myMSP API v2

System Interface - Specifications

21st Century Mobile
Marketing & Sales
+46 (0)8 21 21 55
sales@21st.se

www.21st.se
Research & Development
+46 (0)8 21 21 55
tech@21st.se

1 Purpose of Document

The purpose of this document is to define the API interfaces made available by myMSP to access the application functionality. Accessing myMSP can be done from the computer systems of external parties/customers. This document has several appendixes which describe each of the available API:s in more detail.

1.1 Author

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

1.2 Document Status

- Published 2007-01-08

1.2.1 Document Changes

- Since API v1
  - Separated the API document into several separate documents:
    - myMSP API v2 (an overview of the myMSP APIs)
    - myMSP API v2 Appendix ws (describes the Web Service API)
    - myMSP API v2 Appendix http (describes the HTTP API)
    - myMSP API v2 Appendix xml (describes the XML API)
  - Expanded the Web Service API document greatly
- B [2007-06-15]
  - Minor changes and improvements
- C [2007-08-20]
  - Added a note that the maximum length of a sender alias is ten characters.
- D [2007-12-14]
  - Updated the headers and footers with new information.
  - Updated the result codes in chapter 3.3
  - Added a note that the maximum length of a sender alias is eleven characters.
- E [2008-03-28]
  - Updated all URL-references from the old domain (www.mymsp.eu) to the new domain (mymsp.21st.se).
- F [2009-04-29]
  - Added a short description of the SMPP-API to 4.4
- G [2009-12-09]
  - Added a description of the push API to 3.1, 3.4, 3.5 and 4.5.
- H [2011-08-24]
  - Added a new result code: 503 (3.3)
- I [2012-10-26]
  - Added information regarding Device Push Notifications (DPN) to the document.
- J [2019-03-11]
  - Updated graphic profile and URL:s.
- K [2020-04-14]
  - Added information regarding URL shortener to the document.

1.3 Document Perimeter

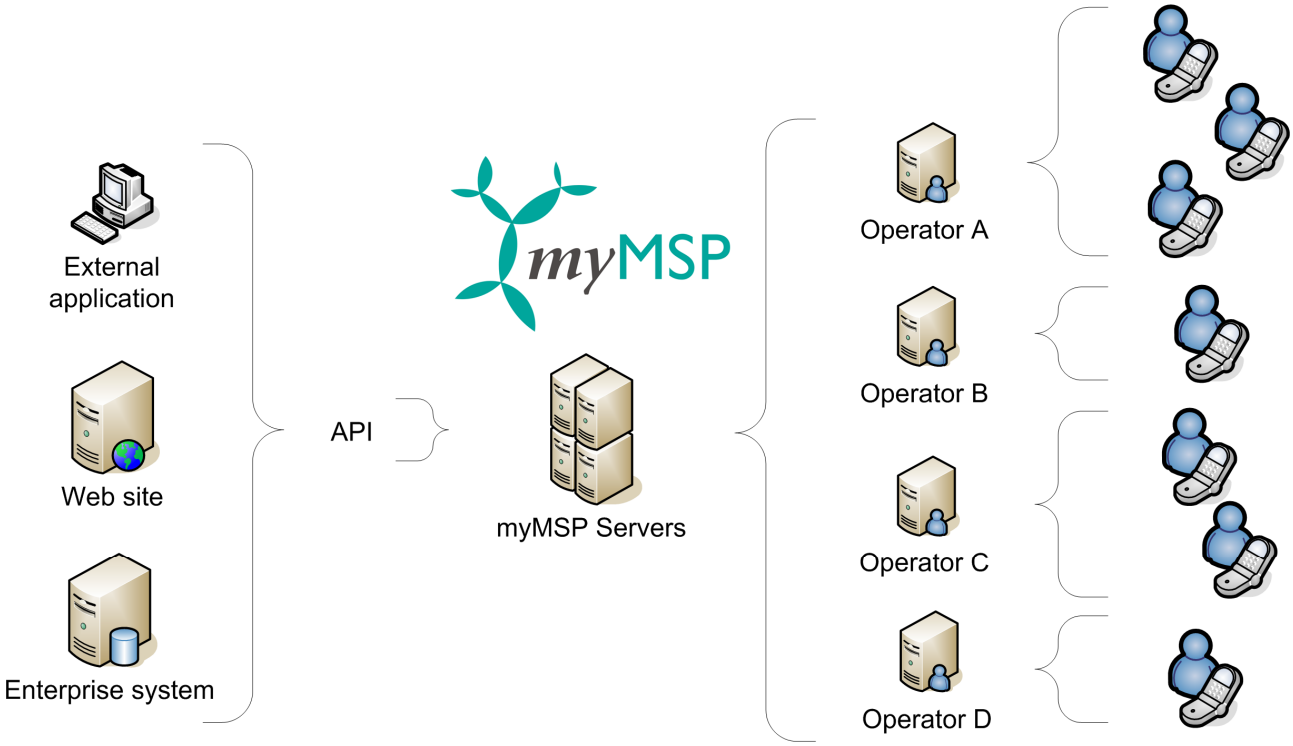
Neither economical questions nor questions regarding agreements/contracts will be dealt with in this document.

