SMS GATEWAY

HTTP Interface Guide

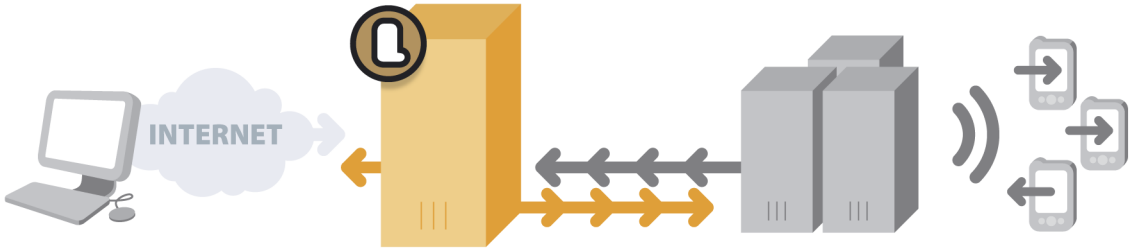
VERSION HISTORY

Version	Date	Major changes
2.0	Aug 16, 2010	• New SMS Gateway IP addresses. Please allow them all in your firewall. [5.1] • Added *message-count* and *error-code* to send response. [2.2] • Added status *code* to delivery report. [4.1] • New parameters *wap-url* and *wap-text* for sending WAP Push messages. [2.1, 3.2] • New send parameter *dests* for specifying recipient numbers separated with commas. [2.1]
2.1	Aug 22, 2011	• New parameter *timestamp* in HTTP requests made by SMS Gateway. [3.1, 4.1] • Support for SMS Gateway availability monitoring. [6]

CONTENTS

1. Introduction....4
1.1 Glossary....5
2. Sending SMS Messages....6
2.1 Parameters from Application to SMS Gateway....6
2.2 Response from SMS Gateway....8
2.3 HTTP Example....9
3. SMS Services....10
3.1 Parameters from SMS Gateway to Application....10
3.2 Response from Application....11
3.3 Retry Logic....12
3.4 HTTP Example....13
4. Delivery Status Reports....14
4.1 Parameters from SMS Gateway to Application....14
4.2 Response from Application....15
4.3 Retry Logic....15
4.4 HTTP Example....16
5. Security....17
5.1 IP and User Restriction on Customer's Server....17
5.2 IP and User Restriction on SMS Gateway....17
5.3 Encryption....18
6. Monitoring....19
6.1 Parameters from Application to SMS Gateway....19
6.2 Response from SMS Gateway....19
6.3 HTTP Example....20
7. Examples....21
7.1 PHP: Sending SMS Message....21
7.2 PHP: SMS Service....21
7.3 Java: Sending SMS Message....22
7.4 Java: SMS Service....23

1. INTRODUCTION



Customer / Labyrintti Media [SMS Gateway] / Operators / Mobile phones
Figure 1. Role of SMS Gateway

Labyrintti Media SMS Gateway provides application developers a simple interface for developing SMS services and sending SMS messages, thus allowing developers to concentrate on the core application. Applications communicate with SMS Gateway using the standard HTTP protocol. This allows for applications to be written in almost any programming language, like C/C++, Java, PHP, ASP.net, Perl, etc. Usually service applications work on top of a web server software that already contains the required HTTP and CGI functionality.

This document describes the various parameters available in the SMS Gateway HTTP interface and provides simple examples for some common programming languages. Labyrintti Media also provides customers with PHP and Java libraries for sending and receiving SMS messages.

1.1 GLOSSARY

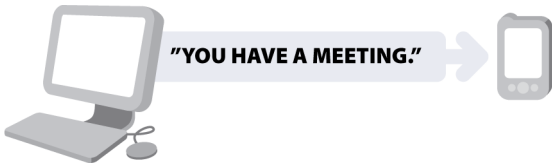
Concatenation
The maximum length of an SMS message is 160 characters (or 140 bytes). However, modern mobile phones allow writing longer messages that consist of multiple SMS messages which are concatenated (joined) back to one long message in the receiving phone. Each of the individual SMS messages that form the long message must have a special user data header. SMS Gateway allows both sending and receiving long messages and handles the required headers automatically, also for binary messages.

