

Intel[®] NetStructure[™] SS7 Protocols TCAP Programmer's Manual

Document Reference: U06SSS

Disclaimer

The product may contain design defects or errors known as errata, which may cause the product to deviate from published specifications.

Information in this document is provided in connection with Intel® products. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document. Except as provided in Intel's Terms and Conditions of Sale for such products, Intel assumes no liability whatsoever, and Intel disclaims any express or implied warranty, relating to sale and/or use of Intel products including liability or warranties relating to fitness for a particular purpose, merchantability, or infringement of any patent, copyright or other intellectual property right. Intel products are not designed, intended or authorized for use in any medical, life saving, or life sustaining applications or for any other application in which the failure of the Intel product could create a situation where personal injury or death may occur. Intel may make changes to specifications and product descriptions at any time, without notice.

Intel and Intel NetStructure are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

* Other names and brands may be claimed as the property of others.

Copyright © 1993-2003 Intel Corporation. All rights reserved. No part of this document may be copied, or reproduced in any form, or by any means without prior written consent of Intel.

REVISION HISTORY

ISSUE	DATE	BY	CHANGES
1	06-Nov-95	SRG	Minor typographical corrections and the addition of the TC-NULL-IND primitive.
2	15-May-97	SFP	ANSI operation added.
3	24-Dec-98	SFP	Dialogue Groups and message tracing added.
4	23-Feb-99	SFP	Address format parameter added to module configuration. Operation with multiple local application programs (sub-systems) described. Description of TCP_MSG_S_TCU_ID added. Additional information provided for parameters in TCP_MSG_CONFIG. Structure of timer messages added in an appendix Message type reference appendix added
5	17-Aug-00	JET	Addition of message definitions for dialogue discard indications, component discards indications and module version identification.
6	19-Nov-01	JET	Messages to monitor module and dialogue status added.
7	16-Jul-03	ML	Branding changed: references to System7 removed.

CONTENTS

1. INTRODUCTION	6
2. ABBREVIATIONS	7
3. RELATED DOCUMENTATION	7
4. FEATURE OVERVIEW.....	7
5. GENERAL DESCRIPTION	8
5.1 Module Overview	8
5.2 Module Configuration.....	9
5.3 Dialogue ID assignment.....	9
5.4 Dialogue ID groups	10
5.5 Local Transaction ID format.....	10
5.6 Constant definitions.	11
6. INTERFACE TO SYSTEM SERVICES.....	12
6.1 System Functions	12
6.2 Timer Operation.....	12
7. INTERFACE TO NETWORK LAYER	13
8. INTERFACE TO TC-USER.....	13
8.1 Introduction	13
8.2 Multiple TC-User applications	14
8.3 Primitive parameters.....	15
8.4 Component Primitive Types.....	16
8.5 Dialogue Primitive Types	17
8.6 TC-COMPONENT-REQUEST	18
8.7 TC-COMPONENT-INDICATION.....	22
8.8 TC-DIALOGUE-REQUEST	24
8.9 TC-DIALOGUE-INDICATION	32
9. MANAGEMENT INTERFACE.....	34
10. NON-PRIMITIVE INTERFACE	35
10.1 TCAP Configuration Request.....	36
10.2 Configure Dialogue Group Request.....	42
10.3 Configure TC-User Request	44
10.4 TCAP Set Default Parameters Request.....	46
10.5 Read TCAP Statistics Request	48
10.6 Read TCAP RAM Request	49
10.7 Read TCAP Dialogue Request	50
10.8 Read TCAP Module Status Request	51
10.9 Read TCAP Dialogue Status Request	53
10.10 Maintenance Event Indication.....	55
10.11 Software Event Indication	57
10.12 TCAP Dialogue discard indication	59
10.13 TCAP Component discard indication	61

10.14 Read Revision Request	62
10.15 Management Event Indication	63
10.16 Set Trace Mask Request	64
10.17 Trace Event Indication	66
APPENDIX A	67
A.1 Timer Services.....	67
A.2 Keep Time	67
A.3 Timer Expiry	68
APPENDIX B	69
B.1 Message Type reference.....	69

1. INTRODUCTION

The TCAP module is a portable software implementation of the Signalling System Number 7, Transaction Capabilities Application Part (TCAP). It operates according to either ITU-T Q.771-Q.774 (1992) or ANSI T1.114-1996 selection being by a run-time option. This is the Programmer's Manual, which is intended for users developing their own applications that will interface with and use the functionality provided by the TCAP module.

The module uses the services provided by the underlying network-layer service provider for the transfer of information between nodes, and provides generic services to applications whilst remaining independent of both the network layer and the application.

The TCAP module is an event driven task, which uses standard structured message types for communication with other layers of the protocol stack. These messages are used to convey the protocol primitives between TCAP and the TC-User and TCAP and the network layer. Each message contains the primitive parameters as defined in the CCITT recommendations thereby ensuring that the module can easily be interfaced with other vendors' implementations of the adjacent layers. Typically the module is used in conjunction with the SCCP and MTP modules.

This manual provides an overview of the internal operation of the TCAP module and defines the structure of all messages used to interface with the module.

2. ABBREVIATIONS

ANSI	American National Standards Institute.
APDU	Application Protocol Data Unit.
CCITT	The International Telegraph & Telephone Consultative Committee.
ITU-T	International Telecommunication Union (formerly CCITT).
MTP	Message Transfer Part.
SCCP	Signalling Connection Control Part.
TCAP	Transaction Capabilities Application Part.

3. RELATED DOCUMENTATION

- [1] ITU-T Recommendations Q.771, Q.772, Q.773, Q.774 & Q.775.
- [2] ANSI T1.114-1996
- [3] U05SSS, SCCP Programmer's Manual
- [4] U10SSS, Software Environment Programmer's Manual

4. FEATURE OVERVIEW

Key features of the TCAP module include:

- Full implementation of ITU-T Q.771-Q.774 (1992) and ANSI T1.114 (1996).
- Inter working with ITU-T 1988 and ANSI 1992 recommendations.
- Class 1, 2, 3, and 4 operations.
- Dialogue support for application context and user information.
- Automatic generation of Transaction ID.
- Supports the use of multiple distributed instances of TCAP.
- Message oriented interface.
- Grouping of dialogue id ranges for operation with multiple application programs.
- Debug tracing of messages exchanged with the TC-User and SCCP.

5. GENERAL DESCRIPTION

5.1 Module Overview

The TCAP module is a full implementation of the 1992 ITU-T recommendations Q.771 - Q.774 and ANSI T1.114-1996, including support for the optional dialogue portion for conveying information relating to application context and user information. Internally the module is sub-divided into two layers: the Component Sub-Layer (CSL) and the Transaction Sub-Layer (TSL).

The component sub-layer accumulates the user-supplied Application Protocol Data Units (APDU) (i.e. components) and stores them in an internal buffer. On receipt of the appropriate dialogue-handling primitive from the user the components are combined with the (optional) dialogue portion and passed to the transaction sub-layer. An invocation state machine is started for each invoke component. Messages received from the transaction sub-layer are checked and conveyed to the user. The dialogue primitive is issued first (including the optional dialogue portion), followed by each component (in the same order that they were received for transmission at the sending end).

The transaction sub-layer receives messages from the component sub-layer and ensures they are valid for the current state of the transaction. It then adds the transaction portion (containing address and quality of service information) to the message and passes it to the network layer. Messages received from the network layer are validated by the transaction sub-layer; a transaction ID is assigned for each new transaction and the message is conveyed to the component sub-layer.

The module is event driven; it has a single input queue into which events from other modules (TC-User, network-layer, management etc.) are written. The module processes each event in turn until the input queue is empty in which case it will do nothing until the next event is received. Output from the module is directed depending on the type of event to the TC-User module, the network-layer module, the Management module or the Maintenance module.

Internally there are a number of data structures used by the module. The maximum dimensions of these structures are determined by compile time constants. The three constants of importance to the user are:

- a) The maximum number of simultaneous dialogues (and hence transactions) supported by the module.
- b) The maximum number of components internally accumulated by the module awaiting transmission.
- c) The maximum number of invocations active at any time.

In addition the module requires a periodic timer tick notification be issued to it (using the input queue), typically every tenth of a second. (This can either be generated by a timer module or using the services of the selected operating system).

5.2 Module Configuration

The module provides maximum flexibility by allowing a large number of user configuration options to be set up at run time. This allows the users to customise the operation of the module to suit the particular requirements of the final application. All configuration parameters are sent to the module's input event queue in the same manner as other protocol messages.

The first message that must be sent to the module is a global configuration message (any messages received prior to the global configuration message will be discarded). This message specifies the operating mode for the TCAP module as being either ITU-T or ANSI. It also contains the module id for all modules to which TCAP issues messages, user supplied values for the maximum number of dialogues (incoming and outgoing), the maximum number of buffered components and the maximum number of active invocations that are required to be available to the user. The module checks that the values requested are compatible with the values it can support.

A second configuration message allows the user to supply default values for a number of protocol parameters (e.g. originating address, destination address, quality of service etc.). These default values will then be used whenever the particular parameter is required by the protocol but not present in the message received from the user.

5.3 Dialogue ID assignment

The TCAP module supports a number of active dialogues at a single instant in time. TC-User primitives are associated with a particular dialogue using a Dialogue ID, which is of purely local significance between the TC-User and TCAP.

A Dialogue ID is assigned at the start of a dialogue, when the first primitive is exchanged between the TC-User and TCAP. For a dialogue initiated by the TC-User (an 'outgoing dialogue'), the value is selected by the TC-User. For a dialogue initiated from a remote TCAP (an 'incoming dialogue') the value is set by the TCAP module. Once a dialogue has started, all user primitives, both requests and indications that refer to the same dialogue will include the same Dialogue ID value.

