InteleSMS AS
MMS-gateway Implementation Guide

English version

Author: Ronny Reinertsen <ronny@intelesms.no>

Last updated: 2011-12-19

InteleSMS AS
Mail: support@intelesms.no
Phone: +47 815 68668
Cell phone: +47 975 13 609
Fax: +47 815 22 170
Web: www.intelesms.no

Table of Contents

1. Introduction

This document describes the services on the InteleSMS MMS-platform and how to implement the API. Basic knowledge of HTTP and SOAP protocol is required.

Below is a definition of general terms used in the document.

Term	Definition
API	Application Programming Interface
Bulk-route	One of many internal routes for sending MMS MT messages.
Customer	The company that has integrated a mobile content service with the supplier's infrastructure.
DR	Delivery Report that describes the success rate of a MMS message sent to End-Users.
End-user	A mobile subscriber that uses the Customer's mobile content service.
Gateway	The short-number used to send mobile content.
Keyword	The first word or character in a MMS MO message or subject. Used to determine what service an End-User is sending message to, or to identify who will receive the MMS MO message.
Mobile content service	Content delivered to the end-user by MMS.
MO	Mobile originated MMS message. In this case the End-User sends a MMS message to a Short-number.
MSISDN	Mobile Station International Subscriber Directory Number
MT	Mobile terminated MMS message. In this case the End-user is the recipient of a MMS message.
Operator	A company that provides services for mobile phone subscribers.
Platform	InteleSMS' technical platform for connecting to the Operators' mobile network to distribute mobile content to End-users.
Service	SOAP Web Service, web script or software that handles messages.
Short-number	A short-number is a 4-5 digit access number (e.g. 1933).
MMS	Multimedia Message Service
Supplier	InteleSMS AS, the supplier of the platform and the services.

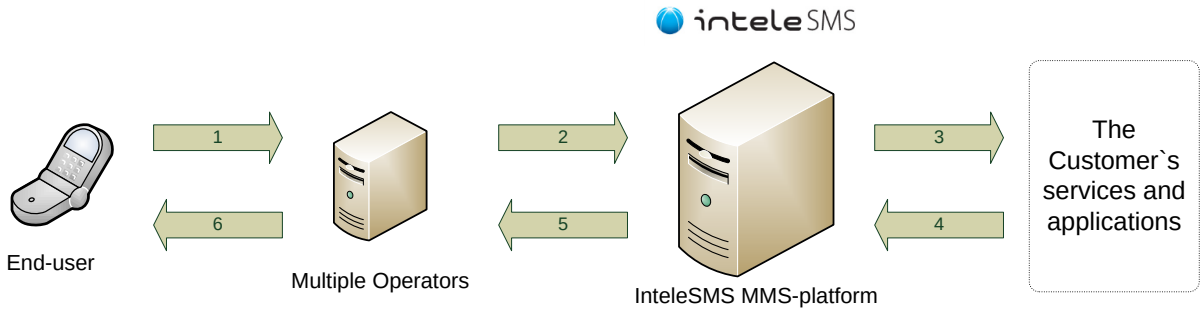
2. Functional overview

The following section will describe and provide a quick overview over the MMS-platforms main functions, products, service locations and the message flow between the End-user and the Mobile content service. In addition you will get an exact illustration of the message flow between the involved parties.

2.1. Main functions

The MMS-platform supports:

- Sending mobile terminated (MT) multimedia messages to End-user
- Receiving mobile originated (MO) multimedia messages sent by End-user
- Receiving delivery report (DR) messages for all submitted MT messages
- Charging the End-user for each received MT message



The figure above provides an overview of the traffic flow between the End-user and the Customer. The message flow can be initiated both by the End-user (Mobile Originating) and by the Customer (Mobile Terminating).

1. The End-user sends a MMS-message to InteleSMS by using an assigned short-number
2. InteleSMS routes the MMS-message to the Customer based on defined rules, e.g. an assigned keyword or a short-number
3. InteleSMS forwards the message to the Customer using a HTTP POST call, or by sending the MMS-message to an email
4. The Customer sends a MMS-message to the End-user by specifying the mobile identity (MsIsdn) as the recipient, assigns the price that should be charged the End-user and the short-number the message should be routed trough. The message is then submitted to the platform-interface with a SOAP service call.
5. InteleSMS validates the Customer's request and selects the best route for the message based on the charge price and the specified short-number. The message is then forwarded to the Operator
6. The Operator delivers the message to the End-user.

2.2. MMS-products

There are two main MMS-products:

- MMS Content supports to send and receive messages and to charge the End-user by MMS.
- MMS Bulk supports sending messages to any international phone number. The Customer will be charged for the message according to the MMS-bulk agreement. There is no charge to the End-user

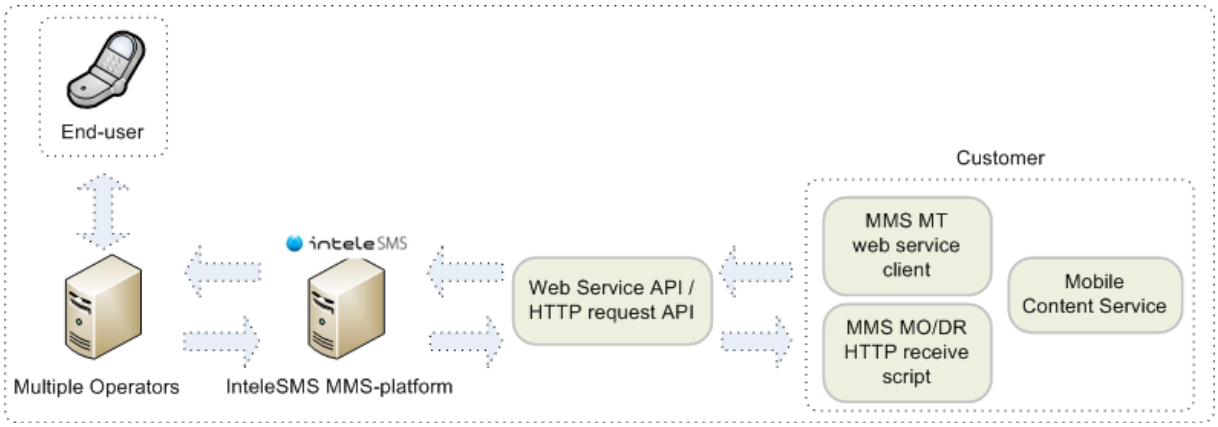
For more information on the MMS-products, see www.intelesms.no.

2.3. Service architecture

The MMS-platform uses a set of SOAP-compliant web services and HTTP POST requests to offer the MMS-services.

In order to send MMS MT messages, the Customer must provide a client-side implementation of the MT web-service. Standard SOAP-compliant toolkits can be used to generate clients, thus no client library is necessary. The platform provides the server-side implementation of the MMS web-service that forwards the received MT message to the Operator or MMS bulk-vendor.

In order to receive MMS MO and DR-messages, the Customer must provide a server-side implementation of the MMS MO/DR HTTP POST request based on the documentation provided.



The platform will put any received MT message in a queue before the message is delivered to the Operator or bulk vendor. The message is forwarded only if the message complies with the implemented formatting policy.
For each MMS MT message sent by the Customer, a corresponding delivery report will be returned. The delivery report contains information about the delivery status of a single message. The Customer must keep track of received delivery reports in order to determine which messages that have been delivered to the mobile subscriber.

