Oracle Communications Network Charging and Control

Product: OCNCC 4.3

Component: Short Message Peer-to-Peer Protocol (SMPP)

Protocol Implementation Conformance Statement

S’ware version: Release 3.1.1

Guide version: 02.00

Release date: December 2010

Status: Approved
Copyright

Short Message Peer-to-Peer Protocol (SMPP) Protocol Implementation
Conformance Statement, Release 3.1.1

02.00

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.
# Contents

Copyright .......................................................................................................................... Error! Bookmark not defined.
About this Document ........................................................................................................ v
Document Conventions ..................................................................................................... vi

## Chapter 1

**Messaging Manager and SMPP Document Versions**

Overview ............................................................................................................................. 1
Messaging Manager ............................................................................................................ 2
SMPP ................................................................................................................................. 3

## Chapter 2

**Compliance Statements For SMPP Sessions (2)**

Overview ............................................................................................................................. 5
References to The Specification ......................................................................................... 6
Specification Clauses 2.1 and 2.2 ....................................................................................... 7
Session States (2.3) ............................................................................................................ 8
Operation Matrix (2.4) ..................................................................................................... 9
PDU Sequencing (2.6) ....................................................................................................... 12
Session Timers (2.7) ......................................................................................................... 13
Error Handling (2.8) ........................................................................................................ 14
Flow Control and Congestion Avoidance (2.9) ............................................................... 15
Session Security and Encryption (2.10) .......................................................................... 16
Forward and Backward Compatibility (2.11) ............................................................... 17

## Chapter 3

**Compliance Statements For SMPP Parameter and PDU Format (3)**

Overview .......................................................................................................................... 19
Parameter Type Definitions (3.1) ..................................................................................... 20
General PDU Format (3.2) ............................................................................................. 21

## Chapter 4

**Compliance Statements For SMPP PDU Definitions (4)**

Overview .......................................................................................................................... 23
Session Management Operations (4.1) ........................................................................... 24
Message Submission Operations (4.2) ......................................................................... 26
Message Delivery Operations (4.3) .............................................................................. 33
Message Broadcast Operations (4.4) .......................................................................... 35
Ancillary Submission Operations (4.5) ....................................................................... 36
Ancillary Broadcast Operations (4.6) ........................................................................... 37
PDU Field Definitions (4.7) ......................................................................................... 38
PDU TLV Definitions (4.8) ........................................................................................... 45
Glossary of Terms ............................................................................................................ 51
Index ................................................................................................................................. 55
About this Document

Scope
This document describes the extent to which Messaging Manager conforms to the Short Message Peer-to-Peer Protocol Specification.

Audience
This document is intended to be read by Oracle staff. It has been prepared on the assumption that the reader is familiar with Messaging Manager as well as short message peer-to-peer protocols.

Changes in this document
Here are the changes to the document since the last release.

<table>
<thead>
<tr>
<th>Version no.</th>
<th>Revision Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>01.00</td>
<td>2007-02-05</td>
<td>Initial release.</td>
</tr>
<tr>
<td>02.00</td>
<td>2010-11-05</td>
<td>Re-branded</td>
</tr>
</tbody>
</table>
Document Conventions

**Typographical conventions**

Before you start using this guide, it is important to understand the terms and typographical conventions used in the documentation.

Specialised terms and acronyms are defined in the Glossary at the end of this guide.

<table>
<thead>
<tr>
<th>Formatting convention</th>
<th>Type of information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Special Bold</strong></td>
<td>Items you must select such as menu options, or names of tabs.</td>
</tr>
<tr>
<td></td>
<td>Emphasis within text.</td>
</tr>
<tr>
<td></td>
<td>Names of database tables and fields.</td>
</tr>
<tr>
<td><strong>Italics</strong></td>
<td>Name of a document, chapter, topic or other publication.</td>
</tr>
<tr>
<td><strong>Button</strong></td>
<td>The name of a button to click or a key to press.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> To close the window, either click Close or press Esc.</td>
</tr>
<tr>
<td><strong>Key+Key</strong></td>
<td>Key combinations for which the user must press and hold down one key and then press another.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> Ctrl+P, or Alt+F4.</td>
</tr>
<tr>
<td><strong>Monospace</strong></td>
<td>Text that you must type and examples of code or standard output.</td>
</tr>
<tr>
<td><strong>variable</strong></td>
<td>Used to indicate variables or text that should be replaced.</td>
</tr>
<tr>
<td><strong>menu option &gt; menu option &gt;</strong></td>
<td>Used to indicate the cascading menu option to be selected, or the location path of a file.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> Operator Functions &gt; Report Functions</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> /IN/html/SMS/HelpText/</td>
</tr>
<tr>
<td><strong>hypertext link</strong></td>
<td>Used to indicate a hypertext link on an HTML page.</td>
</tr>
</tbody>
</table>

**Icons**

The following icons are used as visual cues to draw attention to important information.

- **Note:** Indicates useful and complementary information. Explanation, comment, or short expansion of the text object that is intended to catch your attention.

- **Tip:** Indicates practical but non-essential information that makes the solution easier to use or operate (e.g. keyboard shortcut, alternative way to perform a step in a procedure, etc).

- **Warning:** Indicates a caution. If this information is ignored, it could cause possible and irreversible damage to the equipment, data or software.

**Terminology**

This topic explains any terminology specific to this manual.
# Chapter 1

## Messaging Manager and SMPP Document Versions

### Overview

**Introduction**

This chapter defines the version of the Messaging Manager implementation and the SMPP document against which it is compared.

**In this chapter**

This chapter contains the following topics.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messaging Manager</td>
<td>2</td>
</tr>
<tr>
<td>SMPP</td>
<td>3</td>
</tr>
</tbody>
</table>
Messaging Manager

Target platform
- Platform - SPARC Solaris
- OS - SunOS 5.9
- Oracle - 9.2.05

Build environment
- Compiler - GNU GCC 3.2.3
- Binutils - GNU binutils 2.1.4
- bison - 1.35
- flex - 2.5.4

Oracle packages
- Full installation of:
  - SLEE - 3.2.0
  - HssSclf - 3.4.27
    - Plus the following SCP packages:
      - smsScp - 3.0.0
      - acsScp - 2.4.0
      - beApiScp - 2.2.0.5
      - acsCbScp - 2.2.0.6
    - Plus the following SMP packages:
      - smsSms - 3.0.0
      - acsSms - 2.4.0
      - beApiSms - 2.2.0.5
      - acsCbSms - 2.2.0.6
This statement of compliance refers to SMS Forum document entitled:

*Short Message Peer-to-Peer Protocol Specification Version 5.0*

For the purpose of this document, *Short Message Peer-to-Peer Protocol Specification Version 5.0* will be referred to as *The Specification*. 
Chapter 2

Compliance Statements For SMPP Sessions (2)

Overview

Introduction
This chapter states the compliance of Messaging Manager with clauses of Section 2 of The Specification.

In this chapter
This chapter contains the following topics.

- References to The Specification ................................................................. 6
- Specification Clauses 2.1 and 2.2 ................................................................. 7
- Session States (2.3) .................................................................................... 8
- Operation Matrix (2.4) .............................................................................. 9
- PDU Sequencing (2.6) ............................................................................. 12
- Session Timers (2.7) ................................................................................. 13
- Error Handling (2.8) ................................................................................. 14
- Flow Control and Congestion Avoidance (2.9) ...................................... 15
- Session Security and Encryption (2.10) ............................................... 16
- Forward and Backward Compatibility (2.11) ...................................... 17
## References to The Specification

<table>
<thead>
<tr>
<th>Convention</th>
<th>As a cross reference, the clause number of <em>The Specification</em> is included in brackets at the end of each compliance statement title.</th>
</tr>
</thead>
</table>
# Specification Clauses 2.1 and 2.2

| Application Layer Communication (2.1) | For TCP/IP connections, Messaging Manager complies.  
|--------------------------------------|-----------------------------------------------------|
|                                      | For X.25 connections, Messaging Manager does not comply.  
|                                      | Messaging Manager does not support X.25.  
| Establishing a SMPP Session (2.2)   | For TCP/IP connections, Messaging Manager complies.  
|                                      | For X.25 connections, Messaging Manager does not comply.  
|                                      | For SMPP, Messaging Manager can be configured to use any port including IANA standard port 2775.  
|                                      | X.25 is not supported by Messaging Manager.  

### Session States (2.3)

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open (2.3.1)</td>
<td>For TCP/IP connections, Messaging Manager complies. For X.25 connections, Messaging Manager does not comply.</td>
</tr>
<tr>
<td>Bound_TX (2.3.2)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Bound_RX (2.3.3)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Bound_TRX (2.3.4)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Unbound (2.3.5)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Closed (2.3.6)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Outbound (2.3.7)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td></td>
<td>Messaging Manager can ignore outbind requests to a port if the port is configured not to receive.</td>
</tr>
</tbody>
</table>
### Operation Matrix (2.4)

<table>
<thead>
<tr>
<th>Relevance</th>
<th>The following compliance statements refer to Table 2-1 Operation Matrix of The Specification.</th>
</tr>
</thead>
</table>
| alert_notification | Messaging Manager complies.  
Messaging Manager does not construct these messages but does relay them. |
| bind_receiver | Messaging Manager complies. |
| bind_receiver_resp | Messaging Manager complies. |
| bind_transceiver | Messaging Manager complies. |
| bind_transceiver_resp | Messaging Manager complies. |
| bind_transmitter | Messaging Manager complies. |
| bind_transmitter_resp | Messaging Manager complies. |
| broadcast_sm | Messaging Manager does not comply.  
Code exists to decode the incoming broadcast message but ProtocolHandler::stateBound() does not consider this possibility and returns genericNack.  
Messaging Manager does not construct this message. |
| broadcast_sm_resp | Messaging Manager does not comply.  
Code exists to decode the incoming broadcast message but ProtocolHandler::stateBound() does not consider this possibility and returns genericNack.  
Messaging Manager does not construct this message. |
| cancel_broadcast_sm | Messaging Manager does not comply.  
Code exists to decode the incoming broadcast message but ProtocolHandler::stateBound() does not consider this possibility and returns genericNack.  
Messaging Manager does not construct this message. |
| cancel_broadcast_sm_resp | Messaging Manager does not comply.  
Code exists to decode the incoming broadcast message but ProtocolHandler::stateBound() does not consider this possibility and returns genericNack.  
Messaging Manager does not construct this message. |
| cancel_sm | Messaging Manager complies.  
Messaging Manager does not construct these messages but, if in a bound state, sends on received cancel messages. These messages are sent straight to the outgoing ProtocolHandler, bypassing ACS and xmsTrigger. |

*Continued on next page*
### Operation Matrix (2.4), Continued

<table>
<thead>
<tr>
<th>Message Type</th>
<th>Compliance</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>cancel_sm_res</td>
<td>Messaging Manager complies.</td>
<td></td>
</tr>
</tbody>
</table>
  - Messaging Manager sends received cancel response messages if in a bound state.  
  - These messages are sent straight to the outgoing ProtocolHandler, bypassing ACS and xmsTrigger.  
  - Messaging Manager only constructs these messages to reply to the cancel_sm if it is unable to forward the cancel_sm. |
| data_sm      | Messaging Manager complies. | |
| data_sm_res  | Messaging Manager complies. | |
| deliver_sm   | Messaging Manager complies. | |
| deliver_sm_res | Messaging Manager complies. | |
| enquire_link | Messaging Manager complies. | |
| enquire_link_res | Messaging Manager complies. | |
| generic_nack | Messaging Manager complies. | |
| outbind      | Messaging Manager complies. | |
| query_broadcast_sm | Messaging Manager does not comply. |  
  - Code exists to decode the incoming broadcast message, but ProtocolHandler::stateBound() does not consider this possibility and returns a genericNack.  
  - Messaging Manager does not construct this type of message. |
| query_broadcast_sm_res | Messaging Manager does not comply. |  
  - Code exists to decode the incoming broadcast message, but ProtocolHandler::stateBound() does not consider this possibility and returns a genericNack.  
  - Messaging Manager does not construct this type of message. |
| query_sm     | Messaging Manager complies. |  
  - Messaging Manager does not construct these messages.  
  - Messaging Manager relays received query messages if in a bound state. These messages are sent straight to the outgoing ProtocolHandler, bypassing ACS and smsTrigger. |
| query_sm_res | Messaging Manager complies. |  
  - Messaging Manager relays received query messages if in a bound state. These messages are sent straight to the outgoing ProtocolHandler, bypassing ACS and smsTrigger.  
  - Messaging Manager only constructs these messages to reply to a query_sm if it is unable to forward the query_sm. |
### Operation Matrix (2.4), Continued

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>replace_sm</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td></td>
<td>• Messaging Manager does not construct these messages.</td>
</tr>
<tr>
<td></td>
<td>• Messaging Manager relays received replace messages if in a bound state.</td>
</tr>
<tr>
<td></td>
<td>These messages are sent straight to the outgoing ProtocolHandler, bypassing</td>
</tr>
<tr>
<td></td>
<td>ACS and smsTrigger.</td>
</tr>
<tr>
<td>replace_sm_resp</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td></td>
<td>• Messaging Manager relays received replace response messages if in a bound</td>
</tr>
<tr>
<td></td>
<td>state. These messages are sent straight to the outgoing ProtocolHandler,</td>
</tr>
<tr>
<td></td>
<td>bypassing ACS and smsTrigger.</td>
</tr>
<tr>
<td></td>
<td>• Messaging Manager only constructs these messages to reply to a replace_sm</td>
</tr>
<tr>
<td></td>
<td>if it is unable to forward the replace_sm.</td>
</tr>
<tr>
<td>submit_multi</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>submit_multi_resp</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>submit_sm</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>submit_sm_resp</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>unbind</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>unbind_resp</td>
<td>Messaging Manager complies.</td>
</tr>
</tbody>
</table>
## PDU Sequencing (2.6)

