Short Message Peer-to-Peer Protocol Specification

Version 5.0-Draft06

Short Message Peer to Peer Protocol Specification
v5.0 Draft06 11-Dec-2001

Table Of Contents

List Of Tables

List Of Figures

1 Introduction

The SMS Forum, a non-profit organisation dedicated to the promotion of SMS within the wireless industry, manages the Short Message Peer to Peer (SMPP) protocol. The specification and related documentation is available from the SMS Forum Website http://www.smsforum.net

Support for SMPP is available via email at support@smsforum.net or via online discussion at the SMS Forum website http://www.smsforum.net.

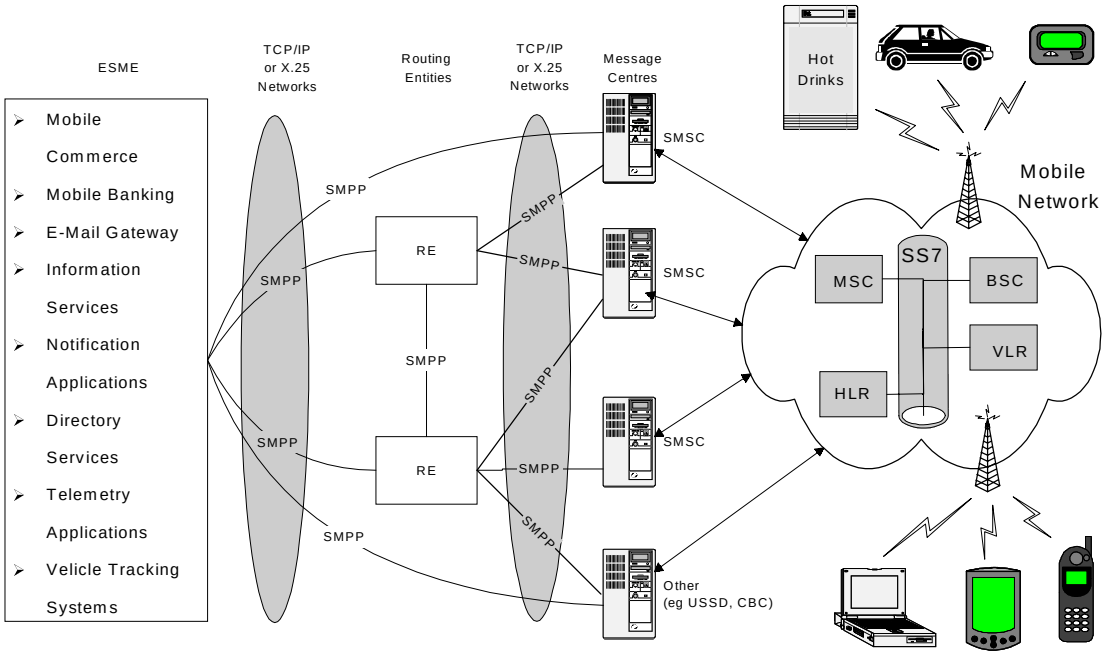
SMPP is an open, industry standard protocol designed to provide a flexible data communications interface for the transfer of short message data between External Short Message Entitys (ESME) , Routing Entities (RE)and Message Centers.

A Message Center (MC) is a generic term used to describe systems such as a Short Message Service Centre (SMSC), GSM Unstructured Supplementary Services Data (USSD) Server, and Cell Broadcast Centre (CBC).

An ESME typically represents a fixed network SMS client, such as a WAP Proxy Server, E-Mail Gateway, and Voice Mail Server.

A Routing Entity (RE) is a generic term for a network element that is utilized for MC to MC, and ESME to MC message routing. A RE has the ability to emulate the functionality associated with both a MC and an ESME. To an ESME, a RE appears as a MC and to a MC, an RE appears as an ESME. A carrier may utilise REs to hide a network of Message Centres, presenting only the REs as the external interface point for ESMEs.[CL1]

The following diagram illustrates the context of SMPP in a mobile network:



[CL2]

Figure 1-1 SMPP Network Diagram

1.1 Scope Of This Document

This document defines Version 5.0 of the SMPP protocol. It is intended for designers and implementers of an SMPP v5.0 interface between Message Centres, Routing Entities (RE)and External Short Message Entities (ESME).[CL3]

1.2 SMPP Overview

The following sub-sections overview the basic concepts and characteristics of SMPP. Many of these characteristics will be discussed in greater detail throughout the document.

1.2.1 Supported Cellular Technologies

SMPP is designed to support short messaging functionality for any cellular technology and has specific applications and features for technologies such as:

- GSM
- IS-95 (CDMA)
- ANSI-136 (TDMA)
- iDEN

1.2.2 Typical Applications of SMPP

There is no end to the applications of SMPP and SMS in general. Wireless Operators, Message Centre vendors, Infrastructure Providers, and application developers are constantly developing new applications for SMS. SMPP is ideal as an access protocol for these applications. The following summarises common applications of SMPP:[CL4]

- Voicemail alerts originating from a VPS (Voice Processing System), indicating voice messages at a customer's mailbox. This is arguably one of the first ESME-based applications of SMS and is still heavily used in the industry.

- Numeric and alphanumeric paging services. With an SMS-capable phone, the need to carry both pager and phone is drastically reduced.

- Information services. For example, an application that enables mobile subscribers to query currency rates or share-price information from a database or the WWW and have it displayed as a short message on the handsets.

- Calls directly dialled or diverted to a message-bureau operator, who forwards the message to the SMSC, for onward delivery to a subscriber's handset.

- Directory services. For example a subscriber calls a directory service requesting information on restaurants in a given area. The operator lists out available restaurants and sends the appropriate information as an SMS to the caller.

- Location-based services. These include applications that use mobile hardware to send GPS or cell data across SMS and using an SMSC, relay these messages to an ESME. The may then use the collected data to manage services such as taxi assignment, stolen vehicle tracking, logistics control.

- Telemetry applications. For example, a household meter that transmits a short message to a utility company's billing system to automatically record customer usage.

- Security applications such as alarm systems that can use SMS services for remote access and alerting purposes. For example, a parent receives an SMS from his security company to inform him that his daughter has arrived home and keyed in her access code.

- WAP Proxy Server. A WAP Proxy Server acts as the WAP gateway for wireless Internet applications. A WAP Proxy Server may select an SMS or USSD bearer for sending WDP datagrams to and receiving WDP datagrams from a mobile station.

- Online Banking, Share Dealing and E-Commerce, A mobile user could use SMS to send messages to an ESME requesting the purchase of products, shares etc. Likewise, a subscriber may use SMS to access banking services such as bill payment and funds transfer.

- Gaming and SMS Chat. Mobile users can interact with each other by means of a central server (ESME) and use this interaction as a means of playing wireless games, dating or SMS chat services similar to the concept of instant messaging and Internet room. These services have already appeared in the form of SMS-TV and SMS-Radio services.

1.2.3 SMPP Sessions

In order to make use the SMPP protocol, an SMPP session must be established between the ESME and Message Centre or SMPP Routing Entity where appropriate. The session is based on an application layer TCP/IP or X.25 connection between the ESME and MC/RE and is usually initiated by the ESME. [CL5]

There are three types of ESME-initiated session:

- Transmitter (TX)
  when authenticated as a transmitter, an ESME may submit short messages to the MC for onward delivery to Mobile Stations (MS). A transmitter session will also allow an ESME cancel, query or replace previously submitted messages. Messages sent in this manner are often called mobile terminated messages.
- Receiver (RX)
  A receiver session enables an ESME to receive messages from a MC. These messages typically originate from mobile stations and are referred to as mobile originated messages.
- Transceiver (TRX)
  A TRX ESME is a combination of TX and RX, such that a single SMPP session can be used to submit mobile terminated messages and receive mobile originated messages.

Additionally, the Message Centre can establish an SMPP session by connecting to the ESME. This is referred to as an Outbind Session.

1.2.4 Protocol Operations and PDUs

The SMPP protocol is basically a set of operations, each one taking the form of a request and response Protocol Data Unit (PDU). For example, if an ESME wishes to submit a short message, it may send a *submit_sm* PDU to the MC. The MC will respond with a *submit_sm_resp* PDU, indicating the success or failure of the request. Likewise, if a MC wishes to deliver a message to an ESME, it may send a *deliver_sm* PDU to an ESME, which in turn will respond with a *deliver_sm_resp* PDU as a means of acknowledging the delivery.

Some operations are specific to an ESME with others specific to the MC. Others may be specific to a given session type. Referring to the *submit_sm* and deliver_sm examples above, an ESME may send a submit_sm to a MC only if it has established a TX or TRX session with that Message Centre. Likewise, a MC may send a *deliver_sm* PDUs only to ESMEs that have established RX or TRX sessions.

A RE has been established to allow operations to be passed between MCs. The RE has the ability to translate a *deliver_sm* from a source MC to a *submit_sm* destined for a destination MC. [CL6]

Operations are broadly categorised into the following groups:

- Session Management
  Operations designed to enable the establishment of SMPP sessions between an ESME and MC and provide means of handling unexpected errors.
- Message Submission
  Operations geared specifically towards the submission of messages from ESMEs to the MC.
- Message Delivery
  Operations will enable the MC to deliver messages to the ESME
- Ancillary Operations
  Operations designed to provide enhance features such as cancellation, query or replacement of messages.

The following sub-sections list each category and its associated operations.

1.2.4.1 Session Management Operations

SMPP PDU Name	Description
bind_transmitter	Authentication PDU used by a transmitter ESME to bind to the Message Centre. The PDU contains identification information and an access password for the ESME.
bind_transmitter_resp	Message Centre response to a bind_transmitter PDU. This PDU indicates the success or failure of the ESME's attempt to bind as a transmitter
bind_receiver	Authentication PDU used by a transmitter ESME to bind to the Message Centre. The PDU contains identification information, an access password for the ESME and may also contain routing information specifying the range of addresses serviced by the ESME.
bind_receiver_resp	Message Centre response to a bind_receiver PDU. This PDU indicates the success or failure of the ESME's attempt to bind as a receiver
bind_transceiver	Authentication PDU used by a transceiver ESME to bind to the Message Centre. The PDU contains identification information, an access password for the ESME and may also contain routing information specifying the range of addresses serviced by the ESME.
bind_transceiver_resp	Message Centre response to a *bind_transceiver* PDU. This PDU indicates the success or failure of the ESME's attempt to bind as a transceiver
outbind	Authentication PDU used by a Message Centre to Outbind to an ESME to inform it that messages are present in the MC. The PDU contains identification, and access password for the ESME. If the ESME authenticates the request, it will respond with a *bind_receiver* or *bind_transceiver* to begin the process of binding into the MC.
unbind	This PDU can be sent by the ESME or MC as a means of initiating the termination of an SMPP session.
unbind_resp	This PDU can be sent by the ESME or MC as a means of acknowledging the receipt of an unbind request. After sending this PDU the MC typically closes the network connection.
enquire_link	An ESME or MC as a means of testing the network connection sends this PDU. The receiving end is expected to acknowledge the PDU as a means of verifying the test.
enquire_link_resp	This PDU is used to acknowledge an *enquire_link* request sent by an ESME or MC.
generic_nack	This PDU can be sent by an ESME or MC as a means of indicating the receipt of an invalid PDU. The receipt of a *generic_nack* usually indicates that the remote peer either cannot identify the PDU or has deemed it an invalid PDU due to its size or content.

Table 1-1 Session Management Operations

1.2.4.2 Message Submission Operations

SMPP PDU Name	Description
submit_sm	A transmitter or transceiver ESME, wishing to submit a short message, can use this PDU to specify the sender, receiver and text of the short message. Other attributes include message priority, data coding scheme, validity period etc XXXX
submit_sm_resp	The MC response to a *submit_sm* PDU, indicating the success or failure of the request. Also included is a MC *message_id* that can be used in subsequent operations to query, cancel or replace the contents of an undelivered message.
submit_sm_multi	A variation of the *submit_sm* PDU that included support for up to 255 recipients for the given message.
submit_sm_multi_resp	The MC response to a *submit_multi* PDU. This is similar to the *submit_sm_resp* PDU. The main difference is that where some of the specified recipients were either invalid or rejected by the Message Centre, the PDU can specify the list of failed recipients, appending a specific error code for each one, indicating the reason the recipient was invalid. Also included is a MC *message_id* that can be used in subsequent operations to query, cancel or replace the contents of an undelivered message.
data_sm	*Data_sm* is a streamlined version of the *submit_sm* operation, designed for packet-based applications that do not demand extended functionality normally available in the *submit_sm* operation. ESMEs implementing WAP over SMS typically use this operation.
data_sm_resp	The MC response to a *data_sm* PDU, indicating the success or failure of the request. Also included is a MC *message_id* that can be used in subsequent operations to query, cancel or replace the contents of an undelivered message.

Table 1-2 Message Submission Operations

1.2.4.3 Message Delivery Operations

SMPP PDU Name	Description
deliver_sm	Deliver_sm is the symmetric opposite to submit_sm and is used by a MC to deliver a message to a receiver or transceiver ESME.
deliver_sm_resp	This PDU indicates the ESMEs acceptance or rejection of the delivered message. The error returned by the ESME can cause the message to be retried at a later date or rejected there and then.
data_sm	*Data_sm* can also be used for message delivery from Message Centre to the ESME. ESMEs implementing WAP over SMS typically use this operation.
data_sm_resp	The ESME response to a *data_sm* PDU, indicating the success or failure of the MC-initiated delivery request.
alert_notification	A MC sends an *alert_notification* to an ESME as a means of altering it to the availability of an SME.

Table 1-3 Message Delivery Operations

1.2.4.4 Anciliary Operations

SMPP PDU Name	Description
query_sm	This PDU is used to query the state of a previously submitted message. The PDU contains the source address of the original message and the message_id returned in the original submit_sm_resp, submit_multi_resp or data_sm_resp PDU.
query_sm_resp	The MC returns a *query_sm_resp* PDU as a means of indicating the result of a message query attempt. The PDU will indicate the success or failure of the attempt and for successful attempts will also include the current state of the message.
cancel_sm	This PDU is used to cancel a previously submitted message. The PDU contains the source address of the original message and the *message_id* returned in the original *submit_sm_resp*, *submit_multi_resp* or *data_sm_resp* PDU. This PDU may also omit the *message_id* and instead contain a source address, destination address and optional *service_type* field as a means of cancelling a range of messages sent from one address to another.
cancel_sm_resp	The MC returns this PDU to indicate the success or failure of a *cancel_sm* PDU.
replace_sm	The *replace_sm* PDU is used by an ESME to pass a *message_id* of a previously submitted message along with several other fields used to update the text, validity period and other attributes of the message.
replace_sm_resp	The *replace_sm_resp* PDU indicates the success or failure of a *replace_sm* PDU

Table 1-4 Anciliary Operations

1.2.4.5 Operation Matrix

When we consider the various SMPP operations and states where ESMEs and MCs alike are allowed to use these operations, we fully appreciate the concept of SMPP Session States. The following table lists each SMPP operation PDU by name and the appropriate Session states for its usage, indicating in terms of ESME and MC, which peer can issue the PDU.

Note: An SMPP Routing Entity (RE) is capable of emulating an ESME and MC at the same time and therefore, all MC or ESME operations listed below are also simultaneously applicable to a RE. For example, a RE may issue a *bind_transmitter* to a MC while a session is in an open state (RE binding to Message Center). Additionally, the RE may return a *bind_transmitter_resp* PDU to an ESME with an open state session (ESME binding to RE).[CL7]

Session State	Open		Outbound		Bound_TX		Bound_RX		Bound_TRX		Unbound	
SMPP Entity / PDU	ESME	MC	ESME	MC	ESME	MC	ESME	MC	ESME	MC	ESME	MC
bind_transmitter	•		•									
bind_transmitter_resp		•		•								
bind_receiver	•		•									
bind_receiver_resp		•		•								
bind_transceiver	•		•									
bind_transceiver_resp		•		•								
Outbind				•								
Unbind					•	•	•	•	•	•		
unbind_resp					•	•	•	•	•	•		
submit_sm					•				•			
submit_sm_resp						•						
submit_sm_multi					•				•			
submit_sm_multi_resp						•						
data_sm					•			•	•	•		
data_sm_resp						•	•		•	•		
deliver_sm								•		•		
deliver_sm_resp							•		•			
query_sm					•				•			
query_sm_resp						•				•		
cancel_sm					•				•			
cancel_sm_resp						•				•		
replace_sm					•				•			
replace_sm_resp						•				•		
enquire_link	•	•	•	•	•	•	•	•	•	•	•	•
enquire_link_resp	•	•	•	•	•	•	•	•	•	•	•	•
alert_notification								•		•		
generic_nack	•	•	•	•	•	•	•	•	•	•	•	•

Table 1-5 Operation Matrix

1.3 Glossary

Term	Definition
ACK	Acknowledgement
API	Application Programming Interface
CDR	Call Detail Record
ESME	External Short Message Entity.
ETSI	European Telecommunications Standards Institute
HEADER	Leading portion of the SMPP message, common to all SMPP PDUs
MB	Message Bureau - This is typically an operator message bureau.
MC	Message Centre - A generic term used to describe various types of SMS Gateways.
MSB	Most Significant Byte
MSC	Mobile Switching Centre
MS	Mobile Station
MWI	Message Waiting Indication
NACK	Negative Acknowledgement
NSAP	Network Service Access Point
PDU	Protocol Data Unit
PSSD	Process Unstructured Supplementary Services Data
PSSR	Process Unstructured Supplementary Services Request
RE	Routing Entity[CL8]
SME	Short Message Entity
SMSC	Short Message Service Centre
SMPP	Short Message Peer to Peer Protocol
UDHI	User Data Header Indicator
URL	Uniform Resource Locator
USSN	Unstructured Supplementary Services Notification
USSR	Unstructured Supplementary Services Request
VMA	VoiceMail Alert
VPS	Voice Processing System
TIA	Telecommunications Industry Association
WAP	Wireless Application Protocol (http://www.wapforum.org)
WCMP	Wireless Control Message Protocol
WDP	Wireless Datagram Protocol

Table 1-6 Glossary

1.4 References

Ref.	Document Title	Document Number	Version Number
[GSM 03.40]	Technical Realisation of the Short Message Service Point to Point	GSM 03.40 http://www.etsi.fr	v5.7.1
[GSM 03.38]	"Digital Cellular telecommunications system (Phase 2+); Alphabets and language specific information".	[GSM 03.38] http://www.etsi.fr	v5.5.1 Sept. '97
[GSM MAP 09.02]	GSM Mobile Application Part	[GSM MAP 09.02] http://www.etsi.fr	v5.11.0
[IS637]	Short Message Service for Spread Spectrum Systems	TIA/EIA/IS-637-A	Rev A
[TSAR]	Teleservice Segmentation and Reassembly (TSAR)	TIA/EIA-136-620	Rev 0
[CMT-136]	Short Message Service - Cellular Messaging Teleservice	TIA/EIA-136-710-A	Rev A
[GUTS]	General UDP Transport Service (GUTS)	TIA/EIA-136-750	Rev 0
[WAPARCH]	Wireless Application Protocol Architecture Specification	WAP Forum http://www.wapforum.org	Version 30-Apr.-1998
[WCMP]	Wireless Control Message Protocol Specification	WAP Forum http://www.wapforum.org	Version 12-June-1998
[WDP]	Wireless Datagram Protocol Specification	WAP Forum http://www.wapforum.org	Version 10-Feb.-1999
[ITUT X.213]	Open Systems Interconnection - Net work Service Definition	[ITUT X.213]	11/95
[KOR ITS]	PCS operators common standards for handset-SMS functionalities	PCS standardization committee PCS-SMS-97-05-28	1.06 Rev 99-04-30
[3GPP TS 23.040]	Technical Realization of the Short Message Service (SMS) (Release 4)	3GPP	Version 4.4.0[CL9]
[3GPP TS 23.038]	Alphabets and language-specific information (Release 4)	3GPP	Version 4.3.0 [CL10]
[3GPP TS 23.038]	Mobile Application Part (MAP) Specification (Release 4)	3GPP	Version 4.5.0 [CL11]

Table 1-7 References

2 SMPP Sessions

As described earlier earlier, the ESME and MC communication is based on an SMPP session. This section describes this in detail.

2.1 Application Layer Communication

The Open Systems Interconnect model or OSI stack as it is more commonly known, defines a layered approach to data communications working form the most basic electronic data communications (physical), up to link level communication involving the transmission of octets and onwards to fully formed network packets (network) and ultimately to transport layers that manage packets and the reliable transport of data, ensuring the appropriate resending of packets that are not properly received at the remote end.

SMPP is an application layer protocol, using the same underlying communications as protocols such as well known services like telnet, ftp, http etc. An application layer connection is typically presented as a buffer through which an application can send and receive data. The transport of this data between one peer and another is completely hidden from the ESME or MC. In fact, nowhere in SMPP is there any support for parity, CRC checking or any other form of corruption detection .This is all automatically handled by the application layer functionality of TCP/IP or X.25. The only assumption made by an SMPP-based application, ESME or MC, is that the remote peer conforms to the protocol and that a PDU sent from one peer to another is fully recignisable as an SMPP PDU.

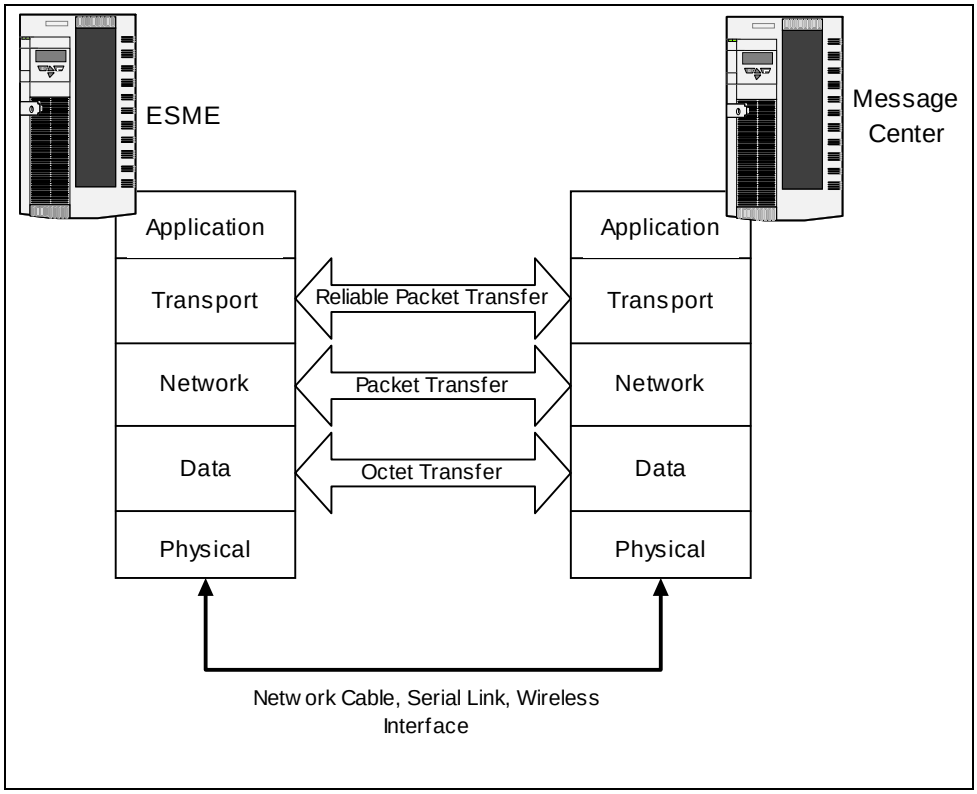


Figure 2-1 Application Layer Communication Between ESME and MC

2.2 Establishing an SMPP Session

Establishing a Session first requires the ESME to connect to the MC. This is achieved using a TCP/IP or X.25 connection. The MC will typically be listening for connections one one or more TCP/IP ports or X.25 interfaces (X.25 programmatic interfaces, DTE addresses and X.25 protocol Ids). For TCP/IP, IANA has standardised port 2775 for SMPP. However this will vary across MC vendors and operators.

2.3 Session States

As already described, an ESME begins a session by connecting to the MC across TCP/IP or X.25. This connection is referred to as an SMPP session and can have several states:

2.3.1 Open

An ESME has established a network connection to the SMSC but has not yet issued a Bind request. The MC is only aware of the TCP/IP or X.25 connection. No identification details have yet been exchanged.

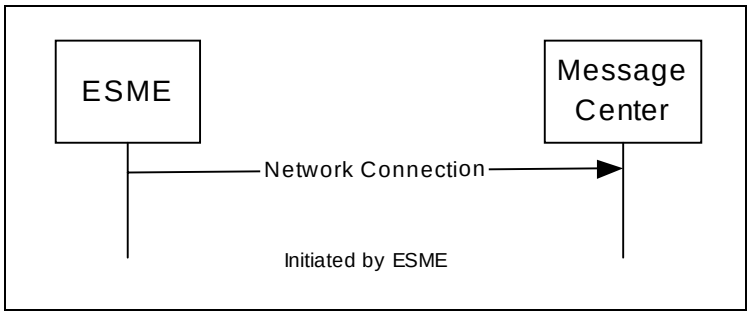


Figure 2-2 Open State

2.3.2 Bound_TX

A connected ESME has requested to bind as a Transmitter (by issuing a *bind_transmitter* PDU) and has received a *bind_transmitter_resp* PDU from the SMSC authorising its bind request. An ESME bound as a transmitter may send short messages to an SMSC for onward delivery to a Mobile Station or to another ESME. The ESME may also replace, query or cancel a previously submitted short message.