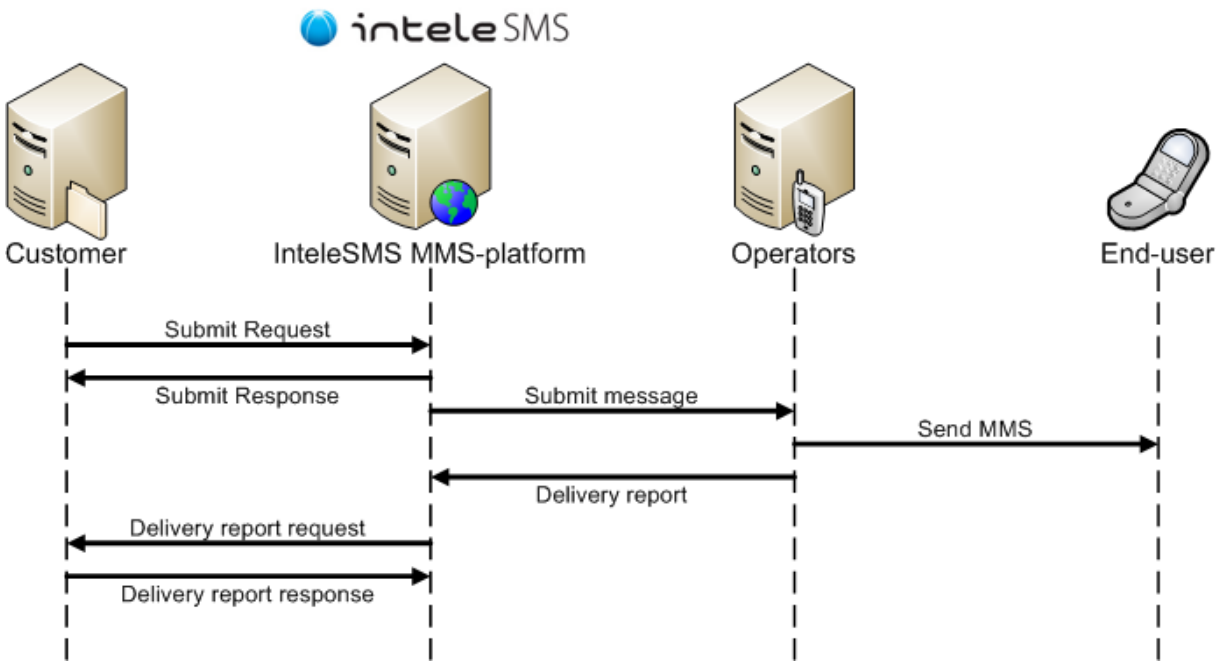
2.4. Message flow

The MMS-platform consists of a pair of request / response messages sent to / from the Customer and the platform. The following sections describe the message transmission involved in sending and receiving MMS-messages and Delivery Reports.

2.4.1. MMS mobile terminating (MT) message flow

The first part of the sequence below shows the Customer initiating a Submit message exchange in order to send a MMS MT message to the MMS-platform. If the Submit request is valid, InteleSMS puts the received message in a queue before the Submit response message. The MMS-platform forwards any received message to the Operator or bulk-route, which finally delivers the message to the End-user.

Once the End-user receives the MMS MT message, a delivery report is returned to InteleSMS. InteleSMS thereafter transmit the delivery report back to the Customer, using a HTTP request.



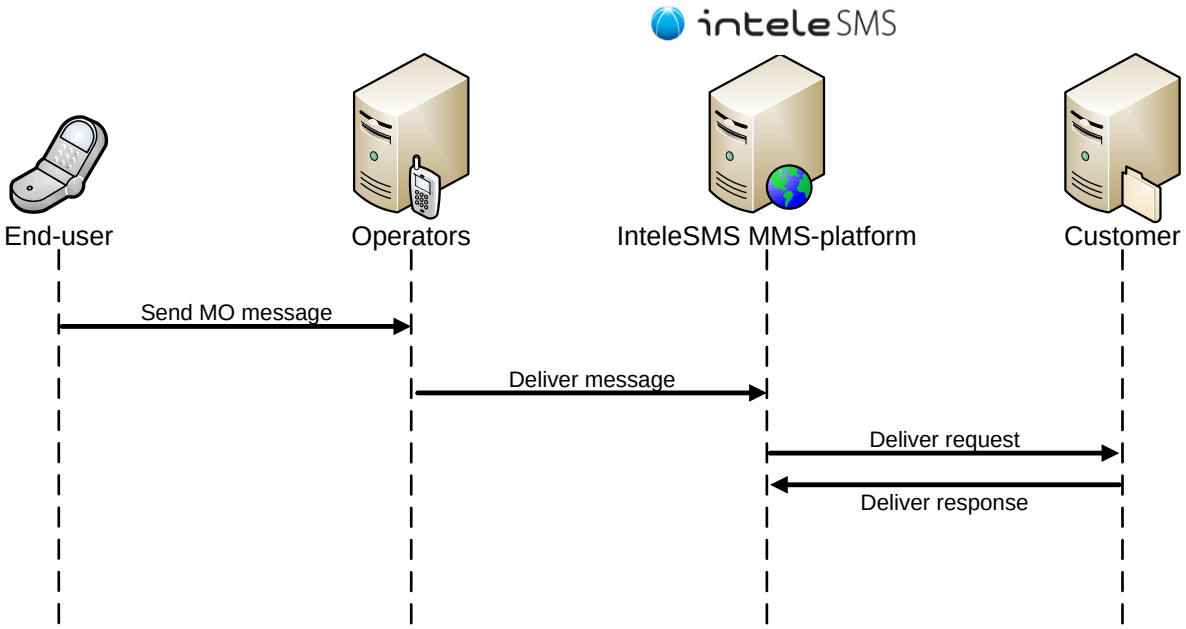
If the Submit message violates the policy or the End-user's mobile subscription is restricted, the Submit response contain an error description and the message is not delivered to the End-user. The Customer must keep track of such errors to know if the message is queued for delivery or if the message is rejected by the MMS-platform.

2.4.2. MMS mobile originating (MO) message flow

The next sequence illustrates how a MMS-message sent from an End-user is routed from Operators to the Customer through the MMS-platform.

An End-user sends a MMS-message to a specified short number. The message is received by a mobile Operator and then forwarded to the InteleSMS MMS-platform. Depending on how the message is configured, the platform will attempt a deliver request to the Customer. The Customer then must return a correct delivery response if the message has been successfully received.

If the MMS-platform is unable to deliver the MMS MO message to the Customer, it attempts to deliver the message again after a pre-configured time-interval and a maximum number for retries.



3. Installation

This section describes the necessary steps to get started with your MMS-service, including a detailed description of number formats, message formats and properties used in messages.

3.1. Gateway agreement

Access to the MMS-platform requires a Gateway agreement with InteleSMS AS. For more information see www.intelesms.no.

3.2. System requirements

To implement the MMS-platform a Customer needs the following:

- SOAP 1.2-compliant Software Development Kit (SDK)
- Web server with fixed IP address.

The InteleSMS platform MMS-API provides a SOAP-based web service interface which is compliant with WS-I Basic Profile 1.0 recommendations. As such, the web service interface should be compliant with most commonly used SOAP toolkits. There is also a set of HTTP requests for delivery of MMS (MO) messages and Delivery reports (DR) that use the standard RFC1945 – Hypertext Transfer Protocol-HTTP/1.0.

The web service API and HTTP requests are verified against the following platforms:

- Microsoft .NET Framework version 1.1, 2.0, 3.0, 3.5 and 4.0
- Apache web server version 1.3.4 and 2.2.14
- PHP version 5.0 – 5.3.1
- PHP version 4.x with Pear SOAP version 0.81

3.3. Web service URL

The MMS (MT) web service definition (WSDL):
https://smsgw.intelesms.no/mms/scriptv2.asmx?WSDL.

The MMS (MT) web service:
https://smsgw.intelesms.no/mms/scriptv2.asmx

Hostname for testing is test.smsgw.intelesms.no. The gateway platform will validate data and conduct all normal checks, but the last step to queue the message for delivery is not called.

3.4. Addressing and number formats

The MMS-platform routes MMS-messages between mobile Operators and bulk-routes. InteleSMS provides access for short-numbers. The Customer can use a shared short-number or get its own dedicated number. If using a shared number, the message is routed by keywords in the MMS-message text or subject.

Each message sent from an End-user routed through the MMS-platform contains the following addressing information:

- gateway – This identifies the recipient of the MMS-message. (e.g. 1933 or 4799700999)
- from_number – This identifies the dispatcher of a MMS-message (e.g. 4712345678)
- keyword – This identifies the service and/or the Customer that the message should be forwarded to.

