息（Msg）、参数内容（Data）和数字签名（Sig）组成。

（a）Ret：必填字段，返回参数编码参看下方《返回参数编码表》。

（b）Msg：必填字段，有错误表示具体错误信息，无错误返回成功信息。

（c）Data：参数内容，具体参看各接口定义，所有数据采用UTF-8编码，JSON格式。

返回参数编码表：

Ret取值	说明
0	请求成功
1	请求失败
500	系统错误
4000	请求参数为空
4003	参数不合法
4009	鉴权失败

5.4 数据加解密及签名

5.4.1 密钥内容

参数体名称	类型
OperatorID	运营商平台标识：工商注册的组织机构代码，固定9位，作为唯一标识使用
OperatorSecret	运营商平台密钥：可采用32H、48H、和64H，由0-F字符组成，为申请认证使用
DataSecret	消息密钥：用于对Request和Response中的Data字段的内容进行加密
DataSecretIV	初始化向量：固定16位，用户AES加密过程的混合加密
SigSecret	可采用32H、48H、和64H，由0-F字符组成，为计算签名的密钥

5.4.2 密钥使用

发起https请求的一方，称为请求方；接收https请求的一方，称为响应方。数据对接时，双方会互发请求，因此双方地位对等，会根据请求发生的方向实时确定双方的角色。

一条HTTP的完整链条如下：

（a）请求方将要发送的参数进行JSON序列化，然后使用*响应方的消息密钥和初始化向量*进行加密，将加密后的结果写入到Data字段；

（b）请求方使用响应方的签名密钥对OperatorID+Data+TimeStamp+Seq组成的字符串计算签名，将结果写入到Sig字段；

（c）响应方收到请求数据后，token验证通过后，使用和第2步相同的方式计算Sig，和参数中的Sig对比是否相同；

（d）Sig验证通过后，使用响应方的消息密钥和初始化向量解密Data字段，获取真正的请求参数；

（e）响应方处理完毕后，将要返回的内容进行JSON序列化，然后使用响应方的消息密钥和初始化向量进行加密，将加密后的结果写入到Data字段；

（f）Response中的Sig。

5.4.3 加解密算法

使用对称加解密算法AES 128 位加解密，加解密模式采用CBC，填充模式采用PKCS5Padding 方式。

加密示例：

消息密钥（DataSecret）：1234567890abcdef

初始向量（DataSecretIV）：1234567890abcdef

明文（JSON格式的业务数据）：

{"ConnectorID":"123456789001CX01001","ConnectorName":"中心站1号直流桩枪A","ConnectorType":4,"VoltageUpperLimit":440,"VoltageLowerLimit":220,"Current":60,"Power":240.0,"ParkNo":"1002","NationalStandard":2}

密文（最后的Data字段）：

11F5ry/2y93X0Dg86V2SgFyyzr0qt60OJYQwzjUPIh72gJCAK+ trhQrbcxfNJITrxYllXsfQ5mGbzN plQAajhT5eiOi1UTzlWWsOT41sgfKfrYOgseT9INAESyrLuKj4oyRAjIp3RfDZ6fj6N4LYVNZ1ujE4 mO5e46xAkTkTnyzI+ AZdV01TccvNt/Cl/wmMoS5NSHfUUfpPDqeNpSUcRj+ 06jOizk4PQF1Re6+ + 0 h e1tPyCh+ T87b7Qusx9NDu8Px3pDvQWb6dSscF1UNGEC4bwQbkqdoTRWvFWZ9VeGzo=