<table>
<thead>
<tr>
<th>The PDU Sequence Number (2.6.1)</th>
<th>Messaging Manager complies. Messaging Manager follows the recommended practice of monotonically increasing a sequence number that starts at 1. The number will clock-over at $2^{31}-1$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Why use Monotonically Increasing Sequence numbers? (2.6.2)</td>
<td>Messaging Manager complies. Each Messaging Manager SMPP connection maintains its own sequence number.</td>
</tr>
<tr>
<td>Sequence Numbers Across Sessions (2.6.3)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Synchronous Vs. Asynchronous (2.6.4)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Why Asynchronous? (2.6.5)</td>
<td>Messaging Manager complies.</td>
</tr>
</tbody>
</table>
### Session Timers (2.7)

<table>
<thead>
<tr>
<th>Topic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Relevance</strong></td>
<td>The following compliance statements refer to <em>Table 2-2 SMPP Session Timers of The Specification</em>.</td>
</tr>
<tr>
<td><strong>Session Init Timer</strong></td>
<td>Messaging Manager does not comply. The timer value is the configured outgoingTimeout value. However, when a TCP/IP outbind connection has been established and the ESME is waiting for the outbound message, Messaging Manager obtains a timer value from the idleTimeout configuration option.</td>
</tr>
<tr>
<td><strong>Enquire Link Timer</strong></td>
<td>Messaging Manager complies. The timer value is obtained from the heartbeatTimeout value.</td>
</tr>
<tr>
<td><strong>Inactivity Timer</strong></td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td></td>
<td>• The timer value is obtained from the heartbeatTimeout value.</td>
</tr>
<tr>
<td></td>
<td>• The timer value expires in all states but the expiry handling code checks the current state before taking action.</td>
</tr>
<tr>
<td><strong>Response Timer</strong></td>
<td>Messaging Manager complies. The timer value is the configured outgoingTimeout value.</td>
</tr>
</tbody>
</table>
## Error Handling (2.8)

<table>
<thead>
<tr>
<th>Handling</th>
<th>Messaging Manager complies.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Connection Failure (2.8.1)</strong></td>
<td></td>
</tr>
<tr>
<td>• After a failed SMSC connection attempt, the IP plugin tries every 10 seconds to reconnect.</td>
<td></td>
</tr>
<tr>
<td>• If an established connection is lost, reconnection attempts are only made if the connection is to an SMSC.</td>
<td></td>
</tr>
<tr>
<td>• Section 2.8.1 recommends retry for outbounds, but this is not performed by Messaging Manager itself.</td>
<td></td>
</tr>
<tr>
<td><strong>Operation Failure (2.8.2)</strong></td>
<td>The following six statements refer to the bulleted list in <em>The Standard</em>.</td>
</tr>
<tr>
<td><strong>The PDU is unrecognised</strong></td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>If a command ID cannot be determined, a genericNack is returned with ESME_RINVCMDID set.</td>
<td></td>
</tr>
<tr>
<td><strong>The PDU is malformed</strong></td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>• If a command ID cannot be determined, a genericNack is returned with ESME_RINVCMDID set.</td>
<td></td>
</tr>
<tr>
<td>• If the section length of a PDU is the reason for the command ID not being determined, ESME_RINVCMDLEN is returned in a genericNack.</td>
<td></td>
</tr>
<tr>
<td>• See also <code>command_status, error_status_code (4.7.6)</code> (on page 38).</td>
<td></td>
</tr>
<tr>
<td><strong>Invalid Field Length</strong></td>
<td>Messaging Manager does not comply.</td>
</tr>
<tr>
<td>The type of message returned is a genericNack with ESME_RINVCMDID. This is not a response of the appropriate type.</td>
<td></td>
</tr>
<tr>
<td><strong>The PDU data is unexpected and deemed invalid</strong></td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Messaging Manager does not need to consider any message data as invalid.</td>
<td></td>
</tr>
<tr>
<td><strong>The PDU is not allowed in the current session state</strong></td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Where the received message does not have an appropriate response message type, genericNack is used.</td>
<td></td>
</tr>
<tr>
<td><strong>The ESME or MC is restricting the use of certain PDUs or features</strong></td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Messaging Manager does not restrict the use of certain PDUs.</td>
<td></td>
</tr>
</tbody>
</table>
Flow Control and Congestion Avoidance (2.9)

**Compliance statement**

Messaging Manager does not comply.

- Messages that are unaltered by Messaging Manager and leave via the originating plugin will pass on any encoded congestion_state TLV correctly. However, all Messaging Manager constructed messages, and messages arriving from other plugins will not add a congestion_state TLV to a response.

- To become fully compliant, the GenericSM class:
  - needs to be extended to include a representation of the congestion_state TLV, and
  - needs to populate the TLV from the GenericSM in the outgoing plugin.

Some method of determining Messaging Manager's own congestion state (and populating GenericSM with it) would also be desirable.
## Session Security and Encryption (2.10)

<table>
<thead>
<tr>
<th>Leased Lines (2.10.1)</th>
<th>Messaging Manager complies. The privacy of the network where Messaging Manager is deployed is obviously not determined by Messaging Manager itself.</th>
</tr>
</thead>
</table>
| Secure Transport Layer (2.10.2) | Messaging Manager does not comply.  
- The SMPP plugin uses cmn::Socket for its connections. cmn::Socket does not support SSL.  
- To become compliant:  
  - SSL support needs to be added to the socket class, and  
  - configuration for the SSL connection needs to be added to the plugin. |
| Secure VPN (2.10.3) | Messaging Manager complies.  
For this type of encryption, there are no demands placed on either the ESME or MC. |
| Secure Tunnel (2.10.4) | Messaging Manager complies.  
For this type of encryption, there are no demands placed on either the ESME or MC. |
## Forward and Backward Compatibility (2.11)

**General**

Messaging Manager complies with clause 2.11.

With bind requests, Messaging Manager sets `interface_version` to either the ASP's version or the version in Messaging Manager's `eserv.config`, whichever is the smaller. Only 0x34 and 0x50 are considered valid values.

**Forward Compatibility (2.11.1)**

Messaging Manager complies.

If a message leaves by the plugin that received it, unrecognised TLVs are inserted into the outgoing message.

**Backward Compatibility (2.11.2)**

Messaging Manager does not comply.

- Messaging Manager does not correctly support connections to an ESME or MC that only supports SMPP version 3.3 or earlier.
- In several places Messaging Manager adds TLVs to messages which may be SMPP version 3.3.
- No check on the `messageId` size is made, so it is possible to send a `messageId` greater than 8 octets in size.
Overview

Introduction
This chapter states the compliance of Messaging Manager with clauses of Section 3 of *The Specification*.

In this chapter
This chapter contains the following topics.

Parameter Type Definitions (3.1) ................................................................. 20
General PDU Format (3.2) ........................................................................... 21
### Parameter Type Definitions (3.1)

<table>
<thead>
<tr>
<th>SMPP PDU Parameter Types (Table 3-1)</th>
<th>Messaging Manager complies.</th>
</tr>
</thead>
<tbody>
<tr>
<td>NULL Settings (3.1.1)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>SMPP Parameter Field Size Notation (3.1.2)</td>
<td>Messaging Manager complies.</td>
</tr>
</tbody>
</table>
# General PDU Format (3.2)

<table>
<thead>
<tr>
<th>SMPP PDU Format (Table 3-4)</th>
<th>Messaging Manager complies.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDU Format (3.2.1)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Command_length (3.2.1.1)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Command_id (3.2.1.2)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Command_status (3.2.1.3)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Sequence_number (3.2.1.4)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>Standard Parameters (3.2.1.5)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>TLV Parameters (3.2.1.6)</td>
<td>Messaging Manager complies.</td>
</tr>
</tbody>
</table>

**A sample PDU (3.2.2)**

Responses will not include a message body if the command status is non-zero.

Note that for mandatory TLVs, Messaging Manager expects the order to be the same as that specified in *The Standard*. 

Messaging Manager complies.
Chapter 4

Compliance Statements For SMPP PDU Definitions

(4)

Overview

<table>
<thead>
<tr>
<th>Introduction</th>
<th>This chapter states the compliance of Messaging Manager with clauses of Section 4 of The Specification.</th>
</tr>
</thead>
<tbody>
<tr>
<td>In this chapter</td>
<td>This chapter contains the following topics.</td>
</tr>
<tr>
<td></td>
<td>Session Management Operations (4.1) ...........................................................................24</td>
</tr>
<tr>
<td></td>
<td>Message Submission Operations (4.2) ...........................................................................26</td>
</tr>
<tr>
<td></td>
<td>Message Delivery Operations (4.3) ...............................................................................33</td>
</tr>
<tr>
<td></td>
<td>Message Broadcast Operations (4.4) ...............................................................................35</td>
</tr>
<tr>
<td></td>
<td>Ancillary Submission Operations (4.5) ..........................................................................36</td>
</tr>
<tr>
<td></td>
<td>Ancillary Broadcast Operations (4.6) .............................................................................37</td>
</tr>
<tr>
<td></td>
<td>PDU Field Definitions (4.7) .........................................................................................38</td>
</tr>
<tr>
<td></td>
<td>PDU TLV Definitions (4.8) ...........................................................................................45</td>
</tr>
</tbody>
</table>
Session Management Operations (4.1)

When specified, no limits are enforced for the parameters described in this section. However limits exist in the Messaging Manager database and the Routing Scheme subsystem which will limit the size of the specific parameter being passed on to the SMPP interface:

- The database and routing scheme limit the password to 50 and 51 characters respectively.
- The database and routing Scheme limit the system_id to 15 and 16 characters respectively.

### General

- The database and routing scheme limit the password to 50 and 51 characters respectively.
- The database and routing Scheme limit the system_id to 15 and 16 characters respectively.

### Bind Operation

#### bind_transmitter Syntax (4.1.1)

- No limit is placed on the length of a C-Octet string.

- If configured incorrectly, Messaging Manager creates bind operations with passwords longer than nine characters.

#### bind_transmitter_resp Syntax (4.1.1.2)

- The maximum of 16 characters for system_id is not enforced.

- sc_interface_version is not used.

#### bind_receiver Syntax (4.1.1.3)

- The maximum of 16 characters for system_id is not enforced.

#### bind_receiver_resp Syntax (4.1.1.4)

- The maximum of 16 characters for system_id is not enforced.

- sc_interface_version is not used.

#### bind_transceiver Syntax (4.1.1.5)

- The maximum number of characters is not enforced for the variable length fields.

#### bind_transceiver_resp Syntax (4.1.1.6)

- The maximum number of characters is not enforced for the variable length fields.

- sc_interface_version is not used.

#### outbind Syntax (4.1.1.7)

- The maximum number of characters is not enforced for the variable length fields.

#### unbind Syntax (4.1.1.8)

- Messaging Manager complies.

#### unbind_resp Syntax (4.1.1.9)

- Messaging Manager complies.

#### Enquire Link Operation (4.1.2)

- Messaging Manager complies.

- Messaging Manager takes any message type as a valid response.

#### enquire_link Syntax (4.1.2.1)

- Messaging Manager complies.
### Session Management Operations (4.1), Continued

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>enquire_link_resp</strong> Syntax (4.1.2.2)</td>
<td>Messaging Manager complies.</td>
</tr>
</tbody>
</table>
| **Alert Notification Operation (4.1.3)** | Messaging Manager complies.  
Messaging Manager does not construct these messages. It forwards them from the sender to the receiver. |
| **alert_notification Syntax (4.1.3.1)** | Messaging Manager complies.  
Messaging Manager can decode and encode these messages correctly, but does not create them. In its current form, if Messaging Manager were to create an alert_notification message, the Address size maximum would not be enforced. |
| **Generic NACK Operation (4.1.4)** | Messaging Manager complies. |
| **generic_nack Syntax (4.1.4.1)** | Messaging Manager complies. |
Message Submission Operations (4.2)

Messaging Manager complies.

- Messaging Manager does not check for exceeding the maximum length of the variable length fields.
- The SMPP submit_sm message is stored as a GenericSM object. The GenericSM class is sub-classed from the GenericMessage parent class with message type set to Submit.
- The mapping from SMPP to GenericSM is described for each parameter as follows.
  - `service_type` is set to null for outgoing messages. For incoming messages `service_type` is used to set the teleservice and allowAlternateDelivery via the teleserviceRoutingMap.
  - `source_addr_ton`, `source_addr_npi`, `source_addr` is stored as the GenericMessage::OriginatingAddress in GenericSM.
  - `dest_addr_ton`, `dest_addr_npi`, `dest_addr` is stored as the GenericMessage::DestinationAddress in GenericSM.
  - `esm_class` is not stored as one field in submit_sm, but individual bits are set/read from many fields. The esm_class is stored in multiple fields in the GenericMessage/GenericSM:
    - `0x3c` (Message Type: Bits 2-5) Determines how the message type is set. See MC Delivery Receipt (4.3.5.1) (on page 33), Intermediate Notification (4.3.5.2) (on page 33), SME Delivery Acknowledgement (4.3.5.3) (on page 34), SME Manual/User Acknowledgement (4.3.5.4) (on page 34) and Conversation Abort (4.3.5.5) (on page 34) for information on delivery receipt handling.
    - `0x40` (GSM Specific: UDHI Bit) Used to determine the presence of a userDataHeader (GenericSM::userDataHeaderPresent).
    - `0x03` (Messaging Mode: Bits 1-0) GenericSM::singleShot (0x01 => true, all others => false).
    - `0x80` (GSM Specific: Reply Path Bit) ProvideReplyPath (direct copy) GenericMessage::allowAlternateDelivery (false for non zero).
  - `protocol_id` is stored in GenericSM::protocolIdentifier.
  - `priority_flag` is stored in GenericMessage::priorityIndicator (0 => PriorityNormal, 1 => PriorityInteractive).