Keyword
If more than one SMS service uses the same service phone number, each service needs one or more keywords that identify the service. When the beginning of an incoming SMS message matches the keywords of some SMS service, SMS Gateway forwards the message to that service. Keywords are case-insensitive: “ORDER”, “Order” and “order” all match the same SMS service.

Unicode
Unicode is a universal character encoding that supports most of the world’s languages. SMS Gateway optionally supports the SMS UTF-16 encoding which uses two bytes per character when sending or receiving SMS messages.

User data header
SMS messages can contain an optional data header which includes binary data specific to a certain protocol/service. For example, header is required when sending WAP Push messages, electronic business cards and calendar events. Header is also used with message concatenation but SMS Gateway normally handles all concatenation headers automatically.

WAP Push
Many phones support WAP Push messages which are special binary SMS messages that contain a link URL and an optional description text. A supporting phone allows the recipient to easily open the link in the phone browser. WAP Push is typically used to deliver links to mobile content. SMS Gateway supports easy sending of WAP Push messages but also allows sending low-level binary SMS messages.



Customer's server / Mobile phone
Figure 2. Sending SMS

2. SENDING SMS MESSAGES

To send messages to mobile phones without first receiving a request SMS from the phone, a service application connects to internal HTTP server of SMS Gateway and sends a HTTP request (acting like a web browser). SMS Gateway accepts messages at URL

http://gw.labyrintti.com:28080/sendsms

If you have purchased the option to use secure sending, you can use the URL

https://gw.labyrintti.com:28443/sendsms

Messages can be sent using either the HTTP **GET** method,
or **POST** with **Content-Type application/x-www-form-urlencoded**.

2.1 PARAMETERS FROM APPLICATION TO SMS GATEWAY

Parameters that the application must specify in the request:

Parameter	Description	Supported values or example
user	Username that identifies message sender. Alternatively, HTTP Basic authentication can be used.	customer
password	Password for the user.	3fGDy6bp
dests	One or more destination phone numbers, separated with commas (,). Numbers can be in national or international format and include separators like spaces, hyphens and parentheses.	+358401234567, (050)-765 4321

Also, exactly one of the following parameters must be specified. If content does not fit into one SMS message, SMS Gateway can split it into multiple messages, based on user account configuration (see *concatenate*).

Parameter	Description	Supported values or example
text	Message in plain text. **\n** is replaced with a line feed, **\r** with a carriage return, and **** with a backslash, although they can be specified using URL encoding also. Normally, \n is enough for SMS newline.	Hello World
binary	Message coded as binary hexadecimal ASCII string. This will send a binary SMS message.	5468616E6B20796F7521
wap-url	Link URL for sending WAP Push SI message.	http://www.example.com/

Optional parameters:

Parameter	Description	Supported values or example
source-name	Name or phone number displayed instead of service number in receivers' mobile phones. Default is set in user account configuration (initially unspecified).	Company, 0401234567 (max 11 characters for names, max 16 for phone numbers)
source	Service number used to send the message. You do not need this unless you have access to multiple service numbers. Default is set in user account configuration.	16130
service	Name (keywords) of service that sent the message. This associates the message to correct service in your message statistics.	ORDER
header	Message data header, coded as binary hexadecimal ASCII string.	060504158A0000
wap-text	Message text for WAP Push SI message, when using the *wap-url* parameter.	Open the link to download the content
class	Message class. Flash messages are displayed instantly on receivers' mobile phones. Default is set in user account configuration (initially *normal*).	normal, flash
concatenate	If enabled, SMS Gateway adds a special data header that causes multiple response SMS messages to be shown as one long message on modern mobile phones. Default is set in user account configuration (initially *yes*).	yes, no
unicode	If enabled, message will be sent to phones using UTF-16 encoding. This allows only 70 characters per message. Default is *no*.	yes, no
validity	Message validity period. Either a minute value or absolute time in format **yyyy-mm-dd hh:mm**. If a message has not been received by mobile phone when its validity expires, SMS center deletes the message. Default is set in user account configuration (initially *1440* minutes).	1440, 2011-12-31 23:59
delivery	Delayed delivery of message. Delivery time given either as a minute value or absolute time in format **yyyy-mm-dd hh:mm**.	60, 2011-12-31 23:59
report	URL that receives delivery status reports for the message. See *4. Delivery Status Reports* for more information.	http://www.example.com/sms-report.php?msgid=1