加解密代码实现（java）：

```
package com.evs.laos.utils;

import lombok.extern.slf4j.Slf4j;
import org.apache.commons.codec.binary.Base64;

import javax.crypto.Cipher;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;
import java.nio.charset.StandardCharsets;
import java.util.Locale;
import java.util.Properties;


/**
 * AES加解密算法
 *
 * @author 电动汽车
 */
@Slf4j
public class AESUtil {
```

```
private static final String DEFAULT_SECRET_IEVSR = "1234567890123456";
private static final int KEY_LENGTH = 16;
private static final String VALID_SKEY = "secretKey 不能为空 !";
private static final String VALID_SKEY_LENGTH = "secretKey 长度必须为 16!";
private static final String VALID_IVSTR = "ivStr 不能为空 !";
private static final String VALID_IVSTR_LENGTH = "ivStr 长度必须为 16!";

/**
 * 加密
 *
 * @param sSrc 加密data
 * @param sKey 加密KEY
 * @return
 * @throws Exception
 */
public static String Encrypt(String sSrc, String sKey) throws Exception {
    return Encrypt(sSrc, sKey, DEFAULT_SECRET_IEVSR);
}

/**
 * 加密（加密向量）
 *
 * @param sSrc  加密data
 * @param sKey  加密密钥
 * @param ivStr 加密向量
 * @Description:加密操作
 */
public static String Encrypt(String sSrc, String sKey, String ivStr) throws Exception {
    //校验秘钥信息
    if (!validKeyInfo(sKey, ivStr)) {
        //校验未通过，返回空
        return null;
    }
    byte[] raw = sKey.getBytes();
    SecretKeySpec skeySpec = new SecretKeySpec(raw, "AES");
    // "算法/模式/补码方式"
    Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
    // 密码器
    // 使用CBC模式，需要一个向量iv，可增加加密算法的强度1234567890123456
    IvParameterSpec iv = new IvParameterSpec(ivStr.getBytes());
    // 初始化
    cipher.init(Cipher.ENCRYPT_MODE, skeySpec, iv);
    // 执行最终的加密操作
    byte[] encrypted = cipher.doFinal(sSrc.getBytes());
    String str = Base64.encodeBase64String(encrypted);
    // 获取操作系统的类型（目前默认无mac操作系统）
    Properties prop = System.getProperties();
    String os = prop.getProperty("os.name");
    if (os != null && os.toLowerCase(Locale.ENGLISH).startsWith("win")) {
        // windows 系统 windows 换行\r\n
        str = str.replaceAll("\r\n", "");
    } else {
```

```
        // linux 系统 linux 换行\n ;mac 换行 \r
        str = str.replaceAll("\n", "");
    }
    return str;
}


/**
 * 解密
 *
 * @param sSrc 解密Data
 * @param sKey 解密KEY
 * @return
 * @throws Exception
 */
public static String Decrypt(String sSrc, String sKey) throws Exception {
    return Decrypt(sSrc, sKey, DEFAULT_SECRET_IEVSR);
}


/**
 * 解密（解密向量）
 *
 * @param sSrc  解密Data
 * @param sKey  解密KEY
 * @param ivStr 解密向量
 * @return
 * @throws Exception
 */
public static String Decrypt(String sSrc, String sKey, String ivStr) throws Exception {
    //校验秘钥信息
    if (!validKeyInfo(sKey, ivStr)) {
        //校验未通过，返回空
        return null;
    }
    try {
        byte[] raw = sKey.getBytes(StandardCharsets.UTF_8);
        SecretKeySpec skeySpec = new SecretKeySpec(raw, "AES");
        Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        IvParameterSpec iv = new IvParameterSpec(ivStr.getBytes());
        cipher.init(Cipher.DECRYPT_MODE, skeySpec, iv);
        byte[] encrypted1 = Base64.decodeBase64(sSrc);
        try {
            byte[] original = cipher.doFinal(encrypted1);
            String originalString = new String(original, StandardCharsets.UTF_8);
            originalString = unEscapeChar(originalString);
            return originalString;
        } catch (Exception e) {
            return null;
        }
    } catch (Exception ex) {
        return null;
    }
}
```

```
    /**
     * 反向字符转换 解密后分别把&lt; &gt;&amp;&quot;&copy; 的转义字符 转换成 <，>，&，"，©
     *
     * @param beforeDecryptString
     * @return
     */
    private static String unEscapeChar(String beforeDecryptString) {
        String unEscapeStr = beforeDecryptString;
        unEscapeStr = unEscapeStr.replaceAll("&lt;", "<");
        unEscapeStr = unEscapeStr.replaceAll("&gt;", ">");
        unEscapeStr = unEscapeStr.replaceAll("&amp;", "&");
        unEscapeStr = unEscapeStr.replaceAll("&quot;", "\"");
        unEscapeStr = unEscapeStr.replaceAll("&copy;", "©");
        unEscapeStr = unEscapeStr.replaceAll(" ", " ");
        return unEscapeStr;
    }

    /**
     * 校验秘钥数据
     *
     * @param sKey  秘钥
     * @param ivStr 秘钥向量
     * @return
     */
    private static Boolean validKeyInfo(String sKey, String ivStr) {
        // 判断Key是否正确
        if (sKey == null) {
            log.info("AES加密输出信息：" + VALID_SKEY);
            return false;
        }
        // 判断Key是否为16位
        if (sKey.length() != KEY_LENGTH) {
            log.info("AES加密输出信息：" + VALID_SKEY_LENGTH);
            return false;
        }

        if (ivStr == null) {
            LoggerUtils.info("AES加密输出信息：" + VALID_IVSTR);
            return false;
        }
        // 判断ivStr是否为16位
        if (ivStr.length() != KEY_LENGTH) {
            LoggerUtils.info("AES加密输出信息：" + VALID_IVSTR_LENGTH);
            return false;
        }
        return true;
    }
}
```

5.4.4　签名算法

使用 HMAC-MD5 参数签名算法。

HMAC-MD5 算法说明：

公式如下：

HMAC（K，M) = H(K ⊕ opad| H(K ⊕ ipad| M))