Each message sent from the Customer to an End-user contains the following information:

- Gateway – This is used to route messages to an account, a short-number or a bulk-route
- Recipient – This is used to specify the mobile subscriber's identity (MsIsdn). The recipient of the message
- FromNumber – This is used to specify the originating address. (E.g. CoolService or +4712345678). This value can be an international MsIsdn, or an alphanumeric value.

When using alphanumeric values in the FromNumber it can be a maximum of 11 characters. Some characters are restricted and the message must be a bulk-MMS with Price parameter set to 0. Premium MMS sent through a short-number can only have the short-number for the specified Gateway as FromNumber. If the FromNumber is blank the value will default to the Gateway value specified in the Submit request.

3.5. Submit messages

MMS MT messages are sent using a Send/SendResponse message exchange with the SOAP interface. This section demonstrates a Send/ SendResponse exchange with the SOAP API.

Bellow you find an encoded MMS MT SOAP envelope header and body according to SOAP 1.2 specification.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <soap:Header>
            <Authorizer xmlns="intelesms.services">
            </Authorizer>
      </soap:Header>
      <soap:Body>
            <Send xmlns="intelesms.services">
            </Send>
      </soap:Body>
</soap:Envelope>
```

3.5.1. Submit authentication

The attached XML fragment describes the authorization SOAP header for a Submit request. There must be an Authorizer SOAP header and the CustomerID and Password must be correct in order to successfully send a MMS MO message. The SubCustomerID parameter should always have the value of 0.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <soap:Header>
            <Authorizer xmlns="intelesms.services">
                  <CustomerID>999</CustomerID>
                  <SubCustomerID>0</SubCustomerID>
                  <Password>MySecret</Password>
            </Authorizer>
      </soap:Header>
      <soap:Body>
      </soap:Body>
</soap:Envelope>
```

3.5.2. Submit request

The following XML fragment illustrates a basic MMS Message sent from the Customer to an End-user with the mobile number 4712345678, where the End-user is charged NOK 5.

```
<Send xmlns="intelesms.services">
      <mms>
            <Gateway>1933</Gateway>
            <Subject>Test message</Subject>
            <Recipient>4712345678</Recipient>
            <Price>500</Price>
            <ServiceID>0</ServiceID>
            <ValidationPeriod>168</ValidationPeriod>
            <MessagePriority>Low</MessagePriority>
            <MessageID>MMS_1</MessageID>
            <AgeLimit>0</AgeLimit>
            <BillingDescription />
            <BillingCategory />
            <Content>
                  <File>base64 encoded binary data</File>
                  <FileName>Test.txt</FileName>
            </Content>
```

```
            <ContentType>Raw</ContentType>
        </mms>
</Send>
```

In the table below you will find all elements available in the SendMMS object.

Element	Comment
Gateway	Mandatory – The short-number or bulk route number used to send the message.
Recipient	Mandatory – The recipient of the message. Supports only “number” formats.
FromNumber	Optional – Originating address as displayed in the mobile terminal. By default the Gateway value is displayed as originating address. The following formats are supported: • 4- or 5-digit short-number. • Mobile subscriber number. International MsIsdn. • Alphanumeric value. Max length is 11 characters where characters can be one of the following: • English characters and digits [a-zA-Z0-9] • Space, Percentage (%), Ampersand (&), Minus (-), Plus (+), Period (.) and Comma (,).
Price	Mandatory – End-user message price. See the Appendix C for what values that is available. For bulk-MMS set price to 0.
Subject	Optional - Specifies the subject for the message or the content. Maximum of 128 characters are allowed.
ValidationPeriod	Optional – Specifies the maximum time span a message should be resent before the message expires. If the message expires a Delivery report (DR) with the corresponding error code is generated. The value can be either relative or absolute. For an absolute value the timestamp format is ddMMyyHHmm. A relative value is a number from 0 to 255, where the representation is as follows: 0 to 143 – (Value + 1) x 5 minutes 144 to 167 – 12h + (Value-143) x 30 minutes 168 to 173 – (Value-166) x 1 day 174 to 255 – (Value-172) x 1 week The default value has the relative value of 168. Please note that most mobile Operators have a max timeout of 1 week.
MessageID	Optional – A string value that must be unique for each message. The client reference is used to track submitted messages and correlate any delivery report messages. If this value is not set, you cannot track if a message is successfully delivered. Max length is 32 characters.
ServiceID	Reserved. Set the value to 0
MessagePriority	Optional – Set the priority for the message. Valid values are Low, Normal or High.
AgeLimit	Optional - Specify minimum age limit for the content in the MMS-message. Valid values are 16 or 18. The value denotes that the mobile subscriber must be at least 16 or 18 years old to receive the content.
BillingDescription	Optional - Description of the charged content. Only used when the End-user is charged. Maximum

	80 characters. Not all operators supports this setting.
BillingCategory	Optional – Used for grouping messages together to have more control of what service, service center, help-desk or other defined instance that have sent the messages. The messages will be categorized and summarized under each category group at the monthly settlement from InteleSMS. The string can include the following characters: [0-9a-zæøåA-ZÆØÅ], Period (.), Minus (-), Slash (/), Backslash (\) and Underscore (_). The maximum number of characters is 30.
Content	Add content to the MMS message in this section. Each file that is sent should have the {object}.File to the base 64 encoded data and a valid filename. If the ContentType is set to Raw, the MMS-platform will check each file and compress the files into a single zip-file before the messages is forwarded to the operator or bulk-route. While using ContentType Raw you can add multiple files in the Content Section. If using ContentType Zip, the MMS-platform will accept only one compressed zip-file in the Content section. The zip-file can contain multiple files inside. In each case the raw content cannot exceed 300Kb in size. This is a maximum limit set by the mobile operators as of April 2010
ContentType	Specify how the content is sent. Valid values are Raw or Zip

3.5.3. Simple message

The following XML fragment describes a simple MMS MT message sent to the End-user using ContentType Raw and we're added two files to the Content section.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <soap:Header>
            <Authorizer xmlns="intelesms.services">
                  <CustomerID>999</CustomerID>
                  <SubCustomerID>0</SubCustomerID>
                  <Password>MySecret</Password>
            </Authorizer>
      </soap:Header>
      <soap:Body>
            <Send xmlns="intelesms.services">
                  <mms>
                        <Gateway>1933</Gateway>
                        <Subject>Test message</Subject>
                        <Recipient>4712345678</Recipient>
                        <Price>500</Price>
                        <ServiceID>0</ServiceID>
                        <ValidationPeriod>168</ValidationPeriod>
                        <MessagePriority>Low</MessagePriority>
                        <MessageID>MMS_1</MessageID>
                        <AgeLimit>0</AgeLimit>
                        <BillingDescription />
                        <BillingCategory />
                        <Content>
                              <File>base64 encoded binary data</File>
                              <FileName>this_is_me.jpg</FileName>
                        </Content>
                        <Content>
                              <File>base64 encoded binary data</File>
                              <FileName>About.txt</FileName>
                        </Content>
```

```
                        <ContentType>Raw</ContentType>
                  </mms>
            </Send>
      </soap:Body>
</soap:Envelope>
```

The next XML fragment describes a simple MMS MT message sent to the End-user using ContentType Zip

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <soap:Header>
            <Authorizer xmlns="intelesms.services">
                  <CustomerID>999</CustomerID>
                  <SubCustomerID>0</SubCustomerID>
                  <Password>MySecret</Password>
            </Authorizer>
      </soap:Header>
      <soap:Body>
            <Send xmlns="intelesms.services">
                  <mms>
                        <Gateway>1933</Gateway>
                        <Subject>Test message</Subject>
                        <Recipient>4712345678</Recipient>
                        <Price>500</Price>
                        <ServiceID>0</ServiceID>
                        <ValidationPeriod>168</ValidationPeriod>
                        <MessagePriority>Low</MessagePriority>
                        <MessageID>MMS_1</MessageID>
                        <AgeLimit>0</AgeLimit>
                        <BillingDescription />
                        <BillingCategory />
                        <Content>
                              <File>base64 encoded binary data</File>
                              <FileName>all_data.zip</FileName>
                        </Content>
                        <ContentType>Zip</ContentType>
                  </mms>
            </Send>
      </soap:Body>
</soap:Envelope>
```

3.5.4. Message delivery and expiry

MMS is a “best-effort” service (meaning that the Operators do not guarantee 100 % delivery to End-users – e.g. the hand-set can be without coverage, switched of, etc.). By default the MMS-platform set a 5-day expiry value for all messages submitted to the gateway if the ValidationPeriod is not present in the submitted MMS MT message. This is the same default as used by the Operators. If it is not possible to deliver the message to the End-user before the expiry value, the MMS-platform creates a delivery report (DR) with the expiry status. A Customer can change the default expiry value. The value can be specified as relative or absolute. Default is the relative value 168 (5 days). To set the correct relative value, use the table below:

0 to 143 – (Value + 1) x 5 minutes
144 to 167 – 12h + (Value-143) x 30 minutes

168 to 173 – (Value-166) x 1 day
174 to 255 – (Value-172) x 1 week

The format of an absolute value is ddMMyyHHmm.