<table>
<thead>
<tr>
<th>SMPP esm_class bits</th>
<th>GenericSM fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x3c (Message Type: Bits 2-5)</td>
<td>Determines how the message type is set. See MC Delivery Receipt (4.3.5.1) (on page 33), Intermediate Notification (4.3.5.2) (on page 33), SME Delivery Acknowledgement (4.3.5.3) (on page 34), SME Manual/User Acknowledgement (4.3.5.4) (on page 34) and Conversation Abort (4.3.5.5) (on page 34) for information on delivery receipt handling.</td>
</tr>
<tr>
<td>0x40 (GSM Specific: UDHI Bit)</td>
<td>Used to determine the presence of a userDataHeader (GenericSM::userDataHeaderPresent).</td>
</tr>
<tr>
<td>0x03 (Messaging Mode: Bits 1-0)</td>
<td>GenericSM::singleShot (0x01 =&gt; true, all others =&gt; false).</td>
</tr>
<tr>
<td>0x80 (GSM Specific: Reply Path Bit)</td>
<td>ProvideReplyPath (direct copy) GenericMessage::allowAlternateDelivery (false for non zero).</td>
</tr>
</tbody>
</table>

Continued on next page
Message Submission Operations (4.2), Continued

submit_sm Syntax (4.2.1.1) (continued)

- `schedule_delivery_time` is not stored in GenericSM. Therefore only maintained if the message exits Messaging Manager via the incoming plugin and is not modified.
- `validity_period` qos_time_to_live is ignored by Messaging Manager when creating GenericSM. The validity period is converted to the GenericSM::ValidityPeriod class and stored in GenericSM::validityPeriod.
- `registered_delivery` is stored in GenericSM::statusReportRequest for non-deliver_sm messages. deliver_sm requests with registered_delivery are ignored and not stored in GenericSM::statusReportRequest. Messaging Manager outbound messages have registeredDelivery updated to reflect the statusReportRequest field of GenericSM.

<table>
<thead>
<tr>
<th>GenericSM::statusReportRequest</th>
<th>Effect on registeredDelivery</th>
</tr>
</thead>
<tbody>
<tr>
<td>xmsRequested, bothRequested</td>
<td>Set bit 0 to 1 and bit 1 to 0.</td>
</tr>
<tr>
<td>SmeRequested</td>
<td>Set to 1 if registeredDelivery is 0.</td>
</tr>
<tr>
<td>NotRequired</td>
<td>Set to 0.</td>
</tr>
</tbody>
</table>

- `replace_if_present_flag` is not stored in GenericSM.
- `data_coding` is generally stored in GenericSM as GenericSM::desiredAlphabet and also in several other fields for GSM data coding values. DataCodingElement structure is used as an intermediary in the mapping of data_coding to and from GenericSM parameters. The SMPP data_coding value is mapped and stored in one or more of the following parameters of the GenericSM:
  - `desiredAlphabet`
  - `messageClass` - only for GSM MC data_coding values.
  - `mwg` (MessageWaitingGroup) - only for GSM MWI data_coding values.
  - `mwi` (messageWaitingIndicator) - only for GSM MWI data_coding values.
  - `mwt` (MessageWaitingType) - only for GSM MWI data_coding values.

See `data_coding` (4.7.7) (on page 41) for additional details.

- `sm_default_msg_id` is not stored in GenericSM.
- `sm_length` is not stored in GenericSM. Instead the field is generated on outgoing messages from the userData.
- `short_message` The message_payload TLV is read by Messaging Manager with priority over the short_message field. If no message_payload is present, the short_message field is stored in GenericSM::userData. Outgoing messages have short_message set to the userData (with no message_payload TLV present) if less than 255 in size, otherwise it is set in the message_payload TLV.
- `message_submission TLVs` are considered in section Message Submission Request TLVs (4.2.4) (on page 28).

submit_sm resp Syntax (4.2.1.2)

Messaging Manager complies.

- Messaging Manager does not check for exceeding the maximum length of the variable length fields.
- Messaging Manager stores the message as a GenericSMResult.
- `message_id` is stored as GenericSMResult::deliverReceiptId.
- `message_submission response TLVs` are considered in section Message Submission Response TLVs (4.2.5) (on page 30).

Continued on next page
Message Submission Operations (4.2), Continued

**data_sm Syntax (4.2.2.1)**

Messaging Manager complies.

- Messaging Manager does not check for exceeding the maximum length of the variable length fields. The fields are used in the same way as submit_sm to construct an Messaging Manager GenericSM.
- The GenericMessage message type is set to MT_Submit if the message comes from an SME, or MT_Deliver if the message comes from an SMSC.

**data_sm_resp Syntax (4.2.2.2)**

Messaging Manager complies.

- Messaging Manager does not check for exceeding the maximum length of the variable length fields.
- The fields are used in the same way as submit_sm_resp to construct an Messaging Manager GenericSMResult.

**submit_multi Syntax (4.2.3.1)**

Messaging Manager complies.

- submit_multi messages are processed internally in Messaging Manager by creating a GenericSM for each terminating address. The handling of each field is thus the same as for submit_sm, with the exception of destinationAddress.
- The GenericMessage message type is set to Submit.
- Distribution Lists are recognised but not supported. ESME_RCNTSUBDL, "Cannot Submit to Distribution List", is returned.
- Although individual GenericSM components may be modified by Messaging Manager, these changes are not incorporated into the submit_multi forwarded to the SMSC, as any choice would be arbitrary. The forwarded submit_multi is derived from the originating message.

**submit_multi_res Syntax (4.2.3.2)**

Messaging Manager complies.

Messaging Manager does not check for exceeding the maximum length of the variable length fields. The fields are used in the same way as submit_sm_resps to construct a Messaging Manager GenericSMResult. The GenericSMResult has no knowledge of the unsuccess_sme structure, as it deals with only a single message. The SMPP plugin does, however, create the unsuccess_sme from the individual submit_sm_resps.

**Message Submission Request TLVs (4.2.4)**

Messaging Manager does not comply.

The following table sets out the way Messaging Manager manages each of the TLVs listed in Table 4-20 of The Specification. A TLV stated as being ignored by Messaging Manager can still be passed on if the message is unchanged and the message uses the same outgoing and incoming plugin.

<table>
<thead>
<tr>
<th>TLV Name</th>
<th>Messaging Manager treatment</th>
</tr>
</thead>
<tbody>
<tr>
<td>alert_on_msg_delivery</td>
<td>Ignored</td>
</tr>
<tr>
<td>billing_identification</td>
<td>Ignored</td>
</tr>
<tr>
<td>callback_num</td>
<td>Ignored</td>
</tr>
<tr>
<td>callback_num_atag</td>
<td>Ignored</td>
</tr>
<tr>
<td>callback_num_pres_ind</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_addr_np_country</td>
<td>Ignored</td>
</tr>
</tbody>
</table>

Continued on next page
## Message Submission Operations (4.2), Continued

### Message Submission Request TLVs (4.2.4) (continued)

<table>
<thead>
<tr>
<th>TLV Name</th>
<th>Messaging Manager treatment</th>
</tr>
</thead>
<tbody>
<tr>
<td>dest_addr_np_information</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_addr_np_resolution</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_addr_subunit</td>
<td>Stored in GenericSM::messageClass. Present in outgoing messages if value is not GenericSM::MessageClassNone.</td>
</tr>
<tr>
<td>dest_bearer_type</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_network_id</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_network_type</td>
<td>Stored in GenericMessage::messageProtocol.</td>
</tr>
<tr>
<td>dest_node_id</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_subaddress</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest(telematics_id)</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_port</td>
<td>Ignored</td>
</tr>
<tr>
<td>display_time</td>
<td>Ignored</td>
</tr>
<tr>
<td>its_reply_type</td>
<td>Ignored</td>
</tr>
<tr>
<td>its_session_info</td>
<td>Ignored</td>
</tr>
<tr>
<td>language_indicator</td>
<td>Ignored</td>
</tr>
<tr>
<td>message_payload</td>
<td>Used to create the GenericSM::userData. Present on outgoing messages when userData &gt; 255 characters.</td>
</tr>
<tr>
<td>more_messages_to_send</td>
<td>Ignored</td>
</tr>
<tr>
<td>ms_msg_wait_facilities</td>
<td>Stored in GenericSM::mwt (MessageWaitingType) and GenericSM::mwi (MessageWaitingIndicator).</td>
</tr>
<tr>
<td>ms_validity</td>
<td>Store in GenericSM::mwg (MessageWaitingGroup). Only value 0 (Store Indefinitely) will be correctly saved. Other values will be treated as GenericSM::MessageWaitingGroupDiscard. Outgoing messages will contain values 0 (Store Indefinitely) or 3 (Display Only) only.</td>
</tr>
<tr>
<td>number_of_messages</td>
<td>Ignored</td>
</tr>
<tr>
<td>payload_type</td>
<td>Ignored</td>
</tr>
<tr>
<td>privacy_indicator</td>
<td>Ignored</td>
</tr>
<tr>
<td>qos_time_to_live</td>
<td>Ignored</td>
</tr>
<tr>
<td>sar_msg_ref_num</td>
<td>Stored in GenericSM::segmentReference. Outgoing messages will either have this reference in the userDataHeader or this TLV (depending on if the message was modified by MMX).</td>
</tr>
<tr>
<td>sar_segment_seqnum</td>
<td>Stored in GenericSM::segmentNumber. Outgoing messages will either have this reference in the userDataHeader or this TLV (depending on if the message was modified by Messaging Manager).</td>
</tr>
</tbody>
</table>

*Continued on next page*
Message Submission Operations (4.2), Continued

Message Submission Request TLVs (4.2.4) (continued)

<table>
<thead>
<tr>
<th>TLV Name</th>
<th>Messaging Manager treatment</th>
</tr>
</thead>
<tbody>
<tr>
<td>sar_total_segments</td>
<td>Stored in GenericSM::segmentCount. Outgoing messages will either have this reference in the userDataHeader or this TLV (depending on if the message was modified by Messaging Manager).</td>
</tr>
<tr>
<td>set_dpf</td>
<td>Ignored</td>
</tr>
<tr>
<td>sms_signal</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_addr_subunit</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_bearer_type</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_network_id</td>
<td>Used to set the GenericSM::sourceLocationInformation, which is triggered to ACS as the location number.</td>
</tr>
<tr>
<td>source_network_type</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_node_id</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_port</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_subaddress</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_telematics_id</td>
<td>Ignored</td>
</tr>
<tr>
<td>user_message_reference</td>
<td>Stored in GenericSM::messageReference. Present in outgoing messages if value greater than zero.</td>
</tr>
<tr>
<td>user_response_code</td>
<td>Ignored</td>
</tr>
<tr>
<td>ussd_service_op</td>
<td>Ignored</td>
</tr>
</tbody>
</table>

Message Submission Response TLVs (4.2.5)

A TLV stated as being ignored by Messaging Manager can still be passed on if the message is unchanged and the message uses the same outgoing and incoming plugin.

The following table sets out the way Messaging Manager manages each of the TLVs listed in Table 4-21 of The Specification.

<table>
<thead>
<tr>
<th>TLV Name</th>
<th>Messaging Manager treatment</th>
</tr>
</thead>
<tbody>
<tr>
<td>additional_status_info_text</td>
<td>Ignored</td>
</tr>
<tr>
<td>delivery_failure_reason</td>
<td>Ignored</td>
</tr>
<tr>
<td>dpf_result</td>
<td>Ignored</td>
</tr>
<tr>
<td>network_error_code</td>
<td>Ignored</td>
</tr>
</tbody>
</table>

Source and Destination Addressing (4.2.6)

Messaging Manager does not comply.

- Messaging Manager does not consider that the source_addr may be NULL. The originating address is populated by the source fields, regardless of their values.
- Messaging Manager complies partially in that it understands the TON, NPI and address fields of a mobile number.

International and National Format (4.2.6.1.1)

Messaging Manager complies.

Continued on next page
Message Submission Operations (4.2), Continued

<table>
<thead>
<tr>
<th>Alphanumeric Format (4.2.6.1.2)</th>
<th>Messaging Manager does not comply.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• The AMC part of Messaging Manager does not comply because ACS is triggered by BcdDigits which cannot handle alphabetical characters.</td>
</tr>
<tr>
<td></td>
<td>• Messaging Manager complies if a message does not trigger a call-plan.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NPI (4.2.6.2)</th>
<th>Messaging Manager complies.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Messaging Manager also handles an NPI of 13 to represent PC:SSN. This value should only be used for SCCP level addresses.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ESME Addresses (4.2.6.3)</th>
<th>Compliance statements are made under the following headings:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Service Short Code</td>
</tr>
<tr>
<td></td>
<td>• International Number, and</td>
</tr>
<tr>
<td></td>
<td>• NULL Address.</td>
</tr>
<tr>
<td></td>
<td>These headings correspond to the bullet list in clause 4.2.6.3 of The Specification.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service Short Code</th>
<th>Messaging Manager complies.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>International Number</th>
<th>Messaging Manager complies.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>NULL Address</th>
<th>Messaging Manager does not comply.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Messaging Manager cannot substitute a default source address into the GenericSM. A non-NULL address is required for a delivery receipt to be sent. This could be implemented in the future with a simple change to the originating plugin.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message Replace operation in submit_sm (4.2.7)</th>
<th>Messaging Manager does not comply.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The replace_if_present flag is ignored by Messaging Manager, and not placed in outgoing messages (unless the message is unaltered and goes out the incoming plugin). The service_type field is also not preserved by Messaging Manager. replace_sm messages are forwarded on, so this is the only way to send a message replace through Messaging Manager.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message Length (4.2.8)</th>
<th>Messaging Manager complies.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Messaging Manager can handle up to 255 characters in short_message. Messages that are too long will be placed in the message_payload TLV. Messaging Manager does not consider the possibility of the MC only having space for 140 octets (that is, the 255 limit is hard-coded).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Registered (4.2.9.1)</th>
<th>Messaging Manager does not comply.</th>
</tr>
</thead>
<tbody>
<tr>
<td>See submit_sm Syntax (4.2.1.1) (on page 26).</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scheduled (4.2.9.2)</th>
<th>Messaging Manager does not comply.</th>
</tr>
</thead>
<tbody>
<tr>
<td>The scheduled_delivery_value is ignored in constructing a GenericSM. However, Messaging Manager is capable of detecting the presence of the scheduled_delivery_value and such messages will be FDA-barred internally so that these messages, where appropriate, will be sent to an alternative Message Centre for proper handling at the scheduled delivery time.</td>
<td></td>
</tr>
</tbody>
</table>

