Especificaciones de la Interfaz REST para envío de SMS

Altiria TIC, S.L.

Versión: 1.0

Historial de cambios

Versión	Cambios
1.0	Primera versión del documento.

Índice general

Capítulo 1

Introducción

En este documento se presenta la API disponible para el envío de mensajes cortos sobre la interfaz de *Altiria* a través de recursos REST.

El servicio de envío de mensajes cortos está disponible en muchos paises. Para conocer los paises permitidos, las operadoras válidas en cada país y las posibles restricciones geográficas (salvedades al funcionamiento general detallado en este documento que pudieran aplicar en cada caso) se puede enviar un correo electrónico a comercial@altiria.com.

El servicio opcional de confirmación de entrega requiere que el cliente exponga un recurso REST para recibir la información de confirmación (ver la sección 2.6).

Capítulo 2

Descripción de la API

2.1. Acceso a los recursos REST

La **URL base** de acceso a los recursos REST es **http://www.altiria.net/apirest/ws**

Cada petición REST enviada se corresponde con un recurso expuesto de la API, según se detalla en los siguientes apartados.

El cuerpo de cada petición REST está compuesto por un mensaje en formato JSON con el *"content-type:application/json;charset=UTF-8"*.

Se debe usar la **codificación UTF-8** en la comunicación con el servidor.

En el apartado 2.8 se dan varios ejemplos.

2.2. Respuesta a la petición REST

Cada petición de recurso REST lleva asociada una respuesta desde el servidor de Altiria en formato **JSON codificada en UTF-8**.

En los siguientes apartados se detalla la respuesta para cada petición siempre que resulte exitosa. En esos casos el código de estado HTTP será 200.

Si se produjese algún error en el servidor el código de estado HTTP denotará un fallo particular (será distinto de 200). Puede ocurrir si se ha producido un error de "binding", se intenta acceder a un recurso que no existe o se viola alguna restricción de alguno de los elementos del JSON de la petición. En esos casos el JSON de respuesta tendrá un único elemento "error" para informar sobre el problema.

Un ejemplo de respuesta debido a un error en la petición del cliente (falta el parámetro obligatorio "login") es el siguiente:

```
{"error":"LOGIN_NOT_NULL"}
```

2.3. Recursos de la API

A continuación se detallan los recursos disponibles en la API REST y los elementos que componen el mensaje JSON de la petición. Cada elemento puede ser obligatorio u opcional.

Cabe citar que el nombre de todos los elementos del JSON podrá ser remitido en diferentes formatos:

- Según la notación Java ("domainId").

- Según la notación REST ("domain_id").
- Todo en minúsculas ("domainid").

Las **peticiones** a cada recurso se harán sobre la **URL base** suministrada, **concatenando el "path"** propio de cada recurso.

2.3.1. Envío de un mensaje de texto

Permite enviar un mensaje corto de texto a uno o a varios teléfonos destinatarios.

Se trata del recurso con **"path" /sendSms**. El JSON está compuesto por los **elementos obligatorios** detallados en el cuadro 2.1.

Nombre	Valores	Composición
credentials	Datos de identificación del usuario suministrados por *Altiria*	Elementos domainId, login y passwd
destination	Lista de números de teléfono móvil de los destinatarios del mensaje	Cada número se especificará en formato de numeración internacional sin prefijo '00' ni el signo '+'. Ej: 34645852126. Es fundamental incluir el prefijo del país (34 para España) para que el mensaje llegue al destino esperado. No debe superar los 16 dígitos
message	Datos propios del mensaje a enviar	Elementos msg, senderId, ack, idAck, dPort, sPort, encoding y concat

Cuadro 2.1: Parámetros de entrada del recurso sendSms

Para enviar el **mensaje a varios destinatarios** basta con añadir los diferentes números de teléfono al elemento "destination" sin sobrepasar el límite máximo permitido (consultar al soporte técnico de *Altiria* en soporte@altiria.com), asignándole cada vez el valor de un número de teléfono distinto (los teléfonos repetidos son descartados). Se recomienda en cualquier caso no exceder de 100 destinatarios por petición.

En el cuadro 2.2 se detallan los elementos restantes constituyentes del JSON de la petición.

Nombre	Valor	Obligatorio
domainId	Identificador suministrado por *Altiria* al cliente. Se puede omitir si el login es un email.	no
login	Identificador de usuario suministrado por *Altiria* al cliente.	sí
passwd	Clave del usuario suministrada por *Altiria* al cliente.	sí
msg	Mensaje a enviar. La lista de caracteres válidos y la longitud máxima permitida se detalla en la sección 2.4. No puede estar vacío (cadena vacía).	sí
senderId	Remitente del mensaje a enviar, autorizado por *Altiria* . La posibilidad de personalizar el remitente depende del país destinatario del mensaje. Puede tomar dos posibles valores: 1) valor alfanumérico de hasta 11 caracteres (números y letras de la "a" a la "z" tanto mayúsculas como minúsculas excluyendo la "Ñ" y la "ñ"); 2) valor numérico de hasta 15 dígitos decimales comenzando por el carácter "+". **Los caracteres inválidos serán suprimidos automáticamente**. Si se pretende que el receptor pueda responder al mensaje corto recibido se debería usar un remitente numérico (opción 2) incluyendo el prefijo de país. Si no se incluye, el mensaje se enviará con el remitente por defecto seleccionado por Altiria.	no

ack	Solicitud de confirmación de entrega de los mensajes enviados (ver sección 2.6). Si vale "true" solicita confirmación de entrega de los SMS enviados. En su ausencia o si tiene otro valor no se solicita confirmación de entrega.	no
idAck	Código identificativo para la confirmación de entrega (ver sección 2.6). Valor alfanumérico de hasta 20 caracteres (números y letras de la "a" a la "z" tanto mayúsculas como minúsculas sin incluir ni "Ñ" ni "ñ"). De rebasar la longitud máxima permitida será truncado. Los caracteres no permitidos serán eliminados. Solo será considerado si el parámetro ack se envía con valor "true". Si se incluye explícitamente este parámetro y toma como valor cadena vacía, anula la solicitud de confirmación de entrega.	no
dPort	Puerto destino del SMS a enviar. Valor numérico entre 1 y 65535 (solo dígitos decimales). Solo necesario si se desea cambiar el valor habitual de esta propiedad. Si se define, la longitud máxima del mensaje a enviar (parámetro "msg") se verá reducida (ver la sección 2.4.3) y se invalidará la posibilidad de enviar mensajes concatenados (ver parámetro "concat"). Si solo se define "sPort", este tomará el valor 0.	no
sPort	Puerto origen del SMS a enviar. Valor numérico entre 1 y 65535 (solo dígitos decimales). Solo necesario si se desea cambiar el valor habitual de esta propiedad. Si se define, la longitud máxima del mensaje a enviar (parámetro "msg") se verá reducida (ver la sección 2.4.3) y se invalidará la posibilidad de enviar mensajes concatenados (ver parámetro "concat"). Si solo se define "dPort", este tomará el valor 0.	no
encoding	El único valor permitido es "unicode" para cambiar la codificación del SMS a Unicode (ver la sección 2.4.2). En su ausencia o si tiene otro valor el SMS tomará la codificación por defecto.	no
concat	Si vale "true" permite concatenar mensajes para enviar un mensaje corto de longitud mayor que la habitual (ver la sección 2.4.3). En su ausencia, si tiene otro valor o si se define el parámetro "sPort" o "dPort" se deshabilita la concatenación de mensajes.	no

Cuadro 2.2: Lista de parámetros para sendsms

Un ejemplo de petición REST al recurso sendSms solicitando el envío de un mensaje concatenado a dos destinatarios sería este:

```
{
"credentials":{"domainId":"XXXXX",
               "login":"YYYYY",
               "passwd":"ZZZZZ"},
"destination":["346XXXXXXXX","346YYYYYYYY"],
"message":{"msg":"Ejemplo de mensaje concatenado enviado a más de un destinatario con
                 la codificación UNICODE para admitir las vocales acentuadas y
                 solicitud de confirmación de entrega.",
   "senderId":"remitente",
   "ack":"true",
   "idAck":"123456789",
   "concat":"true",
   "encoding":"unicode"}
}
```

La respuesta a la petición del recurso sendSms está compuesta por el elemento "status" con valor un código de estado general de los descritos en el apartado 2.7

Si la operación ha resultado exitosa (código de estado "000") la respuesta contendrá adicionalmente un elemento "details" incluyendo para cada destinatario del envío (cada elemento de la lista "destination" de entrada) los siguientes datos:

- destination: se corresponde con el número de teléfono del destinatario. Si se hubiese enviado un mensaje concatenado (ver el parámetro "concat") a un único destinatario le corresponderán varios mensajes, tantos como fragmentos compongan el mensaje concatenado. En ese caso aparecerán datos independientes para cada fragmento siendo cualificado el valor de "destination" con un sufijo que diferencie cada fragmento con un índice numérico comenzando por 0. Por ejemplo para un mensaje concatenado de tres fragmentos enviado al número "xxxxxxxxxxx" se recibirán datos para destination=xxxxxxxxxxx(0), destination=xxxxxxxxxxx(1) y destination=xxxxxxxxxxx(2).
- status: se corresponde con uno de los códigos de estado del apartado 2.7.
- idAck: se corresponde con el código de identificación asociado a la solicitud de confirmación de entrega (ver sección 2.6). Solo aparecerá si se solicita la confirmación de entrega y es aceptada.