2.2 RESPONSE FROM SMS GATEWAY

SMS Gateway responds with a text/plain HTTP 200 OK response that contains message sending status for every destination phone number. The response has one line per destination, and on each line there is the destination phone number in **international format** and status of sending.

For messages which have been accepted for sending, the response format is:

phone-number OK message-count description

For example:

+358401234567 OK 1 message accepted for sending

For messages which have been denied, the response format is:

phone-number ERROR error-code message-count description

For example:

12345 ERROR 2 1 message failed: Too short phone number

Current error codes:

Code	Description
1	**Unknown error**: Error whose reason is not known or specified.
2	**Invalid recipient**: Recipient phone number syntax is invalid. For example, too short or long.
3	**Duplicate recipient**: The same recipient phone number has been specified multiple times. Only one message will be sent to each phone number.
4	**Unallowed recipient**: Recipient phone number is not allowed in user account configuration. For example, foreign recipient phone numbers can be denied.
5	**Routing error**: There is no operator route that supports sending to the recipient phone number.

If the HTTP request parameters are invalid, username or password wrong, client IP address unallowed for user account, protocol (HTTP or HTTPS) unallowed, or SMS sending unallowed, SMS Gateway responds with a text/plain error message and HTTP error status code.

2.3 HTTP EXAMPLE

This example uses HTTP POST but GET could be used as well.

2.3.1 Request (application -> SMS Gateway)

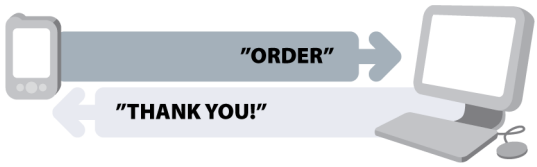
```
POST /sendsms HTTP/1.1
Host: gw.labyrintti.com:28080
Content-Type: application/x-www-form-urlencoded
Content-Length: 83

user=customer&password=3fGDy6bp&dests=%2B358401234567%2C0401234567&text=Hello+World
```

2.3.2 Response (SMS Gateway -> application)

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=ISO-8859-1
Content-Length: 125

+358401234567 OK 1 message accepted for sending
+358401234567 ERROR 3 1 message failed: Duplicate destination phone number
```



Mobile phone / SMS service at customer's server
Figure 3. SMS service

3. SMS SERVICES

An SMS message received from a mobile phone is passed to a service application in the form of a HTTP request (like a web browser asking a web server for a page).
Response SMS message is then generated from the HTTP response returned by the service application (which is acting like a web server).
SMS Gateway chooses the correct service application based on the first words of the SMS message, called keywords.

SMS Gateway uses either the HTTP **GET** method, or **POST** with **Content-Type application/x-www-form-urlencoded**, depending on service configuration. POST is the default.

Usually a response message must be sent, otherwise the mobile phone user will not be billed. Response message can be left out only if a separate service number with a fixed price is used.

3.1 PARAMETERS FROM SMS GATEWAY TO APPLICATION

When generating the HTTP request, SMS Gateway adds the following parameters:

Parameter	Description	Supported values or example
source	Phone number of message sender in international format.	+358401234567
operator	Name of message sender's operator. Operator name starts with a two-letter country code.	FI DNA, FI Elisa, FI Saunalahti, FI Sonera
dest	The service number that received the message.	16130
keyword	One or more words from the beginning of the message. SMS Gateway identified the service based on these words. Always in UPPER case.	ORDER
params	The rest of the message after keyword, words separated with single spaces.	sample magazine
text	The entire message text. This can be longer than 160 characters if the message consists of multiple concatenated SMS messages.	order sample magazine
timestamp	Date and time of message arrival at operator SMS center in format **yyyy-mm-dd hh:mm:ss**.	2011-12-31 23:59:59

SMS Gateway adds the following optional parameters only if they are available in the received message:

Parameter	Description	Supported values or example
header	If message contains user data header, this is added as binary hexadecimal ASCII string.	060504158A0000
binary	If message data is binary, it is added as binary hexadecimal ASCII string.	5468616E6B20796F7521

Note: New parameters can be added to future releases of the SMS Gateway HTTP interface. Your application should work even if the HTTP request made by SMS Gateway includes parameters not listed in this guide. Your application should just ignore any extra parameters.

3.2 RESPONSE FROM APPLICATION

When application sends the HTTP response, it must use **Content-Type text/plain** and respond with HTTP status code **200 OK**. Response content consists of parameter lines in the following format:

```
parameter1=value1
parameter2=value2
```

The only parameter that the application must specify is the response data content, given using one of the following parameters. If the response content does not fit into one SMS message, SMS Gateway can split it into multiple messages, based on service configuration (see *concatenate*).

Parameter	Description	Supported values or example
text	Response message in plain text. **\n** is replaced with a line feed, **\r** with a carriage return, and **** with a backslash. Normally, \n is enough for SMS newline.	Thank you for your order!
binary	Response message coded as binary hexadecimal ASCII string. This will send a binary SMS message.	5468616E6B20796F7521
wap-url	Link URL for sending WAP Push SI message.	http://www.example.com/

Other **optional** response parameters:

Parameter	Description	Supported values or example
error	If enabled, indicates that either the user made a mistake or the service failed, and the response contains an error message to be sent to the end-user. This will increase the number of failed transactions in service statistics and may cancel the end-user billing. Default is *no*.	yes, no
header	Message data header, coded as binary hexadecimal ASCII string.	060504158A0000
wap-text	Message text for WAP Push SI message, when using the *wap-url* parameter.	Open the link to download the content
class	Message class. Flash message is displayed instantly on receiver's mobile phone. Default is set in service configuration (initially *normal*).	normal, flash

concatenate	If enabled, SMS Gateway adds a special data header that causes multiple response SMS messages to be shown as one long message on modern mobile phones. Default is set in service configuration (initially *yes*).	yes, no
unicode	If enabled, message will be sent to phone using UTF-16 encoding. This allows only 70 characters per message. Default is *no*.	yes, no
validity	Message validity period. Either a minute value or absolute time in format **yyyy-mm-dd hh:mm**. If message has not been received by mobile phone when its validity expires, SMS center deletes the message. Default is set in service configuration (initially *1440* minutes).	1440, 2011-12-31 23:59
delivery	Delayed delivery of message. Delivery time given either as a minute value or absolute time in format **yyyy-mm-dd hh:mm**.	60, 2011-12-31 23:59
report	URL that receives delivery status reports for the message. See *4. Delivery Status Reports* for more information.	http://www.example.com/sms-report.php?msgid=1
type	Specifies the type of the response message. MMS can be used only if MMS has been enabled for the service. See *MMS Gateway HTTP Interface Guide* for MMS response parameters.	SMS, MMS

3.3 RETRY LOGIC

If SMS Gateway cannot connect the customer's server, waiting for response times out, or customer's server responds with a HTTP error status code, Gateway will retry the HTTP request after a delay. All timeouts and retry parameters can be changed per service. The default parameters are connect timeout 60 seconds, response timeout 60 seconds, delay between retries 15 seconds, and maximum requests per received message 3. If all retries fail, the message will be discarded.

If you see three HTTP requests per received SMS message, check your service's response. Response HTTP status must be 200 OK and content-type text/plain.

3.4 HTTP EXAMPLE

This example uses HTTP POST but GET could be used as well.

3.4.1 Request (SMS Gateway -> application)

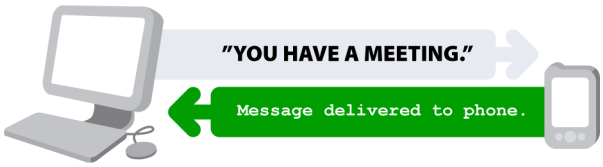
POST /sms.php HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-urlencoded; charset=ISO-8859-1
Content-Length: 151

source=%2B358401234567&operator=FI+Sonera&dest=16130&keyword=ORDER&params=sample+magazine
&text=order+sample+magazine&timestamp=2011-12-31+23%3A59%3A59