Continued on next page
### Message Submission Operations (4.2), Continued

#### Pre-defined (4.2.9.3)

<table>
<thead>
<tr>
<th>Description</th>
<th>Compliance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messing Manager does not comply. The sm_default_msg_id is not stored in the GenericSM class. Messing Manager does not examine this value on incoming messages either.</td>
<td></td>
</tr>
</tbody>
</table>

#### Message Modes (4.2.10)

<table>
<thead>
<tr>
<th>Description</th>
<th>Compliance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messing Manager does not comply. The esm_class value is not directly stored in GenericSM. See the following four compliance statements for more detail.</td>
<td></td>
</tr>
</tbody>
</table>

#### Default Message Mode (4.2.10.1)

<table>
<thead>
<tr>
<th>Description</th>
<th>Compliance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messing Manager complies. If the incoming message has bits 0 and 1 set to zero, so will the outgoing message.</td>
<td></td>
</tr>
</tbody>
</table>

#### Store and Forward Message Mode (4.2.10.2)

<table>
<thead>
<tr>
<th>Description</th>
<th>Compliance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messing Manager does not comply. This part of the esm_class is reconstructed as 00 (default message mode) if the outgoing message is changed and is not singleShot.</td>
<td></td>
</tr>
</tbody>
</table>

#### Datagram Message Mode (4.2.10.3)

<table>
<thead>
<tr>
<th>Description</th>
<th>Compliance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messing Manager complies. The singleShot variable in GenericSM correctly captures this behaviour. Note that the delivery receipt may still be requested via the registered_delivery field.</td>
<td></td>
</tr>
</tbody>
</table>

#### Transaction Message Mode (4.2.10.4)

<table>
<thead>
<tr>
<th>Description</th>
<th>Compliance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messing Manager does not comply. Messing Manager does not set bit 1 to 1 for an altered message. Note that an SMPP Transaction mode message which came into Messing Manager has allowAlternateDelivery set to false, so we will go out the same plugin, and the outgoing message will be Transaction mode, provided the singleShot nature was not changed by Messing Manager.</td>
<td></td>
</tr>
</tbody>
</table>
# Message Delivery Operations (4.3)

**deliver_sm Syntax (4.3.1.1)**

- Messaging Manager complies.
  - A deliver_sm is handled similarly to a submit_sm. See *submit_sm Syntax (4.2.1.1)* on page 26 for detailed handling of each message tag.
  - genericMessage's message type is set to Deliver unless the deliver_sm contains a delivery receipt. See *MC Delivery Receipt (4.3.5.1)* on this page, *Intermediate Notification (4.3.5.2)* on this page, *SME Delivery Acknowledgement (4.3.5.3)* on page 34, *SME Manual/User Acknowledgement (4.3.5.4)* on page 34 and *Conversation Abort (4.3.5.5)* on page 34.

**deliver_sm_resp Syntax (4.3.1.2)**

- Messaging Manager complies.
  - A deliver_sm is handled similarly to a submit_sm. The main difference is that the GenericMessage::messageType is changed to MT_Notify for a Status Report.
  - See *submit_sm_resp Syntax (4.2.1.2)* on page 27 for a detailed description of the handling of each message tag.

**data_sm Operation (4.3.2)**

See *data_sm Syntax (4.2.2.1)* on page 28 and *data_sm_resp Syntax (4.2.2.2)* on page 28 for more information.

**Message Delivery Request TLVs (4.3.3)**

TLVs not covered in *Message Submission Request TLVs (4.2.4)* on page 28 and *Message Submission Response TLVs (4.2.5)* on page 30 are listed in the following table.

<table>
<thead>
<tr>
<th>TLV Name</th>
<th>Messaging Manager treatment</th>
</tr>
</thead>
<tbody>
<tr>
<td>message_state</td>
<td>Stored in GenericSM::deliverySucceeded (as equal to DELIVERED or not). This field is set to false if the TLV is not present.</td>
</tr>
<tr>
<td>receipted_message_id</td>
<td>Stored in GenericSM::deliveryReceiptId. This field is set to blank if the TLV is not present.</td>
</tr>
</tbody>
</table>

**Message Delivery Response TLVs (4.3.4)**

For compliance statements see *Message Submission Response TLVs (4.2.5)* on page 30.

**MC Delivery Receipt (4.3.5.1)**

- Messaging Manager complies.
  - message_state and receipted_message_id are observed, but the network_error_id field is ignored by Messaging Manager.
  - The GenericMessage's message type is set to Notify and message contents set to Delivery Receipt.

**Intermediate Notification (4.3.5.2)**

- Messaging Manager complies.
  - If the MC passes Messaging Manager one of these messages, it will pass it on (setting allowAlternateDelivery to false). Of the fields listed as important, only network_error_id is ignored by Messaging Manager.
  - The GenericMessage's message type is set to Notify and message contents set to Delivery Receipt.

*Continued on next page*
## Message Delivery Operations (4.3), Continued

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
</table>
| SME Delivery Acknowledgment (4.3.5.3) | Messaging Manager complies.  
- If the MC passes Messaging Manager one of these messages, it will pass it on (setting allowAlternateDelivery to false).  
- The GenericMessage's message type is set to Notify and message contents set to Delivery Receipt. |

| SME Manual/User Acknowledgment (4.3.5.4) | Messaging Manager complies.  
If the MC passes Messaging Manager one of these messages, it will pass it on (setting allowAlternateDelivery to false).  
The GenericMessage's message type is set to Notify and message contents set to Delivery Receipt. |

| Conversation Abort (4.3.5.5) | Messaging Manager complies.  
If the MC passes Messaging Manager one of these messages, Messaging Manager will pass it on. |
## Message Broadcast Operations (4.4)

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>broadcast_sm Operation (4.4.1)</td>
<td>Messaging Manager does not comply. Messaging Manager responds to broadcast_sm messages with a genericNack.</td>
</tr>
<tr>
<td>broadcast_sm Syntax (4.4.1.1)</td>
<td>Messaging Manager does not comply.</td>
</tr>
<tr>
<td></td>
<td>• Messaging Manager does not attempt to construct a GenericMessage.</td>
</tr>
<tr>
<td></td>
<td>• Messaging Manager does not attempt to handle individual fields or interpreted them.</td>
</tr>
<tr>
<td></td>
<td>• Messaging Manager can construct a fully compliant internal representation of a broadcast_sm, but it cannot translate this object to GenericMessage.</td>
</tr>
<tr>
<td>broadcast_sm_resp Syntax (4.4.1.2)</td>
<td>Messaging Manager does not comply. See broadcast_sm Operation (4.4.1), above.</td>
</tr>
<tr>
<td>Broadcast Request Optional TLVs (4.4.2)</td>
<td>Messaging Manager does not comply. See broadcast_sm Operation (4.4.1), above.</td>
</tr>
<tr>
<td>Broadcast Response Optional TLVs (4.4.3)</td>
<td>Messaging Manager does not comply. See broadcast_sm Operation (4.4.1), above.</td>
</tr>
<tr>
<td>Message Replacement with broadcast_sm (4.4.4)</td>
<td>Messaging Manager does not comply. See broadcast_sm Operation (4.4.1), above.</td>
</tr>
</tbody>
</table>
## Ancillary Submission Operations (4.5)

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
</table>
| **cancel_sm** (4.5.1)      | Messaging Manager complies.  
  **cancel_sm** operations are sent straight to the outgoing protocolHandler, never entering xmsTrigger or ACS. Thus, the message passed on is an exact copy of the incoming message. |
| **Syntax (4.5.1.1)**      | Messaging Manager complies.  
  The incoming message is copied to the outgoing message, so Messaging Manager is compliant, assuming that the message originator is compliant. |
| **cancel_sm_resp** (4.5.1.2) | Messaging Manager complies.  
  See **cancel_sm Syntax (4.5.1.1)** on page 36. |
| **query_sm** (4.5.2)       | Messaging Manager complies.  
  **query_sm** operations are sent straight to the outgoing protocolHandler, never entering xmsTrigger or ACS. Thus, the message passed on is an exact copy of the incoming message. |
| **Syntax (4.5.2.1)**      | Messaging Manager complies.  
  The incoming message is copied to the outgoing message, so Messaging Manager is compliant, assuming that the message originator is compliant. |
| **query_sm_resp** (4.5.2.2) | Messaging Manager complies.  
  See **query_sm Syntax (4.5.2.1)** on page 36. |
| **replace_sm** (4.5.3)     | Messaging Manager complies.  
  **replace_sm** operations are sent straight to the outgoing protocolHandler, never entering xmsTrigger or ACS. Thus, the message passed on is an exact copy of the incoming message. |
| **Syntax (4.5.3.1)**      | Messaging Manager complies.  
  The incoming message is copied to the outgoing message, so Messaging Manager is compliant, assuming that the message originator is compliant. |
| **replace_sm_resp** (4.5.3.2) | Messaging Manager complies.  
  See **replace_sm Syntax (4.5.3.1)** on page 36. |
| **Message Replacement TLVs** (4.5.3.3) | Messaging Manager complies.  
  See **replace_sm Syntax (4.5.3.1)** on page 36. |
### Ancillary Broadcast Operations (4.6)

#### Query Broadcast sm Operation (4.6.1)
- Messaging Manager does not comply.
- Messaging Manager responds to `query_broadcast_sm` messages with a genericNack.

#### Query Broadcast sm Syntax (4.6.1.1)
- Messaging Manager does not comply.
  - Messaging Manager does not attempt to construct a GenericMessage. Messaging Manager does not attempt to handle individual fields nor does it interpret them.
  - Messaging Manager can construct a fully compliant internal representation of a `query_broadcast_sm`, but it cannot translate the representation to a GenericMessage.

#### Query Broadcast Request Optional TLVs (4.6.1.2)
- Messaging Manager does not comply.
  - See `query_broadcast_sm Operation (4.6.1)` in this topic.

#### Query Broadcast sm resp Syntax (4.6.1.3)
- Messaging Manager does not comply.
  - See `query_broadcast_sm Operation (4.6.1)` in this topic.

#### Query Broadcast Response Optional TLVs (4.6.1.4)
- Messaging Manager does not comply.
  - See `query_broadcast_sm Operation (4.6.1)` in this topic.

#### Cancel Broadcast sm Operation (4.6.2)
- Messaging Manager does not comply.
  - Messaging Manager does not attempt to construct a GenericMessage. Messaging Manager does not attempt to handle individual fields nor does it interpret them.
  - Messaging Manager can construct a fully compliant internal representation of a `cancel_broadcast_sm`, but it cannot translate the representation to a GenericMessage.

#### Cancel broadcast sm Syntax (4.6.2.1)
- Messaging Manager does not comply.
  - See `cancel_broadcast_sm Operation (4.6.2)` in this topic.

#### Cancel Broadcast Optional TLVs (4.6.2.2)
- Messaging Manager does not comply.
  - See `cancel_broadcast_sm Operation (4.6.2)` in this topic.

#### cancel broadcast sm resp Syntax (4.6.2.3)
- Messaging Manager does not comply.
  - See `cancel_broadcast_sm Operation (4.6.2)` in this topic.
PDU Field Definitions (4.7)

**addr_ton**, **source_addr_ton**, **dest_addr_ton**, **esme_addr_ton**

(4.7.1)

Messaging Manager complies.

**addr_npi**, **source_addr_npi**, **dest_addr_npi**, **esme_addr_npi**

(4.7.2)

Messaging Manager does not comply.

The values for Internet (IP) and WAP Client Id are not considered. Messaging Manager also has an extra value of 13 for point codes.

address_range

(4.7.3)

Messaging Manager does not comply.

address_range is always NULL in messages constructed by Messaging Manager. The value of the field is ignored in interpreting messages received by Messaging Manager.

UNIX Regular Expressions

(4.7.3.1)

Not relevant as the address_range is never used by Messaging Manager. See address_range (4.7.3) (on page 38).

**command_length**

(4.7.4)

Messaging Manager complies.

**command_id**

(4.7.5)

Messaging Manager complies.

**command_status**, **error_status_code**

(4.7.6)

Messaging Manager complies.

- All values are correct.
- Errors not explicitly mentioned as transientFailures are treated as permanentFailures.
- If a message is throttled by Messaging Manager, a status code determined by the throttledCommandStatus configuration parameter will be returned. This defaults to ESME_RTHROTTLED.
- "Not used" implies that incoming messages are not checked for the associated error.
- For the Reject action, Messaging Manager can be configured to return any SMPP error_code. To do this Messaging Manager uses a configured mapping from ACS CS1ReleaseCause. If configured by the user, all cause codes are treated as "Compliant in outbound direction".

Default "Not Used" cause code values are shown in the following table.

