

**5G 消息**

# **开发指南**

文档版本 01

发布日期 2023-08-14



华为技术有限公司



**版权所有 © 华为技术有限公司 2023。保留一切权利。**

未经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

## **商标声明**



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

## **注意**

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

# 目 录

---

<b>1 开发准备.....</b>	<b>1</b>
<b>2 API 代码样例.....</b>	<b>3</b>
2.1 JAVA.....	3
2.1.1 多媒体文件上传.....	3
2.1.2 多媒体文件下载.....	7
2.1.3 媒体文件删除.....	9
2.1.4 单卡片 5G 消息发送.....	11
2.1.5 带建议回复消息的多卡片 5G 消息.....	15
2.1.6 码流转成 json 的多卡片+悬浮菜单.....	18
<b>3 错误码.....</b>	<b>23</b>

# 1 开发准备

5G消息开发有如下两种接入方式：

- 通过直接调用API接口
  - 详情请参见[开发流程、API代码样例](#)
- 通过应用魔方实现接入
  - 详情请参考[进入AstroZero开发环境、创建5G消息工程](#)

## 开发流程

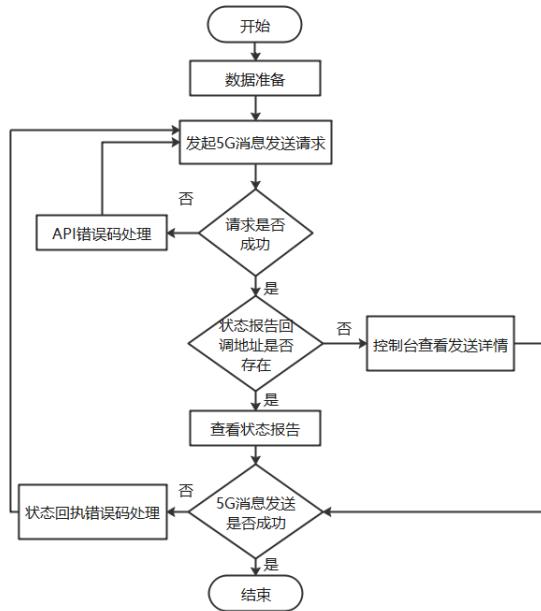
主要进行下述操作：

1. 参考[数据准备](#)，获取调用5G消息API的关联数据。
2. 参考[代码样例](#)，进行多媒体文件上传、审核、下载、删除。
3. 根据请求响应消息，判断请求是否成功。

### 说明

若请求失败，请参考[错误码](#)进行修正，并重新执行2。

4. 根据短信发送请求，判断是否存在状态报告回调地址。
5. 根据状态报告通知，判断5G消息是否发送成功。



## 数据准备

序号	字段	数据类型	可选属性	描述
1	serverRoot	string	M	服务器基础URL: hostname(或ip)+port+base path Port和base path可选 例: ip:port/openchatbot
2	apiVersion	string	M	客户端想使用的API版本号. 例: “v2” .
3	chatbotAddress	string	M	行业消息的统一服务地址, 客户端可根据此地址将所有通知集合展现。 chatbotAddress是5G云服务平台分配给购买者的chatbotID。如果您已经购买了5G消息应用资产, 则可以登录消费者门户, 选择“应用管理”查看chatbotID。

# 2 API 代码样例

JAVA

## 2.1 JAVA

### 说明

- 本文档所述Demo在提供服务的过程中，可能会涉及个人数据的使用，建议您遵从国家的相关法律采取足够的措施，以确保用户的个人数据受到充分的保护。
- 本文档所述Demo仅用于功能演示，不允许客户直接进行商业使用。
- 本文档信息仅供参考，不构成任何要约或承诺。