- dd indicates the day
- MM indicates the month
- yy indicates the two-digit year value. 10 for year 2010
- HH indicates the hour
- Mm indicates the minute

Absolute timestamps are specified using local time zones.

Example of using a relative value:

```
<Send xmlns="intelesms.services">
	<mms>
		<Gateway>1933</Gateway>
		<Subject>Test message</Subject>
		<Recipient>4712345678</Recipient>
		<Price>500</Price>
		<ServiceID>0</ServiceID>
		<ValidationPeriod>168</ValidationPeriod>
		<MessagePriority>Low</MessagePriority>
		<MessageID>MMS_1</MessageID>
		<AgeLimit>0</AgeLimit>
		<BillingDescription />
		<BillingCategory />
		<Content>
			<File>base64 encoded binary data</File>
			<FileName>all_data.zip</FileName>
		</Content>
		<ContentType>Zip</ContentType>
	</mms>
</Send>
```

Example of using an absolute value for the timestamp 25.12.2010 10:15:

```
<?xml version="1.0" encoding="utf-8" ?>
<Send xmlns="intelesms.services">
	<mms>
		<Gateway>1933</Gateway>
		<Subject>Test message</Subject>
		<Recipient>4712345678</Recipient>
		<Price>500</Price>
		<ServiceID>0</ServiceID>
		<ValidationPeriod>2512101015</ValidationPeriod>
		<MessagePriority>Low</MessagePriority>
		<MessageID>MMS_1</MessageID>
		<AgeLimit>0</AgeLimit>
		<BillingDescription />
		<BillingCategory />
		<Content>
			<File>base64 encoded binary data</File>
			<FileName>all_data.zip</FileName>
		</Content>
		<ContentType>Zip</ContentType>
	</mms>
</Send>
```

3.5.5. Age Limit

All Customers are responsible for ensuring that content sold is not inappropriate based on the End-users age and the prevailing regulations in the country. The Customer is responsible for classifying their content according to a minimum age. The Operators CPA platform blocks content for End-users that are not compliant with the age limit. If an End-user is under age a Delivery report is generated with the corresponding error code.

The age limit is specified by assigning the minimum age to each MMS MT message. In the following example, the content is only delivered to End-users that are at least 16 years old.

Example:

```
<?xml version="1.0" encoding="utf-8" ?>
<Send xmlns="intelesms.services">
	<mms>
		<Gateway>1933</Gateway>
		<Subject>Test message</Subject>
		<Recipient>4712345678</Recipient>
		<Price>500</Price>
		<ServiceID>0</ServiceID>
		<ValidationPeriod>168</ValidationPeriod>
		<MessagePriority>Low</MessagePriority>
		<MessageID>MMS_1</MessageID>
		<AgeLimit>16</AgeLimit>
		<BillingDescription />
		<BillingCategory />
		<Content>
			<File>base64 encoded binary data</File>
			<FileName>all_data.zip</FileName>
		</Content>
		<ContentType>Zip</ContentType>
	</mms>
</Send>
```

Note: The age limit check uses the End-users birth date, and not the birth year.

3.5.6. SendMMS response

The SendMMS response message depends on whether the received message is accepted by the MMS-platform.

The following XML fragment is returned as a response to a Send request message for a successful received message.

```
<?xml version="1.0" encoding="utf-8" ?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
	<soap:Body>
		<SendResponse xmlns="intelesms.services">
			<SendResult>
				<error>Success</error>
				<errorMessage />
			</SendResult>
		</SendResponse>
	</soap:Body>
</soap:Envelope>
```

Info element	Comment
error	Status of the submission. If the message complies with the implemented policy the status will be success. For a list of error codes, see Appendix B.
errorMessage	Specifies more detailed information about the status of the submission.

When a SendMMS request fails, the client receives different response messages based on the type of errors. The response behavior also depends on the SOAP toolkit used.

- Network problems will be returned to clients as standard HTTP error codes.
- Server errors or SOAP validation errors will result in a SOAP Fault response.

3.6. Handling delivery reports

MMS DR messages are sent using a HTTP GET message exchange.

3.6.1. Delivery Report request

A callback URL can be configured for each Customer account. The MMS-platform will forward Delivery Reports to the specified URL if the callback service is configured and activated. A Delivery Report request can contain the following values:

Info element	Type	Comment
message_id	String	Specifies the id used when sending MMS MT message.
status	Integer	Specifies the status of the message delivery. Different status codes are as follows. - 128 = Message expired in delivery - 129 = Message successfully delivered to End-user - 130 = The End-users phone is not correctly configured for MMS service. - 131 = Message delivery is temporary delayed - 132 = Unspecified error in delivery - 133 = Message has been forwarded to the operators customer web interface for reading, since the direct delivery to the phone has failed. - 201 = The End-user has been charged for the content, but delivery of the content itself has failed. - 1005 = Not received by End-user due to insufficient balance on pre-paid account - 1052 = Subscriber has requested barring service for one or all of sms, mms and wap content services - 9016 = An error occurred in the charging/billing operation - 10002 = Operator does not support MMS services. - 900,2001,10001 and other codes means an error occurred while sending message from the MMS-platform to the operator or bulk-route. These status codes are also listed in Appendix A
gateway	Long	Specifies the short-number where the message was sent thought.

The parameter names can be customized for the callback URL in the Customer's web administration pages at https://customer.intelesms.no. The Customer can select which of the parameters that are implemented.

3.6.2. Delivery Report response

When the MMS-platform forwards a Delivery Report to the Customer's web-server, the web-server must be set-up to respond with a HTTP header containing the status HTTP 200 OK when the DR is received. If the web-server fails it must respond with a status HTTP 501 Internal Server Error or similar.

If an error occurs or there is a network problem, the MMS-platform will attempt to forward the message after a predefined time interval and a predefined maximum number of retries.

3.6.3. Callback lookup service

For Customers that are prohibited to use the automatic callback function, there is an alternative to get the Delivery Report for the MMS MT messages. By creating a HTTP request against the URL: https://smsgw.intelesms.no/callback/mms.aspx you can check whether messages are received by the End-user.

The request parameters for the callback script are:

Info element	Type	Comment
customer_id	Integer	Specifies the Customer id for your InteleSMS account.
password	String	Specifies the password for your InteleSMS account.
msgid	String	Specifies the unique id for the message you want to check. This id should be the id you provide when sending the MMS MT message (see the MessageID field). To check multiple messages in one request you must separate multiple messages id`s with comma (,).