其中：

K 是签名密钥（SigSecret），长度可为 64 字节，若小于该长度，后面补"0"；

M 是消息内容；

H 是散列函数；

opad 和 ipad 分别是由若干个 0x5c 和 0x36 组成的字符串；

⊕ 表示异或运算；

| 表示连接操作；

HMAC-MD5 签名计算流程：

(a)再签名密钥（SigSecret）后面添加 0 来创建一个长度为 64 的字符串（str）；

(b)将上一步生成的字符串（str）与 ipad（0x36）做异或运算，形成结果字符串（istr）；

(c)将消息内容（Data）附加到第二步的结果字符串（istr）末尾，形成新的结果字符串（istr-data）；

(d)对第三步生成的结果字符串（istr-data）做 MD5 运算，形成新的结果字符串（istr-data-md5）；

(e)将第一步生成的字符串（str）与 opad（0x5c）做抑或运算，形成结果字符串（ostr）；

(f)将第四步生成的结果字符串（istr-data-md5）附加到第五步的结果字符串（ostr）的末尾。形成新的结果字符串（ostr-istr-data-md5）；

(g)对第六步生成的结果字符串（ostr-istr-data-md5）做 MD5 运算，输出最终结果（out）。

签名示例：

签名密钥（SigSecret）：1234567890abcdef

运营商平台 ID（OperatorID）：123456789

业务数据（加密后的 Data 字段）：

```
11F5ry/2y93X0Dg86V2SgFyyzr0qt60OJYQwzjUPIh72gJCAK+trhQrbcxfNJITrxYllXsfQ5mGbzN
plQAajhT5eiOi1UTzlWWsOT41sgfKfrYOgseT9INAESyrLuKj4oyRAjIp3RfDZ6fj6N4LYVNZ1ujE4
mO5e46xAkTkTnyzI+AZdV01TccvNt/Cl/wmMoS5NSHfUUfpPDqeNpSUcRj+06jOizk4PQF1Re6++0
h e1tPyCh+T87b7Qusx9NDu8Px3pDvQWb6dSscF1UNGEC4bwQbkqdoTRWvFWZ9VeGzo=
```

时间戳（TimeStamp）：2019070103040506

自增序列（Seq）：0001

消息内容：OperatorID + Data + TimeStamp + Seq

签名（最后计算得到的Sig字段）：60387814F5C7B8D148496DA044A24459

签名代码实现（java）：

```
/**
 *
 * @param key  签名密钥
 * @param data 待签名字符串
 * @return
 */
```

```
public static String getHmacMd5Str(String key, String data) {
        String result = "";
        try {
                byte[] keyByte = key.getBytes("UTF-8");
                byte[] dataByte = data.getBytes("UTF-8");
                byte[] hmacMd5Byte = getHmacMd5Bytes(keyByte, dataByte);
                StringBuffer md5StrBuff = new StringBuffer();
                for (int i = 0; i < hmacMd5Byte.length; i++) {
                        if (Integer.toHexString(0xFF & hmacMd5Byte[i]).length() == 1)
                                md5StrBuff.append("0").append(
                                                Integer.toHexString(0xFF & hmacMd5Byte[i]));
                        else
                                md5StrBuff.append(Integer
                                                .toHexString(0xFF & hmacMd5Byte[i]));
                }
                result = md5StrBuff.toString().toUpperCase();

        } catch (Exception e) {
                e.printStackTrace();
        }
        return result;

}
```

5.4.5 加密算法增强说明

采用 AES 128 位加密算法的原因：

运算效率：在 Java 环境下，AES 128 的加密速度约为 200MB/s，满足数据实时加密需求。

安全性：128 位密钥强度可抵御当前已知的暴力破解攻击，符合《信息安全技术 信息系统等级保护基本要求》中第三级安全要求。

兼容性：广泛支持硬件加速（如 Intel AES-NI 指令集），降低服务器计算资源消耗。

5.4.6 签名算法应用场景

HMAC-MD5 签名算法适用于以下场景：

接口防篡改：确保数据在传输过程中未被非法修改，如档案信息推送、负荷调控指令下发等关键业务。

身份认证：通过签名验证接口调用方身份，防止恶意伪造请求，如 token 获取、市场清分数据交互等。

数据完整性校验：对实时数据上传（如电流、电压等监测数据）进行签名，确保数据真实性。

5.5 权限认证

5.5.1 接口说明

交互双方互为对方分配 服务商ID和 数据加密密钥信息。密钥信息四个， 运营商平台密钥（OperatorSecret）、消息密钥（DataSecret）、消息密钥初始化向量（DataSectetIV）、签名密钥（SigSecret）。在进行数据传递前，向对方系统获取token ,在具体业务请求header中携带（Bearer Token :token）。Token有效期为2小时，过期则失效；失效时再次访问时，需再次获取token信息。

