movistar

API
Mensajería Negocios
v3.0

Índice de contenidos

1 INTRODUCCIÓN

Para poder hacer uso del servicio Mensajería Negocios sin tener que acceder a través del portal Web existen dos opciones u APIs diferenciados:

- Protocolo de comunicación XML-RPC

  Es el primer API del que dispuso el servicio, que permite acceder a la mayoría de las funcionalidades de Mensajería Negocios

- Interfaz simplificada

  Es un interfaz más moderno que facilita el envío de mensajes y la recepción de los estados resultantes de los envíos; es una versión simplificada y compatible del servicio Mensajería Integrada de Telefónica y por ello se recomienda su implantación si se prevé que pueda llegar a necesitarse alguna de las capacidades avanzadas www.movistar.es/grandes-empresas/soluciones/fichas/mensajeria-integrada.

En ambos casos se necesita una URL de conexión, que varía de un método al otro, y unas credenciales para el API, que son las mismas y se configuran en la web del servicio.

- **Usuario de acceso**: Se trata del usuario que establece el usuario gestor a través del menú Usuario Api de la Aplicación [1]. Este usuario puede cambiarse posteriormente a través del mismo menú.
- **Contraseña**: Contraseña asociada al usuario API, se establece de igual forma que el usuario de acceso.

Para cambiar la contraseña posteriormente es necesario conocer la contraseña previa, aunque puede ser consultada pulsando el botón enviar del mismo menú, y se enviará al destinatario de email y sms definido en el contacto.

1.1 Protocolo de comunicación XML-RPC

El protocolo de comunicación XML-RPC proporciona la posibilidad de hacer un uso del servicio Mensajería Negocios sin tener que acceder a través del portal Web.

Mediante este acceso se pueden realizar conexiones directas entre sus máquinas y los servidores de Mensajería Negocios.

En el apartado 2 se especifica cómo acceder a las siguientes funcionalidades a través del API:

[1] Los usuarios autorizados (no gestores) que se pueden crear a través de la web del servicio no pueden utilizarse como usuarios de acceso al API.

- Envío Libre
- Envío a Grupo
- Crear un nuevo Grupo de Contactos
- Añadir Contacto a Grupo
- Obtener los números de teléfono de los Contactos de un Grupo
- Obtener el número de Contactos que tiene un Grupo
- Obtener los mensajes recibidos en el Buzón de Entrada
- Obtener los mensajes enviados y su estado

Además podrá encontrar toda la información necesaria para realizar llamadas XML-RPC a través del API y ejemplos de implementación de clientes, tanto en Java como en PHP.

Para hacer un uso adecuado del API es necesaria la siguiente información:

- **URL de conexión** (URL a la que realizar las peticiones XML-RPC): Debe conectarse a la siguiente dirección utilizando conexión segura (acceso HTTPS):

  https://www.mensajerianegocios.movistar.es/SrvConexion

1.2 Interfaz Simplificada

La Interfaz Simplificada proporciona la posibilidad de hacer un uso del servicio Mensajería Negocios sin tener que acceder a través del portal Web.

Mediante este acceso se pueden realizar conexiones directas entre sus máquinas y los servidores de Mensajería Negocios.

Mediante el Interfaz Simplificado se puede acceder a servicios de mensajería SMS y WAPPUSH. El Interfaz Simplificado se describe en los documentos WSDL de forma que los Proveedores de Servicios puedan realizar fácilmente la integración con servicios de la plataforma de mensajería.

Esta guía tiene como objetivo describir el formato de los diferentes mensajes soportados con el Interfaz Simplificado. Se complementa la guía con una serie de ejemplos de escenarios de utilización.

Igualmente, se describen los procesos de instalación y configuración del Interfaz Simplificado. Para una mejor comprensión del presente documento, se recomienda tener un conocimiento previo del protocolo SOAP.

El Interfaz Simplificado es un servicio web que permite, mediante su WSDL, la publicación de los distintos servicios de mensajería.

Este WSDL permite la generación de aplicaciones/servicios que facilitan el acceso a la plataforma de mensajería con independencia de la plataforma del cliente donde se ejecuten y el lenguaje de programación que se utilice para su generación.

Sin embargo del nuevo Interfaz descrito en el WSDL sólo están soportado el envío de mensajes MT (Mobile Terminated) procedentes de aplicaciones/servicios con destino a un usuario.

La utilización de funcionalidad adicional contemplada en todos los protocolos de mensajería, como entrega diferida, período de validez, notificación de entrega, etc.

Para hacer un uso adecuado de esta Interfaz Simplificada es necesaria la siguiente información:

- **URL de conexión** (URL a la que realizar las peticiones): Debe conectarse a la siguiente dirección utilizando conexión segura (acceso HTTPS):

  https://www.mensajerianegocios.movistar.es/ISServer/services/MI-InterfazSimplificado-Port

2 API DE COMUNICACIÓN XML-RPC

A continuación se describen los métodos remotos disponibles para conectarse al servidor de envío de mensajes de Mensajería Negocios.

Para mayor información acerca del formato de los parámetros de la llamada consultar *apartado 2.3 Formato de los parámetros para las llamadas XML-RPC*.

- **URL de conexión** (URL a la que realizar las peticiones XML-RPC): Debe conectarse a la siguiente dirección utilizando conexión segura (acceso HTTPS):

  https://www.mensajerianegocios.movistar.es/SrvConexion

2.1.1 Envío Libre

Este método permite el envío de mensajes SMS personalizando el mensaje para cada destinatario, es decir, para cada destinatario se indicará el texto a enviar y el remitente (217812, Remitente de texto configurado o 638444812) a utilizar.

Las características más importantes de este tipo de envío son:

- Los destinatarios de los mensajes deben estar almacenados en sus sistemas, es decir, en la aplicación que invoca al API.
- Cada operación de envío permite enviar un máximo de 500 SMS.

Las especificaciones de la llamada son las siguientes:

Nombre del método remoto

MensajeriaNegocios.enviarSMS

Parámetros de entrada

El parámetro de entrada es una lista con los siguientes valores:

- **Nombre de Usuario**: usuario de acceso (se corresponde con el usuario proporcionado por Telefónica para acceder al servicio).
- **Contraseña**: Contraseña asociada al usuario API (clave proporcionada al acceder al servicio y que el usuario gestor puede cambiar).
- **Lista de mensajes**, donde cada mensaje es una lista con los siguientes campos:
  - Teléfono, es el destinatario del mensaje.
  - Texto del mensaje a enviar.

- Remitente del mensaje, el numérico de la aplicación 217812 o uno de los alfanuméricos asignados.

Ejemplo: Una lista de SMS podría estar formada por SMS1 y SMS2, de manera que:

SMS1:

- Destinatario: 609000001
- Texto: Texto SMS 1
- Remitente: 217812
- Remitente Alfanumérico Principal: MiRPrincipal

SMS2:

- Destinatario: 609000003
- Texto: Texto SMS 2
- Remitente: MiRemitente

➢ El SMS 1 se enviaría al destinatario 609000001, que recibiría el texto 'Texto SMS 1 Fdo. MiRPrincipal' por el remitente 217812.

➢ El SMS2 se enviaría al destinatario 609000003, que recibiría el texto 'Texto SMS 2' por el remitente MiRemitente.

Códigos de retorno

Esta llamada devuelve un código de error o una lista de valores (ver tabla del *Anexo 2.4*):

- Si la llamada se ejecuta con éxito, devuelve una lista de valores que representa que se ha realizado el envío de cada uno de los SMS indicados como parámetro en la lista de mensajes. Cada uno de los elementos de la lista son códigos numéricos que representan el código de retorno de cada mensaje

  Ejemplo: Si en la llamada incluimos 3 mensajes, los dos primeros son correctos y el último da error, nuestro retorno sería (0,0,4).

- Caso contrario, devuelve un código de error numérico representa que se ha producido alguna anomalía que ha imposibilitado el envío de todos y cada uno de los mensajes pasados como parámetro en la lista de mensajes.

  Ejemplo: Si en la llamada indicamos un usuario de acceso al API no válido, recibiremos el código de error numérico -2, y el envío no se habrá realizado.

2.1.2 Envío a Grupo de Contactos

Este método permite el envío de un mensaje a un Grupo de Contactos. Todos los contactos del grupo recibirán el mismo mensaje.

