



# PSWinCom SMS Gateway WebService (SOAP) Interface Specification

Version date: 2011-08-12

This document and its content is copyrighted by  
PSWinCom AS, Norway

**PSWinCom SMS Gateway service**  
Company: PSWinCom AS, Norway  
E-mail: [support@pswin.com](mailto:support@pswin.com)  
Phone: +47 57748484  
Fax: +47 57748485  
Web: [www.pswin.com](http://www.pswin.com)



## Table of contents

1	Preface .....	3
2	Gateway connect information .....	4
3	Security Considerations .....	4
4	Objects/Complex types .....	5
4.1	ReturnValue .....	5
4.1.1	Return code values .....	6
4.2	SMSMessage .....	6
4.3	WapPushMessage .....	8
4.4	DeliveryReport .....	10
4.4.1	Delivery report states .....	11
4.5	MMSMessage .....	12
4.6	IncomingSMSMessage .....	14
4.7	IncomingMMSMessage .....	15
4.8	GSMPosition .....	17
5	Methods .....	18
5.1	Send SMS methods .....	18
5.1.1	SendSingleMessage() .....	18
5.1.2	SendMessages() .....	18
5.2	Send Wap Push methods .....	19
5.2.1	SendSingleWapPush() .....	19
5.2.2	SendMultipleWapPush() .....	19
5.3	Send MMS methods .....	20
5.3.1	SendSingleMMSMessage() .....	20
5.3.2	SendMMSMessages() .....	20
5.4	Receiving methods .....	21
5.4.1	ReceiveDeliveryReport() .....	21
5.4.2	ReceiveSMSMessage() .....	21
5.4.3	ReceiveMMSMessage() .....	22
5.4.4	Retry scheme for incoming messages .....	22
6	CPA Goods and Services .....	23
6.1	ServiceCodes .....	23
7	Sub-numbering/SMS dialogues .....	25
8	Sample implementations .....	26
8.1	PHP5 Sample .....	26
9	WSDL .....	30
9.1	Send SMS WSDL .....	30
9.2	Send MMS WSDL .....	34
9.3	Receive WSDL .....	37

# 1 Preface

This document describes how to use the WebService (SOAP) Interface for the PSWinCom SMS/MMS Gateway. This document is intended for developers only, and basic knowledge of WebServices and the SOAP protocol is required.

The WebService Interface of the PSWinCom SMS/MMS Gateway service is suitable for application developers that have integrated support for utilizing WebServices in their tools. The interface is "firewall-friendly" since SOAP is running over a standard HTTP protocol and port number 80 is used for client connections.

This interface has methods for both single message sending and bulk SMS message sending as well as for Wap Push.

## 2 Gateway connect information

SMS Webservice URL:	<a href="http://sms.pswin.com/SOAP/SMS.asmx">http://sms.pswin.com/SOAP/SMS.asmx</a>
SMS WSDL:	<a href="http://sms.pswin.com/SOAP/SMS.asmx?wsdl">http://sms.pswin.com/SOAP/SMS.asmx?wsdl</a>
Secure SMS Webservice URL:	<a href="https://secure.pswin.com/SOAP/SMS.asmx">https://secure.pswin.com/SOAP/SMS.asmx</a>
MMS Webservice URL:	<a href="http://sms.pswin.com/SOAP/MMS.asmx">http://sms.pswin.com/SOAP/MMS.asmx</a>
MMS WSDL:	<a href="http://sms.pswin.com/SOAP/MMS.asmx?wsdl">http://sms.pswin.com/SOAP/MMS.asmx?wsdl</a>
Secure MMS Webservice URL:	<a href="https://secure.pswin.com/SOAP/MMS.asmx">https://secure.pswin.com/SOAP/MMS.asmx</a>
Receive WSDL:	<a href="https://secure.pswin.com/SOAP/Receive.asmx?wsdl">https://secure.pswin.com/SOAP/Receive.asmx?wsdl</a>
Account web:	<a href="http://www.pswin.com">http://www.pswin.com</a>
Product web:	<a href="http://www.pswin.com">http://www.pswin.com</a>
Support inquiries:	<a href="mailto:support@pswin.com">support@pswin.com</a>

You can test the SOAP interface from the following page:

SOAP Test:	<a href="https://secure.pswin.com/SOAPTest/Test.aspx">https://secure.pswin.com/SOAPTest/Test.aspx</a>
SOAP Test	<a href="http://download.pswin.com/SoapTestSampleCode.zip">http://download.pswin.com/SoapTestSampleCode.zip</a>
Source Code in C#:	<a href="#">p</a>

## 3 Security Considerations

Using plain HTTP should be done with caution when dealing with information. All data, including username and password, are sent as cleartext over the Internet and may be intercepted by a third-party. To increase security, the SSL secured transport layer (HTTPS) can be used.

All PSWinCom services are secured using a single \*.pswin.com SSL Server certificate. Please make sure that you have the following root and intermediate certificates in your certificate store:

- § AddTrust External CA Root
- § Comodo High-Assurance Secure Server CA

If you are missing any of these, please download and install them from the following file:

<http://download.pswin.com/RootAndIntermediateForStarPSWinCom.zip>

## 4 Objects/Complex types

Several complex types are defined to support the SOAP communication with the Gateway. Following is a description of these complex types and their class representation (C# syntax)

### 4.1 ReturnValue

All methods will return a response value object called ReturnValue. This object has three attributes described as follows:

Attribute	Description
Code	Integer. Statuscode indicating whether the operation was successful or failed. A statuscode of 200 represents a successful operation. Otherwise the Code will indicate an failure, and the Description attribute will contain further information. An overview of statuscodes used can be found in chapter 4.1.1.
Description	String. Textual descriptions of the statuscode given by the Code attribute.
Reference	Optional String. Certain operations may render an unique reference ID that the client may store and later use to track or correlate messages. Typically submit message operations will return a Reference value that the client later will use to correlate with a delivery report. This value must be treated as a string with a length of at least 36 characters.

C# notation of the ReturnValue object:

```
public class ReturnValue
{
    public int Code;
    public string Description;
    public string Reference;

    public ReturnValue()
    {
    }
}
```

Sample XML:

```
<ReturnValue>
  <Code>200</Code>
  <Description>OK</Description>
  <Reference>6542424363623</Reference>
</ReturnValue>
```

### 4.1.1 Return code values

The following table shows the possible values of the Code element on the ReturnValue object:

Code	Description
100	Gateway rejected message. Detailed description can be found in the Description element of the ReturnValue object. Reference value will be empty.
200	Message successfully submitted. Reference (if requested and enabled on account) will be set to a unique value.
500	General SOAP Interface error. Message was not submitted.

## 4.2 SMSMessage

This object is used when submitting SMS messages to the Gateway.