### 2.1.1 多媒体文件上传

#### 代码样例

```
import static java.nio.charset.StandardCharsets.UTF_8;
import com.fasterxml.jackson.databind.ObjectMapper;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.HttpURLConnection;
import java.net.URL;
import java.security.NoSuchAlgorithmException;
import java.util.HashMap;
import java.util.Map;
import java.util.logging.Level;
import java.util.logging.Logger;
/**
 * 5G媒体文件上传代码样例
 */
public final class FileUpload {
    static final int SUCCESS = 200;
    static final int ERR = 204;
    private FileUpload() {
    }

    /**
     *
     */
}
```

```
* 功能描述
*
* @param args
* @throws IOException
* @throws NoSuchAlgorithmException
*/
public static void main(String[] args) throws IOException, NoSuchAlgorithmException {
    File file = new File("src/1.txt");
    file.setFileId("12341234");
    file.setFileSize(20);
    file.setFileUrl("src/1.txt");
    file.setStatus("OK");
    file.setValidity("");
    Map<String, Object> map = new HashMap<>();
    map.put("File",file);
    post("http://10.120.207.128:8323/openchatbot/v2"
        + "/sip:chatbotid32test1@botplatform.rcs.chinamobile.com/files",map);
}
/**
* 定义post方法
*
* @param url
* @param params
* @Parameters: parms, 访问参数,Map形式, 以K-v存放多组值。
* @return null
* @throws IOException
* @throws NoSuchAlgorithmException
*/
public static String post(String url, Map<String, Object> params) throws IOException,
NoSuchAlgorithmException {
    Logger logger = Logger.getLogger(FileUpload.class.getName());
    logger.setLevel(Level.WARNING);
    logger.warning("url" + url);

    // 获取认证信息authorization参数信息
    String authorization = "Username=\"appId32test1\",Password=\"*****\"";

    // 获取认证信息X-WSSE
    String xwsse = "UsernameToken Username=\"appId32test1\"";
    URL url1 = new URL(url);
    InputStream is;

    // 将url以open方法返回的URLConnection连接强转为HttpURLConnection连接(标识一个url所引用的远程对象连接),此时connection只是为一个连接对象,待连接中
    HttpURLConnection conn = (HttpURLConnection) url1.openConnection();

    // 设置连接输出流为true,默认false (post请求是以流的方式隐式的传递参数)
    conn.setDoOutput(true);

    // 设置连接输入流为true
    conn.setDoInput(true);

    // 设置请求方式为post
    conn.setRequestMethod("POST");

    // post请求缓存设为false
    conn.setUseCaches(false);

    // 设置该HttpURLConnection实例是否自动执行重定向
    conn.setInstanceFollowRedirects(true);

    // 设置内容的类型,设置为经过urlEncoded编码过的from参数
    conn.setRequestProperty("Content-Type", "application/json");
    conn.setRequestProperty("Authorization", authorization);
}
```

```
conn.setRequestProperty("X-WSSE", xwsse);
conn.setRequestProperty("User-Agent", "2");
ObjectMapper objectMapper = new ObjectMapper();
PrintWriter pw = new PrintWriter(new OutputStreamWriter(conn.getOutputStream(), UTF_8));
pw.write(objectMapper.writeValueAsString(params));

// 查看传参信息
logger.warning(objectMapper.writeValueAsString(params));
pw.flush();
pw.close();

// 建立连接(请求未开始,直到connection.getInputStream()方法调用时才发起,以上各个参数设置需在此方法之前进行)
conn.connect();

// 输出连接信息, 实际使用时去掉
outConnInfo(conn, url1);

// 连接发起请求,处理服务器响应 (从连接获取到输入流并包装为bufferedReader)
int status = conn.getResponseCode();
logger.warning("status:" + status);
if (status != SUCCESS) {
    logger.warning("No Content");
    if (status == ERR) {
        logger.warning("err");
        is = conn.getInputStream();
    } else { // 400/401
        is = conn.getErrorStream();
    }
} else { // 200
    logger.warning("OK,SUCESS");
    is = conn.getInputStream();
}
BufferedReader br = new BufferedReader(new InputStreamReader(is, UTF_8));
String line;
StringBuilder resultMsg = new StringBuilder();
while ((line = br.readLine()) != null) { // 读取数据
    resultMsg.append(line);
}

// 关流, 这个很重要
is.close();
br.close();

// 关闭连接
conn.disconnect();
logger.warning("resultMsg.toString()" + resultMsg.toString());
return null;
}

// 定义传入的文件,携带需要上传的参数
/**
 * 功能描述
 *
 */

public static class File {
    private String fileId;
    private Integer fileSize;
    private String status;
    private String fileUrl;
    private String validity;

    /**
     * 功能描述
     *
     */
    @param s
}
```

```
/*
public File(String s) {
}

public String getFileId() {
    return fileId;
}

public void setFileId(String fileId) {
    this.fileId = fileId;
}

public Integer getFileSize() {
    return fileSize;
}

public void setFileSize(Integer fileSize) {
    this.fileSize = fileSize;
}

public String getStatus() {
    return status;
}

public void setStatus(String status) {
    this.status = status;
}

public String getFileUrl() {
    return fileUrl;
}

public void setFileUrl(String fileUrl) {
    this.fileUrl = fileUrl;
}

public String getValidity() {
    return validity;
}

public void setValidity(String validity) {
    this.validity = validity;
}

}

/**
 * 输出连接信息 ,实际业务不涉及
 */
private static void outConnInfo(HttpURLConnection conn, URL url) throws IOException {
    Logger logger = Logger.getLogger(FileUpload.class.getName());
    logger.setLevel(Level.WARNING);
    logger.warning("conn.getResponseCode():" + conn.getResponseCode());
    logger.warning("conn.getURL():" + conn.getURL());
    logger.warning("conn.getRequestMethod():" + conn.getRequestMethod());
    logger.warning("conn.getContentType():" + conn.getContentType());
    logger.warning("conn.getReadTimeout():" + conn.getReadTimeout());
    logger.warning("conn.getResponseMessage():" + conn.getResponseMessage());
    logger.warning("url.getDefaultPort():" + url.getDefaultPort());
    logger.warning("url.getFile():" + url.getFile());
    logger.warning("url.getHost():" + url.getHost());
    logger.warning("url.getPath():" + url.getPath());
    logger.warning("url.getPort():" + url.getPort());
    logger.warning("url.getProtocol():" + url.getProtocol());
}
}
```

表 2-1 代码样例中可变参数说明

参数名称	参数说明
File	创建上传的内容（文件）
Map	封装发送请求时，请求的内容
Connection	发起请求时要携带的请求认证信息

## 消息样例

这是传入的请求  
urlhttp://10.120.207.128:8323/openchatbot/v2/sip:chatbotid32test1@botplatform.rcs.chinamobile.com/files  
以下是请求参数信息---  
{"File":{"fileId":"12341234","fileSize":20,"status":"OK","fileUrl":"src/1.txt","validity":""}}  
以上是请求参数信息---  
以下是获取响应信息---  
conn.getResponseCode():200  
conn.getURL():http://10.120.207.128:8323/openchatbot/v2/  
sip:chatbotid32test1@botplatform.rcs.chinamobile.com/files  
conn.getRequestMethod():POST  
conn.getContentType():application/xml  
conn.getReadTimeout():0  
conn.getResponseMessage():OK  
url.getDefaultPort():80  
url.getFile():/openchatbot/v2/sip:chatbotid32test1@botplatform.rcs.chinamobile.com/files  
url.getHost():10.120.207.128  
url.getPath():/openchatbot/v2/sip:chatbotid32test1@botplatform.rcs.chinamobile.com/files  
url.getPort():8323  
url.getProtocol():http  
url.getQuery():null  
url.getRef():null  
url.getUserInfo():null  
以上是获取响应信息====DDD=  
status:200