Las características más importantes de este tipo de envío son:

- El Grupo de Contactos destinatario del mensaje habrá sido creado previamente en Mensajería Negocios, bien a través del API, bien a través de la página web del servicio. Es decir, independientemente de la forma de creación del grupo, los números de teléfono de dichos contactos se obtienen de la base de datos de Mensajería Negocios.
- El mismo mensaje es enviado a todos los contactos del grupo, con el mismo contenido y el mismo remitente.
- Este mensaje se envía a todos los contactos del grupo y al propio administrador.

Las especificaciones de la llamada son las siguientes:

Nombre del método remoto

MensajeriaNegocios.enviarAGrupoContacto

Parámetros de entrada

El parámetro de entrada es una lista con los siguientes valores y en el siguiente orden:

- **Nombre de Usuario**: usuario de acceso (se corresponde con el usuario configurado en la aplicación para acceder al servicio vía API).
- **Contraseña**: Contraseña asociada al usuario API (clave que el usuario gestor establece en la aplicación y que él mismo puede consultar o cambiar).
- **Nombre del Grupo** al que enviar el mensaje
- **Texto** del mensaje
- **Remitente** del mensaje

Códigos de retorno

Esta llamada devuelve un valor numérico que corresponde a un código de retorno de la tabla del *Anexo 2.4*. El código de retorno indica si la operación de envío de grupo ha sido aceptada por el servidor de Mensajería Negocios. El resultado del envío a cada destinatario del grupo se puede visualizar a través de la página Web del servicio.

2.1.3 Agregar nuevo Grupo de Contactos

Este método permite crear un nuevo Grupo de Contactos en Mensajería Negocios.

Las especificaciones de la llamada son las siguientes:

Nombre del método remoto:

MensajeriaNegocios.agregarGrupoContactos

Parámetros de entrada

El parámetro de entrada es una lista con los siguientes valores y en el siguiente orden:

- **Nombre de Usuario**: usuario de acceso (se corresponde con el usuario configurado en la aplicación para acceder al servicio vía API).
- **Contraseña**: Contraseña asociada al usuario API (clave que el usuario gestor establece en la aplicación y que él mismo puede consultar o cambiar).
- **Nombre del Grupo** a crear
- **Mensaje de Bienvenida** que se envía a un nuevo contacto del grupo que se suscribe en el mismo. Este mensaje tiene un máximo de 160 caracteres y se envía con el remitente alfanumérico asignado al cliente.

Códigos de retorno

Esta llamada devuelve un valor numérico que corresponde a un código de retorno de la tabla del *Anexo 2.4*.

2.1.4 Agregar Contacto a Grupo

Este método permite incluir un nuevo contacto en un Grupo de Contactos existente en Mensajería Negocios.

Las especificaciones de la llamada son las siguientes:

Nombre del método remoto

MensajeriaNegocios.agregarContactoaGrupoContactos

Parámetros de entrada

El parámetro de entrada es una lista con los siguientes valores y en el siguiente orden:

- **Nombre de Usuario**: usuario de acceso (se corresponde con el usuario configurado en la aplicación para acceder al servicio vía API).
- **Contraseña**: Contraseña asociada al usuario API (clave que el usuario gestor establece en la aplicación y que él mismo puede consultar o cambiar).
- **Nombre del Grupo** al que se quiere añadir el Contacto
- **Número de teléfono** del contacto a añadir

Códigos de retorno

Esta llamada devuelve un valor numérico que corresponde a un código de retorno de la tabla del *Anexo 2.4*

2.1.5 Obtener números de teléfono de Contactos de un Grupo

Este método permite consultar los números de teléfono de los Contactos pertenecientes a un Grupo de Contactos en Mensajería Negocios.

Las especificaciones de la llamada son las siguientes:

Nombre del método remoto

MensajeriaNegocios.obtenerContactosdeGrupoContactos

Parámetros de entrada

El parámetro de entrada es una lista con los siguientes valores y en el siguiente orden:

- **Nombre de Usuario**: usuario de acceso (se corresponde con el usuario configurado en la aplicación para acceder al servicio vía API).
- **Contraseña**: Contraseña asociada al usuario API (clave que el usuario gestor establece en la aplicación y que él mismo puede consultar o cambiar).
- **Nombre del Grupo** a consultar

Códigos de retorno

Los posibles códigos de retorno son:

- Si la llamada se ejecuta con éxito, devuelve un número positivo que indica el número de contactos que tiene el grupo.

  Ejemplo: Devuelve 609000001 606000002 si el grupo tiene dos contactos.
- Caso contrario, devuelve un número negativo que corresponde a un código de retorno de la tabla del *Anexo 2.4*.

2.1.6 Obtener número de Contactos que tiene Grupo

Este método permite consultar el número de Contactos que tiene asociados un Grupo dado de alta en Mensajería Negocios.

Las especificaciones de la llamada son las siguientes:

Nombre del método remoto

MensajeriaNegocios.obtenerNumeroContactosdeGrupoContactos

Parámetros de entrada

El parámetro de entrada es una lista con los siguientes valores:

- **Nombre de Usuario**: usuario de acceso (se corresponde con el usuario configurado en la aplicación para acceder al servicio vía API).
- **Contraseña**: Contraseña asociada al usuario API (clave que el usuario gestor establece en la aplicación y que él mismo puede consultar o cambiar).
- **Nombre del Grupo** a consultar

Códigos de retorno

Los posibles códigos de retorno son:

- Si la llamada se ejecuta con éxito, devuelve un número positivo que indica el número de contactos que tiene el grupo.

  Ejemplo: Devuelve 5 si el grupo tiene 5 contactos.
- Caso contrario, devuelve un número negativo que corresponde a un código de retorno de la tabla del *Anexo 2.4*.

2.1.7 Obtener mensaje en el Buzón de Entrada

Este método permite leer los mensajes recibidos en el Buzón de Entrada.

Las especificaciones de la llamada son las siguientes:

Nombre del método remoto

MensajeriaNegocios.obtenerSMSRecibidos

Parámetros de entrada

El parámetro de entrada es una lista con los siguientes valores y en el siguiente orden:

- **Nombre de Usuario**: usuario de acceso (se corresponde con el usuario configurado en la aplicación para acceder al servicio vía API).

- **Contraseña**: Contraseña asociada al usuario API (clave que el usuario gestor establece en la aplicación y que él mismo puede consultar o cambiar).

- **Fecha Inicial** a consultar, ejemplo 2016-01-01 00:00:00

- **Fecha Final** a consultar, ejemplo 2016-02-01 00:00:00

N**ota:** El rango de fechas no puede ser superior a un día.

Códigos de retorno

Los posibles códigos de retorno son:

- Si la llamada se ejecuta con éxito, devuelve una array con la lista de mensajes recibidos en el buzón de entrada con el formato fecha;MSISDN;TEXTO;1/0. El valor final indica si el mensaje ha sido leído o no. 0=> No leído, 1=>Leído.

  Ejemplo respuesta:

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<methodResponse xmlns:ex="http://ws.apache.org/xmlrpc/namespaces/extensions">
 <params>
  <param>
   <value>
    <array>
     <data>
      <value>01/10/2013 12:35:07;34650424244;test;1</value>
      <value>01/10/2013 12:27:59;34650424244;test;0</value>
     </data>
    </array>
   </value>
  </param>
 </params>
</methodResponse>
```

- Caso contrario, devuelve un número negativo que corresponde a un código de retorno de la tabla del *Anexo **2.4***.

2.1.8 Obtener mensajes enviados y su estado

Este método permite obtener los mensajes enviados y su estado.

Las especificaciones de la llamada son las siguientes:

Nombre del método remoto

MensajeriaNegocios.obtenerSMSEnviados

Parámetros de entrada

El parámetro de entrada es una lista con los siguientes valores y en el siguiente orden:

- **Nombre de Usuario**: usuario de acceso (se corresponde con el usuario configurado en la aplicación para acceder al servicio vía API).
- **Contraseña**: Contraseña asociada al usuario API (clave que el usuario gestor establece en la aplicación y que él mismo puede consultar o cambiar).
- **Fecha Inicial** a consultar, ejemplo 2016-01-01 00:00:00
- **Fecha Final** a consultar, ejemplo 2016-02-01 00:00:00

N**ota:** El rango de fechas no puede ser superior a un día.

Códigos de retorno

Los posibles códigos de retorno son:

- Si la llamada se ejecuta con éxito, devuelve un array con la lista de mensajes enviados y su estado.

  Ejemplo respuesta:

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<methodResponse xmlns:ex="http://ws.apache.org/xmlrpc/namespaces/extensions">
 <params>
  <param>
   <value>
    <array>
     <data>
      <value>
       <array>
        <data>
         <value>MNALERTAS</value>
         <value>34699486685</value>
         <value>2013-01-11 02:00 Testing new probe</value>
         <value>ALERTAS</value>
         <value>11/01/13 00:00</value>
         <value>1</value>
         <value>DELIVRD</value>
        </data>
       </array>
      </value>
      <value>
     <value>
       <array>
        <data>
         <value>MNALERTAS</value>
         <value>34699486685</value>
         <value>2013-01-12 01:45 Testing new probe</value>
         <value>ALERTAS</value>
         <value>11/01/13 23:45</value>
```

```
            <value>1</value>
            <value>DELIVRD</value>
          </data>
        </array>
      </value>
    </data>
  </array>
 </value>
</param>
</params>
</methodResponse>
</methodResponse>
```