3.4.2 Response (application -> SMS Gateway)

HTTP/1.1 200 OK
Content-Type: text/plain
Content-Length: 45

text=Thank you for your order!
validity=60



Customer's server / Mobile phone
Figure 4. Delivery status report

4. DELIVERY STATUS REPORTS

If message delivery status reports are requested, they are sent to service applications whenever a change occurs in delivery state of message. For example, report could indicate that the message has reached the mobile phone, or that message validity period has expired before the mobile phone could be reached and the message is deleted. Reports can also be temporary, for example, message stored to SMS center for later delivery.

Message senders and services request delivery reports using the **report** parameter. This contains an URL that the SMS Gateway will contact whenever a delivery report is received. SMS Gateway uses HTTP **POST** with **Content-Type application/x-www-form-urlencoded** to send the request. The report URL can contain some service specific query parameters that help the service to identify the reported message.

4.1 PARAMETERS FROM SMS GATEWAY TO APPLICATION

SMS Gateway always adds the following parameters:

Parameter	Description	Supported values or example
source	Service number used to send the message.	16130
dest	One destination phone number in international format.	+358401234567
status	Delivery status. For one message/destination, there can be multiple delivery reports with the WAITING status but only one final OK or ERROR.	OK, ERROR, WAITING
code	Delivery status code. See delivery code table.	0
message	Text describing the status.	Message delivered
timestamp	Date and time of delivery report arrival at operator SMS center in format **yyyy-mm-dd hh:mm:ss**.	2011-12-31 23:59:59

Current delivery codes:

Code	Description
0	**OK**: Successfully delivered (status=OK), or buffered to SMS center (status=WAITING).
1	**Unknown error**: Error whose reason is not known or specified.
2	**Invalid recipient**: Recipient phone number is invalid or unknown.
3	**Unreachable recipient**: Recipient is temporarily unreachable, e.g. phone off or memory full.
4	**Barred recipient**: Recipient is out of credits or blacklisted.
5	**Subscription error**: Error related to recipient's service subscription.
6	**Expired**: Message validity period has expired.

7	**Routing**: No route to recipient or roaming not allowed.
8	**Network**: Problem with SMS center network connection.
9	**Capacity**: SMS center capacity temporarily exceeded.
10	**Operator**: General error related to SMS center operation.
11	**Protocol**: Error in message data/parameters or other protocol related error.
12	**Canceled**: Message sending has been canceled.

Note: New parameters can be added to future releases of the SMS Gateway HTTP interface. Your application should work even if the HTTP request made by SMS Gateway includes parameters not listed in this guide. Your application should just ignore any extra parameters.

4.2 RESPONSE FROM APPLICATION

When application responds to the HTTP request, it must respond with HTTP status code **200 OK**. The content of the response is ignored and can be left empty.

4.3 RETRY LOGIC

If SMS Gateway cannot connect the customer's server, waiting for response times out, or customer's server responds with a HTTP error status code, Gateway will retry the HTTP request after a delay. Gateway will retry each delivery report for many days before finally discarding the report. After each failed attempt, the delay waited before the next attempt will be doubled.

4.4 HTTP EXAMPLE

4.4.1 Message request (application -> SMS Gateway)

First, a message with a delivery report request is sent using SMS Gateway HTTP server. The report URL used here is http://www.example.com/sms-report.php?msgid=1