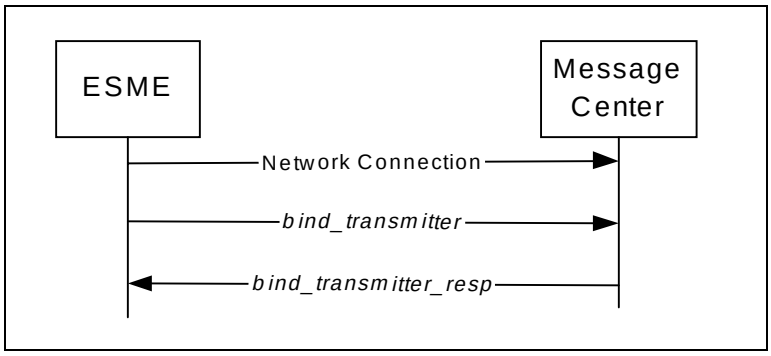


Figure 2-3 Bound_TX State

2.3.3 Bound_RX

A connected ESME has requested to bind as a Receiver (by issuing a *bind_receiver* PDU) and has received a *bind_receiver_resp* PDU from the SMSC authorising its Bind request. An ESME bound as a receiver may receive short messages from an SMSC, which may be originated, by a mobile station, by another ESME or by the SMSC itself (for example an SMSC delivery receipt).

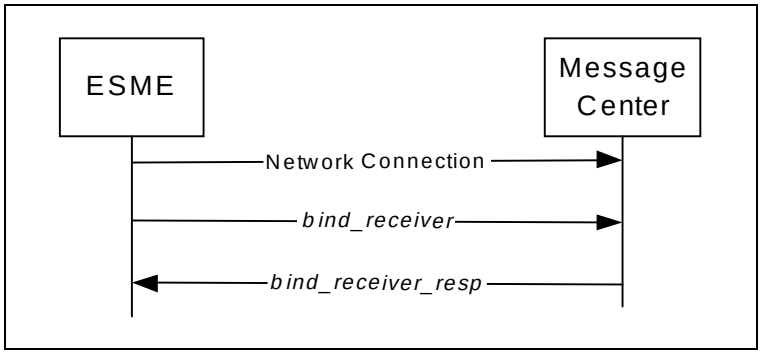


Figure 2-4 Bound_RX State

2.3.4 Bound_TRX

A connected ESME has requested to bind as a Transceiver (by issuing a *bind_transceiver* PDU) and has received a *bind_transceiver_resp* PDU from the SMSC authorising its Bind request. An ESME bound as a Transceiver is authorised to use all operations supported by a Transmitter ESME and a Receiver ESME. A transceiver ESME is effectively the combination of Transmitter and Receiver.

Thus an ESME bound as a transceiver may send short messages to an SMSC for onward delivery to a Mobile Station or to another ESME and may also receive short messages from an SMSC, which may be originated by a mobile station, by another ESME or by the SMSC itself (for example an SMSC delivery receipt).

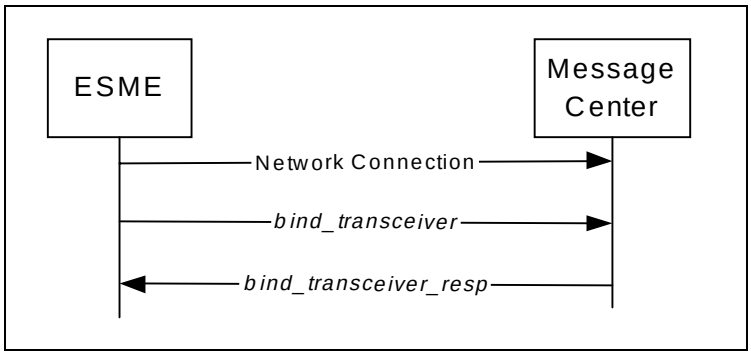


Figure 2-5 Bound_TRX State

2.3.5 Unbound

An ESME bound as a TX, RX or TRX ESME has issued an *unbind* request to the MC requesting termination of the SMPP session. The MC may also issue an unbind request to the ESME. The receiving peer will then respond with an *unbind_resp* PDU acknowledging the request to end the session.

2.3.6 Closed

The ESME or SMSC has closed the network connection. This typically results as a follow-on to an Unbound state where one peer has requested termination of the session. Closed state can also result from either peer terminating the connection unexpectedly or due to a communications error within the underlying network that results in termination of the network connection.

2.3.7 Outbound

The purpose of the *outbind* operation is to allow the SMSC signal an ESME to originate a *bind_receiver* or *bind_transceiver* request to the SMSC. An example of where such a facility might be applicable would be where the SMSC had outstanding messages for delivery to the ESME.

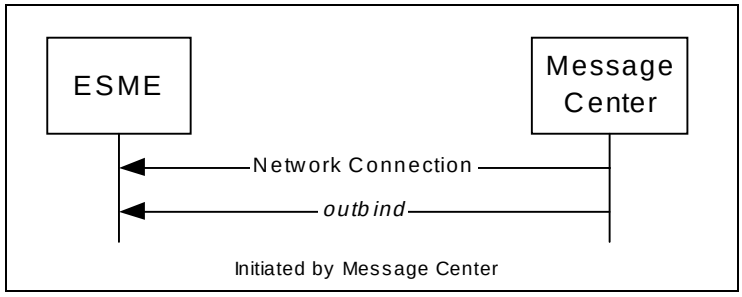


Figure 2-6 Outbound State

The following diagram illustrates the concept of Outbind when used to request a receiver or transceiver ESME to bind.

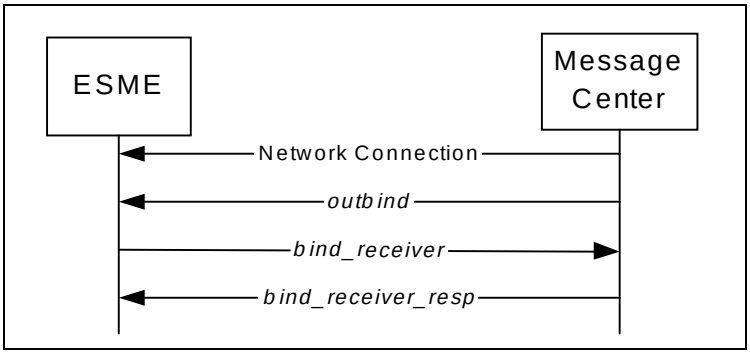


Figure 2-7 Bound_RX State from Outbound State

2.4 Sample Sessions

To help explain the context of SMPP operations and their related states, the following examples illustrate typical dialogues for the three types of ESME.

2.4.1 Example Transmitter Session

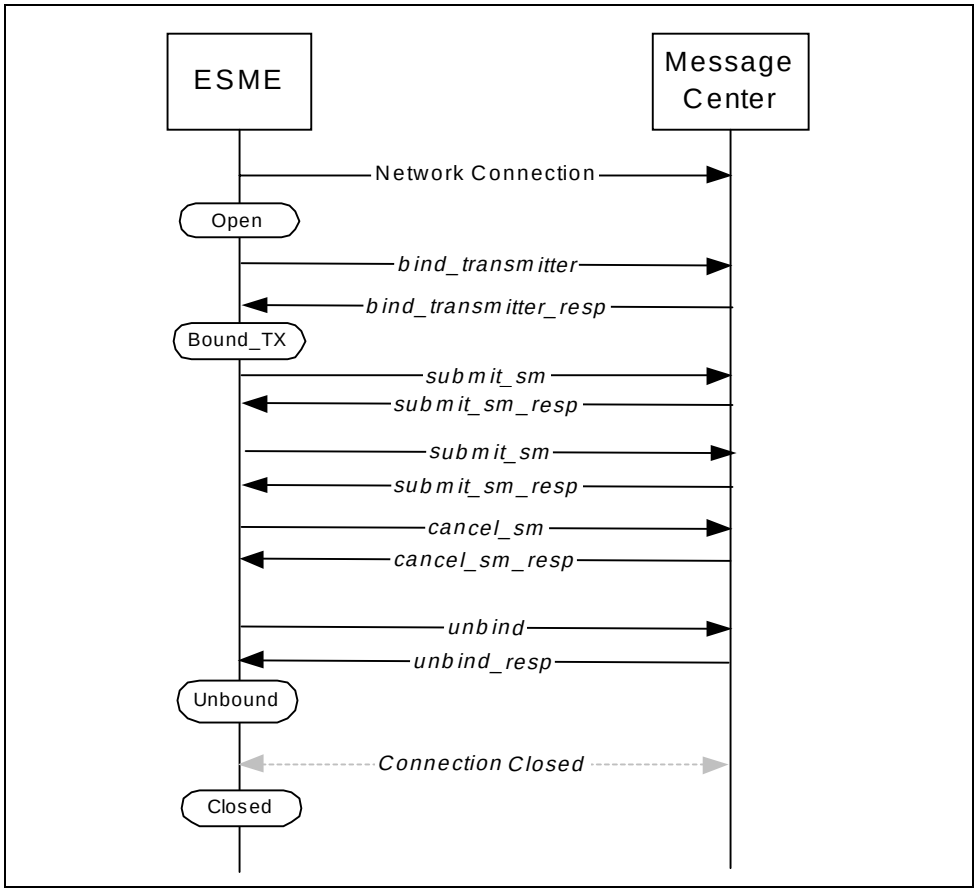


Figure 2-8 Example Transmitter Session

2.4.2 Example Receiver Session

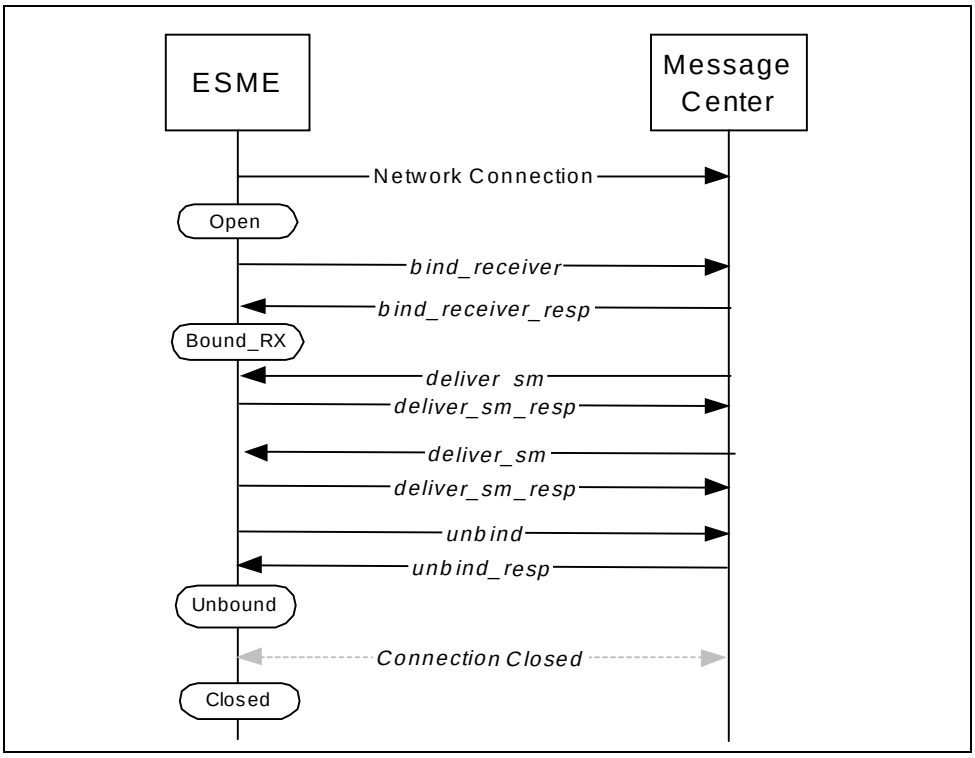


Figure 2-9 Example Receiver Session

2.4.3 Example Transceiver Session

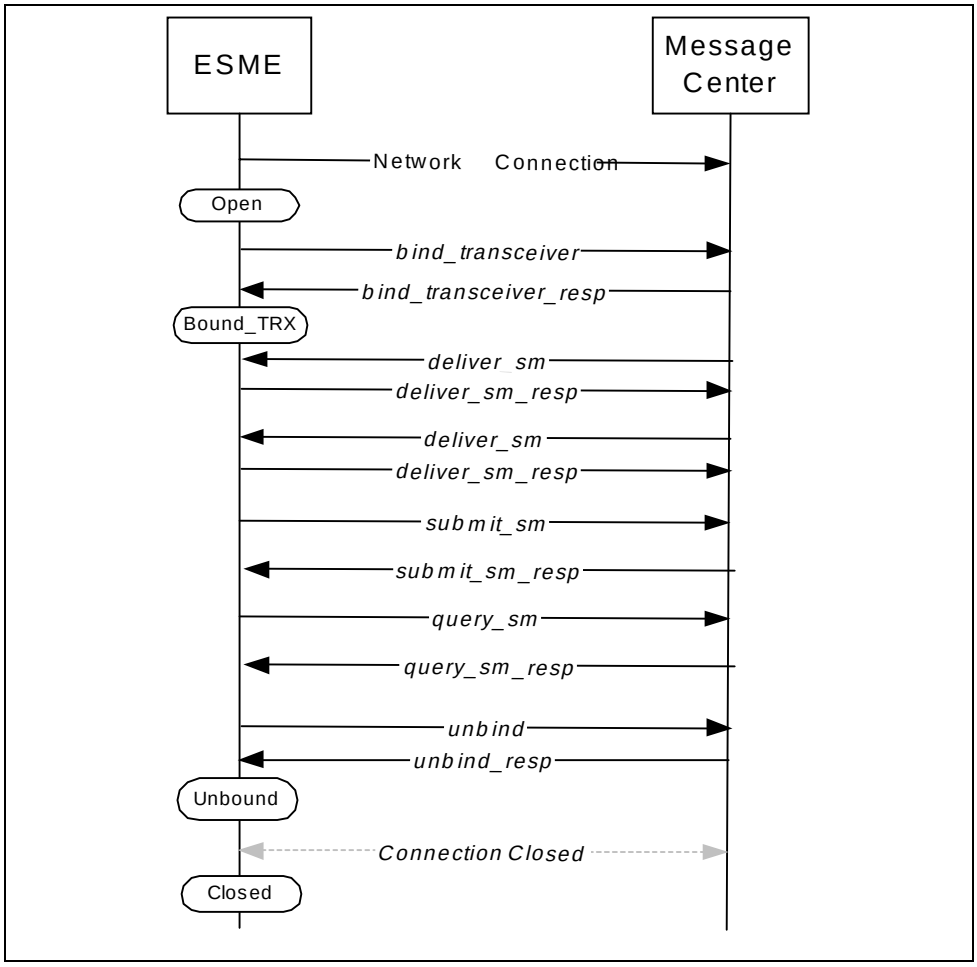


Figure 2-10 Example Transceiver Session

2.4.4 Example Outbind Session

This example depicts an outbind session that results in the binding of a receiver ESME.

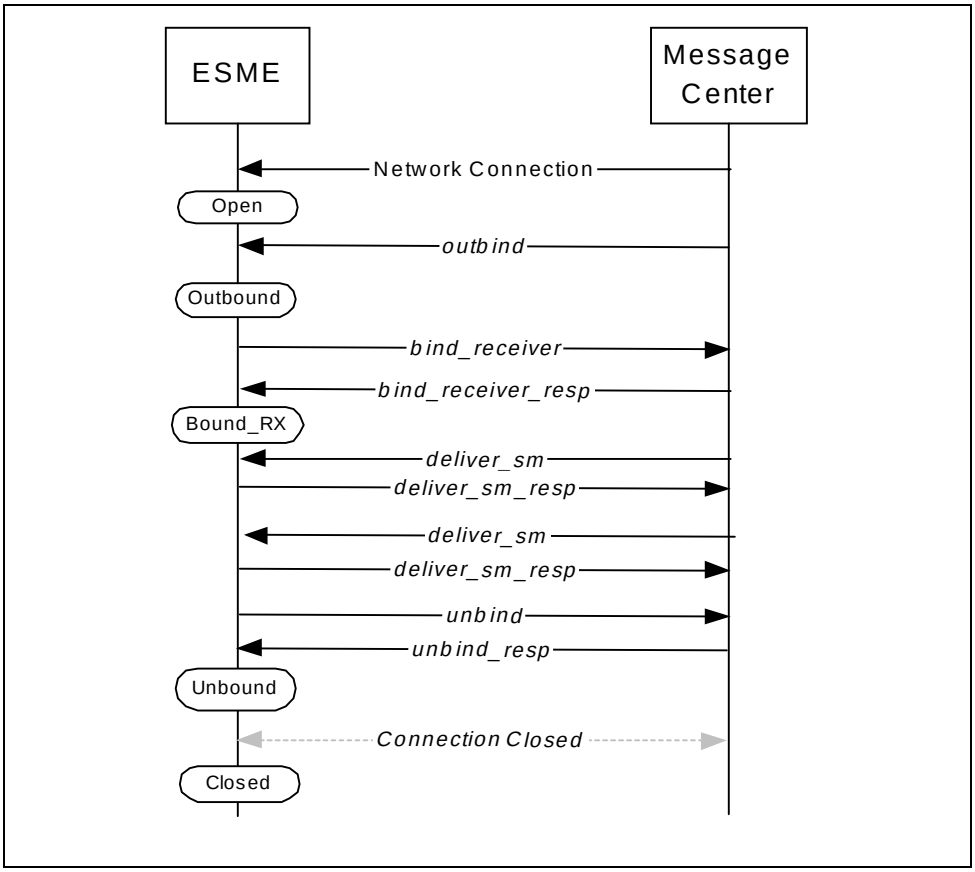


Figure 2-11 Example Outind Session

2.5 PDU Sequencing

Up to now, we have referred to PDUs by name and indicating the request/response pairs in the form of the session diagrams. The impression can be formed that SMPP is a handshake protocol where each request is first acknowledged before issuing the next request. However this is not the case.

2.5.1 The PDU Sequence Number

Each SMPP PDU has an identifier called a sequence number that is used to uniquely identify the PDU in the context of the SMPP session. The resulting response PDU is expected to mirror the sequence number of the original request. The following diagram illustrates the use of sequence numbers.

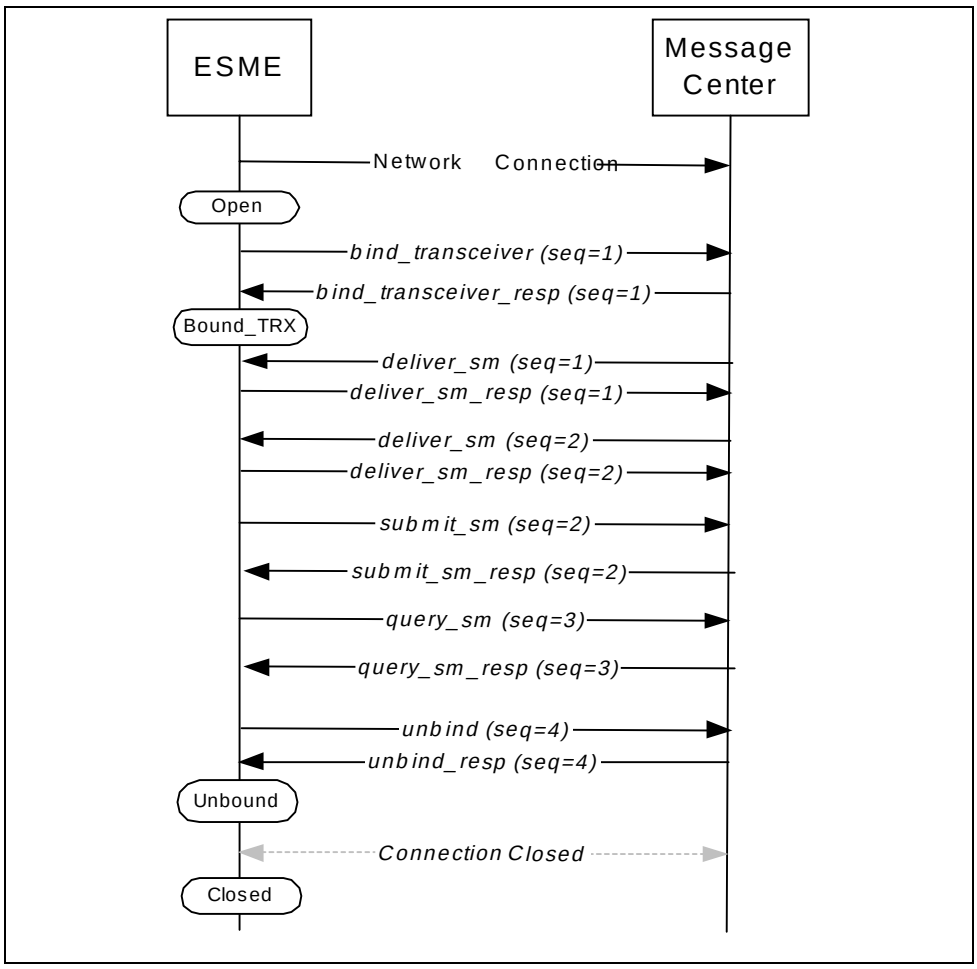


Figure 2-12 Transceiver Session demonstrating PDU Sequencing

Referring to the above example, sequence numbers are uniquely issued per request PDU. This means that for every PDU request issued by an ESME, it must use a different sequence number. The recommended approach is to use monotonic increasing sequence numbers, starting at 1. The first issued PDU request has a sequence number of 1, the next uses 2 and so on.

Each response PDU must carry the sequence number used in the matching request. In the above example, the MC responded with a *bind_transceiver_resp* carrying a sequence number of 1, simply because the *bind_transceiver* request had a sequence number of 1. The next request issued by the ESME was the *submit_sm* PDU and it carried a sequence number of 2. The third and forth request PDUs to be sent by the ESME further incremented this value to 3 and 4 respectively.

At the same time, all *deliver_sm* requests issued by the MC use monotonic increasing numbers. These can very easily match those issued by the ESME. But the confusion is avoided by the context of a PDU. Basically every request must emanate from either the ESME or MC and the resulting response must emanate from the other party. So in effect, each SMPP session involves two sets of sequence numbers; the set used by the ESME for ESME-originated requests and the set used by the MC for MC-originated requests. They can cross over each other in value, but will or should not be confused with each other.

2.5.2 Why Monotonic?

While the recommendation is that sequence numbers should be monotonically increased, this is not a mandatory requirement and some ESMEs or MCs may deliberately select random numbers or value based on some predetermined sequence. Either way, the responding party is expected to honour these values and use the same sequence number in the response PDU.

Monotonic increasing sequence numbers are easy to implement and handle and also have the additional characteristic of providing a running operation count for the session. If the next sequence number to be used by an ESME is 100, then that ESME will be issuing its 100th PDU within the session.

2.5.3 Sequence Numbers Across Sessions

Sequence numbers are designed for use within a single session. The recommended approach is to begin each session with a sequence number of 1 and increase monotonically from that point. The numbers used will not affect any other sessions that may already exist between the ESME and MC. Nor can PDUs from one session be acknowledged through another.

If the ESME-MC connection is closed or lost, then the expected recovery would be that the ESME or MC may re-establish the session. All unacknowledged PDUs from the lost session will not be acknowledged in the new recovery session, i.e. if the session was lost at a point where the MC had yet to send a *submit_sm_resp* to the ESME, then a new session established between the ESME and MC will not result in this response PDU being returned. Instead, the ESME is expected to begin numbering PDUs from 1. The same expectation applies for the Message Center.

2.5.4 Synchronous Vs. Asynchronous

SMPP is an asynchronous protocol. This means that an ESME or MC can send several requests at a time to the other party. The PDU sequence number plays a crucial role in supporting the asynchronous nature of SMPP. All example sessions shown in the previous sections have been synchronous. Here is an example of an asynchronous session.

