WINWAP TECHNOLOGIES

HOW TO SEND AND RECEIVE MMS MESSAGES

REVISION HISTORY

Date	Description
05 July 2005	Document created
06 May 2010	Updated

TERMINOLOGY

WPS	= Wireless Protocol Stack for WAP (WSP/WTP) Gateway connections and WP-HTTP connections
MMSS	= MMS Stack SDK (SDK for receiving and sending MMS Messages, which includes the WPS for transport)
SMSS	= SMS Client SDK (Library for reading/sending SMS messages)
APN	= Access Point Name (GPRS connection setting)
SMIL	= Synchronized Multimedia Integration Language, a markup language for describing multimedia presentations

Table of Contents

1. THIS DOCUMENT

1.1. Overview

This document provides general overview of the MMS Stack SDK from Winwap can be used for sending and receiving/retrieving MMS messages. To understand the contents of this document the reader needs basic knowledge in the mobile technologies and basic programming knowledge.

1.2. Author

The document has been written by Winwap Technologies Oy.

Any questions or comments should be sent to:

Winwap Technologies Oy
Melkonkatu 16 B
00210 Helsinki
FINLAND

Tel. +358-207-661868
Fax. +358-9-6822187
Email: winwap@winwap.com
http://www.winwap.com

2. Winwap MMS Technology Explained

A full MMS Client (*application that can receive and send MMS Messages, and let users write and play MMS messages*) is primarily based on three modules:

1) Wireless Application Protocol Stack (WAP Stack)
2) MMS Stack
3) SMIL media player
4) Host Application UI

Each module provides its specific functionality and together they make up the full MMS Client. However, for carrier interoperability it is significantly important to look at the WAP Stack separately as it provides the core communication functionality with the Telecom carrier servers, such as the WAP Gateway and MMS Center.

2.1. WAP Stack

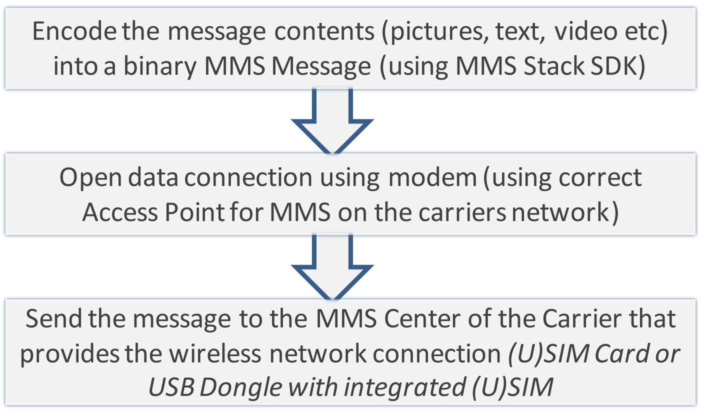
The WAP Stack is the actual transport protocol used for sending and retrieving MMS messages. The WAP Stack rides on top of the wireless broadband technology (GPRS, 3g, CDMA etc.) and is an integral part of any MMS Client and WAP Browsers. Winwap Technologies developed its WAP Protocol Stack in the year 2000 and has kept improving and porting to new platforms since.

2.2. MMS Stack

The MMS Stack provides the message encoding and decoding functionality as well as it also embeds the transport protocol (WAP Stack) to let it send and receive the MMS Messages

3. Sending MMS Messages

This section describes how MMS message sending works, on a mobile phone, PND, DFP or any other device with MMS Capabilities.


Encode the message contents (pictures, text, video etc) into a binary MMS Message (using MMS Stack SDK)
Open data connection using modem (using correct Access Point for MMS on the carriers network)
Send the message to the MMS Center of the Carrier that provides the wireless network connection *(U)SIM Card or USB Dongle with integrated (U)SIM*

3.1. Settings needed for sending MMS messages

To send MMS messages the phone will need to have the following things set up correctly in the phone. Different phones have different ways of setting things up so a phone specific explanation is not provided here. Typically a user does not have to set these manually as most telecom operators simply send them to the user as a SMS (Text) message. However, when using the WinWAP SDK's, these settings must be known and used correctly in order for message sending to work.