The attributes supported are defined as follows:

Attribute	Description
ReceiverNumber	Number of the receiver, internationally formatted with country-code but no "+" or "00" prefixed.
SenderNumber	Number of sender to be displayed on receiver's handset. Numeric with no "+" or space, max 15 digits. Alphanumeric up to 11 characters can also be used.
Text	The SMS message body. Messages of up to 160 characters will be sent as one SMS. If the length exceeds 160 characters, it will be sent as up to 6 SMS messages of each 134 characters that are concatenated by the phone. The maximum length is thus $6 * 134 = 804$ characters. The ISO-8859-1 character set is used.
Tariff	The amount (in local currency as cents/"ører") to charge the receiver. This feature requires an CPA agreement. Should be set to 0 (zero) when not used.
Network	Optional. Used to indicate special routing of the message. Should only be set upon request from gateway-owner.
TypeOfMessage	Optional. Set this property to send other types of messages than just plain text messages. Supported values are: Text (default)

	vCard vCalendar Ringtone CallerGroupGraphics OperatorLogo RawBinaryUDH OTABookmark OTASettings WapPush Unicode Picture
TimeToLive	Set this value to the number of minutes that the message should be valid before it expires if not delivered. Set to 0 to use the default value.
CPATag	Optional. Used with CPA messages to indicate further details on the subscribers telephone bill what he/she has paid for.
RequestReceipt	If this attribute is set to true and forwarding of delivery reports are enabled for your Gateway account, then the delivery report for this message will be forwarded to your application.
SessionData	Optional. A free text field that can be used to tag the session with customer specific data such as the application name, username, reference-id etc. Must be the same value for all messages in a request when submitting multiple messages in the same request. The maximum length is 200 characters. Leave empty unless required.
AffiliateProgram	Optional. If you are part of an Affiliate-Program offered by the PSWinCom Gateway provider, you can use this element to specify your Affiliate Program code. Leave empty unless required.
DeliveryTime	Optional. String representing a date and time when the Gateway should try to deliver the message. If this parameter is present the message will be considered to be a deferred message that will be queued for future delivery instead of immediately being forwarded to operator. The format is as follows: YYYYMMDDHHmm Sample: 20 <sup>th</sup> of June 2008 at 14:32 should be specified as: 200806201432 DeliveryTime is always in CET. Maximum delay of message is currently one week (7 days).
ServiceCode	Must be specified for CPA Goods And Services transactions, but should not be set for other messages. Refer to chapter 6 for more details.

**C# notation of the SMSMessage object:**

```

public class SMSMessage
{
    public string ReceiverNumber;
    public string SenderNumber;
    public string Text;
    public string Network;
    public string TypeOfMessage;
    public int Tariff;
    public int TimeToLive;
    public string CPATag;
    public bool RequestReceipt;
    public string SessionData;
    public string AffiliateProgram;
    public string DeliveryTime;
    public string ServiceCode;

    public SMSMessage()
    {
    }
}

```

**Sample XML:**

```

<SMSMessage>
  <ReceiverNumber>4792112233</ReceiverNumber>
  <SenderNumber>2077</SenderNumber>
  <Text>Test</Text>
  <Network> </Network>
  <TypeOfMessage> </TypeOfMessage>
  <Tariff>0</Tariff>
  <TimeToLive>0</TimeToLive>
  <CPATag></CPATag>
  <RequestReceipt>true</RequestReceipt>
  <SessionData></SessionData>
  <AffiliateProgram></AffiliateProgram>
  <DeliveryTime></DeliveryTime>
  <ServiceCode></ServiceCode>
</SMSMessage>

```

### 4.3 WapPushMessage

This object is used when submitting Wap Push messages to the Gateway.

The attributes supported are defined as follows:

Attribute	Description
ReceiverNumber	Number of the receiver, internationally formatted with country-code but no "+" or "00" prefixed.
SenderNumber	Number of sender to be displayed on receiver's handset. Numeric with no "+" or space, max 15

	digits. Alphanumeric up to 11 characters can also be used.
Url	The URL of the link that points to the WAP resource.
Description	A short description/name to show together with the Wap link.
Tariff	The amount (in local currency as cents/"ører") to charge the receiver. This feature requires an CPA agreement. Should be set to 0 (zero) when not used.
Network	Optional. Used to indicate special routing of the message. Should only be set upon request from gateway-owner.
TimeToLive	Set this value to the number of minutes that the message should be valid before it expires if not delivered. Set to 0 to use the default value.
CPATag	Optional. Used with CPA messages to indicate further details on the subscribers telephone bill what he/she has paid for.
RequestReceipt	If this attribute is set to true and forwarding of delivery reports are enabled for your Gateway account, then the delivery report for this message will be forwarded to your application.
SessionData	Optional. A free text field that can be used to tag the session with customer specific data such as the application name, username, reference-id etc. Must be the same value for all messages in a request when submitting multiple messages in the same request. The maximum length is 200 characters. Leave empty unless required.
AffiliateProgram	Optional. If you are part of an Affiliate-Program offered by the PSWinCom Gateway provider, you can use this element to specify your Affiliate Program code. Leave empty unless required.
DeliveryTime	Optional. String representing a date and time when the Gateway should try to deliver the message. If this parameter is present the message will be considered to be a deferred message that will be queued for future delivery instead of immediately being forwarded to operator. The format is as follows: YYYYMMDDHHmm Sample: 20th of June 2008 at 14:32 should be specified as: 200806201432

	DeliveryTime is always in CET. Maximum delay of message is currently one week (7 days).
--	---

C# notation of the WapPushMessage object:

```
public class WapPushMessage
{
    public string ReceiverNumber;
    public string SenderNumber;
    public string Url;
    public string Description;
    public string Network;
    public int Tariff;
    public int TimeToLive;
    public string CPATag;
    public bool RequestReceipt;
    public string SessionData;
    public string AffiliateProgram;
    public string DeliveryTime;

    public WapPushMessage ()
    {
    }
}
```

Sample XML:

```
<WapPushMessage>
  <ReceiverNumber>4792112233</ReceiverNumber>
  <SenderNumber>2077</SenderNumber>
  <Url>http://wap.pswin.com</Url>
  <Description>PSWinCom Wap site</Description>
  <Network></Network>
  <Tariff>0</Tariff>
  <TimeToLive>0</TimeToLive>
  <CPATag></CPATag>
  <RequestReceipt>true</RequestReceipt>
  <SessionData></SessionData>
  <AffiliateProgram></AffiliateProgram>
  <DeliveryTime></DeliveryTime>
</WapPushMessage>
```

## 4.4 DeliveryReport

When one has requested a Delivery Report for a message by setting the RequestReceipt parameter, you will receive a delivery report for each message. The DeliveryReport object has the following attributes:

Attribute	Description
ReceiverNumber	The number which the message was sent to.
State	Final state as assigned by the GSM Network or Gateway. See 4.4.1

DeliveryTime	The actual time (in local timezone of the SMSC used) when the message was delivered. Only present for positive delivery reports (State is DELIVRD).
Reference	The unique reference value assigned to the message that this delivery report corresponds to. This value must be treated as a string with a length of at least 36 characters.