The dialogue ID is a 16-bit value in the range 0 to 32767 for outgoing dialogues and from 32768 to 65535 for incoming dialogues (i.e. the most significant bit is set when the dialogue ID refers to an incoming dialogue).

The range of valid dialogue ID values to be supported by a particular instance of TCAP is set up at configuration time. Two ranges of dialogue ID must be configured by the user, one for use with outgoing dialogues and the other for use with incoming dialogues. The total number of dialogue ID's must not exceed the maximum number of simultaneous dialogues that the module can support.

The dialogue ID selected by the TC-User for an outgoing dialogue must lie within the configured range of outgoing dialogue ID's. Dialogue ID's for incoming dialogues are allocated automatically by the TCAP module (from the configured range of incoming dialogue ID's) so that the dialogue ID that has been unused for the longest period is used next. Setting the most significant bit of the dialogue ID for all incoming dialogues ensures that it is not possible for the TC-User to select an ID for an outgoing dialogue at the same instant that TCAP selects the same ID for use with an incoming dialogue.

Internally to the TCAP module the dialogue ID is used to generate a 'dialogue reference' which lies in the range from 0 up to one less than the maximum number of simultaneous dialogues supported. The dialogue reference is used by the TCAP module to generate the local transaction ID.

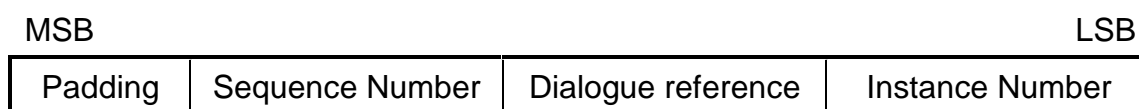
5.4 Dialogue ID groups

A dialogue group enables common attributes to be assigned to a number of dialogues identified by their dialogue IDs, such as user application instance. This enables a unique range of dialogue identifiers to be permanently assigned to different instances of a user application; hence TCAP is able to support a distributed application (this would be used for example in a high availability environment). A dialogue group is configured with a single message. The TCAP module supports up to 32 dialogue groups.

5.5 Local Transaction ID format

Peer TCAP entities use transaction IDs to associate TCAP protocol data units (PDU's) with a particular transaction. These IDs are included as a parameter in each TCAP PDU sent across the SS7 network. The TCAP module automatically generates the ID used to reference the transaction locally. For an outgoing dialogue, this will be the Originating transaction ID in the TCAP PDU, for an incoming dialogue, this will be the Destination or Responding Transaction ID in the TCAP PDU.

The local transaction ID is assigned by the TCAP module in such a manner as to ensure that the same transaction ID is not re-allocated until some time after the dialogue has finished. The transaction ID is made up of 4 fields: the instance number, the dialogue reference, a sequence number and a padding field as follows:



The 'instance number' allows TCAP to be distributed over a number of separate hardware platforms, each running as a separate instance and responsible for handling a different range of dialogue ID's (from the TC-User viewpoint) and using a different Instance Number within the Transaction ID (from the Network-layer viewpoint).

The 'dialogue reference' has a one-to-one mapping with the dialogue ID at any single instance of TCAP and ranges from 0 up to one less than the total number of dialogues supported. Note that the dialogue reference is not the same value as the dialogue ID.

The 'sequence number' is allocated on a cyclic basis for each individual dialogue ID and ensures the maximum possible time elapses before re-use of a transaction ID.

The width of each transaction ID field (in bits) is a run-time configuration option allowing the user to adjust the relative field widths to suit the application.

Internally the size of the transaction ID is rounded up to a multiple of 8 bits (by adding zeros in the padding field). The transaction ID is then converted to an octet string with the first octet containing the most significant 8 bits of the transaction ID.

5.6 Constant definitions.

To assist the user when writing an application, a 'C' language header file (*tcp_inc.h*) is available containing all the definitions and constants necessary to interface with the TCAP module. This file contains definitions for all the mnemonics listed in this Programmer's Manual.

6. INTERFACE TO SYSTEM SERVICES

6.1 System Functions

In addition to the primitive interface and the management interface to the TCAP module (which are described in later sections) the module requires a few basic system services to be supplied by the underlying operating system.

The following functions are required for inter-task communication:

GCT_send	Sends a message to another task.
GCT_receive	Accept next message from input event queue, blocking the task if no message is ready.
GCT_grab	As receive but not blocking if no message is ready.

The following functions are required for allocation and release of inter task messages:

getm	Allocate a message.
relm	Release a message.

These functions are described in the Software Environment Programmers Manual.

6.2 Timer Operation

In order to provide internal implementation of the TCAP protocol timers the module needs to receive a periodic timer tick message. This is usually achieved using either the Enhanced Driver Module or the Timer module in which case the following messages are used by the TCAP module:

KEEP_TIME	Issued by TCAP to initialise the timer services.
TM_EXP	Issued by the timer module to notify of time-out.

The format of these messages is described in Appendix 1.

The user should note that whilst the timer functionality is usually provided by the given modules, the timer functionality required by the TCAP module is very basic (just a single message being issued on a periodic basis). In most cases it is a trivial exercise to implement this functionality using the users own choice of operating environment if required.

7. INTERFACE TO NETWORK LAYER

The TCAP module communicates with the Network-layer Service Provider using the following primitives, all of which are defined in CCITT Recommendation Q.711:

N-UNITDATA-REQ
N-UNITDATA-IND
N-NOTICE-IND

The message format used to convey these primitives is defined in the SCCP Programmer's Manual. The following messages are used:

SCP_MSG_TX_REQ	Messages issued by TCAP
SCP_MSG_RX_IND	Messages issued to TCAP

The TCAP module is usually used in conjunction with the SCCP module. However, the use of primitives in accordance with Q.711 ensures that it can also be integrated with other Network-layer Service Provider implementations if required.

8. INTERFACE TO TC-USER

8.1 Introduction

All primitives at the application interface (i.e. between the TCAP module and the TC-User) are passed by sending messages between the modules. Primitive requests are originated from the TC-User and request TCAP to carry out a specified action. Primitive indications are sent from TCAP to indicate received TCAP PDU data or local protocol events to the TC-User.

The following messages are used:

TC-COMPONENT-REQ	Conveys component from TC-User to TCAP.
TC-DIALOGUE-REQ	Conveys dialogue primitive from TC-User to TCAP.
TC-COMPONENT-IND	Conveys component from TCAP to TC-User.
TC-DIALOGUE-IND	Conveys dialogue primitive from TCAP to TC-User.

The basic structure of each message (irrespective of the TCAP primitive contained within it) is the same and is described in the Software Environment Programmer's Manual. The message contains a message header, the length of the user data and the user data itself. The message must be contained in a single buffer which should be allocated by the sending module (using the **getm** function) and either released (using the **relm** function) or passed to another module by the receiving module. The **getm** and **relm** functions are described in Section 6.

The message header contains a 'type', the value of this parameter indicating that either a dialogue or component-handling primitive is being conveyed by the message. The following message types are defined:

Primitive	Message type	Value
TC-COMPONENT-REQ	TCP_MSG_CPT_REQ	0xc781
TC-COMPONENT-IND	TCP_MSG_CPT_IND	0x8782
TC-DIALOGUE-REQ	TCP_MSG_DLG_REQ	0xc783
TC-DIALOGUE-IND	TCP_MSG_DLG_IND	0x8784

8.2 Multiple TC-User applications

In a multi-tasking operating system it is possible to have more than one TC-User application program running as a separate task. The message passing environment identifies each task with a unique module identifier (or module id), used as the destination for any message sent to this task from other processes in the system. TCAP exchanges messages with peer TCAP entities using SCCP format addressing. This assigns a unique sub-system number and point code to each unique TC-User.

The TCAP configuration sets a default module id for dialogue and component indications; this value identifies the destination task (user application) that will receive these indications if no additional configuration data is supplied.

In addition to the default user application module id, it is possible to set a different module id for each different local sub-system. In this environment, each unique user application behaves as a unique local sub-system, with a unique sub-system number (used by the SCCP addressing) and also a unique module identifier (used by the inter-process message passing).

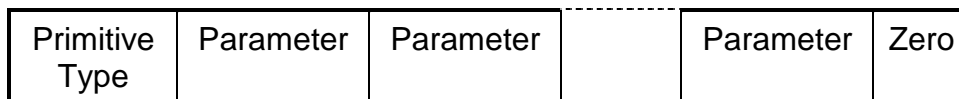
The association between a locally initiated dialogue and a user application program module id is made when the first primitive request is sent. The dialogue is associated with the module id being taken from the 'source' field of the first primitive request message.

Dialogues initiated from a remote TCAP entity are received by TCAP from SCCP containing a called address, identifying a local sub-system. The sub-system number in this address is matched to module id from a user provided configuration setting. The association is made using the first SCCP message received for each of these dialogues. If the local sub-system has not been configured, the dialogue is associated with the 'default' user application module id.

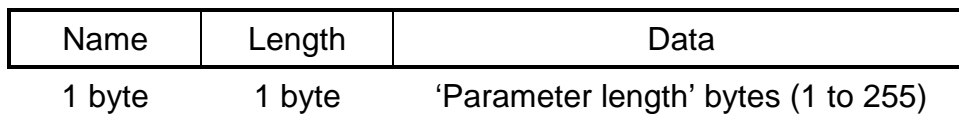
8.3 Primitive parameters

Each TC-User primitive includes a number of parameters. These parameters are conveyed in the parameter area of the message that conveys the primitive.