Un ejemplo de respuesta exitosa correspondiente a la petición al recurso **sendSms** del ejemplo anterior sería el mostrado a continuación. Al tratarse de un mensaje concatenado aparecen datos para cada uno de los tres fragmentos que lo componen para cada destinatario del envío:

```
{
"details":
[{"destination":"346XXXXXXXX(0)","idAck":"123456789","status":"000"},
 {"destination":"346XXXXXXXX(1)","idAck":"123456789","status":"000"},
 {"destination":"346XXXXXXXX(2)","idAck":"123456789","status":"000"},
 {"destination":"346YYYYYYYY(0)","idAck":"123456789","status":"000"},
 {"destination":"346YYYYYYYY(1)","idAck":"123456789","status":"000"},
 {"destination":"346YYYYYYYY(2)","idAck":"123456789","status":"000"}]
,"status":"000"
}
```

Un ejemplo de respuesta notificando un error en la autentificación sería este:

```
{"status":"020"}
```

La infomación de éxito para un destinatario concreto implica que el mensaje ha sido aceptado por la pasarela, no que haya sido enviado y recibido por el destinatario. Un mensaje puede ser aceptado aún cuando no se disponga de crédito suficiente para su envío (ver sección 2.3.4).

Para asegurar el adecuado funcionamiento de este recurso se recomienda probar la correcta recepción de todos los caracteres permitidos en un teléfono móvil antes de poner el sistema en producción.

2.3.2. Envío de múltiples mensajes de texto

Permite enviar una lista de mensajes cortos de texto, cada cual al destinatario indicado.

Se trata del recurso con **"path" /sendSmsMulti**. El JSON está compuesto por los **elementos obligatorios** detallados en el cuadro 2.3.

Nombre	Valores	Composición
credentials	Datos de identificación del usuario suministrados por *Altiria*	Elementos domainId, login y passwd
messages	Lista de mensajes. Cada elemento de la lista define los datos propios de un mensaje a enviar	Elementos msg, senderId, ack, idAck, dPort, sPort, encoding y concat

Cuadro 2.3: Parámetros de entrada del recurso sendSmsMulti

Para enviar más de un mensaje basta con añadir los diferentes mensajes al elemento "messages", tantos como sea preciso sin sobrepasar el límite máximo permitido (consultar al soporte técnico de *Altiria* en soporte@altiria.com).

En el cuadro 2.4 se detallan los elementos restantes constituyentes del JSON de la petición.

Nombre	Valor	Obligatorio
domainId	Identificador suministrado por *Altiria* al cliente. Se puede omitir si el login es un email.	no
login	Identificador de usuario suministrado por *Altiria* al cliente.	sí
passwd	Clave del usuario suministrada por *Altiria* al cliente.	sí
destination	Número de teléfono móvil del destinatario del mensaje. Se especificará en formato de numeración internacional sin prefijo '00' ni el signo '+'. Ej: 34645852126. Es fundamental incluir el prefijo del país (34 para España) para que el mensaje llegue al destino esperado. No debe superar los 16 dígitos.	sí
msg	Mensaje a enviar. La lista de caracteres válidos y la longitud máxima permitida se detalla en la sección 2.4. No puede estar vacío (cadena vacía).	sí
senderId	Remitente del mensaje a enviar, autorizado por *Altiria* . La posibilidad de personalizar el remitente depende del país destinatario del mensaje. Puede tomar dos posibles valores: 1) valor alfanumérico de hasta 11 caracteres (números y letras de la "a" a la "z" tanto mayúsculas como minúsculas excluyendo la "Ñ" y la "ñ"); 2) valor numérico de hasta 15 dígitos decimales comenzando por el carácter "+". **Los caracteres inválidos serán suprimidos automáticamente**. Si se pretende que el receptor pueda responder al mensaje corto recibido se debería usar un remitente numérico (opción 2) incluyendo el prefijo de país. Si no se incluye, el mensaje se enviará con el remitente por defecto seleccionado por Altiria.	no
idMsg	Referencia del mensaje para facilitar su identificación al procesar la respuesta a la petición de envío múltiple.	no
ack	Solicitud de confirmación de entrega de los mensajes enviados (ver sección 2.6). Si vale "true" solicita confirmación de entrega de los SMS enviados. En su ausencia o si tiene otro valor no se solicita confirmación de entrega.	no

idAck	Código identificativo para la confirmación de entrega (ver sección 2.6). Valor alfanumérico de hasta 20 caracteres (números y letras de la "a" a la "z" tanto mayúsculas como minúsculas sin incluir ni "Ñ" ni "ñ"). De rebasar la longitud máxima permitida será truncado. Los caracteres no permitidos serán eliminados. Solo será considerado si el parámetro ack se envía con valor "true". Si se incluye explícitamente este parámetro y toma como valor cadena vacía, anula la solicitud de confirmación de entrega.	no
dPort	Puerto destino del SMS a enviar. Valor numérico entre 1 y 65535 (solo dígitos decimales). Solo necesario si se desea cambiar el valor habitual de esta propiedad. Si se define, la longitud máxima del mensaje a enviar (parámetro "msg") se verá reducida (ver la sección 2.4.3) y se invalidará la posibilidad de enviar mensajes concatenados (ver parámetro "concat"). Si solo se define "sPort", este tomará el valor 0.	no
sPort	Puerto origen del SMS a enviar. Valor numérico entre 1 y 65535 (solo dígitos decimales). Solo necesario si se desea cambiar el valor habitual de esta propiedad. Si se define, la longitud máxima del mensaje a enviar (parámetro "msg") se verá reducida (ver la sección 2.4.3) y se invalidará la posibilidad de enviar mensajes concatenados (ver parámetro "concat"). Si solo se define "dPort", este tomará el valor 0.	no
encoding	El único valor permitido es "unicode" para cambiar la codificación del SMS a Unicode (ver la sección 2.4.2). En su ausencia o si tiene otro valor el SMS tomará la codificación por defecto.	no
concat	Si vale "true" permite concatenar mensajes para enviar un mensaje corto de longitud mayor que la habitual (ver la sección 2.4.3). En su ausencia, si tiene otro valor o si se define el parámetro "sPort" o "dPort" se deshabilita la concatenación de mensajes.	no

Cuadro 2.4: Lista de parámetros para sendsmsmulti

Un ejemplo de petición REST al recurso sendSmsMulti solicitando el envío de tres mensajes sería este:

```
{"credentials":{"domain_id":"XXXXX",
                "login":"YYYYY",
                "passwd":"ZZZZZ"},
"messages":[{"msg":"Mensaje de prueba 1",
             "destination":"346XXXXXXXX"},
            {"msg":"Lorem Ipsum es simplemente el texto de relleno de las imprentas
                    y archivos de texto. Lorem Ipsum ha sido el texto de relleno
                    estandar de las industrias desde el año 1500",
             "destination":"346YYYYYYYY","concat":true,"id_msg":"id2"},
            {"msg":"Mensaje de prueba 3",
             "destination":"346ZZZZZZZZ",
             "sender_id":"remitente",
             "ack":true,
             "id_ack":"123456789",
             "d_port":5000,
             "s_port":4000,
             "id_msg":"id3"}
            ]
}
```

La respuesta a la petición del recurso sendSmsMulti está compuesta por el elemento "status" con valor un código de estado general de los descritos en el apartado 2.7

Si la operación ha resultado exitosa (código de estado "000") la respuesta contendrá adicionalmente un elemento "details" incluyendo para cada mensaje del envío (cada elemento de la lista "messages" de entrada) los siguientes datos:

- destination: se corresponde con el número de teléfono del destinatario. Si se hubiese enviado un mensaje concatenado (ver el parámetro "concat") al destinatario le corresponderán varios mensajes, tantos como fragmentos compongan el mensaje concatenado. En ese caso aparecerán datos independientes para cada fragmento siendo cualificado el valor de "destination" con un sufijo que diferencie cada fragmento con un índice numérico comenzando por 0. Por ejemplo para un mensaje concatenado de tres fragmentos enviado al número "xxxxxxxxxxx" se recibirán datos para destination=xxxxxxxxxxx(0), destination=xxxxxxxxxxx(1) y destination=xxxxxxxxxxx(2).
- status: se corresponde con uno de los códigos de estado del apartado 2.7.
- idAck: se corresponde con el código de identificación asociado a la solicitud de confirmación de entrega (ver sección 2.6). Solo aparecerá si se solicita la confirmación de entrega y es aceptada.
- idMsg: se corresponde con la referencia asociada al mensaje remitida en la petición de envío. Solo aparecerá si se ha incluido en la petición al servidor.