<table>
<thead>
<tr>
<th>Command status name</th>
<th>Usage compliance</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESME_ROK</td>
<td>Messaging Manager complies. Treated as GenericSMResult::resultSuccess.</td>
</tr>
<tr>
<td>ESME_RINVMSGLEN</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>ESME_RINVCMDLEN</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>ESME_RINVCMDID</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>ESME_RINVBNSTS</td>
<td>Messaging Manager complies. Receipt is treated as a GenericSMResult::resultTransientFailure.</td>
</tr>
</tbody>
</table>

Continued on next page
## Command status, error status code (4.7.6) (continued)

<table>
<thead>
<tr>
<th>Command status name</th>
<th>Usage compliance</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESME_RALYBND</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>ESME_RINVPRTFLG</td>
<td>Not used.</td>
</tr>
<tr>
<td>ESME_RINVREGDLVFLG</td>
<td>Not used.</td>
</tr>
<tr>
<td>ESME_RSYSERR</td>
<td>Messaging Manager complies. • In the message outbound direction, receipt is treated as a GenericSMResult::resultTransient-Failure. • In the message inbound direction, an abort result type (GenericSMResult::resultAbort) will cause this code to be sent.</td>
</tr>
<tr>
<td>ESME_RINVSRCADR</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVDSTADR</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVMGSID</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RBINDFAIL</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>ESME_RINVPASWD</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVSYSID</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RCCANCELFAIL</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>ESME_RREPLACEFAIL</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>ESME_RMSGQFUL</td>
<td>Messaging Manager complies. • Messaging Manager is configured with maxConcurrentTransactions. When this is exceeded, MSGQFUL is replied to the sender. • Receipt of a MSGQFUL is treated as an GenericSMResult::resultTransient-Failure</td>
</tr>
<tr>
<td>ESME_RINVSERTYP</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVNUMDESTS</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVDLNAME</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVDSTFLAG</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVSUBREP</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVESMCLASS</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RCNTSUBDL</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>ESME_RSUBMITFAIL</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVSRCCTON</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVSRCNPI</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVDSTTON</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVDSTNPI</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVSYSTYP</td>
<td>Not used</td>
</tr>
</tbody>
</table>

Continued on next page
### PDU Field Definitions (4.7), Continued

**command_status, error_status_code (4.7.6) (continued)**

<table>
<thead>
<tr>
<th>Command status name</th>
<th>Usage compliance</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESME_RINVREPFLAG</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVNUMMSGS</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RTHROTTLED</td>
<td>Messaging Manager complies. The throttling response code can be changed via the eserv.config default smpp parameter 'throttledCommandStatus'.</td>
</tr>
<tr>
<td>ESME_RINVSCHE D</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVEXPIRY</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVDFTMSGID</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RX_T_APPN</td>
<td>Treated as a GenericSMResult::resultTransient-Failure. TransientFailures are mapped to this value. Also used for duplicate sequence numbers and failure to send a message to transaction or to construct a transaction object.</td>
</tr>
<tr>
<td>ESME_RX_P_APPN</td>
<td>PermanentFailures are mapped to this.</td>
</tr>
<tr>
<td>ESME_RX_R_APPN</td>
<td>Treated as a GenericSMResult::resultTransient-Failure.</td>
</tr>
<tr>
<td>ESME_RQUERYFAIL</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>ESME_RINVTLVSTREAM</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RTLVNOTALLWD</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVTLVLEN</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RMISSENGTLV</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>ESME_RINVTLVVAL</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RDELIVERYFAILURE</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RUNKOWNERR</td>
<td>Messaging Manager complies. Receipt is treated as a GenericSMResult::resultTransient-Failure.</td>
</tr>
<tr>
<td>ESME_RSERTYPUNAUTH</td>
<td>Receipt is treated as a GenericSMResult::resultTransient-Failure.</td>
</tr>
<tr>
<td>ESME_RPROHIBITED</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RSERTYPUNAVAIL</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RSERTYPDENIED</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVDCS</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVSRCDADDERSUBUNIT</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVDSTADDRSUBUNIT</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVBCASTFREQINT</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVBCASTALIAS_NAME</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVBCASTAREAFMT</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVNUMBCAST AREAS</td>
<td>Not used</td>
</tr>
</tbody>
</table>

*Continued on next page*
PDU Field Definitions (4.7), Continued

**command_status, error_status_code (4.7.6)** (continued)

<table>
<thead>
<tr>
<th>Command status name</th>
<th>Usage compliance</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESME_RINVBCASTCNTTYPE</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVBCASTMSGCLASS</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RBCASTFAIL</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RBCASTQUERYFAIL</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RBCASTCANCELFAIL</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVBCAST_REP</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVBCASTSRVGRP</td>
<td>Not used</td>
</tr>
<tr>
<td>ESME_RINVBCASTCHANIND</td>
<td>Not used</td>
</tr>
</tbody>
</table>

**data_coding (4.7.7)**

Messaging Manager complies.  
See submit_sm Syntax (4.2.1.1) (on page 26).

- Generally, data_coding is stored in GenericSM::desiredAlphabet, but depending on its value (for GSM MWI and GSM MC values), data_coding may also be stored in:
  - GenericSM::mwi (message waiting indicator)
  - GenericSM::mwg (message waiting group)
  - GenericSM::mwt (message waiting type)
  - GenericSM::messageClass
- Except for GenericSM::desiredAlphabet, the presence of SMPP's optional parameters such as
  - TLV ms_validity,
  - TLV ms_msg_wait_facilities, and
  - TLV dest_addr_subunit
  will override the GenericSM mwi, mwg, mwt and/or messageClass parameters described above.
- In Messaging Manager, data_coding is mapped to and from a dataCodingElement structure. Messaging Manager uses inboundDataCodingMap and outboundDataCodingMap of the SMPP Plugin. The dataCodingElement:
  - is used to populate the GenericSM parameters described above for the inbound case, and
  - is populated from the GenericSM parameters described above for the outbound case.

**destination_addr (4.7.8)**

Messaging Manager complies.

**dest_flag (4.7.9)**

Messaging Manager complies.

Messaging Manager does not support distribution lists themselves.

**dl_name (4.7.10)**

Messaging Manager does not comply.

**esme_addr (4.7.11)**

Messaging Manager complies.

Continued on next page
### PDU Field Definitions (4.7), Continued

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>esm_class</td>
<td>Messaging Manager does not comply.</td>
</tr>
<tr>
<td><strong>(4.7.12)</strong></td>
<td>• Set Reply Path Bit (Bit 7) is stored in GenericSM::provideReplyPath.</td>
</tr>
<tr>
<td></td>
<td>• UDHI Bit (Bit 6) is recognised but not stored in GenericSM. It is used to stop alternate delivery of concatenated messages when no UDHI is present. Messaging Manager will not use esm_class to carry segmentation information if it is carried in TLVs.</td>
</tr>
<tr>
<td></td>
<td>• Bit 4 (Conversation Abort and manual/user ack) is ignored by Messaging Manager.</td>
</tr>
<tr>
<td></td>
<td>• See submit_sm Syntax (4.2.1.1) (on page 26).</td>
</tr>
<tr>
<td>interface_version</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td><strong>(4.7.13)</strong></td>
<td>MMC complies.</td>
</tr>
<tr>
<td>message_id</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td><strong>(4.7.14)</strong></td>
<td>Query messages are not interpreted by Messaging Manager, simply passed on, so Messaging Manager does not react to or alter this field.</td>
</tr>
<tr>
<td>message_state</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td><strong>(4.7.15)</strong></td>
<td></td>
</tr>
<tr>
<td>no_unsuccess</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td><strong>(4.7.16)</strong></td>
<td></td>
</tr>
<tr>
<td>number_of_dests</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td><strong>(4.7.17)</strong></td>
<td></td>
</tr>
<tr>
<td>password</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td><strong>(4.7.18)</strong></td>
<td></td>
</tr>
<tr>
<td>priority_flag</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td><strong>(4.7.19)</strong></td>
<td>• Stored in GenericMessage::priorityIndicator.</td>
</tr>
<tr>
<td></td>
<td>• Messaging Manager uses the IS-95/ANSI-41 compliant priority mapping. The mapping from these values to ANSI-136 is described below (converting from left to right).</td>
</tr>
<tr>
<td><strong>ANSI-136</strong></td>
<td><strong>ANSI-41</strong></td>
</tr>
<tr>
<td>Bulk</td>
<td>Normal</td>
</tr>
<tr>
<td>Normal</td>
<td>Interactive</td>
</tr>
<tr>
<td>Urgent</td>
<td>Urgent</td>
</tr>
<tr>
<td>Very urgent</td>
<td>Emergency</td>
</tr>
</tbody>
</table>

| protocol_id      | Messaging Manager complies.                                                 |
| **(4.7.20)**     | Value is stored in GenericSM::protocolIdentifier.                           |
| registered_delivery| Messaging Manager does not comply.                                          |
| **(4.7.21)**     | Bits 0 and 1 are altered. All other bits are left alone. Outgoing messages have the same pattern. Messaging Manager does not compliantly set bit 1. See submit_sm Syntax (4.2.1.1) (on page 26). |

Continued on next page
PDU Field Definitions (4.7), Continued

<table>
<thead>
<tr>
<th>Field Definition</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>replace_if_present_flag (4.7.22)</code></td>
<td>Messaging Manager does not comply. Value is not stored in GenericSM, so is essentially ignored by Messaging Manager, unless the message is copied to the outgoing plugin unaltered.</td>
</tr>
<tr>
<td><code>scheduled_delivery_time (4.7.23.1)</code></td>
<td>Messaging Manager does not comply. scheduled_delivery_time is not stored in GenericSM and is ignored by Messaging Manager, unless scheduled_delivery_time is copied to the outgoing plugin unaltered. However, for cases where FDA may be relevant for the message, Messaging Manager recognises the presence of scheduled_delivery_time, bypasses FDA and passes scheduled_delivery_time to the SMSC for proper handling at the correct scheduled delivery time.</td>
</tr>
<tr>
<td><code>validity_period (4.7.23.2)</code></td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td><code>final_date (4.7.23.3)</code></td>
<td>Messaging Manager complies. Only used for queries not deciphered by Messaging Manager.</td>
</tr>
<tr>
<td>Absolute Time Format (4.7.23.4)</td>
<td>Messaging Manager does not comply. • Tens of seconds are ignored by Messaging Manager. • Messaging Manager stores absolute time internally as seconds since midnight UTC on 1 January 1970.</td>
</tr>
<tr>
<td>Relative Time Format (4.7.23.5)</td>
<td>Messaging Manager complies. Messaging Manager stores relative time internally as a number of seconds to offset.</td>
</tr>
<tr>
<td><code>sequence_number (4.7.24)</code></td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td><code>service_type (4.7.25)</code></td>
<td>Messaging Manager does not comply. • For outgoing messages that are altered or generated, service_type is set to null. • For incoming messages: ▪ service_type is used to set the teleservice and allowAlternateDelivery via the teleserviceRoutingMap. ▪ Implicit association of a function from a service type such as &quot;replace if present&quot; is not supported.</td>
</tr>
<tr>
<td><code>short_message (4.7.26)</code></td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td><code>sm_default_msg_id (4.7.27)</code></td>
<td>Messaging Manager does not comply. sm_default_msg_id is not stored in GenericSM. Messaging Manager ignores sm_default_msg_id unless it is copied, unmodified, to the outgoing plugin.</td>
</tr>
<tr>
<td><code>sm_length (4.7.28)</code></td>
<td>Messaging Manager complies. This value is not stored internally in Messaging Manager, but calculated from the current message length. The value is correctly set to 0 if a message_payload TLV is being used.</td>
</tr>
</tbody>
</table>

Continued on next page
## PDU Field Definitions (4.7), Continued

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| **source_addr** (4.7.29) | Messaging Manager complies.  
  Stored in the originatingAddress field of GenericMessage. A value of NULL is not supported by Messaging Manager. |
| **system_id** (4.7.30)   | Messaging Manager complies.                                                 |
| **system_type** (4.7.31) | Messaging Manager complies.  
  Set from the configuration option "systemType". |
PDU TLV Definitions (4.8)

### Position of TLVs in SMPP messages (4.8)
- Messaging Manager complies.

### TLV Tag (4.8.1)
- Messaging Manager uses some TLV tag values internally, mainly for holding values from the EmiProtocolHandler. These TLV values are listed in the following table.

<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Tag Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmsc_address</td>
<td>0x3680</td>
</tr>
<tr>
<td>num_septets</td>
<td>0x3681</td>
</tr>
<tr>
<td>tdma_priority</td>
<td>0x3682</td>
</tr>
<tr>
<td>message_modified</td>
<td>0x3683</td>
</tr>
</tbody>
</table>

### TLV Length (4.8.2)
- Messaging Manager complies.

### TLV Value (4.8.3)
- Messaging Manager complies.

### TLV Definitions (4.8.4)
- Messaging Manager does not comply.

Where a TLV definition is stated as "ignored" by Messaging Manager, the TLV will only be compliant if the message exits via the originating plugin. In this case unaltered TLVs will be preserved into the outgoing message.