3.1.1. GPRS connection Access Point Name

MMS messages are sent over a GPRS (or 3G / CDMA) data connection. As telecom operators often have different ways of charging for different use of GPRS data connections, phones can have many connections in them, and the Access Point Name (APN) tells the telecom operator what kind of connection the phone is using so they can charge the user accordingly. Most telecom operators require that the phone uses a specific APN when sending MMS messages as messages are charged on a per message basis in contrast to other data use that is charged on a per byte (amount of data transferred) basis.

3.1.2. WAP Gateway / Proxy

In order to send the MMS messages the phone needs to connect to a WAP Gateway, or WAP Proxy, that sits in between the phone and the MMS Centre that actually receives the message. The WAP Gateway/Proxy used for MMS messages can be different then the one used for other data connections (WAP or Internet connections), so the phone must have the correct WAP Gateway/Proxy settings in place for MMS messages.

3.1.3. MMS Centre Settings

The MMS Centre (MMSC) is the server that actually receives the message sent by the phone. The MMSC then forwards the message to the recipient's phone.

3.2. Step by step description of sending messages

This section will identify the steps that should be taken to create and send a MMS Message.

3.2.1. Step 1 – Create the message

You must know where any images or other content you want to add to the message is, and then use the MMSS methods to create an internal message with all the content that should be sent with that message. After that the message should be encoded using the MMS methods before it can be sent anywhere. This can be compared to a real phone where you use the phones interface to write a new MMS message and add any images/content that you want to be sent with the message. Encoding is done internally by the phone.

3.2.2. Step 2 – Open the connection

Before you can send the message you must open a dial-up connection (RAS connection) using a 3G Modem (e.g. USB Dongle modem or similar modem embedded with platform) that is connected to the device/computer. The modem must be connected to the computer using a cable or some wireless connection and provide a USB or Serial port interface. When opening the data connection it MUST use the correct APN (Access Point) for MMS on that particular Network. See section 2.2.2 for more information on opening the data connection.

When the data connection has been opened you use the methods of the MMSS to connect to the WAP Gateway/Proxy.

Opening the connection is not a part of the MMS SDK. The GPRS/3G/CDMA data connection should be set up before you attempt to send MMS messages.

The data connection (GPRS, 3G or similar) can be set up via AT commands or using a PPPD program on Linux (it is preferable, since it tunes routing and DNS servers). Proper APN should be set, otherwise MMS Center, where the MMS is located, cannot be reached.

The example shell script below demonstrates how GPRS with APN "cmwap" can be set up. Additionally, old routing and DNS servers are stored before GPRS is set up and are returned after GPRS is shut down.

Example shell script for data connection on Linux:

```
#!/bin/sh
DIALTIMEOUT=20

MODEM=ttyUSB0
SPEED=115200

MODEM_INIT='AT+CGDCONT=1,\"IP\",\"cmwap\" OK'
IH_IP=" ipcp-accept-local ipcp-accept-remote noipdefault
        debug usepeerdns user wap mru 576
        novj nobsdcomp novjccomp nopcomp noaccomp"
LOGSCRIPT="CONNECT"
PHONE="*99***1#"

DR=`route -n | egrep '^0.0.0.0'| grep -v ppp | sed 's/^[^ ]*  *\([^ ]*\) .*/default
gw \1/'` ;
if [ -n "$DR" ]
then
  trap "echo route add $DR ; route add $DR ; exit"  2 3 9 15
  route delete $DR
fi

while  true
do
  pppd \
         connect 'chat -v ABORT "NO DIALTONE" ABORT "NO CARRIER" ABORT BUSY ""
'"$MODEM_INIT"' ATDP'$PHONE' '"$LOGSCRIPT"' ;' \
         crtscts defaultroute modem -detach mru 576 \
         $NASH_IP:$IH_IP /dev/$MODEM $SPEED
  echo $?
    cat /etc/ppp/resolv.conf | tee /etc/resolv.conf
    sleep $DIALTIMEOUT
done
```