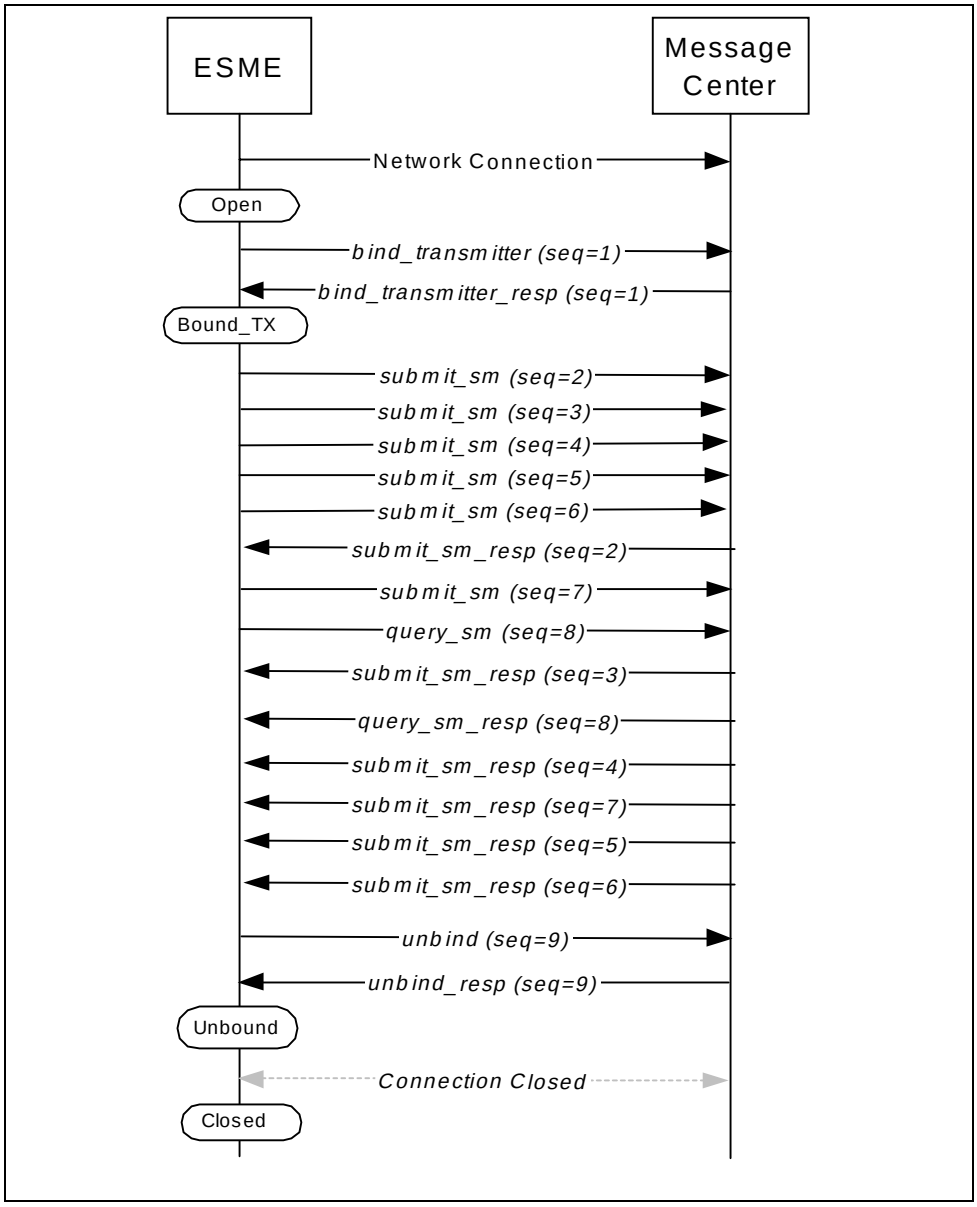


Figure 2-13 Asynchronous Transmitter Session

The asynchronous behaviour of both the ESME and Message Center is evident in the above session. The ESME issues 5 *submit_sm* requests to the MC before receiving its first *submit_sm_resp*. Note that the order in which the MC acknowledges these requests is by no means guaranteed to match the transmission order of the original requests. A Message Centre, because of it using a distributed architecture or because it is in fact SMPP Routing Entity and is distributing the ESME requests to other Message Centres, is likely to acknowledge requests in the order they are completed. Some requests may be distributed to busier parts of the system than others and as such may take longer to process. The result is that the asynchronous sequence of acknowledgements returned to the ESME may not carry the same order as used bye the requests. The same applies for messages delivered by the

MC to the ESME. The MC must support the ability to process response PDUs in a skew order.

The PDU sequence numbers make the request/response matching possible. So regardless of when an asynchronous response arrives, it can be immediately matched to the original request PDU. It is for this reason alone that each PDU request should use a unique sequence number within the context of the session.

2.5.5 Why Asynchronous?

Synchronous behaviour may appear the easier route to take when developing an application. Sending at most one PDU request, waiting for the response before sending the next can be an easy alternative over the management of an entire window of PDUs which may be acknowledged in a skew order.

However for an application to efficiently utilise the SMPP session, asynchronous transmission can make significant improvements. The following diagram helps explain the reasoning for using the protocol in an asynchronous manner.

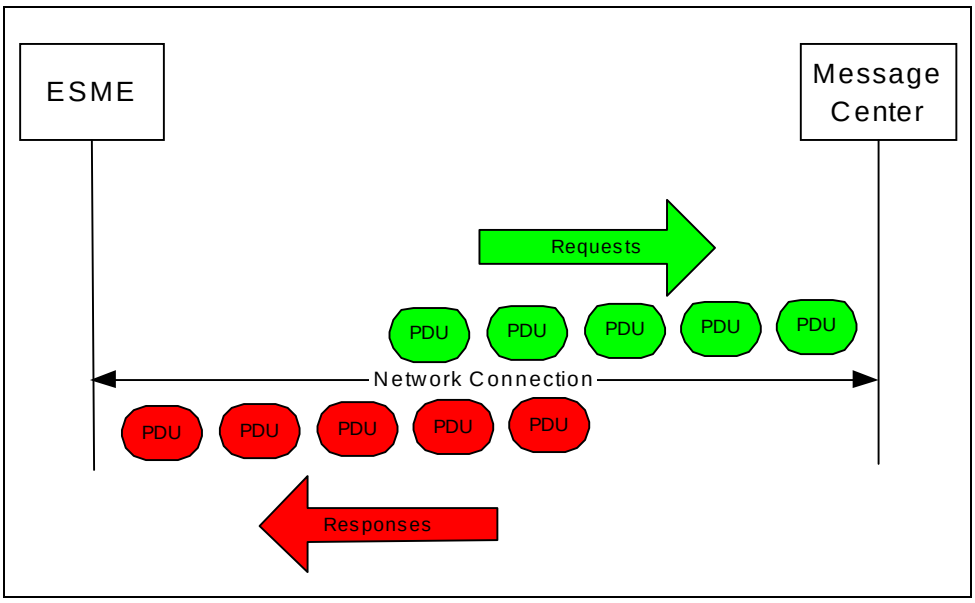


Figure 2-14 Asychronous Windowing

The above example shows 5 PDUs asynchronously transmitted across the SMPP session from the ESME to MC. If the MC can process only one PDU at a time, then the benefit of asynchronous transmission is actually not lost. In this situation, the window of PDUs becomes a queue of requests for the MC. As soon as the MC acknowledges each request, it will immediately find another request waiting. If the MC has the ability to dispatch several PDUs at one time for processing, then again the benefit of asynchronous transmission will ensure greater throughput.

If the ESME is synchronous and can only send a single PDU at a time, then as soon as the MC acknowledges the PDU, the SMPP session is idle until the response PDU is received by the ESME and the next request is dispatched and received by the MC.

If we consider the transport time from ESME to MC to take α microseconds, then the overall idle time per operation is 2α. This is the elapsed idle time while the response is in transit to the ESME and when the next request is in transit to the MC. An entria

asynchronous window of requests effectively avoids this inefficiency.

2.6 Session Timers

SMPP operations are based on the exchange of operation PDUs between ESME and MC. In order to control the amount of time spent waiting for a response to arrive or particular operation to occur, the following timers are defined:

Timer	Required SMPP Session State	Action on expiration	Description
Session Init Timer	Open Outbound	The network connection should be terminated.	This timer specifies the time lapse allowed between a network connection being established by an ESME and a *bind_transmitter*, *bind_receiver* or *bind_transceiver* request being sent to the MC. The timer can also be used by a MC supporting Outbind and applied to the time interval between an Outbind request being sent to an ESME and its response with a bind request. This timer can also be used by an ESME supporting Outbind where an ESME will close the MC-initiated connection within the defined period if the MC fails to send an Outbind request. This timer should always be active on the MC and on ESMEs supporting Outbind.
Enquire Link Timer	Open Unbound Outbound Bound_TX Bound_RX Bound_TRX	An *enquire_link* request should be initiated.	This timer specifies the time lapse allowed between operations after which an SMPP entity should interrogate whether it's peer still has an active session. This timer may be active on either side of the SMPP session (i.e. MC or ESME).

Timer	Required SMPP Session State	Action on expiration	Description
Inactivity Timer	Bound_TX Bound_RX Bound_TRX	The SMPP session should be dropped.	This timer specifies the maximum time lapse allowed between transactions, after which period of inactivity, an SMPP entity may assume that the session is no longer active. The resulting behaviour is to either close the session or issue an unbind request. This timer may be active on the MC or ESME side of the session.
Response Timer	Open Unbound Outbound Bound_TX Bound_RX Bound_TRX	The entity, which originated the SMPP Request, may assume that Request has not been processed and should take the appropriate action for the particular SMPP operation in question.	This timer specifies the time lapse allowed between an SMPP request and the corresponding SMPP response. This timer may be active on either communicating SMPP entity (i.e. SMSC or ESME).

Table 2-1 SMPP Session Timers

2.7 Error Handling

This section addresses typical error scenarios that can occur and how an ESME or MC should go about handling such scenarios

2.7.1 Handling Connection Failure

An ESME or MC may experience a failed connection attempt or may suddenly lose its connection to the other peer. This can be due to any of the following reasons.

- The IP address and port or X.25 details are incorrect (for failed connection attempts)
- The remote MC or ESME is down or unable to accept the connection
- The network between the two hosts is down

The recommended approach is to continually try to connect or reconnect again at intervals. Many ESME systems provide crucial services to an SMS network. If a network or Message Centre becomes unavailable, causing ESMEs to lose their SMPP connections, then as long as an ESME enters a mode whereby it attempts to reconnect at intervals of say five seconds, then as soon as the network or Message Centre service is restored, the ESME will be quickly reconnected to resume service.

Most SMPP sessions will be ESME-initiated, but as we have seen from Outbind, the MC can itself be in a position to connect to an ESME that is configured for Outbind. The likely reason for attempting an Outbind is that messages for the ESME have arrived in the MC. If the connection attempt fails, it is recommend that the MC use a similar mechanism of periodically attempting to Outbind to the ESME. This may be driven by a retry sequence that controls the frequency of delivery attempts made for a given message. Every time the ESMEs message is scheduled for retry, the MC attempts to Outbind to the ESME.

2.7.2 Operation Failure

There are a number of reasons why a MC or ESME may reject an operation request PDU. These are:

- The PDU is unrecognised
  this is one of the most common reasons for rejecting a PDU. It can result from an ESME or MC sending a PDU that the receiving peer does not recognise. The typical response in this scenario is a *generic_nack* PDU being returned to the sender with the sequence number of the offending PDU. The command status is usually ESME_RINVCMDID, which indicates that the ESME or MC cannot recognise the PDU. In this situation the error is with the receiving peer and is effectively a non-compliance scenario. However some ESMEs and MCs will not support all operations and will usually provide this information in their SMPP PICS document.
- The PDU is malformed
  when this happens; it indicates that the sending entity is at fault for sending a non-standard PDU. Typical responses will depend on how the malformed PDU is detected. If the *command_id* the reason for rejection, the receiving peer should respond with a *generic_nack* and command status set to ESME_RINVCMDID. If the *command_length* of the PDU appeared too large, suggesting that it was invalid, the ESME or MC should respond with a *generic_nack* and command status set to ESME_RINVCMDLEN.
- Invalid Field Length
  If any PDU field is to long or too short, then the PDU is essentially malformed, but an ESME or MC may indeed recognise the PDU and as such will respond with a *submit_sm_resp* or whatever is the appropriate PDU to use and set the command status to the appropriate error. For example, if an ESME submits a message with a 20 character scheduled delivery time, the rejection should be a *command_status* set to ESME_RINVSCHED
- The PDU data is unexpected and deemed invalid
  this type of rejection tends to be very application specific and an absolute list of causes is well beyond the scope of this document. These types of rejections are not protocol non-compliance issues, rather application errors. An example of such a scenario could be an Email gateway that provides a service to convert Mobile originated SMS messages into emails. The email addressing might be based on the text content of the message. If the ESME found that the message forwarded by the MC did not contain the correct formatting it would reject the message. The typical error code would be ESME_RX_R_APPN, which is a means of rejecting a message and ensuring that it is not retried at a later point.
- The PDU is not allowed in the current session state
  This is a violation of the rules of the SMPP sessions and a simple example would be an ESME in Bound_RX state, attempting to submit a message by sending a *submit_sm* PDU or by attempting to submit a message across an Open State session without first binding. The expected command status in the *submit_sm_resp* PDU would be ESME_RINVBNDSTS
- The ESME or MC is restricting the use of certain PDUs or features
  SMPP has a broad scope of functionality and some Message Centres or ESMEs may deliberately provide mechanisms to disable certain features. For example if an operator configured a message centre to reject attempts by ESMEs to request delivery receipts for messages, it would force the MC to reject the message with a command status set to ESME_RINVREGDLVFLG. Although the field may not incorrectly encoded, its usage is disabled and the MC is authorised to reject the message using that error code.

2.8 Flow Control and Congestion Avoidance

It is a common misconception that windowing provides full flow control. However all that is gained is a finite limit to an asynchronous window and as already discussed section 2.5.4, asynchronous transmission has many advantages in the area of maximising throughput.

Flow control relates to the concept of a receiver informing the sender that it can't accept any more data.

In TCP, this concept is supported by a 'receiver buffer advertisement', which can be passed with every packet acknowledgement. The sender uses this data as a means of judging how much data can be sent in subsequent transmissions. This mechanism works fine for TCP, given that congestion is based mostly on volumes of data being sent across busy networks or to a congested receiver.

In terms of SMPP, if an ESME or MC submits/delivers messages at a rate that exceeds the capabilities of its peer, congestion may occur. Relying on windowing to solve the problem is not enough. The ESME will continue to top up its Window of unacknowledged requests, keeping the MC under load to process these requests.

To better assist a peer (ESME or MC) in avoiding congestion, we would need to provide a means for the receiving peer to indicate to the sending peer, its state of congestion.

This is accomplished with the addition of an optional *congestion_state* TLV. This parameter may be optionally included in any response PDU sent between ESME and MC. This TLV contains a simple integer from 0-100 to indicate the Congestion state ranging from idle to congested. Refer to 4.6.4.45 for details on the values acceptable for this TLV.

The ESME or MC can use this value as a means of detecting increased load scenarios on its peer and take the appropriate action to reduce its input rates. In order to get the maximum throughput from the transfer of messages, the MC or ESME should try to maintain the Congestion state between 80-90 (Optimum Load).

When commencing an SMPP session, the ESME or MC would begin transmission of requests, with a maximum window of N. If the ESME or MC supports the Congestion_State TLV, then as the responses arrive, the ESME/MC can increase/decrease its transmission rate according to the indicated Congestion state of its peer. If the Congestion_State TLV is supported, then the Window of N can actually be discarded as the Congestion_State itself acts as a better indicator and preventative measure of congestion avoidance.

If the TLV is not present in response PDUs, then the simple windowing is the only means of applying flow control within the session.

The advantage of using *congestion_state* over a fixed window is that the ESME can avail of the most optimum performance available at a particular time instead of predetermining some window limit and using this consistently. This recognises that a MC or ESME may be under varying levels of stress and that predetermined performance is not always guaranteed.

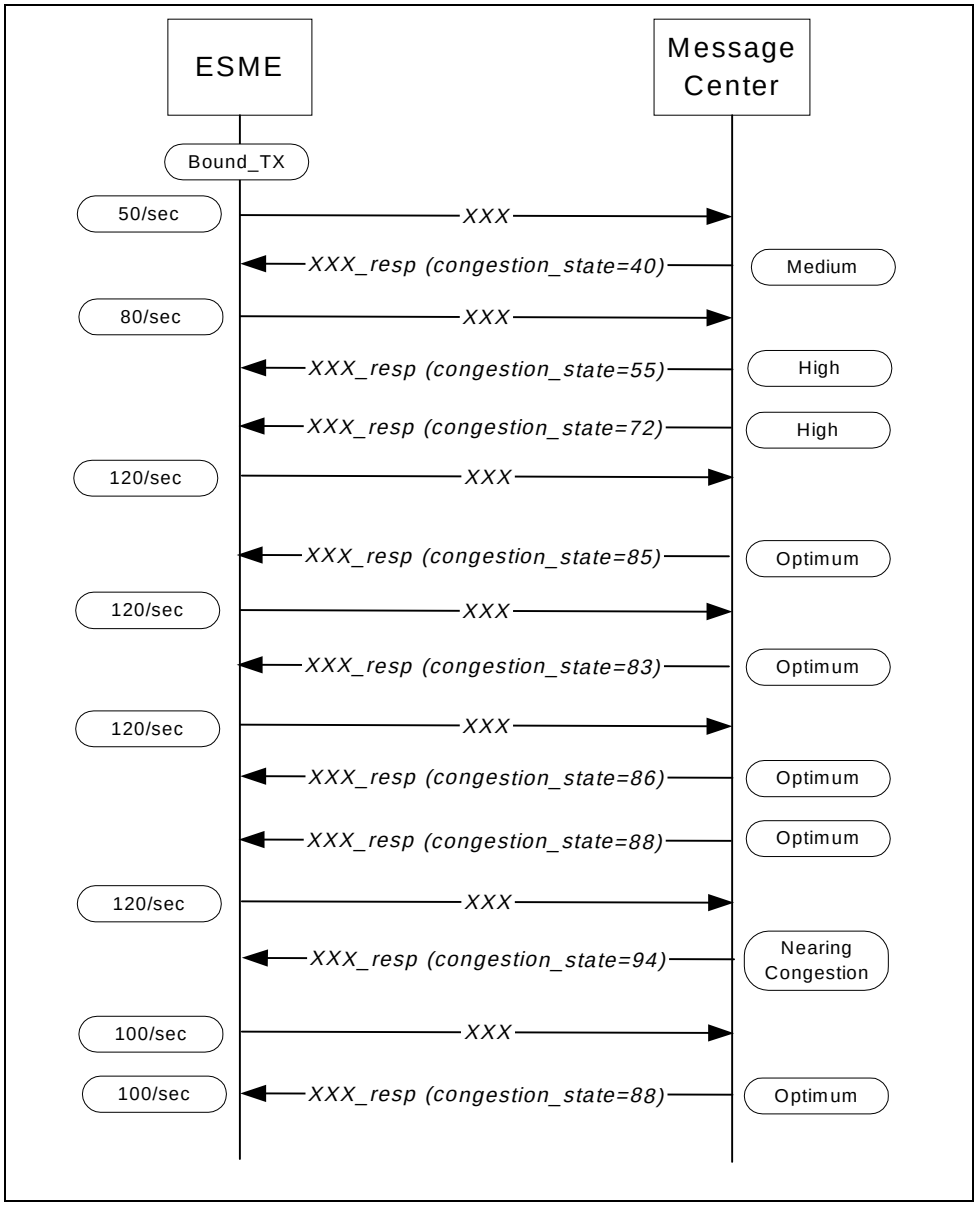


Figure 2-15 Flow Control & Congestion Avoidance using the *congestion_state* TLV.

The above diagram shows a session where an ESME is transmitting PDUs at a rate of 50/second. On recognising the *congested_state* TLV and its below Optimum value, the ESME increased its rate until the *congestion_state* enters an optimum range. At this point, the ESME maintains the 120 PDUs/second until the *congestion_state* enters 'Nearing Congestion', at which time the ESME relaxes the messaging rate to return the *congestion_state* to an optimum level.[CL12]

2.9 Session Security and Encryption

SMPP itself, does not define any native encryption mechanisms. The content exchanged across an SMPP session is open to hacking if it is transmitted across an open medium such as the Internet. There are two recommended approaches to securing SMPP sessions.

2.9.1 Secure Transport Layer

The well known Secure Socket Layer (SSL) and its standardised form TLS provide an excellent means of securing a connection between two peers by using a variety of encryption ciphers and authentication certificates. SSL/TLS is the backbone of how E-commerce is managed and has provedn itself very reliable. Many commercial and some open source libraries exist to facilitate the adaptation of SSL/TLS-based transport layers into applications. The ESME or MC, in supporting the SSL/TLS approach would connect using the SSL/TLS mechanism, first establishing a secure and authenticated conenction before commencing the SMPP session.

2.9.2 Secure Tunnel

Another approach to securing an SMPP session is to use a secure tunnel. This is where the ESME-MC connection is not made directly from one peer to another but by connecting to a secure tunnel server. The tunnel server is configured to match a particular insecure connection from an ESME with a secure connection to another tunnel usually located in the peers network. The remote tunnel, then matches the incoming connection with another insecure connection onwards to the other peer. The result is that an ESME and MC combination unable to suport direct SSL/TLS can still be secured for Internet ready communication by means the tunnel servers

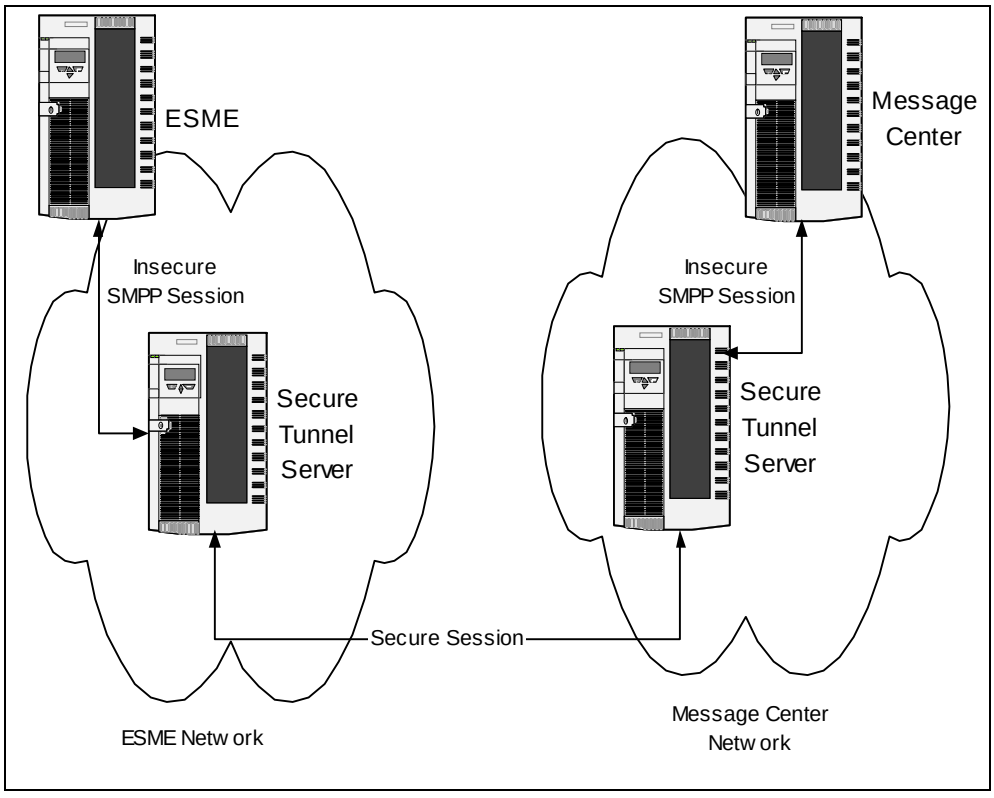


Figure 2-16 ESME-MC SMPP session using a secure tunnel

2.10 Forward and Backward Compatibility

SMPP is an evolving protocol and V5.0 represents another version of the protocol. The previous versions that are commonly used are V3.3 and V3.4.

Version 3.4 introduced many new features that enhanced existing PDUs to support optional parameters. To preserve backwards compatibility for older V3.3 compliant applications, V3.4 required the *bind* operations to set the *interface_version* to 0x34. V5.0 continues this approach and requires the *interface_version* to be set to 0x50.

2.10.1 Forward Compatibility

Forward Compatibility procedures allow a functional entity (i.e. MC or ESME) using one version of the SMPP protocol to easily communicate with an entity using a later, more enhanced version of the protocol. Hence, new enhancements to existing SMPP PDUs are implemented using TLVs.
The following guidelines must be followed in SMPP implementations to ensure that this process is implemented successfully and consistently:

- If an SMPP entity receives an unrecognized PDU/command, it must return a *generic_nack* PDU indicating an invalid *command_id* in the *command_status* field of the header.

- The SMPP entity receiving a message which includes Optional Parameters shall first inspect the Tag field of the Operational Parameter, as follows:
    - If the Optional Parameter Tag is recognized and supported by the receiving SMPP entity for the particular SMPP operation, the Optional Parameter shall be processed.
    - If an Optional Parameter Tag is recognized but not expected for the particular SMPP operation, the optional parameter shall be ignored.

- If the Optional Parameter Tag is unrecognized or unsupported by the receiving SMPP entity, that particular Optional Parameter shall be ignored and the next Optional Parameter in the sequence shall be processed.

- An SMPP entity receiving a parameter value defined as "reserved" should use the default value if a "default" setting is defined, otherwise the parameter should be ignored.

- If the Parameter value is otherwise unrecognized or invalid, the SMPP entity should return an error indicating the Parameter Value is invalid.

- An SMPP entity detecting that an Optional Parameter, which is required in the context of the operation, is not present should return a response message with an "Expected Optional Parameter missing" error.

- A Variable length field Parameter may have it's maximum length definition extended in subsequent versions of the SMPP protocol. An SMPP entity receiving a variable length Parameter whose length is greater than the maximum length the entity supports for that Parameter should reject the Parameter with an error indicating "invalid parameter length".

2.10.2 Backward Compatibility

Backward Compatibility procedures allow a functional entity using one version of the SMPP protocol to communicate with an entity using an older version of the protocol.
The following guidelines must be followed in SMPP implementations to ensure that this process is implemented successfully and consistently:

- Existing SMPP PDUs must not be removed from the protocol.
- The effect of receiving any existing message in a new modified format must be same as that in previous versions. Thus the addition of new parameters or parameter values is purely additive.
- Optional parameters shall not become mandatory parameters.
- Mandatory parameters shall not become optional parameters.
- Additional mandatory parameters shall not be added to an existing SMPP PDU.
- Existing mandatory parameters shall not be removed from an existing SMPP PDU.
- The meaning of any existing parameter value shall not be changed in the new version of the protocol.