- Ignored
- Ignored
- Ignored
- Ignored
- Ignored
- Ignored
- Ignored
- Ignored
- Ignored

**Continued on next page**
# PDU TLV Definitions (4.8), Continued

<table>
<thead>
<tr>
<th>Field</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>broadcast_channel_indicator</td>
<td>Ignored</td>
</tr>
<tr>
<td>broadcast_content_type</td>
<td>Ignored</td>
</tr>
<tr>
<td>broadcast_end_time</td>
<td>Ignored</td>
</tr>
<tr>
<td>broadcast_error_status</td>
<td>Ignored</td>
</tr>
<tr>
<td>broadcast_frequency_interval</td>
<td>Ignored</td>
</tr>
<tr>
<td>broadcast_message_class</td>
<td>Ignored</td>
</tr>
<tr>
<td>broadcast_rep_num</td>
<td>Ignored</td>
</tr>
<tr>
<td>broadcast_service_group</td>
<td>Ignored</td>
</tr>
<tr>
<td>callback_num</td>
<td>Ignored</td>
</tr>
<tr>
<td>callback_num_atag</td>
<td>Ignored</td>
</tr>
<tr>
<td>callback_num_pres_ind</td>
<td>Ignored</td>
</tr>
<tr>
<td>congestion_state</td>
<td>Ignored</td>
</tr>
<tr>
<td>delivery_failure_reason</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_addr_np_country</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_addr_np_information</td>
<td>Ignored</td>
</tr>
</tbody>
</table>

Continued on next page
**PDU TLV Definitions (4.8), Continued**

<table>
<thead>
<tr>
<th>TLV Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dest_addr_np_resolution (4.8.4.22)</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_addr_subunit (4.8.4.23)</td>
<td>Messaging Manager complies. Stored in GenericSM::messageClass.</td>
</tr>
<tr>
<td>dest_bearer_type (4.8.4.24)</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_network_id (4.8.4.25)</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_network_type (4.8.4.26)</td>
<td>Messaging Manager does not comply. Stored in GenericMessage::messageProtocol. Only the following values will be stored:</td>
</tr>
<tr>
<td></td>
<td>• 0x02 - ANSI-136/TDMA</td>
</tr>
<tr>
<td></td>
<td>• 0x03 - IS-95/CDMA</td>
</tr>
<tr>
<td></td>
<td>• Other values treated as GenericSM::UNKNOWN message protocol.</td>
</tr>
<tr>
<td>dest_node_id (4.8.4.27)</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_subaddress (4.8.4.28)</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_telematics_id (4.8.4.29)</td>
<td>Ignored</td>
</tr>
<tr>
<td>dest_port (4.8.4.30)</td>
<td>Ignored</td>
</tr>
<tr>
<td>display_time (4.8.4.31)</td>
<td>Ignored</td>
</tr>
<tr>
<td>dpf_result (4.8.4.32)</td>
<td>Ignored</td>
</tr>
<tr>
<td>its_reply_type (4.8.4.33)</td>
<td>Ignored</td>
</tr>
<tr>
<td>its_session_info (4.8.4.34)</td>
<td>Ignored</td>
</tr>
<tr>
<td>language_indicator (4.8.4.35)</td>
<td>Ignored</td>
</tr>
<tr>
<td>message_payload (4.8.4.36)</td>
<td>Messaging Manager complies. Only used if message_size exceeds 255 characters.</td>
</tr>
</tbody>
</table>
### PDU TLV Definitions (4.8), Continued

<table>
<thead>
<tr>
<th>TLV Name</th>
<th>Compliance</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>message_state</strong> (4.8.4.37)</td>
<td>Messaging Manager does not comply.</td>
<td>Stored (as a bool) in GenericSM::deliverySucceeded. This field is set to true if the state is DELIVERED, and false for all other values or if the TLV is not present. Outgoing messages originating from other protocols will have a value of UNKNOWN. For delivery receipts, the message_state may be set to DELIVERED or UNDELIVERED depending on the value of GenericSM::deliverySucceeded.</td>
</tr>
<tr>
<td><strong>more_messages_to_send</strong> (4.8.4.38)</td>
<td>Ignored</td>
<td></td>
</tr>
<tr>
<td><strong>ms_availability_status</strong> (4.8.4.39)</td>
<td>Ignored</td>
<td></td>
</tr>
<tr>
<td><strong>ms_msg_wait_facilities</strong> (4.8.4.40)</td>
<td>Messaging Manager complies.</td>
<td>Stored in GenericSM::mwi (MessageWaitingIndicator) and GenericSM::mwt (MessageWaitingType).</td>
</tr>
</tbody>
</table>
| **ms_validity** (4.8.4.41)                    | Messaging Manager does not comply. | • Stored in GenericSM::mwg (messageWaitingGroup).  
• Messaging Manager complies for value 0 (Store Indefinitely).  
• Value 3 (Display Only) used only for outbound messages. |
| **network_error_code** (4.8.4.42)              | Ignored             |                                                                                                                                              |
| **number_of_messages** (4.8.4.43)              | Ignored             |                                                                                                                                              |
| **payload_type** (4.8.4.44)                    | Ignored             |                                                                                                                                              |
| **privacy_indicator** (4.8.4.45)               | Ignored             |                                                                                                                                              |
| **qos_time_to_live** (4.8.4.46)                | Ignored             |                                                                                                                                              |
| **receipted_message_id** (4.8.4.47)            | Messaging Manager complies. | Stored in GenericSM::deliveryReceiptId. Set to blank if not present.                                                                      |
| **sar_msg_ref_num** (4.8.4.48)                 | Messaging Manager complies. | Stored in GenericSM::segmentReference.                                                                                                       |
| **sar_segment_seqnum** (4.8.4.49)              | Messaging Manager complies. | Stored in GenericSM::segmentNumber.                                                                                                          |
| **sar_total_segments** (4.8.4.50)              | Messaging Manager complies. |                                                                                                                                              |
### PDU TLV Definitions (4.8), Continued

<table>
<thead>
<tr>
<th>TLV Description</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>sar_total_segments (4.8.4.50)</td>
<td>Stored in GenericSM::segmentCount.</td>
</tr>
<tr>
<td>sc_interface_version (4.8.4.51)</td>
<td>Ignored</td>
</tr>
<tr>
<td>set_dpf (4.8.4.52)</td>
<td>Ignored</td>
</tr>
<tr>
<td>sms_signal (4.8.4.53)</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_addr_subunit (4.8.4.54)</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_bearer_type (4.8.4.55)</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_network_id (4.8.4.56)</td>
<td>Messaging Manager complies.</td>
</tr>
<tr>
<td>source_network_type (4.8.4.57)</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_node_id (4.8.4.58)</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_port (4.8.4.59)</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_subaddress (4.8.4.60)</td>
<td>Ignored</td>
</tr>
<tr>
<td>source_telematics_id (4.8.4.61)</td>
<td>Ignored</td>
</tr>
<tr>
<td>user_message_reference (4.8.4.62)</td>
<td>Messaging Manager complies.  Stored in GenericSM::messageReference. Placed in outgoing messages if messageReference is greater than zero.</td>
</tr>
<tr>
<td>user_response_code (4.8.4.63)</td>
<td>Ignored</td>
</tr>
<tr>
<td>ussd_service_op (4.8.4.64)</td>
<td>Ignored.</td>
</tr>
</tbody>
</table>
# Glossary of Terms

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AAA</strong></td>
<td>Authentication, Authorisation, and Accounting. Specified in Diameter RFC 3588.</td>
</tr>
<tr>
<td><strong>ACS</strong></td>
<td>Advanced Control Services configuration platform.</td>
</tr>
</tbody>
</table>
| **AMC** | The Advanced Message Control component of Messaging Manager. It provides:  
  - Real time billing interaction  
  - Customized Service Control  
  - SMS "service plan" capability  
  - Enhanced service support  
  **Note:** Also known as "SMS Director". |
| **ASP** |  
  - Application Service Provider, or  
| **CC** | Country Code. Prefix identifying the country for a numeric international address. |
| **CDMA** | Code Division Multiple Access is a method for describing physical radio channels. Data intended for a specific channel is modulated with that channel's code. These are typically pseudo-random in nature, and possess favourable correlation properties to ensure physical channels are not confused with one another. |
| **Connection** | Transport level link between two peers, providing for multiple sessions. |
| **Diameter** | A feature rich AAA protocol. Utilises SCTP and TCP transports. |
| **FDA** | First Delivery Attempt - the delivery of a short message directly to the SME rather than relaying it via the MC. |
| **GSM** | Global System for Mobile communication.  
  It is a second generation cellular telecommunication system. Unlike first generation systems, GSM is digital and thus introduced greater enhancements such as security, capacity, quality and the ability to support integrated services. |
| **HTML** | HyperText Markup Language, a small application of SGML used on the World Wide Web.  
  It defines a very simple class of report-style documents, with section headings, paragraphs, lists, tables, and illustrations, with a few informational and presentational items, and some hypertext and multimedia. |
| **IP** |  
  1) Internet Protocol  
  2) Intelligent Peripheral - a box that is able to play announcements |
<p>| <strong>ISUP</strong> | ISDN User Part - part of the SS7 protocol layer and used in the setting up, management, and release of trunks that carry voice and data between calling and called parties. |
| <strong>MC</strong> | Message Centre. Also known as SMSC. |
| <strong>MMX</strong> | Messaging Manager. |
| <strong>MT</strong> | Mobile Terminated |
| <strong>MTP</strong> | Message Transfer Part (part of the SS7 protocol stack). |
| <strong>NPI</strong> | Number Plan Indicator |
| <strong>Octet</strong> | Byte - 8 bits. |
| <strong>Oracle</strong> | Oracle Corporation |
| <strong>PC</strong> | Point Code. The Point Code is the address of a switching point. |
| <strong>Peer</strong> | Remote machine, which for our purposes is capable of acting as a Diameter agent. |
| <strong>SCCP</strong> | Signalling Connection Control Part (part of the SS7 protocol stack). |
| <strong>SCP</strong> | Service Control Point. Also known as UAS. |
| <strong>SCTP</strong> | Stream Control Transmission Protocol. A transport-layer protocol analogous to the TCP or User Datagram Protocol (UDP). SCTP provides some similar services as TCP (reliable, in-sequence transport of messages with congestion control) but adds high availability. |
| <strong>Session</strong> | Diameter exchange relating to a particular user or subscriber access to a provided service (i.e. a phone call). |
| <strong>SGML</strong> | Standard Generalized Markup Language. The international standard for defining descriptions of the structure of different types of electronic document. |
| <strong>SLEE</strong> | Service Logic Execution Environment |
| <strong>SME</strong> | Short Message Entity - an entity which may send or receive Short Messages. It may be located in a fixed network, a mobile, or an SMSC. |
| <strong>SMP</strong> | Service Management Platform (also referred to as USMS). |
| <strong>SMPP</strong> | Short Message Peer-to-Peer protocol |
| <strong>SMS</strong> | Short Message Service. |
| <strong>SMSC</strong> | Short Message Service Centre - stores and forwards a short message to the indicated destination subscriber number. |
| <strong>SS7</strong> | A Common Channel Signalling system used in many modern telecoms networks that provides a suite of protocols which enables circuit and non circuit related information to be routed about and between networks. The main protocols include MTP, SCCP and ISUP. |
| <strong>SSN</strong> | Subsystem Number. An integer identifying applications on the SCCP layer. |
| <strong>TCP</strong> | Transmission Control Protocol. This is a reliable octet streaming protocol used by the majority of applications on the Internet. It provides a connection-oriented, full-duplex, point to point service between hosts. |</p>
<table>
<thead>
<tr>
<th><strong>TDMA</strong></th>
<th>Time Division Multiple Access - a communications technique that uses a common channel for communications among multiple users by allocating each a unique time slot.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TLV</strong></td>
<td>Tag-Length-Value. Optional parameters introduced in the SMPP protocol since version 3.4.</td>
</tr>
<tr>
<td><strong>UAS</strong></td>
<td>Universal Application Server - hardware on which applications run.</td>
</tr>
<tr>
<td><strong>USMS</strong></td>
<td>Universal Service Management System hardware platform.</td>
</tr>
<tr>
<td><strong>WAP</strong></td>
<td>Wireless Application Protocol. A standard designed to allow the content of the Internet to be viewed on the screen of a mobile device such as mobile phones, personal organisers and pagers. It also overcomes the processing limitation of such devices. The information and services available are stripped down to their basic text format.</td>
</tr>
</tbody>
</table>
Index

A
A sample PDU (3.2.2)
General PDU Format (3.2) • 21
About this Document
Audience • v
Changes in this document • v
Scope • v
Absolute Time Format (4.7.23.4)
PDU Field Definitions (4.7) • 43
ACS • 9
additional_status_info_text (4.8.4.1)
PDU TLV Definitions (4.8) • 45
addr_npi, source_addr_npi, dest_addr_npi,
esme_addr_npi (4.7.2)
PDU Field Definitions (4.7) • 38
addr_ton, source_addr_ton, dest_addr_ton,
esme_addr_ton (4.7.1)
PDU Field Definitions (4.7) • 38
address_range (4.7.3) • 38
PDU Field Definitions (4.7) • 38
Alert Notification Operation (4.1.3)
Session Management Operations (4.1) • 25
alert_notification
Operation Matrix (2.4) • 9
alert_notification Syntax (4.1.3.1)
Session Management Operations (4.1) • 25
alert_on_message_delivery (4.8.4.2)
PDU TLV Definitions (4.8) • 45
Alphanumeric Format (4.2.6.1.2)
Message Submission Operations (4.2) • 31
AMC • 31
Ancillary Broadcast Operations (4.6)
Cancel Broadcast Optional TLVs (4.6.2.2) • 37
cancel_broadcast_sm Operation (4.6.2) • 37
cancel_broadcast_sm Syntax (4.6.2.1) • 37
cancel_broadcast_smResp Syntax (4.6.2.3) • 37
Query Broadcast Request Optional TLVs (4.6.1.2) • 37
Query Broadcast Response Optional TLVs (4.6.1.4) • 37
query_broadcast_sm Operation (4.6.1) • 37
query_broadcast_sm Syntax (4.6.1.1) • 37
query_broadcast_smResp Syntax (4.6.1.3) • 37
Ancillary Submission Operations (4.5)
cancel_sm Operation (4.5.1) • 36
cancel_sm Syntax (4.5.1.1) • 36
cancel_smResp Syntax (4.5.1.2) • 36
Message Replacement TLVs (4.5.3.3) • 36
query_sm Operation (4.5.2.3) • 36
query_sm Syntax (4.5.2.1) • 36
query_smResp Syntax (4.5.2.2) • 36
replace_sm Operation (4.5.3) • 36
replace_sm Syntax (4.5.3.1) • 36
replace_smResp Syntax (4.5.3.2) • 36
Application Layer Communication (2.1)
Specification Clauses 2.1 and 2.2 • 7
AS • 51
ASP • 17
Audience
About this Document • v
Backward Compatibility (2.11.2)
Forward and Backward Compatibility (2.11) • 17
billing_identification (4.8.4.3)
PDU TLV Definitions (4.8) • 45
Bind Operation (4.1.1)
Session Management Operations (4.1) • 24
bind_receiver
Operation Matrix (2.4) • 9
bind_receiver Syntax (4.1.1.3)
Session Management Operations (4.1) • 24
bind_receiverResp
Operation Matrix (2.4) • 9
bind_receiverResp Syntax (4.1.1.4)
Session Management Operations (4.1) • 24
bind_transceiver
Operation Matrix (2.4) • 9
bind_transceiver Syntax (4.1.1.5)
Session Management Operations (4.1) • 24
bind_transceiverResp
Operation Matrix (2.4) • 9
bind_transceiverResp Syntax (4.1.1.6)
Session Management Operations (4.1) • 24
bound_transmitter
Operation Matrix (2.4) • 9
bound_transmitter Syntax (4.1.1.1)
Session Management Operations (4.1) • 24
bound_transmitterResp
Operation Matrix (2.4) • 9
bound_transmitterResp Syntax (4.1.1.2)
Session Management Operations (4.1) • 24
Bound_RX (2.3.3)
Session States (2.3) • 8
Bound_TRX (2.3.4)
Session States (2.3) • 8
Bound_TX (2.3.2)
Session States (2.3) • 8
Broadcast Area Format types (4.8.4.4.1)
PDU TLV Definitions (4.8) • 45
Broadcast Request Optional TLVs (4.4.2)
Message Broadcast Operations (4.4) • 35
Broadcast Response Optional TLVs (4.4.3)
Message Broadcast Operations (4.4) • 35
broadcast_channel_indicator (4.8.4.7)
PDU TLV Definitions (4.8) • 46
broadcast_content_type (4.8.4.8)
PDU TLV Definitions (4.8) • 46
broadcast_content_type_info (4.8.4.6)
PDU TLV Definitions (4.8) • 45
broadcast_frequency_interval (4.8.4.11)
PDU TLV Definitions (4.8) • 46
broadcast_message_class (4.8.4.12)
PDU TLV Definitions (4.8) • 46
broadcast_service_group (4.8.4.14)
PDU TLV Definitions (4.8) • 46
broadcast_area_identifier,
failed_broadcast_area_identifier (4.8.4.4)
PDU TLV Definitions (4.8) • 45
broadcast_area_success (4.8.4.5)
PDU TLV Definitions (4.8) • 45
broadcast_end_time (4.8.4.9)
PDU TLV Definitions (4.8) • 46
broadcast_error_status (4.8.4.10)
PDU TLV Definitions (4.8) • 46
broadcast_rep_num (4.8.4.13)
PDU TLV Definitions (4.8) • 46
broadcast_sm
Operation Matrix (2.4) • 9
broadcast_sm Operation (4.4.1)
Message Broadcast Operations (4.4) • 35
broadcast_sm Syntax (4.4.1.1)
Message Broadcast Operations (4.4) • 35
broadcast_sm_respond
Operation Matrix (2.4) • 9
broadcast_sm_respond Syntax (4.4.1.2)
Message Broadcast Operations (4.4) • 35