5.5.2 请求方式

接口名称	请求方式	发起方
/api/cec/query_token	HTTP-POST	双方鉴权

5.5.3 请求参数

参数名称	参数类型	必填	参数描述
OperatorID	String	Y	运营商平台ID
OperatorSecret	String	Y	运营商平台密钥

5.5.4 输出参数

参数名称	参数类型	必填	参数描述
OperatorID	String	Y	运营商平台ID
Code	int	Y	访问结果
AccessToken	String	Y	Token
TokenAvailableTime	Int	Y	持续时间
Message	String	Y	成功

5.6 数据交互流程

5.6.1 流程图

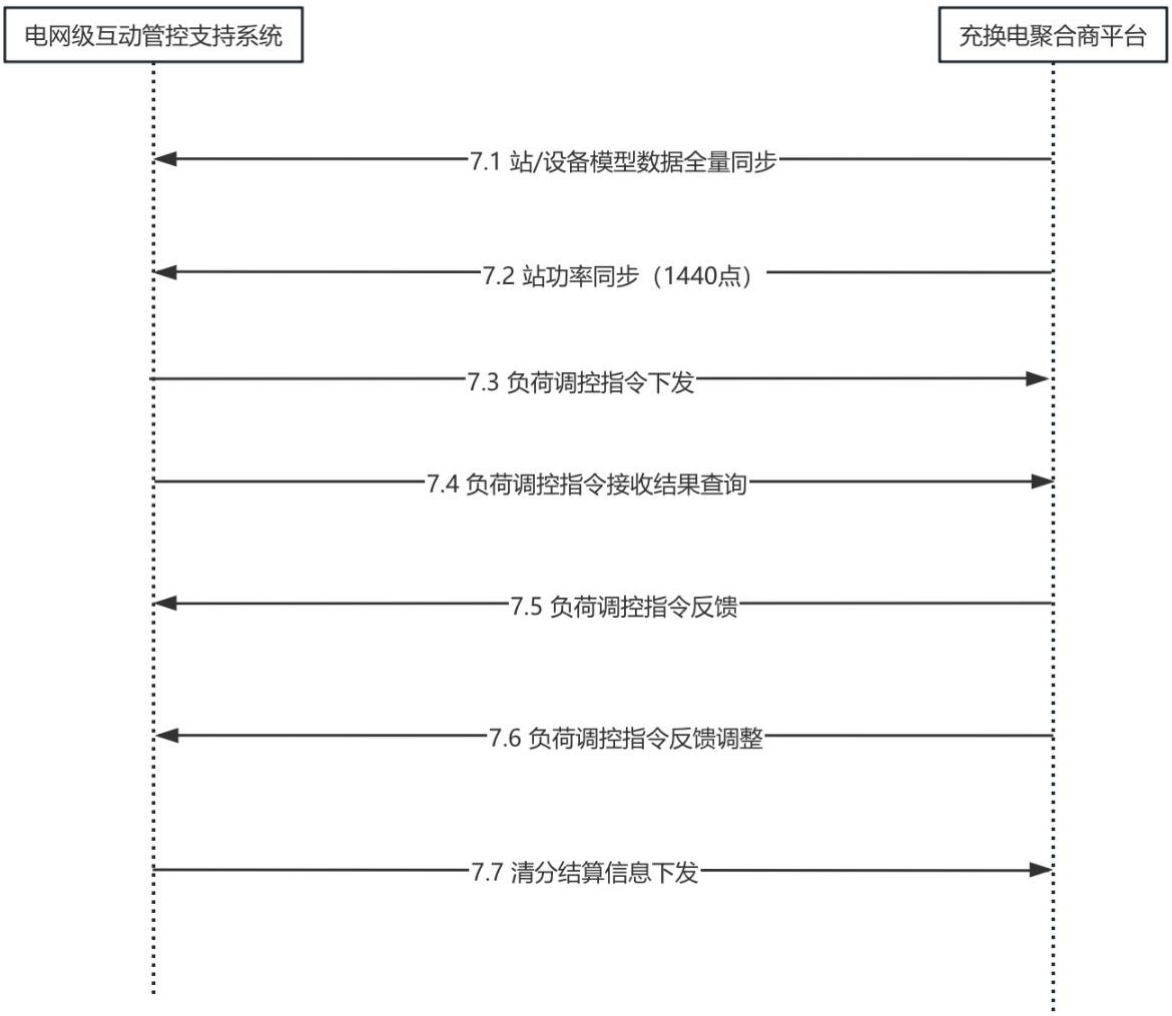

图 1 互动管控系统数据交互流程图

5.7 接口列表

接口名称	功能描述	请求方向
站/设备模型数据全量同步	运营商平台主动向电网级互动管控支持系统转发站/设备模型数据，主动同步数据为全量数据。触发形式定时任务触发（每天一次，时间为上午9点）。	运营商平台 → 电网级互动管控支持系统
站功率同步（1440点）	运营商平台向电网级互动管控支持系统发送站1440点动态数据，频率1分钟一次（时间按照整分钟）。数据上送过程中，若出现掉点或null，电网级互动管控支持系统则默认该点为0。 运营商平台24小时内数据的采集与上送成功率≥95%。	运营商平台 → 电网级互动管控支持系统
负荷调控指令下发	电网级互动管控支持系统向运营商平台发送/变更/取消负荷调控指令数据。	电网级互动管控支持系统 → 运营商平台