The URL below demonstrates how to check 3 messages in one request:
https://smsgw.intelesms.no/callback/mms.aspx?customer_id=99999&password=mySecret&msgid=2591802%2C2591798%2C2591796

The server has following XML format:

```
<?xml version="1.0" encoding="utf-8" ?>
<response>
      <msg system_id="" id="" ack="" error="" description=""></msg>
      <message/>
      <error>0</error>
</response>
```

The XML response has multiple msg nodes if there are multiple message ids in the server request. Example of multiple check response:

```
<?xml version="1.0" encoding="utf-8" ?>
<response>
      <msg system_id="68704818" id="591802" ack="129" error="0">1</msg>
      <msg system_id="" id="2591798" ack="0" error="1"
            description="Message not found">0</msg>
      <msg system_id="68704819" id="2591796" ack="1005" error="0">0</msg>
      <message/>
      <error>0</error>
</response>
```

If an error occurs at the server-side processing the request, the XML response message and error elements will represent the error. See the example below of a check response when an internal database error occurs at the remote server:

```
<?xml version="1.0" encoding="utf-8"?>
<response>
  <message>An internal database error occured</message>
  <error>1</error>
</response>
```

The server responses are as follows:

Info element	Type	Comment
response	Root Node	Specifies the root element of the XML structure
message	String	Specifies a description of the error that has occurred.
error	Integer	Specifies if the server has processed the request successfully or not. Value 0 means success. Any other value is an exception. The detailed error information can be found in the message node.
msg	Node	The msg element represents the Delivery Report for the MMS MT message. This XML node has 3 attributes and also an integer value in the InnerText field.
Attributes and values in the msg node:		
id	String	Represents the requested message id where the status are found
system_id	Integer	Represents the unique id for the message in the MMS-platform
ack	Integer	Specifies the delivery status for the MMS MT message. See Appendix A for more information about these values.
error	Integer	Indicates if an error occurred while checking this specific error.
description	String	Optional. If a message is not found in the system, the error attribute value will be 1 and this field will describe why the check failed for that specific message.
InnerText value	Integer	Specifies if the message is successfully delivered or not. A value of 1 indicates that the End-user has received the message. Value 0 indicates that the message has not been delivered.

3.7. Receive messages

The following sections will describe how to receive MMS MO messages, and how to receive Delivery Reports.

3.7.1. Deliver request message

Bellow you find a URL example for a MMS MO HTTP POST deliver request:

http://www.yourserver.com/services/receive_mms.aspx?gateway=1933
&operator_id=4&operator_name=Telenor&from_number=4712345678&country_id=1&id=123456
&customer_id=999&keyword=TEST&sub_keyword=&k_identifier=file&k_file=message.txt&k_string=
TEST

The table shows the default parameters of a MMS MO HTTP POST delivery request. Parameter names can be changed on request, but we recommend using the default parameter names.

Info element	Type	Comment
gateway	Long	Specifies on what short number or mobile gateway number the messages are received.

operator_id	Integer	Specifies an integer identifier for the mobile Operator. If this value is not available it is set to 0. To identify the Operator, use the XML output from this URL: https://smsgw.intelesms.no/operators/ where the operator\id node in the XML output represents the operator_id in the HTTP delivery request.
operator_name	String	Specifies the operator name or an alias for the operator, if available.
from_number	Long	Mandatory – Specifies the mobile subscriber MsIsdn identifier (e.g. 4712345678). This value do not have a corresponding "00" or "+" values. Digits only.
country_id	Integer	Optional - Specifies an integer value that denotes which country the message was received from. To identify the country, use the XML output from this URL: https://smsgw.intelesms.no/info/ where the gateway\country_id in the XML output represents the country_id in the HTTP delivery request.
id	Integer	Optional – Specifies a unique integer identifier message id from the MMS-platform.
customer_id	Integer	Optional – Specifies the Customer id as an integer for the account where the messages are received.
keyword	String	Optional – Specifies on what keyword the received message is routed. This keyword represents the first word of the received MMS-message text or subject. It is only used for shared short. This value is always presented in upper case.
sub_keyword	String	Optional – Specifies the second word in a received MMS-message text or subject. It can be used to conduct actions under for main keyword based on sub keywords. This value is always presented in upper case.
k_identifier	String	Mandatory – Specifies where the identifier keyword is found. The value can be subject or file. The value of subject means that the keyword has been recognized in the subject field of the message. If the value is file, the keyword has been recognized in the text of a file. In this case you'll find the filename where the keyword has been recognized in the k_file parameter described below.
k_file	String	Optional – If the k_identifier value is set to file, then this field will specify where the recognized keyword is to be found.

In the body of the HTTP POST request there will be a field called file. This field contains a compressed zip-file with all the raw files sent from the End-users phone. The filename will have the id of the message including the .zip extension at the end. E.g. 5124587.zip

A full HTTP POST request will look like this (binary data have been replaced by "..binary data.."):

```
POST
/receive_mms.aspx?customer_id=1&gateway=1933&from_number=4712345678&subject
=&operator_id=1&country_id=1&keyword=TEST&sub_keyword=MESSAGE&k_identifier=
file&k_file=GiwXf.txt&k_string=TEST+MESSAGE HTTP/1.1
Connection: close
Content-Length: 32103
Content-Type: multipart/form-data; boundary="----
=intelesms_httpcli_632370257691353235"
Host: www.yourhostname.no
User-Agent: InteleSMS MMS forward client

------=intelesms_httpcli_632370257691353235
Content-Type: application/x-zip-compressed
Content-Disposition: form-data; name="file"; filename="3327.zip"
```

```
..binary data..
------=intelesms_httpcli_632370257691353235—
```

3.7.2. Deliver response message

When the MMS-platform forwards a MMS MO message to a web-server, the web-server must be set up to respond with a HTTP header containing the status HTTP 200 OK if the message is received. If the web-server fails, it must respond with a status HTTP 501 Internal Server Error or similar.

If an error occurs, the MMS-platform continues to forward the message after a predefined time interval and a predefined maximum number of attempts.

3.7.3. Receive MMS MO message by email

As an alternative delivery method you can receive the MMS MO message by email. The forwarded email could either have the zip-file attached and the parameters with values listed in the email body, or you can have a template string to describe how the content should appear in the email. An example of a template string could be like this:

MMS message received with subject [$SUBJECT], sent from [$ORIGINATOR] with the keyword [$KEYWORD_STRING]. See attached files. [$FILES]

The parameters with brackets will be replaced with the original values from the MMS message. The parameter [$FILES] tells the forward engine to extract all raw files from the compressed zip-file into the email body. In case of using a template with the MMS to email forwarder you will get a more readable email.
To set up MMS to email forwarding you will have to contact InteleSMS to create the MMS keyword for this service type. Thereafter you can configure the service from InteleSMS customer web interface. You can edit the auto response SMS text message and the charge price for the response to the End-user when sending a MMS message to this service.

3.7.4. Polling service

This service provides access to all incoming MMS MO messages for your account. This is an alternative way to get MMS MO messages if you cannot use the default automatic forward service (e.g. if you have a dedicated mobile number for receiving messages from several countries, and/or using intranet). The web service is located here: https://smsgw.intelesms.no/fetch/inbox.asmx

The web service has two functions for retrieving MMS messages. GetMMSMessages and GetMMSMessagesByGateway. The function GetMMSMessages takes 2 arguments: LastId and Keyword. The function GetMMSMessagesByGateway takes 3 arguments: GatewayNumber, LastId and Keyword. The SOAP request must also include a SOAP header with account credentials.