Un ejemplo de respuesta exitosa correspondiente a la petición al recurso **sendSmsMulti** del ejemplo anterior sería el mostrado a continuación. Al ser el segundo mensaje concatenado aparecen datos para cada uno de los dos fragmentos que lo componen:

```
{"details":[
  {"destination":"346XXXXXXXX","status":"000"},
  {"destination":"346YYYYYYYY(0)","idMsg":"id2","status":"000"},
  {"destination":"346YYYYYYYY(1)","idMsg":"id2","status":"000"},
  {"destination":"346ZZZZZZZZ","idAdk":"123456789","idMsg":"id3","status":"000"}
  ],
"status":"000"}
```

Un ejemplo de respuesta notificando un error en la autentificación sería este:

```
{"status":"020"}
```

La infomación de éxito para un mensaje concreto implica que el mensaje ha sido aceptado por la pasarela, no que haya sido enviado y recibido por el destinatario. Un mensaje puede ser aceptado aún cuando no se disponga de crédito suficiente para su envío (ver sección 2.3.4).

Para asegurar el adecuado funcionamiento de este recurso se recomienda probar la correcta recepción de todos los caracteres permitidos en un teléfono móvil antes de poner el sistema en producción.

2.3.3. Envío de un mensaje multimedia WAP-PUSH

Permite enviar un mensaje multimedia a traves de WAP-PUSH a uno o a varios teléfonos destinatarios.

Para conocer en qué consisten los mensajes de este tipo se aconseja leer la sección 2.5.

Para saber más sobre el proceso de envío de mensajes WAP-PUSH a través de la pasarela se aconseja leer la sección 2.5.7.

Se trata del recurso con **"path" /sendWapPush**. El JSON está compuesto por los **elementos obligatorios** detallados en el cuadro 2.5.

Nombre	Valores	Composición
credentials	Datos de identificación del usuario suministrados por *Altiria*	Elementos domainId, login y passwd
destination	Lista de números de teléfono móvil de los destinatarios del mensaje	Cada número se especificará en formato de numeración internacional sin prefijo '00' ni el signo '+'. Ej: 34645852126. Es fundamental incluir el prefijo del país (34 para España) para que el mensaje llegue al destino esperado. No debe superar los 16 dígitos
message	Datos propios del mensaje a enviar	Elementos msg, url, ack y idAck

Cuadro 2.5: Parámetros de entrada del recurso sendWapPush

Para enviar el **mensaje a varios destinatarios** basta con añadir los diferentes números de teléfono al elemento "destination" sin sobrepasar el límite máximo permitido (consultar al soporte técnico de *Altiria* en soporte@altiria.com), asignándole cada vez el valor de un número de teléfono distinto (los teléfonos repetidos son descartados). Se recomienda en cualquier caso no exceder de 100 destinatarios por petición.

En el cuadro 2.6 se detallan los elementos restantes constituyentes del JSON de la petición.

Nombre	Valor	Obligatorio
domainId	Identificador suministrado por *Altiria* al cliente. Se puede omitir si el login es un email.	no
login	Identificador de usuario suministrado por *Altiria* al cliente.	sí
passwd	Clave del usuario suministrada por *Altiria* al cliente.	sí
msg	Texto a enviar adjunto al mensaje WAP-PUSH. Representa una breve descripción del contenido multimedia. Debería contener exclusivamente caracteres incluidos en la lista del apartado 2.5.1. No debe sobrepasar los 115 caracteres junto a la longitud del parámetro "url" o bien los 88 caracteres si se opta por el envío de clave en el parametro "url" (ver sección 2.5.7). No puede estar vacío.	sí
url	Dirección de Internet desde donde el teléfono móvil se descargará el contenido. Debe contener caracteres validos en una URL (ver sección 2.5.2). No debe sobrepasar los 115 caracteres junto a la longitud del parámetro "msg" o bien los 88 caracteres si se opta por el envío de clave en este parametro (ver sección 2.5.7). No puede estar vacío. No es posible seleccionar un puerto de conexión diferente al 80, el habitual en la navegación WEB.	sí
ack	Solicitud de confirmación de entrega de los mensajes enviados (ver sección 2.6). Si vale "true" solicita confirmación de entrega de los SMS enviados. En su ausencia o si tiene otro valor no se solicita confirmación de entrega.	no
idAck	Código identificativo para la confirmación de entrega (ver sección 2.6). Valor alfanumérico de hasta 20 caracteres (números y letras de la "a" a la "z" tanto mayúsculas como minúsculas sin incluir ni "Ñ" ni "ñ"). De rebasar la longitud máxima permitida será truncado. Los caracteres no permitidos serán eliminados. Solo será considerado si el parámetro ack se envía con valor "true". Si se incluye explícitamente este parámetro y toma como valor cadena vacía, anula la solicitud de confirmación de entrega.	no

Cuadro 2.6: Lista de parámetros para sendwappush

Un ejemplo de petición REST al recurso sendWapPush sería este:

```
{"credentials":{"domainId":"XXXXX","login":"YYYYYY","passwd":"ZZZZZZ"},
 "message":{"msg":"Texto de alerta","url":"http://www.altiria.com?k=",
           "ack":"true","idAck":123456789},
 "destination":["346XXXXXXXX","346YYYYYYYY"]}
```

La respuesta a la petición del recurso sendWapPush está compuesta por el elemento "status" con valor un código de estado general de los descritos en el apartado 2.7

Si la operación ha resultado exitosa (código de estado "000") la respuesta contendrá adicionalmente un elemento "details" incluyendo para cada destinatario del envío (cada elemento de la lista "destination" de entrada) los siguientes datos:

- destination: se corresponde con el número de teléfono del destinatario.
- status: se corresponde con uno de los códigos de estado del apartado 2.7.
- idAck: se corresponde con el código de identificación asociado a la solicitud de confirmación de entrega (ver sección 2.6). Solo aparecerá si se solicita la confirmación de entrega y es aceptada.
- key: se corresponde con la clave única asociada al destinatario (ver sección 2.5.7). Solo aparecerá si se solicita la clave para cada destinatario

Un ejemplo de respuesta exitosa correspondiente al recurso **sendWapPush** del ejemplo anterior sería el mostrado a continuación:

```
{"details":[{"destination":"346XXXXXXXX","idAck":"123456789",
             "key":"346XXXXXXXX-22879196","status":"000"},
            {"destination":"346YYYYYYYY","idAck":"123456789",
             "key":"346YYYYYYYY-22879196","status":"000"}],
"status":"000"}
```

Un ejemplo de respuesta notificando un error en la autentificación sería este:

```
{"status":"020"}
```

La infomación de éxito para un destinatario concreto implica que el mensaje ha sido aceptado por la pasarela, no que haya sido enviado y recibido por el destinatario. Un mensaje puede ser aceptado aún cuando no se disponga de crédito suficiente para su envío (ver sección 2.3.4).

Para asegurar el adecuado funcionamiento de este recurso se recomienda probar un ciclo completo de servicio, desde el envío del mensaje WAP-PUSH hasta la descarga del contenido en el teléfono móvil, como paso previo a poner el sistema en producción.

2.3.4. Consulta del crédito disponible

Permite conocer el crédito instantáneo disponible para enviar mensajes.

Se trata del recurso con **"path" /getCredit**. El JSON está compuesto por los **elementos obligatorios** detallados en el cuadro 2.7.

Nombre	Valores	Composición
credentials	Datos de identificación del usuario suministrados por *Altiria*	Elementos domainId, login y passwd

Cuadro 2.7: Parámetros de entrada del recurso getCredit

En el cuadro 2.8 se detallan los elementos restantes constituyentes del JSON de la petición.

La única forma de averiguar si se tiene crédito suficiente para enviar los mensajes, aparte de llevar un contador propio de saldo disponible, es mediante una consulta previa a través de este recurso.