负荷调控指令接收结果查询	电网级互动管控支持系统在指令执行前向运营商平台查询负荷调控指令接收结果数据。	电网级互动管控支持系统 → 运营商平台
负荷调控指令反馈	运营商平台向电网级互动管控支持系统发送负荷调控指令反馈数据。	运营商平台 → 电网级互动管控支持系统
负荷调控指令反馈调整	虚拟电厂向运营商平台发送负荷调控指令数据，运营商对调控指令数据接受但是不完全接受，包括降低调控目标和增高调控目标，通过该接口进行上送	运营商平台 → 电网级互动管控支持系统
清分结算信息下发	虚拟电厂向运营商平台发送负荷调控清分结算信息	电网级互动管控支持系统 → 运营商平台

6 公共信息对象定义

6.1 站信息<Station>

参数名称	是否必填	类型	中文名称	描述
code	是	string	充电站编号	站唯一编码
operatorID	是	string	站所属平台组织机构代码	站所属平台组织机构代码
creditCode	是	string	站所属运营商统一社会信用码	所属运营商统一社会信用码；若为运营商平台互联互通设备，则保留原有统一社会信用码（加密）
name	是	string	充电站名称	充电站名称（加密）
depositorNoList	否	List<String>	营销户号	该站在营销系统报装时分配的营销户号，即交电费时的户号，以集合形式传输
electricMeterNoList	否	List<String>	电表表号	营销户号对应电表的表地址码，以集合形式传输
countryCode	是	string	充电站国家代码	国家代码，比如 CN
areaCode	是	string	充电站省市县辖区编码	省市辖区的区域代码，参考 GB/T 2260-2007
address	是	string	详细地址	充电站详细地址（加密）
type	是	int	站类型	1：公共； 50：个人； 100：公交（专用）； 101：环卫（专用）； 102：物流（专用）； 103：出租车（专用）； 255：其他
equipmentType	是	int	站负荷明细类型	1 充电站，2 储能 3 电采暖，4 V2G，5 换电站，6 其他
isPublic	是	int	公专用	1：公用； 0：专用；
state	是	int	站点状态	0：未知；

				1：建设中； 5：关闭下线； 6：维护中； 50：正常使用
openTime	否	string	站点开放时间	yyyy-MM-dd HH:mm
closeTime	否	string	站点关闭时间	yyyy-MM-dd HH:mm
longitude	是	double	经度	经度，GCJ-02坐标系，保留小数点后6位（加密）
latitude	是	double	纬度	纬度，GCJ-02坐标系，保留小数点后6位（加密）
construction	否	int	建设场所	1：居民区； 2：公共机构； 3：企事业单位； 4：写字楼； 5：工业园区； 6：交通枢纽； 7：大型文体设施； 8：城市绿地； 9：大型建筑配建停车场； 10：路边停车位； 11：城际高速服务区； 255：其他
putDate	否	string	站投运日期	yyyy-MM-dd
stakeList	是	List<Stake>	设备信息列表	参考《6.2.设备信息》

6.2 设备信息<Stake>

参数名称	是否必填	类型	中文名称	描述
code	是	string	设备编号	设备唯一编码
name	是	string	设备名称	设备名称（加密）
depositorNo	否	string	营销户号	该设备用电时，营销系统分配的营销户号
electricMeterNo	否	string	电表表号	营销户号对应电表的表地址编码
operatingState	是	int	设备运营状态	1：投运； 2：待投运: 3：停运: 4：退运；
debugStatus	否	int	设备调试状态	1：已调试通过

				2：待检测； 3：待调试；
productionDate	否	string	设备生产日期	yyyy-MM-dd
putDate	是	string	设备投运日期	yyyy-MM-dd
isPublic	是	int	公专用	1：公用 0：专用
isAdjust	是	int	是否可调	1：可调 0：不可调
type	是	int	设备类型	1：直流设备； 2：交流设备； 3：交直流一体设备； 4：无线设备； 5：V2G 6：储能 7：其他
model	是	string	设备型号	设备型号
longitude	否	double	设备经度	设备经度，GCJ-02坐标系，保留小数点后6位（加密）
latitude	否	double	设备纬度	设备纬度，GCJ-02坐标系，保留小数点后6位（加密）
ratedPower	是	double	设备额定功率	单位:kW，保留小数点后2位
ratedVoltage	否	double	额定电压	单位:V，保留小数点后1位
ratedAmpere	否	double	额定电流	单位：A，保留小数点后1位
topPower	是	double	可调节功率上限	单位:kW，保留小数点后2位
bottomPower	是	double	可调节功率下限	单位:kW，保留小数点后2位