- Caso contrario, devuelve un número negativo que corresponde a un código de retorno de la tabla del *Anexo **2.4***.

2.2 Límite del número de conexiones desde la aplicación cliente

La aplicación únicamente puede establecer una conexión simultánea con los servidores de SMS de Mensajería Negocios. Además, **la comunicación** entre su aplicación y los servidores de Mensajería Negocios **es siempre síncrona**, es decir, sólo se puede realizar una petición simultánea al API, hasta que el API no devuelve respuesta a una operación la aplicación cliente no puede lanzar la siguiente petición.

Por ejemplo, si se quiere realizar un envío de 700 mensajes, considerando que el caudal máximo es de 500 SMS por envío, deberá realizar una primera conexión invocando al método *EnviarSMS* con los 500 primeros destinatarios. Una vez que el API haya devuelto el código de retorno que indique que los 500 envíos han finalizado, podrá realizar una segunda invocación para enviar los 200 mensajes restantes.

El caudal máximo es de 500 SMS por envío, es decir, se podrán realizar tantas operaciones de envío como se requiera pero en bloques de 500 SMS, y no se podrá enviar más hasta que finalice la conexión.

2.3 Formato de los parámetros para las llamadas XML-RPC

A continuación se describe la sintaxis que deben cumplir los distintos parámetros del sistema:

- **Nombre de Usuario**: usuario de acceso (se corresponde con el usuario configurado en la aplicación para acceder al servicio vía API).

- **Contraseña**: Contraseña asociada al usuario API (clave que el usuario gestor establece en la aplicación y que él mismo puede consultar o cambiar).

- **Nombre de Grupo**

  El nombre de un grupo de contactos permite los siguientes caracteres: 0123456789_ABCDEFGHIJKLMNÑOPQRSTUVWXYZabcdefghijklmnñopqrstuvwxyz.

  La longitud máxima es de 8 caracteres.

- **Mensaje de Bienvenida**

  El mensaje de bienvenida que recibe un usuario que se suscribe a un grupo de contactos tiene una longitud máxima de 160 caracteres.

  Los caracteres permitidos son los mismos que los permitidos para el texto de un mensaje enviado a través de API (descrito en este mismo apartado, en "Texto del mensaje").

- **Teléfonos o Contactos**

  Los teléfonos móviles permitidos en la aplicación, son exclusivamente los de **numeración española,** con o sin prefijo de España (34).

  Se define como numeración española aquellos teléfonos que cumplen las siguientes características:

  - Tienen una longitud de 9 dígitos, sin contar el prefijo
  - El primer dígito del número debe de ser 6 ó 7, sin contar el prefijo
  - Pueden ir precedidos del prefijo de España: 34

  Cualquier otro formato de numeración se toma como incorrecto.

  Ejemplo de numeración correcta:

  606000007 ó 34606000007 ó 706000007 ó 34706000007

- **Remitente del mensaje**

  El remitente de los SMS enviados puede ser de dos tipos:

  - **Numérico** por defecto: 217812
  - **Alfanumérico(s)** asignado(s) por Telefónica en el momento del alta de cliente. Existen tres tipos:
    - Remitente Alfanumérico Principal: Es obligatorio disponer al menos de un remitente de texto. Será utilizado como firma en los mensajes enviados con remitente 217812.
    - Remitentes Alfanuméricos Secundarios: Hasta 5 remitentes de texto opcionales.
    - Remitente largo: Igualmente se puede utilizar el número 638444812 como remitente.

Para seleccionar el remitente más adecuado para un envío recuerde elegir el 217812 si desea que quien lo reciba pueda contestar a dicho mensaje. Elija un remitente alfanumérico si no desea contestación.

- **Texto del mensaje**

  El contenido del mensaje de texto no debe exceder los 459 caracteres (3 SMS concatenados). Existen dos casos:

  - En caso de realizar el envío de un mensaje con remitente numérico, la longitud máxima del mensaje es de **459 caracteres**.
  - Sin embargo, en caso de realizar un envío con el remitente por defecto 217812, se añade automáticamente una firma al final del contenido del mensaje, por lo que se contabiliza la longitud de la firma junto con la longitud del mensaje. Es decir, si elige como remitente el remitente por defecto 217812 la longitud máxima de su texto podrá ser **459 - <longitud de la firma>.**

Por ejemplo:

Si el texto del mensajes es "Se adelanta la reunión a las 12 horas.", y el remitente alfanumérico principal que tiene asignado su empresa es "MiRemitente" y se realiza un envío con remitente numérico (217812), el mensaje que recibirán en sus teléfonos los clientes es:"Se adelanta la reunión a las 12 horas. Fdo: MiRemitente"

Los caracteres permitidos en los contenidos de los mensajes son:

01234567890
ABCDEFGHIJKLMNÑOPQRSTUVWXYZ
abcdefghijklmnñopqrstuvwxyz
ÇäöüÄÖÜÆÅå@£$¥Øø¡!"#
%&'()*+,-./:;<=>§¿?
y Espacio

Los caracteres no permitidos se convierten automáticamente a otros válidos siguiendo la siguiente tabla de conversiones:

Carácter	Conversión
Á	A
á	a
É	E
é	e
Í	I
í	i
Ó	O
ó	o
Ú	U
ú	u
à	a
è	e
ì	i
ò	o
ù	u
_	espacio
ç	c
^	‘
{	(
}	)
|	/
[	(
]	)
€	E

IMPORTANTE: Se debe utilizar la codificación LATIN-15 o ISO 8859-15, de no ser así, es muy posible que no se pueda enviar el mensaje.

2.4 Códigos de retorno

La siguiente tabla detalla el significado de los códigos de retorno de los métodos del API XML-RPC:

SIGNIFICADO	CÓDIGO DE RETORNO
Operación correcta La operación se ha realizado correctamente.	**0**
Contraseña incorrecta La contraseña introducida en la autenticación no es correcta, no corresponde al usuario de acceso indicado.	**-1**
Usuario incorrecto El usuario introducido en la autenticación no es un usuario válido o no es un administrador de la aplicación (los usuarios delegados no pueden acceder al API).	**-2**
Usuario bloqueado El usuario introducido en la autenticación está bloqueado o ha consumido 50 mensajes, por lo que tiene restringido el acceso al servicio.	**-3**
Error en la aplicación Ha ocurrido un error grave en la aplicación. Póngase en contacto con Atención Telefónica Movistar (900510041)	**-4**
Teléfono móvil incorrecto El teléfono introducido tiene sintaxis incorrecta, está repetido o pertenece a un gestor o usuario autorizado.	**-5**
Número destinatarios sobrepasado El número de destinatarios del mensaje excede el máximo número de destinatarios permitidos (máximo 500) en un solo envío.	**-6**
Lista de mensajes vacía La lista de mensajes SMS pasada como parámetro está vacía.	**-7**
Texto del mensaje - Longitud errónea La longitud del texto del mensaje es errónea, excede el número de caracteres permitidos o está vacío.	**-8**
Texto del mensaje - Sintaxis errónea El texto del mensaje contiene caracteres no permitidos.	**-9**