As the concept of Optional Parameters was first introduced V3.4 of the protocol, the following special guidelines were defined:

- An SMSC that implements SMPP V3.4 or a later version of this protocol must not send optional parameters to an ESME that implements an earlier SMPP version (e.g. V3.3). An SMSC shall determine the SMPP version supported by an ESME during the bind operation. An ESME supporting SMPP V3.3 or earlier will set the *interface_version* parameter in the bind operation to a value less than 0x34 or equal to 0x00.
- An SMSC supporting V3.4 or later should return the SMPP version it supports in the *sc_interface_version* parameter of the bind response PDU. If a bind response does not contain the *sc_interface_version* parameter, then the ESME should assume that the SMSC does not support the use of optional parameters.
- An ESME that implements SMPP V3.4 or a later version of this protocol must not send optional parameters to an SMSC that implements an earlier version of this protocol. The ESME shall determine the SMSC version support from the SMPP bind response PDU.
- An SMSC that implements SMPP V3.4 or later must not generate message IDs greater than 8 octets when communicating with an ESME that supports SMPP V3.3 or earlier.

3 SMPP Parameter and PDU Format

This section defines the various types used to build SMPP PDUs. It also describes the layout of a PDU and how it appears when encoded.

3.1 Parameter Type Definitions

All SMPP PDUs comprise of organised sets of parameters. These parameters can have any of the following formats:

Note: Values depicted with a 0x prefix are in Hexidecimal format, meaning that each digit represents 4 binary bits. In short, a 2-digit hex number is actually only 1 octet of data.

Parameter Type	Description
Integer	An unsigned integer value, which can be 1, 2 or 4 octets in size. The octets are always encoded in Most Significant Byte (MSB) first order, otherwise known as Big Endian Encoding. A 1-octet Integer with a value 5, would be encoded in a single octet with the value 0x05 A 2-octet integer with the decimal value of 41746 would be encoded as 2 octets with the value 0xA312 A 4-octet integer with the decimal value of 31022623 would be encoded as 4 octets with the value 0x1D95E1F
C-Octet String	A C-Octet String is a sequence of ASCII characters terminated with a NULL octet (0x00). The string “Hello” would be encoded in 6 octets (5 characters of “Hello” and NULL octet) as follows: 0x48656C6C6F00 Two special variants exist for use within SMPP. These are C-octet String (Decimal) and C-Octet String (Hexadecimal), which are used to carry decimal and hexadecimal digit sequences respectively. These fields are encoded the same way as any ASCII string, but are specifically used to designate decimal and hexadecimal numbers when presented in string format. A Decimal C-Octet String “123456789” would be encoded as follows: 0x31323334353637383900 A Hexadecimal C-Octet String “A2F5ED278FC” would be encoded as follows: 0x413246354544323738464300

Parameter Type	Description
Octet String	An Octet String is a sequence of octet not necessarily terminated with a NULL octet. Such fields using Octet String encoding typically represent fields that can be used to encode raw binary data. In all circumstances, the field will be either a fixed length field or explicit length field where another field indicate the length of the Octet String field. An example of this is the *short_message* field of the submit_sm PDU which is Octet String encoded and its length is specified by the previous *message_length* field.
Tagged Length Value (TLV)	A Tagged Length Value Field is a special composite field that comprises or three parts: A 2-octet Integer (Tag) A 2-octet Integer (Length) An Octet String (Value) The Tag identifies the parameter. The Length indicates the size of the Value field in octets. An example of a TLV is the *dest_bearer_type*. Its Tag is 0x0007 and had a value size of 1 octet. The value 0x04 indicates USSD as a bearer type. In its encoded form, this TLV would be as follows: 0x0007000104 the first 2 octets 0x0007 identifies the Tag *dest_bearer_type*. The next two octets 0x0001 indicate the 1-octet length of the value field. The value field 0x04 indicates USSD ref XXXX

Table 3-1 SMPP PDU Parameter Types

3.1.1 NULL Settings

When we refer to a NULL setting for a field, we imply that the field is not carrying a value. However it must still be encoded in the PDU. The following examples indicate how NULL settings are handled for each of the SMPP PDU Parameter Types.

Parameter Type	NULL Setting Encoding
Integer	1-Octet: 0X00 2-Octet: 0X0000 4-Octet: 0X00000000
C-Octet String	A NULL string “” is encoded as 0x00
Octet String	A NULL Octet-String is not encoded. The explicit length field that indicates its length should be set to NULL
Tagged Length Value (TLV)	There are two types of NULL encoding for a TLV. The first is a TLV that may not carry a value part. An example of such a TLV is *alert_on_message_delivery*. This TLV is typically used as in indicator only, i.e. its function is driven by its very presence in the PDU. No data is typically present. However it can carry up to XXX octets of data if required.[CL13] Here are two examples of how this TLV can be encoded, the first example caries a value, the second example does not: Tag=0x130C Length=0x0002 Value=0x0102 Encoded Format: 0x130C00020102 Tag=0x130C Length=0x0000 Value=NULL Encoded Format: 0x130C0000 Note: Only the Tag and Length are encoded. No NULL octets are specified for the zero length Value field. If the TLV itself is not required, then it is not encoded at all. The very absence of the TLV from the PDU is the means by which we set the values to NULL. Note: Encoding C-Octet strings in TLVs carries a certain exception case. Where a TLV specifies only a C-Octet string in the value part or specifies a format where the last encoded part is a C-Octet String, then it is perfectly acceptable to exclude the NULL octet and leave the TLV dimensioning terminate the data flow.[CL14]

Table 3-2 SMPP PDU Parameter Type NULL Settings

3.1.2 SMPP Parameter Field Size Notation

The following notation style is used throughout. Note that some SMPP strings are optional and others mandatory.

Size octets	Type	Description of String type specified
1	Integer	Fixed size integer field. In this example the integer is of size 8 bits (1 octets)
2	Integer	Fixed size integer field. In this example the integer is of size 16 bits (2 octets)
4	Integer	Fixed size integer field. In this example the integer is of size 32 bits (4 octets)
Var Max 16	C-Octet String	This string is of variable length from 1-15 ASCII characters, followed by an octet containing the NULL terminator. An empty string is encoded as a single octet containing the NULL character (0x00).
Fixed 1 or 17	C-Octet String	This string has two possible lengths: 1 octet containing the NULL character or a fixed number of characters terminated with the NULL character (in this example 16 characters plus the NULL character).
Var 0 - 254	Octet String	Variable size octet string field. In this example the size of the octet string field can vary from 0 to 254 octets.

Table 3-3 SMPP PDU Parameter Type Size Notation

3.2 General PDU Format

The general format of an SMPP PDU consists of a PDU header followed by a body as outlined in the following:

SMPP PDU				
PDU Header (mandatory)				PDU Body (Optional)
Command length	Command id	Command status	Sequence number	PDU Body
4 octets	4 octets	4 octets	4 octets	Length = (Command Length value - 16) octets

Table 3-4 SMPP PDU Format

The 16-0ctet SMPP Header is a mandatory part of every SMPP PDU and must always be present. The SMPP PDU Body is optional and may not be included with every SMPP PDU.

3.2.1 PDU Format

	SMPP PDU Field	Size (Octets)	Type	Description
PDU HEADER	*command_length*	4	Integer	Overall size of PDU including header and body
	command_id	4	Integer	Identifies the PDU
	command_status	4	Integer	Used to carry an SMPP error code
	sequence_number	4	Integer	Used to uniquely identify an SMPP PDU in the context of an SMPP session
BODY	Standard Parameters	var.	mixed	The Body part of a PDU differs from PDU to PDU and in some cases, there is no body at all.
	TLV Parameters	var.	mixed	

Table 3-5 SMPP PDU Format

The layout of every SMPP PDU consists of a mandatory 16 octets representing the PDU header. The above fields are now explained in detail:

3.2.1.1 Command_length

SMPP is a binary protocol and also supports asynchronous transmission. This means that an ESME or MC must support the ability to decode a PDU from a network connection buffer that may contain several PDUs and not just one. The key to achieving the means of decoding each PDU within a buffer is based on the knowledge of how big each PDU is. The *command_length* represents the first field of a PDU and its value contains the overall size of the PDU. An application can easily decode a PDU from a buffer by extracting the first 4 octets, assuming these to represent the *command_length* and then deduce from the value, the overall size of the PDU. Considering that 4 octets have been read from the buffer, the deduced value is decremented by 4 to indicate the remaining PDU data to extract from the buffer.

An alternative approach to this is to view each SMPP PDU as a mandatory block of 16 octets representing the PDU header. So an application wishing to decode a PDU for processing must wait until there are at least 16 octets available in the network connection buffer or continually read octets of data until 16 octets have been read. This can now be broken down into the four 4-octet fields that make up the PDU Header. By subtracting 16 from *command_length* value, the application can evaluate the size of the PDU Body and use the same means of reading data form its buffers until the remaining data has been received.

3.2.1.2 Command_id

The *command_id* identifies the SMPP operation and although we refer to these operations by name in the form of *submit_sm*, *bind_transmitter* etc., in PDU format, these operations are represented by numerical ids.

Command_ids for request PDUs are allocated from a range of numbers; 0x00000000 to 0x000001FF.

Command_ids for response PDUs are allocated from a range of numbers; 0x80000000 to 0x800001FF.

The relationship between the *command_id* for a request PDU and its associated response PDU is that bit 31 is cleared for the request and set for the response. For example, *replace_sm* has a *command_id* = 0x**0**0000007 and is response PDU *replace_sm_resp* has a *command_id* = 0x**8**0000007.

3.2.1.3 Command_status

The *command_status* represents the means by which an ESME or MC sends an error code to its peer. This field is only relevant in response PDUs. There is no sense in issuing a request to submit a message and at the same time include an error code with the request. So every PDU request always has this field set to NULL (0x00000000).

Note: When a response PDU carries a non-NULL *command_status* field, it is indicating some form of error or rejection of the original request PDU. In such circumstances, a PDU body should not be included in the PDU and the *command_length* of the PDU should therefore be set to 16 (0x00000010). However some ESMEs or Message Centers may always include a PDU body regardless of the *command_status* being returned. In such circumstances, the receiving ESME or MC should decode the PDU body but ignore its contents, based on the knowledge that the original request failed.

3.2.1.4 Sequence_number

Sequence_number as already described in section 2.5 represents a means of uniquely identifying each PDU within an SMPP session. It also provides a means of correlating request and response PDUs based on matching sequence number.

3.2.1.5 Standard Parameters

Standard Parameters consist of combinations of Integer, C-Octet Strings and Octet-Strings that form the basic layout of any PDU body. These fields are always present even if specified in NULL form.

3.2.1.6 TLV Parameters

Tagged Length Value (TLV) parameters as described in section 3.1 are identified by a tag, length and value and can be appended to a PDU in any order. The only requirement is that the PDU's standard fields are first encoded, and then followed by the TLV parameters. Otherwise, the PDU decoding by the peer would be unable to decode the PDU.

TLVs were originally referred to as Optional Parameters in SMPP V3.4. In Version 5.0 of the protocol, the term Optional Parameter applies only to TLVs that are not mandatory within a PDU. Therefore TLVs are described in the context of a PDU as Mandatory TLVs and Optional TLVs. The Term Optional Parameter is no longer used.

The reason for the change of terminology is that some new PDUs now feature TLVs as mandatory fields that must always be present. In this context, generally referring to these fields as Optional Parameters, is misleading, hence the change of terminology.

3.2.2 A sample PDU

The following PDU example illustrates how an SMPP PDU is decoded:

Sample PDU (Values are shown in Hex format):

```
00 00 00 2F 00 00 00 02 00 00 00 00 00 00 00 01
53 4D 50 50 33 54 45 53 54 00 73 65 63 72 65 74
30 38 00 53 55 42 4D 49 54 31 00 00 01 01 00
```

The 16-octet header would be decoded as follows:

`00 00 00 2F`	Command Length	0x0000002F
`00 00 00 02`	Command ID	0x00000002 (*bind_transmitter*)
`00 00 00 00`	Command Status	0x00000000
`00 00 00 01`	Sequence Number	0x00000001

The remaining data represents the PDU body (which in this example relates to the *bind_transmitter* PDU).

This is diagnosed as follows:

`53 4D 50 50 33 54 45 53 54 00`	System_id (“SMPP3TEST”)
`73 65 63 72 65 74 30 38 00`	System_type (“secret08”)
`53 55 42 4D 49 54 31 00`	Password (“SUBMIT1”)
`00`	Interface_version (0x00)
`01`	addr_ton (0x01)
`01`	addr_npi (0x01)
`00`	addr_range (NULL)

4 SMPP PDU Definitions

This section defines the various Operation PDUs that make up the SMPP protocol. The Operations are described in 4 categories: Session Management, Message Submission, Message Delivery and Anciliary Operations.

4.1 Session Management Operations

4.1.1 Bind Operation

The purpose of the SMPP bind operation is to register an instance of an ESME with the SMSC system and request an SMPP session over this network connection for the submission or delivery of messages. Thus, the Bind operation may be viewed as a form of SMSC login request to authenticate the ESME entity wishing to establish a connection.

As described previously, an ESME may bind to the SMSC as a Transmitter (called ESME Transmitter), a Receiver (called ESME Receiver) or a Transceiver (called ESME Transceiver). There are three SMPP bind PDUs to support the various modes of operation, namely *bind_transmitter*, *bind_transceiver* and *bind_receiver*. The *command_id* field setting specifies which PDU is being used.

An ESME may bind as both an SMPP Transmitter and Receiver using separate *bind_transmitter* and *bind_receiver* operations (having first established two separate network connections). Alternatively an ESME can also bind as a Transceiver having first established a single network connection.

4.1.1.1 *bind_transmitter* Syntax

The format of the SMPP *bind_transmitter* PDU is defined in the following table:

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Defines the overall length of the bind_transmitter PDU.	4.5.1
command_id	4	Integer	0x00000002	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to a unique sequence number. The associated bind_transmitter_resp PDU will echo the same sequence number.	4.5.4
system_id	Var. max 16	C-Octet String	Identifies the ESME system requesting to bind as a transmitter with the SMSC.	4.5.5
password	Var. max 9	C-Octet String	The password may be used by the SMSC to authenticate the ESME requesting to bind.	4.5.6
system_type	Var. max 13	C-Octet String	Identifies the type of ESME system requesting to bind as a transmitter with the SMSC.	4.5.7
interface_version	1	Integer	Indicates the version of the SMPP protocol supported by the ESME.	4.5.8
addr_ton	1	Integer	Indicates Type of Number of the ESME address. If not known set to NULL	4.5.9
addr_npi	1	Integer	Numbering Plan Indicator for ESME address. If not known set to NULL.	4.5.9
address_range	Var. max 41	C- Octet String	The ESME address. If not known set to NULL.	4.5.11

Table 4-1 *bind_transmitter* PDU

4.1.1.2 *bind_transmitter_resp* Syntax

The SMPP *bind_transmitter_resp* PDU is used to reply to a bind_transmitter request. The format of the SMPP *bind_transmitter_resp* PDU is defined in the following table.

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Defines the overall length of the bind_transmitter_resp PDU.	4.5.1
command_id	4	Integer	0x80000002	4.5.2
command_status	4	Integer	Indicates status (success or error code) of original bind_transmitter request.	4.5.3
sequence_number	4	Integer	Set to sequence number of original bind_transmitter request.	4.5.4
system_id	Var. max 16	C- Octet String	SMSC identifier. Identifies the SMSC to the ESME.	4.5.5
Optional TLVs:				
sc_interface_version		TLV	SMPP version supported by SMSC	4.6.4.25

Table 4-2 *bind_transmitter_resp* PDU

4.1.1.3 *bind_receiver* Syntax

The format of the SMPP *bind_receiver* PDU is defined in the following table.

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Defines the overall length of the PDU in octets.	4.5.1
command_id	4	Integer	0x00000001	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to a unique sequence number. The associated bind_receiver_resp PDU will echo the same sequence number.	4.5.4
system_id	Var. max 16	C- Octet String	Identifies the ESME system requesting to bind as a receiver with the SMSC.	4.5.5
password	Var. max 9	C- Octet String	The password may be used by the SMSC for security reasons to authenticate the ESME requesting to bind.	4.5.6
system_type	Var. max 13	C- Octet String	Identifies the type of ESME system requesting to bind as a receiver with the SMSC.	4.5.7
interface_version	1	Integer	Identifies the version of the SMPP protocol supported by the ESME.	4.5.8
addr_ton	1	Integer	Type of Number (TON) for ESME address(es) served via this SMPP receiver session. Set to NULL if not known.	4.5.9
addr_npi	1	Integer	Numbering Plan Indicator (NPI) for ESME address(es) served via this SMPP receiver session. Set to NULL if not known.	4.5.9
address_range	Var. max 41	C- Octet String	A single ESME address or a range of ESME addresses served via this SMPP receiver session. The parameter value is represented in UNIX regular expression format (see Appendix A). Set to NULL if not known.	4.5.11

Table 4-3 *bind_receiver* PDU

4.1.1.4 *bind_receiver_resp* Syntax

The format of the SMPP *bind_receiver_resp* PDU is defined in the following table.

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Defines the overall length of the PDU.	4.5.1
command_id	4	Integer	0x80000001	4.5.2
command_status	4	Integer	Indicates status (success or error code) of original bind_receiver request.	4.5.3
sequence_number	4	Integer	Set to sequence number of original bind_receiver request.	4.5.4
system_id	Var. max 16	C- Octet String	SMSC identifier. Identifies the SMSC to the ESME.	4.5.5
Optional TLVs:				
Optional Parameter Name		Type	Description	
sc_interface_version		TLV	SMPP version supported by SMSC	4.6.4.25

Table 4-4 *bind_receiver_resp* PDU

4.1.1.5 *bind_transceiver* Syntax

The format of the SMPP *bind_transceiver* PDU is defined in the following table.

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Defines the overall length of the PDU.	4.5.1
command_id	4	Integer	0x00000009	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to a unique sequence number. The associated bind_transceiver_resp PDU will echo the same sequence number.	4.5.4
system_id	Var. max 16	C- Octet String	Identifies the ESME system requesting to bind as a transceiver with the SMSC.	4.5.5
password	Var. max 9	C- Octet String	The password may be used by the SMSC to authenticate the ESME requesting to bind.	4.5.6
system_type	Var. max 13	C- Octet String	Identifies the type of ESME system requesting to bind as a transceiver with the SMSC.	4.5.7
interface_version	1	Integer	Identifies the version of the SMPP protocol supported by the ESME.	4.5.8
addr_ton	1	Integer	Type of Number (TON) for ESME address(es) served via this SMPP transceiver session. Set to NULL (Unknown) if not known.	4.5.9
addr_npi	1	Integer	Numbering Plan Indicator (NPI) for ESME address(es) served via this SMPP transceiver session. Set to NULL (Unknown) if not known.	4.5.9
address_range	Var. max 41	C- Octet String	A single ESME address or a range of ESME addresses served via this SMPP transceiver session. This field may be used by the SMSC for authentication, verification or routing purposes. Set to NULL if not known.	4.5.11

Table 4-5 *bind_transceiver* PDU

4.1.1.6 *bind_transceiver_resp* Syntax

The format of the SMPP *bind_transceiver_resp* PDU is defined in the following table.

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Defines the overall length of the PDU.	4.5.1
command_id	4	Integer	0x80000009	4.5.2
command_status	4	Integer	Indicates status (success or error code) of original bind_transceiver request.	4.5.3
sequence_number	4	Integer	Set to sequence number of original bind_transceiver request.	4.5.4
system_id	Var. max 16	C-Octet String	SMSC identifier. Identifies the SMSC to the ESME.	4.5.5
Optional TLVs:				
Optional Parameter Name		Type	Description	
sc_interface_version		TLV	SMPP version supported by SMSC	4.6.4.25

Table 4-6 *bind_transceiver_resp* PDU

4.1.1.7 *outbind* Syntax.

This operation is used by the SMSC to signal an ESME to originate a ***outbind*** request to the SMSC.The format of the SMPP ***outbind*** PDU is defined in the following table.

Field Name	Size Octets	Type	Description	Ref.
command_length	4	Integer	Defines the overall length of the PDU.	4.5.1
command_id	4	Integer	0x0000000B	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to a unique sequence number.	4.5.4
system_id	Var. max 16	C- Octet String	SMSC identifier. Identifies the SMSC to the ESME.	4.5.5
Password	Var. max 9	C- Octet String	The password may be used by the ESME for security reasons to authenticate the SMSC originating the outbind.	4.5.6

Table 4-7*outbind PDU*

4.1.1.8 *unbind* Syntax

The purpose of the SMPP *unbind* operation is to deregister an instance of an ESME from the SMSC and inform the SMSC that the ESME no longer wishes to use this network connection for the submission or delivery of messages.

Thus, the *unbind* operation may be viewed as a form of SMSC logoff request to close the current SMPP session. The format of the SMPP ***unbind*** PDU is defined in the following table:

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Defines the overall length of the PDU.	4.5.1
command_id	4	Integer	0x00000006	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to a unique sequence number. The associated unbind_resp PDU will echo the same sequence number.	4.5.4

Table 4-8 *unbind* PDU

4.1.1.9 *unbind_resp* Syntax

The SMPP *unbind_resp* PDU is used to reply to an *unbind* request. It comprises the SMPP message header only.

The format of the SMPP *unbind_resp* PDU is defined in the following table:

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Defines the overall length of the PDU.	4.5.1
command_id	4	Integer	0x80000006	4.5.2
command_status	4	Integer	Indicates outcome of original unbind request.	4.5.3
sequence_number	4	Integer	Set to sequence number of original unbind request.	4.5.4

Table 4-9 *unbind_resp* PDU

4.1.2 Enquire Link Operation

This message can be sent by either the ESME or SMSC and is used to provide a confidence-check of the communication path between an ESME and an SMSC. On receipt of this request the receiving party should respond with an *enquire_link_resp*, thus verifying that the application level connection between the SMSC and the ESME is functioning. The ESME may also respond by sending any valid SMPP primitive.

4.1.2.1 *Enquire_link* Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU	4.5.1
command_id	4	Integer	0x00000015	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to a unique sequence number. The associated enquire_link_resp PDU should echo the same sequence number	4.5.4

Table 4-10 *enquire_link* PDU

4.1.2.2 *Enquire_link_resp* Syntax

The *enquire_link_resp* PDU is used to reply to an *enquire_link* request.

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU	4.5.1
command_id	4	Integer	0x80000015	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to the same sequence number of original enquire_link PDU	4.5.4

Table 4-11 *enquire_link_resp* PDU

4.1.3 Generic NACK Operation

4.1.3.1 *generic_nack* Syntax

Following is the format of the SMPP *generic_nack* PDU*.* It comprises the SMPP message header only*.*

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Defines the overall length of the PDU.	4.5.1
command_id	4	Integer	0x80000000	4.5.2
command_status	4	Integer	Error code corresponding to reason for sending the generic_nack.	4.5.3
sequence_number	4	Integer	Set to sequence number of original PDU or to NULL if the original PDU cannot be decoded.	4.5.4

Table 4-12 *generic_nack* PDU

4.2 Message Submission Operations

4.2.1 *submit_sm* Operation

This operation is used by an ESME to submit a short message to the SMSC for onward transmission to a specified short message entity (SME).

4.2.1.1 *submit_sm* Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU.	4.5.1
command_id	4	Integer	0x00000004	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to a Unique sequence number. The associated submit_sm_resp PDU will echo this sequence number.	4.5.4
service_type	Var. max 6	C- Octet String	The service_type parameter can be used to indicate the SMS Application service associated with the message. Specifying the service_type allows the ESME to avail of enhanced messaging services such as “replace by service” type to control the teleservice used on the air interface. Set to NULL for default SMSC settings	4.5.15
source_addr_ton	1	Integer	Type of Number for source address. If not known, set to NULL (Unknown).	4.5.9
source_addr_npi	1	Integer	Numbering Plan Indicator for source address. If not known, set to NULL (Unknown).	4.5.10
source_addr	Var. max 21	C- Octet String	Address of SME which originated this message. If not known, set to NULL (Unknown).	4.5.12
dest_addr_ton	1	Integer	Type of Number for destination	4.5.9
dest_addr_npi	1	Integer	Numbering Plan Indicator for destination	4.5.10

Field Name	Size octets	Type	Description	Ref.
destination_addr	Var. max 21	C-Octet String	Destination address of this short message For mobile terminated messages, this is the directory number of the recipient MS	4.5.13
esm_class	1	Integer	Indicates Message Mode and Message Type	4.5.16
protocol_id	1	Integer	Protocol Identifier. Network specific field.	4.5.17
priority_flag	1	Integer	Designates the priority level of the message	4.5.18
schedule_delivery_time	1 or 17	C-Octet String	The short message is to be scheduled by the SMSC for delivery. Set to NULL for immediate message delivery	4.5.19.1
validity_period	1 or 17	C- Octet String	The validity period of this message. Set to NULL to request the SMSC default validity period	4.5.19.2
registered_delivery	1	Integer	Indicator to signify if an SMSC delivery receipt or an SME acknowledgement is required.	4.5.20
replace_if_present_flag	1	Integer	Flag indicating if submitted message should replace an existing message.	4.5.21
data_coding	1	Integer	Defines the encoding scheme of the short message user data.	4.5.22
sm_default_msg_id	1	Integer	Indicates the short message to send from a list of pre- defined (‘canned’) short messages stored on the SMSC. If not using an SMSC canned message, set to NULL.	4.5.23
sm_length	1	Integer	Length in octets of the short_message user data.	4.5.24

Field Name	Size octets	Type	Description	Ref.
short_message	Var. 0-255	Octet String	Up to 255 octets of short message user data. The exact physical limit for short_message size may vary according to the underlying network Applications which need to send messages longer than 254 octets should use the message_payload parameter. In this case the sm_length field should be set to zero	4.5.25
Message Submission TLVs	Var.	TLV		4.2.4

Table 4-13 *submit_sm* PDU

4.2.1.2 submit_sm_resp Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU.	4.5.1
command_id	4	Integer	0x80000004	4.5.2
command_status	4	Integer	Indicates outcome of submit_sm request.	4.5.3
sequence_number	4	Integer	Set to sequence number of original submit_sm PDU.	4.5.4
message_id	Var. max 65	C- Octet String	This field contains the SMSC message ID of the submitted message. It may be used at a later stage to query the status of a message, cancel or replace the message.	4.5.26
Message Submission Response TLVs	Var.	TLV		4.2.5

Table 4-14 *submit_sm_resp* PDU

4.2.2 data_sm Operation

The *data_sm* operation is similar to the *submit_sm* in that it provides a means to submit a mobile-terminated message. However, *data_sm* is specifically geared towards packet-based applications such as WAP in that it features a reduced PDU body containing fields relevant to WAP or packet-based applications.