### 2.1.2 多媒体文件下载

#### 代码样例

```
import static java.nio.charset.StandardCharsets.UTF_8;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 * 功能描述
 */
public class FileDownload {
    /**
     * 正确码
     */
    public static final int SUCCESS_VALUE = 200;
    /**
     * 错误码
     */
```

```
/*
public static final int ERR_VALUE = 204;

private FileDownload() {
}
/** 
 * 文件下载功能
 *
 * @param args
 * @throws IOException
 *
 */
public static void main(String[] args) throws IOException {
Logger logger = Logger.getLogger(FileDownload.class.getName());
logger.setLevel(Level.WARNING);
InputStream is;
String url = "http://127.0.0.1:8323/openchatbot/v2/"
+ "sip:chatbotid32test1@botplatform.rcs.chinamobile.com/files";
URL realUrl = new URL(url);
HttpURLConnection connection = (HttpURLConnection) realUrl.openConnection();

// 设置请求方式
connection.setRequestMethod("GET");

connection.setRequestProperty("Authorization", "Username=\"appId32test1\",Password=\"*****\"");
connection.setRequestProperty("X-WSSE", "UsernameToken Username=\"appId32test1\"");
connection.setRequestProperty("File-Location", "http://10.134.204.203:8087/Content111");
connection.connect();
int status = connection.getResponseCode();
if (status != SUCCESS_VALUE) {
if (status == ERR_VALUE) {
logger.warning("No Content,sucess, nobody。");
is = connection.getInputStream();
} else { // 400/401
logger.warning("err");
is = connection.getErrorStream();
}
} else { // 200

logger.warning("OK,DELETE");
is = connection.getInputStream();
}
BufferedReader br;
br = new BufferedReader(new InputStreamReader(is, UTF_8));
String line;
StringBuilder result = new StringBuilder();

// 读取数据
while ((line = br.readLine()) != null) {
result.append(line);
}

// 关流操作，这个很重要
is.close();
br.close();

// 关闭连接
connection.disconnect();

logger.warning("mabye is null" + result.toString());
}
}
```

表 2-2 代码样例中可变参数说明

参数名称	参数说明
url	请求地址
Connection	发起请求时，建立连接的认证信息

## 消息样例

GET status:200

OK,DELETE请求成功，响应消息有消息体。

响应消息(可能消息体存在但是为空)。

### 2.1.3 媒体文件删除

#### 代码样例

```
import static java.nio.charset.StandardCharsets.UTF_8;
import com.fasterxml.jackson.databind.ObjectMapper;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.HashMap;
import java.util.Map;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 * 媒体文件删除样例
 */
public final class FileDelete {
    // 带有值的返回成功状态码
    static final int SUCCESS_200_RES = 200;

    // 不带值的返回成功状态码
    static final int SUCCESS_204_RES = 204;

    private FileDelete() {
    }

    /**
     * 媒体文件删除
     *
     * @param args
     * @throws IOException
     */
    public static void main(String[] args) throws IOException {
        Logger logger = Logger.getLogger(SendMsg.class.getName());
        logger.setLevel(Level.WARNING);

        String url =
            "http://127.0.0.1:8323/openchatbot/v2/sip:chatbotid32test1@botplatform.rcs.chinamobile.com/files";
        URL realUrl = new URL(url);
```

```
Map<String, Object> map = new HashMap<>();  
  
// 媒体删除文件要依据tid  
map.put("tid", "1234567890123459");  
HttpURLConnection connection = (HttpURLConnection) realUrl.openConnection();  
  
// 设置可输入  
connection.setDoInput(true);  
  
// 设置该连接是可以输出的  
connection.setDoOutput(true);  
  
// 设置请求方式  
connection.setRequestMethod("DELETE");  
  
connection.setRequestProperty("Authorization", "Username=\"appId32test1\"", "Password=\"*****\"");  
connection.setRequestProperty("X-WSSE", "UsernameToken Username=\"appId32test1\"");  
ObjectMapper objectMapper = new ObjectMapper();  
PrintWriter pw = new PrintWriter(new OutputStreamWriter(connection.getOutputStream(), UTF_8));  
pw.write(objectMapper.writeValueAsString(map));  
pw.flush();  
  
// 关流操作，这个很重要  
pw.close();  
connection.connect();  
int status = connection.getResponseCode();  
logger.warning("DELETE status: " + status);  
InputStream is;  
if (status != SUCCESS_200_RES) {  
if (status == SUCCESS_204_RES) {  
logger.warning("No Content, Success");  
is = connection.getInputStream();  
} else { // 400/401  
logger.warning("Error, DELETE status: " + status);  
is = connection.getErrorStream();  
}  
} else { // 200  
logger.warning("OK,DELETE Success, DELETE status: " + status);  
is = connection.getInputStream();  
}  
  
if (is != null) {  
BufferedReader br;  
br = new BufferedReader(new InputStreamReader(is, UTF_8));  
String line;  
StringBuilder result = new StringBuilder();  
  
// 读取数据  
while ((line = br.readLine()) != null) {  
result.append(line + System.lineSeparator().intern());  
}  
br.close();  
  
// 关流操作，这个很重要  
is.close();  
  
// 关闭连接  
connection.disconnect();  
logger.warning("Content: " + result.toString());  
}  
}  
}
```

