myMSP API v2 Appendix HTTP Push

System Interface – Specifications

1 Purpose of Document

The purpose of this document is to define the HTTP push interface made available by myMSP to send data to the computer systems of external parties/customers.

1.1 Author

21st Century Mobile AB
Mikael Rosvall
mikael.rosvall@21st.se
+46 (0)8 21 21 55

1.2 Document Status

- Published 2009-12-08

1.2.1 Document changes

- B [2012-02-23]
  - New parameter added: destination (5.4)
  - New parameter added: externalRef (5.4)
  - New parameter added: isPremium (5.4)
- C [2019-03-11]
  - Updated graphic profile.

1.3 Document Perimeter

Neither economical questions nor questions regarding agreements/contracts will be dealt with in this document.

The information given in this document may change without notice and describes only the matters defined in the general part of this document. Please verify that your company has the most recent version. This information is intended for the use of customers and partners of 21st Century Mobile. The information or statements given in this document concerning the suitability, capacity or performance of the mentioned service cannot be considered binding but shall be defined in the agreement concluded between 21st Century Mobile and the customer, if applicable. 21st Century Mobile shall not be responsible in any event for errors in this document or for any damages, incidental or consequential (including monetary losses), that might arise from the use of this publication or the information in it. This material and the service described in this document are copyrighted in accordance with the applicable laws.

2 Technical description

myMSP can transmit to an external system (client) through HTTP or HTTPS using the POST method to submit the data. The push interface is used to send data, like incoming messages and delivery status updates, directly to a client thus eliminating the need for that system to constantly poll myMSP for updates. The client system must set up a HTTP listener that can process the request and return a response. The listener must be accessible through an URL, for example:

- http://www.mysite.com/listener

It is possible to specify optional parameters that should be included in every request send from myMSP, e.g. a password to increase security:

- http://www.mysite.com/listener?pwd=123456

The myMSP HTTP push interface is designed to be compatible with the Web service interface, i.e. it directly maps the Web service classes to request parameters. This relationship is more thoroughly discussed in the next chapter. However, it is also possible to use the push interface in conjunction with both the HTTPS- and XML-API:s, but not with the SMPP interface.

To activate HTTP-push send an e-mail to support@21st.se and include the following information:

- The **user name** of the account that the push functionality should be activated for.
- The **target URL** to which the push messages should be sent.
- Any **optional parameters** that should be included in every request. Not required.

3 Usage scenarios

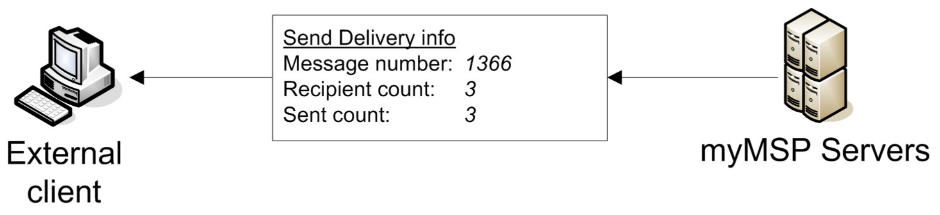
The following scenarios are intended to give a brief overview of how to use the myMSP push interface.

Note that all messages sent from myMSP require the client application to **always** respond with HTTP response code 200 if the message has been correctly received. If myMSP does not receive a response code with the value 200 then the message will be sent again. The client application should acknowledge the message as soon as it arrives as the connection has a read timeout of 10 seconds. Any processing of the message, like updating the database, should be done after the acknowledgment.

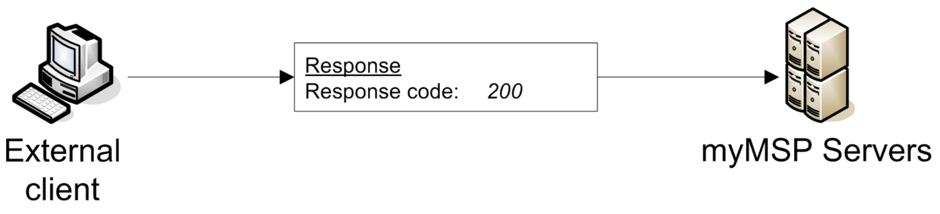
3.1 Receive delivery info

It can take some time for a message to be sent and delivered to all recipients. When myMSP has finished sending a message a *delivery info-message* is sent to the external client. A delivery info-message contains the same parameters as the Web service class *SendResultResponse* (see 4.4.2 in the Web service appendix document), with the exception of *msgStatus*, *recipientResults*, *deliveredOkCount* and *readOkCount* which are not included since they are redundant. A new parameter, *smsCount*, is added.