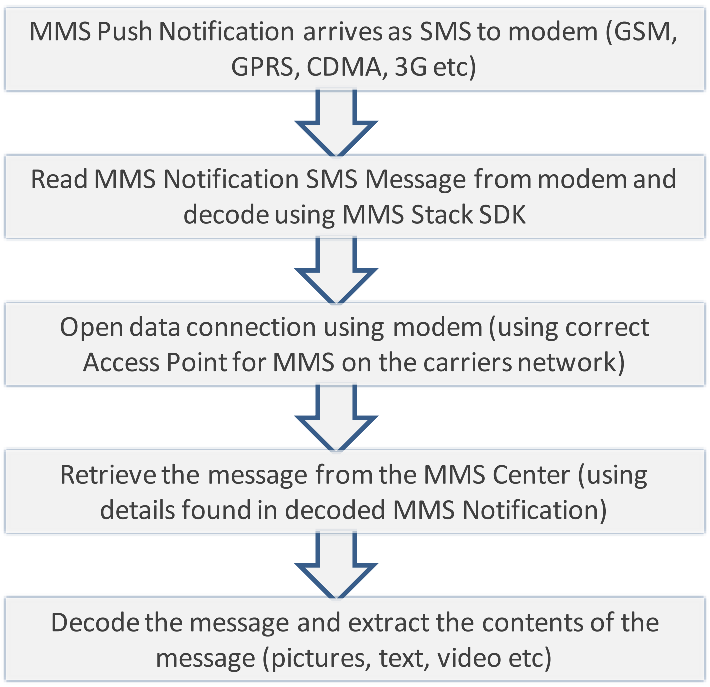
3.2.3. Step 3 – Send the message, and close connections

Once the above steps have been taken, you use the methods of the MMSS to actually send the message to the MMSC. Once the MMSS has returned a successful code for sending you can disconnect from the WAP Gateway/Proxy and close the data connection.

4. Receiving MMS messages

This section will describe how to receive MMS messages. In actuality MMS messages aren't pushed to the phone in the same way as SMS messages are. Only a notification message is pushed to the phone over SMS (or WAP Push in some rare cases), and then the phone must retrieve the actual message itself.

This simple diagram shows the order of events when receiving a MMS Message.


MMS Push Notification arrives as SMS to modem (GSM, GPRS, CDMA, 3G etc)
Read MMS Notification SMS Message from modem and decode using MMS Stack SDK
Open data connection using modem (using correct Access Point for MMS on the carriers network)
Retrieve the message from the MMS Center (using details found in decoded MMS Notification)
Decode the message and extract the contents of the message (pictures, text, video etc)

4.1. Settings needed for receiving MMS messages

To receive MMS messages the phone will need to have the following things set up correctly in the phone. Different phones have different ways of setting things up so a phone specific explanation is not provided here. Typically a user does not have to set these manually as most telecom operators simply send them to the user as a SMS (Text) message. However, when using the WinWAP SDK's, these settings must be known and used correctly in order for message receiving to work. Further, you need access to the SMS messages of a phone or GPRS modem box.

4.1.1. GPRS connection Access Point Name

MMS messages are retrieved over a GPRS (or 3G / CDMA) data connection. As telecom operators often have different ways of charging for different use of GPRS data connections, phones can have many connections in them, and the Access Point Name (APN) tells the telecom operator what kind of connection the phone is using so they can charge the user accordingly. Most telecom operators require that the phone uses a specific APN for retrieving MMS messages as retrieving messages typically is free of charge, but other use of data connections has a per byte fee (per amount of data transferred).