4.2.2.1 *data_sm* Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU.	4.5.1
command_id	4	Integer	0x00000103	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to a Unique sequence number. The associated submit_sm_resp PDU will echo this sequence number.	4.5.4
service_type	Var. max 6	C- Octet String	The service_type parameter can be used to indicate the SMS Application service associated with the message. Specifying the service_type allows the ESME to avail of enhanced messaging services such as “replace by service” type to control the teleservice used on the air interface. Set to NULL for default SMSC settings	4.5.15
source_addr_ton	1	Integer	Type of Number for source address. If not known, set to NULL (Unknown).	4.5.9
source_addr_npi	1	Integer	Numbering Plan Indicator for source address. If not known, set to NULL (Unknown).	4.5.10
source_addr	Var. max 21	C- Octet String	Address of SME which originated this message. If not known, set to NULL (Unknown).	4.5.12
dest_addr_ton	1	Integer	Type of Number for destination	4.5.9
dest_addr_npi	1	Integer	Numbering Plan Indicator for destination	4.5.10
destination_addr	Var. max 21	C-Octet String	Destination address of this short message For mobile terminated messages, this is the directory number of the recipient MS	4.5.13

Field Name	Size octets	Type	Description	Ref.
esm_class	1	Integer	Indicates Message Mode and Message Type	4.5.16
registered_delivery	1	Integer	Indicator to signify if an SMSC delivery receipt or an SME acknowledgement is required.	4.5.20
data_coding	1	Integer	Defines the encoding scheme of the short message user data.	4.5.22
Message Submission TLVs	Var.	TLV		4.2.4

Table 4-15 *data_sm* PDU

4.2.2.2 data_sm_resp Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU.	4.5.1
command_id	4	Integer	0x80000103	4.5.2
command_status	4	Integer	Indicates outcome of submit_sm request.	4.5.3
sequence_number	4	Integer	Set to sequence number of original submit_sm PDU.	4.5.4
message_id	Var. max 65	C- Octet String	This field contains the SMSC message ID of the submitted message. It may be used at a later stage to query the status of a message, cancel or replace the message.	4.5.26
Message Submission Response TLVs	Var.	TLV		4.2.5

Table 4-16 *data_sm_resp* PDU

4.2.3 submit_multi Operation

The *submit_multi* operation is an enhanced variation of *submit_sm* designed to support up to 255 different destinations instead of the default single destination. It provides an efficient means of sending the same message to several different subscribers at the same time.

4.2.3.1 submit_multi Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU.	4.5.1
command_id	4	Integer	0x00000021	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to a Unique sequence number. The associated submit_sm_resp PDU will echo this sequence number.	4.5.4
service_type	Var. max 6	C-Octet String	The service_type parameter can be used to indicate the SMS Application service associated with the message. Specifying the service_type allows the ESME to avail of enhanced messaging services such as “replace by service” type to control the teleservice used on the air interface. Set to NULL for default SMSC settings	4.5.15
source_addr_ton	1	Integer	Type of Number for source address. If not known, set to NULL (Unknown).	4.5.9
source_addr_npi	1	Integer	Numbering Plan Indicator for source address. If not known, set to NULL (Unknown).	4.5.10
source_addr	Var. max 21	C-Octet String	Address of SME which originated this message. If not known, set to NULL (Unknown).	4.5.12

number_of_dests	1	Integer	Number of destination addresses – indicates the number of destinations that are to follow. A maximum of 255 destination addresses are allowed. Note: Set to 1 when submitting to one SME Address or when submitting to one Distribution List.	
dest_address	Var. max 24		SME Format Destination Address (Composite field)	
dest_flag	1	Integer	0x01 (SME Address)	
dest_addr_ton	1	Integer	Type of Number for destination	4.5.9
dest_addr_npi	1	Integer	Numbering Plan Indicator for destination	4.5.10
destination_addr	Var. max 21	C-Octet String	Destination address of this short message For mobile terminated messages, this is the directory number of the recipient MS	4.5.13
dest_address	Var. max 22		Distribution List Format Destination Address (Composite Field)	
dest_flag	1	Integer	0x02 (Distribution List	
dl_name	Var. max 21	C-Octet String	Name of Distribution List	
dest_address	…	…	Additional destination_address fields can be added as delimited by the *number_of_dests* field	…
esm_class	1	Integer	Indicates Message Mode and Message Type	4.5.16
protocol_id	1	Integer	Protocol Identifier. Network specific field.	4.5.17
priority_flag	1	Integer	Designates the priority level of the message	4.5.18
schedule_delivery_time	1 or 17	C-Octet String	The short message is to be scheduled by the SMSC for delivery. Set to NULL for immediate message delivery	4.5.19.1
validity_period	1 or 17	C-Octet String	The validity period of this message. Set to NULL to request the SMSC default validity period	4.5.19.2
registered_delivery	1	Integer	Indicator to signify if an SMSC delivery receipt or an SME acknowledgement is required.	4.5.20

replace_if_present_flag	1	Integer	Flag indicating if submitted message should replace an existing message.	4.5.21
data_coding	1	Integer	Defines the encoding scheme of the short message user data.	4.5.22
sm_default_msg_id	1	Integer	Indicates the short message to send from a list of pre- defined ('canned') short messages stored on the SMSC. If not using an SMSC canned message, set to NULL.	4.5.23
sm_length	1	Integer	Length in octets of the short_message user data.	4.5.24
short_message	Var. 0-255	Octet String	Up to 255 octets of short message user data. The exact physical limit for short_message size may vary according to the underlying network Applications which need to send messages longer than 254 octets should use the message_payload parameter. In this case the sm_length field should be set to zero	4.5.25
Message Submission TLVs	Var.	TLV		4.2.4

Table 4-17 *submit_multi* PDU

4.2.3.2 submit_multi_resp Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU.	4.5.1
command_id	4	Integer	0x80000021	4.5.2
command_status	4	Integer	Indicates outcome of *submit_multi* request.	4.5.3
sequence_number	4	Integer	Set to sequence number of original *submit_multi* PDU.	4.5.4
message_id	Var. max 65	C-Octet String	This field contains the SMSC message ID of the submitted message. It may be used at a later stage to query the status of a message, cancel or replace the message.	4.5.26
no_unsuccess	1	Integer	The number of messages to destination SME addresses that were unsuccessfully submitted to the SMSC. This is followed by the specified number of unsuccessful SMEs, each specified in a unsuccess_sme field.	
unsuccess_sme	Var. max 27		Unsucessful SME (Composite Field)	
dest_addr_ton	1	Integer	Type of number for destination	
dest_addr_npi	1	Integer	Numbering Plan Indicator for SME	
destination_addr	Var. max 21	C-Octet String	Destination Address of SME	
error_status_code	4	Integer	Indicates the success or failure of the *submit_multi* request to this SME address	
unsucess_sme	…	…	Additional unsuccess_sme fields can be specified as delimited by the *no_unsuccess* field.	…
Message Submission Response TLVs	Var.	TLV		4.2.5

Table 4-18 *submit_multi_resp* PDU

4.2.4 Message Submission Request TLVs

This section lists TLVs that may be used for message submission operations.

TLV Name	Description	Ref.
user_message_reference	ESME assigned message reference number.	4.6.4.17
source_port	Indicates the application port number associated with the source address of the message. This parameter should be present for WAP applications.	4.6.4.20
source_addr_subunit	The subcomponent in the destination device, which created the user data.	4.6.4.2
destination_port	Indicates the application port number associated with the destination address of the message. This parameter should be present for WAP applications.	4.6.4.21
dest_addr_subunit	The subcomponent in the destination device for which the user data is intended.	4.6.4
sar_msg_ref_num	The reference number for a particular concatenated short message.	4.6.4.22
sar_total_segments	Indicates the total number of short messages within the concatenated short message.	4.6.4.23
sar_segment_seqnum	Indicates the sequence number of a particular short message fragment within the concatenated short message.	4.6.4.24
more_messages_to_send	Indicates that there are more messages to follow for the destination SME.	4.6.4.34
qos_time_to_live	Time to live as a relative time in seconds from submission.	4.6.4.9
payload_type	defines the type of payload (e.g. WDP, WCMP, etc.).	4.6.4.10
message_payload	Contains the extended short message user data. Up to 64K octets can be transmitted. Note: The short message data should be inserted in either the *short_message* or *message_payload* fields. Both fields should not be used simultaneously. The *sm_length* field should be set to zero if using the *message_payload* parameter. Note: In the case of *data_sm*, the *message_payload* TLV is the only means of specifying text.	4.6.4.32
set_dpf	Indicator for setting Delivery Pending Flag on delivery failure.	4.6.4.29
privacy_indicator	Indicates the level of privacy associated with the message.	4.6.4.14
callback_num	A callback number associated with the short message. This parameter can be included a number of times for multiple callback addresses.	4.6.4.36

TLV Name	Description	Ref.
callback_num_pres_ind	Defines the callback number presentation and screening. If this parameter is present and there are multiple instances of the *callback_num* parameter then this parameter must occur an equal number of instances and the order of occurrence determines the particular *callback_num_pres_ind* which corresponds to a particular *callback_num*.	4.6.4.37
callback_num_atag	Associates a displayable alphanumeric tag with the callback number. If this parameter is present and there are multiple instances of the *callback_num* parameter then this parameter must occur an equal number of instances and the order of occurrence determines the particular *callback_num_atag* which corresponds to a particular *callback_num*.	4.6.4.38
source_subaddress	The sub-address of the message originator.	4.6.4.15
dest_subaddress	The sub-address of the message destination.	4.6.4.16
user_response_code	A user response code. The actual response codes are implementation specific.	4.6.4.18
display_time	Provides the receiving MS with a display time associated with the message.	4.6.4.26
sms_signal	Indicates the alerting mechanism when the message is received by an MS.	4.6.4.40
ms_validity	Indicates validity information for this message to the recipient MS.	4.6.4.27
ms_msg_wait_facilities	This parameter controls the indication and specifies the message type (of the message associated with the MWI) at the mobile station.	4.6.4.13
number_of_messages	Indicates the number of messages stored in a mail box	4.6.4.39
alert_on_msg_delivery	Request an MS alert signal be invoked on message delivery.	4.6.4.41
language_indicator	Indicates the language of an alphanumeric text message.	4.6.4.19
its_reply_type	The MS user's reply method to an SMS delivery message received from the network is indicated and controlled by this parameter.	4.6.4.42
its_session_info	Session control information for InteractiveTeleservice.	4.6.4.43
ussd_service_op	This parameter is used to identify the required USSD Service type when interfacing to a USSD system.	4.6.4.44

Table 4-19 Message Submission Request TLVs

4.2.5 Message Submission Response TLVs

The following table contains TLVs that can be returned in a *submit_sm_resp* or *data_sm_resp* PDU. All TLVs are relevant to transaction message mode only (ref. 4.2.10.4)

TLV Name	Description	Ref.
delivery_failure_reason	Include to indicate reason for delivery failure.	4.6.4.33
network_error_code	Error code specific to a wireless network.	4.6.4.31
additional_status_info_text	ASCII text giving a description of the meaning of the response	4.6.4.11
dpf_result	Indicates whether the Delivery Pending Flag was set.	4.6.4.28

Table 4-20 Message Submision Response TLVs

4.2.6 Source and Destination Addressing

The *submit_sm* and *data_*sm PDUs include provision for both 'source address' (message sender) and 'destination address' (message recipient). The common concept of a source or destination address is a sequence of digits representing a mobile or fixed-line number. However the reality is more complex. Both addresses comprise of three parts, namely the TON (Type of Number), NPI (Numbering Plan Indicator) and Address (Digit sequence). Every time a message is sent either to or from a mobile, the source and destination addresses comprise of the three parts. This three-part relationship is visible in the PDUs in the form of *source_addr_ton*, *source_addr_npi*, *source_addr*, *dest_addr_ton*, *dest_addr_npi* and *destination_addr* fields.

4.2.6.1 TON

TON is used to specify the number type. While there are several values defined for TON, the most common values are:

- National
  A national number consists of a mobile or fixed line operators prefix, followed by the mobile number itself. When a mobile originates a message to a national number, the TON value of the address is usually set to 2.
- International
  International format from a mobile usually involves an international dial-out prefix or the '+' symbol followed by the country code, operator code and mobile number. International numbers usually carry a TON value of 1. However it is important to note that the '+' character is never actually encoded in the *source_addr* or *dest_addr* fields. For example, a mobile sending a message to +4467811223344 is in fact using a TON=1 and dest_addr = 4467811223344
- Alphanumeric
  Alphanumeric addressing provides a means of using human-readable names for addresses. In SMPP, an alphanumeric address can carry any digit 0-9 and alphabetical character a-z or A-Z. For example a voice mail server may send "VoiceMail" as a alphanumeric source address and as a means of indicating this, it will set the TON=5. Some mobiles are capable of sending alphanumeric numbers and accomplish this by means of a TON value of 5.

4.2.6.2 NPI

NPI is generally set to 1 by all mobile devices. Its purpose is to specify the numbering plan of the target device, but because these all tend to be mobiles, the value is generally set to 1.

4.2.6.3 ESME Addresses

Specifying addresses for ESMEs is very much prone to the network operator's preferences and the message center implementation. An ESME will typically use one of the following approaches:

- Service Short Code
  Short codes are small and often easy to remember. Operators providing services that allow a subscriber avail of stock quotes, lotto results etc, will often configure the service to respond to a short digit sequence such as "1234". The subscriber will mobile originate a message addressed to "1234" (TON=2) and the message center routing will deliver this to an ESME receiver or transceiver. In responding to the message, the ESME will usually set the source address to the short code, making it easy for the recipient to reply to the message if another request is desired.
- International Number
  With more and more subscribers roaming internationally with their mobiles, ESME services that they use, need to accessible regardless of location. One approach to supporting this accessibility is for the ESME to use an international number in the source address of the message (TON=1, NPI=1).
- NULL Address
  An ESME Transmitter may enter NULL values in the 'source address' fields. In this event, the SMSC may then substitute a default address for that particular ESME. This feature is designed for interfaces that are not normally familiar with the notion of a source address for a short message, e.g., paging systems, voice mail systems which developed long before the advent of SMS technologies and were based on one-way paging, hence the lack of a source address.

4.2.7 Message Replace operation in submit_sm

Though SMPP offers a dedicated *replace_sm* operation, the *submit_sm* operation also facilitates replacement of a short message which has been previously submitted but has not yet been delivered to the designated destination.

This feature is designed for applications needing the ability to update an underlivered message. The most common application of this feature is with voicemail systems. The first message may read "You have 1 message in your mail box". If the mobile is unavailable, the message will remain undelivered and in retry within the message center. If another message is left for the same subscriber, the voicemail server will send another message with the updated text "You have 2 messages in your mail box". If the *replace_flag* of the *submit_sm* PDU is set to 1, then this message should replace the underlivered first message. The result is that the subscriber gets the latest message instead of all messages.

Alternatively, an SMSC administrator may define a specific *service_type* to provide 'replace-if-present' functionality. In this case, the replace function can be activated in the *submit_sm* PDU by setting the *service_type* field to the defined value.

For both methods of replacing a message using the *submit_sm* operation, the data contained in the short message found in the SMSC, whose source and destination addresses and *service_type* match those addresses specified in the latest submit_sm operation, will be replaced with the text contained in the *short_message* field of that latest submit_sm operation.

If the *submit_sm* PDU is used to replace an as yet undelivered message in the SMSC, and a matching message is not found in the SMSC, a new short message will be submitted to the SMSC.

If a matching message is not found when using the *replace_sm* operation, the *replace_sm* operation will not result in a new message being submitted to the SMSC - an SMPP error will be returned to the ESME in the *replace_sm_resp* PDU.

4.2.8 Message Length

Each network variation is limited to some fixed maximum length. This may be further affected by data coding scheme. In the case of GSM, a message can have a maximum length of 140 octets (8 bits each). However, if the message is being sent using the standard GSM 7-bit alphabet or one of the supported 7-bit alphabets, 160 characters can be encoded.

The ANSI-41 technologies, CDMA and TDMA, are more complicated in defining limits on the text length. Both technologies encode the text in a generic bearer data field that is also used to populate other optional fields. The restriction on text is based on the amount of optional attributes included with the message.

The SMSC, depending on configuration, may also reject or truncate messages that exceed the network allowed maximum.

4.2.9 Message Types

4.2.9.1 Registered

The *registered_delivery* field (ref. 4.5.20) allows an ESME request a delivery receipt for the message. Under normal circumstances, a receipt is typically sent to the ESME when the message reached a final delivery state, regardless of whether the message was actually delivered or not. However the *registered_delivery* field provides a number of settings that dictate the requirements for generating the receipt. One such example is the value of 2 which requests a receipt only if the message is not delivered when it reaches its final state.

The following diagram illustrates the use of registered delivery as a means of obtaining a delivery confirmation.

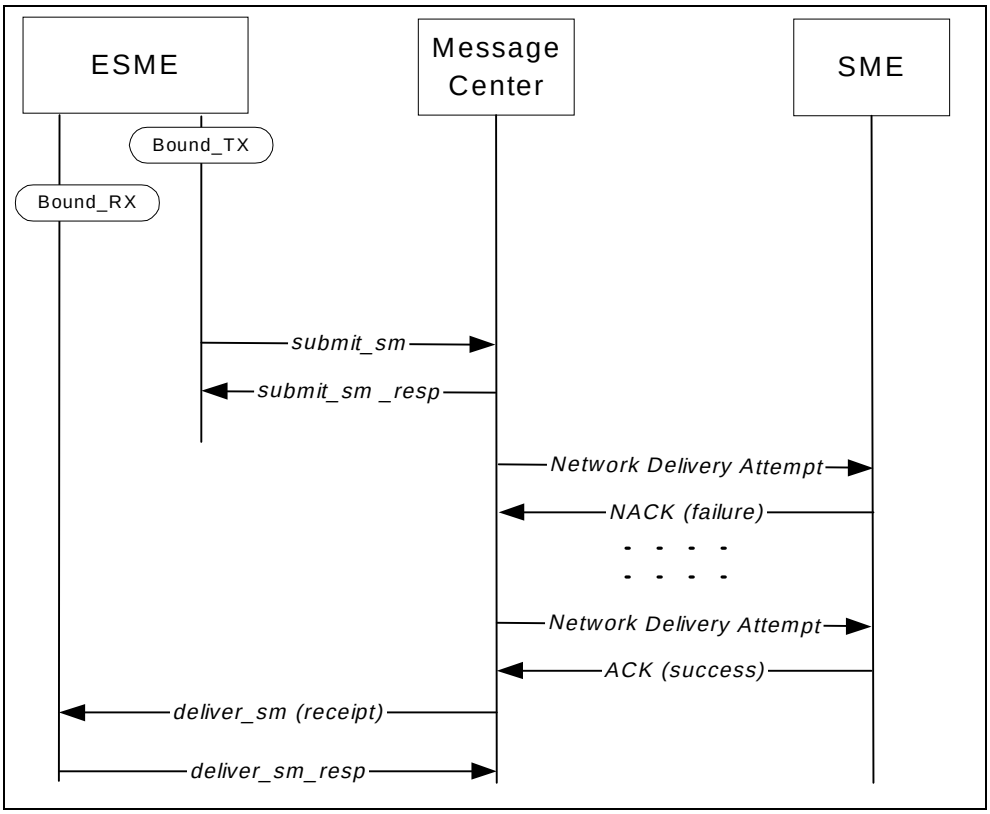


Figure 4-1 Registered Delivery

4.2.9.2 Scheduled

A scheduled message is one that is not immediately dispatched for delivery to the destination SME. Instead the scheduled date provided with the message (ref. 4.5.19.1) dictates the time that the message will become eligible for delivery. Typical uses of this feature include timed services such as news reports that a user wishes to receive at a certain time. The newsagent

ESME may submit batches of messages every few hours, each message scheduled according to the preferences of the recipient.

Another use of scheduled messages is to provide for birthday services or SMS-based reminder applications that pre populate the MC with various scheduled messages for a user.

4.2.9.3 Pre-defined

A pre-defined message is a “canned” message that is provisioned on a MC. The ESME specifies the message by providing its ID in the *default_msg_id* field (ref. 4.5.23). The purpose of the predefined message is to relieve the ESME from specifying the actual text, allowing the operator control over what goes to the subscriber.

4.2.10 Message Modes

The *esm_class* field (ref. 4.5.16) provides a Message Mode feature, which, if supported on the SMSC, allows an ESME to select the SMSC message submission/delivery mechanism. Theses options are as follows:

- Default Message Mode
  This can be any of the following three modes, but usually maps to “Store and Forward” mode
- Datagram Mode
  Message delivery is attempted only once
- Transaction Mode
  Message delivery is attempted once, delaying the *submit_sm_resp* or *data_sm_resp* until the delivery attempt has been made.
- Store and Forward Mode
  The message is stored in the MC message database.

These Message Modes are described in more detail in the following sections.

4.2.10.1 Default Message Mode

Te concept of a default message mode rates to scenarios where an ESME does not specifically request a message mode. In this scenario, bits 0-1 of the *esm_class* are both set to 0. However a message mode will apply and is usually dependent on the Message Centre and possibly deployment strategy of the operator.

4.2.10.2 Store and Forward Message Mode

The conventional approach to SMS has been to store the message in a MC storage area (e.g. message database) before forwarding the message for delivery to the recipient SME. With this model, the message remains securely stored until the MC has made all delivery attempts. This mode of messaging is commonly referred to as "store and forward".

The following diagram illustrates store and forward mode. The message is first accepted by the Message Center and acknowledged with the *submit_sm_resp* PDU that is returned to the ESME. The message center then makes an attempt to deliver the message, which fails due to some network error. Several retry attempts may occur until the message is finally delivered to the subscriber. Store and forward is based on the concept of giving the message to the MC and relying on the MC to do a “best effort” attempt to deliver the message to its destination.

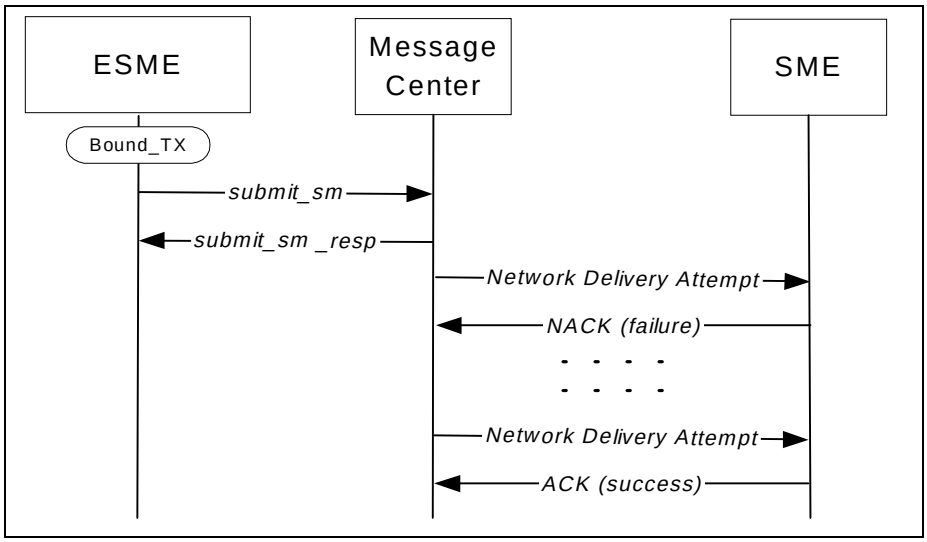


Figure 4-2 Store and Forward Mode

4.2.10.3 Datagram Message Mode

The Datagram Message Mode emulates the datagram paradigm used in other data communication protocols such as UDP datagram packet transfer and focuses on high message throughput without the associated secure storage and retry guarantees of Store and Forward Message Mode. In Datagram Message Mode the message originator (i.e. the ESME) does not receive any form of delivery acknowledgement.

In Datagram Message Mode, typical SMSC functions such as scheduled delivery, registered delivery etc. do not apply. Datagram Message Mode is designed for high throughput applications that may not require the highly secure delivery functionality offered by the Store and Forward message mode. It is ideally suited for applications where the data content is transient in nature, for example, vehicle tracking applications.