El recurso ofrece información del crédito disponible en un momento dado. Puesto que el sistema decrementa el crédito justo al enviar el mensaje al destinatario, es necesario seguir el siguiente esquema para utilizar adecuadamente el recurso de consulta de crédito disponible:

- Antes de comenzar con los envíos, se calcula cuantos mensajes en total se desean enviar, por ejemplo 5000.
- Se consulta el valor del crédito disponible una única vez.
- A partir del coste en créditos de cada mensaje a enviar y del saldo disponible se estima si se podrán enviar o no todos los mensajes.
- En caso positivo, se usan las peticiones de envío de mensajes. En caso negativo se debe adquirir más crédito

Cuando el sistema haya finalmente enviado todos los mensajes, una nueva consulta del crédito disponible ofrecerá el valor actualizado.

En cualquier caso la comprobación efectiva del saldo disponible para efectuar un envío se realiza en un proceso interno justo antes de efectuar el envío. En caso de que no se disponga de crédito suficiente, el mensaje no será enviado y el cliente será informado a través de correo electrónico. Si posteriormente se adquiere más crédito disponible, se podrá avisar a *Altiria* para reintentar los envíos pendientes.

Nombre	**Valor**	**Obligatorio**
domainId	Identificador suministrado por *Altiria* al cliente. Se puede omitir si el login es un email.	no
login	Identificador de usuario suministrado por *Altiria* al cliente.	sí
passwd	Clave del usuario suministrada por *Altiria* al cliente.	sí

Cuadro 2.8: Lista de parámetros para getcredit

Un ejemplo de petición REST al recurso getCredit sería este:

```
{"credentials":{"domainId":"XXXXXX","login":"YYYYYY","passwd":"ZZZZZZ"}}
```

La respuesta a la petición del recurso getCredit está compuesta por el elemento "status" con valor un código de estado general de los descritos en el apartado 2.7

Si la operación ha resultado exitosa (código de estado "000") la respuesta contendrá adicionalmente el valor del crédito disponible como un número con dos decimales.

Un ejemplo de respuesta exitosa correspondiente al servicio **getCredit** del ejemplo anterior sería el mostrado a continuación:

```
{"credit":"1000000.70","status":"000"}
```

Un ejemplo de respuesta notificando un error en la autentificación sería este:

```
{"status":"020"}
```

2.4. Mensajes de texto

Los **caracteres permitidos** para el texto del mensaje corto y la **longitud máxima** dependerán de la codificación de caracteres seleccionada: codificación por defecto (ver sección 2.4.1) o la codificación UNICODE (ver sección 2.4.2).

2.4.1. Codificación por defecto

La **codificación por defecto** permite los caracteres de la tabla 2.9.

La **longitud máxima permitida** se detalla en la sección 2.4.3.

Las vocales con tilde o acento agudo (á) son aceptadas pero se enviarán al teléfono móvil sin acentuar.

Adicionalmente se admiten los **caracteres extendidos** de la tabla 2.10. Cada carácter extendido **ocupa el doble espacio que un carácter normal**, esto debe considerarse para el cómputo de la longitud máxima del mensaje.

En caso de que el mensaje a enviar contenga **caracteres fuera de las listas** presentadas, estos serán **reemplazados por el carácter "?"** antes de enviar el mensaje.

@	(	4	_	L	W	h	s	Ú	ù
cr[1]	)	5	A	M	X	i	t	á	
lf[2]	*	6	B	N	Y	j	u	é	
Ç	+	7	C	Ñ	Z	k	v	í	
sp[3]	,	8	D	O	¿	l	w	ó	
!	-	9	E	P	a	m	x	ú	
"	.	:	F	Q	b	n	y	Ü	
#	/	;	G	R	c	ñ	z	ü	
$	0	<	H	S	d	o	Á	à	
%	1	=	I	T	e	p	É	è	
&	2	>	J	U	f	q	Í	ì	
'	3	?	K	V	g	r	Ó	ò	

Cuadro 2.9: Lista de caracteres permitidos para mensajes de texto en la codificación por defecto

[	]	\	^	{	}	|	~	€

Cuadro 2.10: Lista de caracteres extendidos permitidos para mensajes de texto

2.4.2. Unicode

La **codificación UNICODE**, forzada mediante el parámetro "unicode" (ver el cuadro 2.2), permite todo el juego de caracteres UNICODE de 16bits.

La **longitud máxima permitida** se detalla en la sección 2.4.3, siendo siempre menor que usando la codificación por defecto (ver la sección 2.4.1).

Con esta codificación sería posible por ejemplo el envío de vocales con tilde.

[1] Retorno de carro
[2] Nueva línea
[3] Espacio blanco

2.4.3. Longitud del mensaje

La longitud máxima de un mensaje de texto es un valor variable que depende de la codificación de caracteres usada y de la posibilidad de concatenación. Los mensajes que **excedan la longitud máxima aplicable serán rechazados** (no enviados).

- La longitud máxima de un mensaje corto con la **codificación por defecto es de 160 caracteres** (ver sección 2.4.1). Los caracteres extendidos (ver la tabla 2.10) ocupan el doble, por tanto la longitud máxima se reduce. Por ejemplo si el texto del SMS contuviera el símbolo del euro "€" y los corchetes "[ ]", la longitud máxima del mensaje corto se reduciría a 157 caracteres.
- La longitud máxima de un mensaje corto con la **codificación UNICODE es de 70 caracteres** (ver sección 2.4.2).

En caso de **definir puertos origen o destino** del SMS (ver los parámetros sPort y dPort en el cuadro 2.2) la **longitud máxima se reduce** de la siguiente forma:

- **152 caracteres para la codificación por defecto** (ver sección 2.4.1). Igualmente hay que considerar que los caracteres extendidos (ver la tabla 2.10) ocupan el doble.
- **66 caracteres para la codificación UNICODE** (ver sección 2.4.2).

Mediante el uso de **mensajes concatenados es posible ampliar esos límites**. Un mensaje concatenado consiste en varios mensajes en secuencia recibidos como un único mensaje en el teléfono del destinatario.

Los mensajes concatenados se posibilitan mediante el parámetro "concat" (ver el cuadro 2.2) siempre que no se esten definiendo ni el puerto origen ni el puerto destino del SMS (ver los parámetros sPort y dPort en el cuadro 2.2).

La plataforma de *Altiria* permite **concatenar hasta 10 mensajes**, aplicando en ese caso los límites siguientes a la longitud del mensaje:

- **1530 caracteres para la codificación por defecto** (ver sección 2.4.1). Igualmente hay que considerar que los caracteres extendidos (ver la tabla 2.10) ocupan el doble.
- **670 caracteres para la codificación UNICODE** (ver sección 2.4.2).

2.5. Mensajes WAP-PUSH

La interfaz HTTP de *Altiria* permite el envío de mensajes multimedia (imágenes, sonidos, juegos, etc) mediante la tecnología de los mensajes WAP-PUSH.

Los mensajes WAP-PUSH incluyen información sobre la ubicación de un determinado contenido multimedia, una dirección de Internet. En este sentido son completamente diferentes a los mensajes de texto normales, puesto que en estos el contenido relevante es el propio texto.

Básicamente un mensaje de este tipo se compone de un pequeño texto a modo de presentación del contenido que se ofrece y la dirección de Internet donde se ubica dicho contenido.

Cuando un teléfono móvil recibe un mensaje WAP-PUSH, le presenta al usuario la breve descripción mencionada junto con la posibilidad de descargarse el contenido multimedia referenciado. Si el usuario acepta, el teléfono de manera automática accede al contenido a través de HTTP, se lo descarga como si de un navegador WEB se tratara y lo almacena, mostrando además otras opciones en función del tipo de contenido (ver una imagen, reproducir un sonido...).

2.5.1. Caracteres permitidos

El cuadro 2.11 detalla los caracteres admisibles para los mensajes WAP-PUSH (parámetro “msg” del cuadro 2.6).

cr[1]	lf[2]	sp[3]	!	”	#	&
’	(	)	*	+	,	-
.	/	0	1	2	3	4
5	6	7	8	9	:	;
<	=	>	?	A	B	C
D	E	F	G	H	I	J
K	L	M	N	O	P	Q
R	S	T	U	V	W	X
Y	Z	a	b	c	d	e
f	g	h	i	j	k	l
m	n	o	p	q	r	s
t	u	v	w	x	y	z
Á	É	Í	Ó	Ú	á	é
í	ó	ú				

Cuadro 2.11: Lista de caracteres permitidos para los mensajes WAP-PUSH

Las vocales con tilde o acento agudo (á) son aceptadas pero se enviarán al teléfono móvil sin acentuar.

En caso de que el mensaje a enviar contenga caracteres fuera de la lista presentada, estos serán reemplazados por el carácter “?” y el mensaje será enviado.

2.5.2. Especificación de la URL

La URL de descarga de los contenidos multimedia suministrados a través de mensajes WAP-PUSH (parámetro “url” del cuadro 2.6) debe seguir las siguientes normas de composición:

- Los caracteres del cuadro 2.12 son seguros y se pueden incluir sin codificar.