4.1.2. WAP Gateway / Proxy

In order to retrieve the MMS messages the phone needs to connect to a WAP Gateway, or WAP Proxy, that sits in between the phone and the web server from where the actual message should be retrieved. The WAP Gateway/Proxy used for MMS messages can be different then the one used for other data connections (WAP or Internet connections), so the phone must have the correct WAP Gateway/Proxy settings in place for MMS messages.

4.1.3. MMS Notification message

The MMS Notification SMS message that is sent to the phone to let it know that a new message has arrived, contains a http:// address from where the actual message should be retrieved.

4.2. Step by step description of receiving MMS messages

To receive MMS messages the following steps have to take place.

4.2.1. Step 1 – Receive the MMS Notification message using PushFilter

To receive the MMS Notification message you need to connect a GPRS Modem (Device) to the computer using a cable or some wireless connection. You should on your computer have a COM port over which you can communicate with the device. When the connection is set up, you use the MMS Stack SDK to receive a MMS Notification message which is delivered over SMS (a.k.a. text message).

Winwap provides a sample program called `Pushfilter`, which demonstrates how incoming push messages encapsulated in SMS messages can be processed. It listens to the USB/serial port and waits for incoming SMS messages.

It uses the MMS Stack API method `serial_listen()` to start listening and `serial_stop_listen()` on the application exit to stop listening. The serial listener calls a callback function `serial_callback()` passed through the `serial_listen()`.

The function processes data received in different ways, depending on data type:

a) **"push"** indicates push message is received. If it's content type is "application/vnd.wap.mms-message", data bytes are saved to /var/spool/mms/incoming/<Epoch-time>.<milliseconds>.mms.
b) **"raw data"** indicates unknown data. Generally it is data generated by a SIM card or modem. It is ignored.
c) **"sms"** indicates SMS message is received. It is also ignored, but pushfilter source file contains sample in comments, how its data can be sent via UDP socket to another application.

4.2.2. Step 2 – Retrieve and Decode the MMS Notification message

The MMS Notification message arrives in a binary format, so it must be decoded in order to get access to the information about from where the actual message should be retrieved. The MMS Stack SDK contains methods for decoding the MMS Notification message.

AT commands are used to set up the modem for receiving MMS notifications.

`"AT E0"` switches echo off. It is optional.
`"AT+CMGF=0"` enables ability to get SMS text datagrams. It is set by default, the command is required.
`"AT+CNMI=1,1,0,1,0"` enables SMS notifications via a serial port. The value to use depends on the modem, but `"1,1,0,1,0"` is commonly used.

After the last command, the notifications will be received like `+CMTI: "SM",<index>` (message arrival indication), `<index>` indicates the position in a (U)SIM message storage. As one SMS message does not include enough data bytes to store the entire MMS notification, the MMS notifications are received in two sequential SMS messages.

SMS message datagram could be read with a command `"AT+CMGR=<index>"`.

Unread SMS messages stored in a (U)SIM message storage could be listed with a command `"AT=CMGL"` or `"AT+CMGL=0"`. It lists SMS messages as shown here:

```
    +CMGL: <index1>,0,0,<length1>
    <SMSPDU>
    +CMGL: <index2>,0,0,<length2>
    <SMSPDU>
  ...
```

A complete SMS message's datagram `<SMSPDU>` can be received with a command `"AT+CMGR=<index>"`.