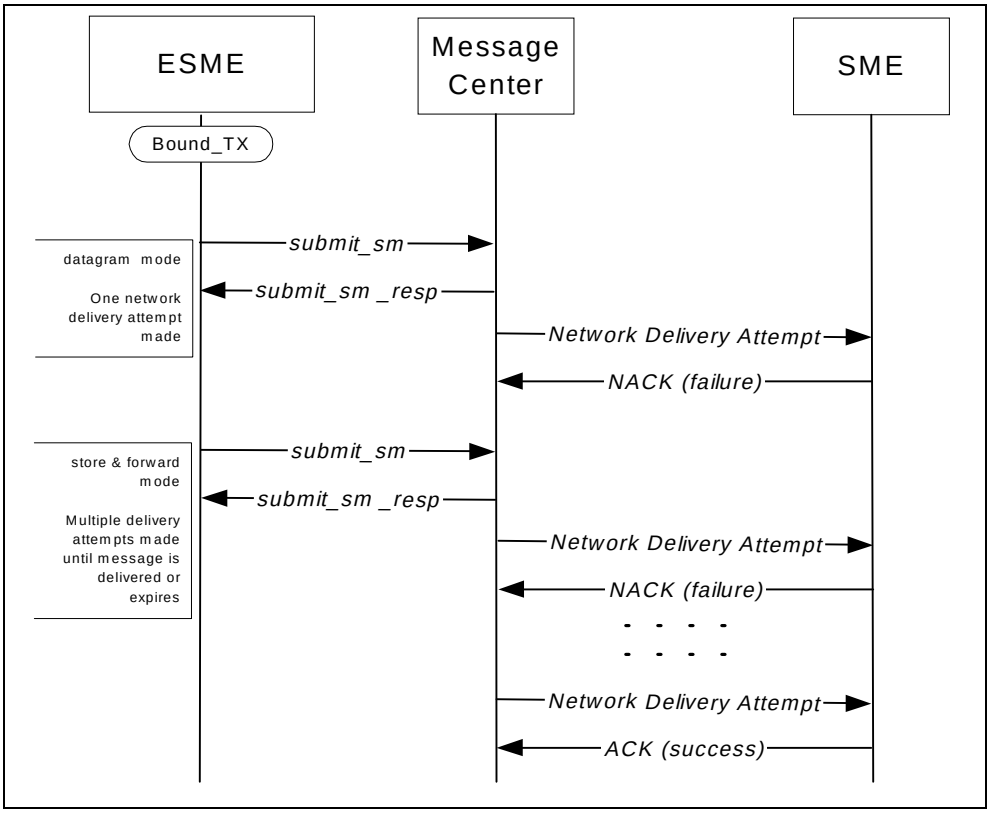


Figure 4-3 Datagram Message Mode

4.2.10.4 Transaction Message Mode

Transaction Message Mode allows the ESME message originator to receive a form of delivery acknowledgment (that indicates if the message has been successfully or unsuccessfully delivered to the destination MS) within the SMPP response PDU.

Transaction Message Mode is designed for applications that involve real-time messaging where an ESME requires a synchronous end-to-end delivery outcome, without the need for long term SMSC storage.

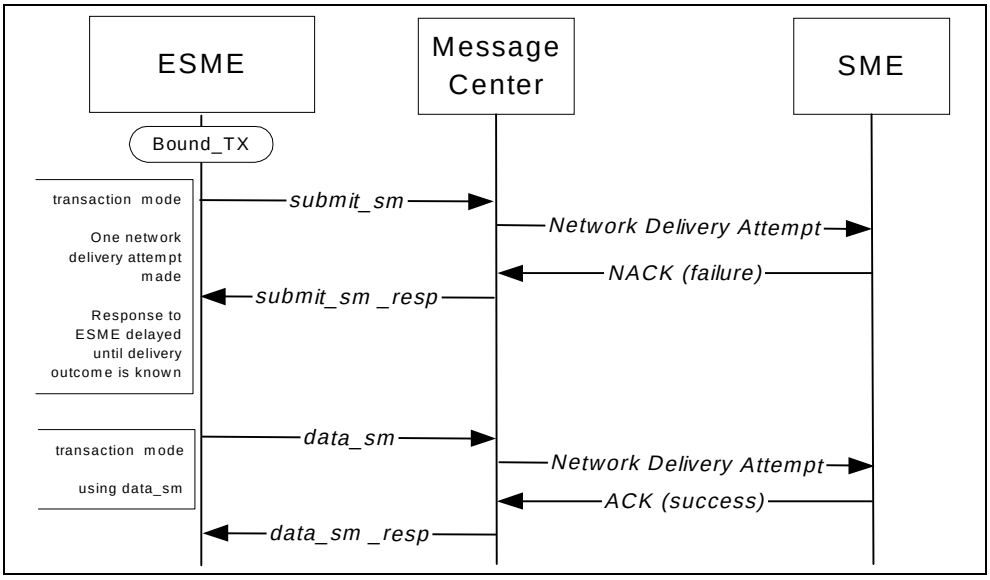


Figure 4-4 Transaction Mode

4.3 Message Delivery Operations

4.3.1 deliver_sm Operation

The deliver_sm is issued by the SMSC to send a message to an ESME. Using this command, the SMSC may route a short message to the ESME for delivery.

4.3.1.1 deliver_sm Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU.	4.5.1
command_id	4	Integer	0x00000005	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to a Unique sequence number. The associated deliver_sm_resp PDU will echo this sequence number.	4.5.4
service_type	Var. max 6	C- Octet String	The service_type parameter can be used to indicate the SMS Application service associated with the message. Specifying the service_type allows the ESME to avail of enhanced messaging services such as “replace by service” type to control the teleservice used on the air interface. Set to NULL if not known by SMSC	4.5.15
source_addr_ton	1	Integer	Type of Number for source address.	4.5.9
source_addr_npi	1	Integer	Numbering Plan Indicator for source address.	4.5.10
source_addr	Var. max 21	C- Octet String	Address of SME which originated this message.	4.5.12
dest_addr_ton	1	Integer	Type of Number for destination	4.5.9
dest_addr_npi	1	Integer	Numbering Plan Indicator for destination	4.5.10
destination_addr	Var. max 21	C-Octet String	Destination address of this short message For mobile terminated messages, this is the directory number of the recipient MS	4.5.13
esm_class	1	Integer	Indicates Message Mode and Message Type	4.5.16
protocol_id	1	Integer	Protocol Identifier. Network specific field.	4.5.17
priority_flag	1	Integer	Designates the priority level of the message	4.5.18

Field Name	Size octets	Type	Description	Ref.
schedule_delivery_time	1 or 17	C-Octet String	The short message is to be scheduled by the SMSC for delivery. Set to NULL if message not known or message is not scheduled	4.5.19.1
validity_period	1 or 17	C- Octet String	The validity period of this message. Set to NULL if not available	4.5.19.2
registered_delivery	1	Integer	Indicator to signify if an SMSC delivery receipt or an SME acknowledgement is required.	4.5.20
replace_if_present_flag	1	Integer	Flag indicating if delivered message should replace an existing message.	4.5.21
data_coding	1	Integer	Defines the encoding scheme of the short message user data.	4.5.22
sm_default_msg_id	1	Integer	Indicates the short message to send from a list of pre-defined (‘canned’) short messages stored on the SMSC. Set to NULL. If not known	4.5.23
sm_length	1	Integer	Length in octets of the short_message user data.	4.5.24
short_message	Var. 0-255	Octet String	Up to 255 octets of short message user data. The exact physical limit for short_message size may vary according to the underlying network Applications which need to send messages longer than 255 octets should use the message_payload parameter. In this case the sm_length field should be set to zero	4.5.25
Message Delivery TLVs	Var.	TLV		4.3.3

Table 4-21 *deliver_sm* PDU

4.3.1.2 deliver_sm_resp Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU.	4.5.1
command_id	4	Integer	0x8000005	4.5.2
command_status	4	Integer	Indicates outcome of deliver_sm request.	4.5.3
sequence_number	4	Integer	Set to sequence number of original deliver_sm PDU.	4.5.4
message_id	Var. Max 65	C- Octet String	This field is unused and should be set to NULL	4.5.26
Message Delivery Response TLVs	Var.	TLV		4.3.3

Table 4-22 *deliver_sm_resp* PDU

4.3.2 data_sm Operation

The *data_sm* operation is symmetrically used for delivery as it is used to submit messages. For detail on the specification of data_sm, refer to section 4.2.2[CL15].

4.3.3 Message Delivery TLVs

The following table lists TLVs appropriate for message delivery using *deliver_sm* or *data_sm* operations.

TLV Name	Description	Ref.
user_message_reference	ESME assigned message reference number.	4.6.4.17
source_port	Indicates the application port number associated with the source address of the message. This parameter should be present for WAP applications.	4.6.4.20
source_addr_subunit	The subcomponent in the destination device, which created the user data.	4.6.4.2
destination_port	Indicates the application port number associated with the destination address of the message. This parameter should be present for WAP applications.	4.6.4.21
dest_addr_subunit	The subcomponent in the destination device for which the user data is intended.	4.6.4
sar_msg_ref_num	The reference number for a particular concatenated short message.	4.6.4.22
sar_total_segments	Indicates the total number of short messages within the concatenated short message.	4.6.4.23
sar_segment_seqnum	Indicates the sequence number of a particular short message fragment within the concatenated short message.	4.6.4.24
payload_type	defines the type of payload (e.g. WDP, WCMP, etc.).	4.6.4.10

TLV Name	Description	Ref.
message_payload	Contains the extended short message user data. Up to 64K octets can be transmitted. Note: The short message data should be inserted in either the *short_message* or *message_payload* fields. Both fields should not be used simultaneously. The *sm_length* field should be set to zero if using the *message_payload* parameter. Note: In the case of *data_sm*, the *message_payload* TLV is the only means of specifying text.	4.6.4.32
set_dpf	Indicator for setting Delivery Pending Flag on delivery failure.	4.6.4.29
privacy_indicator	Indicates the level of privacy associated with the message.	4.6.4.14
callback_num	A callback number associated with the short message. This parameter can be included a number of times for multiple callback addresses.	4.6.4.36
callback_num_pres_ind	Defines the callback number presentation and screening. If this parameter is present and there are multiple instances of the *callback_num* parameter then this parameter must occur an equal number of instances and the order of occurrence determines the particular *callback_num_pres_ind* which corresponds to a particular *callback_num*.	4.6.4.37
callback_num_atag	Associates a displayable alphanumeric tag with the callback number. If this parameter is present and there are multiple instances of the *callback_num* parameter then this parameter must occur an equal number of instances and the order of occurrence determines the particular *callback_num_atag* which corresponds to a particular *callback_num*.	4.6.4.38
source_subaddress	The sub-address of the message originator.	4.6.4.15
dest_subaddress	The sub-address of the message destination.	4.6.4.16
user_response_code	A user response code. The actual response codes are implementation specific.	4.6.4.18
display_time	Provides the receiving MS with a display time associated with the message.	4.6.4.26
sms_signal	Indicates the alerting mechanism when the message is received by an MS.	4.6.4.40
ms_validity	Indicates validity information for this message to the recipient MS.	4.6.4.27
ms_msg_wait_facilities	This parameter controls the indication and specifies the message type (of the message associated with the MWI) at the mobile station.	4.6.4.13
number_of_messages	Indicates the number of messages stored in a mail box	4.6.4.39
alert_on_msg_delivery	Request an MS alert signal be invoked on message delivery.	4.6.4.41
language_indicator	Indicates the language of an alphanumeric text message.	4.6.4.19

TLV Name	Description	Ref.
its_reply_type	The MS user's reply method to an SMS delivery message received from the network is indicated and controlled by this parameter.	4.6.4.42
its_session_info	Session control information for InteractiveTeleservice.	4.6.4.43
ussd_service_op	This parameter is used to identify the required USSD Service type when interfacing to a USSD system.	4.6.4.44 [CL16]
message_state	Should be present for SMSC Delivery Receipts and Intermediate Notifications.	4.6.4.35
network_error_code	May be present for delivery receipts and Intermediate Notifications	4.6.4.31
receipted_message_id	SMSC message ID of message being receipted. Should be present for MC Delivery Receipts and Intermediate Notifications.	4.6.4.12

Table 4-23 Message Delivery TLVs

4.3.4 Delivery Message Types

There are a number of different types of message that can be delivered to an ESME Receiver or Transceiver. These are as follows:

- Normal Message
  This can be any normal message originally submitted by a mobile phone or another ESME.
- MC Delivery Receipt
- Intermediate Notification
- SME User/Manual Acknowledgement
- SME Delivery Acknowledgement
- Conversation Abort
- MC-MC Handover Message

4.3.4.1 MC Delivery Receipt

This message type is used to carry a MC delivery receipt. The MC, on detecting the final state of a registered message (ref. 4.5.20), would normally generate a new receipt message addressed to the originator of the first message. The MC Delivery Receipt is then delivered to the ESME in a *deliver_sm* or *data_sm* operation.

The following fields are relevant in the *deliver_sm* and *data_sm* operations when used for transmitting delivery receipts.

- source address (i.e. *source_addr_ton, source_addr_npi, source_addr*)
  The source address will be taken from the destination address of the original short message which generated the delivery receipt. The receipt appears as if it emanated from the recipient of the original registered message.
- destination address (i.e. *dest_addr_ton, dest_addr_npi, destination_addr*)
  The destination address will be taken from the source address of the original short message which generated the delivery receipt. The receipt is addresses to the SME that originally sent the first registered message.
- *esm_class*
  Bit 2 of the *esm_class* is set to 1 to indicate that the message is a delivery receipt.

- *message_state* TLV
  This TLV indicates the final state of the original message. Remember a receipt will be sent
- *network_error_code* TLV
- *receipted_message_id* TLV

Note: Many pre-V3.4 interfaces and Message Centers supporting V3.3 are likely to have a means of passing receipt information within the *short_message* field. The format specifics of this information is product specific and beyond the scope of this specification.

Note: The returning of a message receipt is not always guaranteed. The criteria used to control the returning of a receipt depend on the value set in the *registered_delivery* field (ref. 4.5.20) of the original message.

[CL17]

4.3.4.2 Intermediate Notification

An intermediate notification is a special form of message that the MC may send to an ESME for a mobile terminated message delivery. It provides an intermediate status of a message delivery attempt.

Typical uses are to report the outcome of delivery attempts made during the message's retry lifetime within the MC. This could be used to track the various reasons why a message is not delivered to its destination and use this profile the subscriber's availability.

In such cases the following TLVs would be of importance:

- *message_state* TLV
- *network_error_code* TLV
- *receipted_message_id* TLV

However, support for Intermediate Notification functionality is specific to the MC implementation and the MC Service Provider and is beyond the scope of this specification.

4.3.4.3 SME Delivery Acknowledgement

Despite its name, an SME Delivery Acknowledgement is not an indication that the short message has arrived at the SME, but rather an indication from the recipient SME that the user has read the short message.

For an MS-based SME, an SME Delivery Acknowledgement is sent when the MS user or MS application has read the message from the SMS storage unit (e.g. SIM card).

For a fixed SME (i.e. ESME) the circumstances in which an SME Delivery Acknowledgement may be sent are beyond the scope of this specification.

4.3.4.4 SME Manual/User Acknowledgement

A Manual/User Acknowledgement is an application generated reply message sent in response to an application request message. For example, this message type could contain a selected menu item number from a menu list sent in an application request message.

4.3.4.5 Conversation Abort

This message type is unique to Interactive Teleservice defined by the Korean CDMA Carriers Organization. It is sent by a MS-based SME to indicate the unexpected termination of an interactive session. A Conversation Abort may be carried in a *deliver_sm* or *data_sm* PDU.

4.4 Ancillary Operations

4.4.1 cancel_sm Operation

This command is issued by the ESME to cancel one or more previously submitted short messages that are still pending delivery. The command may specify a particular message to cancel, or all messages for a particular source, destination and *service_type* are to be cancelled.

If the *message_id* is set to the ID of a previously submitted message, then provided the source address supplied by the ESME matches that of the stored message, that message will be cancelled.

If the *message_id* is NULL, all outstanding undelivered messages with matching source and destination addresses given in the PDU are cancelled. If provided, *service_type* is included in this matching.

Where the original *submit_sm, data_sm* or *submit_multi* 'source address' is defaulted to NULL, then the source address in the *cancel_sm* command should also be NULL.

4.4.1.1 cancel_sm Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU.	4.5.1
command_id	4	Integer	0x00000008	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to a unique sequence number. The associated cancel_sm_resp PDU should echo the same sequence number.	4.5.4
service_type	Var. max 6	C- Octet String	Set to indicate SMS Application service, if cancellation of a group of application service messages is desired. Otherwise set to NULL.	4.5.15
message_id	Var. max 65	C- Octet String	Message ID of the message to be cancelled. This must be the SMSC assigned Message ID of the original message. Set to NULL if cancelling a group of messages.	4.5.26
source_addr_ton	1	Integer	Type of Number of message originator. This is used for verification purposes, and must match that supplied in the original message submission request PDU. If not known, set to NULL.	4.5.9
source_addr_npi	1	Integer	Numbering Plan Identity of message originator. This is used for verification purposes, and must match that supplied in the original message submission request PDU. If not known, set to NULL.	4.5.10
source_addr	Var. max 21	C- Octet String	Source address of message(s) to be cancelled. This is used for verification purposes, and must match that supplied in the original message submission request PDU(s).	4.5.12
dest_addr_ton	1	Integer	Type of number of destination SME address of the message(s) to be cancelled. This is used for verification purposes, and must match that supplied in the original message submission request PDU (e.g. submit_sm). May be set to NULL when the message_id is provided.	4.5.9

dest_addr_npi	1	Integer	Numbering Plan Indicator of destination SME address of the message(s)) to be cancelled. This is used for verification purposes, and must match that supplied in the original message submission request PDU. May be set to NULL when the message_id is provided.	4.5.10
destination_addr	Var. max 21	C- Octet String	Destination address of message(s) to be cancelled. This is used for verification purposes, and must match that supplied in the original message submission request PDU. May be set to NULL when the message_id is provided.	4.5.13

Table 4-24 *cancel_sm* PDU

4.4.1.2 cancel_sm_resp Syntax

The *cancel_sm_resp* PDU is used to reply to a *cancel_sm* request. It comprises the SMPP message header only.

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU.	4.5.1
command_id	4	Integer	0x80000008	4.5.2
command_status	4	Integer	Indicates outcome of cancel_sm request.	4.5.3
sequence_number	4	Integer	Set to sequence number of cancel_sm PDU.	4.5.4

Table 4-25 *cancel_sm_resp* PDU

4.4.1.3 query_sm Operation

This command is issued by the ESME to query the status of a previously submitted short message.

The matching mechanism is based on the SMSC assigned *message_id* and source address. Where the original *submit_sm, data_sm* or *submit_multi* 'source address' was defaulted to NULL, then the source address in the *query_sm* command should also be set to NULL.

4.4.1.4 query_sm Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU	4.5.1
command_id	4	Integer	0x00000003	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to a unique sequence number. The associated query_sm_resp PDU should echo the same sequence number	4.5.4
message_id	Var. Max 65	C- Octet String	Message ID of the message whose state is to be queried. This must be the SMSC assigned Message ID allocated to the original short message when submitted to the SMSC by the submit_sm, data_sm or submit_multi command, and returned in the response PDU by the SMSC.	4.5.26
source_addr_ton	1	Integer	Type of Number of message originator. This is used for verification purposes, and must match that supplied in the original request PDU (e.g. submit_sm). If not known, set to NULL.	4.5.9
source_addr_npi	1	Integer	Numbering Plan Identity of message originator. This is used for verification purposes, and must match that supplied in the original request PDU (e.g. submit_sm). If not known, set to NULL.	4.5.10
source_addr	Var. Max 21	C- Octet String	Address of message originator. This is used for verification purposes, and must match that supplied in the original request PDU (e.g. submit_sm). If not known, set to NULL.	4.5.12

Table 4-26 *query_sm* PDU

4.4.1.5 query_sm_resp Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU.	4.5.1
command_id	4	Integer	0x8000003	4.5.2
command_status	4	Integer	Indicates outcome of query_sm request	4.5.3
sequence_number	4	Integer	Set to sequence number of original query_sm PDU.	4.5.4
message_id	Var. max 65	C- Octet String	SMSC Message ID of the message whose state is being queried.	4.5.26
final_date	1 or 17	C- Octet String	Date and time when the queried message reached a final state. For messages which have not yet reached a final state this field will contain a single NULL octet.	
message_state	1	Integer	Specifies the status of the queried short message.	4.5.31
error_code	1	Integer	Where appropriate this holds a network error code defining the reason for failure of message delivery.	

Table 4-27 *query_sm_resp* PDU

4.4.2 replace_sm Operation

This command is issued by the ESME to replace a previously submitted short message that is still pending delivery. The matching mechanism is based on the message_id and source address of the original message.

Where the original submit_sm 'source address' was defaulted to NULL, then the source address in the replace_sm command should also be NULL.

4.4.2.1 replace_sm Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU.	4.5.1
command_id	4	Integer	0x00000007	4.5.2
command_status	4	Integer	0x00000000	4.5.3
sequence_number	4	Integer	Set to a Unique sequence number. The associated submit_sm_resp PDU will echo this sequence number.	4.5.4
message_id	Var. Max 65	C-Octet String	Message ID of the message to be replaced. This must be the SMSC assigned Message ID allocated to the original short message when submitted to the SMSC by the submit_sm, data_sm or submit_multi command, and returned in the response PDU by the SMSC.	4.5.26
source_addr_ton	1	Integer	Type of Number for source address used in original message. If not known, set to NULL (Unknown).	4.5.9
source_addr_npi	1	Integer	Numbering Plan Indicator for source address of original message. If not known, set to NULL (Unknown).	4.5.10
source_addr	Var. max 21	C-Octet String	Address of SME which originated this message. If not known, set to NULL (Unknown).	4.5.12
schedule_delivery_time	1 or 17	C-Octet String	The short message is to be scheduled by the SMSC for delivery. Set to NULL for immediate message delivery	4.5.19.1

Field Name	Size octets	Type	Description	Ref.
validity_period	1 or 17	C- Octet String	The validity period of this message. Set to NULL to request the SMSC default validity period	4.5.19.2
registered_delivery	1	Integer	Indicator to signify if an SMSC delivery receipt or an SME acknowledgement is required.	4.5.20
sm_default_msg_id	1	Integer	Indicates the short message to send from a list of pre- defined ('canned') short messages stored on the SMSC. If not using an SMSC canned message, set to NULL.	4.5.23
sm_length	1	Integer	Length in octets of the short_message user data.	4.5.24
short_message	Var. 0-255	Octet String	Up to 255 octets of short message user data. The exact physical limit for short_message size may vary according to the underlying network Applications which need to send messages longer than 254 octets should use the message_payload parameter. In this case the sm_length field should be set to zero	4.5.25

Table 4-28 *replace_sm* PDU

4.4.2.2 replace_sm_resp Syntax

Field Name	Size octets	Type	Description	Ref.
command_length	4	Integer	Set to overall length of PDU.	4.5.1
command_id	4	Integer	0x80000007	4.5.2
command_status	4	Integer	Indicates outcome of replace_sm request.	4.5.3
sequence_number	4	Integer	Set to sequence number of original replace_sm PDU.	4.5.4

Table 4-29 *replace_sm_resp* PDU

4.5 PDU Field Definitions

4.5.1 command_length

This 4-octet integer represents the overall length of a PDU. This is described in detail in section 3.2.1.1

4.5.2 command_id

The complete set of SMPP Command IDs and their associated values are defined in the following table.

Command ID	Value
generic_nack	0x80000000
bind_receiver	0x00000001
bind_receiver_resp	0x80000001
bind_transmitter	0x00000002
bind_transmitter_resp	0x80000002
query_sm	0x00000003
query_sm_resp	0x80000003
submit_sm	0x00000004
submit_sm_resp	0x80000004
deliver_sm	0x00000005
deliver_sm_resp	0x80000005
unbind	0x00000006
unbind_resp	0x80000006
replace_sm	0x00000007
replace_sm_resp	0x80000007
cancel_sm	0x00000008
cancel_sm_resp	0x80000008
bind_transceiver	0x00000009
bind_transceiver_resp	0x80000009
Reserved	0x0000000A 0x8000000A
outbind	0x0000000B
Reserved	0x0000000C - 0x00000014 0x8000000B - 0x80000014
enquire_link	0x00000015
enquire_link_resp	0x80000015
Reserved	0x00000016 - 0x00000020 0x80000016 - 0x80000020
submit_multi	0x00000021
submit_multi_resp	0x80000021
Reserved	0x00000022 - 0x000000FF

Command ID	Value
	0x80000022 - 0x800000FF
Reserved	0x00000100
Reserved	0x80000100
Reserved	0x00000101 0x80000101
alert_notification	0x00000102
Reserved	0x80000102
data_sm	0x00000103
data_sm_resp	0x80000103
Reserved for SMPP extension	0x00000104 - 0x0000FFFF 0x80000104 - 0x8000FFFF
Reserved	0x00010000 - 0x000101FF 0x80010000 - 0x800101FF
Reserved for SMSC Vendor	0x00010200 - 0x000102FF 0x80010200 - 0x800102FF
Reserved	0x00010300 - 0xFFFFFFFF

Table 4-30 *command_id* Values

4.5.3 command_status

The *command_status* field of an SMPP message response indicates the success or failure of an SMPP request. It is relevant only in the SMPP response message and should be set to NULL in SMPP request messages.

The SMPP Error status codes are returned by the SMSC in the *command_status* field of the SMPP message header and in the *error_status_code* field of a *submit_multi_resp* message.

The complete set of SMPP Error Codes and their associated values are defined in the following table.