表 2-3 代码样例中可变参数说明

参数名称	参数说明
url	请求地址

参数名称	参数说明
Map	封装请求发送的信息、参数
Connection	封装建立连接的认证信息、请求方式等

## 消息样例

DELETE status:200

OK,DELETE请求成功，响应消息有消息体。

响应消息(可能消息体存在但是为空)。

### 2.1.4 单卡片 5G 消息发送

#### 代码样例

```
package Demo;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.URL;
import java.security.NoSuchAlgorithmException;
import java.security.cert.CertificateException;
import java.security.cert.X509Certificate;
import java.util.HashMap;
import java.util.Map;
import java.util.logging.Level;
import java.util.logging.Logger;

import javax.net.ssl.HostnameVerifier;
import javax.net.ssl.HttpsURLConnection;
import javax.net.ssl.SSLContext;
import javax.net.ssl.SSLSession;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;

import com.fasterxml.jackson.databind.ObjectMapper;

import static java.nio.charset.StandardCharsets.UTF_8;

/**
 *单卡片消息请求
 */
public class SingleCardMs2 {

    static final int SUCCESS = 200;
    static final int ERR = 204;

    private static final HostnameVerifier DO_NOT_VERIFY= new HostnameVerifier() {

        @Override
        public boolean verify(String hostname, SSLSession session) {
            return true;
        }
    }
}
```

```

};

/**
 * 单卡片消息请求
 */
public static void main(String[] args) throws Exception {

    Map<String, Object> map1 = new HashMap<>();
    map1.put("contributionID", "contributionID");
    map1.put("senderAddress", "sip:471400.huawei@botplatform.rcs.chinamobile.com");
    map1.put("address", "sip:+861391111018@ims.mnc000.mcc460.3gppnetwork.org");
    map1.put("destinationAddress", "sip:+861391111018@ims.mnc000.mcc460.3gppnetwork.org");
    map1.put("bodyText", "\n{\n    \"layout\": {\n        \"cardWidth\": \"MEDIUM_WIDTH\"\n    },\n    \"content\": {\n        \"media\": {\n            \"mediaUrl\": \"http://10.243.50.178:9090/Access/PF?\nID=Q0Q5QjAxRUU1NDBFRDUxQUQ5RkFFOTY0RjQ3MjM4NjBGQkM2NDFDNzQ2OTBCMENBOUJDRjlBRUE3\nQ0U0NDA5NTA3QkQ4M0UxNzM4NzIERTJCQzMxNEM5QUFDRDg2Nzg1\"\n        },\n        \"mediaContentType\": \"image/png\"\n    },\n    \"thumbnailUrl\": \"http://10.243.50.178:9090/Access/PF?\nID=MkE2REU4QTQ2RjU4OEJBQjNBNDg3Q0ZCRUI0NUVBQzUxQUU1MkE5QjY0MDFCN0U5NkYwMzNFQjc5\nMjjGRDlGOEYxOTUwQjc5QUREQUCRky5ODBERUQ2ODMwNjEwOTY1\"\n    },\n    \"description\": \"This is the description of the rich card. It's the first field that will be truncated if it exceeds the maximum width or height of a card.\",\n    \"suggestions\": [\n        {\n            \"action\": {\n                \"showLocation\": {\n                    \"latitude\": 37.4220041,\n                    \"label\": \"Googleplex\"\n                },\n                \"fallbackUrl\": \"https://www.google.com/maps/@37.4219162,-122.078063,15z\"\n            },\n            \"mapAction\": {\n                \"location\": {\n                    \"longitude\": -122.0862515,\n                    \"label\": \"Googleplex\"\n                }\n            }\n        }\n    ],\n    \"postback\": {\n        \"data\": {\n            \"set_by_chatbot_open_map\": {\n                \"action\": {\n                    \"createCalendarEvent\": {\n                        \"title\": \"Meeting\"\n                    }\n                }\n            }\n        }\n    },\n    \"calendarAction\": {\n        \"startTime\": \"2017-03-14T00:00:00Z\",\n        \"endTime\": \"2017-03-14T23:59:59Z\"\n    },\n    \"description\": \"GSG review meeting\"\n},\n    \"displayText\": \"Schedule Meeting\"\n},\n    \"postback\": {\n        \"data\": {\n            \"set_by_chatbot_create_calendar_event\": {\n                \"title\": \"This is the second rich card in the carousel.\"\n            }\n        }\n    },\n    \"description\": \"Carousel cards need to specify a card width in the 'layout' section.\nFor small width cards, only short and medium height media are supported.\"\n},\n    \"data\": {\n        \"conversationID\": \"conversationID\",\n        \"contentType\": \"application/vnd.gsma.botmessage.v1.0+json\"\n    }\n},\n    Map<String, Object> map = new HashMap<>();\n    map.put("destinationAddress", "tel:+861391111030");\n    map.put("clientCorrelator", "567895");\n    map.put("outboundIMMessage", map1);\n\n    post("https://10.120.207.128:18323/openchatbot/v2/sip:\n888.chatbot@botplatform.rcs.chinamobile.com/outbound", map);\n}

/**
 * 定义post方法
*/

```

```
public static String post(String url, Map<String, Object> params) throws Exception {
    trustAllHttpsCertificates();
    Logger logger = Logger.getLogger(SingleCardMs2.class.getName());
    logger.setLevel(Level.WARNING);
    logger.warning("url" + url);

    URL url1 = new URL(url);
    InputStream is;

    // 将url以open方法返回的URLConnection连接强转为HttpURLConnection连接(标识一个url所引用的远程对象连接),此时connection只是为一个连接对象,待连接中
    HttpsURLConnection conn = (HttpsURLConnection) url1.openConnection();

    conn.setHostnameVerifier(DO_NOT_VERIFY);

    // 设置连接输出流为true,默认false (post请求是以流的方式隐式的传递参数)
    conn.setDoOutput(true);

    // 设置连接输入流为true
    conn.setDoInput(true);

    // 设置请求方式为post
    conn.setRequestMethod("POST");

    // post请求缓存设为false
    conn.setUseCaches(false);

    // 设置该HttpURLConnection实例是否自动执行重定向
    conn.setInstanceFollowRedirects(true);

    // 设置内容的类型,设置为经过urlEncoded编码过的from参数
    conn.setRequestProperty("Content-Type", "application/json");
    conn.setRequestProperty("Authorization",
    "Username=\"chatbottest12swt96\",Password=\"*****\"");

    ObjectMapper objectMapper = new ObjectMapper();
    PrintWriter pw = new PrintWriter(new OutputStreamWriter(conn.getOutputStream(), UTF_8));
    pw.write(objectMapper.writeValueAsString(params));

    // 查看传参信息
    logger.warning(objectMapper.writeValueAsString(params));
    pw.flush();
    pw.close();

    // 建立连接(请求未开始,直到connection.getInputStream()方法调用时才发起,以上各个参数设置需在此方法之前进行)
    conn.connect();

    // 连接发起请求,处理服务器响应 (从连接获取到输入流并包装为bufferedReader)
    int status = conn.getResponseCode();
    logger.warning("status:" + status);
    if (status != 200) {
        logger.warning("OK,SUCESSNo Content");
        if (status == 201) {
            logger.warning("SUCESS BUT No Content");
            is = conn.getInputStream();
        } else if (status == 204) {
            logger.warning("err");
            is = conn.getInputStream();
        } else { // 400/401
            is = conn.getErrorStream();
        }
    } else { // 200
        logger.warning("OK,SUCESS");
        is = conn.getInputStream();
    }
}
```

```
BufferedReader br = new BufferedReader(new InputStreamReader(is, "UTF_8"));
    String line;
    StringBuilder resultMsg = new StringBuilder();
    while ((line = br.readLine()) != null) { // 读取数据
        resultMsg.append(line);
    }

    // 关流，这个很重要
    is.close();
    br.close();

    // 关闭连接
    conn.disconnect();
    logger.warning("resultMsg.toString() " + resultMsg.toString());
    return null;
}

static void trustAllHttpsCertificates() throws Exception {
    TrustManager[] trustAllCerts = new TrustManager[] {
        new X509TrustManager() {
            @Override
            public void checkClientTrusted(X509Certificate[] chain, String authType) throws CertificateException {
                return;
            }

            @Override
            public void checkServerTrusted(X509Certificate[] chain, String authType) throws CertificateException {
                return;
            }
            @Override
            public X509Certificate[] getAcceptedIssuers() {
                return null;
            }
        }
    };
    SSLContext sc = SSLContext.getInstance("SSL");
    sc.init(null, trustAllCerts, null);
    HttpsURLConnection.setDefaultSSLSocketFactory(sc.getSocketFactory());
}
}
```