callback_num (4.8.4.15)
PDU TLV Definitions (4.8) • 46
callback_num_atag (4.8.4.16)
PDU TLV Definitions (4.8) • 46
callback_num_pres_ind (4.8.4.17)
PDU TLV Definitions (4.8) • 46
Cancel Broadcast Optional TLVs (4.6.2.2)
Ancillary Broadcast Operations (4.6) • 37
cancel_broadcast_sm
Operation Matrix (2.4) • 9
cancel_broadcast_sm Operation (4.6.2)
Ancillary Broadcast Operations (4.6) • 37
cancel_broadcast_sm Syntax (4.6.2.1)
Ancillary Broadcast Operations (4.6) • 37
cancel_broadcast_sm_respond
Operation Matrix (2.4) • 9
cancel_broadcast_sm_respond Syntax (4.6.2.3)
Ancillary Broadcast Operations (4.6) • 37
cancel_sm
Operation Matrix (2.4) • 9
cancel_sm Operation (4.5.1)
Ancillary Submission Operations (4.5) • 36
cancel_sm Syntax (4.5.1.1) • 36
Ancillary Submission Operations (4.5) • 36
cancel_sm_respond
Operation Matrix (2.4) • 10
cancel_sm_respond Syntax (4.5.1.2)
Ancillary Submission Operations (4.5) • 36
CDMA • 47
Changes in this document
About this Document • v
Closed (2.3.6)
Session States (2.3) • 8
Command_id (3.2.1.2)
General PDU Format (3.2) • 21
command_id (4.7.5)
PDU Field Definitions (4.7) • 38
command_length (3.2.1.1)
General PDU Format (3.2) • 21
command_length (4.7.4)
PDU Field Definitions (4.7) • 38
command_status (3.2.1.3)
General PDU Format (3.2) • 21
command_status, error_status_code (4.7.6) • 14
PDU Field Definitions (4.7) • 38
Compliance statement
Flow Control and Congestion Avoidance (2.9) • 15
congestion_state (4.8.4.18)
PDU TLV Definitions (4.8) • 46
Connection • 52
Convention
References to The Specification • 6
Conversation Abort (4.3.5.5) • 26, 33
Message Delivery Operations (4.3) • 34

data_coding (4.7.7) • 27
PDU Field Definitions (4.7) • 41
data_sm Operation Matrix (2.4) • 10
data_sm Operation (4.3.2)
Message Delivery Operations (4.3) • 33
data_sm_syntax (4.2.2.1)
Message Submission Operations (4.2) • 28
data_sm_respond
Operation Matrix (2.4) • 10
data_sm_respond Syntax (4.2.2.2) • 33
Message Submission Operations (4.2) • 28
Datagram Message Mode (4.2.10.3)
Message Submission Operations (4.2) • 32
Default Message Mode (4.2.10.1)
Message Submission Operations (4.2) • 32
deliver_sm
Operation Matrix (2.4) • 10
deliver_sm_syntax (4.3.1.1)
Message Delivery Operations (4.3) • 33
deliver_sm_resp
   Operation Matrix (2.4) • 10
deliver_sm_resp Syntax (4.3.1.2)
   Message Delivery Operations (4.3) • 33
delivery_failure_reason (4.8.4.19)
PDU TLV Definitions (4.8) • 46
dest_addr_subunit (4.8.4.23)
PDU TLV Definitions (4.8) • 47
dest_addr_np_country (4.8.4.20)
PDU TLV Definitions (4.8) • 46
dest_addr_np_information (4.8.4.21)
PDU TLV Definitions (4.8) • 46
dest_addr_np_resolution (4.8.4.22)
PDU TLV Definitions (4.8) • 47
dest_bearer_type (4.8.4.24)
PDU TLV Definitions (4.8) • 47
dest_flag (4.7.9)
PDU Field Definitions (4.7) • 41
dest_network_type (4.8.4.26)
PDU TLV Definitions (4.8) • 47
dest_network_id (4.8.4.25)
PDU TLV Definitions (4.8) • 47
dest_node_id (4.8.4.27)
PDU TLV Definitions (4.8) • 47
dest_port (4.8.4.30)
PDU TLV Definitions (4.8) • 47
dest_subaddress (4.8.4.28)
PDU TLV Definitions (4.8) • 47
dest_telematics_id (4.8.4.29)
PDU TLV Definitions (4.8) • 47
destination_addr (4.7.8)
PDU Field Definitions (4.7) • 41
display_time (4.8.4.31)
PDU TLV Definitions (4.8) • 47
dl_name (4.7.10)
PDU Field Definitions (4.7) • 41
Document Conventions
   Icons • vi
   Terminology • vi
   Typographical conventions • vi
dpf_result (4.8.4.32)
PDU TLV Definitions (4.8) • 47

E
Enquire Link Operation (4.1.2)
   Session Management Operations (4.1) • 24
Enquire Link Timer
   Session Timers (2.7) • 13
enquire_link
   Operation Matrix (2.4) • 10
enquire_link Syntax (4.1.2.1)
   Session Management Operations (4.1) • 24
enquire_link_resp
   Operation Matrix (2.4) • 10
enquire_link_resp Syntax (4.1.2.2)
   Session Management Operations (4.1) • 25
Error Handling (2.8)
   Handling Connection Failure (2.8.1) • 14
   Invalid Field Length • 14
   Operation Failure (2.8.2) • 14
   The ESME or MC is restricting the use of certain PDUs or features • 14
   The PDU data is unexpected and deemed invalid • 14
   The PDU is malformed • 14
   The PDU is not allowed in the current session state • 14
   The PDU is unrecognised • 14
esm_class (4.7.12)
PDU Field Definitions (4.7) • 42
ESME Addresses (4.2.6.3)
   Message Submission Operations (4.2) • 31
esme_addr (4.7.11)
PDU Field Definitions (4.7) • 41
Establishing a SMPP Session (2.2)
   Specification Clauses 2.1 and 2.2 • 7
F
FDA • 31
final_date (4.7.23.3)
PDU Field Definitions (4.7) • 43
Flow Control and Congestion Avoidance (2.9)
   Compliance statement • 15
Forward and Backward Compatibility (2.11)
   Backward Compatibility (2.11.2) • 17
   Forward Compatibility (2.11.1) • 17
   General • 17
Forward Compatibility (2.11.1)
   Forward and Backward Compatibility (2.11) • 17
G
General
   Forward and Backward Compatibility (2.11) • 17
   Session Management Operations (4.1) • 24
General PDU Format (3.2)
   A sample PDU (3.2.2) • 21
   Command_id (3.2.1.2) • 21
   Command_length (3.2.1.1) • 21
   Command_status (3.2.1.3) • 21
   PDU Format (3.2.1) • 21
   Sequence_number (3.2.1.4) • 21
   SMPP PDU Format (Table 3-4) • 21
   Standard Parameters (3.2.1.5) • 21
   TLV Parameters (3.2.1.6) • 21
Generic NACK Operation (4.1.4)
   Session Management Operations (4.1) • 25
generic_nack
   Operation Matrix (2.4) • 10
generic_nack Syntax (4.1.4.1)
   Session Management Operations (4.1) • 25
GSM • 26
Handling Connection Failure (2.8.1)
  Error Handling (2.8) • 14
HTML • vi

Icons
  Document Conventions • vi
IN • vi
Inactivity Timer
  Session Timers (2.7) • 13
interface_version (4.7.13)
  PDU Field Definitions (4.7) • 42
Intermediate Notification (4.3.5.2) • 26
Message Delivery Operations (4.3) • 33
International and National Format (4.2.6.1.1)
  Message Submission Operations (4.2) • 30
International Number
  Message Submission Operations (4.2) • 31
Invalid Field Length
  Error Handling (2.8) • 14
IP • 7
ISDN • 51
ISUP • 52
  its_reply_type (4.8.4.33)
    PDU TLV Definitions (4.8) • 47
  its_session_info (4.8.4.34)
    PDU TLV Definitions (4.8) • 47

Language_indicator (4.8.4.35)
  PDU TLV Definitions (4.8) • 47
Leased Lines (2.10.1)
  Session Security and Encryption (2.10) • 16