Example of SMS PDUs:

```
07919730071101F24403B925F1000440906131812161 8C0C05040B8423F00804DF8602012306396170706
C69636174696F6E2F766E642E7761702E6D6D732D6D65737361676500AF84456E636F64696E672D566572
```

```
73696F6E00312E3400B4878C8298353836373537340 08D908918802B37393033393338313537372F54595
0453D504C4D4E00967465737400 8A808E01548805810302A2FF83687474703A2F2F6D6D
```

```
07919730071101F24403B925F100044090613181316123 0C05040B8423F00804DF860202732F3F6D65737
36167652D69643D3538363735373400
```

When the two SMS message PDUs are retrieved, like below

```
07919730071101F24403B925F1000440906131812161 8C0C05040B8423F00804DF86020123063961707 06
C69636174696F6E2F766E642E7761702E6D6D732D6D65737361676500AF84456E636F64696E672D566572
73696F6E00312E3400B4878C82983538363735373400 8D908918802B37393033393338313537372F54595
0453D504C4D4E00967465737400 8A808E01548805810302A2FF83687474703A2F2F6D6D
```

```
07919730071101F24403B925F1000440906131813161230C05040B8423F00804DF860202732F3F6D65737
36167652D69643D3538363735373400
```

The first PDU should be passed to the `mms_decode_sms_message()`. The functions returns `MMS_ERR_MORE_DATA`, that indicates the second PDU should be passed to `mms_decode_sms_message()`. After the second PDU is processed and `MMS_ERR_SUCCESS` is returned, decoded MMS notification is returned via the `msg` parameter.

The functions `mms_get_header_str()`, `mms_get_header_encstr()`, `mms_get_header_str_array()` and `mms_get_header_encstr_array()` can be used for getting information like sender phone number, subject etc., and the main information is the Content-Location URL, indicating where the MMS message should be downloaded from.

4.2.3. Step 3 – Open the data connection

Before you can retrieve the message you must open a dial-up connection (RAS connection) using a 3G Modem (e.g. USB Dongle modem or similar modem embedded with platform) that is connected to the device/computer. The modem must be connected to the computer using a cable or some wireless connection and provide a USB or Serial port interface. When opening the data connection it MUST use the correct APN (Access Point) for MMS on that particular Network. See section 2.2.2 for more information on opening the data connection.

When the data connection has been opened you use the methods of the MMSS to connect to the WAP Gateway/Proxy.

Opening the connection is not a part of the MMS SDK. The GPRS/3G/CDMA data connection should be set up before you attempt MMS downloading from indicated Content-Location URL.

The data connection (GPRS, 3G or similar) can be set up via AT commands or using a PPPD program on Linux (it is preferable, since it tunes routing and DNS servers). Proper APN should be set, otherwise MMS Center, where the MMS is located, cannot be reached.

The example shell script below demonstrates, how GPRS with APN "cmwap" can be set up. Additionally, old routing and DNS servers are stored before GPRS is set up and are returned after GPRS is shut down.

Example shell script for data connection on Linux:

```
#!/bin/sh
DIALTIMEOUT=20

MODEM=ttyUSB0
SPEED=115200

MODEM_INIT='AT+CGDCONT=1,\"IP\",\"cmwap\" OK'
IH_IP=" ipcp-accept-local ipcp-accept-remote noipdefault
        debug usepeerdns user wap mru 576
        novj nobsdcomp novjccomp nopcomp noaccomp"
LOGSCRIPT="CONNECT"
PHONE="*99***1#"

DR=`route -n | egrep '^0.0.0.0'| grep -v ppp | sed 's/^[^ ]*  *\([^ ]*\) .*/default
gw \1/'` ;
if [ -n "$DR" ]
then
  trap "echo route add $DR ; route add $DR ; exit"  2 3 9 15
  route delete $DR
fi

while  true
do
  pppd \
         connect 'chat -v ABORT "NO DIALTONE" ABORT "NO CARRIER" ABORT BUSY ""
'"$MODEM_INIT"' ATDP'$PHONE' '"$LOGSCRIPT"' ;' \
         crtscts defaultroute modem -detach mru 576 \
         $NASH_IP:$IH_IP /dev/$MODEM $SPEED
  echo $?
    cat /etc/ppp/resolv.conf | tee /etc/resolv.conf
    sleep $DIALTIMEOUT
done
```

4.2.4. Step 4 – Retrieve the message

Once you have opened the data connection you must with the information (Content-Location URL) provided in the MMS Notification message use the methods of MMSS to retrieve the actual message from the MMSC (web server where the message is found).