C# notation of the DeliveryReport object:

```
public class DeliveryReport
{
    public string State;
    public string ReceiverNumber;
    public string DeliveryTime;
    public string Reference;

    public DeliveryReport()
    {
    }
}
```

Sample XML:

```
<DeliveryReport>
  <State>DELIVRD</State>
  <ReceiverNumber>4792112233</ReceiverNumber>
  <DeliveryTime>2006.11.30 13:53</DeliveryTime>
  <Reference>65435452572</Reference>
</DeliveryReport>
```

#### 4.4.1 Delivery report states

A delivery report indicates the final state of each message, described with a set of predefined states as follows:

State	Description
DELIVRD	SMS was successfully delivered to the receivers phone. If the message is a CPA (Premium SMS), this will be the only state that will result in a charge on the subscribers account and a pay-out to the content-provider.
EXPIRED	The SMS expired while waiting to be delivered. The phone may be out of coverage or not switched on.
UNDELIV	The SMS was undeliverable (not a valid number or no available route to destination).
FAILED	The SMS failed to be delivered because no operator accepted the message or due to internal Gateway error.
BARRED	The receiver number is barred/blocked/not in use. Do not retry message, and remove number from any subscriber list. (Relevant for CPA messages only)
BARREDT	The receiver number is temporarily blocked. May be an

	empty pre-paid account. (Relevant for CPA messages only)
BARREDC	The receiver has blocked for Premium (CPA) messages. (Relevant for CPA messages only)
ZERO_BAL	The receiver has an empty prepaid account. (Relevant for CPA messages only)
INV_NET	Invalid network. Receiver number is not recognized by the target operator.

## 4.5 MMSMessage

This object is used when submitting MMS messages to the Gateway.

To simplify the packaging and transmission of MMS Messages, we have decided to use a simple ZIP file as the packaging of the MMS content. This means that all the content files/parts of the MMS must be zipped together into one zip-file before transmission to the MMS Gateway. The zip-file should not have any internal directory structure; all content files must reside in the root directory of the zip file, like this:

```
MyZippedMMS.zip contains
    Pres.smil
    Image.gif
    Message.txt
```

How to construct an MMS presentation with text, audio and image content is beyond the scope of this interface documentation. Most will use some kind of MMS composer or other software to create the files required. Refer to <http://www.w3.org/AudioVideo/> for more information about creating mobile content and SMIL files.

The zip-file is transmitted as a Byte-array to the MMS Gateway the attribute below named Data.

The table below shows all attributes and their description:

Attribute	Description
ReceiverNumber	Number of the receiver, internationally formatted with country-code but no "+" or "00" prefixed.
SenderNumber	Number of sender to be displayed on receiver's handset. Numeric with no "+" or space, max 15 digits. Alphanumeric up to 11 characters can also be used.
Subject	This is the subject of the MMS to send or receive.
Tariff	Specifies the amount to charge the end-user in units of cents/"ører". For example, to charge the end-user NOK 5,- you specify "500" as the TARIFF value. Only valid values must be used. Valid values are described in the CPA Agreement from PSWinCom that is required to use this

	<b>property.</b>
Data	Byte-array containing the MMS Message content as a packed zip-file.
Network	Optional. Used to indicate special routing of the message. Should only be set upon request from gateway-owner.
TimeToLive	Set this value to the number of minutes that the message should be valid before it expires if not delivered. Set to 0 to use the default value.
CPATag	Optional. Used with CPA messages to indicate further details on the subscribers telephone bill what he/she has paid for.
RequestReceipt	If this attribute is set to true and forwarding of delivery reports are enabled for your Gateway account, then the delivery report for this message will be forwarded to your application.
SessionData	Optional. A free text field that can be used to tag the session with customer specific data such as the application name, username, reference-id etc. Must be the same value for all messages in a request when submitting multiple messages in the same request. The maximum length is 200 characters. Leave empty unless required.
AffiliateProgram	Optional. If you are part of an Affiliate-Program offered by the PSWinCom Gateway provider, you can use this element to specify your Affiliate Program code. Leave empty unless required.

#### C# notation of the SMSMessage object:

```

public class MMSMessage
{
    public string ReceiverNumber;
    public string SenderNumber;
    public string Subject;
    public string Network;
    public int Tariff;
    public int TimeToLive;
    public string CPATag;
    public bool RequestReceipt;
    public string SessionData;
    public string AffiliateProgram;
    public byte[] Data;

    public MMSMessage()
    {
    }
}

```

#### Sample XML:

```

<MMSMessage>
  <ReceiverNumber>4792112233</ReceiverNumber>
  <SenderNumber>2077</SenderNumber>
  <Subject>My MMS</Text>
  <Network> </Network>
  <Tariff>0</Tariff>
  <TimeToLive>0</TimeToLive>
  <CPATag></CPATag>
  <RequestReceipt>true</RequestReceipt>
  <SessionData></SessionData>
  <AffiliateProgram></AffiliateProgram>
  <Data>____</Data>
</MMSMessage>

```

## 4.6 IncomingSMSMessage

This object is used when receiving an incoming SMS message from the Gateway.

The attributes supported are defined as follows:

Attribute	Description
ReceiverNumber	Number that the message was sent to. This may be an international formatted number (with country prefix) or an operator specific short/long number (for example 2077)
SenderNumber	Number of the subscriber that sent the message. This will be an internationally formatted number (with country prefix).
Text	The message text received.
Network	Optional. May contain information about the Network which the message is received through. This information can be ignored unless otherwise instructed by PSWinCom.
Address	<p>Optional. This attribute may contain detailed information about the sender, such as name and address. The information is retrieved by the Gateway which is requesting such data from a phone directory service. The internal format is as follows:</p> <p>Firstname;lastname;address; ZipCode;City;RegionNumber;CountyNumber</p> <p>Sample result: Kari;;Nordmann;Hjemmeveien 46;5211; BERGEN;12;1201</p> <p>Additional values may be added at the end in the future.</p> <p>This is a value added feature that requires an</p>

	additional agreement with PSWinCom.
GSMPosition	Optional. May contain geographical position information about the subscriber. This is a value added feature that requires an additional agreement with PSWinCom. See chapter 4.8.