6.3 站功率数据<StationPower>

参数名称	是否必填	类型	中文名称	描述
stationCode	是	string	站编码	站唯一编码
creditCode	是	string	运营商统一社会信用码	站运营商统一社会编码（加密）
adjustTopPower	是	double	可调节上限	单位：kW,保留小数点后2位
adjustBottomPower	是	double	可调节下限	单位：kW,保留小数点后2位
powerPointList	是	List<PowerPoint>	时间-功率曲线	参考《6.4时间-功率曲线》

6.4 时间-功率曲线<PowerPoint>

参数名称	是否必填	类型	中文名称	描述
point	是	string	时标	格式：yyyy-MM-dd HH:mm 举例：2020-04-09 12:29
power	是	double	功率	单位：kW,保留小数点后2位

6.5 负荷调控目标数据<AdjustTarget>

参数名称	类型	中文名称	描述
targetPower	double	目标功率	单位：kW，保留两位小数
startTime	string	开始时间	格式：yyyy-MM-dd HH:mm 举例：2020-04-09 12:29
endTime	string	结束时间	格式：yyyy-MM-dd HH:mm 举例：2020-04-09 12:29

7 数据交互接口

7.1 站/设备模型数据全量同步

接口名称：send_station_info。

7.1.1 场景

运营商平台调用电网级互动管控支持系统接口进行同步。

运营商平台主动向电网级互动管控支持系统转发站/设备模型数据，主动同步数据为全量数据，数据范围为全域的站/设备模型数据，以创建时间为节点进行同步。触发形式定时任务触发（每天一次，时间为上午9点）。

7.1.2 上行：模型数据同步

业务请求参数体，具体包含参数如下

参数名称		是否必选	类型	中文名称	描述
data	totalCount	是	int	数据总条数	
	pageCount	是	int	总页数	
	pageNo	是	int	当前页	
	stationList	是	List<Station>	站信息集合	参考《6.1.站信息》

7.1.3 上行 data 示例（需加密）

```
{
"totalCount":2000,
"pageCount":2,
"pageNo": 1,
"stationList": [...]
}
```

7.1.4 下行示例

```
{
"data": null,
```

```
"timestamp": "202209091233",
 "msg": "请求成功",
 "ret": 200
}
```


7.2 站功率同步（1440 点）

接口名称：send_station_power_1440。

7.2.1 场景

运营商平台调用电网级互动管控支持系统接口进行同步。

运营商平台向电网级互动管控支持系统发送站1440点动态数据，频率1分钟一次（时间按照整分钟），数据范围为全域站数据。数据上送过程中，若出现掉点或null，电网级互动管控支持系统则默认该点为0。

运营商平台24小时内数据的采集与上送成功率≥95%。

7.2.2 上行：站功率数据同步

业务请求参数体，具体包含参数如下：

参数名称		是否必填	类型	中文名称	描述
data	stationPowerList	是	List<StationPower>	站-功率数据集合	参考《6.3.站功率数据》

7.2.3 上行 data 示例（需加密）

可将多个站数据同时同步。


```
{
 "stationPowerList": [
{
 "stationCode": 105, //站编号
 "creditCode": "92131102MA09D2DT17", //统一社会信用码
 "adjustTopPower":30.25,
 "adjustBottomPower":19.25,
 "powerPointList": [
            {"point":"2020-04-09 12:15",
             "power":25.45
                    }
                    ]
 },
{
 "stationCode": 106, //站编号
 "creditCode": "92131102MA09D2DT17", //统一社会信用码（加密）
 "adjustTopPower":31.25,
 "adjustBottomPower":14.25,
```

```
 "powerPointList": [
            {"point":"2020-04-09 12:16",
             "power":26.28
                        }
                        ]
 },
.....
]
}
```

7.2.4 下行示例

```
{
"data": null,
"timestamp": "202209091233",
 "msg": "请求成功",
 "ret": 200
}
```