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 Overview

The myMSP application is a platform for structured mobile message management that enables communication between multiple mobile phones and different types of computer systems, using both two-way SMS, voice messages (VOI) and Device Push Notification (DPN) communication. myMSP is directly connected to the major domestic mobile operators that handle the actual sending and receiving of the mobile messages over the phone net.

Anyone using myMSP can easily connect their own computer systems, e.g. an ERP or CRM system, to myMSP through several Application Programming Interfaces (API).



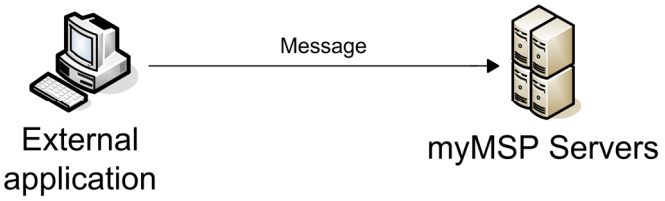
3 Communication with myMSP using an API

When accessing myMSP through an API it is important to have a basic understanding of how myMSP operates. The three central functions of myMSP are: *send messages*, *check the status of a sent message* and *retrieve incoming messages*. Different API:s can access these services in different ways. However, all API:s have a core set of concepts in common which are described here.

3.1 The send message sequence

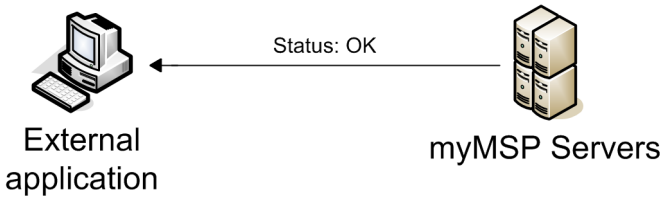
The time it takes for a message to be sent by the operator can be considerable and therefore the transmission sequence is a two-step process:

1. The message, with its intended recipients, is sent to myMSP. If the message is accepted (i.e. it is correctly formatted) then it is assigned a message number by myMSP and placed in a send queue.

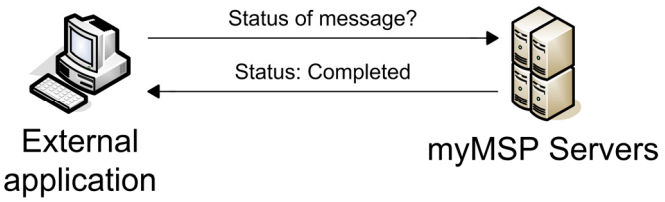


2. The send status of a message can be obtained by the external application in two ways:

    a. myMSP transmits the send status of a message to the external application. Updates are only sent when the send status has changed (e.g. a message has been delivered).



    b. The external application sends a request to myMSP querying the send status of a message. A response is returned containing the current send status of the message (e.g. delivered). This step can be repeated until it has been confirmed that the message has been sent to all recipients.



3.2 Message status

A message has a status (state) indicating the send progress of a message:

- 0 Created (myMSP has received the send request and accepted it)
- 1000 Sending (the message is being sent)
- 1100 Completed (the message is no longer being sent)
- 1200 Failed (the message could not be sent)

It is important to realize that the message may be sent to the operator in fragments, and that the acceptance is received per recipient (mobile number). A message with multiple recipients will not always be delivered to all recipients (for instance if a recipient's mobile number is invalid). When a message is no longer being transmitted to its recipients it will achieve a message status of 1100 (Completed), even if not all recipients have received the message. Information regarding the number of messages that have been successfully sent to recipients, delivered, and read is reported by myMSP. The message status should not be confused with result codes described below.