How to download the message is demonstrated in the `"mmsget"` sample that comes with the MMS Stack SDK. It waits for new files in the `"/var/spool/mms/incoming/"` directory `using inotify Linux subsystem`. When a new file containing MMS notification is found, it extracts Content-Location using the mms_get_header_str() API function and downloads MMS message from a MMS center using the synchronous `mmss_retrieve_message()` API function. The data connection should be enabled and set up before `mss_retrieve_message()` is called.

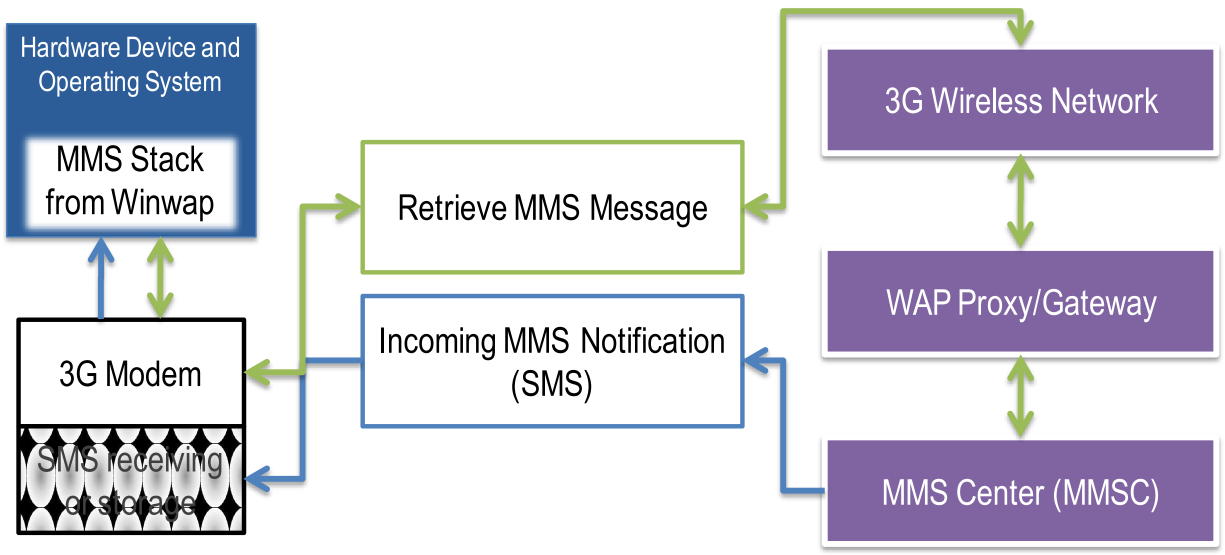
4.2.5. Step 5 – Decode and extract content

Once you have received the MMS Message you must use the methods of the MMSS to decode the message so you have access to the different parts of the message. The message can contain any kind of data and you can extract the data into files on your computer so your own host application, or other applications, can access and use the data.

The mmsget in the sample application saves the downloaded MMS message bytes and its extracted contents/attachments into the `"/var/spool/mms/incoming/<Epoch-time>.<milliseconds>/"` directory. The filename of the content are extracted from Content-ID or Content-Disposition headers of the content. Content is extracted from MMS bytes using the `mms_get_content()` API function. First content is a SMIL presentation file in common cases, other contents/attachments are files, which are referenced from a SMIL.

5. GENERAL INFORMATION ABOUT THE CONNECTION

In the beneath diagram you can see the general schematics of how the different devices and servers are interconnected.



The device (Smartphone, PND, DPF or other platform) uses the GPRS/3G/CDMA/etc. modem in order receive the incoming MMS Notification (pushed to the modem over SMS). Then the device opens wireless the data connection using the modem and retrieves the actual MMS message from the MMS Center over the wireless connection.