The following section describes an encoded polling service SOAP envelope header and a body according to SOAP 1.2 specification.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
      <soap:Header>
            <sHeader xmlns="intelesms.services">
            </sHeader>
      </soap:Header>
```

```
        <soap:Body>
                <GetMMSMessages xmlns="intelesms.services">
                </GetMMSMessages>
                OR
                <GetMMSMessagesByGateway xmlns="intelesms.services">
                </GetMMSMessagesByGateway>
        </soap:Body>
</soap:Envelope>
```

The sHeader section in the SOAP header should at least include these parameters:

- CustomerId – Specifies your InteleSMS account id. This is an Integer field
- Password – Specifies your InteleSMS account password. This is a String field.

* The CustomValues parameter in the SOAP header is reserved for future use only.

The GetMMSMessages section in the SOAP body must have the following parameters:

- LastId – Specifies the last message id so that the service will only give you messages with a higher value. If this value is set to 0, all messages within the last 60 days are read
- Keyword – Specifies a search string for what messages you want to read from the service. The Keyword will match the first word in the MMS MO subject or message text. To read all messages, set this value to a blank string. If you want to read all messages that start with the word “HELLO”, then set the Keyword value to HELLO. The Keyword value is not case-sensitive.

The GetMMSMessagesByGateway section in the SOAP body must have the following parameters:

- GatewayNumber - Used to filter messages that is received on a specific gateway number
- LastId – Specifies the last message id so that the service will only give you messages with a higher value. If this value is set to 0, all messages within the last 60 days are read
- Keyword – Specifies a search string for what messages you want to read from the service. The Keyword will match the first word in the MMS MO subject or message text. To read all messages, set this value to a blank string. If you want to read all messages that start with the word “HELLO”, then set the Keyword value to HELLO. The Keyword value is not case-sensitive.

It is important to keep track of the message id from the SOAP response and use the highest id in the next SOAP request. By doing this you ensure that only new messages are read in the next request. LastId should only be set to value 0 the first time you read messages.

The following section describes a valid encoded polling service request:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
          xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <sHeader xmlns="intelesms.services">
      <CustomerId>999</CustomerId>
      <CustomValues xsi:nil="true" />
      <Password>MySecret</Password>
    </sHeader>
  </soap:Header>
  <soap:Body>
    <GetMMSMessages xmlns="intelesms.services">
      <LastId>1</LastId>
      <Keyword>MITT</Keyword>
```

```
    </GetMMSMessages>
  </soap:Body>
</soap:Envelope>
```

The following section shows an example of a response from the polling service:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
                                xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
                                xmlns:xsd="http://www.w3.org/2001/XMLSchema">
       <soap:Body>
              <GetMMSMessagesResponse xmlns="intelesms.services">
                     <GetMMSMessagesResult>
                            <Status>Success</Status>
                            <StatusDescription />
                            <Messages>
                                   <MMSMessage>
                                          <Id>123456789</Id>
                                          <CustomerId>999</CustomerId>
                                          <Timestamp>2011-01-06T13:50:28.833</Timestamp>
                                          <OperatorId>1</OperatorId>
                                          <Destination>1933</Destination>
                                          <Originator>4799999999</Originator>
                                          <CountryId>1</CountryId>
                                          <Subject>Mitt svar er 5</Subject>
                                          <KeywordString>MITT</KeywordString>
                                          <Filenames>
                                                 <string>message.txt</string>
                                                 <string>pres.smil</string>
                                          </Filenames>
                                          <KeywordIdentifier>Subject</KeywordIdentifier>
                                          <KeywordFile />
                                          <ServiceId>10</ServiceId>
                                          <CompressedContent>
                                          base64data
                                          </CompressedContent>
                                          <CompressedContentFilename>
                                          123456789.zip
                                          </CompressedContentFilename>
                                   </MMSMessage>
                            </Messages>
                     </GetMMSMessagesResult>
              </GetMMSMessagesResponse>
       </soap:Body>
</soap:Envelope>
```

If the request fails, the Status field will have the value Error and the StatusDescription field will contain more details about the error. The following table describes the parameters in the MMSMessage section of the SOAP response:

Info element	Type	Comment
Id	Integer	Specifies the unique message id from the SMS-platform.
CustomerId	Integer	Specifies the InteleSMS customer account id. By default you will only be able to read messages for the account set in the SOAP header.
Timestamp	DateTime	Specifies the timestamp for when the message was received by the SMS-platform.
OperatorId	Integer	Specifies the end-user operator id, if any. If no operator is available, the value will be set to 0. To identify the operator, use the XML output from this URL:

		https://smsgw.intelesms.no/operators/ where the operator\id node in the XML output represents the OperatorId in the SOAP Message section.
Destination	Long	Specifies the recipient of the message. This value could be a short-number or a mobile gateway number.
Originator	Long	Specifies the sender address (MsIsdn) of the mobile subscriber or system.
CountryId	Integer	Specifies the country id where the message was received. To identify the country, use the XML output from this URL: https://smsgw.intelesms.no/info/ where the gateway\country_id in the XML output represents the CountryId in the SOAP Message section.
Subject	String	Represents the Subject value of the MMS MO message. This value can be null.
KeywordString	String	Represents the keyword string if gateway is using keyword check. Or the search value for the Keyword parameter in the service request. This value can be null.
Filenames	String[]	Specifies all the filenames that exists in the CompressedContent file. This is an array of string. This array may have no entries.
KeywordIdentifier	String	Specifies where the keyword was found if gateways is using keyword check. Possible values are: None, Subject or File. This value defaults to None if no keyword match or if the gateway is not using keyword check.
KeywordFile	String	Specifies the filename where the keyword was found if KeywordIdentifier has the value of File. This value can be null.
ServiceId	Integer	Specifies the internal service id used by the SMS-platform. This value should always be 10.
CompressedContent	Byte[]	Specifies the binary data of the compressed file. This would be a zip file, containing all the files sent from the mobile subscriber
CompressedContentFilename	String	Specifies the filename of the CompressedContent. This would normally be the message id and a .zip extension. E.g. 123456789.zip

See also the example code for the polling service client here:
https://kunde.intelesms.no/files/Inbox_Polling_Sample_Client.rar

4. Implementation examples

The following sections will show some examples on how to send MMS MT messages in different programming languages. Complete sample code can be downloaded here:
https://customer.intelesms.no/files/InteleSMS_MMSGatewaySamples.rar

4.1. PHP 5 SOAP client

In order to run the PHP sample, the following components are required:

- PHP release 5.x (http://php.net)
- PHP SOAP extension (http://php.net/soap)
- PHP OpenSSL extension (http://php.net/openssl)
- PHP ZIP extension (http://php.net/zip)

- GNOME XML library (http://www.xmlsoft.org)

Example of sending a MMS MT message using SOAP client from PHP5:

```
<?php

	// Throw exception on all errors.
	set_error_handler(create_function('$x, $y', 'throw new Exception($y, $x);'),
	E_ALL & ~E_NOTICE);

	$ns = "intelesms.services";
	$wsdl = "https://smsgw.intelesms.no/mms/scriptv2.asmx?WSDL";
	// Initialize Soap client
	$client = new SoapClient($wsdl);
	// Set auth soap header
	$header->CustomerID = 999;
	$header->SubCustomerID = 0;
	$header->Password = "MySecret";
	$header = new SoapHeader($ns,"Authorizer", $header);
	// Apply auth header to soap client object
	$client->__setSOAPHeaders(array($header));

	//Remember to add checks to see if the file
	//exists before trying to read from it.

	//add one single file
	$contentFile1 = getFileAsContent("c:/test.jpg");

	// Create mms message object
	$mmsObj = array("Gateway" => 1933,
		"Subject" => "Test message",
		"Recipient" => 4712345678,
		"FromNumber" => "",
		"Price" => 0,
		"ServiceID" => 0,
		"ValidationPeriod" => "168",
		"MessagePriority" => "Normal",
		"MessageID" => "MMS_1",
		"AgeLimit" => "0",
		"BillingDescription" => "",
		"BillingCategory" => "",
		"Content" => $contentFile1,
		"ContentType" => "Raw"
		);

		//To add multiple files you'll have to add
		//multiple Content params to the SOAP message
		//"Content" => $contentFile1,
		//"Content" => $contentFile2,
		//"Content" => $contentFile3,

	try {

		// Make soap call to send message
		$result = $client->Send(array('mms' => $mmsObj) );

		// check result/response from soap service
		if ($result->SendResult->error=="Success") {
			//message successfully stored
			echo "OK";
		} else {
			// failed
			echo "FAILED";
```

```
		}
		//show the result
		echo "<br />";
		echo "Status={$result->SendResult->error}<br />";
		echo "StatusText={$result->SendResult->errorMessage}";

	}
	catch (SoapFault $fault)
	{
		echo "SOAP Fault: (faultcode: {$fault->faultcode},";
		echo  "faultstring: {$fault->faultstring})";
	}

	function getFileAsContent($filename)
	{

		$fh = fopen($filename, "rb");
		$data = fread($fh, filesize($filename));
		fclose($fh);

		return array("File" => $data,
		"FileName" => basename($filename));