1. myMSP sends a delivery status report containing the message number, the send status which will be either 1100 (completed) or 1200 (failed), and some aggregated information regarding recipients.



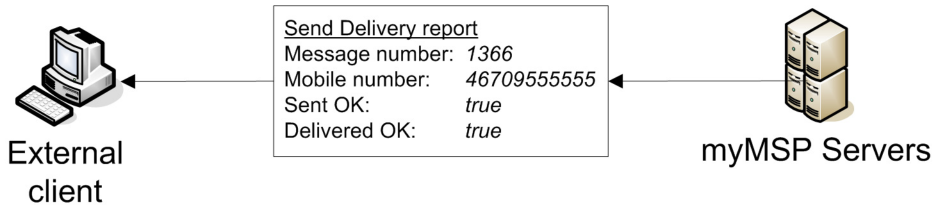
2. The client application responds with HTTP response code 200.



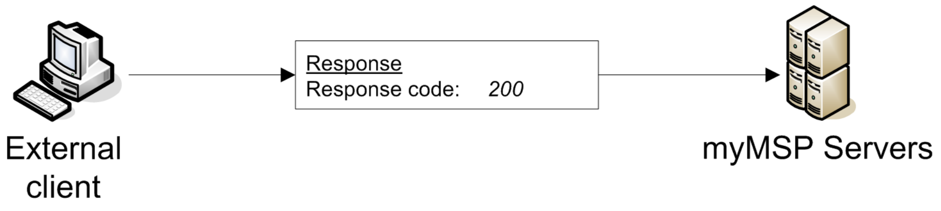
3.2 Receive delivery report

After the delivery info-message has been successfully sent to the client myMSP will begin sending *delivery reports* for each recipient of the message. A delivery report will be sent each time the delivery status has been updated, e.g. if the message has been delivered to a recipient or if it has expired. A delivery report corresponds with the *RecipientResult* and *Recipient* classes in the Web service interface (see 4.3.2 and 4.4.3. in the Web service appendix document).

1. myMSP sends a delivery report containing a message number, recipient information, and the delivery status of the message for the current recipient.



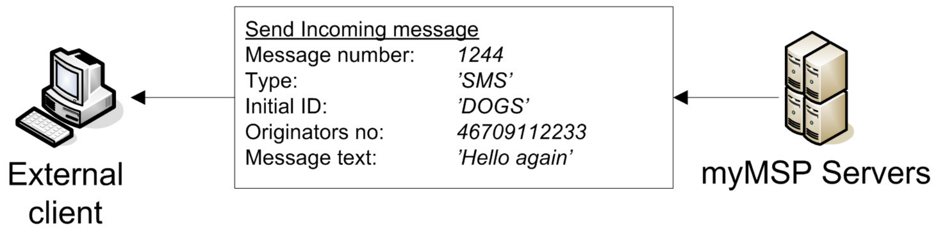
2. The client application responds with HTTP response code 200. The message number is used to link the delivery report with a previously sent message.



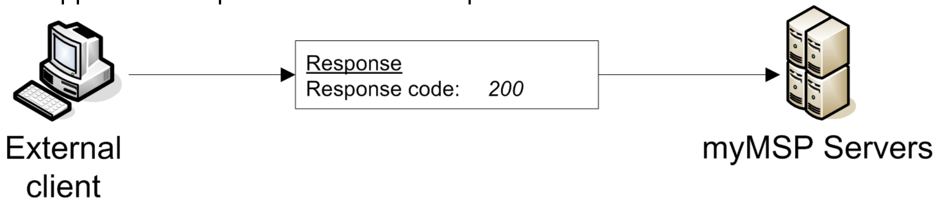
3.3 Receive an incoming message

myMSP can receive SMS sent from mobile phones and send these messages to a client application. An *incoming message* corresponds with the *MessageReceived* class of the Web Service interface (see 4.6.3 in the Web service appendix document).