Fechas fuera de rango Se ha solicitado obtener mensajes del buzón de entrada con un rango de fechas superior a un día.	**-10**
Remitente erróneo El remitente utilizado es incorrecto o está vacío o no es uno de sus remitentes disponibles en el servicio.	**-11**
Grupo Al crear el grupo se ha indicado un nombre de grupo vacío, con sintaxis errónea o el grupo ya existe. Al modificar un grupo existente, se devuelve este código si el grupo no existe.	**-13**
Grupo – No contiene contactos El Grupo no tiene ningún contacto.	**-14**
Número de parámetros incorrecto En un envío libre, la lista de mensajes no está correctamente definida ya que el número de parámetros de alguno de los SMS es incorrecto. Revise que se ha definido para cada SMS su remitente, texto, destinatario y rango de fechas.	**-16**
Mensaje de Bienvenida incorrecto El mensaje de bienvenida contiene caracteres no permitidos o está vacío.	**-18**
Grupo o Mensaje de bienvenida erróneo El nombre de Grupo o el mensaje de bienvenida excede el máximo número de caracteres permitidos (8 y 160 caracteres respectivamente).	**-19**
Conexiones simultáneas al servidor de Mensajería Negocios No se pueden establecer dos conexiones simultáneas al sistema (ver *Anexo - Límite del número de conexiones desde la aplicación cliente*).	**-20**

2.5 Ejemplos de implementación

Para el correcto funcionamiento de los ejemplos indicados a continuación es necesario el uso de una librería XML-RPC tanto para Java como para PHP. Se pueden usar, por ejemplo, las siguientes (incluyendo sus dependencias si las hubiese):

- Java: *xmlrpc-3.1.3 jar* disponible en https://github.com/infusionsoft/API-Sample-Code/tree/master/Java/apache-xmlrpc-3.1.3
- PHP: *IXR_Library.inc.php* es la librería utilizada en estos ejemplos y actualmente no está soportada por los desarrolladores originales. Aun así, Telefónica pone la librería a disposición de los clientes en la url https://www.mensajerianegocios.movistar.es/IXR_Library.inc.php.gz

NOTA: Telefónica no facilita soporte sobre la librería en PHP. Si se desea utilizar un cliente no JAVA ni PHP consultar: http://xmlrpc.scripting.com/

2.5.1 Ejemplo Java Envío Libre

```
import java.net.URL;
import java.util.Vector;

import org.apache.xmlrpc.client.XmlRpcClient;
import org.apache.xmlrpc.client.XmlRpcClientConfigImpl;

public class EnvioSMS {
	/**
	 * @param args
	 */
	public static void main(String[] args) {
	String url_conexion=  "https://www.mensajerianegocios.movistar.es/SrvConexion";
		String login= "login"; //Nombre de usuario
		String password= "contrasena"; //Contraseña

		try {
			//Conectar con el Servidor
			XmlRpcClientConfigImpl config = new XmlRpcClientConfigImpl();
			  config.setServerURL(new URL(url_conexion));
			 config.setEncoding("ISO-8859-15");
			//Construir llamada
			Vector params = new Vector();
			//Primer parámetro: Login
			params.addElement(login);
			//Segundo parámetro: Contraseña
			params.addElement(password);

			//Composición de los SMS
			Vector SMS = new Vector();

			//Crear primer SMS
			Vector primer_SMS = new Vector();
			//Introducir telefono
			primer_SMS.addElement("609000001");
			//Introducir texto del mensaje
			primer_SMS.addElement("Texto primer mensaje");
			//Introducir remitente
			primer_SMS.addElement("miRemite");
```

```
                                //Incluir primer SMS en la lista
                                SMS.addElement(primer_SMS);

                                //Crear segundo SMS
                                Vector segundo_SMS = new Vector();
                                //Introducir telefono
                                segundo_SMS.addElement("609000003");
                                //Introducir texto del SMS
                                segundo_SMS.addElement("Texto segundo mensaje");
                                //Introducir remitente, elegimos remitente por defecto 217812
                                segundo_SMS.addElement("217812");
                                //Incluir segundo SMS en la lista
                                SMS.addElement(segundo_SMS);

                                //Introducir los SMS para la llamada
                                params.addElement(SMS);

                                //Realizar la petición al servidor
                                XmlRpcClient client = new XmlRpcClient();
                                 client.setConfig(config);
                                Object result = client.execute("MensajeriaNegocios.enviarSMS",
params);
                                //Imprimir los resultados
                                if(result instanceof Integer) {
                                        System.out.println(result);
                                }else{
                                        for(int i=0; i<((Object[])result).length;i++){
                                                System.out.println(((Object[])result)[i]);
                                        }
                                }
                                }catch (Exception e) {
                                        e.printStackTrace();
                                }


                }

        }
```

2.5.2 Ejemplo PHP Envío Libre

```
<?php
include ('IXR_Library.inc.php');  //Libreria para el uso de xml-rpc

$url_conexion='https://www.mensajerianegocios.movistar.es/SrvConexion';
$login='Login'; //Nombre de usuario
$password='Contrasena'; //Contraseña

$client = new IXR_Client($url_conexion);

//Ejecuta el metodo rpc
$client->query ('MensajeriaNegocios.enviarSMS', $login, $password,
array(array('606000001','Mensaje 1', 'miRemite'), array('606000003', 'Mensaje 2 ', '217812')));

echo '<pre>';
print_r($client->getResponse());
echo '</pre>';
?>
```

2.5.3 Ejemplo Java Envío a Grupo de Contactos

```
import java.net.URL;
import java.util.Vector;
import org.apache.xmlrpc.client.XmlRpcClient;
import org.apache.xmlrpc.client.XmlRpcClientConfigImpl;

public class EnvioGrupo {

	public static void main(String[] args) {
	String url_conexion= "https://www.mensajerianegocios.movistar.es/SrvConexion";
		String login= "login"; //Nombre de usuario
		String password= "contrasena"; //Contraseña
		String colectivo="Grupo"; //Nombre de Grupo
		String texto="Texto del mensaje"; //Texto del mensaje
		String remite="Remite del mensaje"; //Remite del mensaje, 217812 o remite de
usuario
		try {
			//Conectar con el Servidor
			XmlRpcClientConfigImpl config = new XmlRpcClientConfigImpl();
			config.setServerURL(new URL(url_conexion));
			 config.setEncoding("ISO-8859-15");
			//Construir llamada
			Vector params = new Vector();
			//Primer parámetro: Login
			params.addElement(login);
			//Segundo parámetro: Contraseña
			params.addElement(password);
			//Tercer parámetro: Grupo
			params.addElement(colectivo);
			//Cuarto parámetro: Mensaje de texto
			params.addElement(texto);
			//Quinto parámetro: Remite
			params.addElement(remite);
			//Realizar la petición al servidor
			XmlRpcClient client = new XmlRpcClient();
			 client.setConfig(config);
			Object result =
client.execute("MensajeriaNegocios.enviarAGrupoContacto", params);
			//Imprimir los resultados
			System.out.println("Resultado:"+result);

		}catch (Exception e) {
			e.printStackTrace();
		}
		}
	}
```

2.5.4 Ejemplo PHP Envío a Grupo de Contactos

```
<?php
include ('IXR_Library.inc.php');  //Libreria para el uso de xml-rpc

$url_conexion='https://www.mensajerianegocios.movistar.es/SrvConexion';
$login='Login'; //Nombre de usuario
$password='Contrasena'; //Contraseña
$remite='miRemite';//Remite 217812 o remite de usuario

$client = new IXR_Client($url_conexion);

//Ejecuta el metodo rpc
$client->query ('MensajeriaNegocios.enviarAGrupoContacto', $login, $password, 'Grupo',
'Mensaje a grupo', $remite);

echo '<pre>';
print_r($client->getResponse());
echo '</pre>';
?>
```