表 2-4 代码样例中可变参数说明

参数名称	参数说明
url	请求地址
Map	封装请求的具体信息
Connection	封装建立连接的认证信息、请求方式等

## 消息样例

```
resultMsg.toString(){"senderAddress":"*****","senderName":"Default
Name","address":"*****","destinationAddress":"*****","clientCorrelator":"567895","outboundIMMessage":
{"contributionID":"c67771cf-
bd31-483e-8f8a-479df74db92e","storeSupported":"true","bodyText":"*****","reportRequest":
["Delivered","Displayed","Failed"],"conversationID":"fSFDSFDR$%#$%$%$%","subject":"Default
Subject","shortMessageSupported":"false","contentEncoding":"utf8","contentType":"multipart/mixed;
boundary=\\"next\\","serviceCapability":[{"capabilityId":"ChatbotSA","version":"+g.gsma.rcs.botversion=
\"#=1\""}, {"capabilityId":"ChatbotSA222","version":"+g.gsma.rcs.botversion=\"#=1\""}]}}
```

## 2.1.5 带建议回复消息的多卡片 5G 消息

### 代码样例

```
import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.MalformedURLException;
import java.net.URL;
import java.security.cert.X509Certificate;
import java.util.HashMap;
import java.util.Map;
import java.util.logging.Level;
import java.util.logging.Logger;

import javax.net.ssl.HostnameVerifier;
import javax.net.ssl.HttpsURLConnection;
import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;

import com.fasterxml.jackson.databind.ObjectMapper;

import static java.nio.charset.StandardCharsets.UTF_8;

/**
 * 功能描述
 */
public class FivegMessageOutBoundMultiCardMessageWithResponse {
    // 带有值的返回成功状态码
    static final int SUCCESS_200_RES = 200;

    // 带有值的返回成功状态码
    static final int SUCCESS_201_RES = 201;

    // 不带值的返回成功状态码
    static final int SUCCESS_204_RES = 204;

    /**
     * 设置不验证主机
     */
    private static final HostnameVerifier DO_NOT_VERIFY = (hostname, session) -> true;

    private FivegMessageOutBoundMultiCardMessageWithResponse() {
    }

    /**
     * 多卡片发送消息
     *
     * @param args
     * @throws Exception
     */
    public static void main(String[] args) throws Exception {
        Logger logger =
Logger.getLogger(FivegMessageOutBoundMultiCardMessageWithResponse.class.getName());
        logger.setLevel(Level.WARNING);

        trustAllHttpsCertificates();
        URL realUrl = getUrl();

        HttpsURLConnection connection = (HttpsURLConnection) realUrl.openConnection();

        // 设置请求方式
    }
}
```

```
connection.setDoOutput(true);
connection.setHostnameVerifier(DO_NOT_VERIFY);
connection.setRequestMethod("POST");
connection.setRequestProperty("Authorization", "Username=\"chatbottest12swt96\", Password=\"*****\n\"");
connection.setRequestProperty("Content-Type", "application/json");
ObjectMapper objectMapper = new ObjectMapper();
PrintWriter pw = new PrintWriter(new OutputStreamWriter(connection.getOutputStream(), UTF_8));
pw.write(objectMapper.writeValueAsString(getMap()));
pw.flush();

connection.connect();
int status = connection.getResponseCode();
logger.warning("5GMessageResponse status: " + status);
InputStream is;
if (status != SUCCESS_200_RES) {
    if (status == SUCCESS_201_RES) {
        is = connection.getInputStream();
    } else if (status == SUCCESS_204_RES) {
        logger.warning("No Content Success");
        is = connection.getInputStream();
    } else { // error
        logger.warning("Error, DELETE status: " + status);
        is = connection.getErrorStream();
    }
} else { // 200
    logger.warning("OK,DELETE Success, DELETE status: " + status);
    is = connection.getInputStream();
}

if (is != null) {
    BufferedReader br;
    br = new BufferedReader(new InputStreamReader(is, UTF_8));
    String line;
    StringBuilder result = new StringBuilder();

    // 读取数据
    while ((line = br.readLine()) != null) {
        result.append(line + System.lineSeparator().intern());
    }
    br.close();
    logger.warning("Content: " + result.toString());

    // 关流
    is.close();
}

// 关闭连接
connection.disconnect();
}

/**
 * 获取URL
 *
 * @return url
 * @throws MalformedURLException
 */
public static URL getUrl() throws MalformedURLException {
    URL realUrl;

    // CSP接口 不需要sag带tid字段
    String cspUrl = "https://10.120.207.128:18323/openchatbot/v2/"
        + "sip:888.chatbot@botplatform.rcs.chinamobile.com/outbound";

    return new URL(cspUrl);
}

/**
```



```
        return;
    }

    @Override
    public void checkServerTrusted(X509Certificate[] chain, String authType) {
        return;
    }

    @Override
    public X509Certificate[] getAcceptedIssuers() {
        return null;
    }
}

SSLContext sc = SSLContext.getInstance("SSL");
sc.init(null, trustAllCerts, null);
HttpsURLConnection.setDefaultSSLSocketFactory(sc.getSocketFactory());
}
```