The first byte in the parameter area is the primitive type octet and the last byte is a zero byte to indicate that there are no further parameters in the parameter area. Any parameters associated with the message are placed between the message type code and the final (zero) byte. Therefore the parameter area is formatted as follows:



The parameters may be placed in any order. The first byte of a parameter is the parameter name, the second byte is the length of the parameter data to follow (excluding the parameter name and the length byte itself), and this is followed by the parameter data. The encoding of the parameter data aligns exactly with the parameter format specified in the appropriate ITU-T recommendation whenever possible. Therefore each parameter is formatted as follows:



Within each message there are mandatory parameters which must always be present and optional parameters which may or may not be present. In some cases the optional parameters may have default values (set up at configuration time) which are added by the TCAP module if not provided by the user.

8.4 Component Primitive Types

Component primitives convey a request to perform an operation, or a reply. The following component primitive types are provided:

Mnemonic	Originator	Function	Value
TCPPT_TC_INVOKE	TC-User	Request an operation to be performed. This primitive is used for ITU-T Invoke and ANSI Invoke (last).	8
TCPPT_TC_INVOKE_NL	TC-User	Request an operation to be performed. Further components will convey additional information.	17
TCPPT_TC_RESULT_L	TC-User	Report successful completion of an operation.	9
TCPPT_TC_RESULT_NL	TC-User	Report a segment of the successful completion of an operation, further components will convey additional information.	10
TCPPT_TC_U_ERROR	TC-User	Report the unsuccessful completion of an operation.	11
TCPPT_TC_U_REJECT	TC-User	Report receipt and rejection of a component (other than a reject).	16
TCPPT_TC_U_CANCEL	TC-User	Terminate an operation initiated from the local TCAP prematurely (local action only).	13
TCPPT_TC_L_CANCEL	TCAP	Report that the response expected to an operation was not received within a specified time.	12
TCPPT_TC_L_REJECT	TCAP	Report that a received component was rejected locally due to protocol error.	14
TCPPT_TC_R_REJECT	TCAP	Report that a component was rejected by a responding TCAP.	15
TCPPT_TC_NULL	TCAP	Issued in the cases that a component is discarded by the TCAP module but when it is still necessary to indicate a component to the TC-User in order to preserve the 'LAST-COMPONENT' indication.	0

TC-User originated components may be initiated by the local or responding TC-User, hence these primitives may be both TC-User Component Requests and TC-User Component Indications.

TCAP originated components are always TC-User Component Indications.

8.5 Dialogue Primitive Types

Dialogue handling primitives provide the mechanism by which components are exchanged between peer TCAP entities. The following primitive types are provided:

Mnemonic	Originator	Function	Value
TCPPT_TC_UNI	TC-User	Unstructured Dialogue	1
TCPPT_TC_BEGIN	TC-User	Begin a structured Dialogue.	2
TCPPT_TC_CONTINUE	TC-User	Continue a dialogue, responding TCAP may end this dialogue.	3
TCPPT_TC_END	TC-User	End a dialogue.	4
TCPT_TC_U_ABORT	TC-User	Abort a dialogue.	5
TCPPT_TC_P_ABORT	TCAP	Abort a dialogue due to and abnormal protocol event.	6
TCPPT_TC_NOTICE	TCAP	Report that the network layer was unable to deliver a TCAP PDU to the destination.	7

TC-User originated dialogue primitives may be initiated by the local or responding TC-User, hence these primitives may be both TC-User Dialogue Requests and TC-User Dialogue Indications.

TCAP originated primitives are always TC-User Dialogue Indications.

The following alternate set of definitions is provided for ANSI TCAP users.

ANSI Mnemonic	Equivalent ITU-T Mnemonic	Value
TCPPTA_TC_UNI	TCPPT_TC_UNI	1
TCPPTA_TC_QUERY	TCPPT_TC_BEGIN	2
TCPPTA_TC_CONVERSATION	TCPPT_TC_CONTINUE	3
TCPPTA_TC_RESPONSE	TCPPT_TC_END	4
TCPPTA_TC_U_ABORT	TCPT_TC_U_ABORT	5
TCPPTA_TC_P_ABORT	TCPPT_TC_P_ABORT	6
TCPPTA_TC_NOTICE	TCPPT_TC_NOTICE	7

The following sections define the message format and content of the parameter area for each of the messages exchanged between the TC-User and TCAP.

8.6 TC-COMPONENT-REQUEST

Synopsis:

Protocol message sent from the TC-User to TCAP containing a TC-Component for association with a dialogue.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_CPT_REQ (0xc781)	
id	dialogue_ID	
src	Sending module_id	
dst	TCP_TASK_ID	
rsp_req	0	
hclass	0	
status	0	
err_info	0	
len	Number of bytes of user data	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	1	Component primitive type octet.
1	len - 2	Parameters in Name-Length-Data format.
len - 1	1	Set to zero indicating end of message.

Description:

This message is used by the TC-User to send Component sub-layer primitives to TCAP. The primitives are accumulated within TCAP for the specified dialogue ID until the appropriate dialogue handling primitive is issued by the TC-User when the component will be assembled into a message and passed to the network-layer service provider.

All component request primitives contain a dialogue ID that is encoded in the message header. It does not form part of the parameter area.

Parameter area contents:

The component primitive type octet is coded as defined in Section 8.3, Component Primitive Types.

The following parameter names are defined for use in component primitive messages:

Parameter	Mnemonic	Value
Component portion	TCPN_COMPONENT	1
Last component	TCPN_LAST_CPT	2
Class	TCPN_CLASS	3
Timeout	TCPN_TIMEOUT	4
Invoke ID	TCPN_INVOKE_ID	5

Parameters of local significance (i.e. those that do not form part of the transmitted or received network-layer message) are allocated their own parameter names whilst the remaining parameters (i.e. those that form the 'Component Portion' of the transmitted message) are allocated a single parameter name.

The data section of the 'component portion' parameter is encoded in accordance with the specification for the component as defined by recommendation Q.773 or T1.114.3, commencing with the Component Type Tag. The following table details the component type required for each TC-User component primitive:

Primitive	Component type
TCPPT_TC_INVOKE	Invoke, Invoke (last) †
TCPPT_TC_INVOKE_NL	Invoke (not last) †
TCPPT_TC_RESULT_L	Result (last)
TCPPT_TC_RESULT_NL	Result (not last)
TCPPT_TC_U_ERROR	Return error
TCPPT_TC_U_REJECT	Reject
TCPPT_TC_L_REJECT	Reject
TCPPT_TC_R_REJECT	Reject

† Invoke (last) and Invoke (not last) are available for ANSI operation only. For ITU-T operation, Invoke should be used.

*Note that whilst the component portion of a message transferred across the network may contain more than one component, each component primitive message between the TC-user and TCAP must contain exactly one component (except the TC-U-CANCEL request which is not sent to the network. This should contain only the **Invoke ID** parameter).*

The coding for each parameter type is given in the following tables:

Parameter name	TCPN_COMPONENT
Parameter length	Variable, ranging from 1 to 255
Parameter data	Component data encoded as specified in Q.773 or T1.114.3, commencing with the Component Type tag

Parameter name	TCPN_LAST_CPT
Parameter length	Fixed, set to 1
Parameter data	Single octet set to 0 if there are more components to follow and set to 1 if this is the last component.

Parameter name	TCPN_CLASS
Parameter length	Fixed, set to 1
Parameter data	Single octet indicating the required class of operation. The following values may be used : 1 = Both success and failure are reported. 2 = Only failure is reported. 3 = Only success is reported. 4 = Neither success nor failure are reported.

Parameter name	TCPN_TIMEOUT
Parameter length	Fixed, set to 2
Parameter data	The invocation time-out in seconds in the range 0 ... 1800. The first octet is the most significant byte of the time-out. <i>Note that the maximum permitted time-out value is 1800 seconds (i.e.30 minutes).</i>

Parameter name	TCPN_INVOKE_ID
Parameter length	Fixed, set to 1
Parameter data	Single octet representing the invoke ID which is in the range -128 to +127

The following table lists the parameters associated with each component request primitive and shows whether the parameter is Mandatory (M), in which case the message will be discarded if the parameter is omitted, Optional (O), in which case the parameter is not essential or Defaulted (D), in which case the parameter will be set to the configured default value by TCAP if not supplied by the user.

Parameter	ITU-T and ANSI Primitive						
	I N V O K E L	R E S U L T - N O N L	R E S U L T - N O N L	R E S U L T - N O N L	U - R E C E I V E R O R	U - C A N C E L	U - R E J E C T
Invoke ID						M	
Class	D						
Timeout	D						
Component	M	M	M	M	M		M
Last Component							

8.7 TC-COMPONENT-INDICATION

Synopsis:

Protocol message sent from TCAP to the TC-User containing a TC-Component associated with a dialogue.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_CPT_IND (0x8782)	
id	dialogue_ID	
src	TCP_TASK_ID	
dst	TC-User module id	
rsp_req	0	
hclass	0	
status	0	
err_info	0	
len	Number of bytes of user data	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	1	Component primitive type octet.
1	len - 2	Parameters in Name-Length-Data format.
len - 1	1	Set to zero indicating end of message.

Description:

This message is used by TCAP to send Component sub-layer primitives to the TC-User. On receipt of a dialogue-handling primitive from the network, TCAP first issues a dialogue handling primitive to the TC-User, this is followed by a number of component primitive messages (each containing a single component) until all the components have been conveyed to the user. The last component primitive will have the data in the 'last component' parameter set to indicate that there are no further components to follow.