3.3 Result codes

The result code indicates the success of invoking a myMSP service. Example: a send request is sent to myMSP and the result code 0 is returned indicating that myMSP has accepted the request. If the result code is not 0 then something is wrong, e.g. the password is invalid (code 202). Note that the HTTP-API uses a slightly different method of indicating the success of a request, and that different codes apply (these are described in the HTTP appendix document).

The result codes are:

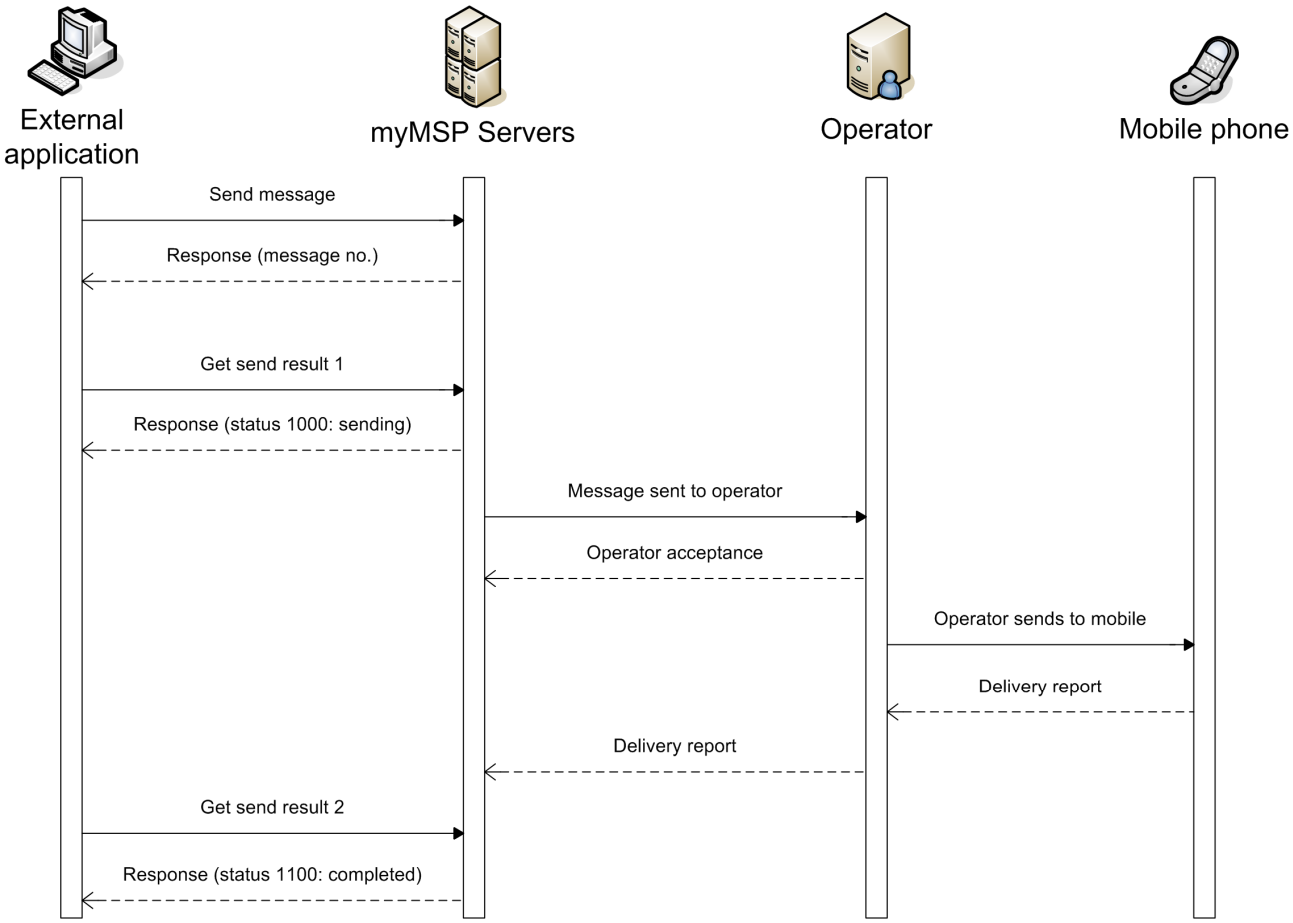
- 0 OK
- 100 User not owner of message!
- 101 User information missing!
- 102 Unknown message type
- 103 Recipient information missing!
- 104 Missing message text!
- 105 Message text longer than 765 characters!
- 106 Missing message data!
- 107 Message subject longer than max 40 characters!
- 108 Wrong message format!
- 109 Invalid time format (expected yyyyMMddHHmmssSSS)!
- 110 Invalid time (expected value later then 2004-01-01 00:00:00)!
- 201 Failed authorization, missing user
- 202 Failed authorization, bad password for user
- 211 Failed client validation, missing client type!
- 212 Failed client validation, invalid client type
- 213 Failed client validation, expected client version string, found
- 221 Failed client validation, missing client version!
- 222 Client validation, a newer version is available!
- 223 Failed client validation, not allowed version
- 301 Failed adding recipients to message!
- 401 Failed adding content to message!
- 501 Failed saving message!
- 502 Limit maximum messages exceeded!
- 503 Not enough credits left to send message!
- 601 Unknown message number!
- 701 Unknown request!