MC • 16
MC Delivery Receipt (4.3.5.1)
  Message Delivery Operations (4.3) • 33
Message Broadcast Operations (4.4)
  Broadcast Request Optional TLVs (4.4.2) • 35
    Broadcast Response Optional TLVs (4.4.3) • 35
  broadcast_sm Operation (4.4.1) • 35
  broadcast_sm Syntax (4.4.1.1) • 35
  broadcast_sm resp Syntax (4.4.1.2) • 35
  Message Replacement with broadcast_sm (4.4.4) • 35
Message Delivery Operations (4.3)
  Conversation Abort (4.3.5.5) • 34
  data_sm Operation (4.3.2) • 33
  deliver_sm Syntax (4.3.1.1) • 33
  deliver_sm resp Syntax (4.3.1.2) • 33
  Intermediate Notification (4.3.5.2) • 33
  MC Delivery Receipt (4.3.5.1) • 33
  Message Delivery Request TLVs (4.3.3) • 33
  Message Delivery Response TLVs (4.3.4) • 33
  SME Delivery Acknowledge-ment (4.3.5.3) • 34
  SME Manual/User Acknowledge-ment (4.3.5.4) • 34
  Message Delivery Request TLVs (4.3.3)
    Message Delivery Operations (4.3) • 33
  Message Delivery Response TLVs (4.3.4)
    Message Delivery Operations (4.3) • 33
  Message Length (4.2.8)
    Message Submission Operations (4.2) • 31
  Message Modes (4.2.10)
    Message Submission Operations (4.2) • 32
  Message Replace operation in submit_sm (4.2.7)
    Message Submission Operations (4.2) • 31
  Message Replacement TLVs (4.5.3.3)
    Ancillary Submission Operations (4.5) • 36
  Message Replacement with broadcast_sm (4.4.4)
    Message Broadcast Operations (4.4) • 35
  Message Submission Operations (4.2)
    Alphanumeric Format (4.2.6.1.2) • 31
    data_sm Syntax (4.2.2.1) • 28
    data_sm resp Syntax (4.2.2.2) • 28
    Datagram Message Mode (4.2.10.3) • 32
    Default Message Mode (4.2.10.1) • 32
    ESME Addresses (4.2.6.3) • 31
    International and National Format (4.2.6.1.1) • 30
    International Number • 31
    Message Length (4.2.8) • 31
    Message Modes (4.2.10) • 32
    Message Replace operation in submit_sm (4.2.7) • 31
    Message Submission Request TLVs (4.2.4) • 28
  Message Submission Response TLVs (4.2.5) • 30
NPI (4.2.6.2) • 31
NULL Address • 31
Pre-defined (4.2.9.3) • 32
Registered (4.2.9.1) • 31
Scheduled (4.2.9.2) • 31
Service Short Code • 31
Source and Destination Addressing (4.2.6) • 30
Store and Forward Message Mode (4.2.10.2) • 32
  submit_multi Syntax (4.2.3.1) • 28
  submit_multi resp Syntax (4.2.3.2) • 28
  submit_sm Syntax (4.2.1.1) • 26
  submit_sm resp Syntax (4.2.1.2) • 27
  Transaction Message Mode (4.2.10.4) • 32
Message Submission Request TLVs (4.2.4) • 27, 33
Message Submission Operations (4.2) • 28
Message Submission Response TLVs (4.2.5) • 27, 33
message_id (4.7.14)
PDU Field Definitions (4.7) • 42
message_payload (4.8.4.36)
PDU Field Definitions (4.8) • 47
message_state (4.7.15)
PDU Field Definitions (4.7) • 42
message_state (4.8.4.37)
PDU TLV Definitions (4.8) • 48
Messaging Manager
Messaging Manager implementation • 2
Messaging Manager implementation
Messaging Manager • 2
MMX • 29
more_messages_to_send (4.8.4.38)
PDU TLV Definitions (4.8) • 48
ms_availability_status (4.8.4.39)
PDU TLV Definitions (4.8) • 48
ms_msg_wait_facilities (4.8.4.40)
PDU TLV Definitions (4.8) • 48
ms_validity (4.8.4.41)
PDU TLV Definitions (4.8) • 48
MT • 28
MTP • 52
N
network_error_code (4.8.4.42)
PDU TLV Definitions (4.8) • 48
no_unsuccess (4.7.16)
PDU Field Definitions (4.7) • 42
NPI • 30
NPI (4.2.6.2)
Message Submission Operations (4.2) • 31
NULL Address
Message Submission Operations (4.2) • 31
NULL Settings (3.1.1)
Parameter Type Definitions (3.1) • 20
number_of_messages (4.8.4.43)
PDU TLV Definitions (4.8) • 48
number_of_dests (4.7.17)
PDU Field Definitions (4.7) • 42
O
Octet • 24
Open (2.3.1)
Session States (2.3) • 8
Operation Failure (2.8.2)
Error Handling (2.8) • 14
Operation Matrix (2.4)
alert_notification • 9
bind_receiver • 9
bind_receiver_resp • 9
bind_transceiver • 9
bind_transceiver_resp • 9
bind_transmitter • 9
bind_transmitter_resp • 9
broadcast_sm • 9
broadcast_sm_RESP • 9
cancel_broadcast_sm • 9
cancel_broadcast_sm_RESP • 9
cancel_sm • 9
cancel_sm_RESP • 10
data_sm • 10
data_sm_RESP • 10
deliver_sm • 10
deliver_sm_RESP • 10
enquire_link • 10
enquire_link_RESP • 10
generic_nack • 10
outbind • 10
query_broadcast_sm • 10
query_broadcast_sm_RESP • 10
query_sm • 10
query_sm_RESP • 10
Relevance • 9
replace_sm • 11
replace_sm_RESP • 11
submit_multi • 11
submit_multi_RESP • 11
submit_sm • 11
submit_sm_RESP • 11
unbind • 11
unbind_RESP • 11
Oracle • ii
outbind
Operation Matrix (2.4) • 10
outbind Syntax (4.1.1.7)
Session Management Operations (4.1) • 24
Outbound (2.3.7)
Session States (2.3) • 8
P
Parameter Type Definitions (3.1)
NULL Settings (3.1.1) • 20
SMPP Parameter Field Size Notation (3.1.2) • 20
SMPP PDU Parameter Types (Table 3-1) • 20
password (4.7.18)
PDU Field Definitions (4.7) • 42
payload_type (4.8.4.44)
PDU TLV Definitions (4.8) • 48
PC • 31
PDU Field Definitions (4.7)
Absolute Time Format (4.7.23.4) • 43
addr_npi, source_addr_npi, dest_addr_npi,
esme_addr_npi (4.7.2) • 38
addr_ton, source_addr_ton, dest_addr_ton,
esme_addr_ton (4.7.1) • 38
address_range (4.7.3) • 38
command_id (4.7.5) • 38
command_length (4.7.4) • 38
command_status, error_status_code (4.7.6) • 38
data_coding (4.7.7) • 41
dest_flag (4.7.9) • 41
destination_addr (4.7.8) • 41
dl_name (4.7.10) • 41
esm_class (4.7.12) • 42
esme_addr (4.7.11) • 41
final_date (4.7.23.3) • 43
interface_version (4.7.13) • 42
message_id (4.7.14) • 42
message_state (4.7.15) • 42
no_unsuccess (4.7.16) • 42
number_of_dests (4.7.17) • 42
password (4.7.18) • 42
priority_flag (4.7.19) • 42
protocol_id (4.7.20) • 42
registered_delivery (4.7.21) • 42
Relative Time Format (4.7.23.5) • 43
replacement_flag (4.7.22) • 43
scheduled_delivery_time (4.7.23.1) • 43
sequence_number (4.7.24) • 43
service_type (4.7.25) • 43
short_message (4.7.26) • 43
sm_default_msg_id (4.7.27) • 43
sm_length (4.7.28) • 43
source_addr (4.7.29) • 44
system_id (4.7.30) • 44
UNIX Regular Expressions (4.7.3.1) • 38
validity_period (4.7.23.2) • 43
PDU Format (3.2.1)
General PDU Format (3.2) • 21
PDU Sequencing (2.6)
Sequence Numbers Across Sessions (2.6.3) • 12
Synchronous Vs. Asynchronous (2.6.4) • 12
The PDU Sequence Number (2.6.1) • 12
Why Asynchronous? (2.6.2) • 12
Why use Monotonically Increasing Sequence numbers? (2.6.2) • 12
PDU TLV Definitions (4.8)
additional_status_info_text (4.8.4.1) • 45
alert_on_message_delivery (4.8.4.2) • 45
billing_identification (4.8.4.3) • 45
Broadcast Area Format types (4.8.4.4.1) • 45
broadcast_channel_indicator (4.8.4.7) • 46
broadcast_content_type (4.8.4.8) • 46
broadcast_content_type_info (4.8.4.6) • 45
broadcast_frequency_interval (4.8.4.11) • 46
broadcast_message_class (4.8.4.12) • 46
broadcast_service_group (4.8.4.14) • 46
failed_broadcast_area_identifier (4.8.4.4) • 45
broadcast_area_success (4.8.4.5) • 45
callback_num (4.8.4.15) • 46
callback_num_atag (4.8.4.16) • 46
callback_num_pres_ind (4.8.4.17) • 46
congestion_state (4.8.4.18) • 46
delivery_failure_reason (4.8.4.19) • 46
dest_addr_subunit (4.8.4.23) • 47
dest_addr_np_country (4.8.4.20) • 46
dest_addr_np_information (4.8.4.21) • 46
dest_addr_np_resolution (4.8.4.22) • 47
dest_bearer_type (4.8.4.24) • 47
dest_network_type (4.8.4.26) • 47
dest_network_id (4.8.4.25) • 47
dest_node_id (4.8.4.27) • 47
dest_port (4.8.4.30) • 47
dest_subaddress (4.8.4.28) • 47
dest_telematics_id (4.8.4.29) • 47
display_time (4.8.4.31) • 47
dpf_result (4.8.4.32) • 47
its_reply_type (4.8.4.33) • 47
its_session_info (4.8.4.34) • 47
language_indicator (4.8.4.35) • 47
message_payload (4.8.4.36) • 47
message_state (4.8.4.37) • 48
more_messages_to_send (4.8.4.38) • 48
ms_availability_status (4.8.4.39) • 48
ms_msg_wait_facilities (4.8.4.40) • 48
ms_validity (4.8.4.41) • 48
network_error_code (4.8.4.42) • 48
number_of_messages (4.8.4.43) • 48
payload_type (4.8.4.44) • 48
Position of TLVs in SMPP messages (4.8) • 45
privacy_indicator (4.8.4.45) • 48
qos_time_to_live (4.8.4.46) • 48
receipted_message_id (4.8.4.47) • 48
sar_msg_ref_num (4.8.4.48) • 48
sar_segment_seqnum (4.8.4.49) • 48
sar_total_segments (4.8.4.50) • 48
sc_interface_version (4.8.4.51) • 49
set_dpf (4.8.4.52) • 49
sms_signal (4.8.4.53) • 49
source_subaddress (4.8.4.60) • 49
source_telematics_id (4.8.4.61) • 49
source_addr_subunit (4.8.4.54) • 49
source_bearer_type (4.8.4.55) • 49
source_network_id (4.8.4.56) • 49
source_network_type (4.8.4.57) • 49
source_node_id (4.8.4.58) • 49
source_port (4.8.4.59) • 49
TLV Definitions (4.8) • 45
TLV Length (4.8.2) • 45
TLV Tag (4.8.1) • 45
TLV Value (4.8.3) • 45
user_message_reference (4.8.4.62) • 49
user_response_code (4.8.4.63) • 49
ussd_service_op (4.8.4.64) • 49
Position of TLVs in SMPP messages (4.8)
PDU TLV Definitions (4.8) • 45
Pre-defined (4.2.9.3)
Message Submission Operations (4.2) • 32
priority_flag (4.7.19)
PDU Field Definitions (4.7) • 42
privacy_indicator (4.8.4.45)
PDU TLV Definitions (4.8) • 48
protocol_id (4.7.20)
PDU Field Definitions (4.7) • 42

Q
qos_time_to_live (4.8.4.46)
PDU TLV Definitions (4.8) • 48
Query Broadcast Request Optional TLVs (4.6.1.2)
Ancillary Broadcast Operations (4.6) • 37
Query Broadcast Response Optional TLVs (4.6.1.4)
Ancillary Broadcast Operations (4.6) • 37
query_broadcast_sm
Operation Matrix (2.4) • 10
query_broadcast_sm Operation (4.6.1.1)
Ancillary Broadcast Operations (4.6) • 37
query_broadcast_sm Syntax (4.6.1.3)
Ancillary Broadcast Operations (4.6) • 37
query_sm
Operation Matrix (2.4) • 10
query_sm Operation (4.5.2)
Ancillary Submission Operations (4.5) • 36
query_sm Syntax (4.5.2.1) • 36
Ancillary Submission Operations (4.5) • 36
query_smresp
Operation Matrix (2.4) • 10
query_smresp Syntax (4.5.2.2)
Ancillary Submission Operations (4.5) • 36

R
receipted_message_id (4.8.4.47)
PDU TLV Definitions (4.8) • 48
References to The Specification
Convention • 6
Registered (4.2.9.1)
Message Submission Operations (4.2) • 31
registered_delivery (4.7.21)
PDU Field Definitions (4.7) • 42
Relative Time Format (4.7.23.5)
PDU Field Definitions (4.7) • 43
Relevance
Operation Matrix (2.4) • 9
Session Timers (2.7) • 13
replace_if_present_flag (4.7.22)
PDU Field Definitions (4.7) • 43
replace_sm
Operation Matrix (2.4) • 11
replace_sm Operation (4.5.3)
Ancillary Submission Operations (4.5) • 36
replace_sm Syntax (4.5.3.1) • 36
Ancillary Submission Operations (4.5) • 36
replace_smresp
Operation Matrix (2.4) • 11
replace_smresp Syntax (4.5.3.2)
Ancillary Submission Operations (4.5) • 36
Response Timer
Session Timers (2.7) • 13

S
sar_msg_ref_num (4.8.4.48)
PDU TLV Definitions (4.8) • 48
sar_segment_seqnum (4.8.4.49)
PDU TLV Definitions (4.8) • 48
sar_total_segments (4.8.4.50)
PDU TLV Definitions (4.8) • 48
sc_interface_version (4.8.4.51)
PDU TLV Definitions (4.8) • 49
SCCP • 31
Scheduled (4.2.9.2)
Message Submission Operations (4.2) • 31
scheduled_delivery_time (4.7.23.1)
PDU Field Definitions (4.7) • 43
Scope
About this Document • v
SCP • 2
SCTP • 51
Secure Transport Layer (2.10.2)
Session Security and Encryption (2.10) • 16
Secure Tunnel (2.10.4)
Session Security and Encryption (2.10) • 16
Secure VPN (2.10.3)
Session Security and Encryption (2.10) • 16
Sequence Numbers Across Sessions (2.6.3)
PDU Sequencing (2.6) • 12
Sequence_number (3.2.1.4)
General PDU Format (3.2) • 21
sequence_number (4.7.24)
PDU Field Definitions (4.7) • 43
Service Provider • 51
Service Short Code
Message Submission Operations (4.2) • 31
service_type (4.7.25)
PDU Field Definitions (4.7) • 43
Session • 13
Session Init Timer
Session Timers (2.7) • 13
Session Management Operations (4.1)
The ESME or MC is restricting the use of certain PDUs or features.

- **Error Handling (2.8) • 14**
  - The PDU data is unexpected and deemed invalid.
  - The PDU is malformed.
  - The PDU is not allowed in the current session state.
  - The PDU is unrecognised.

**TLV**

- **TLV Definitions (4.8.4) • 45**
  - PDU TLV Definitions (4.8) • 45
- **TLV Length (4.8.2) • 45**
  - PDU TLV Definitions (4.8) • 45
- **TLV Parameters (3.2.1.6) • 21**
  - General PDU Format (3.2) • 21
- **TLV Tag (4.8.1) • 45**
  - PDU TLV Definitions (4.8) • 45
- **TLV Value (4.8.3) • 45**
  - PDU TLV Definitions (4.8) • 45
- **Transaction Message Mode (4.2.10.4) • 32**
  - Message Submission Operations (4.2) • 32

**Typographical conventions**

- Document Conventions • vi

**U**

- **UAS • 52**
- **unbind**
  - Operation Matrix (2.4) • 11
  - Session Management Operations (4.1) • 24
- **unbind_resp**
  - Operation Matrix (2.4) • 11
  - Session Management Operations (4.1) • 24
- **Unbound (2.3.5)**
  - Session States (2.3) • 8
- **UNIX Regular Expressions (4.7.3.1) • 38**
- **user_message_reference (4.8.4.62) • 49**
- **user_response_code (4.8.4.63) • 49**

**V**

- **validity_period (4.7.23.2) • 43**
  - PDU TLV Definitions (4.8) • 49

**W**

- **WAP • 38**
- **Why Asynchronous? (2.6.5) • 12**
  - PDU Sequencing (2.6) • 12
- **Why use Monotonically Increasing Sequence numbers? (2.6.2) • 12**
  - PDU Sequencing (2.6) • 12