1. myMSP sends an incoming message containing the message number, In-ID, text etc. to the client.



2. The client application responds with HTTP response code 200.

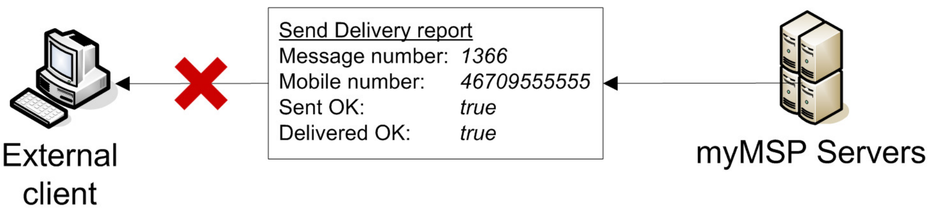


Document name:	myMSP_API_v2_Appendix_http_push.pdf
Revision:	C

3.4 Ping

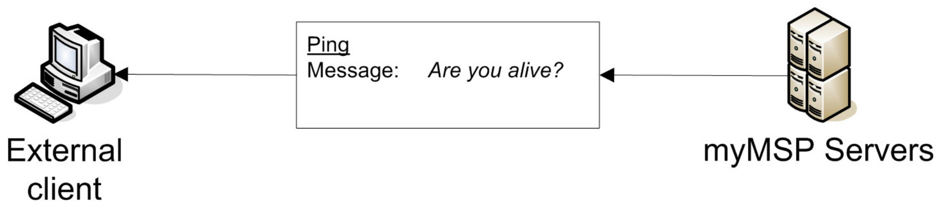
If myMSP does not receive any responses after sending several messages to a client application, that client’s push queue will be stopped. myMSP will start sending *ping messages* to the client until it receives a positive response.

1. myMSP puts a client’s push queue on hold after 10 failed attempts to send messages.

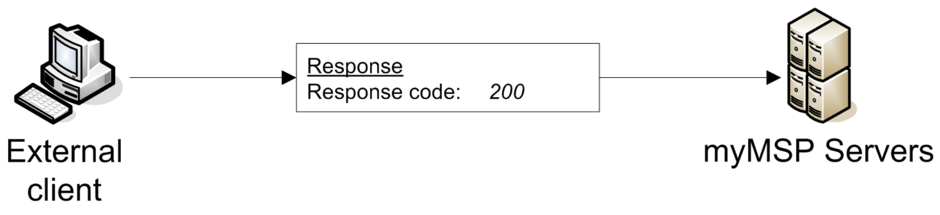


Send Delivery report
Message number: *1366*
Mobile number: *46709555555*
Sent OK: *true*
Delivered OK: *true*

2. myMSP sends a ping message every 20 seconds until it receives a positive response from the client.