表 2-5 代码样例中可变参数说明

参数名称	参数说明
url	请求地址
Map	封装请求发送的信息、参数
Connection	封装建立连接的认证信息、请求方式等

## 消息样例

```
resultMsg.toString(){"senderAddress":"*****","senderName":"Default Name","address":"*****","destinationAddress":"*****","clientCorrelator":"567895","outboundIMMessage":{"contributionID":"c67771cf-bd31-483e-8f8a-479df74db92e","storeSupported":"true","bodyText":"*****","reportRequest":["Delivered","Displayed","Failed"],"conversationID":"fSFDSFDR$%#$%$%$%","subject":"Default Subject","shortMessageSupported":"false","contentEncoding":"utf8","contentType":"multipart/mixed; boundary=\\"next\\","serviceCapability":[{"capabilityId":"ChatbotSA","version":"+g.gsma.rcs.botversion="#=1"}, {"capabilityId":"ChatbotSA222","version":"+g.gsma.rcs.botversion="#=1"}]}}
```

### 2.1.6 码流转成 json 的多卡片+悬浮菜单

#### 代码样例

```
import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.MalformedURLException;
import java.net.URL;
import java.security.cert.X509Certificate;
import java.util.HashMap;
import java.util.Map;
import java.util.logging.Level;
import java.util.logging.Logger;

import javax.net.ssl.HostnameVerifier;
import javax.net.ssl.HttpsURLConnection;
import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
```

```
import javax.net.ssl.X509TrustManager;

import com.fasterxml.jackson.databind.ObjectMapper;

import static java.nio.charset.StandardCharsets.UTF_8;

/**
 * 多卡片发送消息悬浮菜单
 */
public class FivegMessageOutBoundMultiCardAndFloatMenu {
    // 带有值的返回成功状态码
    static final int SUCCESS_200_RES = 200;

    // 带有值的返回成功状态码
    static final int SUCCESS_201_RES = 201;

    // 不带值的返回成功状态码
    static final int SUCCESS_204_RES = 204;

    /**
     * 设置不验证主机
     */
    private static final HostnameVerifier DO_NOT_VERIFY = (hostname, session) -> true;

    private FivegMessageOutBoundMultiCardAndFloatMenu() {
    }

    /**
     * 多卡片发送消息悬浮菜单
     *
     * @param args
     * @throws Exception
     */
    public static void main(String[] args) throws Exception {

        Logger logger = Logger.getLogger(FivegMessageOutBoundMultiCardMessage.class.getName());
        logger.setLevel(Level.WARNING);

        trustAllHttpsCertificates();
        URL realUrl = getUrl();

        HttpsURLConnection connection = (HttpsURLConnection) realUrl.openConnection();

        // 设置请求方式
        connection.setDoOutput(true);
        connection.setHostnameVerifier(DO_NOT_VERIFY);
        connection.setRequestMethod("POST");
        connection.setRequestProperty("Authorization", "Username=\"chatbottest12swt96\", Password=\"*****");
        connection.setRequestProperty("Content-Type", "application/json");
        ObjectMapper objectMapper = new ObjectMapper();
        PrintWriter pw = new PrintWriter(new OutputStreamWriter(connection.getOutputStream(), UTF_8));
        pw.write(objectMapper.writeValueAsString(getMap()));
        pw.flush();

        connection.connect();
        int status = connection.getResponseCode();
        logger.warning("5GMessageResponse status: " + status);
        InputStream is;
        if (status != SUCCESS_200_RES) {
            if (status == SUCCESS_201_RES) {
                is = connection.getInputStream();
            } else if (status == SUCCESS_204_RES) {
                logger.warning("No Content Success");
                is = connection.getInputStream();
            } else { // error
                logger.warning("Error, DELETE status: " + status);
            }
        }
    }
}
```

```
        is = connection.getErrorStream();
    }
} else { // 200
    logger.warning("OK,DELETE Success, DELETE status: " + status);
    is = connection.getInputStream();
}

if (is != null) {
    BufferedReader br;
    br = new BufferedReader(new InputStreamReader(is, UTF_8));
    String line;
    StringBuilder result = new StringBuilder();

    // 读取数据
    while ((line = br.readLine()) != null) {
        result.append(line + System.lineSeparator().intern());
    }
    br.close();
    logger.warning("Content: " + result.toString());

    // 关流
    is.close();
}

// 关闭连接
connection.disconnect();
}

/***
 * 获取URL
 *
 * @return url
 * @throws MalformedURLException
 */
public static URL getUrl() throws MalformedURLException {
    URL realUrl;

    // CSP接口 不需要sag带tid字段
    String cspUrl = "https://10.120.207.128:18323/openchatbot/v2/"
        + "sip:888.chatbot@botplatform.rcs.chinamobile.com/outbound";

    return new URL(cspUrl);
}

/***
 * 返回Body体
 *
 * @return Body
 */
public static Map<String, Object> getMap() {
    Map<String, Object> map = new HashMap<>();
    Map<String, Object> childMap = new HashMap<>();

    childMap.put("imFormat", "IM");
    childMap.put("contentType", "multipart/mixed; boundary=\\"next\\\"");
    childMap.put("bodyText", "<![CDATA[--next\r\nContent-Type: application/vnd.gsma.botmessage.v1.0+json\r\nTransfer-Encoding: chunked\r\n\r\n    \"generalPurposeCardCarousel\"\r\n        \"layout\": [\r\n            \"cardOrientation\": \"VERTICAL\", \r\n            \"imageAlignment\": \"LEFT\"\r\n        ],\r\n        \"titleFontStyle\": [\r\n            \"underline\", \r\n            \"bold\"\r\n        ],\r\n        \"descriptionFontStyle\": [\r\n            \"calibri\", \r\n            \"content\"\r\n        ],\r\n        \"media\": [\r\n            \"mediaUrl\": \"http://120.198.247.156:9090/Access/PF?ID=MzdCQjE4RjcwQzM0RkZENzkyNklyNEI3MDUwRDAwOUlyOUQ3Qzk2QTNBQklzQzl2QTE2RDQ3OEZGOTY1NEE3Qk5NDQwNDNGNjc3RDE1NzU3OExxE0E3ODEzQjNDMDYw\", \r\n            \"mediaContentType\": \"image/png\", \r\n            \"thumbnailUrl\": \"http://120.198.247.156:9090/Access/PF?\r\nID=QjZBQjE0QjUxRjA4NEZFRjhDQ0MyRkE4MzhENTM1Mzc5MTY5RkM0Qjk0QzgzREM5ODZGQzk1QUM1QkJGnkVGNgQjQ3RDYwNURDQkl3MTMwQjU2QUVEOTQ4OUM0NjA5N0M1\", \r\n            \"thumbnailContentType\": \"image/png\", \r\n            \"height\": \"MEDIUM HEIGHT\", \r\n            \"width\": \"MEDIUM WIDTH\"\r\n        ]\r\n    ]\r\n}\r\n");\r\n    map.put("bodyText", childMap);
}
```



```
    @Override
    public X509Certificate[] getAcceptedIssuers() {
        return null;
    }
}
SSLContext sc = SSLContext.getInstance("SSL");
sc.init(null, trustAllCerts, null);
HttpsURLConnection.setDefaultSSLSocketFactory(sc.getSocketFactory());
}

}
```