2.5.5 Ejemplo Java Agregar nuevo Grupo de Contactos

```
import java.net.URL;
import java.util.Vector;
import org.apache.xmlrpc.client.XmlRpcClient;
import org.apache.xmlrpc.client.XmlRpcClientConfigImpl;

public class AddGrupoContactos {
	/**
	 * @param args
	 */
	public static void main(String[] args) {
	String url_conexion= "https://www.mensajerianegocios.movistar.es/SrvConexion";
			String login= "login"; //Nombre de usuario
			String password= "contrasena"; //Contraseña
			String colectivo="Grupo"; //Nombre de Grupo
			String bienvenida="Texto de bienvenida"; //Texto bienvenida

			try {
				//Conectar con el Servidor
				XmlRpcClientConfigImpl config = new XmlRpcClientConfigImpl();
				 config.setServerURL(new URL(url_conexion));
				  config.setEncoding("ISO-8859-15");

				//Construir llamada
				Vector params = new Vector();
				//Primer parámetro: Login
				params.addElement(login);
				//Segundo parámetro: Contraseña
				params.addElement(password);
				//Tercer parámetro: Grupo
				params.addElement(colectivo);
				//Cuarto parámetro: Texto de bienevenida
				params.addElement(bienvenida);

				//Realizar la petición al servidor
				XmlRpcClient client = new XmlRpcClient();
				client.setConfig(config);
				Object result =
client.execute("MensajeriaNegocios.agregarGrupoContactos", params);
				//Imprimir los resultados
				System.out.println("Resultado:"+result);

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


	}
}
```

2.5.6 Ejemplo PHP Agregar nuevo Grupo de Contactos

```
<?php
include ('IXR_Library.inc.php');  //Libreria para el uso de xml-rpc

$url_conexion='https://www.mensajerianegocios.movistar.es/SrvConexion';
$login='Login'; //Nombre de usuario
$password='Contrasena'; //Contraseña

$client = new IXR_Client($url_conexion);

//Ejecuta el metodo rpc
$client->query ('MensajeriaNegocios.agregarGrupoContactos', $login, $password, , 'Grupo',
'Bienvenido al grupo');

echo '<pre>';
print_r($client->getResponse());
echo '</pre>'; ?>
```

2.5.7 Ejemplo Java Agregar Contacto a Grupo

```
import java.net.URL;
import java.util.Vector;
import org.apache.xmlrpc.client.XmlRpcClient;
import org.apache.xmlrpc.client.XmlRpcClientConfigImpl;

public class AddContactoaGrupo {
	/**
	 * @param args
	 */
	public static void main(String[] args) {
	  //URL de acceso al servicio
	  String url_conexion= "https://www.mensajerianegocios.movistar.es/SrvConexion";
		String login= "login"; //Nombre de usuario
		String password= "contrasena"; //Contraseña
		String colectivo="Grupo"; //Nombre de Grupo
		String movil="609000001";
		try {
			//Conectar con el Servidor
			XmlRpcClientConfigImpl config = new XmlRpcClientConfigImpl();
			config.setServerURL(new URL(url_conexion));
			 config.setEncoding("ISO-8859-15");

			//Construir llamada
			 Vector params = new Vector();
			 //Primer parámetro: Login
			 params.addElement(login);
			 //Segundo parámetro: Contraseña
			 params.addElement(password);
			 //Tercer parámetro: Grupo
			 params.addElement(colectivo);
			 //Cuarto parámetro: Móvil
			 params.addElement(movil);

			//Realizar la petición al servidor
			XmlRpcClient client = new XmlRpcClient();
			 client.setConfig(config);
			Object result =
client.execute("MensajeriaNegocios.agregarContactoaGrupoContactos",
					params);

			 System.out.println("Resultado:"+(Integer)result);

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


	}

}
```

2.5.8 Ejemplo PHP Agregar Contacto a Grupo

```
<?php
include ('IXR_Library.inc.php');  //Libreria para el uso de xml-rpc

$url_conexion='https://www.mensajerianegocios.movistar.es/SrvConexion';
$login='Login'; //Nombre de usuario
$password='Contrasena'; //Contraseña

$client = new IXR_Client($url_conexion);

//Ejecuta el metodo rpc
$client->query  ('MensajeriaNegocios.agregarContactoaGrupoContactos',  $login,  $password,
'Grupo', '609090909');

//Respuesta de la ejecución
echo '<pre>';
print_r($client->getResponse());
echo '</pre>';
?>
```

2.5.9 Ejemplo Java Obtener número de Contactos que tiene un Grupo

```
import java.net.URL;
import java.util.Vector;
import org.apache.xmlrpc.client.XmlRpcClient;
import org.apache.xmlrpc.client.XmlRpcClientConfigImpl;

public class ObtenerNumeroContactos {

	public static void main(String[] args) {
	String url_conexion= "https://www.mensajerianegocios.movistar.es/SrvConexion";
		String login= "login"; //Nombre de usuario
		String password= "contrasena"; //Contraseña
		String colectivo="Grupo"; //Nombre de Grupo
		try {
			//Conectar con el Servidor
			XmlRpcClientConfigImpl config = new XmlRpcClientConfigImpl();
			 config.setServerURL(new URL(url_conexion));
			 config.setEncoding("ISO-8859-15");
			//Construir llamada
			Vector params = new Vector();
			//Primer parámetro: Login
			params.addElement(login);
			//Segundo parámetro: Contraseña
			params.addElement(password);
			//Tercer parámetro: Grupo
			params.addElement(colectivo);

			//Realizar la petición al servidor
			XmlRpcClient client = new XmlRpcClient();
			 client.setConfig(config);
			 Object result =
client.execute("MensajeriaNegocios.obtenerNumeroContactosdeGrupoContactos", params);

			System.out.println("Resultado:"+(Integer)result);

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

	}
}
```

2.5.10 Ejemplo PHP Obtener número de Contactos que tiene un Grupo

```
<?php
include ('IXR_Library.inc.php');  //Libreria para el uso de xml-rpc

$url_conexion='https://www.mensajerianegocios.movistar.es/SrvConexion';
$login='Login'; //Nombre de usuario
$password='Contrasena'; //Contraseña

$client = new IXR_Client($url_conexion);

//Ejecuta el metodo rpc
$client->query ('MensajeriaNegocios.obtenerNumeroContactosdeGrupoContactos', $login,
$password, 'Grupo');

echo '<pre>';
print_r($client->getResponse());
echo '</pre>'; ?>
```

2.5.11 Ejemplo Java Obtener números de teléfono de Contactos de un Grupo

```
import java.net.URL;
import java.util.Vector;
import org.apache.xmlrpc.client.XmlRpcClient;
import org.apache.xmlrpc.client.XmlRpcClientConfigImpl;

public class ObtenerContactosGrupo {

	public static void main(String[] args) {
	String url_conexion= "https://www.mensajerianegocios.movistar.es/SrvConexion";
		String login= "login"; //Nombre de usuario
		String password= "contrasena"; //Contraseña
		String colectivo="Grupo"; //Nombre de Grupo

		try {
			//Conectar con el Servidor
			XmlRpcClientConfigImpl config = new XmlRpcClientConfigImpl();
			config.setServerURL(new URL(url_conexion));
			 config.setEncoding("ISO-8859-15");
			//Construir llamada
			Vector params = new Vector();
			//Primer parámetro: Login
			params.addElement(login);
			//Segundo parámetro: Contraseña
			params.addElement(password);
			//Tercer parámetro: Grupo
			params.addElement(colectivo);

			//Realizar la petición al servidor
			XmlRpcClient client = new XmlRpcClient();
			 client.setConfig(config);
			Object result =
client.execute("MensajeriaNegocios.obtenerContactosdeGrupoContactos", params);
			//Imprimir los resultados
			if(result instanceof Integer) {
				System.out.println(result);
			}else{
				for(int i=0; i<((Object[])result).length;i++){
					System.out.println(((Object[])result)[i]);
				}
			}
		}catch (Exception e) {
			e.printStackTrace();
		}

	}
}
```