[1] Retorno de carro
[2] Nueva línea
[3] Espacio blanco

- Los caracteres del cuadro 2.13 son reservados y se pueden incluir sin codificar si se emplean dentro de la URL de acuerdo a su uso reservado. Por ejemplo el carácter "&" se usa para separar los parametros de un formulario. Si estos caracteres se emplean de otro modo se deben codificar.
- Otros caracteres se pueden incluir previa codificación. De todos modos pueden no ser seguros y es posible que algunos presenten problemas en algunos teléfonos. Se recomienda prescindir de ellos siempre que sea posible.

a	b	c	d	e	f	g	h	i	j	k	l	m	n
o	p	q	r	s	t	u	v	w	x	y	z	A	B
C	D	E	F	G	H	I	J	K	L	M	N	O	P
Q	R	S	T	U	V	W	X	Y	Z	0	1	2	3
4	5	6	7	8	9	-	_	.	!	*	'	(	)

Cuadro 2.12: Lista de caracteres seguros

$	&	+	,	/	:	;	=	?	@

Cuadro 2.13: Lista de caracteres reservados

La codificación de un carácter se logra a partir de su representación en hexadecimal en un determinado juego de caracteres, insertando el simbolo "%" por cada par de dígitos hexadecimales. Por ejemplo la "ñ" en UTF-8 se codificaría como "%C3%B1".

El juego de caracteres a escoger debería ser el del servidor que albergue el contenido a descargar mediante el mensaje WAP-PUSH.

Según lo visto si se desea permitir la descarga de un contenido de la URL:

```
http://www.miempresa.com/contenidos/imagen[1].jpg
```

se debe enviar como (usando ISO8859-1):

```
http://www.miempresa.com/contenidos/imagen%5B1%5D.jpg
```

Es importante reseñar que cada carácter codificado ocupa un número mayor de caracteres en el cómputo de la longitud completa de la URL.

En cualquier caso se recomienda probar la correcta descarga de los contenidos desde la URL seleccionada para comprobar que todo el proceso se efectúa correctamente.

2.5.3. Tipos de contenido

El contenido más general que se puede enviar es una página "wml". Las páginas "wml" son similares a las conocidas páginas web, adaptadas a los requisitos de un teléfono móvil.

De este modo se podrá enviar un contenido formado por texto y otros tipos de archivos como imágenes (ej: jpg y gif) o sonidos (ej: midi), incluidos en la propia página.

También es posible enviar directamente el archivo multimedia al teléfono, evitando incluirlo en una página "wml".

Actualmente hay mucha diversidad de teléfonos móviles, cada uno con sus propias capacidades multimedia. Es posible que determinados teléfonos no sean capaces de manejar algunos tipos de archivos. Para esas situaciones, la posibilidad de incluir texto en una página "wml", un recurso manejado por

todos los terminales WAP, permite al menos que el teléfono acceda a parte de la información. A este respecto una buena práctica es incluir en el texto información para el destinatario sobre la opción de acceder al fichero multimedia a través de un navegador web convencional, adjuntando la información relativa a la dirección de Internet.

En caso de optar por la inclusión de texto en una página "wml" se recomienda emplear un juego de caracteres sencillo, reconocible por la mayoría de los teléfonos. Como referencia se puede usar el detallado en el cuadro 2.11, sin incluir las vocales acentuadas.

2.5.4. Formato del contenido

Independientemente del tipo de contenido escogido, siempre se debería considerar que el medio habitual de acceso al mismo será un teléfono móvil.

Esto tiene importantes incidencias en cuanto al tamaño máximo de la información suministrada. Se recomienda no enviar contenidos que ocupen más de 10kB, sobre todo si se suministran embebidos en páginas "wml".

Para optimizar el tamaño, se sugiere adaptar los contenidos a los requerimientos de un teléfono móvil. Por ejemplo si se trata de una imagen es conveniente ajustar su tamaño al habitual de la pantalla, guardando además una relación de aspecto adecuada para que al recibirla ocupe el máximo en todas las direcciones. Una buena medida como referencia pueden ser 240 x 240 "pixels".

2.5.5. Dirección del contenido

El teléfono móvil conoce la ubicación del contenido multimedia mediante la información de dirección que le llega en el mensaje WAP-PUSH.

Es obvio que para que el teléfono pueda descargarse la información la dirección debe respresentar la ubicación de un recurso accesible públicamente a través de HTTP, mediante navegación WEB.

Un detalle importante asociado a la dirección del contenido es que muchos teléfonos la emplean como identificador de los mensajes WAP-PUSH recibidos. Esto supone que si se recibe un mensaje WAP-PUSH con la misma dirección del contenido asociado que un mensaje ya recibido y almacenado en el teléfono, el nuevo mensaje reemplazará al antiguo.

Existen sin embargo teléfonos que no siguen este patrón y almacenan los dos mensajes con idéntica dirección del contenido de forma independiente.

2.5.6. Envío del contenido

Cuando el teléfono móvil solicita el contenido referenciado en el mensaje WAP-PUSH, envía una petición HTTP GET (en algunos casos se envía un HTTP HEAD previamente) a la dirección apropiada.

Es necesario entonces un servidor HTTP que atienda la petición y entregue el contenido apropiadamente.

2.5.7. Control de acceso al contenido

La URL indicada en el campo "url" del recurso "sendWapPush" será enviada en el mensaje WAP-PUSH a cada destinatario para que los interesados se descarguen el contenido ahí alojado. Para poder distinguir qué destinatarios han accedido realmente al contenido *Altiria* ofrece la posibilidad de enviar un mensaje diferente a cada uno de ellos. A la URL a suministrar al teléfono se le añadirá un parámetro identificador, una clave. Este parámetro estará unívocamente asociado al teléfono del destinatario mediante la respuesta generada por la pasarela al recurso "sendWapPush". Cuando se reciba una petición de descarga se podrá extraer la clave y de esta manera obtener información del destinatario.

Si se desea solicitar el envío de clave es preciso que la URL suministrada en el parámetro "url" termine con la subcadena "k=". *Altiria* agregará en ese caso a la URL enviada en cada mensaje la clave única con el formato: "xxxxxxxxxx-zzzzzzzzzz"; 'xxxxxxxxxx" representa el número de teléfono del destinatario en formato internacional y "zzzzzzzzzz" representa un número asociado a cada petición al recurso "sendWapPush", como máximo de diez cifras.

Es necesario resaltar que si se opta por este método la longitud total para los parametros "msg" y "url" pasará de 115 caracteres a 88.

Finalmente para clarificar todos los elementos del servicio de envío de mensajes multimedia a través del recurso "sendWapPush" se esquematizan los procesos involucrados en el envío de un mensaje a dos destinatarios solicitando claves independientes:

1. El cliente accede al recurso "sendWapPush" de la pasarela de *Altiria* incluyendo los siguientes parámetros:

   - destination=346xxxxxxxx.
   - destination=346yyyyyyyy.
   - url=www.miempresa.com/contenidos/descarga.php?k=

   Se observa que se desea enviar el contenido multimedia a dos destinatarios y que además la URL acaba con la subcadena "k=", solicitando entonces la generación de claves identificadoras.

2. La pasarela de *Altiria* recibe la petición y remite dos mensajes WAP-PUSH individuales a los destinatarios seleccionados. Como URL de descarga, los respectivos teléfonos móviles recibirán (la segunda parte de la clave es simplemente un ejemplo) :

   - El móvil "346xxxxxxxx": www.miempresa.com/contenidos/descarga.php?k=346xxxxxxxx-12365
   - El móvil "346yyyyyyyy": www.miempresa.com/contenidos/descarga.php?k=346yyyyyyyy-12365

   Además la pasarela de *Altiria* responde al cliente un status general "000" y los siguientes datos para cada destinatario:

   ```
   status=000; destination=346xxxxxxxx; key=346xxxxxxxx-12365
   status=000; destination=346yyyyyyyy; key=346yyyyyyyy-12365
   ```

3. Ambos destinatarios reciben en su teléfono la solicitud de aceptación de descarga del contenido multimedia. Suponemos que el destinatario con el número de teléfono "346xxxxxxxx" acepta la descarga.

4. El servidor WEB del cliente, habilitado para ofrecer los contenidos multimedia, recibe una petición de descarga. La petición estará dirigida a la URL

   ```
   http://www.miempresa.com/contenidos/descarga.php?k=346xxxxxxxx-12365
   ```

   por lo que podrá asociarla directamente al destinatario con número de teléfono "346xxxxxxxx".