- 901 Unknown myMSP error, contact support!
- 902 Unknown error, contact support!

3.4 Send message example

The figure below illustrates a typical transmission sequence using the querying method for obtaining the send status (3.1 2a). For an example on how the sequence changes when delivery reports are pushed by myMSP to the external application see the push appendix document.

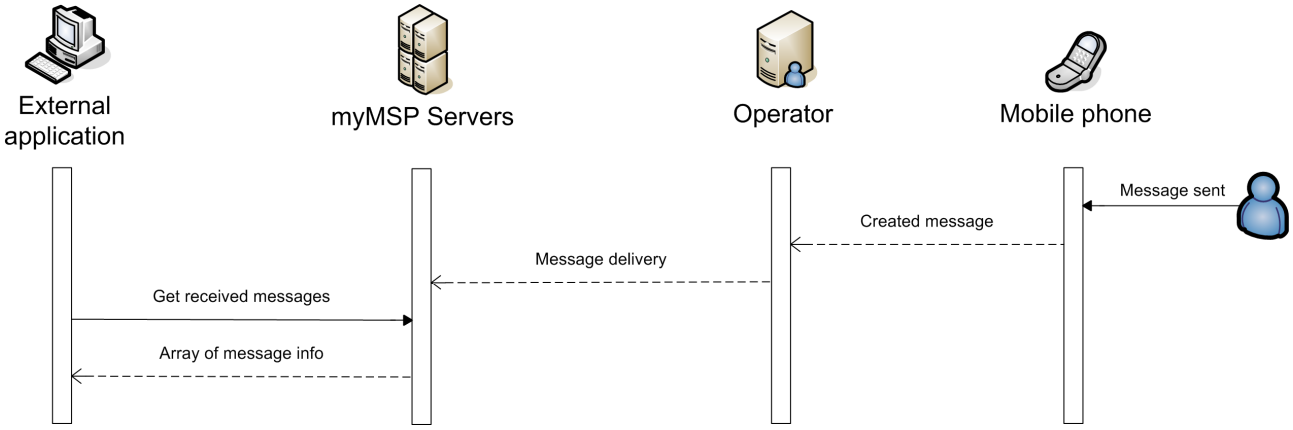
An external application connects to myMSP through an API and transmits a *send message*-request containing a message and a list of recipients. A myMSP server receives the message and returns a *response* that includes the message number that the new message has been given. The external application then checks the status of the message by sending the message number to the server in a *send result*-request, and receives a response from the server in the form of a status number. The status number 1000 indicates that myMSP has placed the message in the send queue. myMSP sends the message to the recipients through their respective mobile phone-operators. When an operator has accepted a message to a recipient's number an *operator acceptance*-response is sent to myMSP. The operator then sends the message to the recipient's phone and a *delivery report* to myMSP. When the message has been sent to all operators it is considered completed and receives a status of 1100. This status, along with the delivery status of each recipient, is returned to the external application when it transmits a second send result-request to myMSP.



3.5 Retrieve message example

The figure below illustrates an incoming message sequence using the querying method for obtaining the send status. For an example on how the sequence changes when incoming messages are pushed by myMSP to the external application see the push appendix document.

The figure below illustrates an incoming message sequence. A mobile phone user that received the message from the previous example decides to send a reply. The operator receives the message and sends it to myMSP. In the text-body of the message the user has included an incoming ID (In-ID) which makes it possible for myMSP to direct the message to the right myMSP-account(s). When the external application send a *retrieve received messages*-request the mobile phone-user's message is returned to the application along with any other messages that have been redirected to the myMSP user's account.