2.5.12 Ejemplo Java Obtener mensaje recibidos en el Buzón de Entrada

```
import java.net.MalformedURLException;
import java.util.Vector;

import org.apache.xmlrpc.XmlRpcException;
import org.apache.xmlrpc.client.XmlRpcClient;
import org.apache.xmlrpc.client.XmlRpcClientConfigImpl;
import org.apache.xmlrpc.client.XmlRpcCommonsTransportFactory;

public class obtenerSMSRecibidos {

	// public static String url =
	// "https://www.mensajerianegocios.movistar.es/SrvConexion";
	public static String url = "http://localhost:8400/mnegocios/SrvConexion";
	public static String login = "user2588";
	public static String password = "ab12cd34";
	public static String metodo = "MensajeriaNegocios.obtenerSMSRecibidos";

	public static String fechaInicial = "2016-01-01 00:00:00";
	public static String fechaFinal = "2017-02-01 00:00:00";

	public static void main(String[] args) {

		System.setProperty("javax.net.ssl.trustStore", "mn2.jks");

		XmlRpcClientConfigImpl config = new XmlRpcClientConfigImpl();
		try {
			config.setServerURL(new java.net.URL(url));
		} catch (MalformedURLException e) {
			System.out
					.println("La URI introducida no tiene una sintaxis
correcta...");
		}
		config.setEncoding("ISO-8859-1");
		XmlRpcClient client = new XmlRpcClient();
		client.setTransportFactory(new XmlRpcCommonsTransportFactory(client));
		client.setConfig(config);

		Vector params = new Vector();

		params.addElement(login);
		params.addElement(password);
		params.addElement(fechaInicial);
		params.addElement(fechaFinal);

		try {
			Object resultado = client.execute(metodo, params);

			if (resultado.getClass().getName()
					.compareToIgnoreCase("java.lang.Integer") == 0) {
				System.out.println("Error: " + resultado);
			} else {
				Object[] result = (Object[]) resultado;

				for (int i = 0; i < result.length; i++) {
					System.out.println("Numero: " + result[i]);
				}
			}
```

```
                } catch (XmlRpcException e) {
                        System.out.println("Se produjo un excepcion: "
                                        + e.getLocalizedMessage());
                }

        }

}
```

2.5.13 Ejemplo Java Obtener mensaje enviados y su estado

```
import java.net.MalformedURLException;
import java.util.Vector;

import org.apache.xmlrpc.XmlRpcException;
import org.apache.xmlrpc.client.XmlRpcClient;
import org.apache.xmlrpc.client.XmlRpcClientConfigImpl;
import org.apache.xmlrpc.client.XmlRpcCommonsTransportFactory;

public class obtenerSMSEnviados {

	// public static String url =
	// "https://www.mensajerianegocios.movistar.es/SrvConexion";
	public static String url = "http://localhost:8400/mnegocios/SrvConexion";
	public static String login = "user2588";
	public static String password = "ab12cd34";
	public static String metodo = "MensajeriaNegocios.obtenerSMSEnviados";

	public static String fechaInicial = "2017-01-23 00:00:00";
	public static String fechaFinal = "2017-01-24 00:00:00";

	public static void main(String[] args) {

		System.setProperty("javax.net.ssl.trustStore", "mn2.jks");

		XmlRpcClientConfigImpl config = new XmlRpcClientConfigImpl();
		try {
			config.setServerURL(new java.net.URL(url));
		} catch (MalformedURLException e) {
			System.out
					.println("La URI introducida no tiene una sintaxis
correcta...");
		}
		config.setEncoding("ISO-8859-1");
		XmlRpcClient client = new XmlRpcClient();
		client.setTransportFactory(new XmlRpcCommonsTransportFactory(client));
		client.setConfig(config);

		Vector params = new Vector();

		params.addElement(login);
		params.addElement(password);
		params.addElement(fechaInicial);
		params.addElement(fechaFinal);

		try {
			Object resultado = client.execute(metodo, params);

			if (resultado.getClass().getName()
					.compareToIgnoreCase("java.lang.Integer") == 0) {
				System.out.println("Error: " + resultado);
			} else {
				Object[] result = (Object[]) resultado;

				if (result.length > 0) {
					for (int i = 0; i < result.length; i++) {
						Object[] sms = (Object[]) result[i];
						System.out.println("Login: " + sms[0]);
						System.out.println("Movil: " + sms[1]);
```

```
                                                System.out.println("Texto: " + sms[2]);
                                                System.out.println("Remitente: " + sms[3]);
                                                System.out.println("Fecha: " + sms[4]);
                                                System.out.println("Estado: " + sms[5]);
                                                System.out.println("Estado smpp: " +
sms[6]);
                                        }
                                } else {
                                        System.out.println("No hay mensajes.");
                                }
                        }

                } catch (XmlRpcException e) {
                        System.out.println("Se produjo un excepcion: "
                                        + e.getLocalizedMessage());
                }

        }

}
```

2.5.14 Ejemplo PHP Obtener números de teléfono de Contactos de un Grupo

```
<?php
include ('IXR_Library.inc.php');  //Libreria para el uso de xml-rpc

$url_conexion='https://www.mensajerianegocios.movistar.es/SrvConexion';
$login='Login'; //Nombre de usuario
$password='Contrasena'; //Contraseña

$client = new IXR_Client($url_conexion);

//Ejecuta el metodo rpc
$client->query ('MensajeriaNegocios.obtenerContactosdeGrupoContactos', $login, $password,
'Grupo');

echo '<pre>';
print_r($client->getResponse());
echo '</pre>';
?>
```

3 API SIMPLIFICADA

A continuación, se describen los métodos remotos disponibles para conectarse al servidor de envío de mensajes de Mensajería Negocios.

- **URL de conexión** (URL a la que realizar las peticiones): Debe conectarse a la siguiente dirección utilizando conexión segura (acceso HTTPS):

  https://www.mensajerianegocios.movistar.es/ISServer/services/MI-InterfazSimplificado-Port

3.1 Envío de mensajes móvil terminado

De las operaciones disponibles para el envío de mensajes desde una aplicación a un usuario móvil sólo está soportada la siguiente:

- smsTextSubmitReq: para el envío de mensajes cortos de texto.

  Si los clientes desean obtener información sobre el resultado de la operación, podrán solicitar notificación de entrega. En caso de que se utilizase el envío a múltiples destinatarios, la aplicación del cliente recibirá tantas notificaciones como destinatarios, aunque todas ellas tendrán el mismo identificador de mensaje asociado. Con el fin de que el cliente pueda determinar a qué usuario corresponde cada una, se incluye también el remitente y el destinatario dentro de cada notificación.

La plataforma de mensajería, ante la recepción de cada una de estas operaciones generará la correspondiente respuesta:

- smsTextSubmitRes

3.2 Estructura general de los mensajes

El Interfaz Simplificado es un Web Service que utiliza el protocolo SOAP [SOAP] y HTTP [HTTP] para el intercambio de mensajes con el cliente. A continuación se describen los métodos para garantizar la seguridad de las peticiones así como el detalle de las mismas.

3.2.1 Autenticación de las peticiones

Para la correcta autenticación de la aplicación, todas las peticiones que se realicen tendrán que incluir un usuario y contraseña configurado en la aplicación de Mensajería de Negocios. Este usuario y contraseña se incluirá dentro de la petición SOAP [SOAP] en el campo “Authorization”.

El campo “Authorization” utiliza la misma codificación que la autenticación básica utilizada en HTTP (RFC 2617):

Authorization = base64-user-cont

base64-user cont = base64encode(user-cont)

user-cont=usuario ":" contraseña

usuario = Configurado en MN.

contraseña = Configurado en MN.

Como ejemplo de codificación y partiendo de los siguientes datos configurados en MenTeS:

Usuario: user

Contraseña: secret

El valor del campo “Authorization” sería el siguiente:

Base64encode(“user:secret”) -> dXNlcjpzZWNyZXQ=

<Authorization>dXNlcjpzZWNyZXQ=</Authorization>

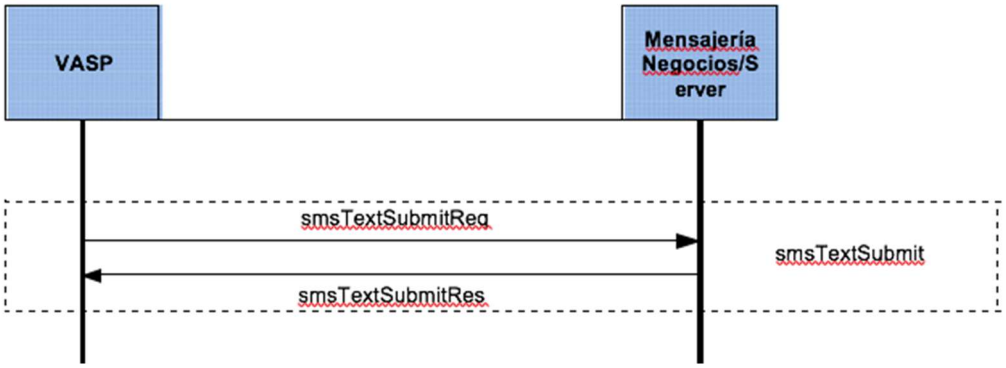
3.2.2 Seguridad para el acceso a través de Internet.