7.3 负荷调控指令下发

接口名称：send_task。

7.3.1 场景

电网级互动管控支持系统调用运营商平台接口。

电网级互动管控支持系统向运营商平台发送负荷调控指令数据。支持以市场、站、设备、台区及运营商维度下发指令。

7.3.2 上行：调控指令

业务请求参数体，具体包含参数如下

参数名称		是否必填	类型	中文名称	描述
data	taskID	是	string	需求编号	需求侧唯一标识（新增、变更、取消均为同一标识）
	regulationID	是	string	指令编号	指令唯一标识（新增、变更、取消均不同）
	type	是	int	指令类型	1:削峰 2:填谷
	synchroType	是	int	同步类型	1:新增 2:变更 3:取消
	rangeType	是	int	范围类型	1:市场 ； 2：台区； 3：运营商；

					4:站（仅支持对单一站下发指令）； 5:设备（仅支持对单一设备下发指令）；
	isForce	是	int	是否强制执行	1:强制 0:非强制
	adjustIDList	是	List<String>	调控ID集合	根据不同rangeType判断此ID集合为市场ID、站/桩编号、台区编号还是运营商统一社会信用码
	creditCode	否	string	运营商统一社会信用码	范围类型若为站或桩，则必填。标识站/桩所属运营商统一社会信用码（加密）
	adjustTargetList	是	List<AdjustTarget>	负荷调控目标数据集合	参考《6.5.负荷调控目标数据》

7.3.3 上行 data 示例（需加密）

同范围类型可多条数据同时上送。

```
{
"taskID": "3231323331", //需求编号
"regulationID": "FHTK-105-20200419", //指令编号
"type": 1,//指令类型
"synchroType": 1,//同步类型
"rangeType": 3,//范围类型
"isForce": 1,//强制类型
"adjustIDList":  ["123","234","345","456","567"],
"creditCode": null, //统一社会信用码
"adjustTargetList":  [
                    {
            "targetPower": 30.45,//目标值
            "startTime": "2020-04-09 12:29",//调控开始时间
            "endTime": "2020-04-10 12:29" //调控结束时间
                },
                    {
            "targetPower": 25.56,//目标值
            "startTime": "2020-04-10 12:30",//调控开始时间
            "endTime": "2020-04-11 12:29" //调控结束时间
                }
                  ]
```

```
}
```

7.3.4 下行示例

```
{
"data": null,
"timestamp": "202209091233",
 "msg": "请求成功",
 "ret": 200
}
```

7.4 负荷调控指令接收结果查询

接口名称：query_task_result。

7.4.1 场景

电网级互动管控支持系统调用运营商平台接口。

电网级互动管控支持系统向运营商平台请求负荷调控指令接收结果，以确认指令执行前运营商平台的处理状态。

7.4.2 上行：调控指令反馈

业务请求参数体，具体包含参数如下。

参数名称		是否必填	类型	中文名称	描述
data	taskID	是	string	需求编号	需求侧唯一标识（新增、变更、取消均为同一标识）
	regulationID	是	string	指令编号	指令唯一标识（新增、变更、取消均不同）

7.4.3 上行 data 示例

```
{
"taskID": "123324123",//需求编号
 "regulationID": "FHTK-105-20200419"//指令编号
}
```

7.4.4 下行：数据

业务请求参数体，具体包含参数如下

参数名称		是否必填	类型	中文名称	描述
data	taskID	是	string	需求编号	需求侧唯一标识（新增、变更、取消均为同一标识

					）
	regulationID	是	string	指令编号	指令唯一标识（新增、变更、取消均不同）
	type	是	int	指令类型	1:削峰 2:填谷
	synchroType	是	int	同步类型	1:新增 2:变更 3:取消
	rangeType	是	int	范围类型	1:市场 ； 2:站（仅支持对单站下发指令）； 3:设备（仅支持对单设备下发指令）； 4：台区； 5：运营商
	creditCode	否	string	运营商统一社会信用码	范围类型若为站或桩，则必填。标识站/桩所属运营商统一社会信用码（加密）
	isForce	是	int	是否强制执行	1:强制 0:非强制
	status	是	int	状态	1:接收 0:拒绝 2:处理中
	adjustIDList	是	List<String>	调控ID集合	根据不同rangeType判断此ID集合为市场ID、站/桩编号、台区编号还是运营商统一社会信用码
	adjustTargetList	是	List<AdjustTarget>	调控目标功率集合	参考《6.5.负荷调控目标功率》

7.4.5 下行 data 示例（需加密）

可多条数据同时上送。

```
{
"taskID": "3231323331", //需求编号
"regulationID": "FHTK-105-20200419", //指令编号
"type": 1,//指令类型
"synchroType": 1,//同步类型
"rangeType": 3,//范围类型
"creditCode": null,
"isForce": 1,//强制类型
"status": 1,//状态
 "adjustIDList":  ["123","234","345","456","567"],
```

```
"adjustTargetList":  [
{
            "targetPower": 30.45,//目标值
            "startTime": "2020-04-09 12:29",//调控开始时间
            "endTime": "2020-04-10 12:29" //调控结束时间
},
{
            "targetPower": 25.56,//目标值
            "startTime": "2020-04-10 12:30",//调控开始时间
            "endTime": "2020-04-11 12:29" //调控结束时间
   }
]
}
```