```
POST /sendsms HTTP/1.1
Host: gw.labyrintti.com:28080
Content-Type: application/x-www-form-urlencoded
Content-Length: 143

user=customer&password=3fGDy6bp&dests=%2B358401234567&text=You+have+a+meeting.
&report=http%3A%2F%2Fwww.example.com%2Fsms-report.php%26msgid%3D1
```

4.4.2 Message response (SMS Gateway -> application)

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=ISO-8859-1
Content-Length: 49

+358401234567 OK 1 message accepted for sending
```

4.4.3 Delivery request (SMS Gateway -> application)

There can be temporary delivery reports before the message has been delivered to recipient's phone. After the message has been delivered, SMS Gateway will send a final delivery status report:

```
POST /sms-report.php HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-urlencoded; charset=ISO-8859-1
Content-Length: 118

msgid=1&source=16130&dest=%2B358401234567&status=OK&code=0&message=Message+delivered
&timestamp=2011-12-31+23%3A59%3A59
```

4.4.4 Delivery response (application -> SMS Gateway)

```
HTTP/1.1 200 OK
Content-Length: 0
```

5. SECURITY

5.1 IP AND USER RESTRICTION ON CUSTOMER'S SERVER

Only SMS Gateway should have access to any SMS services at customer's servers. Otherwise advanced end-users can access the service using their web browsers. Limiting access is done differently on different web server software. Access should be allowed only to SMS Gateway's IP addresses and denied from all other IP addresses.

SMS Gateway IP addresses:

- **81.19.118.90**
- **109.204.225.190**

On Apache HTTP Server, limiting access to all files inside a directory can be done by creating a file called ***.htaccess*** in the directory and putting the following lines in the file:

```
Order Allow,Deny
Allow from 81.19.118.90
Allow from 109.204.225.190
```

SMS Gateway can also use HTTP Basic authentication when connecting customer's server. Each SMS service can have a distinct username and password specified by the customer.

Note that when sending messages, it is recommended to use the **gw.labyrintti.com** hostname instead of a Gateway IP address. If some Gateway server is under maintenance or experiencing network connectivity problems, gw.labyrintti.com will still resolve to a working server.

5.2 IP AND USER RESTRICTION ON SMS GATEWAY

In addition to the username and password, SMS Gateway only allows sending messages from network addresses specified by the customer. This username specific address restriction helps to prevent illegal sending. Allowed addresses can consist of IP addresses, subnets, IP ranges, hostnames and domain names.

When sending SMS messages using any kind of web interface, care should be taken to never show SMS Gateway's username and password to the web user. This should be kept in mind especially when handling error situations. For example, with PHP, *error_reporting(0)* or @ prefix can be used to avoid displaying name and password to the web user.

5.3 ENCRYPTION

Best security is achieved by purchasing the secure SMS Gateway option. This allows SMS message sending, services and delivery reports to use the encrypted **HTTPS** protocol. SMS Gateway's HTTPS server uses a certificate signed by a trusted certificate authority.

By default, SMS Gateway accepts self-signed customer server certificates but denies expired certificates. For even better security, services can be configured to deny self-signed certificates and/or non-matching certificate hostnames.

6. MONITORING

For organizations that want to monitor SMS Gateway availability using third-party network monitoring software, SMS Gateway provides a dedicated URL

http://gw.labyrintti.com:28080/monitor

For a secure connection, you can use the URL

https://gw.labyrintti.com:28443/monitor

6.1 PARAMETERS FROM APPLICATION TO SMS GATEWAY

No request parameters are required for monitoring.

However, if username and password are provided, SMS Gateway will verify them and additionally check that the client IP address and the protocol used (HTTP or HTTPS) are allowed in user account configuration before responding.

Optional parameters:

Parameter	Description	Supported values or example
user	Username. Alternatively, HTTP Basic authentication can be used.	customer
password	Password for the user.	3fGDy6bp

6.2 RESPONSE FROM SMS GATEWAY

SMS Gateway will respond with an empty HTTP **200 OK** response.

If username and password were provided and access was denied, SMS Gateway will respond with a text/plain error message and HTTP error status code.

6.3 HTTP EXAMPLE

This example uses HTTP POST but GET could be used as well.