Cuando el cliente haya configurado “Internet” como tipo de acceso al servicio, las comunicaciones entre la aplicación del cliente y Mensajería de Negocios deben estar protegidas mediante un túnel SSL utilizando HTTPS [HTTPS] en lugar de HTTP [HTTP].

Para poder establecer el túnel en las peticiones provenientes desde la aplicación del cliente es necesario que el cliente incorpore el certificado de Mensajería de Negocios dentro su almacén de certificados de confianza. Este certificado será presentado por el servidor MN al conectarse a la URL del servicio.

3.3 Detalles del mensaje

En este apartado se definen los parámetros del interfaz simplificado disponible en la plataforma de mensajería para el envío de mensajes móvil terminado.



3.3.1 smsTextSubmitReq

Nombre del parámetro	Presencia	Descripción
Version	Obligatorio	El parámetro de versión está relacionado con las futuras versiones del Interfaz Simplificado. En la actualidad el valor permitido es "1.0". Ej: *<Version>1.0</Version>*
Authorization	Obligatorio	El parámetro de autenticación del interfaz simplificado debe contener la cadena de texto "usuario:contraseña" codificada en base64. El usuario y contraseña será facilitado por Telefónica Móviles. Ej: *<Authorization>**dGVzdDp0ZXN0**</Authorization>* *encodeBase64("test:test") -> dGVzdDp0ZXN0*
Sender	Obligatorio	Remitente del mensaje. Los posibles valores son: • Números cortos (7777) • Alfanuméricos (movistar) Sólo se podrán utilizar los remitentes contratados por el cliente. Ej: *<Sender>7777</Sender>* *<Sender>movistar</Sender>*
Recipients	Obligatorio	Destinatario del mensaje. Formato MSISDN: El número del destinatario puede tener uno de los patrones de numeración siguientes: • MSISDN en formato nacional (6xxxxxxxx) • MSISDN en formato internacional (00346xxxxxxxx y +346xxxxxxxx)
SMSText	Obligatorio	Texto del mensaje. Ej: *<SMSText>Texto del SMS</SMSText>*

DeliveryReport	Opcional	Este parámetro indica si se requiere notificación de entrega. Los posibles valores son: • "**All**": Se solicita todos los tipos de notificación de entrega (mensajes entregados, expirados y rechazados) • "**None**": No se solicita notificación de entrega. • "**Success**": Se solicitan las notificaciones sólo para aquellos mensajes que haya podido ser entregados correctamente. Si se selecciona esta opción no se recibirán notificaciones de mensajes rechazados o de mensajes expirados. • "**Failure**": Se solicitan las notificaciones sólo para aquellos mensajes que no hayan podido ser entregados, bien porque hayan sido rechazados o porque hayan expirado. Si se selecciona esta opción no se recibirán notificaciones de mensajes entregados. **Nota**: Este campo es sensible a mayúsculas y minúsculas por lo que los valores a utilizar han de ser los arriba indicados. Si no se incluye este campo el valor poder defecto es "None" Ej: *<DeliveryReport >All</DeliveryReport>*
DeliveryReportURL	Opcional	En este parámetro se indica la URL donde se debe entregar la notificación de entrega del mensaje. Si no se indica esta URL se utilizará la especificada por el cliente en el Formulario de datos del servicio que rellenó cuando contrató el servicio Mensajería Integrada. Ej: *<DeliveryReportURL>https://ip:puerto/listener</DeliveryReportURL>*

3.3.2 smsTextSubmitRes

3.4 Códigos de respuesta

Los distintos códigos de respuesta están estructurados de la siguiente manera:

- 1xxx: Petición procesada correctamente.
- 2xxx: Error en la petición del cliente.
- 3xxx: Error en el servidor de MN.
- 4xxx: Error en el servicio de MN.

A continuación, se muestra la tabla detallada con todos códigos de respuesta. Además, por cada estado se incluye el valor asociado en los distintos campos con información adicional del estado, los pasos que debe seguir el cliente y las primitivas que pueden generarlo:

Respuesta (StatusCode)	Status-Text	Details	Descripción StatusCode	Pasos a seguir	Operación
1000	Success	Vacío	El mensaje se ha procesado correctamente.	Si se ha solicitado notificación de entrega y es una operación de envío de mensaje almacenar el MessageId incluido en la respuesta para así poder procesar la posterior notificación de entrega.	smsTextSubmitRes
2000	Client error	"Error parsing XML"	El cliente envió una petición con un documento XML incorrecto.	Revisar que la petición SOAP enviada es un XML bien formado. En caso de serlo revisar el valor de la cabecera HTTP Content-length y ver que coincide con el tamaño del documento XML. Si todo esto es correcto póngase en contacto con Telefónica para reportar la incidencia.	smsTextSubmitRes
2002	Sender Address Error	El campo Detail incluirá el texto "Sender %sen der %sender% is rejected", Donde %sen- der% será el remitente del mensaje re- chazado.	El remitente proporcionado no es correcto ya que no se encuentra entre los con- tratados en el servicio	Revise el remitente del mensaje. En caso de ser correcto póngase en contacto con Telefónica para reportar la inciden- cia.	smsTextSubmitRes
2003	Recipient Address Error	El campo De- tails incluirá el texto "Reci- pient %reci- pient% is rejected", donde %reci- pient% será el destinatario del mensaje re- chazado.	El destinatario del mensaje o el formato del mismo son incorrectos.	Revise el destinatario así como el formato del mismo y compruebe que coincide con los sopor- tados según esta guía. En caso de coincidir póngase en contacto con Telefónica para reportar la incidencia.	smsTextSubmitRes
2004	Content refused	El campo De- tails incluirá el texto "Content not suppor- ted".	Alguno de los contenidos del mensaje es incorrecto o no está permitido	Revise los contenidos incluidos. En caso de ser correctos póngase en contacto con Telefónica para reportar la incidencia.	smsTextSubmitRes
3000	Server Error	El campo De- tails incluirá el texto "Generic Server error. Please, contact system admi- nistrators"	El Re- lay/Server no está disponible o existe alguna incidencia en alguno de los elementos de la red de Telefónica que impiden cursar la petición del cliente.	Póngase en contacto con Telefónica para reportar la incidencia.	smsTextSubmitRes

4000	General service error	El campo De- tails incluirá el texto "Generic Service error"	El servicio esta caído y no se pudo cursar la petición del cliente.	Póngase en contacto con Telefónica para reportar la incidencia.	smsTextSubmitRes
4001	Improper identifica- tion	El campo De- tails incluirá el texto "Author- ization failed. Check Author- ization field or contact system administra- tors".	Fallo en la autenticación del servicio.	Revise el valor del campo "Authorization" y com- pruebe que coincide con el formato descrito en el punto **3.1.1 Autenticación de peticiones.** En caso de ser correcto póngase en contacto con Telefónica para reportar la incidencia.	smsTextSubmitRes
4002	Unsup- ported version	El campo De- tails incluirá el texto "The version sup- ported for this service Is %version%", donde %ver- sion%" será la versión confi- gurada del servicio.	La versión del protocolo empleada por el cliente es incorrecta.	Revise el valor del campo Version y compruebe que coincide con la versión del protocolo solicitada. En caso de ser correcto póngase en contacto con Telefónica para reportar la incidencia.	smsTextSubmitRes
4004	Validation error	El campo De- tails incluirá información adicional sobre el campo que provoco el fallo en la valida- ción.	Fallo en la validación de la petición contra el es- quema. Falta algún campo obligatorio o el valor de al guno de ellos es incorrecto.	Revise la petición si- guiendo las indicaciones del campo Details. En caso de ser todo correcto póngase en contacto con Telefónica para reportar la incidencia.	smsTextSubmitRes
4007	Service denied	El campo De- tails incluirá el texto "The service is disa- bled. Contact system admi- nistrator"	El servicio del cliente ha sido deshabilitado temporalmen- te.	Póngase en contacto con Telefónica para reportar la incidencia.	smsTextSubmitRes

3.5 Caracteres soportados en SMS

Los caracteres permitidos en los contenidos de los mensajes son:

01234567890
ABCDEFGHIJKLMNÑOPQRSTUVWXYZ
abcdefghijklmnñopqrstuvwxyz
ÇäöüÄÖÜÆÅå@£$¥Øø¡!"#
%&'()*+,-./:;<=>§¿?
y Espacio

Los caracteres no permitidos se convierten automáticamente a otros válidos siguiendo la siguiente tabla de conversiones:

Carácter	Conversión
Á	A
á	a
É	E
é	e
Í	I
í	i
Ó	O
ó	o
Ú	U
ú	u
à	a
è	e
ì	i
ò	o
ù	u
_	espacio
ç	c
^	‘
{	(
}	)
|	/
[	(
]	)
€	E

IMPORTANTE: Se debe utilizar la codificación LATIN-15 o ISO 8859-15, de no ser así, es muy posible que no se pueda enviar el mensaje.