C# notation of the IncomingSMSMessage object:

```
public class IncomingSMSMessage
{
    public string ReceiverNumber;
    public string SenderNumber;
    public string Text;
    public string Network;
    public string Address;
    public GSMPosition Position;

    public IncomingSMSMessage()
    {
    }
}
```

Sample XML:

```
<IncomingSMSMessage>
  <ReceiverNumber>2077</ReceiverNumber>
  <SenderNumber>4792112233</SenderNumber>
  <Text>test incoming</Text>
  <Network></Network>
  <Address></Address>
  <Position>
    <Longitude></Longitude>
    <Lattitude></Lattitude>
    <Radius></Radius>
    <County></County>
    <Council></Council>
    <CouncilNumber></CouncilNumber>
    <Place></Place>
    <SubPlace></SubPlace>
    <ZipCode></ZipCode>
    <City></City>
  </Position>
</IncomingSMSMessage>
```

## 4.7 IncomingMMSMessage

This object is used when receiving an incoming MMS message from the Gateway.

The attributes supported are defined as follows:

Attribute	Description
-----------	-------------

ReceiverNumber	Number that the message was sent to. This may be an international formatted number (with country prefix) or an operator specific short/long number (for example 2077)
SenderNumber	Number of the subscriber that sent the message. This will be an internationally formatted number (with country prefix).
Subject	The subject of the message
Data	Byte-array containing the MMS Message content as a packed zip-file. See chapter 4.5 for more details.
Network	Optional. May contain information about the Network which the message is received through. This information can be ignored unless otherwise instructed by PSWinCom.
Address	Optional. This attribute may contain detailed information about the sender, such as name and address. The information is retrieved by the Gateway which is requesting such data from a phone directory service. This is a value added feature that requires an additional agreement with PSWinCom.
GSMPosition	Optional. May contain geographical position information about the subscriber. This is a value added feature that requires an additional agreement with PSWinCom. See chapter 4.8.

#### C# notation of the IncomingMMSMessage object:

```
public class IncomingMMSMessage
{
    public string ReceiverNumber;
    public string SenderNumber;
    public string Subject;
    public string Network;
    public string Address;
    public byte[] Data;
    public GSMPosition Position;

    public IncomingMMSMessage()
    {
    }
}
```

#### Sample XML:

```
<IncomingMMSMessage>
  <ReceiverNumber>2077</ReceiverNumber>
  <SenderNumber>4792112233</SenderNumber>
```

```
<Subject>test MMS incoming</Subject>
<Data>__</Data>
<Network></Network>
<Address></Address>
<Position>
  <Longitude></Longitude>
  <Latitude></Latitude>
  <Radius></Radius>
  <County></County>
  <Council></Council>
  <CouncilNumber></CouncilNumber>
  <Place></Place>
  <SubPlace></SubPlace>
  <ZipCode></ZipCode>
  <City></City>
</Position>
</IncomingMMSMessage>
```

## 4.8 GSMPosition

This object may contain geographical position information for incoming messages. This feature requires a separate PSWinCom Positioning Gateway agreement. For further details, please contact [support@pswin.com](mailto:support@pswin.com).

## 5 Methods

### 5.1 Send SMS methods

The service and WSDL of the send SMS methods can be found here:

SMS Webservice URL: <http://sms.pswin.com/SOAP/SMS.asmx>  
 SMS WSDL: <http://sms.pswin.com/SOAP/SMS.asmx?wsdl>  
 Secure SMS Webservice URL: <https://secure.pswin.com/SOAP/SMS.asmx>

#### 5.1.1 SendSingleMessage()

This method is used to submit a single SMS message to the Gateway.

Syntax:

```
ReturnValue rv = SendSingleMessage(string username,
                                   string password,
                                   SMSMessage msg)
```

The `username` and `password` parameters are string values containing user id and password for your PSWinCom SMS Gateway account. The `msg` parameter is a complex type named `SMSMessage` (as defined in chapter 4.2) that holds the various attributes of an SMS message. The method has a complex type named `ReturnValue` (as defined in chapter 4.1) as return value.

#### 5.1.2 SendMessages()

This method is used to submit multiple SMS messages to the Gateway with a single operation. This is generally more efficient for bulk sending than the `SendSingleMessage` method.

Syntax:

```
ReturnValue[] rv = SendMessages(string username,
                                string password,
                                SMSMessage[] msg)
```

The `username` and `password` parameters are string values containing user id and password for your PSWinCom SMS Gateway account. The `msg` parameter is an array of the complex type named `SMSMessage` (as defined in chapter 4.2) that holds the various attributes of each SMS message. The method will return an array of the complex type named `ReturnValue` (as defined in chapter 4.1) as return value. The returned array will be of the same size as the `msg` array.

## 5.2 Send Wap Push methods

The service and WSDL of the send Wap Push methods can be found here:

SMS Webservice URL: <http://sms.pswin.com/SOAP/SMS.asmx>  
 SMS WSDL: <http://sms.pswin.com/SOAP/SMS.asmx?wsdl>  
 Secure SMS Webservice URL: <https://secure.pswin.com/SOAP/SMS.asmx>

### 5.2.1 SendSingleWapPush()

This method is used to submit a single Wap Push message to the Gateway.

Syntax:

```
ReturnValue rv = SendSingleWapPush(string username,
                                   string password,
                                   WapPushMessage msg)
```

The `username` and `password` parameters are string values containing user id and password for your PSWinCom SMS Gateway account. The `msg` parameter is a complex type named `WapPushMessage` (as defined in chapter 4.3) that holds the various attributes of an Wap Push message. The method has a complex type named `ReturnValue` (as defined in chapter 4.1) as return value.

### 5.2.2 SendMultipleWapPush()

This method is used to submit multiple Wap Push messages to the Gateway with a single operation. This is generally more efficient for bulk sending than the `SendSingleWapPush` method.

Syntax:

```
ReturnValue[] rv = SendMultipleWapPush(
    string username,
    string password,
    WapPushMessage[] msg)
```

The `username` and `password` parameters are string values containing user id and password for your PSWinCom SMS Gateway account. The `msg` parameter is an array of the complex type named `WapPushMessage` (as defined in chapter 4.3) that holds the various attributes of each Wap Push message. The method will return an array of the complex type named `ReturnValue` (as defined in chapter 4.1) as

return value. The returned array will be of the same size as the `msg` array.

## 5.3 Send MMS methods

The service and WSDL of the send MMS methods can be found here:

MMS Webservice URL: <http://sms.pswin.com/SOAP/MMS.asmx>  
 MMS WSDL: <http://sms.pswin.com/SOAP/MMS.asmx?wsdl>  
 Secure MMS Webservice URL: <https://secure.pswin.com/SOAP/MMS.asmx>

### 5.3.1 SendSingleMMSMessage()

This method is used to submit a single MMS message to the Gateway.