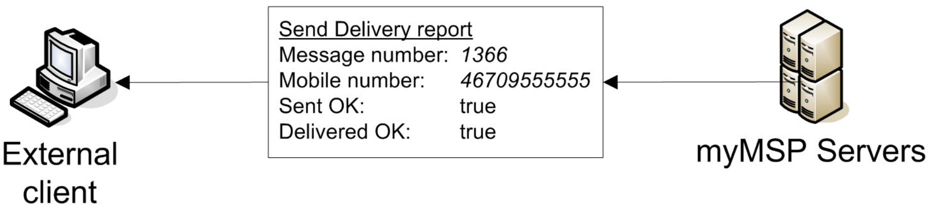


Ping
Message: *Are you alive?*



Response
Response code: *200*

3. The push queue is re-started.

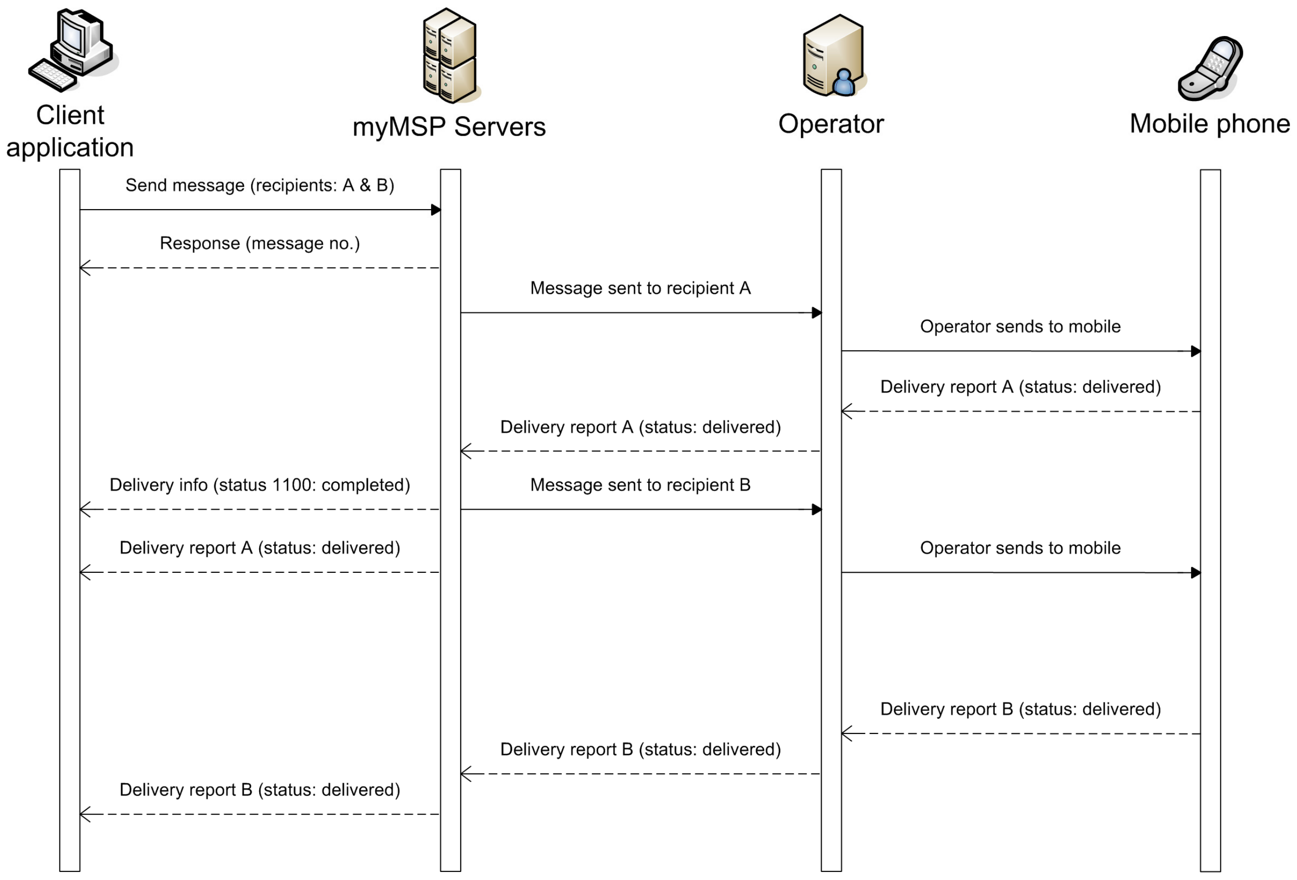


Send Delivery report
Message number: *1366*
Mobile number: *46709555555*
Sent OK: true
Delivered OK: true

4 Transmission sequence scenario

The figure below illustrates a typical transmission sequence where a client application uses an API (e.g. HTTPS or Web service) to send a message to myMSP and receive the send status through the push interface.

A client application connects to myMSP through an API and transmits a *send message*-request containing a message and a list of two recipients. myMSP receives the message and returns a *response* that includes the message number that the new message has been given. myMSP sends the message to the recipients through their respective mobile phone-operators, beginning with recipient A. The operator sends the message to recipient A's phone and a *delivery report* (A) is returned to myMSP. myMSP does not send the delivery report to the client application yet since the message has not yet been sent to all recipients. When the message has been sent to recipient B it is considered finished (message status: 1100) and a *delivery info-message* is sent to the client. The delivery info-message also specifies that the number of successfully sent messages is two, i.e. that it has been successfully sent to all recipients. The client uses the message no. in the delivery info-message to match the report to a previously sent message. Delivery report A is immediately sent by myMSP once the delivery info-message is accepted by the client. A little while later recipient B switches on his phone and delivery report B is sent from the operator to myMSP and finally to the client application.