	}

?>
```

4.2. .NET Framework SOAP client

In order to run the .NET samples you need to install .NET Framework 2.x or above. You also need Internet information server 5.x or above to run the .NET SOAP client in a web server environment. The following samples show how to send a MMS message from a console application and how to send a MMS message from a .NET web page.

4.2.1. C#

Example 1:
Use the .NET tool WSDL included in the .NET SDK to generate the SOAP client. Alternatively you can add a Web Reference directly in a Visual Studio project.

Example of how to use the wsdl tool:
wsdl /l:CS /n:intelesms https://smsgw.intelesms.no/mms/scriptv2.asmx?WSDL

This command creates an MMSGateway.cs file in the corresponding directory. Import this file into your project or web site. The following sample is a console application which uses the generated SOAP client from the previous step:

```
using System;
using System.IO;

namespace SendMMSCSharp
{
class Program
{
	static void Main(string[] args)
	{

		//initialize SOAP client
```

```
            intelesms.MMSGateway client = new intelesms.MMSGateway();

            //create authorize object
            intelesms.Authorizer auth = new intelesms.Authorizer();
            auth.CustomerID = 999;
            auth.SubCustomerID = 0;
            auth.Password = "MySecret";

            //apply authorizer object to client
            client.AuthorizerValue = auth;

            //initialize the MMS message object
            intelesms.MMSMessage msgObj = new intelesms.MMSMessage();
            msgObj.Recipient = "4712345678";
            msgObj.Gateway = 1933;
            msgObj.MessageID = "MMS_1";
            msgObj.Price = 0;
            msgObj.MessagePriority = intelesms.Priority.Normal;
            msgObj.BillingCategory = "";
            msgObj.BillingDescription = "";
            msgObj.ContentType = intelesms.FileType.Raw;
            msgObj.AgeLimit = 0;
            msgObj.FromNumber = "";
            msgObj.ServiceID = 0;
            msgObj.ValidationPeriod = "168";
            msgObj.Subject = "My first mms message";

            //This is the file we'd like to send
            string tmpAddFilename = "c:\\test.jpg";

            //Check if the file exists
            if (!File.Exists(tmpAddFilename)) {
                  Console.WriteLine("The file " + tmpAddFilename + " does not
                  exists!");
                  Console.WriteLine("Press any key to quit");
                  Console.ReadKey();
                  return;
            }

            //now we'll add a single file to the message:
            intelesms.ContentFile contentFile1 = new intelesms.ContentFile();
            contentFile1.FileName = System.IO.Path.GetFileName(tmpAddFilename);
            contentFile1.File = File.ReadAllBytes(tmpAddFilename);

            //Now add the content file to the message object
            msgObj.Content =

      (intelesms.ContentFile[])Array.CreateInstance(typeof(intelesms.ContentFile), 1);
            msgObj.Content[0] = contentFile1;

            //declare object for send response
            intelesms.Response resp = null;

            try
            {
                  resp = client.Send(msgObj);
                  if (resp.error == intelesms.ErrorCode.Success)
                  {
                        //Message successfully stored at InteleSMS platform
                  Console.WriteLine("Message successfully sent:
                  error={0},\r\nerrorMessage={1}",
                                    resp.error, resp.errorMessage);
```

```
                }
                else
                {
                        //Failed to send message
                    Console.WriteLine("Failed to send MMS. Error:
                    error={0},\r\nerrorMessage={1}",
                                        resp.error, resp.errorMessage);

                }
            }
            catch (Exception ex)
            {
                    Console.WriteLine("Failed to send MMS. Error: {0}", ex.Message);
            }
            finally
            {
                    //cleanup
                    auth = null;
                    resp = null;
                    msgObj = null;
                    client = null;
            }

            Console.WriteLine();
            Console.WriteLine("Press any key to quit");
            Console.ReadKey();

        }
}
}
```

Example 2:
In the next sample we use a .NET web page to send a MMS message. In order for this to work, copy the MMSGateway.cs into the App_Code folder located in the web application root, or add a Web Reference to your Visual Studio project using the URL:
https://smsgw.intelesms.no/mms/scriptv2.asmx?WSDL and name the service intelesms.

Create a new web page and add the following two elements somewhere inside the form tag:

```
<asp:Button ID="sendSOAPBtn" Text="Send MMS" runat="server"
OnClick="sendSOAPBtn_Click" />
<asp:Label ID="sendStatus" runat="server" />
```

In the code file for your web page, add the following code:

```
protected void sendSOAPBtn_Click(object sender, EventArgs e)
{

        //initialize SOAP client
        intelesms.MMSGateway client = new intelesms.MMSGateway();

        //create authorize object
        intelesms.Authorizer auth = new intelesms.Authorizer();
        auth.CustomerID = 999;
        auth.SubCustomerID = 0;
        auth.Password = "MySecret";

        //apply authorizer object to client
        client.AuthorizerValue = auth;
```

```
        //initialize the MMS message object
        intelesms.MMSMessage msgObj = new intelesms.MMSMessage();
        msgObj.Recipient = "4712345678";
        msgObj.Gateway = 1933;
        msgObj.MessageID = "MMS_1";
        msgObj.Price = 0;
        msgObj.MessagePriority = intelesms.Priority.Normal;
        msgObj.BillingCategory = "";
        msgObj.BillingDescription = "";
        msgObj.ContentType = intelesms.FileType.Raw;
        msgObj.AgeLimit = 0;
        msgObj.FromNumber = "";
        msgObj.ServiceID = 0;
        msgObj.ValidationPeriod = "168";
        msgObj.Subject = "My first mms message";

        //This is the file we'd like to send
        string tmpAddFilename = "c:\\test.jpg";

        //Check if the file exists
        if (!File.Exists(tmpAddFilename))
        {
                sendStatus.Text = "The file " + tmpAddFilename + " does not exists!";
                return;
        }

        //now we'll add a single file to the message:
        intelesms.ContentFile contentFile1 = new intelesms.ContentFile();
        contentFile1.FileName = System.IO.Path.GetFileName(tmpAddFilename);
        contentFile1.File = File.ReadAllBytes(tmpAddFilename);

        //Now add the content file to the message object
        msgObj.Content =
(intelesms.ContentFile[])Array.CreateInstance(typeof(intelesms.ContentFile), 1);
        msgObj.Content[0] = contentFile1;

        //declare object for send response
        intelesms.Response resp = null;

        try
        {
                resp = client.Send(msgObj);
                if (resp.error == intelesms.ErrorCode.Success)
                {
                        //Message successfully stored at InteleSMS platform
                        sendStatus.Text = String.Format("Message successfully sent:
                        error={0},<br />errorMessage={1}",
                                        resp.error, resp.errorMessage);
                }
                else
                {
                        //Failed to send message
                        sendStatus.Text = String.Format("Failed to send MMS. Error:
                        error={0},<br />errorMessage={1}",
                                        resp.error, resp.errorMessage);

                }
        }
        catch (Exception ex)
        {
                sendStatus.Text = String.Format("Failed to send MMS. Error: {0}",
                ex.Message);
```

```
        }
        finally
        {
                //cleanup
                auth = null;
                resp = null;
                msgObj = null;
                client = null;
        }

}
```

4.2.2. VB.NET

Example 1:
Use the .NET tool WSDL included in the .NET SDK to generate the SOAP client. Alternatively you can add a service reference directly in the Visual Studio project.

Example of using the wsdl tool:
wsdl /l:VB /n:intelesms https://smsgw.intelesms.no/mms/scriptv2.asmx?WSDL

This command creates an MMSGateway.vb file in the corresponding directory. Import this file into your project or web site. The following sample is a console application which uses the generated SOAP client from the previous step:

```
Imports System
Imports System.IO

Module Module1

        Sub Main(ByVal args As String())

                'initialize SOAP client
                Dim client As New intelesms.MMSGateway()

                'create authorize object
                Dim auth As New intelesms.Authorizer()
                auth.CustomerID = 999
                auth.SubCustomerID = 0
                auth.Password = "MySecret"