6.3.1 Request (application -> SMS Gateway)

```
POST /monitor HTTP/1.1
Host: gw.labyrintti.com:28080
Content-Type: application/x-www-form-urlencoded
Content-Length: 31

user=customer&password=3fGDy6bp
```

6.3.2 Response (SMS Gateway -> application)

```
HTTP/1.1 200 OK
Content-Length: 0
```

7. EXAMPLES

This section provides simple examples for sending SMS messages and implementing SMS services using PHP and Java. Labyrintti Media also provides customers with more advanced versions of these and other examples, ready for production use.

7.1 PHP: SENDING SMS MESSAGE

```
<?php
	$sms_user = "customer";
	$sms_password = "3fGDy6bp";
	$sms_url = "http://gw.labyrintti.com:28080/sendsms";
	$dest = "0401234567";
	$msg = "Hello World";

	// send SMS message
	$params = "user=$sms_user&password=$sms_password&dests=$dest&text=" . urlencode($msg);
	$result = @file($sms_url . "?" . $params);	// @ disables error reporting for security reasons
	if (!$result) {
		echo "SMS message sending failed! Check username, password and IP address.\n";
		exit;
	}

	// result has one line per destination, e.g. "+358401234567 OK 1 message accepted for sending"
	$line = explode(" ", $result[0], 4);
	if ($line[1] == "OK") {
		echo "SMS message successfully sent!\n";
	} else {
		echo "SMS message sending failed! Error: $line[3]";
	}
?>
```

7.2 PHP: SMS SERVICE

```
<?php
	$source = $_REQUEST["source"];
	$msg = $_REQUEST["text"];

	// send response SMS message
	header("Content-Type: text/plain");
	echo "text=Hello $source! Thank you for your message which was: $msg\r\n";
?>
```

7.3 JAVA: SENDING SMS MESSAGE

```
import java.io.*;
import java.net.*;
import java.util.StringTokenizer;

public class SMSSender {
	private static final String	SMS_USER = "customer";
	private static final String	SMS_PASSWORD = "3fGDy6bp";
	private static final String	SMS_URL = "http://gw.labyrintti.com:28080/sendsms";

	/**
	 * Send SMS message.
	 * @param dest		destination phone number in any format
	 * @param msg		the message text
	 * @return		true if message was sent, false if sending failed
	 * @throws IOException	if SMS Gateway cannot be connected or network failure occurs
	 */
	public static boolean sendMessage(String dest, String msg) throws IOException {
		String params = "user=" + SMS_USER + "&password=" + SMS_PASSWORD
				+ "&dests=" + dest + "&text=" + URLEncoder.encode(msg, "ISO-8859-1");

		// send SMS message
		URL url = new URL(SMS_URL + "?" + params);
		HttpURLConnection con = (HttpURLConnection)url.openConnection();
		if (con.getResponseCode() != HttpURLConnection.HTTP_OK) {
			throw new IOException(con.getResponseMessage());
		}

		// result has one line per destination, e.g. "+358401234567 OK 1 message accepted for sending"
		BufferedReader r = new BufferedReader(new InputStreamReader(con.getInputStream()));
		try {
			StringTokenizer st = new StringTokenizer(r.readLine());
			String number = st.nextToken();
			String status = st.nextToken();

			// return true if status is OK, false if status is ERROR
			return status.equals("OK");
		} finally {
			r.close();
		}
	}
}
```

7.4 JAVA: SMS SERVICE

```
import java.io.*;
import javax.servlet.ServletException;
import javax.servlet.http.*;

public class SMSServlet extends HttpServlet {
	/** Process HTTP POST request. */
	protected void doPost(HttpServletRequest req, HttpServletResponse resp)
	throws ServletException, IOException {
		String source = req.getParameter("source");
		String msg = req.getParameter("text");

		// send response SMS message
		resp.setContentType("text/plain");
		PrintWriter out = resp.getWriter();
		out.println("text=Hello " + source + "! Thank you for your message which was: " + msg);
		out.flush();
	}
}
```