Client application
myMSP Servers
Operator
Mobile phone
Send message (recipients: A & B)
Response (message no.)
Message sent to recipient A
Operator sends to mobile
Delivery report A (status: delivered)
Delivery report A (status: delivered)
Delivery info (status 1100: completed)
Message sent to recipient B
Delivery report A (status: delivered)
Operator sends to mobile
Delivery report B (status: delivered)
Delivery report B (status: delivered)
Delivery report B (status: delivered)

5 HTTP push methods

This chapter describes the various messages that the myMSP push interface can send in more detail.

The parameter *messageType* is included in every message and indicates which type of message is being sent. It can have the following values:

0	Ping
1	Deliver info
2	Delivery report
3	Incoming message

If any optional parameters have been specified then they are also included in every message.

5.1 Ping

A ping is sent to check if the client application is responding.

messageType	0
pingMessage	Are you alive?

Example:
http://www.mysite.com/listener?messageType=0&pingMessage=Are%20you%20alive%3F

5.2 Delivery info

When a message has been sent to all its' recipients a delivery info-message is sent.

messageType	1
msgNo	Message identification number in myMSP.
sendRequestTime	The time (yyyyMMddHHmmssSSS) the message was requested to be sent.
recipientCount	The number of recipients of the message.
smsCount	If the message text of a SMS is longer then 160 characters then the message is split in to several messages that are linked together. The SMS count indicates how many messages in total where sent (2 recipients + 2 parts = SMS count 4). For normal SMS the SMS count and the recipient count will always be identical.
sentOkCount	The number of messages that where successfully sent to the operator. If all messages have been sent then the sent ok count and the SMS count will have the same value.

Example:
http://www.mysite.com/listener?messageType=1&msgNo=1234567&sendRequestTime=20090101153422000&recipientCount=2&smsCount=4&sentOkCount=4

5.3 Delivery report

After the delivery info-message has been successfully sent to the client myMSP will begin sending delivery reports for each recipient of the message.

messageType	2
msgNo	Message identification number in myMSP.
recipientId	The myMSP internal ID of the recipient.
recipientName	The name of the recipient.
mobileNumber	The mobile number of the recipient.
externalRef	An external reference for the recipient in myMSP.
operatorResultCode	The result code (error code) from the operator. Provided as guidance for an eventual correction of the message.
operatorResultDescription	The result description (error text) from the operator. Provided as guidance for an eventual correction of the message.
sentOk	Indicates if the operator has accepted the message and the recipient.
sentTime	The time (yyyyMMddHHmmssSSS) the message was accepted by the operator.
deliveredOk	Indicates if the operator has successfully delivered the message to the recipient.
deliveredTime	The time (yyyyMMddHHmmssSSS) the message was delivered to the recipient (mobile phone).
readOk	Indicates if the message is opened (read) by the recipient. Not always available, is dependent on operator (not all operators support the feature) and message type.
readTime	The time (yyyyMMddHHmmssSSS) the message was opened (read) by the recipient. Not always available, is dependent on operator (not all operators support the feature) and message type.

Example:
http://www.mysite.com/listener?messageType=2&msgNo=1234567&recipientId=99991&recipientName=John%20Doe&mobileNumber=46709111111&externalRef=1000&operatorResultCode=0&operatorResultDescription=&sentOk=true&sentTime=20090101153425000&deliveredOk=true&deliveredTime=20090101153455000&readOk=false&readTime=

5.4 Incoming message

When myMSP receives a message sent from mobile phone it forwards the information to the client application.

messageType	3
msgNo	Message identification number in myMSP.
createTime	Time and date when the message arrived in myMSP.
creatorName	Name of the myMSP user that received the message.
initialId	The initial Id used to identify the recipient of the incoming message.

msgType	Type of message (1=SMS, 2=DPN, 3=VOI).
originator	The originator's mobile number.
originatorText	Originator text of the message.
destination	The phone number that the message was sent to.
smsText	The SMS text.
subject	[deprecated]
externalRef	An external reference for the message.
isPremium	True if the message is a premium message.

Example:
http://www.mysite.com/listener?messageType=3&msgNo=1234568&createTime=2009%2D01%2D01%2010%3A56&creatorName=aUsername&initialId=anInID&msgType=1&originator=46709111111&originatorText=&smsText=A%20test%20message&subject=&externalRef=&isPremium=false