5. El servidor WEB entrega el contenido al teléfono del destinatario.

6. El teléfono del destinatario muestra el contenido.

2.6. Confirmación de entrega

El servicio de confirmación de entrega, solicitado a través del parámetro "ack" de los recursos de envío, permite recibir notificaciones con información sobre el estado de entrega de los mensajes cortos enviados mediante la pasarela.

Para tener acceso a este servicio es preciso que el cliente haya notificado a *Altiria* la dirección de Internet a donde se enviarán las informaciones de confirmación de entrega. En caso contrario, las solicitudes de confirmación de entrega serán ignoradas aunque los mensajes sí serán enviados.

Para que el cliente pueda asociar la información recibida sobre el estado de entrega de un mensaje con el propio mensaje enviado previamente a través de la pasarela, se usará el identificador devuelto en la respuesta a las peticiones de envio en la parte "idAck".

Este identificador puede albergar dos tipos de valores:

- Si en la propia petición de envío se incluye el parámetro "idAck" (es opcional), contendrá ese valor truncado a veinte caracteres y formado solo por caracteres válidos. Es importante constatar que el identificador devuelto en este caso solo coincidirá con el suministrado en la correspondiente petición de envío si cumple los criterios de composición explicados en las tablas 2.2 y 2.6.
- Si en la propia petición de envío no se incluye el parámetro "idAck", contendrá un valor numérico de diez dígitos como máximo generado por la pasarela automáticamente.

Las **notificaciones del estado de entrega** serán enviadas mediante **una petición al recurso REST** que debe exponer el cliente para desarrollar este servicio.

El cliente debe suministrar la URL donde se accederá al recurso.

La petición consta de un mensaje JSON codificado en UTF-8 (content-type: "application/json;charset=UTF-8") con un único elemento "notification".

En el cuadro 2.14 se detallan los elementos que a su vez lo componen.

Dato	Valor
destination	Número de teléfono móvil al que se refiere la información de estado de entrega. Si esa información es relativa a un mensaje concatenado, el teléfono figurará cualificado con un sufijo que haga referencia al fragmento particular del que se trate, por ejemplo xxxxxxxxxxx(2) (ver la respuesta al enviar un mesaje concatenado en la sección 2.3.1).
idAck	Identificador que coincidirá con el suministrado por *Altiria* al cliente en la parte "idAck" de la respuesta a la operación de envío. El contenido de este campo se ha explicado en el inicio de esta sección.
status	Estado relativo a la entrega del mensaje. Podrá tomar los valores: "ENTREGADO", "NO ENTREGADO", "ERROR_100", "ERROR_101", "ERROR_114" o "ERROR_115".

Cuadro 2.14: Lista de datos de la confirmación de entrega

El estado "NO ENTREGADO" aparece cuando el mensaje no puede ser entregado al teléfono móvil. Es un estado definitivo, por lo que ese mensaje particular nunca será entregado. Las causas pueden ser múltiples, desde que el teléfono haya estado apagado durante un tiempo superior al periodo de validez, hasta que el número no exista. En general no se puede conocer la causa.

El estado "ERROR_100" indica que el mensaje por el momento no ha podido ser entregado al destinatario debido a algún problema en su teléfono móvil. Las causas más comunes son: mala cobertura, buzón de mensajes cortos lleno o teléfono apagado. El mensaje se intentará enviar varias veces

con posterioridad durante un tiempo limitado. Si el problema en el teléfono se subsana a tiempo, el mensaje será finalmente entregado, recibiéndose la correspondiente confirmación.

El estado "ERROR_101" indica que el mensaje por el momento no ha podido ser entregado al destinatario debido a algún problema en la red de telefonía móvil del operador. Habitualmente, cuando el operador solvente los problemas, el mensaje será entregado, recibiéndose la correspondiente confirmación.

El estado "ERROR_114" indica que el mensaje ha sido enviado pero no ha podido ser entregado porque el número de teléfono destinatario no existe.

El estado "ERROR_115" indica que el mensaje ha sido enviado pero no ha podido ser entregado porque el destinatario no acepta mensajes.

Un ejemplo de notificación de entrega correspondiente a uno de los fragmentos del envío del SMS concatenado del ejemplo del recurso sendSms (ver sección 2.3.1) sería este:

```
{"notification": {"destination":"346YYYYYYYY(1)","idAck":"123456789","status":"ENTREGADO"}}
```

Opcionalmente se podría configurar el envío de las notificaciones de entrega mediante la llamada a un servicio web del cliente según la especificación relativa a la confirmación de entrega de la pasarela de servicios web de *Altiria* para el envío de SMS.

Otra alternativa sería configurar el envío de las notificaciones de entrega mediante peticiones HTTP POST como también se detalla en la especificación relativa a la confirmación de entrega de la pasarela HTTP de *Altiria* para el envío de SMS.

Consultar al soporte técnico de *Altiria* (soporte@altiria.com) para concer más detalles sobre estas posibilidades.

El **recurso del cliente deberá responder** un único contenido simple como por ejemplo la cadena "OK".

Finalmente para clarificar todos los elementos de la funcionalidad de confirmación de entrega, se esquematizan los procesos involucrados en el envío de un mensaje a dos destinatarios:

1. El cliente efectúa un envío de SMS a través de la pasarela de *Altiria* incluyendo entre otros los siguientes parámetros:
   - destination=346xxxxxxxx, 346yyyyyyyy.
   - ack=true.
   - idAck=zzzz.
2. La pasarela de *Altiria* recibe la petición y remite el mensaje a los destinatarios seleccionados. Además la pasarela responde al cliente un status general "000" y los siguientes datos para cada destinatario:

   ```
   status=000; destination=346xxxxxxxx; idAck=zzzz
   status=000; destination=346yyyyyyyy; idAck=zzzz
   ```

   Si el cliente no hubiera incluido el parámetro "idAck" en la operación "sendSms" (punto 1 del ejemplo), la pasarela de *Altiria* habría autogenerado el identificador para añadirlo a la respuesta.
3. Ambos destinatarios reciben en su teléfono el mensaje corto enviado.
4. El recurso REST del cliente, habilitado para recibir las notificaciones de estado de entrega, es accedido desde la pasarela de *Altiria* en sendas peticiones con el parámetro "notification" conteniendo:

   ```
   destination=346xxxxxxxx; idAck=zzzz; status=ENTREGADO
   destination=346yyyyyyyy; idAck=zzzz; status=ENTREGADO
   ```

5. El recurso del cliente responde la cadena de texto “OK”.
6. Empleando el identificador zzzz, el cliente podrá asociar la confirmación de entrega con el mensaje previamente enviado a cada uno de los destinatarios.

2.7. Códigos de estado

El cuadro 2.15 presenta la lista de los posibles códigos de estado que podrán aparecer en la respuesta a cada petición.

CÓDIGO	DETALLE
000	Éxito
001	Error interno. Contactar con el soporte técnico
002	Error de acceso al puerto seguro 443. Contactar con el soporte técnico
010	Error en el formato del número de teléfono
011	Error en el envío de los parámetros de la petición o codificación incorrecta.
013	El mensaje excede la longitud máxima permitida
014	La petición HTTP usa una codificación de caracteres inválida
015	No hay destinatarios válidos para enviar el mensaje
016	Destinatario duplicado
017	Mensaje vacío
018	Se ha excedido el máximo número de destinatarios autorizado
019	Se ha excedido el máximo número de mensajes autorizado
020	Error en la autentificación
022	El remitente seleccionado para el envío no es válido
030	La url y el mensaje superan la longitud máxima permitida
031	La longitud de la url es incorrecta
032	La url contiene caracteres no permitidos
033	El puerto destino del SMS es incorrecto
034	El puerto origen del SMS es incorrecto

Cuadro 2.15: Lista de los códigos de estado

2.8. Ejemplos

Se presentan extractos de programación en varios lenguajes.

El **código fuente** todos los ejemplos se puede **descargar** de esta URL [CODIGOFUENTE]

Altiria no se responsabiliza del funcionamiento de los ejemplos presentados. Se deben considerar como fragmentos de código ilustrativos del acceso a algunas funcionalidades de la pasarela documentada.

Con objeto de **facilitar la lectura algunas líneas del código han sido partidas** con saltos de línea que podrían afectar al correcto funcionamiento del programa en su ejecución.