Syntax:

```
ReturnValue rv = SendSingleMMSMessage(
    string username,
    string password,
    MMSMessage msg)
```

The `username` and `password` parameters are string values containing user id and password for your PSWinCom SMS/MMS Gateway account. The `msg` parameter is a complex type named `MMSMessage` (as defined in chapter 4.5) that holds the various attributes of an MMS message. The method has a complex type named `ReturnValue` (as defined in chapter 4.1) as return value.

### 5.3.2 SendMMSMessages()

This method is used to submit multiple MMS messages to the Gateway with a single operation. This is generally more efficient for bulk sending than the `SendSingleMMSMessage` method.

Syntax:

```
ReturnValue[] rv = SendMMSMessages(string username,
    string password,
    MMSMessage[] msg)
```

The `username` and `password` parameters are string values containing user id and password for your PSWinCom SMS/MMS Gateway account. The `msg` parameter is an array of the complex type named `MMSMessage` (as defined in chapter 4.5) that holds the various attributes of each MMS message. The method will return an array of

the complex type named `ReturnValue` (as defined in chapter 4.1) as return value. The returned array will be of the same size as the `msg` array.

## 5.4 Receiving methods

The SMS/MMS Gateway may delivery incoming messages and delivery reports to the client using SOAP/WebServices. To be able to receive those messages and delivery reports the client must implement the interface as defined by the following WSDL:

Receive WSDL: <http://sms.pswin.com/SOAP/Receive.asmx?wsdl>

The customer must implement their service using the same contract (WSDL) as the service above. After implementation, the URL of the service must be provided to the Gateway. Please contact [support@pswin.com](mailto:support@pswin.com) to arrange this.

Each method must have the complex type `ReturnValue` as a return value. The client must set the `Code` attribute to the value 200 to indicate a successful reception. Any other status code value will be interpreted as an unsuccessful delivery and the message or delivery report will be retried according to the retry scheme of the Gateway.

The methods used are as follows:

### 5.4.1 ReceiveDeliveryReport()

Delivery reports will be forwarded to the client when requested with the `RequestReceipt` attribute on the `SMSMessage`, `MMSMessage` or `WapPushMessage` object.

Syntax:

```
ReturnValue rv = ReceiveDeliveryReport(DeliveryReport dr)
```

The `DeliveryReport` parameter is a complex type as defined in chapter 4.4.

### 5.4.2 ReceiveSMSMessage()

Incoming SMS messages will be forwarded to the client according to the rules defined for the client and the access number used for two-way communication.

Syntax:

```
ReturnValue rv = ReceiveSMSMessage(IncomingSMSMessage msg)
```

The `IncomingSMSMessage` parameter is a complex type as defined in chapter 4.6.

### 5.4.3 ReceiveMMSMessage()

Incoming MMS messages will be forwarded to the client according to the rules defined for the client and the access number used for two-way communication.

Syntax:

```
ReturnValue rv = ReceiveMMSMessage(IncomingMMSMessage msg)
```

The `IncomingMMSMessage` parameter is a complex type as defined in chapter 4.7.

### 5.4.4 Retry scheme for incoming messages

Incoming messages (messages received from a subscriber) and delivery reports can be forwarded to the client using the SOAP interface. The Gateway will try to deliver the message/report up to 5 times if the receiver doesn't reply properly. The delay between each delivery attempt will be as follow:

Attempt no	Delay (since latter attempt)
1	Immediately
2	5 minutes later
3	10 minutes later
4	60 minutes later
5	120 minutes later

If a message or delivery reports is received repeatedly, then the client is probably not replying correctly back to the Gateway within reasonable time.

The customer may request a backup/failover destination to be configured as a secondary delivery destination. This destination will then be tried for each delivery attempt if the main destination doesn't reply properly.

## 6 CPA Goods and Services

The PSWinCom Gateway supports billing of Goods and Services (CPA GAS) using mobile phones in Norway. As opposed to traditional CPA/Premium SMS which can only be used to bill mobile content, CPA GAS can only be used to bill goods and services.

Tariff values up to NOK 300,- can be used.

Each transaction must contain a special ServiceCode. The ServiceCode specifies what kind of goods or services the transaction is related to. Valid ServiceCodes are given in chapter 6.1.

The Gateway will use the NRDB (Nasjonal Referansedatabase) to resolve which operator the receiver belongs to.

To use CPA Goods and Services with the PSWinCom Gateway you must sign up for a separate CPA GAS agreement. Terms of settlement and general CPA terms are discussed in separate CPA GAS Agreement document available from [www.pswin.com](http://www.pswin.com).

### 6.1 ServiceCodes

The table below shows valid ServiceCodes as per this document date.

ServiceCode	Description
05001	Teaterbillett
05002	Kinobillett
05003	Konsertbillett
05004	Bok
05005	Lydbok - CD
05006	Film - DVD
05007	Musikk - CD
05008	Avis
05009	Magasin
06001	Togbillett
06002	Bussbillett
06003	T-bane/trikkebillett
06004	Taxi
06005	Parkering
06006	Fergebillett
06007	Veiavgift
07001	Medisinsk behandling
07002	Medisiner
07003	Off. avgifter medisinsk behandling
08001	Reklame
08002	Gavekort
09001	Gavekort
09002	Forsikringstjeneste
10001	Mat og drikke
11001	Servering
12001	Overnatting
13001	Fiskeravgift

13002	Fiskekort
13003	Tilgangsavgift sport
13004	Rådgivning/coaching
14001	Medlemsavgift
14002	Givertjenester
15001	Fysiske varer
89001	Nulltakst informasjonsmelding

## 7 Sub-numbering/SMS dialogues

Certain access numbers can be extended with subnumbers. Subnumbers are extra digits added to the end of the original access number. Typical usage of subnumbers are to create dialogue-based SMS applications.

**Example:**

Accessnumber 26199 in Norway can be extended with 9 digits, like 26199123456789. PSWinCom Gateway can allocate a range of this subnumbered accessnumber to your account, for example 26199123456000 – 26199123456999. When you send SMS, you can set the SenderNumber to a new unique subnumber for each SMS you send. When the receiver replies to the SMS on his phone, you will receive an incoming SMS where the ReceiverNumber is the unique subnumber you assigned that particular message.

## 8 Sample implementations

### 8.1 PHP5 Sample

The following sample shows how to use the SOAP API from PHP5.

To create the client, we suggest using a tool for code-generation such as `wSDL2php` you can find here:

<http://www.urdalen.no/wSDL2php/index.php>

Note: The generator requires PEAR. PHP5 users should install PEAR for PHP5 by running "pear5 install".