4 Available APIs

The following APIs are currently available. Note that the APIs are constantly being improved and new functionality added in accordance with customer needs. Please contact us (tech@21st.se) if you find that the functionality of an API is lacking or if you would like to access myMSP through an API that is not listed here, and we will do our up most to accommodate you.

Appendix documents with more detailed descriptions of the APIs listed below can be downloaded from our file library which you can find here: https://assets.21st.se/docs/API_v2/

4.1 HTTPS

An external system can connect to myMSP through HTTPS by sending a series of parameters using the POST method (POST is a HTTPS-method and is not related to e-mail).

Currently supports:

- Send SMS
- Send VOI (voice messages)
- Check send status of a message
- Retrieve incoming SMS

4.2 Web service

Web Services are application services or functions that are made available through the Internet. The information exchange is carried out using XML-documents that are sent through the HTTP-protocol and WSDL (Web Services Description Language) is used to describe what services are available and how they are accessed. WSDL is constructed to enable an automatic information and service exchange between computers.

Currently supports:

- Send SMS
- Check send status of a message
- Retrieve incoming SMS

4.3 SMPP

The Short Message Peer-to-Peer (SMPP) protocol is the most common protocol for sending SMS messages in use today. myMSP supports SMPP V3.4.

Currently supports:

- Send SMS
- Receive send status of a message
- Receive incoming SMS

4.4 HTTP push (for delivery reports and incoming messages)

myMSP can transmit to an external system through HTTP or HTTPS using the GET method to submit the data. The push interface is used to send data, like incoming messages and delivery status updates, directly to an external system thus eliminating the need for that system to constantly poll myMSP for updates. 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. However, it is also possible to use the push interface in conjunction with the HTTPS-API:s, but not with the SMPP interface.

Currently supports:

- Receive send status of a message
- Receive incoming SMS

4.5 Device Push Notifications (DPN)

Device push notifications (DPN) make it possible to send lightweight messages through the Internet to apps running on, primarily, mobile devices such as smart phones and tablet computers. What sets notifications apart from other forms of mobile communication methods such as SMS and e-mail is that they can be targeted at an individual app running on a specific device. Notifications can be used to send simple text messages or to inform the app that there is new data to be fetched from an external server. myMSP's notification technology enables developers to send notifications to apps through a HTTP API without having to know what operating system the app is running on.

Currently supports:

- Add/remove subscriptions (app)
- Send notifications (external server)

4.6 URL shortener

The URL shortener API is used to create short URLs that forwards all request to longer URLs. This is an efficient way to reduce the length of an SMS message. myMSP uses a default domain name when creating short URLS but custom domains can also be configured. Contact you 21st sales representative to activate this feature for your account.

5 Glossary

Content	The information to be communicated. For message type SMS the content is text.
DPN	A DPN (Device Push Notification) is a lightweight message sent over the Internet to an app running on a device (e.g. a smart phone).
In-ID	When a mobile phone user wants to send a message to myMSP an incoming ID (In-ID) is required. The ID enables myMSP to direct the incoming message to the correct user account.
Message	An entity (container) with some content and an originator who wishes to communicate with one or more recipients.
Message type	An indicator which restricts what content a message may contain. Examples are SMS, VOI and DPN.
Mobile phone number	Mobile Phone Numbers can be represented in various formats. For example, when dealing with a national Swedish number, like 0730430525, the following formats are valid: 0046730430525, +46730430525 and 46730430525.
myMSP	The Mobile Solutions Platform provided by 21st Century Mobile.
Sender Alias	Se Originator Text below.
Operator	The operator, or mobile operator, is the organization responsible for the actual transmission of messages from myMSP to recipients' mobile phones and from mobile phones to myMSP, utilizing their own telephone network. Examples are Tele2, TeliaSonera, Telenor and Tre.
Originator	An originator is the sender of a message, normally a mobile number or a short number.
Originator Text	When a message is received on a mobile phone the originators number is displayed. This number can be replaced by an alias (originator text). For example: 46778899 can be displayed as 'companyx' or another alias. However, recipients can not reply to a message that uses an alias. Only aliases that have been submitted, accepted and registered in myMSP can be used. This functionality is however operator dependent since not all of them supports originator text. The maximum length of an alias is eleven (11) characters.
Recipient	A recipient is the target of the message identified either by recipient id, recipient name or a mobile number. The name may have been created automatically from the mobile number by myMSP.
SMS	A SMS (Short Message Service) can only have content of type text.
VOI	Voice message. The message is delivered by telephone call and the text is read to the receiver.