All component indication primitives contain a dialogue ID which is encoded in the message header. It does not form part of the parameter area.

Parameter area contents:

The parameter area is coded as defined for the TC-COMPONENT-REQUEST message.

NOTE: The TC-NULL-IND is issued to the user in the case that a component is discarded by the TCAP module but when it is still necessary to indicate a component to the user in order to preserve the 'LAST-COMPONENT' indication.

The following table lists the parameters associated with each component indication primitive and shows whether the parameter is Mandatory (M), in which case it will always be present in messages issued by TCAP or Optional (O), in which case the parameter may or may not be present depending on the received message data or event being reported.

Parameter	ITU-T and ANSI Primitive									
	I N V O K E	I N V O K E - N L	R E S U L T - L	R E S U L T - N L	U - E R R O R	L - C A N C E L	L - R E J E C T	R - R E J E C T	U - R E J E C T	N U L L
Invoke ID						M				
Class										
Timeout										
Component	M	M	M	M	M		M	M	M	
Last Component	M	M	M	M	M	M	M	M	M	M

8.8 TC-DIALOGUE-REQUEST

Synopsis:

Protocol message sent from the TC-User to TCAP containing a dialogue-handling primitive.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_DLG_REQ (0xc783)	
id	dialogue_ID	
src	Sending module_id	
dst	TCP_TASK_ID	
rsp_req	0	
hclass	0	
status	0	
err_info	0	
len	Number of bytes of user data	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	1	Dialogue primitive type octet.
1	len - 2	Parameters in Name-Length-Data format.
len - 1	1	Set to zero indicating end of message.

Description:

This message is used by the TC-User to send component sub-layer primitives relating to dialogue handling to TCAP. The primitives cause the generation of a message for sending to the network layer which contains all the components received for the specified dialogue ID.

All dialogue request primitives contain a dialogue ID which is encoded in the message header. It does not form part of the parameter area.

Parameter area contents:

The dialogue primitive type octet is coded as defined in Section 8.4, Dialogue Primitive Types.

The following parameter names are defined for use in dialogue handling primitive messages:

Parameter	Mnemonic	Value
Quality of service	TCPN_QOS	6
Destination address	TCPN_DEST_ADDR	7
Originating address	TCPN_ORIG_ADDR	8
Termination type	TCPN_TERMINATION	9
Abort reason	TCPN_ABORT_REASON	10
Report cause	TCPN_REPORT_CAUSE	11
Components present	TCPN_CPT_PRESENT	12
Application context name	TCPN_APPL_CONTEXT	13
User information	TCPN_USER_INFO	14
P-Abort	TCPN_P_ABORT	15
User Abort Information	TCPN_UABORT_INFO	16
Security context	TCPN_SECURITY	17
Confidentiality indicator	TCPN_CONFIDENTIALITY	18
Permission to release	TCPN_PERMISSION	19

The coding for each parameter type is given in the following tables:

Parameter name	TCPN_QOS
Parameter length	Either 1, 2 or 3 octets.
Parameter data	<p>The first octet is an indicator octet, which must always be present. Subsequent octets must only be present if the appropriate bit is set in the indicator octet. The coding is as follows:</p> <p>Indicator Octet :</p> <p>bit 0 - Set to 1 if the Return Option is selected.</p> <p>bit 1 - Set to 1 if Sequence Control is required.</p> <p>bit 2 - Set to 1 if the SLS Key octet is present in the Quality of Service parameter, in which case it will be the following octet. Otherwise TCAP will generate the SLS key (for passing to SCCP) automatically and the SLS key octet is omitted.</p> <p>bit 3 - Set to 1 if the Message Priority Octet is included in the Quality of Service Parameter. Otherwise TCAP will insert the default message priority.</p> <p>All other bits are reserved for future use and must be set to zero.</p> <p>SLS Key Octet:</p> <p>The SLS Key which is used (by SCCP) to determine the SLS value to be used in the resulting message.</p> <p>Message Priority Octet:</p> <p>Coded as 0, 1, 2 or 3 to indicate the required message priority.</p> <p><i>Note: QOS indications from TCAP contain only the Indicator Octet.</i></p>

Parameter name	TCPN_DEST_ADDR
Parameter length	Variable, in the range 2 to 18
Parameter data	<p>Destination address parameter encoded in the format expected by the network layer.</p> <p>When using ITU-T SCCP, formatted in accordance with the Q.713 definition of 'Called Party Address' commencing with the address indicator and containing optionally signalling point code, sub-system number and global title).</p> <p>When using ANSI SCCP, in accordance with T1.112.3 definition of 'Called party address'.</p>

Parameter name	TCPN_ORIG_ADDR
Parameter length	Variable, in the range 2 to 18
Parameter data	<p>Originating address parameter encoded in the format expected by the network layer.</p> <p>When using ITU-T SCCP, formatted in accordance with the Q.713 definition of 'Calling Party Address' commencing with the address indicator and containing optionally signalling point code, sub-system number and global title).</p> <p>When using ANSI SCCP, in accordance with T1.112.3 definition of 'Calling party address'.</p>

Parameter name	TCPN_TERMINATION
Parameter length	Fixed, set to 1
Parameter data	<p>Single octet set to 0 to indicate BASIC end or 1 to indicate PRE-ARRANGED end.</p> <p>A BASIC end will cause a TR-END (ITU-T) or RESPONSE package (ANSI) to be sent via the network layer to the peer TCAP to terminate a transaction.</p> <p>A PRE-ARRANGED is used to inform the local TCAP that the transaction is to be terminated without exchange of messages with the peer TCAP.</p>

Parameter name	TCPN_ABORT_REASON
Parameter length	Fixed, set to 1
Parameter data	Single octet set to 1 to indicate <i>Application Context not supported</i> or 2 to indicate <i>Other user reason</i> . (Note: The absence of this parameter in an indication implies 'Other user reason')

Parameter name	TCPN_REPORT_CAUSE
Parameter length	Fixed, set to 1
Parameter data	Passed transparently from network-layer. When using SCCP coded as Q.713/T1.112.3 'Return Cause' parameter.

Parameter name	TCPN_CPT_PRESENT
Parameter length	Fixed, set to 1
Parameter data	Single octet set to 0 to indicate that there are no components to follow or 1 to indicate that there are components to follow.

Parameter name	TCPN_APPL_CONTEXT
Parameter length	Variable (subject to satisfying message length limits).
Parameter data	Application Context Name. <u>ITU-T operation</u> Encoded as specified in Q.773 commencing with the Application Context Name tag. <u>ANSI operation</u> Encoded as specified in T1.114.3, commencing with the Integer Application Context or Object Application Context tag.

Parameter name	TCPN_USER_INFO
Parameter length	Variable (subject to satisfying message length limits).
Parameter data	User information encoded as an X.208 EXTERNAL, commencing with the EXTERNAL tag. This formatting is not required when the User Information is carried in a User Abort primitive. Any format may be used in this case.

Parameter name	TCPN_P_ABORT
Parameter length	Fixed, set to 1
Parameter data	Single octet coded as follows: <u>ITU-T operation</u> 0 – Unrecognised message type 1 – Unrecognised transaction ID 2 – Badly formatted transaction portion 3 – Incorrect transaction portion 4 – Resource limitation 126 – Abnormal dialogue 127 – No common dialogue portion <u>ANSI operation</u> 1 – Unrecognised package type 2 – Unrecognised transaction portion 3 – Badly structured transaction portion 4 – Unassigned responding transaction ID 5 - Permission to release problem 6 – Resource unavailable 7 – Unrecognised dialogue portion ID 8 – Badly structured dialogue portion 9 – Missing dialogue portion 10 – Inconsistent dialogue portion 126 – Abnormal dialogue. <i>This code is caused by receipt of an invalid or inappropriate dialogue primitive from the local TC-User.</i>

Parameter name	TCPN_UABORT_INFO
Parameter length	Variable (subject to satisfying message length limits).
Parameter data	User Abort Information in any user defined format.

Parameter name	TCPN_SECURITY
Parameter length	Variable (subject to satisfying message length limits).
Parameter data	Security Context Encoded as defined in ANSI T1.114.3, starting with the Integer Security Context or Object Security Context tag.

Parameter name	TCPN_CONFIDENTIALITY
Parameter length	Variable (subject to satisfying message length limits).
Parameter data	Confidentiality identifier Encoded as defined in ANSI T1.114.3, starting with the Confidentiality Indicator tag.

Parameter name	TCPN_PERMISSION
Parameter length	Fixed, set to 1.
Parameter data	Permission to release Single octet, set to 0 or 1 to indicate the following: 0 = responding TCAP may not terminate the dialogue. 1 = responding TCAP may release the dialogue.

The following table lists the parameters associated with each dialogue request primitive and shows whether the parameter is Mandatory (M), Optional (O) or Defaulted (D).

Parameter	ITU-T Primitive					ANSI Primitive				
	U N I	B E G I N	C O N T I N U E	E N D	A B O R T	U N I	Q U E R Y	C O N V E R S A T I O N	R E S P O N S E	A B O R T
Destination Address	D	D	O ¹	O ¹	O ¹	D	D	O ¹	O ¹	O ¹
Originating Address	D	D	O ¹	O ¹	O ¹	D	D	O ¹	O ¹	O ¹
Quality of Service	D	D	D	D	D	D	D	D	D	D
Application Context	O	O	O	O	O	O	O	O	O	O
User Information	O	O	O		O	O	O	O	O	O
Termination type				D					D	
Abort reason					O					O
Report cause										
Component present										
P-Abort										
User Abort information				O	O					O
Security Context						O	O	O	O	O
Confidentiality identifier						O	O	O	O	O
Permission to release							D	D	D	D

O¹ - Indicates that this parameter may only be present if the associated primitive is issued in response to a TC-BEGIN or TC-QUERY. In all other cases, this parameter is discarded by TCAP.