When running the code generator against the WSDL for the PSWinCom Gateway SOAP Interface, you will get code as shown below (not all methods are shown, just the minimum required to submit a test message):

#### File SMSMessage.php

```
<?php
/**
 *
 *
 * @package
 * @copyright
 */
class SMSMessage {
    /** string */
    public $ReceiverNumber;
    /** string */
    public $SenderNumber;
    /** string */
    public $Text;
    /** string */
    public $Network;
    /** string */
    public $TypeOfMessage;
    /** int */
    public $Tariff;
    /** int */
    public $TimeToLive;
    /** string */
    public $CPATag;
    /** boolean */
    public $RequestReceipt;
    /** string */
    public $SessionData;
    /** string */
    public $AffiliateProgram;
    /** string */
    public $DeliveryTime;
}
?>
```

#### File SendSingleMessage.php

```
<?php
/**
 *
 *
 */
```

```

* @package
* @copyright
*/
class SendSingleMessage {
    /* string */
    public $username;
    /* string */
    public $password;
    /* SMSMessage */
    public $m;
}
?>

```

## File SMSService.php

```

<?php
/**
 * SMSService class file
 *
 * @author {author}
 * @copyright {copyright}
 * @package {package}
 */

/**
 * SendSingleMessage class
 */
require_once 'SendSingleMessage.php';
/**
 * SMSMessage class
 */
require_once 'SMSMessage.php';

/**
 * SMSService class
 *
 * PSWinCom SMS Gateway SOAP Interface
 *
 * @author {author}
 * @copyright {copyright}
 * @package {package}
 */
class SMSService extends SoapClient {

    public function SMSService($wsdl =
"http://sms.pswin.com/SOAP/SMS.asmx?wsdl", $options = array()) {
        parent::__construct($wsdl, $options);
    }

    /**
     *
     *
     * @param SendSingleMessage $parameters
     * @return SendSingleMessageResponse
     */
    public function SendSingleMessage(SendSingleMessage $parameters) {
        return $this->__call('SendSingleMessage', array(
            new SoapParam($parameters, 'parameters')
        ),
        array(
            'uri' => 'http://pswin.com/SOAP/Submit/SMS',
            'soapaction' => ''
        )
    );
    }
}
?>

```

## File Test.php

```
<?php
require_once('SMSService.php');

// Lag en ny melding
$objMessage = new SMSMessage();
$objMessage->ReceiverNumber = '(receiver number including countrycode)';
$objMessage->SenderNumber = 'PSWinCom';
$objMessage->Text = 'Test message';
$objMessage->Tariff = 0;
$objMessage->TimeToLive = 0;
$objMessage->RequestReceipt = false;

// Lag parametere
$objSendSingleMessage = new SendSingleMessage();
$objSendSingleMessage->m = $objMessage;
$objSendSingleMessage->username = '(username)';
$objSendSingleMessage->password = '(password)';

// Koble opp mot tjenesten
$objService = new SMSService();
// Send meldingen
$objReturn = $objService->SendSingleMessage($objSendSingleMessage);

echo '<pre>';
var_dump($objReturn);
echo '</pre>';

?>
```



## 9 WSDL

### 9.1 Send SMS WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:tns="http://pswin.com/SOAP/Submit/SMS"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
targetNamespace="http://pswin.com/SOAP/Submit/SMS"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://pswin.com/SOAP/Submit/SMS">
      <s:element name="SendSingleMessage">
        <s:complexType>
          <s:sequence>
            <s:element name="username"
type="s:string" />
            <s:element name="password"
type="s:string" />
            <s:element name="m"
type="tns:SMSMessage" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="SMSMessage">
        <s:sequence>
          <s:element name="ReceiverNumber"
type="s:string" />
          <s:element name="SenderNumber"
type="s:string" />
          <s:element name="Text"
type="s:string" />
          <s:element name="Network"
nillable="true" type="s:string" />
          <s:element name="TypeOfMessage"
nillable="true" type="s:string" />
          <s:element name="Tariff"
type="s:int" />
          <s:element name="TimeToLive"
type="s:int" />
          <s:element name="CPATag"
nillable="true" type="s:string" />
          <s:element name="RequestReceipt"
type="s:boolean" />
          <s:element name="SessionData"
nillable="true" type="s:string" />
          <s:element name="AffiliateProgram"
nillable="true" type="s:string" />
          <s:element name="DeliveryTime"
nillable="true" type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:element name="SendSingleMessageResponse">
        <s:complexType>
          <s:sequence>
            <s:element name="SendSingleMessageResult" type="tns:ReturnValue" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="ReturnValue">
        <s:sequence>
          <s:element name="Code" type="s:int" />
          <s:element name="Description"
type="s:string" />
          <s:element name="Reference"
type="s:string" />
        </s:sequence>
      </s:complexType>
    </s:schema>
  </wsdl:types>

```

```

</s:complexType>
<s:element name="SendMessages">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="username"
type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="password"
type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="m"
type="tns:ArrayOfSMSMessage" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="ArrayOfSMSMessage">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="SMSMessage"
nillable="true" type="tns:SMSMessage" />
  </s:sequence>
</s:complexType>
<s:element name="SendMessagesResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1"
name="SendMessagesResult" type="tns:ArrayOfReturnValue" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="ArrayOfReturnValue">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded"
name="ReturnValue" nillable="true" type="tns:ReturnValue" />
  </s:sequence>
</s:complexType>
<s:element name="SendSingleWapPush">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="username"
type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="password"
type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="m"
type="tns:WapPushMessage" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="WapPushMessage">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="1" name="ReceiverNumber"
type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="SenderNumber"
type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="Url"
type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="Description"
type="s:string" />
    <s:element minOccurs="1" maxOccurs="1" name="Network"
nillable="true" type="s:string" />
    <s:element minOccurs="1" maxOccurs="1" name="Tariff"
type="s:int" />
    <s:element minOccurs="1" maxOccurs="1" name="TimeToLive"
type="s:int" />
    <s:element minOccurs="1" maxOccurs="1" name="CPATag"
nillable="true" type="s:string" />
    <s:element minOccurs="1" maxOccurs="1" name="RequestReceipt"
type="s:boolean" />
    <s:element minOccurs="1" maxOccurs="1" name="SessionData"
nillable="true" type="s:string" />
    <s:element minOccurs="1" maxOccurs="1" name="AffiliateProgram"
nillable="true" type="s:string" />
    <s:element minOccurs="1" maxOccurs="1" name="DeliveryTime"
nillable="true" type="s:string" />
  </s:sequence>
</s:complexType>
<s:element name="SendSingleWapPushResponse">
  <s:complexType>
    <s:sequence>

```