Command Status Name	Value	Description
ESME_ROK	0x00000000	No Error
ESME_RINVMSGLEN	0x00000001	Message Length is invalid
ESME_RINVCMDLEN	0x00000002	Command Length is invalid
ESME_RINVCMDID	0x00000003	Invalid Command ID
ESME_RINVBNDSTS	0x00000004	Incorrect BIND Status for given command
ESME_RALYBND	0x00000005	ESME Already in Bound State
ESME_RINVPRTFLG	0x00000006	Invalid Priority Flag
ESME_RINVREGDLVFLG	0x00000007	Invalid Registered Delivery Flag
ESME_RSYSERR	0x00000008	System Error
Reserved	0x00000009	Reserved
ESME_RINVSRCADR	0x0000000A	Invalid Source Address
ESME_RINVDSTADR	0x0000000B	Invalid Dest Addr
ESME_RINVMSGID	0x0000000C	Message ID is invalid
ESME_RBINDFAIL	0x0000000D	Bind Failed
ESME_RINVPASWD	0x0000000E	Invalid Password
ESME_RINVSYSID	0x0000000F	Invalid System ID
Reserved	0x00000010	Reserved
ESME_RCANCELFAIL	0x00000011	Cancel SM Failed
Reserved	0x00000012	Reserved
ESME_RREPLACEFAIL	0x00000013	Replace SM Failed

Command Status Name	Value	Description
ESME_RMSGQFUL	0x00000014	Message Queue Full
ESME_RINVSERTYP	0x00000015	Invalid Service Type
Reserved	0x00000016-0x00000032	Reserved
ESME_RINVNUMDESTS	0x00000033	Invalid number of destinations
ESME_RINVDLNAME	0x00000034	Invalid Distribution List name
Reserved	0x00000035-0x0000003F	Reserved
ESME_RINVDESTFLAG	0x00000040	Destination flag is invalid (submit_multi)
Reserved	0x00000041	Reserved
ESME_RINVSUBREP	0x00000042	Invalid 'submit with replace' request (i.e. submit_sm with replace_if_present_flag set)
ESME_RINVESMCLASS	0x00000043	Invalid esm_class field data
ESME_RCNTSUBDL	0x00000044	Cannot Submit to Distribution List
ESME_RSUBMITFAIL	0x00000045	submit_sm or submit_multi failed
Reserved	0x00000046-0x00000047	Reserved
ESME_RINVSRCTON	0x00000048	Invalid Source address TON
ESME_RINVSRCNPI	0x00000049	Invalid Source address NPI
ESME_RINVDSTTON	0x00000050	Invalid Destination address TON
ESME_RINVDSTNPI	0x00000051	Invalid Destination address NPI
Reserved	0x00000052	Reserved
ESME_RINVSYSTYP	0x00000053	Invalid system_type field
ESME_RINVREPFLAG	0x00000054	Invalid replace_if_present flag
ESME_RINVNUMMSGS	0x00000055	Invalid number of messages
Reserved	0x00000056-0x00000057	Reserved
ESME_RTHROTTLED	0x00000058	Throttling error (ESME has exceeded allowed message limits)
Reserved	0x00000059-0x00000060	Reserved
ESME_RINVSCHED	0x00000061	Invalid Scheduled Delivery Time
ESME_RINVEXPIRY	0x00000062	Invalid message validity period (Expiry time)
ESME_RINVDFTMSGID	0x00000063	Predefined Message Invalid or Not Found
ESME_RX_T_APPN	0x00000064	ESME Receiver Temporary App Error Code
ESME_RX_P_APPN	0x00000065	ESME Receiver Permanent App Error Code
ESME_RX_R_APPN	0x00000066	ESME Receiver Reject Message Error Code
ESME_RQUERYFAIL	0x00000067	query_sm request failed
Reserved	0x00000068 - 0x000000BF	Reserved
ESME_RINVOPTPARSTREAM	0x000000C0	Error in the optional part of the PDU Body.
ESME_ROPTPARNOTALLWD	0x000000C1	Optional Parameter not allowed
ESME_RINVPARLEN	0x000000C2	Invalid Parameter Length.
ESME_RMISSINGOPTPARAM	0x000000C3	Expected Optional Parameter missing
ESME_RINVOPTPARAMVAL	0x000000C4	Invalid Optional Parameter Value
Reserved	0x000000C5 - 0x000000FD	Reserved
ESME_RDELIVERYFAILURE	0x000000FE	Delivery Failure (used for data_sm_resp)
ESME_RUNKNOWNERR	0x000000FF	Unknown Error
ESME_RSERTYPUNAUTH	0x00000100	ESME Not authorised to use specified *service_type*[CL18]
ESME_RPROHIBITED	0x00000101	ESME Prohibited from using specified operation.[CL19]

Command Status Name	Value	Description
ESME_RSERTYPUNAVAIL	0x00000102	Specified *service_type* is unavailable[CL20]
ESME_RSERTYPDENIED	0x00000103	Specified *service_type* is denied due to inappropriate message content wrt. Selected *service_type*[CL21]
Reserved for SMPP extension	0x00000103-0x000003FF	Reserved for SMPP extension
Reserved for SMSC vendor specific errors	0x00000400-0x000004FF	Reserved for SMSC vendor specific errors
Reserved	0x00000500-0xFFFFFFFF	Reserved

Table 4-31 *command_status* Values

4.5.4 sequence_number

A sequence number allows a response PDU to be correlated with a request PDU. The associated SMPP response PDU must preserve this field. The allowed *sequence_number* range is from 0x00000001 to 0x7FFFFFFF. In the event of a session using the full range of values for the *sequence_number*, the ESME or MC should wrap around to 0x00000001. The value 0x00000000 is recommended for use when issuing a *generic_nack* where the original PDU was deemed completely invalid and its PDU header, was not used to derive a *sequence_number* for the response PDU. Detailed information on how PDU sequencing works is available in section 2.5.

4.5.5 system_id

The *system_id* parameter is used to identify an ESME or an SMSC at bind time. An ESME *system_id* identifies the ESME or ESME agent to the SMSC. The SMSC *system_id* provides an identification of the SMSC to the ESME.

4.5.6 password

The *password* parameter is used by the SMSC to authenticate the identity of the binding ESME. The Service Provider may require ESME's to provide a password when binding to the SMSC. This password is normally issued by the SMSC system administrator.

The *password* parameter may also be used by the ESME to authenticate the identity of the binding SMSC (e.g. in the case of the *outbind* operation).

4.5.7 system_type

The *system_type* parameter is used to categorize the type of ESME that is binding to the SMSC. Examples include "VMS" (voice mail system) and "OTA" (over-the-air activation system).

Specification of the *system_type* is optional - some SMSCs may not require ESME's to provide this detail. In this case, the ESME can set the *system_type* to NULL.

4.5.8 interface_version

This parameter is used to indicate the version of the SMPP protocol. The following interface version values are defined:

Interface Version	Value
Indicates that the EMSE supports version 3.3 or earlier of the SMPP protocol.	0x00-0x33
Indicates that the ESME is supporting SMPP version 3.4	0x34
Indicates that the ESME is supporting SMPP version 5.0	0x50
All other values reserved	

Table 4-32 *interface_version* Values

4.5.9 addr_ton, source_addr_ton, dest_addr_ton, esme_addr_ton

These fields define the Type of Number (TON) to be used in the SME address parameters. The following TON values are defined:

TON	Value
Unknown	00000000
International	00000001
National	00000010
Network Specific	00000011
Subscriber Number	00000100
Alphanumeric	00000101
Abbreviated	00000110
All other values reserved	

Table 4-33 TON Values

4.5.10 addr_npi, source_addr_npi, dest_addr_npi, esme_addr_npi

These fields define the Numeric Plan Indicator (NPI) to be used in the SME address parameters. The following NPI values are defined:

NPI	Value
Unknown	00000000
ISDN (E163/E164)	00000001
Data (X.121)	00000011
Telex (F.69)	00000100
Land Mobile (E.212)	00000110
National	00001000
Private	00001001
ERMES	00001010
Internet (IP)	00001110
WAP Client Id (to be defined by WAP Forum)	00010010
All other values reserved	

Table 4-34 NPI Values

4.5.11 address_range

The *address_range* parameter is used in the *bind_receiver* and *bind_transceiver* command to specify a set of SME addresses serviced by the ESME client. A single SME address may also be specified in the *address_range* parameter. UNIX Regular Expression notation should be used to specify a range of addresses (Refer to Appendix A.)

Messages addressed to any destination in this range shall be routed to the ESME.

Notes

For IP addresses, it is only possible to specify a single IP address. A range of IP addresses is not allowed. IP version 6.0 is not currently supported in this version of the protocol.

4.5.12 source_addr

Specifies the address of SME which originated this message. An ESME which is implemented as a single SME address, may set this field to NULL to allow the SMSC to default the source address of the submitted message.

Notes

An IP address is specified in “aaa.bbb.ccc.ddd” notation. IP version 6.0 is not supported in V3.4 of the SMPP protocol.

4.5.13 destination_addr

Specifies the destination SME address. For mobile terminated messages, this is the directory number of the recipient MS.

Notes

An IP address is specified in “aaa.bbb.ccc.ddd” notation. IP version 6.0 is not supported in V3.4 of the SMPP protocol.

4.5.14 esme_addr

Specifies the address of an ESME address to which an *alert_notification* should be routed.

Notes

An IP address is specified in “aaa.bbb.ccc.ddd” notation. IP version 6.0 is not supported in V3.4 of the SMPP protocol.

4.5.15 service_type

The *service_type* parameter can be used to indicate the SMS Application service associated with the message. Specifying the *service_type* allows the ESME to:

- Avail of enhanced messaging services such as *replace_if_present* by service type (generic to all network types).
- Control the teleservice used on the air interface (e.g. ANSI-136/TDMA, IS-95/CDMA).

SMSCs may implicitly associate a ‘replace if present’ function from the indicated *service_type* in a message submission operation, i.e., the SMSC will always replace an existing message pending delivery, that has the same originating and destination address as the submitted message. For example, an SMSC can ensure that a Voice Mail System using a *service_type* of “VMA” has at most one outstanding notification per destination MS by automatically invoking the “replace if present” function.

The following generic *service_types* are defined:

Service Type	Description
“” (NULL)	Default
“CMT”	Cellular Messaging
“CPT”	Cellular Paging
“VMN”	Voice Mail Notification
“VMA”	Voice Mail Alerting
“WAP”	Wireless Application Protocol
“USSD”	Unstructured Supplementary Services Data

Table 4-35 *service_type* Values

All other values are carrier specific and are defined by mutual agreement between the SMSC Service Provider and the ESME application.

4.5.16 esm_class

The *esm_class* parameter is used to indicate special message attributes associated with the short message.

The *esm_class* parameter is encoded as follows in the *submit_sm, submit_multi* and *data_sm* (ESME -> SMSC) PDUs:

	esm_class Bits	Meaning
	7 6 5 4 3 2 1 0	
Messaging Mode (bits 1-0)	x x x x x x 0 0	Default SMSC Mode (e.g. Store and Forward)
	x x x x x x 0 1	Datagram mode
	x x x x x x 1 0	Forward (i.e. Transaction) mode
	x x x x x x 1 1	Store and Forward mode (use to select Store and Forward mode if Default SMSC Mode is non Store and Forward)
Message Type (bits 2 and 5)	x x 0 0 0 0 x x	Default message Type (i.e. normal message)
	x x 0 0 0 1 x x	Short Message contains SMSC Delivery Receipt
	x x 1 0 0 0 x x	Short Message contains Intermediate Delivery Notification
ANSI-41 Specific (bits 5-2)	x x 0 0 1 0 x x	Short Message contains ESME Delivery Acknowledgement
	x x 0 1 0 0 x x	Short Message contains ESME Manual User Acknowledgement
	x x 0 1 1 0 x x	Short Message contains Conversation Abort (Korean CDMA)
GSM Specific (bits 7-6)	0 0 x x x x x x	No specific features selected
	0 1 x x x x x x	UDHI Indicator (only relevant for MT short messages)
	1 0 x x x x x x	Set Reply Path (only relevant for GSM network)
	1 1 x x x x x x	Set UDHI and Reply Path (only relevant for GSM network)

Table 4-36 *esm_class* Bit Values

The default setting of the *esm_class* parameter is 0x00.

Notes: If an ESME encodes GSM User Data Header information in the short message user data, it must set the UDHI flag in the *esm_class* field. If the SMSC delivers a short message that contains GSM User Data Header information encoded in the *short_message* or *message_payload* parameter, it must set the UDHI flag in the *esm_class* field. For GSM networks, the concatenation related optional parameters (*sar_msg_ref_num*, *sar_total_segments, sar_segment_seqnum*) or port addressing related optional parameters (*source_port, destination_port*) cannot be used in conjunction with encoded User Data Header in the *short_message* (user data) field. This means that the above listed optional parameters cannot be used if the User Data Header Indicator flag is set.

4.5.17 protocol_id

GSM

Set according to GSM 03.40 [GSM 03.40]

ANSI-136 (TDMA)

For mobile terminated messages, this field is not used and is therefore ignored by the SMSC. For ANSI-136 mobile originated messages, the SMSC should set this value to NULL.

IS-95 (CDMA)

For mobile terminated messages, this field is not used and is therefore ignored by the SMSC. For IS-95 mobile originated messages, the SMSC should set this value to NULL.

4.5.18 priority_flag

The *priority_flag* parameter allows the originating SME to assign a priority level to the short message:

Priority Level	GSM	ANSI-136	IS-95
0	non-priority	Bulk	Normal
1	priority	Normal	Interactive
2	priority	Urgent	Urgent
3	priority	Very Urgent	Emergency
All other values reserved			

Table 4-37 *priority_flag* Values

4.5.19 scheduled_delivery_time and validity_period

4.5.19.1 scheduled_delivery_time

This parameter specifies the scheduled time at which the message delivery should be first attempted.

It defines either the absolute date and time or relative time from the current SMSC time at which delivery of this message will be attempted by the SMSC.

It can be specified in either absolute time format or relative time format.

4.5.19.2 validity_period

The *validity_period* parameter indicates the SMSC expiration time, after which the message should be discarded if not delivered to the destination. It can be defined in absolute time format or relative time format.

4.5.19.3 Absolute Time Format

This is the default format used by SMPP. Scheduled delivery times and expiry times are specified in their global UTC format, including the quarter hour offset and direction symbol '+' or '-'.

Absolute time is formatted as a 16 character string (encoded as a 17-octet C-octet String) "**YYMMDDhhmmsstnnp**" where:

Digits	Meaning
'YY'	last two digits of the year (00-99)
'MM'	month (01-12)
'DD'	day (01-31)
'hh'	hour (00-23)
'mm'	minute (00-59)
'ss'	second (00-59)
't'	tenths of second (0-9)
'nn'	Time difference in quarter hours between local time (as expressed in the first 13 octets) and UTC (Universal Time Constant) time (00-48).
'p'	"+" Local time is in quarter hours advanced in relation to UTC time. "-" Local time is in quarter hours retarded in relation to UTC time.

Table 4-38 Absolute UTC Time Format

4.5.19.4 Relative Time Format

Relative Time can be indicated by setting the UTC orientation flag to 'R' instead of '+' or '-'. In this form, the SMSC interprets the time format as the number of years, months, days, hours, minutes and seconds from the current SMSC time. Values for tenths of seconds 't' and UTC offset 'nn' are ignored and should be set to '0' and '00' respectively.

Absolute time is formatted as a 16 character string (encoded as a 17-octet C-octet String) "**YYMMDDhhmmsstnnp**" where:

Digits	Meaning
'YY'	year (00-99)
'MM'	month (01-12)
'DD'	day (01-31)
'hh'	hour (00-23)
'mm'	minute (00-59)
'ss'	second (00-59)
't'	Unused. Should be set to '0'
'nn'	Unused. Should be set to '00'
'p'	"R" Local time is relative to the current SMSC time.

Table 4-39 Relative Time Format

For example, the following time format '020610233429000R":

would be interpreted as a relative period of 2 years, 6 months, 10 days, 23 hours, 34 minutes and 29 seconds from the current SMSC time. An SMSC operator may choose to impose a limit on relative time offsets, thus either rejecting a message that exceeds such a limit or reducing the offset to the maximum relative time allowed.

For example: a typical validity period might be 7 days and a typical scheduled delivery times might be 14 days from the submission time.

4.5.20 registered_delivery

The *registered_delivery* parameter is used to request an SMSC delivery receipt and/or SME originated acknowledgements. The following values are defined:

	registered_delivery Bits	Meaning
	7 6 5 4 3 2 1 0	
SMSC Delivery Receipt (bits 1 and 0)	x x x x x x 0 0	No SMSC Delivery Receipt requested (default)
	x x x x x x 0 1	SMSC Delivery Receipt requested where final delivery outcome is delivery success or failure
	x x x x x x 1 0	SMSC Delivery Receipt requested where the final delivery outcome is delivery failure
	x x x x x x 1 1	SMSC Delivery Receipt requested where the final delivery outcome is success[CL22]
SME originated Acknowledgement (bits 3 and 2)	x x x x 0 0 x x	No recipient SME acknowledgment requested (default)
	x x x x 0 1 x x	SME Delivery Acknowledgement requested
	x x x x 1 0 x x	SME Manual/User Acknowledgment requested
	x x x x 1 1 x x	Both Delivery and Manual/User Acknowledgment requested
Intermediate Notification (bit 5)	x x x 0 x x x x	No Intermediate notification requested (default)
	x x x 1 x x x x	Intermediate notification requested Support for Intermediate Notification Functionality is specific to the MC implementation and is beyond the scope of the SMPP Protocol Specification.
All Other Bits	Reserved	

Table 4-40 *registered_delivery* Values

4.5.21 replace_if_present_flag

The *replace_if_present_flag* parameter is used to request the SMSC to replace a previously submitted message that is still pending delivery. The SMSC will replace an existing message provided that the source address, destination address and *service_type* match the same fields in the new message.

Value	Meaning
0	Don't replace (default)
1	Replace
2 - 255	reserved

Table 4-41 *replace_if_present* Values

ESME applications that use this SMSC messaging function should use the same *service_type* and set the *replace_if_present_flag* parameter consistently to "1" for all messages, including the first message. This ensures that the SMSC has at most one message pending per destination SME for a particular application (e.g. voice mail notification).

4.5.22 data_coding

The following values are defined for this field:

data_coding Bits 7 6 5 4 3 2 1 0	Meaning
0 0 0 0 0 0 0 0	SMSC Default Alphabet
0 0 0 0 0 0 0 1	IA5 (CCITT T.50)/ASCII (ANSI X3.4)
0 0 0 0 0 0 1 0	Octet unspecified (8-bit binary)
0 0 0 0 0 0 1 1	Latin 1 (ISO-8859-1)
0 0 0 0 0 1 0 0	Octet unspecified (8-bit binary)
0 0 0 0 0 1 0 1	JIS (X 0208-1990)
0 0 0 0 0 1 1 0	Cyrllic (ISO-8859-5)
0 0 0 0 0 1 1 1	Latin/Hebrew (ISO-8859-8)
0 0 0 0 1 0 0 0	UCS2 (ISO/IEC-10646)
0 0 0 0 1 0 0 1	Pictogram Encoding
0 0 0 0 1 0 1 0	ISO-2022-JP (Music Codes)
0 0 0 0 1 0 1 1	Reserved
0 0 0 0 1 1 0 0	Reserved
0 0 0 0 1 1 0 1	Extended Kanji JIS (X 0212-1990)
0 0 0 0 1 1 1 0	KS C 5601
0 0 0 0 1 1 1 1 . . . . . . 1 0 1 1 1 1 1 1	reserved
1 1 0 0 x x x x	GSM MWI control - see [GSM 03.38]
1 1 0 1 x x x x	GSM MWI control - see [GSM 03.38]
1 1 1 0 x x x x	Reserved
1 1 1 1 x x x x	GSM message class control - see [GSM 03.38]

Table 4-42 *data_coding* Values

Notes:

a. These coding schemes are common to GSM, TDMA and CDMA. The SMPP protocol allows ESME applications to use the same DCS value (i.e. the GSM 03.38 value) for all three technologies.
b. In cases where a Data Coding Scheme is defined for TDMA and/ or CDMA but not defined for GSM, SMPP uses GSM 03.38 reserved values.
c. There is no default setting for the *data_coding* parameter.
d. The *data_coding* parameter will evolve to specify Character code settings only. Thus the recommended way to specify GSM MWI control is by specifying the relevant settings in the optional parameters *ms_msg_wait_facilities* and *ms_validity.*
e. The *data_coding* parameter will evolve to specify Character code settings only. Thus the recommended way to specify GSM message class control is by specifying the relevant setting in the optional parameter *dest_addr_subunit.*

4.5.23 sm_default_msg_id

The *sm_default_msg_id* parameter specifies the SMSC index of a pre-defined ('canned') message.

sm_default_msg_id Value	Meaning
0	reserved
1-254	Allowed values
255	Reserved

Table 4-43 *sm_default_msg_id* Values

4.5.24 sm_length

The *sm_length* parameter specifies the length of the *short_message* parameter in octets. The *sm_length* should be set to 0 in the *submit_sm, submit_multi,* and *deliver_sm* PDUs if the *message_payload* parameter is being used to send user data larger than 254 octets.

sm_length Value	Meaning
0	no user data in short message field
1-254	allowed
255	not allowed

Table 4-44 *sm_length* Values

4.5.25 short_message

The *short_message* parameter contains the user data. A maximum of 254 octets can be sent. ESME's should use the optional *message_payload* parameter in *submit_sm, submit_multi,* and *deliver_sm* to send larger user data sizes.

4.5.26 message_id

The unique message identifier reference assigned by the SMSC to each submitted short message. It is an opaque value and is set according to SMSC implementation. It is returned by the SMSC in the *submit_sm_resp, submit_multi_resp, deliver_sm_resp* and *data_sm_resp* PDUs and may be used by the ESME in subsequent SMPP operations relating to the short message, e.g. the ESME can use the *query_sm* operation to query a previously submitted message using the SMSC *message_id* as the message handle***.***

4.5.27 number_of_dests

The *number_of_dests* parameter indicates the number of *dest_address* structures that are to follow in the *submit_multi* operation.

A maximum of 254 destination address structures are allowed.

4.5.28 dest_flag

Flag which will identify whether destination address is a Distribution List (DL) name or SME address.

dest_flag Value	Meaning
1	SME Address
2	Distribution List Name

Table 4-45 *dest_flag* Values

4.5.29 no_unsuccess

The number of unsuccessful SME destinations to which delivery was attempted for a *submit_multi* operation.

4.5.30 dl_name

The reference name for a distribution list provisioned on the SMSC. Distribution list names are defined by mutual agreement between the SMSC and the ESME.

4.5.31 message_state

The following is a list of allowable states for a short message. The *message_state* value is returned by the SMSC to the ESME as part of the *query_sm_resp* PDU.

Message State	Value	Description
ENROUTE	1	The message is in enroute state.
DELIVERED	2	Message is delivered to destination
EXPIRED	3	Message validity period has expired.
DELETED	4	Message has been deleted.
UNDELIVERABLE	5	Message is undeliverable
ACCEPTED	6	Message is in accepted state (i.e. has been manually read on behalf of the subscriber by customer service)
UNKNOWN	7	Message is in invalid state
REJECTED	8	Message is in a rejected state

Table 4-46 *message_state* Values

4.6 PDU TLV Definitions

TLV fields may be optionally included in an SMPP message. TLVs must always appear at the end of an SMPP PDU. However, they may be included in any convenient order and need not be encoded in the order presented in this document.

For a particular SMPP PDU, the ESME or SMSC may include some, all or none of the defined optional parameters as required for the particular application context. For example a paging system may in an SMPP *submit_sm* operation, include only the “callback number” related optional parameters.

4.6.1 TLV Tag

The SMPP protocol defines the following Parameter Tag blocks:

TLV Tag Range	Meaning
0x0000	Reserved
0x0001 - 0x00FF	SMPP defined optional parameters
0x0100 - 0x01FF	Reserved
0x0200 - 0x05FF	SMPP defined optional parameters
0x0600 - 0x10FF	Reserved for SMPP Protocol Extension
0x1100 - 0x11FF	Reserved
0x1200 - 0x13FF	SMPP defined optional parameters
0x1400 - 0x3FFF	Reserved for SMSC Vendor specific optional parameters
0x4000 - 0xFFFF	Reserved

Table 4-47 TLV Tag Value Ranges

The SMPP supported Optional Parameters and their associated Tag Values are listed as follows:

Tag	Value	Wireless Network Technology
dest_addr_subunit	0x0005	GSM
dest_network_type	0x0006	Generic
dest_bearer_type	0x0007	Generic
dest_telematics_id	0x0008	GSM
source_addr_subunit	0x000D	GSM
source_network_type	0x000E	Generic
source_bearer_type	0x000F	Generic
source_telematics_id	0x0010	GSM
qos_time_to_live	0x0017	Generic
payload_type	0x0019	Generic
additional_status_info_text	0x001D	Generic
receipted_message_id	0x001E	Generic
ms_msg_wait_facilities	0x0030	GSM
privacy_indicator	0x0201	CDMA, TDMA
source_subaddress	0x0202	CDMA, TDMA

dest_subaddress	0x0203	CDMA, TDMA
user_message_reference	0x0204	Generic
user_response_code	0x0205	CDMA, TDMA
source_port	0x020A	Generic
destination_port	0x020B	Generic
sar_msg_ref_num	0x020C	Generic
language_indicator	0x020D	CDMA, TDMA
sar_total_segments	0x020E	Generic
sar_segment_seqnum	0x020F	Generic
sc_interface_version	0x0210	Generic
callback_num_pres_ind	0x0302	TDMA
callback_num_atag	0x0303	TDMA
number_of_messages	0x0304	CDMA
callback_num	0x0381	CDMA, TDMA, GSM, iDEN
dpf_result	0x0420	Generic
set_dpf	0x0421	Generic
ms_availability_status	0x0422	Generic
network_error_code	0x0423	Generic
message_payload	0x0424	Generic
delivery_failure_reason	0x0425	Generic
more_messages_to_send	0x0426	GSM
message_state	0x0427	Generic
ussd_service_op	0x0501	GSM (USSD)
display_time	0x1201	CDMA, TDMA
sms_signal	0x1203	TDMA
ms_validity	0x1204	CDMA, TDMA
alert_on_message_delivery	0x130C	CDMA
its_reply_type	0x1380	CDMA
its_session_info	0x1383	CDMA

Table 4-48 TLV Tag Definitions

4.6.2 TLV Length

This is a 2-octet field used to specify the length of the Value field within a TLV.