表 2-6 代码样例中可变参数说明

参数名称	参数说明
url	请求地址
Map	封装请求发送的信息、参数
Connection	封装建立连接的认证信息、请求方式等

## 消息样例

```
resultMsg.toString(){"senderAddress":"*****", "senderName":"Default
Name", "address":"*****", "destinationAddress": "*****", "clientCorrelator": "567895", "outboundIMMessage":
{"contributionID": "c67771cf-
bd31-483e-8f8a-479df74db92e", "storeSupported": "true", "bodyText": "*****", "reportRequest": [
"Delivered", "Displayed", "Failed", "SMS"], "conversationID": "fSFDSFDR$%#$%$%", "subject": "Default
Subject", "shortMessageSupported": "false", "contentEncoding": "utf8", "contentType": "multipart/mixed;
boundary=\\"next\\\"", "serviceCapability": [{"capabilityId": "ChatbotSA", "version": "+"+g.gsma.rcs.botversion=
"\\"#=1\\\"", "capabilityId": "ChatbotSA222", "version": "+"+g.gsma.rcs.botversion="\\"#=1\\\""}]}
```

# 3 错误码

## HTTP 错误码

序号	错误码	描述
1	200	OK, GET请求成功，响应消息有消息体。
2	201	Created, POST/PUT请求成功，响应消息有消息体。
3	204	No Content, 成功，响应消息没有消息体。
4	206	OK, GET请求成功， Partial Content
5	400	Bad Request , 客户端请求的语法错误，服务器无法理解
6	403	Forbidden, 服务器拒绝执行
7	404	Not Found, 服务器无法根据客户端的请求找到资源
8	406	Not Acceptable, 服务器无法根据客户端请求的内容特性完成请求
9	408	Request Timeout , 请求超时
10	503	Server Unavailable, 服务器没有准备好处理请求

## Network API 错误码

序号	HTTP错误码	Network API错误码	描述
1	200	SVC1000	Success -成功
2	403	SVC4001	query user info fail -查询用户信息失败
3	400	SVC0002	msg decode fail -解码失败
4	500	SVC4001	require apurce fail -申请资源失败

序号	HTTP错误码	Network API错误码	描述
5	501	SVC4001	query file status fail -查询文件状态失败
6	500	SVC4001	encode fail -编码失败
7	500	SVC4001	base64 decode fail -Base64解码失败
8	500	SVC4001	generate file path fail -生成文件路径失败
9	500	SVC4001	get file info fail -获取文件信息失败
10	500	SVC4001	get file domain index fail -获取域索引失败
11	500	SVC4001	insert databases fail -写入数据库失败
12	500	SVC4001	get httpserver node fail -查询文件服务器节点失败
13	500	SVC4001	internal communication fail -内部通信失败
14	500	SVC4001	file transfer fail -文件传输失败
15	500	SVC4001	illegal platform identity -非法的平台标识
16	403	POL0009	Maximum number of requests exceeded -消息流控
17	400	SVC0002	AO msg service capability invalid -应用发起消息所带业务标识校验失败
18	603	SVC4001	file size overlimit -文件大小超过平台限制
19	500	SVC4001	query session infomatrion timeout -从查询会话信息超时
20	500	SVC4001	insert session infomatrion timeout -记录会话信息超时
21	500	SVC4001	query report relationship timeout -查询回执对应关系超时
22	403	SVC5201	service is shutdown -用户已停机
23	500	SVC4001	when get shorturl, msg expire -获取短链时，消息已经过期
24	400	SVC4001	a2p message allow fallback sms, but sms port number not exist -MaaP消息允许回落短信，但短信端口号不存在
25	400	SVC4001	a2p message allow fallback sms, but sms content not exist -MaaP消息允许回落短信，但短信内容不存在
26	500	SVC4001	query report relationship failed -查询回执对应关系失败

序号	HTTP错误码	Network API错误码	描述
27	400	SVC4001	Msg Recived has expired -接收到的消息已过期
28	400	SVC4001	A2P message can not fall back to mms or sms -消息无法回落
29	403	SVC4001	query user np info fail -查询用户NP信息失败
30	500	SVC4001	msg expired -消息过期
31	404	SVC5001	dest user account not exist -被叫用户信息不存在
32	500	SVC4001	query dest userinfo failed -被叫用户信息查询失败
33	480	SVC4001	dest user offline -被叫用户离线
34	500	SVC4001	get remote file failed -获取远端文件响应失败
35	500	SVC4001	query original message failed -从数据库查询原始消息失败
36	400	SVC0002	invalid method or url -Maap请求method/url非法
37	400	SVC0002	invalid AS request -AS请求非法
38	400	SVC0002	parse xml fail -xml解析失败
39	500	SVC0001	bulid xml fail -xml构造失败
40	400	SVC0003	missing necessary field -缺少必填字段
41	403	POL0009	flow control interception -流控拦截
42	500	SVC0001	set to db fail -写库失败
43	500	SVC0001	get from db fail -读库失败
44	500	SVC0001	set to db timeout -写库超时
45	500	SVC0001	get from db timeout -读库超时
46	404	SVC0004	send request to as fail -发送至AS失败
47	408	SVC0001	AS response timeout -AS响应超时
48	503	SVC2001	alloc memory fail -资源分配失败
49	500	SVC2001	put into hash fail -会话信息保存失败
50	500	SVC2001	get from hash fail -会话信息读取失败
51	500	SVC2001	invalid http link -http链路异常
52	500	SVC0001	interface inner error -接口内部错误