8.9 TC-DIALOGUE-INDICATION

Synopsis:

Protocol message sent from the TCAP to the TC-User containing a dialogue-handling primitive.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_DLG_IND (0x8784)	
id	dialogue_ID	
src	TCP_TASK_ID	
dst	TC-User module ID	
rsp_req	0	
hclass	0	
status	0	
err_info	0	
reserved	0	
len	Number of bytes of user data	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	1	Dialogue primitive type octet.
1	len – 2	Parameters in Name-Length-Data format.
len – 1	1	Set to zero indicating end of message.

Description:

This message is used by TCAP to send component sub-layer dialogue handling primitives to the TC-User. The primitives are issued on receipt from the network of transaction related messages and are followed by any component primitives for components contained within the received message.

All dialogue indication primitives contain a dialogue ID which is encoded in the message header. It does not form part of the parameter area.

Parameter area contents:

The parameter area is coded as defined for the TC-DIALOGUE-REQUEST message.

The following table lists the parameters associated with each dialogue handling indication primitive and shows whether the parameter is Mandatory or Optional.

Parameter	ITU-T Primitive							ANSI Primitive						
	U N I	B E G I N	C O N T I N U E	E N D	A B O R T	P - A B O R T	N O T I C E	U N I	Q U E R Y	C O N V E R S A T I O N	R E S P O N S E	A B O R T	P - A B O R T	N O T I C E
Destination Address	O	O	O ¹	O ¹	O ¹			O	O	O ¹	O ¹	O ¹		
Originating Address	M	M	O ¹	O ¹	O ¹			M	M	O ¹	O ¹	O ¹		
Quality of Service	O	O	O	O	O	O		O	O	O	O	O	O	
Application Context	O	O	O	O	O			O	O	O	O	O		
User Information	O	O	O	O	O			O	O	O	O	O		
Termination type														
Abort reason					O							O		
Report cause							O							O
Component present	M	M	M	M				M	M	M	M			
P-Abort						M							M	
User Abort information					O							O		
Security Context								O	O	O	O			
Confidentiality identifier								O	O	O	O			
Permission to release									M	M				

O¹ - Indicates that this parameter may only be present if the associated primitive is issued in response to a TC-BEGIN or TC-QUERY. In all other cases, this parameter is discarded by TCAP.

9. MANAGEMENT INTERFACE

The management interface allows the user to interface with the network-layer management entity. All management messages received by the TCAP module are passed transparently to the network layer. The management messages are described in the SCCP Programmer's Manual.

Two messages are currently supported, one for management requests from the user and the other for management indications from the SCCP module. The following management primitives are supported:

Management primitives issued to SCCP:

N-STATE Request	User In Service (UIS)
N-STATE Request	User Out of Service (UOS)
N-STATE Request	User congestion
N-COORD Request	User withdrawal request (UOR)
N-COORD Response	User withdrawal grant (UOG)

Management primitives issued by SCCP:

N-STATE Indication	User In Service (UIS)
N-STATE Indication	User Out of Service (UOS)
N-COORD Indication	User withdrawal indication (UOR)
N-COORD Confirmation	User withdrawal confirmation (UOG)
N-PCSTATE Indication	Signalling point accessible
N-PCSTATE Indication	Signalling point inaccessible
N-PCSTATE Indication	Signalling point congested

10. NON-PRIMITIVE INTERFACE

In addition to the primitive interface for passing protocol messages and management messages between the TCAP module and the TC-user, the TCAP module supports a non-primitive interface for implementation-specific functionality.

The non-primitive interface is used to support requests by the user for configuration and diagnostic purposes and to allow TCAP to report protocol based and software error events to the local system management module.

This section describes the formats of all the messages used in the non-primitive interface.

When the TCAP module returns a confirmation message containing a status value the status will be one of the following:

Mnemonic	Value	Description
none	0	Success
TCPE_BAD_ID	1	Inappropriate or invalid id in request message
TCPE_BAD_MSG	5	Inappropriate or unrecognised message type .
TCPE_BAD_PARAM	6	Invalid parameters contained in message.
TCPE_NO_RESOURCES	7	Insufficient internal resources.

10.1 TCAP Configuration Request

Synopsis:

Message used to configure the TCAP module for operation.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_CONFIG (0x7780)	
id	0	
src	Sending module_id	
dst	TCP_TASK_ID	
rsp_req	used to request a confirmation	
hclass	0	
status	0	
err_info	0	
len	40	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	1	cnf_ver - must be set to zero
1	1	Reserved, must be set to zero.
2	1	user_id - TC-User module id
3	1	nsap_id - Network-layer module id
4	1	mngt_id – Management module id
5	1	maint_id – Maintenance module id
6	2	flags
8	2	nog_dialogues
10	2	nic_dialogues
12	2	num_invokes
14	2	num_components
16	2	base_ogdlg_id
18	2	base_icdlg_id
20	1	tid_ninst
21	1	tid_ndref
22	1	tid_nseq
23	1	tcap_instance
24	2	max_data
26	2	dlg_hunt
28	1	addr_format
29	11	Reserved for future use, must be set to zero

Description:

This message is used to configure the TCAP module for operation. It should be the first message sent to the module (any messages received before a valid configuration message will be discarded) and should only be issued once.

The message contains parameters relating to the environment in which the TCAP module is operating such as the identity of other modules with which it needs to communicate. It also contains run-time options and run-time dimensioning information.

Confirmation Message:

The module sending the message can optionally request that a confirmation is returned by the TCAP module when the message has been processed. This is achieved by setting the sending layer's bit in the **rsp_req** field which will cause a confirmation message of the same format to be returned. The **status** field in this message is zero on success or an error code otherwise.

Parameter Description:

user_id

8-bit module identifier defining the default destination for all dialogue and component indications issued by TCAP to the user application program. It is also possible to configure this parameter on a per local sub-system basis (see TCP_MSG_S_TCU_ID).

nsap_id

(network service application part layer id) 8 bit module identifier defining the destination for all transmit requests made by TCAP to the SS7 network. This will normally be set to the module id of SCCP (SCP_TASK_ID).

mngt_id

8-bit module identifier defining the destination for all management indications and trace information (TCP_MSG_ERROR_IND, MGT_MSG_TRACE_EV) from TCAP to the user management entity. In a system that has no separate management task, this may be set to the same value as user-id (above).

maint_id

8-bit module identifier defining the destination for all maintenance indications (TCP_MSG_MAINT_IND) from TCAP to the user management entity. In a system that has no separate management task, this may be set to the same value as user-id (above).

flags

A 16 bit value, each bit having a particular meaning as detailed in the table below:

Bit	Description
0	Set to 1 to trace discarded TC User primitives to management
1	Selects between ITU-T and ANSI PDU formats. Set to 1 to select ANSI T1.114 format, 0 to select ITU-T Q.773.
2	If set, prevents the Invoke timer from being started for each Invoke operation issued by this module.
3	If set to 1, enables the result of an operation to be conveyed in an Invoke component. If set to zero, receipt of an Invoke in response to an Invoke will be treated as an abnormal condition.
4	If set to 1, unsolicited result components are passed on to the TC-User. If set to zero, unsolicited result components are rejected.
5	If set to 1, transmit component type discrimination is disabled for TC-UNI requests. If not set, TCAP will only allow a class 4 Invoke/Invoke-L and Invoke-NL to be sent with a TR-UNI. Any other accumulated component types will be discarded.
6	If set to 1, the local sub-system number is not recovered from the destination address for incoming dialogues. All primitives will be delivered to the module 'user_id' specified by the TCAP Configuration Request.
7	If set to 1 operation with dialogue groups is enabled, otherwise configuration of dialogue groups will be rejected by TCAP.
Other	All other bits are reserved for future use and must be set to zero.

The combined value of the flags option can be constructed by using a combination of the mnemonics defined in TCP_INC.H and listed below.

Value	Mnemonic	Description
0x0001	TCPF_TR_DISC	Trace discarded primitives to mngt
0x0002	TCPF_ANSI	Select ANSI PDU formats if set
0x0004	TCPF_DIS_TINV	Disable Invoke timer
0x0008	TCPF_INVRES	Allow an Invoke to convey a result
0x0010	TCPF_DIS_RXFILT	Disable filtering of Rx component type
0x0020	TCPF_DIS_TXFILT	Disable filtering of Tx component type
0x0040	TCPF_NO_RXUID	Disable Rx of destination user id by ssn
0x0080	TCPF_DLGRP	Enable dialogue groups

nog_dialogues

The maximum number of simultaneous outgoing dialogues that the module is required to support. This value is compared with a compile time constant to ensure that the module has sufficient internal resources to handle the requested maximum number of outgoing dialogues.

nic_dialogues

The maximum number of simultaneous incoming dialogues that the module is required to support. This value is compared with a compile time constant to ensure that the module has sufficient internal resources to handle the requested maximum number of incoming dialogues.

num_invokes

The maximum number of simultaneous (outgoing) invocations that the module is required to support. This value is compared with a compile time constant to ensure that the module has sufficient internal resources to handle the requested number of simultaneous invokes.

num_components

The maximum number of buffered components required to be supported by the module at any one time. This value is compared with a compile time constant to ensure that the module has sufficient internal resources to support the requested number of buffered components.

base_ogdlg_id

The first dialogue ID for outgoing dialogues that the user wishes to be handled by this module. The subsequent (**nog_dialogues** - 1) dialogue ID's will also be handled by the module. The user must ensure that the values used in the dialogue ID field of all protocol messages pertaining to outgoing dialogues lie within the correct range.