```

        <s:element minOccurs="0" maxOccurs="1"
name="SendSingleWapPushResult" type="tns:ReturnValue" />
    </s:sequence>
</s:complexType>
</s:element>
<s:element name="SendMultipleWapPush">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="username"
type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="password"
type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="m"
type="tns:ArrayOfWapPushMessage" />
        </s:sequence>
    </s:complexType>
</s:element>
<s:complexType name="ArrayOfWapPushMessage">
    <s:sequence>
        <s:element minOccurs="0" maxOccurs="unbounded"
name="WapPushMessage" nillable="true" type="tns:WapPushMessage" />
    </s:sequence>
</s:complexType>
<s:element name="SendMultipleWapPushResponse">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="SendMultipleWapPushResult" type="tns:ArrayOfReturnValue" />
        </s:sequence>
    </s:complexType>
</s:element>
</s:schema>
</wsdl:types>
<wsdl:message name="SendSingleMessageSoapIn">
    <wsdl:part name="parameters" element="tns:SendSingleMessage" />
</wsdl:message>
<wsdl:message name="SendSingleMessageSoapOut">
    <wsdl:part name="parameters" element="tns:SendSingleMessageResponse"
/>
</wsdl:message>
<wsdl:message name="SendMessagesSoapIn">
    <wsdl:part name="parameters" element="tns:SendMessages" />
</wsdl:message>
<wsdl:message name="SendMessagesSoapOut">
    <wsdl:part name="parameters" element="tns:SendMessagesResponse" />
</wsdl:message>
<wsdl:message name="SendSingleWapPushSoapIn">
    <wsdl:part name="parameters" element="tns:SendSingleWapPush" />
</wsdl:message>
<wsdl:message name="SendSingleWapPushSoapOut">
    <wsdl:part name="parameters" element="tns:SendSingleWapPushResponse"
/>
</wsdl:message>
<wsdl:message name="SendMultipleWapPushSoapIn">
    <wsdl:part name="parameters" element="tns:SendMultipleWapPush" />
</wsdl:message>
<wsdl:message name="SendMultipleWapPushSoapOut">
    <wsdl:part name="parameters" element="tns:SendMultipleWapPushResponse"
/>
</wsdl:message>
<wsdl:portType name="SMSServiceSoap">
    <wsdl:operation name="SendSingleMessage">
        <wsdl:input message="tns:SendSingleMessageSoapIn" />
        <wsdl:output message="tns:SendSingleMessageSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="SendMessages">
        <wsdl:input message="tns:SendMessagesSoapIn" />
        <wsdl:output message="tns:SendMessagesSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="SendSingleWapPush">
        <wsdl:input message="tns:SendSingleWapPushSoapIn" />
        <wsdl:output message="tns:SendSingleWapPushSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="SendMultipleWapPush">
        <wsdl:input message="tns:SendMultipleWapPushSoapIn" />
        <wsdl:output message="tns:SendMultipleWapPushSoapOut" />
    </wsdl:operation>

```

```

</wsdl:portType>
<wsdl:binding name="SMSServiceSoap" type="tns:SMSServiceSoap">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
style="document" />
  <wsdl:operation name="SendSingleMessage">
    <soap:operation
soapAction="http://pswin.com/SOAP/Submit/SMS/SendSingleMessage"
style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="SendMessages">
    <soap:operation
soapAction="http://pswin.com/SOAP/Submit/SMS/SendMessages"
style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="SendSingleWapPush">
    <soap:operation
soapAction="http://pswin.com/SOAP/Submit/SMS/SendSingleWapPush"
style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="SendMultipleWapPush">
    <soap:operation
soapAction="http://pswin.com/SOAP/Submit/SMS/SendMultipleWapPush"
style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
<wsdl:service name="SMSService">
  <documentation xmlns="http://schemas.xmlsoap.org/wsdl/">PSWinCom SMS
Gateway SOAP Interface</documentation>
  <wsdl:port name="SMSServiceSoap" binding="tns:SMSServiceSoap">
    <soap:address location="http://localhost/soap/sms.asmx" />
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```

## 9.2 Send MMS WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:tns="http://pswin.com/SOAP/Submit/MMS"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
targetNamespace="http://pswin.com/SOAP/Submit/MMS"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://pswin.com/SOAP/Submit/MMS">
      <s:element name="SendSingleMMSMessage">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="username"
type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="password"
type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="m"
type="tns:MMSMessage" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="MMSMessage">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="ReceiverNumber"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="SenderNumber"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Subject"
type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="Network"
nillable="true" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="Tariff"
type="s:int" />
          <s:element minOccurs="1" maxOccurs="1" name="TimeToLive"
type="s:int" />
          <s:element minOccurs="1" maxOccurs="1" name="CPATag"
nillable="true" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="RequestReceipt"
type="s:boolean" />
          <s:element minOccurs="1" maxOccurs="1" name="SessionData"
nillable="true" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="AffiliateProgram"
nillable="true" type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Data"
type="s:base64Binary" />
        </s:sequence>
      </s:complexType>
      <s:element name="SendSingleMMSMessageResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="SendSingleMMSMessageResult" type="tns:ReturnValue" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="ReturnValue">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="Code" type="s:int"
/>
          <s:element minOccurs="0" maxOccurs="1" name="Description"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Reference"
type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:element name="SendMMSMessages">
        <s:complexType>
          <s:sequence>

```

```

        <s:element minOccurs="0" maxOccurs="1" name="username"
type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="password"
type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="m"
type="tns:ArrayOfMMSMessage" />
    </s:sequence>
</s:complexType>
</s:element>
<s:complexType name="ArrayOfMMSMessage">
    <s:sequence>
        <s:element minOccurs="0" maxOccurs="unbounded" name="MMSMessage"
nillable="true" type="tns:MMSMessage" />
    </s:sequence>
</s:complexType>
<s:element name="SendMMSMessagesResponse">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="SendMMSMessagesResult" type="tns:ArrayOfReturnValue" />
        </s:sequence>
    </s:complexType>
</s:element>
<s:complexType name="ArrayOfReturnValue">
    <s:sequence>
        <s:element minOccurs="0" maxOccurs="unbounded"
name="ReturnValue" nillable="true" type="tns:ReturnValue" />
    </s:sequence>
</s:complexType>
</s:schema>
</wsdl:types>
<wsdl:message name="SendSingleMMSMessageSoapIn">
    <wsdl:part name="parameters" element="tns:SendSingleMMSMessage" />
</wsdl:message>
<wsdl:message name="SendSingleMMSMessageSoapOut">
    <wsdl:part name="parameters"
element="tns:SendSingleMMSMessageResponse" />
</wsdl:message>
<wsdl:message name="SendMMSMessagesSoapIn">
    <wsdl:part name="parameters" element="tns:SendMMSMessages" />
</wsdl:message>
<wsdl:message name="SendMMSMessagesSoapOut">
    <wsdl:part name="parameters" element="tns:SendMMSMessagesResponse" />
</wsdl:message>
<wsdl:portType name="MMSServiceSoap">
    <wsdl:operation name="SendSingleMMSMessage">
        <wsdl:input message="tns:SendSingleMMSMessageSoapIn" />
        <wsdl:output message="tns:SendSingleMMSMessageSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="SendMMSMessages">
        <wsdl:input message="tns:SendMMSMessagesSoapIn" />
        <wsdl:output message="tns:SendMMSMessagesSoapOut" />
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="MMSServiceSoap" type="tns:MMSServiceSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
style="document" />
    <wsdl:operation name="SendSingleMMSMessage">
        <soap:operation
soapAction="http://pswin.com/SOAP/Submit/MMS/SendSingleMMSMessage"
style="document" />
        <wsdl:input>
            <soap:body use="literal" />
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal" />
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SendMMSMessages">
        <soap:operation
soapAction="http://pswin.com/SOAP/Submit/MMS/SendMMSMessages"
style="document" />
        <wsdl:input>
            <soap:body use="literal" />
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal" />
        </wsdl:output>
    </wsdl:operation>

```