2.8.1. Envío de un mensaje en PHP

Ejemplo desarrollado para PHP7. En versiones anteriores es posible que se requiera instalar el soporte para la librería curl.

```
<?php

function AltiriaSMS($sDestination, $sMessage, $sSenderId, $debug){
if($debug)
  echo 'Enter AltiriaSMS <br/>';

//URL base de los recursos REST
$baseUrl = 'http://www.altiria.net/apirest/ws';

//Se inicia el objeto CUrl
$ch = curl_init($baseUrl.'/sendSms');

//XX, YY y ZZ se corresponden con los valores de identificación del
//usuario en el sistema.
$credentials = array(
  'domainId' => 'XX',
  'login'    => 'YY',
  'passwd'   => 'ZZ'
);

$destinations = explode(',', $sDestination);

$jsonMessage = array(
  'msg' => substr($sMessage,0,160),
  'senderId' => $sSenderId
);

$jsonData = array(
  'credentials' => $credentials,
  'destination' => $destinations,
  'message'     => $jsonMessage
);

//Se construye el mensaje JSON
$jsonDataEncoded = json_encode($jsonData);

//Indicamos que nuestra petición sera Post
curl_setopt($ch, CURLOPT_POST, 1);
```

```
//Se fija el tiempo máximo de espera para conectar con el servidor (5 segundos)
curl_setopt($ch, CURLOPT_CONNECTTIMEOUT, 5);

//Se fija el tiempo máximo de espera de la respuesta del servidor (60 segundos)
curl_setopt($ch, CURLOPT_TIMEOUT, 60);

//Para que la peticion no imprima el resultado como un 'echo' comun
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);

//Se añade el JSON al cuerpo de la petición codificado en UTF-8
curl_setopt($ch, CURLOPT_POSTFIELDS, $jsonDataEncoded);

//Se fija el tipo de contenido de la peticion POST
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type: application/json;charset=UTF-8'));

//Se envía la petición y se consigue la respuesta
$response = curl_exec($ch);

$statusCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);

if($debug) {
  //Error en la respuesta del servidor
  if($statusCode != 200){
    echo 'ERROR GENERAL: '.$statusCode;
    echo $response;
  }else{
    //Se procesa la respuesta capturada
    echo 'Código de estado HTTP: '.$statusCode.'<br/>';
    $json_parsed = json_decode($response);
    $status = $json_parsed->status;
    echo 'Código de estado Altiria: '.$status.'<br/>';
    if ($status != '000')
      echo 'Error: '.$response.'<br/>';
    else{
      echo 'Cuerpo de la respuesta: <br/>';
      echo 'destails[0][destination]: '.$json_parsed->details[0]->destination.'<br/>';
      echo 'destails[0][status]: '.$json_parsed->details[0]->status.'<br/>';
      echo 'destails[1][destination]: '.$json_parsed->details[1]->destination.'<br/>';
      echo 'destails[1][status]: '.$json_parsed->details[1]->status.'<br/>';
    }
  }
}

//Si ha ocurrido algún error se lanza una excepción
if(curl_errno($ch))
  throw new Exception(curl_error($ch));

return $response;
}


try{
  echo "The function AltiriaSMS returns: "
   .AltiriaSMS('346xxxxxxxx,346yyyyyyyy','Mensaje de prueba', '', true);
```

```
  //No es posible utilizar el remitente en América pero sí en España y Europa
  //Utilizar esta llamada solo si se cuenta con un remitente autorizado por Altiria
  //echo "The function AltiriaSMS returns: "
  // .AltiriaSMS('346xxxxxxxx,346yyyyyyyy','Mensaje de prueba', 'remitente', true);
}catch(Exception $e){
  echo 'Error: '.$e->getMessage();
}

?>
```

2.8.2. Envío de un mensaje en JAVA

Ejemplo en Java utilizando Httpclient 4.5 como cliente HTTP (ver [HTTPCLIENT]).

Esta librería depende de los siguientes paquetes:

- commons-logging versión 1.2
- commons-codec versión 1.9
- httpcore versión 4.4.3

```
//Se construye el mensaje JSON
JsonObject textMessageFilter = new JsonObject();
JsonObject credentials = new JsonObject();
//XX, YY y ZZ se corresponden con los valores de identificación del
//usuario en el sistema.
credentials.addProperty("domainId","XX");
credentials.addProperty("login","YY");
credentials.addProperty("passwd","ZZ");

JsonArray destinations = new JsonArray();
destinations.add(new JsonPrimitive(new String("346xxxxxxx")));
destinations.add(new JsonPrimitive(new String("346yyyyyyyy")));

JsonObject textMessage = new JsonObject();
textMessage.addProperty("msg", "Mensaje de prueba");

//No es posible utilizar el remitente en América pero sí en España y Europa
//Descomentar la línea solo si se cuenta con un remitente autorizado por Altiria
//textMessage.addProperty("senderId", "remitente");

textMessageFilter.add("credentials", credentials);
textMessageFilter.add("destination", destinations);
textMessageFilter.add("message", textMessage);

//Se fija el tiempo máximo de espera para conectar con el servidor (5000)
//Se fija el tiempo máximo de espera de la respuesta del servidor (60000)
RequestConfig config = RequestConfig.custom()
 .setConnectTimeout(5000)
 .setSocketTimeout(60000)
 .build();

//Se inicia el objeto HTTP
HttpClientBuilder builder = HttpClientBuilder.create();
builder.setDefaultRequestConfig(config);
```

```
CloseableHttpClient httpClient = builder.build();

//Se fija la URL base de los recursos REST
String baseUrl = "http://www.altiria.net/apirest/ws";
HttpPost request = new HttpPost(baseUrl+"/sendSms");

//Se añade el JSON al cuerpo de la petición codificado en UTF-8
request.setEntity(new StringEntity(textMessageFilter.toString(),"UTF-8"));

//Se fija el tipo de contenido de la peticion POST
request.addHeader("content-type", "application/json;charset=UTF-8");

CloseableHttpResponse response = null;

try {
  System.out.println("Enviando petición");
  //Se envía la petición
  response = httpClient.execute(request);
  //Se consigue la respuesta
  String resp = EntityUtils.toString(response.getEntity());

  //Error en la respuesta del servidor
  if (response.getStatusLine().getStatusCode()!=200){
   System.out.println("ERROR: Código de error HTTP:  "
                      + response.getStatusLine().getStatusCode());
   System.out.println("Compruebe que ha configurado correctamente la "
                      + "direccion/url y el content-type");
   return;
  }else {
   //Se procesa la respuesta capturada en la cadena 'response'
   if (resp.startsWith("ERROR")){
    System.out.println(resp);
    System.out.println("Código de error de Altiria. Compruebe las especificaciones");
   }else
    System.out.println(resp);
  }
}
catch (Exception e) {
  System.out.println("Excepción");
  e.printStackTrace();
  return;
}
finally {
  //En cualquier caso se cierra la conexión
  request.releaseConnection();
  if(response!=null) {
   try {
    response.close();
   }
   catch(IOException ioe) {
    System.out.println("ERROR cerrando recursos");
   }
  }
}
```

2.8.3. Envío de un mensaje en Python

Ejemplo en Python utilizando la librería Requests como cliente REST (ver [REQUESTS]).

```
# -*- coding: utf-8 -*-

import requests
import json as JSON

def altiriaSms(destinations, message, senderId, debug):
  if debug:
    print 'Enter altiriaSms: '+destinations+', message: '+message+', senderId: '+senderId

    try:

      #Se fija la URL base de los recursos REST
      baseUrl = 'http://www.altiria.net/apirest/ws'

      #Se construye el mensaje JSON
      #XX, YY y ZZ se corresponden con los valores de identificación del usuario en el sistema.
      credentials = {'domainId': 'XX', 'login': 'YY', 'passwd': 'ZZ'}
      destination = destinations.split(",")

      messageData = {'msg': message, 'senderId': senderId}
      jsonData = {'credentials': credentials, 'destination': destination, 'message': messageData}

      #Se fija el tipo de contenido de la peticion POST
      contentType = {'Content-Type':'application/json;charset=UTF-8'}
      #Se añade el JSON al cuerpo de la petición
      #Se envía la petición y se recupera la respuesta
      r = requests.post(baseUrl+'/sendSms',
                        data=JSON.dumps(jsonData),
                        headers=contentType,
                        #Se fija el tiempo máximo de espera para conectar con el servidor (5 seg.)
                        #Se fija el tiempo máximo de espera de la respuesta del servidor (60 seg)
                        timeout=(5, 60)) #timeout(timeout_connect, timeout_read)

      if debug:
      #Error en la respuesta del servidor
        if str(r.status_code) != '200':
          print 'ERROR GENERAL: '+str(r.status_code)
          print r.text
        else:
        #Se procesa la respuesta capturada
          print 'Código de estado HTTP: '+str(r.status_code)
          jsonParsed = JSON.loads(r.text)
          status = str(jsonParsed['status'])
          print 'Código de estado Altiria: '+status
          if status != '000':
            print 'Error: '+r.text
            else:
              print 'Cuerpo de la respuesta:'
              print "details[0]['destination']: "+str(jsonParsed['details'][0]['destination'])
              print "details[0]['status']: "+str(jsonParsed['details'][0]['status'])
              print "details[1]['destination']: "+str(jsonParsed['details'][1]['destination'])
```

```
            print "details[1]['status']: "+str(jsonParsed['details'][1]['status'])

      return r.text

  except  requests.ConnectTimeout:
    print "Tiempo de conexión agotado"

  except  requests.ReadTimeout:
    print "Tiempo de respuesta agotado"

  except Exception as ex:
    print "Error interno: "+str(ex)

print 'The function altiriaSms returns: \n'
      +altiriaSms('346xxxxxxxx,346yyyyyyyy','Mensaje de prueba', '', True)
#No es posible utilizar el remitente en América pero sí en España y Europa
#Utilizar esta llamada solo si se cuenta con un remitente autorizado por Altiria
#print 'The function altiriaSms returns: \n'
#     +altiriaSms('346xxxxxxxx,346yyyyyyyy','Mensaje de prueba', 'remitente', True)
```