base_icdlg_id

The first dialogue ID for incoming dialogues that the user wishes to be handled by this module. The subsequent (**nic_dialogues** - 1) dialogue ID's will also be handled by the module. The most significant bit (i.e. bit 15) of the dialogue ID must be set to one for incoming dialogues. The TCAP module allocates the dialogue ID for each incoming dialogue. It uses values in the range **base_icdlg_id** (**base_icdlg_id** + **nic_dialogues** - 1) for this purpose.

tid_ninst, tid_ndref, tid_nseq

TCAP generates a local transaction ID (ltid) for each transaction. The ltid is made up of three fields, each of variable width as configured by the user. The fields are **tcap_instance**, **dialogue_reference**, and **sequence_number**.

tcap_instance is as supplied by the user in the configuration message, the field occupies **tid_ninst** bits. (tid_ninst may be set to zero if required).

dialogue_reference is generated internally by the TCAP module and ranges from 0 up to one less than the total number of dialogues supported. The field occupies **tid_ndref** bits. (tid_ndref must be large enough to allow coding of the dialogue_reference).

tid_nseq is a sequence number assigned (by the TCAP module) to each transaction pertaining to a particular dialogue ID. It ensures that the maximum possible time interval elapses before the transaction ID is re-allocated. The field occupies **tid_nseq** bits.

tcap_instance

The TCAP module supports the use of multiple instances of TCAP for a particular application to allow for distributed processing across multiple hardware platforms. When this feature is used each instance handles messages relating to a particular set of transaction ID's. To ensure correct operation each instance of TCAP must be configured with a unique **tcap_instance** value.

max_data

The maximum length for the user data parameter in messages sent to the network layer. This will usually be set to 255 (i.e. the maximum permitted length) however the user can restrict the maximum length if required. The user is responsible for ensuring that the components issued to TCAP do not exceed the configured limit otherwise TCAP will discard the message. TCAP uses the configured limit when generating reject components to determine if there is sufficient space to include the reject component or whether it is necessary to store the reject component for transmission in the next message.

dlg_hunt

Defines how incoming transactions are distributed between dialogue groups. If operation with dialogue id groups is not enabled this parameter is ignored by TCAP. This parameter should be set to 0, 1 or 2 as shown below:

0 - Cyclic Selection. Each new incoming dialogue is allocated to the next TCAP group.

1 - Load Balanced Selection. Each new incoming dialogue is allocated to the group with the least number of active incoming dialogues.

2 - Sequential Selection. Each new incoming dialogue is allocated to the group containing the first inactive incoming dialogue_id.

addr_format

Defines how TCAP should interpret address information from messages received from SCCP in order to direct received TCAP primitives to unique SCCP sub-systems (TCAP user applications). The following table shows the values that may be specified for this parameter.

addr_format	Action
0	If configured to use ITU-T PDU formats (options bit 1 not set) use ITU-T Q.713 SCCP address format If configured to use ANSI PDU formats (options bit 1 set) use ANSI T1.112 SCCP address format
1	Use ITU-T Q.713 SCCP address format (14 bit point codes)
2	Use ITU-T Q.713 SCCP address format modified for 24 bit point codes
3	Use ANSI T1.112 SCCP address format modified for 14 bit point codes
4	Use ANSI T1.112 SCCP address format (24 bit point codes)

10.2 Configure Dialogue Group Request

Synopsis:

This message configures a dialogue id group, and will only be accepted by the TCAP module if the TCPF_DLGRP configuration flags bit is set.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_CNF_DGRP (0x7785)	
id	dialogue group ID	
src	Sending module_id	
dst	TCP_TASK_ID (0x14)	
rsp_req	Used to request a confirmation	
class	0	
status	0	
err_info	0	
len	16	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	2	base_ogdlg_id
2	2	nog_dialogues
4	2	base_icdlg_id
6	2	nic_dialogues
8	2	options
10	2	user_instance
12	4	reserved (must be set to zero)

Description:

This message is used to configure a TCAP dialogue group.

Confirmation Message:

The module sending the message can optionally request that a confirmation is returned by the TCAP module when the message has been processed. This is achieved by setting the sending layer's bit in the **rsp_req** field which will cause a confirmation message of the same format to be returned. The **status** field in this message is zero on success or an error code otherwise.

Parameter Description:

Dialogue Group ID

a logical identifier for this group, the valid range being 0 to 31.

base_ogdlg_id

The first outgoing dialogue id assigned to this dialogue identifier group.

nog_dialogues

The number of outgoing dialogues assigned to this group, hence outgoing dialogue ids base_ogdlg_id to base_ogdlg_id + nog_dialogues-1 are assigned to this group.

base_icdlg_id

The first incoming dialogue id assigned to this dialogue identifier group.

nic_dialogues

The number of incoming dialogues assigned to this group, hence outgoing dialogue ids base_ogdlg_id to base_icdlg_id + nic_dialogues-1 are assigned to this group.

options

Should currently be set to zero.

user_instance

Identifies the instance of the user application that the defined ranges of dialogues will be sent to.

The number of dialogues must lie within the limit specified with the TCAP Configuration request message.

10.3 Configure TC-User Request

Synopsis:

This message configures a `module_id` in the message passing environment to be used by TCAP to distribute primitive indications to different local sub-systems.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_S_TCU_ID (0x5793)	
id	sub-system number	
src	Sending module_id	
dst	TCP_TASK_ID (0x14)	
rsp_req	Used to request a confirmation	
class	0	
status	0	
err_info	0	
len	1	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	1	module_id

Description:

This message is used to set the module id to be used by TCAP as the destination for dialogue and component indications for incoming dialogues (dialogues initiated from a remote signalling point).

Confirmation Message:

The module sending the message can optionally request that a confirmation is returned by the TCAP module when the message has been processed. This is achieved by setting the sending layer's bit in the **rsp_req** field which will cause a confirmation message of the same format to be returned. The **status** field in this message is zero on success or an error code otherwise.

Parameter Description:

sub-system number

The sub-system number for the local sub-system in the range 0 to 255. This is the value that will be matched against the sub-system number in the called address parameter in messages received by TCAP from SCCP (the network transport layer).

module_id

The module id used by the users own application program that will receive dialogue and component indications for the *local sub-system* described by the sub-system number.

10.4 TCAP Set Default Parameters Request

Synopsis:

Message used to set up default protocol parameters for use by the TCAP module.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_DEF_PARAM (0x7781)	
id	0	
src	Sending module_id	
dst	TCP_TASK_ID	
rsp_req	used to request a confirmation	
hclass	0	
status	0	
err_info	0	
len	Number of bytes in parameter area	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	len - 1	Parameters in Name-Length-Data format.
len - 1	1	Set to zero indicating end of message.

Description:

This message is used to set up default protocol parameters for use by the TCAP module. The message may be issued at any time after the module configuration message and may be used to change default parameters.

The parameter area may contain any of the component parameters or dialogue handling parameters which can take a default value. The parameters are encoded in the same way as in the TC-COMPONENT-REQUEST and TC-DIALOGUE-REQUEST messages in Name-Length-Data format.

Confirmation Message:

The module sending the message can optionally request that a confirmation is returned by the TCAP module when the message has been processed. This is achieved by setting the sending layer's bit in the **rsp_req** field which will cause a confirmation message of the same format to be returned. The **status** field in this message is zero on success or an error code otherwise.

Default Values:

The parameters in the following tables can be assigned default values. The tables also show the initial default values (were applicable):

Parameter name	TCPN_CLASS
Default value	Class 1 operation.

Parameter name	TCPN_TIMEOUT
Default value	5 seconds.

Parameter name	TCPN_QOS
Default value	Return option not selected Sequence control not required Default message priority (= 2)selected.

Parameter name	TCPN_DEST_ADDR
Default value	Not assigned

Parameter name	TCPN_ORIG_ADDR
Default value	Not assigned

Parameter name	TCPN_TERMINATION
Default value	BASIC end

Parameter name	TCPN_PERMISSION
Default value	1 = responding TCAP may release the dialogue.

10.5 Read TCAP Statistics Request

Synopsis:

Message used to read the statistics maintained by the TCAP module.

Note: This message is reserved for future use

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_R_STATS (0x6792)	
id	0	
src	0	
dst	TCP_TASK_ID	
rsp_req	Sending layer's bit must be set	
hclass	0	
status	1 to reset all statistics, otherwise 0	
err_info	0	
len	Reserved for future use	
PARAMETER AREA		
OFFSET	SIZE	NAME
0		Reserved for future use

Description:

The TCAP module automatically maintains a number of counters to record the number of occurrences of particular protocol events. The values of the counters can be read using this message. The counters can optionally be reset to zero following the read operation.

If the **status** field is set to 1 the counters will all be reset. If it is not required to reset the counters then the **status** field should be set to zero and the counters will continue to accumulate from the current count after the read operation.

Confirmation Message:

The module sending the message must set the sending layer's bit in the **rsp_req** field to cause a confirmation message containing the statistics to be returned.

10.6 Read TCAP RAM Request

Synopsis:

Message used for diagnostic purposes to return the address of the TCAP modules internal data storage area.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_R_RAM (0x6790)	
id	0	
src	Sending module_id	
dst	TCP_TASK_ID	
rsp_req	Sending layer's bit must be set	
hclass	0	
status	0	
err_info	0	
len	4	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	4	Base address of TCAP module's global RAM structure written by TCAP module in response message.

Description:

This message is provided solely for diagnostic purposes to allow the user to locate the base address of the TCAP modules internal data structure.