4.6.3 TLV Value

This is a variable size field used to contain the TLV payload for each specific TLV.

4.6.4 TLV Definitions

4.6.4.1 dest_addr_subunit

The *dest_addr_subunit* parameter is used to route messages when received by a mobile station, for example to a smart card in the mobile station or to an external device connected to the mobile station.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0005
Length	2	Integer	Length of Value part in octets
Value	1	Integer	0x00 = Unknown (default) 0x01 = MS Display 0x02 = Mobile Equipment 0x03 = Smart Card 1 (expected to be SIM if a SIM exists in the MS) 0x04 = External Unit 1 5 to 255 = reserved

Table 4-49 *dest_addr_subunit* TLV

4.6.4.2 source_addr_subunit

The *source_addr_subunit* parameter is used to indicate where a message originated in the mobile station, for example a smart card in the mobile station or an external device connected to the mobile station.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x000D
Length	2	Integer	Length of Value part in octets
Value	1	Integer	see 4.6.4.1

Table 4-50 *source_addr_subunit* TLV

4.6.4.3 dest_network_type

The *dest_network_type* parameter is used to indicate a network type associated with the destination address of a message. In the case that the receiving system (e.g. SMSC) does not support the indicated network type, it may treat this a failure and return a response PDU reporting a failure.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0006
Length	2	Integer	Length of Value part in octets
Value	1	Integer	0x00 = Unknown 0x01 = GSM 0x02 = ANSI-136/TDMA 0x03 = IS-95/CDMA 0x04 = PDC 0x05 = PHS 0x06 = iDEN 0x07 = AMPS 0x08 = Paging Network 9 to 255 = reserved

Table 4-51 *dest_network_type* TLV

4.6.4.4 source_network_type

The *source_network_type* parameter is used to indicate the network type associated with the device that originated the message.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	source_network_type
Length	2	Integer	Length of Value part in octets
Value	1	Integer	see 5.3.2.3

Table 4-52 *source_network_type* TLV

4.6.4.5 dest_bearer_type

The *dest_bearer_type* parameter is used to request the desired bearer for delivery of the message to the destination address. In the case that the receiving system (e.g. SMSC) does not support the indicated bearer type, it may treat this a failure and return a response PDU reporting a failure.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	dest_bearer_type
Length	2	Integer	Length of Value part in octets
Value	1	Integer	0x00 = Unknown 0x01 = SMS 0x02 = Circuit Switched Data (CSD) 0x03 = Packet Data 0x04 = USSD 0x05 = CDPD 0x06 = DataTAC 0x07 = FLEX/ReFLEX 0x08 = Cell Broadcast (cellcast) 9 to 255 = reserved

Table 4-53 *dest_bearer_type* TLV

4.6.4.6 source_bearer_type

The *source_bearer_type* parameter indicates the wireless bearer over which the message originated.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x000F
Length	2	Integer	Length of Value part in octets
Value	1	Integer	see 4.6.4.5

Table 4-54 *source_bearer_type* TLV

4.6.4.7 dest_telematics_id

This parameter defines the telematic interworking to be used by the delivering system for the destination address. This is only useful when a specific *dest_bearer_type* parameter has also been specified as the value is bearer dependent. In the case that the receiving system (e.g. SMSC) does not support the indicated telematic interworking, it may treat this a failure and return a response PDU reporting a failure.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0008
Length	2	Integer	Length of Value part in octets
Value	2	Integer	to be defined

Table 4-55 *dest_telematics_id* TLV

4.6.4.8 source_telematics_id

The *source_telematics_id* parameter indicates the type of telematics interface over which the message originated.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0010
Length	2	Integer	Length of Value part in octets
Value	1	Integer	see 4.6.4.7

Table 4-56 *source_telematics_id* TLV

4.6.4.9 qos_time_to_live

This parameter defines the number of seconds which the sender requests the SMSC to keep the message if undelivered before it is deemed expired and not worth delivering. If the parameter is not present, the SMSC may apply a default value.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0017
Length	2	Integer	Length of Value part in octets
Value	4	Integer	number of seconds for message to be retained by the receiving system.

Table 4-57 *qos_time_to_live* TLV

4.6.4.10 payload_type

The *payload_type* parameter defines the higher layer PDU type contained in the message payload.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0019
Length	2	Integer	Length of Value part in octets
Value	1	Integer	0 Default. In the case of a WAP application, the default higher layer message type is a WDP message. See [WDP] for details. 1WCMP message. Wireless Control Message Protocol formatted data. See [WCMP] for details. values 2 to 255 are reserved

Table 4-58 *payload_type* TLV

4.6.4.11 additional_status_info_text

The *additional_status_info_text* parameter gives an ASCII textual description of the meaning of a response PDU. It is to be used by an implementation to allow easy diagnosis of problems.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x001D
Length	2	Integer	Length of Value part in octets
Value	1 - 256	C Octet String	Free format text to allow implementations to supply the most useful information for problem diagnosis. Maximum length is 256 octets.

Table 4-59 *additional_status_info_text* TLV

4.6.4.12 receipted_message_id

The *receipted_message_id* parameter indicates the ID of the message being receipted in an SMSC Delivery Receipt. This is the opaque SMSC message identifier that was returned in the *message_id* parameter of the SMPP response PDU that acknowledged the submission of the original message.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x001E
Length	2	Integer	Length of Value part in octets
Value	1 - 65	C Octet String	SMSC handle of the message being receipted.

Table 4-60 *receipted_message_id* TLV

4.6.4.13 ms_msg_wait_facilities

The *ms_msg_wait_facilities* parameter allows an indication to be provided to an MS that there are messages waiting for the subscriber on systems on the PLMN. The indication can be an icon on the MS screen or other MMI indication.

The *ms_msg_wait_facilities* can also specify the type of message associated with the message waiting indication.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0030
Length	2	Integer	Length of Value part in octets
Value	1	Bit mask	Bits 7.....0 I00000TT This parameter controls the indication and specifies the message type (of the message associated with the MWI) at the mobile station. The Indicator is encoded in bit 7 as follows: 0 = Set Indication Inactive 1 = Set Indication Active The Type of Message associated with the MWI is encoded in bits 0 and 1 as follows: 00 = Voicemail Message Waiting 01 = Fax Message Waiting 10 = Electronic Mail Message Waiting 11 = Other Message Waiting

Table 4-61 *ms_msg_wait_facilities* TLV

4.6.4.14 privacy_indicator

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0201
Length	2	Integer	Length of value part in octets
Value	1	Integer	0 = Privacy Level 0 (Not Restricted) (default) 1 = Privacy Level 1 (Restricted) 2 = Privacy Level 2 (Confidential) 3 = Privacy Level 3 (Secret) values 4 to 255 are reserved

Table 4-62 *privacy_indicator* TLV

The *privacy_indicator* indicates the privacy level of the message.

4.6.4.15 source_subaddress

The *source_subaddress* parameter specifies a subaddress associated with the originator of the message.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0202
Length	2	Integer	Length of Value part in octets
Value	Var 2 - 23	Octet String	The first octet of the data field is a Type of Subaddress tag and indicates the type of subaddressing information included, and implies the type and length of subaddressing information which can accompany this tag value in the data field. Valid Tag values are: 00000001 - Reserved 00000010 - Reserved 10000000 - NSAP (Even) [ITUT X.213] 10001000 - NSAP (Odd) [ITUT X.213] 10100000 - User Specified All other values Reserved The remaining octets contain the subaddress. A NSAP address shall be encoded using the preferred binary encoding specified in [ITUT X.213]. In this case the subaddress field contains the Authority and Format Identifier. A User Specified subaddress is encoded according to user specification, subject to a maximum of 22 octets.

Table 4-63 *source_subaddress* TLV

4.6.4.16 dest_subaddress

The *dest_subaddress* parameter specifies a subaddress associated with the destination of the message.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0203
Length	2	Integer	Length of Value part in octets
Value	Var 2 - 23	Octet String	See 4.6.4.15 for parameter encoding.

Table 4-64 *dest_subaddress* TLV

The *dest_subaddress* parameter is not supported in the SMPP ***submit_multi*** PDU.

4.6.4.17 user_message_reference

A reference assigned by the originating SME to the short message.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0204
Length	2	Integer	Length of value part in octets
Value	2	Integer	All values allowed.

Table 4-65 *user_message_reference* TLV

4.6.4.18 user_response_code

A response code set by the user in a User Acknowledgement/Reply message. The response codes are application specific.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0205
Length	2	Integer	Length of value part in octets
Value	1	Integer	0 to 255 (IS-95 CDMA) 0 to 15 (CMT-136 TDMA)

Table 4-66 *user_response_code* TLV

4.6.4.19 language_indicator

The *language_indicator* parameter is used to indicate the language of the short message.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x020D
Length	2	Integer	Length of value part in octets
Value	1	Integer	0 = unspecified (default) 1 = english 2 = french 3 = spanish 4 = german 5 = Portuguese refer to [CMT-136] for other values

Table 4-67 *language_indicator* TLV

4.6.4.20 source_port

The *source_port* parameter is used to indicate the application port number associated with the source address of the message.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x020A
Length	2	Integer	Length of value part in octets
Value	2	Integer	All values allowed.

Table 4-68 *source_port* TLV

4.6.4.21 destination_port

The *destination_port* parameter is used to indicate the application port number associated with the destination address of the message.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x020B
Length	2	Integer	Length of value part in octets
Value	2	Integer	All values allowed.

Table 4-69 *destination_port* TLV

4.6.4.22 sar_msg_ref_num

The *sar_msg_ref_num* parameter is used to indicate the reference number for a particular concatenated short message.

The concatenation related parameters are sar_msg_ref_num, sar_total_segments and sar_segment_seqnum. Where these are present the other parameters of the message should remain unchanged for each short message fragment which forms part of a mobile terminated concatentated short message, with the exception of those parameters for which it makes sense to change them (such as the user data in the short_message parameter).[CL23]

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x020C
Length	2	Integer	Length of value part in octets
Value	2	Integer	This parameter shall contain an originator generated reference number so that a segmented short message may be reassembled into a single original message. This allows the parallel transmission of several segmented messages. This reference number shall remain constant for every segment which makes up a particular concatenated short message. When present, the PDU must also contain the sar_total_segments and sar_segment_seqnum parameters. Otherwise this parameter shall be ignored.

Table 4-70 *sar_msg_ref_num* TLV

4.6.4.23 sar_total_segments

The *sar_total_segments* parameter is used to indicate the total number of short messages within the concatenated short message.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x020E
Length	2	Integer	Length of value part in octets
Value	1	Integer	This parameter shall contain a value in the range 1 to 255 indicating the total number of fragments within the concatenated short message. The value shall start at 1 and remain constant for every short message, which makes up the concatenated short message. When present, the PDU must also contain the sar_msg_ref_num and sar_segment_seqnum parameters. Otherwise this parameter shall be ignored.

Table 4-71 *sar_total_segments* TLV

4.6.4.24 sar_segment_seqnum

The *sar_segment_seqnum* parameter is used to indicate the sequence number of a particular short message within the concatenated short message.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x020F
Length	2	Integer	Length of value part in octets
Value	1	Integer	This octet shall contain a value in the range 1 to 255 indicating the sequence number of a particular message within the concatenated short message. The value shall start at 1 and increment by one for every message sent within the concatenated short message. When present, the PDU must also contain the sar_total_segments and sar_msg_ref_num parameters. Otherwise this parameter shall be ignored.

Table 4-72 *sar_segment_seqnum* TLV

4.6.4.25 sc_interface_version

The *sc_interface_version* parameter is used to indicate the SMPP version supported by the SMSC. It is returned in the bind response PDUs.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0210
Length	2	Integer	Length of value part in octets
Value	1	Integer	values as per 5.2.4. (interface_version)

Table 4-73 *sc_interface_version* TLV

4.6.4.26 display_time

The *display_time* parameter is used to associate a display time of the short message on the MS.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x1201
Length	2	Integer	Length of value part in octets
Value	1	Integer	0 = Temporary 1 = Default (default) 2 = Invoke values 3 to 255 are reserved

Table 4-74 *display_time* TLV

4.6.4.27 ms_validity

The *ms_validity* parameter is used to provide an MS with validity information associated with the received short message.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x1204
Length	2	Integer	Length of value part in octets
Value	1	Integer	0 = Store Indefinitely (default) 1 = Power Down 2 = SID based registration area 3 = Display Only values 4 to 255 are reserved

Table 4-75 *ms_validity* TLV

4.6.4.28 dpf_result

The *dpf_result* parameter is used in the *data_sm_resp* PDU to indicate if delivery pending flag (DPF) was set for a delivery failure of the short message..

If the *dpf_result* parameter is not included in the *data_sm_resp* PDU, the ESME should assume that DPF is not set.

Currently this parameter is only applicable for the Transaction message mode.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0420
Length	2	Integer	Length of value part in octets
Value	1	Integer	0 = DPF not set 1 = DPF set values 2 to 255 are reserved

Table 4-76 *dpf_result* TLV

4.6.4.29 set_dpf

An ESME may use the *set_dpf* parameter to request the setting of a delivery pending flag (DPF) for certain delivery failure scenarios, such as:

- MS is unavailable for message delivery (as indicated by the HLR)

The SMSC should respond to such a request with an *alert_notification* PDU when it detects that the destination MS has become available.

The delivery failure scenarios under which DPF is set is SMSC implementation and network implementation specific. If a delivery pending flag is set by the SMSC or network (e.g. HLR), then the SMSC should indicate this to the ESME in the *data_sm_resp* message via the *dpf_result* parameter.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0421
Length	2	Integer	length of value part in octets
Value	1	Integer	0 = Setting of DPF for delivery failure to MS not requested 1 = Setting of DPF for delivery failure requested (default) values 2 to 255 are reserved

Table 4-77 *set_dpf* TLV

4.6.4.30 ms_availability_status

The *ms_availability_status* parameter is used in the *alert_notification* operation to indicate the availability state of the MS to the ESME.

If the SMSC does not include the parameter in the *alert_notification* operation, the ESME should assume that the MS is in an “available” state.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0422
Length	2	Integer	Length of value part in octets
Value	1	Integer	0 = Available (Default) 1 = Denied (e.g. suspended, no SMS capability, etc.) 2 = Unavailable values 3 to 255 are reserved

Table 4-78 *ms_availability_status* TLV

4.6.4.31 network_error_code

The *network_error_code* parameter is used to indicate the actual network error code for a delivery failure. The network error code is technology specific.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0423
Length	2	Integer	Length of value part in octets
Value	3	Octet String	Sub-field / Size / Type Network Type / 1 / Integer Error Code / 2 / Integer The first octet indicates the network type. The following values are defined: 1 = ANSI 136 Access Denied Reason 2 = IS 95 Access Denied Reason 3 = GSM 4 = ANSI 136 Cause Code 5 = IS 95 Cause Code 6 = ANSI-41 Error 7 = SMPP Error 8 = Message Centre Specific All other values reserved. The remaining two octets specify the actual network error code appropriate to the network type.[CL24]

Table 4-79 *network_error_code* TLV

4.6.4.32 message_payload

The *message_payload* parameter contains the user data. Its function is to provide an alternative means of carrying text lengths above the 255 octet limit of the short_message field.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0424
Length	2	Integer	Set to length of user data
Value	Variable	Octet String	Short message user data. The maximum size is SMSC and network implementation specific.

Table 4-80 *message_payload* TLV

4.6.4.33 delivery_failure_reason

The *delivery_failure_reason* parameter is used in the *data_sm_resp* operation to indicate the outcome of the message delivery attempt (only applicable for transaction message mode). If a delivery failure due to a network error is indicated, the ESME may check the *network_error_code* parameter (if present) for the actual network error code.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0425
Length	2	Integer	Length of value part in octets
Value	1	Integer	0 = Destination unavailable 1 = Destination Address Invalid (e.g. suspended, no SMS capability, etc.) 2 = Permanent network error 3 = Temporary network error values 4 to are 255 reserved

Table 4-81 *delivery_failure_reason* TLV

The *delivery_failure_reason* parameter is not included if the delivery attempt was successful.

4.6.4.34 more_messages_to_send

The *more_messages_to_send* parameter is used by the ESME in the ***submit_sm*** and *data_sm* operations to indicate to the SMSC that there are further messages for the same destination SME. The SMSC may use this setting for network resource optimization.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0426
Length	2	Integer	Length of value part in octets
Value	1		0 = No more messages to follow 1 = More messages to follow (default) values 2 to 255 are reserved

Table 4-82 *more_messages_to_send* TLV

4.6.4.35 message_state

The *message_state* optional parameter is used by the SMSC in the *deliver_sm* and *data_sm* PDUs to indicate to the ESME the final message state for an SMSC Delivery Receipt.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0427
Length	2	Integer	Length of value part in octets
Value	1		Values as per section 5.2.28

Table 4-83 *message_state* TLV

4.6.4.36 callback_num

The *callback_num* parameter associates a call back number with the message. In TDMA networks, it is possible to send and receive multiple callback numbers to/from TDMA mobile stations.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0381
Length	2	Integer	Length of Value part in octets
Value	Var 4 - 19	Octet String	Bits 7.....0 0000000D (octet 1) 00000TTT (octet 2) 0000NNNN (octet 3) XXXXXXXX (octet 4) ⋮ ⋮ XXXXXXXX (octet N) The originating SME can set a Call Back Number for the receiving Mobile Station. The first octet contains the Digit Mode Indicator. Bit D=0 indicates that the Call Back Number is sent to the mobile as DTMF digits encoded in TBCD. Bit D=1 indicates that the Call Back Number is sent to the mobile encoded as ASCII digits. The 2nd octet contains the Type of Number (TON). Encoded as in section 5.2.5. The third octet contains the Numbering Plan Indicator (NPI). Encoded as specified in section 5.2.6 The remaining octets contain the Call Back Number digits encoded as ASCII characters

Table 4-84 *callback_num* TLV

4.6.4.37 callback_num_pres_ind

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0302
Length	2	Integer	Length of Value part in octets
Value	1	Bit mask	Bits 7.....0 0000ppss This parameter controls the presentation indication and screening of the CallBackNumber at the mobile station.If present, the callback_num parameter must also be present. The Presentation Indicator is encoded in bits 2 and 3 as follows: 00 = Presentation Allowed 01 = Presentation Restricted 10 = Number Not Available 11 = Reserved The Screening Indicator is encoded in bits 0 and 1 as follows: 00 = User provided, not screened 01 = User provided, verified and passed 10 = User provided, verified and failed 11 = Network Provided.

Table 4-85 *callback_num_pres_ind* TLV

4.6.4.38 callback_num_atag

The *callback_num_atag* parameter associates an alphanumeric display with the call back number.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0303
Length	2	Integer	Length of Value part in octets
Value	Var max 65	Octet string	Alphanumeric display tag for call back number Bits 7.....0 EEEEEEEE (octet 1) XXXXXXXX (octet 2) ⋮ ⋮ XXXXXXXX (octet N) The first octet contains the encoding scheme of the Alpha Tag display characters. This field contains the same values as for Data Coding Scheme (see section 5.2.19). The following octets contain the display characters: There is one octet per display character for 7-bit and 8-bit encoding schemes. There are two octets per display character for 16-bit encoding schemes.

Table 4-86 *callback_num_atag* TLV

4.6.4.39 number_of_messages

The *number_of_messages* parameter is used to indicate the number of messages stored in a mailbox.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0304
Length	2	Integer	Length of Value part in octets
Value	1	Integer	0 to 99 = allowed values. values 100 to 255 are reserved

Table 4-87 *number_of_messages* TLV

4.6.4.40 sms_signal

The *sms_signal* parameter is used to provide a TDMA MS with alert tone information associated with the received short message.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x1203
Length	2	Integer	Length of Value part in octets
Value	2	Integer	Encoded as per [CMT-136]

Table 4-88 *sms_signal* TLV

4.6.4.41 alert_on_message_delivery

The *alert_on_message_delivery* parameter is set to instruct a MS to alert the user (in a MS implementation specific manner) when the short message arrives at the MS.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x130C
Length	2	Integer	Length of Value part in octets (= 0)
Value	0		There is no Value part associated with this parameter.

Table 4-89 *alert_on_message_delivery* TLV

4.6.4.42 its_reply_type

The *its_reply_type* parameter is a required parameter for the CDMA Interactive Teleservice as defined by the Korean PCS carriers [KORITS]. It indicates and controls the MS user's reply method to an SMS delivery message received from the ESME.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x1380
Length	2	Integer	Length of Value part in octets
Value	1	Integer	0 = Digit 1 = Number 2 = Telephone No. 3 = Password 4 = Character Line 5 = Menu 6 = Date 7 = Time 8 = Continue values 9 to 255 are reserved

Table 4-90 *its_reply_type* TLV

4.6.4.43 its_session_info

The *its_session_info* parameter is a required parameter for the CDMA Interactive Teleservice as defined by the Korean PCS carriers [KORITS]. It contains control information for the interactive session between an MS and an ESME.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x1383
Length	2	Integer	Length of Value part in octets
Value	2	Octet String	Bits 7.....0 SSSSSSSS (octet 1) NNNNNNNE (octet 2) Octet 1 contains the session number (0 - 255) encoded in binary. The session number remains constant for each session. The sequence number of the dialogue unit (as assigned by the ESME) within the session is encoded in bits 7..1 of octet 2. The End of Session Indicator indicates the message is the end of the conversation session and is encoded in bit 0 of octet 2 as follows: 0 = End of Session Indicator inactive. 1 = End of Session Indicator active.

Table 4-91 *its_session_info* TLV

4.6.4.44 ussd_service_op

The *ussd_service_op* parameter is required to define the USSD service operation when SMPP is being used as an interface to a (GSM) USSD system.

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0501
Length	2	Integer	Length of Value part in octets
Value	1	Integer	0 = PSSD indication 1 = PSSR indication 2 = USSR request 3 = USSN request 4 to 15 = reserved 16 = PSSD response 17 = PSSR response 18 = USSR confirm 19 = USSN confirm 20 to 31 = reserved 32 to 255 = reserved for vendor specific USSD operations

Table 4-92 *ussd_service_op* TLV

4.6.4.45 congestion_state

The *congestion_state* parameter is used to pass congestion status information between ESME and MC as a means of providing flow control and congestion avoidance capabilities to the sending peer. The TLV can be used in any SMPP operation response PDU as a means of passing congestion status from one peer to another. Typical uses of this would in submit_sm/submit_sm_resp sequences where an ESME would drive a batch of submissions at a high rate and use continual tracking of the returned *congestion_state* values as a means of gauging the congestion, increasing/decreasing the rate as required to maintain the balance in the Optimum range.[CL25]

Field	Size octets	Type	Description
Parameter Tag	2	Integer	0x0428
Length	2	Integer	Length of Value part in octets
Value	1	Integer	0 = Idle 1-29 = Low Load 30-49 = Medium Load 50-79 = High Load 80-90 = Optimum Load 90-99 = Nearing Congestion 100 = Congested / Maximum Load All other values reserved

Table 4-93 *congestion_state* TLV

5 Change Log

Version Change	Description
V5.0 Draft04 - V5.0 Draft05	Added Change Log to end of document for generalising various changes across draft versions. Incorporated changes from Chris Wright to include definition of REs and rewording of introductary chapter to suit. This includes a newly modified network diagram to depict the ESMEs, MCs & REs. Modified network_error_code TLV to include details from enhancement CR and comments from Seattle Plenary Modified TLV definitions to include actual TLV tag value in Hex instead of TLV name. Change based on similar approach taken with command_id in PDU definitions 3GPP references as provided by Edwin Sandberg, added to references section.
V5.0 Draft05 – V5.0 Draft06	Made several wording corrections, spelling, capitalisation and grammar corrections throughout document. Removed HEADER/BODY side columns from PDUs as these were causing problems with PDU formatting. Something else, possibly shading or heavy borders will have to be added to provide this separation Added Message Delivery TLVs section under data_sm and deliver_sm definition. Added *submit_multi* and *replace_sm* PDU definitions Incorporated CRs: • Minor SAR TLV clarification • Data_sm short_message/message_payload error (This was not incorporated as specified as document structure has separated deliver ACKs and other message content away from submit_sm/data_sm where it is discussed more generally and in a non- “short_message” oriented manner. • AT&T 2 Error Code CRs • BTC ussd_service_op CR for addition of specified TLV to deliver_sm and data_sm operations • FC & CA CR • Network Error Code Enhancement CR

Table 5-1 Change Log