2.8.4. Envío de un mensaje en Ruby

Ejemplo en Ruby de cliente REST.

```
# encoding: UTF-8

require 'net/http'
require 'json'
require 'uri'

def altiriaSms(destinations, message, senderId, debug)
  if debug
    puts "Enter altiriaSms: destinations=#{destinations}, message=#{message}, senderId=#{senderId}"
  end

  begin
    #XX, YY y ZZ se corresponden con los valores de identificación del
    #usuario en el sistema.
    credentials = {:domainId => "XX", :login => "YY", :passwd => "ZZ"}
    destination = destinations.split(",")

    messageData = {:msg => message, :senderId => senderId}
    #Se construye el mensaje JSON
    jsonData = {:credentials => credentials, :destination => destination, :message => messageData}

    #Se fija la URL base de los recursos REST
    baseUrl = 'http://www.altiria.net/apirest/ws'
    uri = URI.parse(baseUrl+"/sendSms")
    http = Net::HTTP.new(uri.host, uri.port)
    #Se fija el tiempo máximo de espera para conectar con el servidor (5 segundos)
    #Se fija el tiempo máximo de espera de la respuesta del servidor (60 segundos)
    http.open_timeout = 5
    http.read_timeout = 60
```

```
    #Se inicia el objeto HTTP y se envía la petición
    #Se añade el JSON al cuerpo de la petición codificado en UTF-8
    request = Net::HTTP::Post.new(uri.request_uri,'Content-Type'=>'application/json;charset=UTF-8')
    request.body = jsonData.to_json

    #Se consigue la respuesta
    response = http.request(request)

    if debug
      #Error en la respuesta del servidor
      unless response.code == "200"
        puts("ERROR GENERAL: #{response.code}")
        puts("#{response.body}")
      else
        #Se procesa la respuesta capturada
        puts("Código de estado HTTP: #{response.code}")
        jsonResponse = JSON.parse(response.body)
        puts("Código de estado de Altiria: #{jsonResponse['status']}")
        unless jsonResponse['status'].include? "000"
          puts("Error de Altiria: #{response.body}")
        else
          puts("Cuerpo de la respuesta:")
          puts("details[0]destination: #{jsonResponse['details'][0]['destination']}")
          puts("details[0]status: #{jsonResponse['details'][0]['status']}")
          puts("details[1]destination: #{jsonResponse['details'][1]['destination']}")
          puts("details[1]status: #{jsonResponse['details'][1]['status']}")
        end
      end
    end

  return response

rescue Net::OpenTimeout
  puts "Tiempo de conexión agotado"
rescue Net::ReadTimeout
  puts "Tiempo de respuesta agotado"
rescue Exception => e
  puts "Error interno: #{e}"
end

end

puts "The function altiriaSms returns: "
     +"#{altiriaSms('346xxxxxxxx,346yyyyyyyy','Mensaje de prueba', '', true).body}"
#No es posible utilizar el remitente en América pero sí en España y Europa
#Utilizar esta llamada solo si se cuenta con un remitente autorizado por Altiria
#puts "The function altiriaSms returns: "
#    +"#{altiriaSms('346xxxxxxxx,346yyyyyyyy','Mensaje de prueba', 'remitente', true).body}"
```

2.8.5. Envío de un mensaje en Perl

Ejemplo en Perl (ver [PERL]) de cliente REST.

```
#!/usr/bin/perl
```

```
require HTTP::Request;
require LWP::UserAgent;
use JSON;
use strict;
use warnings;

binmode(STDOUT, ":utf8");

sub altiriaSms{

  if ($_[3] eq 'true'){
    print 'Enter altiriaSms: destinations='.$_[0].', message='.$_[1].', senderId='.$_[2]."\n";
  }

  try {
    #Se fija la URL base de los recursos REST
    my $base_url = 'http://www.altiria.net/apirest/ws';

    my @array = split(',', $_[0]);

    my $json = JSON->new->utf8;

    #Se construye el mensaje JSON
    #XX, YY y ZZ se corresponden con los valores de identificación del usuario en el sistema.
    my $data_to_json = {credentials=>{domainId=>'XX',login=>'YY',passwd=>'ZZ'},
                        destination=> \@array,
                        message=>{msg=>$_[1],senderId=>$_[2]}
    };

    #Se inicia el objeto HTTP
    my $request = HTTP::Request->new( 'POST', $base_url.'/sendSms' );
    #Se añade el JSON al cuerpo de la petición codificado en UTF-8
    $request->header( 'Content-Type' => 'application/json;charset=UTF-8' );
    #Se añade el JSON a la petición
    $request->content( $json->encode($data_to_json) );

    my $lwp = LWP::UserAgent->new;
    #Se fija el tiempo máximo de espera de la respuesta del servidor (60 segundos)
    $lwp->timeout(60);
    #Se consigue la respuesta
    my $response = $lwp->request( $request );

    my $message = $response->decoded_content;

    #El cliente HTTP en caso de timeout devuelve un estado 500 con su correspondiente mensaje
    if(index($message, 'timeout') != -1){
      die 'Timeout error'; #Se lanza una excepción
    }

    if ($_[3] eq 'true'){
      if ( $response->is_success ) {
        print "Código de estado HTTP: ".$response->code."\n";
        my $decoded = decode_json($message);
        my $altiria_status = $decoded->{'status'};
        print "Código de estado de Altiria: " .$altiria_status. "\n";
```

```
        if($altiria_status eq '000'){#Se procesa la respuesta capturada
          print "Cuerpo de la respuesta: \n";
          print "details[0]destination: ".$decoded->{'details'}[0]{'destination'}."\n";
          print "details[0]status: ".$decoded->{'details'}[0]{'status'}."\n";
          print "details[1]destination: ".$decoded->{'details'}[1]{'destination'}."\n";
          print "details[1]status: ".$decoded->{'details'}[1]{'status'}."\n";
        }else{#Error en la respuesta del servidor
          print "Error de Altiria: ".$message."\n";
        }
      } else {
        print "ERROR GENERAL: ".$response->code."\n";
        print $message."\n";
      }
    }
    return $message;
  };
  catch {
    print "Error: ".$@->what."\n";
  };
}

print "The function altiriaSms returns: "
.altiriaSms('346xxxxxxxx,346yyyyyyyy','Mensaje de prueba', '', 'true')."\n";
#No es posible utilizar el remitente en América pero sí en España y Europa
#Utilizar esta llamada solo si se cuenta con un remitente autorizado por Altiria
#print "The function altiriaSms returns: "
# .altiriaSms('346xxxxxxxx,346yyyyyyyy','Mensaje de prueba', 'remitente', 'true')."\n";
```

Referencias

[FAQ] *Preguntas frecuentes de la pasarela de envío de SMS de Altiria:*
http://www.altiria.com/preguntas-frecuentes-faq-pasarela-sms-api

[HTTPCLIENT] *El proyecto HTTPCLIENT de Apache:*
http://hc.apache.org/httpcomponents-client-ga/

[NUSOAP] *Librería NuSOAP en Sourceforge:*
http://sourceforge.net/projects/nusoap/

[WSIMPORT] *Herramienta para generación de WS en Java:*
http://docs.oracle.com/javase/8/docs/technotes/tools/unix/wsimport.html

[PERL] *Página oficial de descargas de perl:*
https://www.perl.org/get.html

[CODIGOFUENTE] *Enlace a la descarga del código fuente de los ejemplos:*
https://static.altiria.com/especificaciones/altiria-push-ejemplos-codigo-fuente.zip

[REQUESTS] *Página oficial de Requests:*
http://docs.python-requests.org

[SAVON] *Página oficial de Savon:*
http://savonrb.com

[SUDS] *API de Suds:*
https://pypi.python.org/pypi/suds

[SOAPLite] *API de SOAP::Lite:*
http://search.cpan.org/ phred/SOAP-Lite-1.20/lib/SOAP/Lite.pm