7.5 负荷调控指令反馈

接口名称：callback_task_overview。

7.5.1 场景

运营商平台调用电网级互动管控支持系统接口。

运营商平台向电网级互动管控支持系统发送负荷调控指令是否接受反馈数据，目前仅支持全部接受或全部拒绝。

7.5.2 上行：调控指令反馈

业务请求参数体，具体包含参数如下

参数名称		是否必填	类型	中文名称	描述
data	taskID	是	string	需求编号	需求侧唯一标识（新增、变更、取消均为同一标识）
	regulationID	是	string	指令编号	指令唯一标识（新增、变更、取消均不同）
	status	是	int	反馈状态	1、接受 0、拒绝

7.5.3 上行 data 示例

```
{
 "taskID": "1233321412",//需求编号
   "regulationID": "FHTK-105-20200419",//指令编号
   "status": 1//反馈状态
```

```
}
```

7.5.4 下行示例

```
{
"data": null,
"timestamp": "202209091233",
 "msg": "请求成功",
 "ret": 200
}
```

7.6 负荷调控指令反馈调整

接口名称：callback_task_check。

7.6.1 场景

运营商平台调用电网级互动管控支持系统接口。

虚拟电厂向运营商平台发送负荷调控指令数据，运营商对调控指令数据接受但是不完全接受，包括降低调控目标和增高调控目标，通过该接口进行上送。

7.6.2 上行：调控指令反馈

业务请求参数体，具体包含参数如下

参数名称		是否必填	类型	中文名称	描述
data	taskID	是	string	需求编号	需求侧唯一标识（新增、变更、取消均为同一标识）
	regulationID	是	string	指令编号	指令唯一标识（新增、变更、取消均不同）
	type	是	int	指令类型	1:削峰 2:填谷
	synchroType	是	int	同步类型	1:新增 2:变更 3:取消
	rangeType	是	int	范围类型	1:市场 ； 2：台区； 3：运营商； 4:站（仅支持对单一站下发指令）； 5:设备（仅支持对单一设备下发指令）；
	adjustIDList	是	List<String>	调控ID集合	根据不同rangeType判断此ID集合为市场ID、站/桩编号、台区编号还是运营商统一社

					会信用码
	creditCode	否	string	运营商统一社会信用码	范围类型若为站或桩，则必填。标识站/桩所属运营商统一社会信用码（加密）
	adjustTargetList	是	List<AdjustTarget>	负荷调控目标数据集合	参考《6.5.负荷调控目标数据》

7.6.3 上行 data 示例

```
{
"taskID": "3231323331", //需求编号
"regulationID": "FHTK-105-20200419", //指令编号
"type": 1,//指令类型
"synchroType": 2,//同步类型
"rangeType": 3,//范围类型
"adjustIDList":  ["123","234","345","456","567"],
"creditCode": null, //统一社会信用码
"adjustTargetList":  [
                    {
            "targetPower": 30.45,//目标值
            "startTime": "2020-04-09 12:29",//调控开始时间
            "endTime": "2020-04-10 12:29" //调控结束时间
                },
                {
            "targetPower": 25.56,//目标值
            "startTime": "2020-04-10 12:30",//调控开始时间
            "endTime": "2020-04-11 12:29" //调控结束时间
                }
                ]
}
```

7.6.4 下行示例

```
{
"data": null,
"timestamp": "202209091233",
 "msg": "请求成功",
 "ret": 200
}
```

7.7 清分结算信息下发

接口名称：exchange_inviteplan_push。

7.7.1 场景

虚拟电厂调用运营商平台接口。
虚拟电厂向运营商平台发送负荷调控清分结算信息。

7.7.2 上行：清分结算信息下发返回信息

业务请求参数体，具体包含参数如下

7.7.3 上行 data 示例

```
{
"data": null,
"timestamp": "202209091233",
 "msg": "请求成功",
 "ret": 200
}
```

7.7.4 下行数据

参数名称		是否必填	类型	中文名称	描述
data	taskID	是	string	需求编号	需求侧唯一标识（新增、变更、取消均为同一标识）
	regulationID	是	string	指令编号	指令唯一标识（新增、变更、取消均不同）
	type	是	int	指令类型	1:削峰 2:填谷

	points	是	Array	收益 earnings
				价格 price
					[
					{
					"earnings": 1,
					"price": 1
					},
					{
					"earnings": 1,
					"price": 1
					}
					]

7.7.5 下行示例

```
{
"taskID": "2024096544433"
"type":1,
"regulationID": "12e32e4rj333333",
"points":[
     {
```

```
                "earnings": 1,
                "price": 1
        },
        {
                "earnings": 1,
                "price": 1
        }
]
}
```