```
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="MMSService">
  <documentation xmlns="http://schemas.xmlsoap.org/wsdl/">PSWinCom MMS
Gateway SOAP Interface</documentation>
  <wsdl:port name="MMSServiceSoap" binding="tns:MMSServiceSoap">
    <soap:address location="http://localhost/soap/mms.asmx" />
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

## 9.3 Receive WSDL

```

<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:tns="http://pswin.com/SOAP/Receive"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
targetNamespace="http://pswin.com/SOAP/Receive"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://pswin.com/SOAP/Receive">
      <s:element name="ReceiveSMSMessage">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="m"
type="tns:IncomingSMSMessage" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="IncomingSMSMessage">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="ReceiverNumber"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="SenderNumber"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Text"
type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="Network"
nillable="true" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="Address"
nillable="true" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="Position"
nillable="true" type="tns:GSMPosition" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="GSMPosition">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="Longitude"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Latitude"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Radius"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="County"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Council"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="CouncilNumber"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Place"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="SubPlace"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="ZipCode"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="City"
type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:element name="ReceiveSMSMessageResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="ReceiveSMSMessageResult" type="tns:ReturnValue" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="ReturnValue">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="Code" type="s:int"
/>

```

```

        <s:element minOccurs="0" maxOccurs="1" name="Description"
type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="Reference"
type="s:string" />
    </s:sequence>
</s:complexType>
<s:element name="ReceiveDeliveryReport">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="dr"
type="tns:DeliveryReport" />
        </s:sequence>
    </s:complexType>
</s:element>
<s:complexType name="DeliveryReport">
    <s:sequence>
        <s:element minOccurs="0" maxOccurs="1" name="State"
type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="ReceiverNumber"
type="s:string" />
        <s:element minOccurs="1" maxOccurs="1" name="DeliveryTime"
nillable="true" type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="Reference"
type="s:string" />
    </s:sequence>
</s:complexType>
<s:element name="ReceiveDeliveryReportResponse">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="ReceiveDeliveryReportResult" type="tns:ReturnValue" />
        </s:sequence>
    </s:complexType>
</s:element>
<s:element name="ReceiveMMSMessage">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="m"
type="tns:IncomingMMSMessage" />
        </s:sequence>
    </s:complexType>
</s:element>
<s:complexType name="IncomingMMSMessage">
    <s:sequence>
        <s:element minOccurs="0" maxOccurs="1" name="ReceiverNumber"
type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="SenderNumber"
type="s:string" />
        <s:element minOccurs="0" maxOccurs="1" name="Subject"
type="s:string" />
        <s:element minOccurs="1" maxOccurs="1" name="Network"
nillable="true" type="s:string" />
        <s:element minOccurs="1" maxOccurs="1" name="Address"
nillable="true" type="s:string" />
        <s:element minOccurs="1" maxOccurs="1" name="Position"
nillable="true" type="tns:GSMPosition" />
        <s:element minOccurs="0" maxOccurs="1" name="Data"
type="s:base64Binary" />
    </s:sequence>
</s:complexType>
<s:element name="ReceiveMMSMessageResponse">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="ReceiveMMSMessageResult" type="tns:ReturnValue" />
        </s:sequence>
    </s:complexType>
</s:element>
</s:schema>
</wsdl:types>
<wsdl:message name="ReceiveSMSMessageSoapIn">
    <wsdl:part name="parameters" element="tns:ReceiveSMSMessage" />
</wsdl:message>
<wsdl:message name="ReceiveSMSMessageSoapOut">
    <wsdl:part name="parameters" element="tns:ReceiveSMSMessageResponse"
/>
</wsdl:message>

```

```

<wsdl:message name="ReceiveDeliveryReportSoapIn">
  <wsdl:part name="parameters" element="tns:ReceiveDeliveryReport" />
</wsdl:message>
<wsdl:message name="ReceiveDeliveryReportSoapOut">
  <wsdl:part name="parameters"
element="tns:ReceiveDeliveryReportResponse" />
</wsdl:message>
<wsdl:message name="ReceiveMMSMessageSoapIn">
  <wsdl:part name="parameters" element="tns:ReceiveMMSMessage" />
</wsdl:message>
<wsdl:message name="ReceiveMMSMessageSoapOut">
  <wsdl:part name="parameters" element="tns:ReceiveMMSMessageResponse"
/>
</wsdl:message>
<wsdl:portType name="SMSReceiveSoap">
  <wsdl:operation name="ReceiveSMSMessage">
    <wsdl:input message="tns:ReceiveSMSMessageSoapIn" />
    <wsdl:output message="tns:ReceiveSMSMessageSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="ReceiveDeliveryReport">
    <wsdl:input message="tns:ReceiveDeliveryReportSoapIn" />
    <wsdl:output message="tns:ReceiveDeliveryReportSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="ReceiveMMSMessage">
    <wsdl:input message="tns:ReceiveMMSMessageSoapIn" />
    <wsdl:output message="tns:ReceiveMMSMessageSoapOut" />
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="SMSReceiveSoap" type="tns:SMSReceiveSoap">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
style="document" />
  <wsdl:operation name="ReceiveSMSMessage">
    <soap:operation
soapAction="http://pswin.com/SOAP/Receive/ReceiveSMSMessage"
style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="ReceiveDeliveryReport">
    <soap:operation
soapAction="http://pswin.com/SOAP/Receive/ReceiveDeliveryReport"
style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="ReceiveMMSMessage">
    <soap:operation
soapAction="http://pswin.com/SOAP/Receive/ReceiveMMSMessage"
style="document" />
    <wsdl:input>
      <soap:body use="literal" />
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal" />
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
<wsdl:service name="SMSReceive">
  <documentation xmlns="http://schemas.xmlsoap.org/wsdl/" />
  <wsdl:port name="SMSReceiveSoap" binding="tns:SMSReceiveSoap">
    <soap:address location="http://localhost/soap/receive.asmx" />
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```