序号	HTTP错误码	Network API错误码	描述
53	404	SVC0004	send request to ap fail -发送至Maap失败
54	408	SVC0001	Maap response timeout -Maap响应超时
55	400	SVC0002	Http response xml parse fail -Http响应xml解析失败
56	400	SVC0002	Maap response invalid -Maap响应非法
57	400	SVC0002	request field invalid -请求字段非法
58	NA	SVC6001	peer node unreachable or no usable node -对端节点不可及/无可用节点 ( SMS )
59	NA	SVC6001	send message to peer ndoe failed -向短信中心节点投递失败
60	NA	SVC6001	routing failed -路由失败 ( SMS )
61	NA	SVC6001	VMSC:limited due to MS roaming -用户漫游，无法接收
62	NA	SVC6001	service barred 或 operation barred -用户停机
63	NA	SMS	sms success report -回落短信成功
64	NA	SVC6001	Unknown Subscriber -短信下发返回未知用户错误。
65	NA	SVC6001	Unmarked Subscriber -短信下发返回未定义用户错误。
66	NA	SVC6001	HLR:illegal subscriber -短信下发返回非法用户错误。
67	NA	SVC6001	teleservice not supported -短信下发返回电信业务不支持错误。
68	NA	SVC6001	call forbidden -短信下发返回呼叫被禁止错误。
69	NA	SVC6001	VMSC:subscriber absent -短信下发返回用户不在服务区错误。
70	NA	SVC6001	HLR:resource not available due to congestion -短信下发返回HLR消息等待队列满错误。
71	NA	SVC6001	HLR:data lacked -短信下发返回数据丢失错误。
72	NA	SVC6001	HLR:unexpected data -短信下发返回意外数据错误。
73	NA	SVC6001	MS error -短信下发返回MS端错误。
74	NA	SVC6001	SMS not supported by MS -短信push回执-下发返回MS未装备错误。

序号	HTTP错误码	Network API错误码	描述
75	NA	SVC6001	MS store is full -短信下发返回手机内存满错误。
76	NA	SVC6001	illegal device -短信下发返回非法设备错误。
77	NA	SVC6001	failure because the subscriber is busy -短信下发返回用户忙错误。
78	NA	SVC6001	VMSC:the called MS poweroff -短信下发返回用户关机错误。
79	NA	SVC6001	VMSC:no response -短信下发后， MSC无应答。
80	NA	SVC6001	HLR:no response -短信下发后， HLR无应答。
81	NA	SVC6001	VMSC:the service not supported by the peer -短信下发后， MSC拒绝。
82	NA	SVC6001	HLR:the service not supported by the peer -短信下发后， HLR拒绝。
83	NA	SVC6001	HLR:service not complete -短信下发返回HLR系统错误。
84	NA	SVC6001	VMSC:service not complete -短信下发返回MSC系统错误.
85	NA	SVC6001	HLR:illegal destination SCCP address -短信下发后，由于目的信令点或信令转接点SCCP无法传送该消息。
86	NA	SVC6001	sms report-peer node unreachable or no usable node -短信回执对端节点不可及/无可用节点 ( SMS )
87	NA	SVC6001	sms report-send message to peer node failed -短信回执向短信中心节点投递失败
88	NA	SVC6001	sms report-message routing failed -短信回执路由失败 ( SMS )
89	NA	SVC6001	sms report-VMSC:limited due to MS roaming -短信回执用户漫游，无法接收
90	NA	SVC6001	sms report-service barred 或 sms report-operation barred -短信回执用户停机
91	NA	SVC6001	Unknown Subscriber 或 sms report-Unknown Subscriber -短信回执返回未知用户错误。
92	NA	SVC6001	sms report-Unmarked Subscriber -短信回执返回未定义用户错误。
93	NA	SVC6001	sms report-HLR:illegal subscriber -短信回执返回非法用户错误。

序号	HTTP错误码	Network API错误码	描述
94	NA	SVC6001	sms report-teleservice not supported -短信回执返回电信业务不支持错误。
95	NA	SVC6001	sms report-call forbidden -短信回执返回呼叫被禁止错误。
96	NA	SVC6001	sms report-VMSC:subscriber absent -短信回执返回用户不在服务区错误。
97	NA	SVC6001	sms report-HLR:resource not available due to congestion -短信回执返回HLR消息等待队列满错误。
98	NA	SVC6001	sms report-HLR:data lacked -短信回执返回数据丢失错误。
99	NA	SVC6001	sms report-HLR:unexpected data -短信回执返回意外数据错误。
100	NA	SVC6001	sms report-MS error -短信回执返回MS端错误。
101	NA	SVC6001	sms report-SMS not supported by MS -短信回执-下发返回MS未装备错误。
102	NA	SVC6001	sms report-MS store is full -短信回执返回手机内存满错误。
103	NA	SVC6001	sms report-illegal device -短信回执返回非法设备错误。
104	NA	SVC6001	sms report-failure because the subscriber is busy -短信回执返回用户忙错误。
105	NA	SVC6001	sms report-VMSC:the called MS poweroff -短信回执返回用户关机错误。
106	NA	SVC6001	sms report-VMSC:no response -短信回执， MSC无应答。
107	NA	SVC6001	sms report-HLR:no response -短信回执， HLR无应答。
108	NA	SVC6001	sms report-VMSC:the service not supported by the peer -短信回执， MSC拒绝。
109	NA	SVC6001	sms report-HLR:the service not supported by the peer -短信回执， HLR拒绝。
110	NA	SVC6001	sms report-HLR:service not complete -短信回执返回HLR系统错误。
111	NA	SVC6001	sms report-VMSC:service not complete -短信回执返回MSC系统错误.

序号	HTTP错误码	Network API错误码	描述
112	NA	SVC6001	sms report-HLR:illegal destination SCCP address -短信回执，
113	400	SVC0003	invalid maap platform id -因在AO接口产生发送方鉴权失败
114	400	SVC4001	receiver not local area user in application platform send process -在AO接口产生接收方鉴权失败
115	400	SVC0002	AO msg service capability invalid -因在AO接口产生5G消息系统拒绝服务
116	400	SVC4001	a2p msg can not fallback xms -在AO接口，接收方为系统黑名单用户
117	NA	SVC1501	no capability -无能力
118	403	POL0003	dest address out of bounds -由于SP群发地址过多，超过系统设置从而导致失败