Confirmation Message:

The module sending the message must set the sending layer's bit in the **rsp_req** field to cause a confirmation message containing the address to be returned.

10.7 Read TCAP Dialogue Request

Synopsis:

Message used for diagnostic purposes to return the address of the internal dialogue resource structure in the TCAP module.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_R_DLG (0x6791)	
id	dialogue_ID	
src	Sending module_id	
dst	TCP_TASK_ID	
rsp_req	Sending layer's bit must be set	
hclass	0	
status	0	
err_info	0	
len	4	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	4	Base address of internal dialogue resource structure written by TCAP module in response message.

Description:

This message is provided solely for diagnostic purposes to allow the user to locate internal data structures within the TCAP module.

Confirmation Message:

The module sending the message must set the sending layer's bit in the **rsp_req** field to cause a confirmation message containing the address to be returned.

10.8 Read TCAP Module Status Request

Synopsis:

Message used to read the status of the TCAP module and its associated resources.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_R_MOD_STATUS (0x6796)	
id	0	
src	Sending module_id	
dst	TCP_TASK_ID	
rsp_req	Sending layer's bit must be set	
class	0	
status	0	
err_info	0	
len	40	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	1	version - must be set to zero.
1	4	num_ic_dlg - Number of active incoming dialogues
5	4	num_og_dlg - Number of active outgoing dialogues
9	4	num_act_invokes - Number of active invokes
13	4	num_alloc_cpt - Number of allocated CPTs
18	4	num_alloc_dbuf - Number of allocated DBUFs
20	19	reserved

Description:

This message allows the user to read the dialogue usage statistics for TCAP module. The user should send the message with the version initialised as shown above and all other fields set to zero. The TCAP module automatically maintains a number of counters to record the number of each resource allocated. The message returned by the TCAP module will contain a snapshot of the status of the module.

version

Only version zero supported.

num_ic_dlg

Number of active incoming dialogues. These are dialogues that are initiated by the remote node.

num_og_dlg

Number of active outgoing dialogues. These are dialogues that are initiated by the local node.

num_act_invokes

Number of active invokes. An invoke structure is stored for each invoke sent and is not required for incoming invokes.

num_alloc_cpt

Number of allocated component structures. These are used temporarily for pending component requests until an appropriate dialogue request is received.

num_alloc_dbuf

Number of allocated dialogue buffers. These are used temporarily for building dialogue request messages from pending components.

Confirmation Message:

The module sending the message must set the sending layer's bit in the **rsp_req** field to cause a confirmation message containing the statistics to be returned.

10.9 Read TCAP Dialogue Status Request

Synopsis:

Message used to read the status of an individual dialogue in the TCAP module.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_R_DLG_STATUS (0x6797)	
id	dialogue id	
src	Sending module_id	
dst	TCP_TASK_ID	
rsp_req	Sending layer's bit must be set	
class	0	
status	0	
err_info	0	
len	40	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	1	version - must be set to zero.
1	1	DHA_state – dialogue handler state
2	1	TSM_state – dialogue transaction state machinestate
3	1	DLG_state – dialogue control structure state
4	4	num_invokes - number of active invokes in dialogue
8	4	ltid – local transaction id
12	4	rtid – remote transaction id
16	24	reserved

Description:

This message allows the user to read the state of an individual dialogue within the TCAP module. The message should be issued by the user with the version initialised as shown above including setting the dialogue id in the message header id field. All other fields should be set to zero. The message returned by the TCAP module will contain all the appropriate information.

version – only version zero supported.

DHA_state – dialogue handler state.

See the TCAP specifications for details of the meaning of the states (Q.773).

State	Value
DHA_S_IDLE	0
DHA_S_INIT_RXD	1
DHA_S_INIT_SENT	2
DHA_S_ACTIVE	3

TSM_state – dialogue transaction state machine state.

See the TCAP specifications for details of the meaning of the states (Q.773).

State	Value
TSM_S_IDLE	0
TSM_S_INIT_RXD	1
TSM_S_INIT_SENT	2
TSM_S_ACTIVE	3

DLG_state – main dialogue control structure state.

The value of this state variable shows if the dialogue is in use. The dialogue is in the DLG_CPT_PENDING state whilst accumulating components at the start of a dialogue. The DLG_S_DHA_ACTIVE state shows that the dialogue is fully active and the DLG_S_PENDING_ISM state indicates that the dialogue is waiting for invoke state machines to finish.

State	Value
DLG_S_FREE	0
DLG_S_CPT_PENDING	1
DLG_S_DHA_ACTIVE	2
DLG_S_PENDING_ISM	3

num_cpts - number of active components in dialogue

ltid – local transaction id for associated dialogue

rtid – remote transaction id for associated dialogue

Confirmation Message:

The module sending the message must set the sending layer's bit in the **rsp_req** field to cause a confirmation message containing the statistics to be returned.

10.10 Maintenance Event Indication

Synopsis:

Message used by TCAP to indicate a protocol-related event to the local maintenance module.

Message Format:

MESSAGE HEADER	
FIELD NAME	MEANING
type	TCP_MSG_MAINT_IND (0x07a1)
id	See below
src	TCP_TASK_ID
dst	Maintenance module id (maint_id)
rsp_req	Sending layer's bit must be set
hclass	0
status	Maintenance event code (see below)
err_info	0
len	0

Description:

This message is used by TCAP to indicate a protocol-related event to the maintenance module.

Maintenance event code:

The **Maintenance event code** contained in the **status** field of the message indicates the type of event. Possible values are listed in the following table which also lists the meaning of the **id** field in each case:

Mnemonic	Code	id	Description
TCPEV_CPT_REQ_DISCARD	1	dialogue_ID	Component request primitive discarded due to bad format, inappropriate state or lack of internal resources.
TCPEV_DLG_REQ_DISCARD	2	dialogue_ID	Dialogue request primitive discarded due to bad format, inappropriate state or lack of internal resources.
TCPEV_DATA_LEN_ERR	3	dialogue_ID	Message discarded due to exceeding maximum length for user data.
TCPEV_UNREC_TYPE	4	0	Unrecognised TCAP message type received.
TCPEV_UNREC_TID	5	0	Message received relating to unknown local transaction ID.
TCPEV_SYNTAX_ERR	6	0	Syntax error in transaction portion of received message.
TCPEV_BAD_REJ_RXD	7	dialogue_ID	Badly formatted reject component received.

10.11 Software Event Indication

Synopsis:

Message used by TCAP to indicate an implementation specific software related event to the local management module.

Message Format:

MESSAGE HEADER	
FIELD NAME	MEANING
type	TCP_MSG_ERROR_IND (0x07a2)
id	See below
src	TCP_TASK_ID
dst	Management module id (mngt_id)
rsp_req	Used to request a confirmation
hclass	0
status	Software event code (see below)
err_info	0
len	0

Description:

This message is issued by the TCAP module to notify system management of various software events which under normal operating conditions should not occur. These events may be due to lack of system resources or errors within the software.

Software event code

The **Software event code** contained in the **status** field of the message indicates the type of event. Possible values are listed in the following table which also lists the meaning of the **id** field in each case.

Mnemonic	Code	id	Description
TCPSWE_NO_TCPT	1	0	Internal component resources exhausted.
TCPSWE_NO_TISM	2	0	Maximum number of active invocations exceeded.
TCPSWE_NO_DLG	3	0	No internal resource to handle dialogue.
TCPSWE_NO_TCPM	4	0	Internal pool of structured messages exhausted.
TCPSWE_TCPM_LOW	5	0	Internal pool of structured messages running low.
TCPSWE_BAD_MSG	6	message type	Unrecognised inter task message received.
TCPSWE_TX_FMT_ERR	7	0	Internal error during message formatting.
TCPSWE_ISM_ERR	8	0	Internal error in invocation state machine.
TCPSWE_BAD_NSAP_FMT	9	0	Badly formatted message received from network layer.
TCPSWE_DBUF_LOW	10	0	TCAP is running short of resources to accumulate components.
TCPSWE_NO_DBUF	11	0	No more resources are available to accumulate component requests
TCPSWE_DBUF_ABMT	12	0	Number of resources available for component accumulation has recovered.

10.12 TCAP Dialogue discard indication

Synopsis:

If a dialogue request message is discarded by the TCAP module the following message is used to send useful information to the TCAP management module.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_DIS_DLG_IND (0x07a3)	
id	0	
src	TCAP_TASK_ID	
dst	Management module id (mngt_id)	
rsp_req	0	
hclass	0	
status	Message discard code (see below)	
err_info	0	
len	length of discarded message	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	len	Trace of discarded message

Description:

The parameter area of the message contains the parameter area of the message that has been discarded. The status field shows the message discard code that indicates the reason the message was discarded.

Message discard code:

The message discard event code contained in the status field of the message indicates the reason for the discard of the dialogue or component message. Possible values are listed in the following table:

Mnemonic	Code	Description
TCPPD_UNREC_ID	1	Dialogue Id is not in configured range
TCPPD_INACTIVE_DLG	2	Specified dialogue is inactive
TCPPD_BAD_FORMAT	3	Primitive request formatted incorrectly
TCPPD_UNEX_PRIM	4	Primitive request not allowed in current state
TCPPD_INVALID_PRO	5	Primitive request not allowed by protocol rules
TCPPD_INTERNAL_FAIL	6	Internal failure
TCPPD_INVALID_REQ_TYPE	7	Component or Dialogue primitive type is invalid
TCPPD_CPT_SYNTAX_ERR	8	Syntax error (e.g. missing ASN parameter)
TCPPD_CPT_ENCODE_ERR	9	Encoding error (e.g. bad length)
TCPPD_CPT_UNREC_TYPE	10	Unrecognised ASN type in context
TCPPD_CPT_MISSING_PARAM	11	Missing mandatory parameter

10.13 TCAP Component discard indication

Synopsis:

If a component request message is discarded by the TCAP module the following message is used to send useful information to the TCAP management module.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_DIS_CPT_IND (0x07a4)	
id	0	
src	TCAP_TASK_ID	
dst	Management module id (mngt_id)	
rsp_req	0	
hclass	0	
status	Message discard code (See table of codes in TCP_MSG_DIS_DLG_IND message definition)	
err_info	0	
len	length of discarded message	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	len	Message Trace of discarded message

Description:

The parameter area of the message contains the parameter area of the message that has been discarded. The status field shows the message discard code that indicates the reason the message was discarded.