                'apply authorizer object to client
                client.AuthorizerValue = auth

                'initialize the MMS message object
                Dim msgObj As New intelesms.MMSMessage()
                msgObj.Recipient = "4712345678"
                msgObj.Gateway = 1933
                msgObj.MessageID = "MMS_1"
                msgObj.Price = 0
                msgObj.MessagePriority = intelesms.Priority.Normal
                msgObj.BillingCategory = ""
                msgObj.BillingDescription = ""
                msgObj.ContentType = intelesms.FileType.Raw
                msgObj.AgeLimit = 0
                msgObj.FromNumber = ""
                msgObj.ServiceID = 0
                msgObj.ValidationPeriod = "168"
                msgObj.Subject = "My first mms message"
```

```
            Dim tmpAddFilename As String = "c:\test.jpg"
            'This is the file we'd like to send
            'Check if the file exists
            If Not File.Exists(tmpAddFilename) Then
                  Console.WriteLine("The file " & _
                  tmpAddFilename & " does not exists!")
                  Console.WriteLine("Press any key to quit")
                  Console.ReadKey()
                  Exit Sub
            End If

            'now we'll add a single file to the message:
            Dim contentFile1 As New intelesms.ContentFile()
            contentFile1.FileName = System.IO.Path.GetFileName(tmpAddFilename)
            contentFile1.File = File.ReadAllBytes(tmpAddFilename)

            'Now add the content file to the message object
            msgObj.Content = _
            CType(Array.CreateInstance(GetType(intelesms.ContentFile), 1), _
            intelesms.ContentFile())
            msgObj.Content(0) = contentFile1

            'declare object for send response
            Dim resp As intelesms.Response = Nothing

            Try
                  resp = client.Send(msgObj)
                  If resp.[error] = intelesms.ErrorCode.Success Then
                        'Message successfully stored at InteleSMS platform
                  Console.WriteLine("Message successfully sent: error={0}," & _
            vbCr & vbLf & "errorMessage={1}", resp.[error], resp.errorMessage)
                  Else
                        'Failed to send message

                  Console.WriteLine("Failed to send MMS. Error: error={0}," & _
                  vbCr & vbLf & "errorMessage={1}", resp.[error], resp.errorMessage)
                  End If
            Catch ex As Exception
                  Console.WriteLine("Failed to send MMS. Error: {0}", ex.Message)
            Finally
                  'cleanup
                  auth = Nothing
                  resp = Nothing
                  msgObj = Nothing
                  client = Nothing
            End Try

            Console.WriteLine()
            Console.WriteLine("Press any key to quit")

            Console.ReadKey()
      End Sub

End Module
```

Example 2:
In the next sample we use a .NET web page to send a MMS message. In order for this to work, copy the MMSGateway.vb into the App_Code folder located in the web application root, or add a Web Reference to your Visual Studio project using the URL:

https://smsgw.intelesms.no/mms/scriptv2.asmx?WSDL and name the service intelesms.

Create a new web page and add the following two elements inside the form tag:

```
<asp:Button ID="sendSOAPBtn" Text="Send MMS" runat="server"
OnClick="sendSOAPBtn_Click" />
<asp:Label ID="sendStatus" runat="server" />
```

In the code file for your web page, add the following code:

```
  Protected Sub sendSOAPBtn_Click(ByVal sender As Object, ByVal e As EventArgs)

        'initialize SOAP client
        Dim client As New intelesms.MMSGateway()

        'create authorize object
        Dim auth As New intelesms.Authorizer()
        auth.CustomerID = 999
        auth.SubCustomerID = 0
        auth.Password = "MySecret"

        'apply authorizer object to client
        client.AuthorizerValue = auth

        'initialize the MMS message object
        Dim msgObj As New intelesms.MMSMessage()
        msgObj.Recipient = "4712345678"
        msgObj.Gateway = 1933
        msgObj.MessageID = "MMS_1"
        msgObj.Price = 0
        msgObj.MessagePriority = intelesms.Priority.Normal
        msgObj.BillingCategory = ""
        msgObj.BillingDescription = ""
        msgObj.ContentType = intelesms.FileType.Raw
        msgObj.AgeLimit = 0
        msgObj.FromNumber = ""
        msgObj.ServiceID = 0
        msgObj.ValidationPeriod = "168"
        msgObj.Subject = "My first mms message"

        Dim tmpAddFilename As String = "c:\test.jpg"
        'This is the file we'd like to send
        'Check if the file exists
        If Not File.Exists(tmpAddFilename) Then
               sendStatus.Text = "The file " & tmpAddFilename & " does not exists!"
               Exit Sub
        End If

        'now we'll add a single file to the message:
        Dim contentFile1 As New intelesms.ContentFile()
        contentFile1.FileName = System.IO.Path.GetFileName(tmpAddFilename)
        contentFile1.File = File.ReadAllBytes(tmpAddFilename)

        'Now add the content file to the message object
        msgObj.Content = _
         CType(Array.CreateInstance(GetType(intelesms.ContentFile), 1),  _
         intelesms.ContentFile())
        msgObj.Content(0) = contentFile1

        'declare object for send response
        Dim resp As intelesms.Response = Nothing
```

```
        Try

                resp = client.Send(msgObj)
                If resp.[error] = intelesms.ErrorCode.Success Then
                        'Message successfully stored at InteleSMS platform
                        sendStatus.Text = _
        String.Format("Message successfully sent: error={0},<br />errorMessage={1}", _
                          resp.[error], resp.errorMessage)
                Else
                        'Failed to send message

                        sendStatus.Text = _
        String.Format("Failed to send MMS. Error: error={0},<br />errorMessage={1}", _
                          resp.[error], resp.errorMessage)
                End If

        Catch ex As Exception
        sendStatus.Text = String.Format("Failed to send MMS. Error: {0}", ex.Message)
        Finally
                'cleanup
                auth = Nothing
                resp = Nothing
                msgObj = Nothing
                client = Nothing
        End Try

End Sub
```

Appendix A. Delivery status codes

Status	Comment
128	Message expired in delivery
129	Message successfully delivered to End-user
130	The End-users phone is not correctly configured for MMS service
131	Message delivery is temporary delayed
132	Unspecified error in delivery
133	Message has been forwarded to the operators customer web interface for reading, since the direct delivery to the phone has failed
201	The End-user has been charged for the content, but delivery of the content itself has failed
1005	Not received by End-user due to insufficient balance on pre-paid account
1052	Subscriber has requested barring service for one or all of sms, mms and wap content services
9016	An error occurred in the charging/billing operation
10002	Operator does not support MMS services
*	900,2001,10001 and other codes means an error occurred while sending message from the MMS-platform to the operator or bulk-route

Note: If you experience other error codes than described in the table above, please contact technical support. Only the status 129 should be considered as a successfully delivered message. For the status 201 the message is only partially delivered. All other error codes should be considered as failed.

Appendix B. Response error codes

Status code	Comment
0	Success
1	Login failure
2	Gateway failure
3	PriceFailure
4	ValidationPeriodFailure
5	MessagePriorityFailure
6	MessageIDFailure
7	ContentTypeFailure
8	RecipientFailure
9	ContentMaxSizeFailure
10	ContentMissingFailure
11	ContentFilesFailure
12	UnspecifiedFailure
13	DatabaseFailure
14	SubjectFailure
15	ServiceIDFailure
16	IPAccessFailure
17	FromnumberFailure
18	DemoAccountLimit
19	InvalidAgeValue
20	InvaldBillingCategory
21	InvalidSoapFormattedRequest

Appendix C. Short numbers and message prices

Available Short-numbers and valid price values

Norway	Sweden	Denmark
Shared short-number 1933	Shared short-number 72777	Shared short-number 1230
Price value in ØRE 0 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200 2300 2400 2500 2600 2700 2800 2900 3000 3500 3900 4000 4500 4900 5000 5500 5900 6000 6900 7000 7900 8000 8900 9000 9900 10000	Price value in ØRE 0, 100 – 20000 in 100 ØRE step	Price value in ØRE 0 300 500 700 1000 1500 2000 2500 3000 3500 3900 4000 4500 4900 5000 5500 5900 6000 6500 6900 7000 7500