3.6 Ejemplos de implementación

Para entender mejor estos ejemplos asumimos que el cliente dispone de los siguientes datos facilitados por Telefónica:

DATO	VALOR EN LOS EJEMPLOS
URL de envío	https://www.mensajerianegocios.movistar.es/ISServer/services/MI-InterfazSimplificado-Port
Usuario	usuario
Contraseña	contraseña

3.6.1 WSDL – Operaciones envío mensajes

Se adjunta url del WSDL para su uso en las herramientas de desarrollo. Señalar que sólo está soportado el método que se describe en este documento.

https://www.mensajerianegocios.movistar.es/MiInterfazSImplificado.wsdl

3.6.2 Ejemplo Shell script (using curl) Envío Mensaje MT

```
PHONE=$1
if [ -z "$PHONE"  ]
  then
    echo "Uso: $0 movil"
    exit
  fi
if [ -f dump.xml ]
  then
    echo "Deleting existing output file"
    rm dump.xml
fi
curl -k -v -o dump.xml -H "SOAPAction: smsTextSubmit" -H "Content-Type: application/soap+xml;
charset=UTF-8" -d "<?xml version='1.0' encoding='UTF-8'?>
<soapenv:Envelope xmlns:soapenv='http://schemas.xmlsoap.org/soap/envelope/'
xmlns:xsd='http://www.w3.org/2001/XMLSchema' xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'>
<soapenv:Body>
<smsTextSubmitReq xmlns='http://www.telefonica.es/MI/InterfazSimplificado/schemas'>
<Version xmlns=''>1.0</Version>
<Authorization xmlns=''>XXXXXXXX</Authorization>
<Sender xmlns=''>ALERTAS</Sender>
<Recipients xmlns=''>
<To>$1</To>
</Recipients>
<SMSText xmlns=''>Test IS</SMSText>
<DeliveryReport xmlns=''>All</DeliveryReport>
<DeliveryReportURL
xmlns=''>https://www.mensajerianegocios.movistar.es/ISServer/emulator</DeliveryReportURL>
</smsTextSubmitReq>
</soapenv:Body></soapenv:Envelope>"
https://www.mensajerianegocios.movistar.es/ISServer/services/MI-InterfazSimplificado-Port
```

3.6.3 Ejemplo Java Envío Mensaje MT

```
package es.telefonica.www.MI.InterfazSimplificado.client;

import java.rmi.RemoteException;
import javax.xml.rpc.ServiceException;


import   es.telefonica.www.MI.InterfazSimplificado.definitions.MIInterfazSimplificadoBindingStub;
import        es.telefonica.www.MI.InterfazSimplificado.definitions.MIInterfazSimplificadoLocator;
import es.telefonica.www.MI.InterfazSimplificado.schemas.SmsTextSubmitReq;

import es.telefonica.www.MI.InterfazSimplificado.schemas.SubmitRes;

import es.telefonica.www.MI.InterfazSimplificado.schemas.VersionType;


public class ISSMSClient {


    static MIInterfazSimplificadoLocator localizador = new MIInterfazSimplificadoLocator();
    static   MIInterfazSimplificadoBindingStub   cliente   =   null;
    static SmsTextSubmitReq sms = new SmsTextSubmitReq();


    static  String  url  =  "https://www.mensajerianegocios.movistar.es/ISServer/services/MI-InterfazSimplificado-
    Port";

    static String usuario = "usuario";
    static String contraseña = "contraseña";
    static String remitente = "217812";
    static String[] destinatarios = new String[]{"600000000"};
    static String texto_sms = "Texto de prueba";


    public static void main(String[] args) throws ServiceException, RemoteException {

        // Añadimos el almacén de claves que contiene el certificado de la CA de Telefónica
        // como almacén de claves de confianza del sistema para poder enviar mensajes
```

```
        // empleando HTTPS.
        System.setProperty("javax.net.ssl.trustStore", "trustedTelefonicaCACerts.jks");


        // Indicamos la versión del protocolo a utilizar
        sms.setVersion(VersionType.fromString("1.0"));

        // Indicamos los datos de autenticación
        sms.setAuthorization((usuario + ":" + contraseña).getBytes());

        // Indicamos el remitente
        sms.setSender(remitente);

        // Indicamos la lista de destinatarios
        sms.setRecipients(destinatarios);

        // Indicamos el texto del SMS
        sms.setSMSText(texto_sms);

        //Indicamos la URL de envío
        localizador.setMIInterfazSimplificadoPortEndpointAddress(url);

        // Generamos un cliente a partir del localizador
        cliente = (MIInterfazSimplificadoBindingStub) localizador.getMIInterfazSimplificadoPort();


        // Enviamos la petición y almacenamos la respuesta
        SubmitRes respuesta = cliente.smsTextSubmit(sms);


        // Mostramos por pantalla el resultado del envío
        System.out.println("Mensaje enviado...");
        System.out.println("StatusCode:      "      +      respuesta.getStatus().getStatusCode());
        System.out.println("StatusText:      "      +      respuesta.getStatus().getStatusText());
        System.out.println("StatusDetails:      "      +      respuesta.getStatus().getDetails());
        System.out.println("MessageID: " + respuesta.getMessageId());


    }


}
}
```

3.6.4 Ejemplo C# - Envío Mensaje MT

```
using System;

using System.Collections.Generic;
using System.Text;

namespace ISClient

{

   class ISSMSTextClient

   {

      private static SubmitRes respuesta;

      private static MIInterfazSimplificado cliente = new MIInterfazSimplificado(); private
      static smsTextSubmitReq sms = new smsTextSubmitReq();

      private static System.Text.ASCIIEncoding encoding = new System.Text.ASCIIEncoding();
      private static string url = "https://www.mensajerianegocios";

      private static string usuario = "usuario"; private
      static string contraseña = "contraseña"; private
      static string remitente = "217812";

      private static string[] destinatarios = new string[] { "600000000" }; private
      static string texto_sms = "Texto de prueba";

      static void Main(string[] args)

      {

         encoding = new System.Text.ASCIIEncoding();
         System.Net.ServicePointManager.Expect100Continue = false;

         // Indicamos la version del protocolo a utilizar
         sms.Version = versionType.Item10;

         // Indicamos los datos de autenticacion

         sms.Authorization = encoding.GetBytes(usuario + ":" + contraseña);

         // Indicamos el remitente
         sms.Sender = remitente;

         // Indicamos la lista de destinatarios
         sms.Recipients = destinatarios;

         // Indicamos el texto del SMS
         sms.SMSText = texto_sms;


         // Indicamos la URL de envío
         cliente.Url = url;

         // Enviamos la peticion y almacenamos la respuesta
         respuesta = cliente.smsTextSubmit(sms);

         // Mostramos por pantalla el resultado del envio Console.Out.WriteLine("Mensaje
         enviado..."); Console.Out.WriteLine("StatusCode: " +
         respuesta.Status.StatusCode); Console.Out.WriteLine("StatusText: " +
         respuesta.Status.StatusText); Console.Out.WriteLine("StatusDetails: " +
         respuesta.Status.Details); Console.Out.WriteLine("MessageId: " +
         respuesta.MessageId);


      }
   }
}
```