10.14 Read Revision Request

Synopsis:

Message used to request the module type and software revision number.

Message Format:

MESSAGE HEADER			
FIELD NAME		MEANING	
type		GEN_MSG_MOD_IDENT (0x6111)	
id		0	
src		Originating module ID	
dst		TCAP_TASK_ID	
rsp_req		Sending layer's bit must be set	
hclass		0	
status		0	
err_info		0	
len		28	
PARAMETER AREA			
OFFSET	SIZE	NAME	
0	2	type	Currently undefined
2	1	maj_rev	Major version number
3	1	min_rev	Minor version number
4	24	text	Null terminated string giving textual module identity

Description:

This message is provided to request a reply indicating the software version for module under test. The parameter areas are filled in by the TCAP module and do not need to be included by the user. On receipt of this request the module returns the message with status "SUCCESS" to the sender including the information requested.

10.15 Management Event Indication

Synopsis:

This message is issued by the TCAP module to notify system management of general software events that under normal operating conditions should not occur. These events may be due to lack of system resources or errors within the software.

Message Format:

MESSAGE HEADER	
FIELD NAME	MEANING
type	MGT_MSG_EVENT_IND (0x0008)
id	0
src	TCAP_TASK_ID
dst	Management module id (mngt_id)
rsp_req	0
hclass	0
status	Management event code (see below)
err_info	Time-stamp
reserved	0
len	0

Management event code

The **Management event code** contained in the **status** field of the message indicates the type of event. Possible values are listed in the following table which also lists the meaning of the id field in each case.

Mnemonic	Value		id	Description
ERR_SDLSIG_LOW	47	0x2f	0	The internal signal queue is running short of entries. If this fault persist the software should be re-built with more signals allocated to the signal queue.
ERR_NO_SDLSIG	46	0x2e	0	The internal signal queue has been exhausted. If this event occurs then correct operation of the module is not guaranteed.

10.16 Set Trace Mask Request

Synopsis:

Message sent to TCAP to trace primitives exchanged between TCAP and the TC-User and/or SCCP.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TCP_MSG_TRACE_MASK (0x5795)	
id	0	
src	Sending module_id	
dst	TCP_TASK_ID (0x14)	
rsp_req	Used to request a confirmation	
hclass	0	
status	0	
err_info	0	
len	12	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	4	op_evt_mask
4	4	ip_evt_mask
8	4	mng_evt_mask

This message causes a copy of transmit or receive TCAP primitives to be taken and sent to the mngt_id specified in the TCAP Configuration request, facilitating the examination of the raw transaction messages for diagnostic purposes. The format of the traced data is given in the section 'Trace Event Indication'.

The events traced are specified by setting bits in the three event masks as shown below. In each case, bit 0 is the least significant bit:

op_evt_mask

- bit 0 Trace all Dialogue Indications to the TC-User
- bit 1 Trace all Component Indications to the TC-User
- bit 2 Trace all UDT Request sent by TCAP to SCCP
- all other bits must be set to zero

ip_evt_mask

bit 0 Trace all Dialogue Requests from the TC-User
bit 1 Trace all Component Request from the TC-User
bit 2 Trace all UDT Indications received from SCCP
bit 3 Trace all UDTS Indications received from SCCP
all other bits must be set to zero

mgmt_evt_mask

none currently defined, all bits must be set to zero.

10.17 Trace Event Indication

Synopsis:

The TCAP module may be configured to report to management primitives exchanged with the TC-User and SCCP. This is useful for trace and debug purposes. Tracing is enabled by specifying individual bits in trace masks in the set trace masks request message issued to TCAP. The traced primitives are reported as event indications as shown below:

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	MGT_MSG_TRACE_EV (0x0003)	
id	0	
src	TCP_TASK_ID (0x14)	
dst	Management module id (mgmt_id)	
rsp_req	0	
hclass	0	
status	0	
err_info	0	
len	18 + length of traced data	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	1	source module id
1	1	destination module id
2	2	id
4	2	type
6	2	status
8	4	timestamp
12	4	pointer to the message being traced
16	2	data length
18	0 .. 280	data – Data taken from the MSG parameter area.

APPENDIX A

A.1 Timer Services

The notion of time in the TCAP module is based on a periodic timer tick received from every 100ms. This 'tick' is used to run all TCAP protocol timers. This appendix details the messages that are used by the TCAP module to control timer services.

A.2 Keep Time

Synopsis:

This message is issued by TCAP to request the timer module to issue a periodic timer tick (TM_EXP) message to the TCAP module.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	KEEP_TIME (0x7006)	
id	0	
src	TCAP module id (TCP_TASK_ID)	
dst	Timer module ID (0x00)	
rsp_req	0	
hclass	0	
status	0	
err_info	0	
len	6	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	4	Reserved, should be set to zero if issued by the user and are discarded when received by the timer module
4	2	Resolution

Parameter Description:

resolution

The number of operating system ticks between timer expiry messages being issued to the TCAP module. This parameter is set from the **timer_res** parameter in the TCAP module configuration message.

A.3 Timer Expiry

Synopsis:

Periodic timer tick message issued by the timer module.

Message Format:

MESSAGE HEADER		
FIELD NAME	MEANING	
type	TM_EXP (0xc002)	
id	index of timer in table	
src	Timer module ID (0x00)	
dst	TCAP module id (TCP_TASK_ID)	
rsp_req	0	
hclass	0	
status	0	
err_info	0	
len	4	
PARAMETER AREA		
OFFSET	SIZE	NAME
0	4	reserved – must be set to zero

All application messages contain a common header which is used to determine the message type, the source and destination module identities and status information. This header structure is defined in 'C' as follows, the meaning of each field is also described:

APPENDIX B

B.1 Message Type reference

The following table provides a reference of all the message types used by the TCAP module.

Value	Mnemonic	Description
0x07a1	TCP_MSG_MAINT_IND	Maintenance event indication
0x07a2	TCP_MSG_ERROR_IND	Software event indication
0x2790		Confirmation to TCP_MSG_R_RAM (0x6790)
0x2791		Confirmation to TCP_MSG_R_DLG (0x6791)
0x2792		Confirmation to TCP_MSG_R_STATS (0x6792)
0x2796		Confirmation to TCP_MSG_R_MOD_STATUS (0x6796)
0x2797		Confirmation to TCP_MSG_R_DLG_STATUS (0x6797)
0x3780		Confirmation to TCP_MSG_CONFIG (0x7780)
0x3781		Confirmation to TCP_MSG_DEF_PARAM (0x7781)
0x3785		Confirmation to TCP_MSG_CNF_DGRP (0x7785)
0x5793	TCP_MSG_S_TCU_ID	Set local TC-User/sub-system module id
0x5795	TCP_MSG_TRACE_MASK	Set TCAP trace mask
0x6790	TCP_MSG_R_RAM	Read address of internal ram structure (debug)
0x6791	TCP_MSG_R_DLG	Read dialogue address (debug only)
0x6792	TCP_MSG_R_STATS	Read global statistics
0x6796	TCP_MSG_R_MOD_STATUS	Read module resource status
0x6797	TCP_MSG_R_DLG_STATUS	Read dialogue resource status
0x7006	KEEP_TIME	Request timer services from TCAP
0x7780	TCP_MSG_CONFIG	Module configuration message
0x7781	TCP_MSG_DEF_PARAM	Set default parameters
0x7785	TCP_MSG_CNF_DGRP	Configure dialogue id group
0x8002		Confirmation to TM_EXP (0xc002)
0x8740		Confirmation to SCP_MSG_TX_REQ (0xc740)
0x8741		Confirmation to SCP_MSG_RX_IND (0xc741)
0x8744		Confirmation to SCP_MSG_SCMG_REQ (0xc744)
0x8745	SCP_MSG_SCMG_IND	SCCP management indication to TC-User
0x8781		Confirmation to TCP_MSG_CPT_REQ (0xc781)
0x8782	TCP_MSG_CPT_IND	Component indication from TCAP
0x8783		Confirmation to TCP_MSG_DLG_REQ (0xc783)
0x8784	TCP_MSG_DLG_IND	Dialogue indication to TC-user
0xc002	TM_EXP	Timer expiry (tick) to TCAP

0xc740	SCP_MSG_TX_REQ	Transmit request from TCAP to SCCP
0xc741	SCP_MSG_RX_IND	Receive indication from SCCP to TCAP
0xc744	SCP_MSG_SCMG_REQ	SCCP management request from User
0xc781	TCP_MSG_CPT_REQ	Component request from TC-User
0xc783	TCP_MSG_DLG_REQ	Dialogue request from TC-User

The reader is also reminded that if a confirmation is requested for a request sent to the TCAP module, the confirmation consists of the original message with the message type modified by resetting bit 14 to 0. Hence the confirmation to a message type 0x7780 sent to TCAP will be received by the application as type 0x3780.