BLUETOOTH DOC	Date / Year-Month-Day 2003-05-22	Approved Version	Revision 1.00	Document No
Prepared Bluetooth Audio Video Working Group	e-mail address avv-feedback@bluetooth.org			N.B. Confidential

AUDIO/VIDEO CONTROL TRANSPORT PROTOCOL SPECIFICATION

VERSION 1.0 ADOPTED

Abstract

This specification describes the Audio/Video Control Transport Protocol (AVCTP), which is used to transport command and response messages for controlling Audio Video features in conformant devices. This protocol enables a device to support more than one control profile at the same time; each supported profile shall define its own message formatting and/or usage rules.

Release Date: 22 May 2003

Revision History

Revision	Date	Comments
0.5	April 2001	Release to Associates
0.7	June 2001	Release to Associates
0.9	September 2001	Release to Associates and Early Adopters
Draft 0.95	October 2001	Release to Associates and Early Adopters
Draft 0.95a	March 2002	Release after Adoption Review
Draft 1.00	May 2002	Release for Voting Draft
Draft 1.00a	February 2003	Release for Voting Draft
Version 1.0	May 2003	Updated title and header

Contributors

Billy Brackenridge	Microsoft
Kalervo Kontola	Nokia
Vesa Lunden	Nokia
Jurgen Schnitzler	Nokia
Shaun Barrett	Philips
Christian Bouffioux	Philips
Rob J. Davies	Philips
Olivier Hus	Philips
Geert Knapen	Philips
Emmanuel Mellery (Owner)	Philips
Marc Vauclair	Philips
Masakazu Hattori	Sony
Harumi Kawamura	Sony
Ruediger Mosig	Sony
Junko Ami	Toshiba
Yoshiaki Takabatake	Toshiba
Ichiro Tomoda	Toshiba

Disclaimer and Copyright Notice

Document Terminology

The Bluetooth SIG has adopted Section 13.1 of the IEEE Standards Style Manual, which dictates use of the words ``shall'', ``should'', ``may'', and ``can'' in the development of documentation, as follows:

- The word *shall* is used to indicate mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (*shall* equals *is required to*).
- The use of the word *must* is deprecated and shall not be used when stating mandatory requirements; *must* is used only to describe unavoidable situations.
- The use of the word *will* is deprecated and shall not be used when stating mandatory requirements; *will* is only used in statements of fact.
- The word *should* is used to indicate that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain course of action is deprecated but not prohibited (*should* equals *is recommended that*).
- The word *may* is used to indicate a course of action permissible within the limits of the standard (*may* equals *is permitted*).
- The word *can* is used for statements of possibility and capability, whether material, physical, or causal (*can* equals *is able to*).

Contents

1 Introduction

1.1 Purpose

The Bluetooth™ Audio/Video Control Transport Protocol specification, hereafter referred to as AVCTP, describes the transport mechanisms used to exchange messages for controlling Audio and/or Video devices. The actual message contents are defined in the applicable A/V control profiles.

1.2 Scope

AVCTP is used to transport the command/response messages exchanged for the control of distant A/V devices over point-to-point connections. AVCTP does not describe the format and coding of the command and/or response frames used to control an A/V device; command and/or response format and coding rules are specified by the related control profiles.

The AVCTP transaction model is defined in Section 3 - AVCTP Generic Transaction Model.

1.3 Definitions

1.3.1 Forbidden

This bit field combination is not allowed in the specification. The receiver shall check that this bit field combination is not used.

1.3.2 RFA

Reserved for Future Additions. Bits with this designation shall be set to zero. Receivers shall ignore these bits.

1.3.3 RFD

Reserved for Future Definition. These bit value combinations or bit values are not allowed in the current specification but may be used in future versions. The receiver shall check that unsupported bit value combination is not used.

2 Overview

2.1 Operations Between Devices

The Audio/Video Distribution Control Protocol (hereafter referred to as AVCTP) defines the binary transactions issued between a pair of Bluetooth devices for A/V function discovery and control.

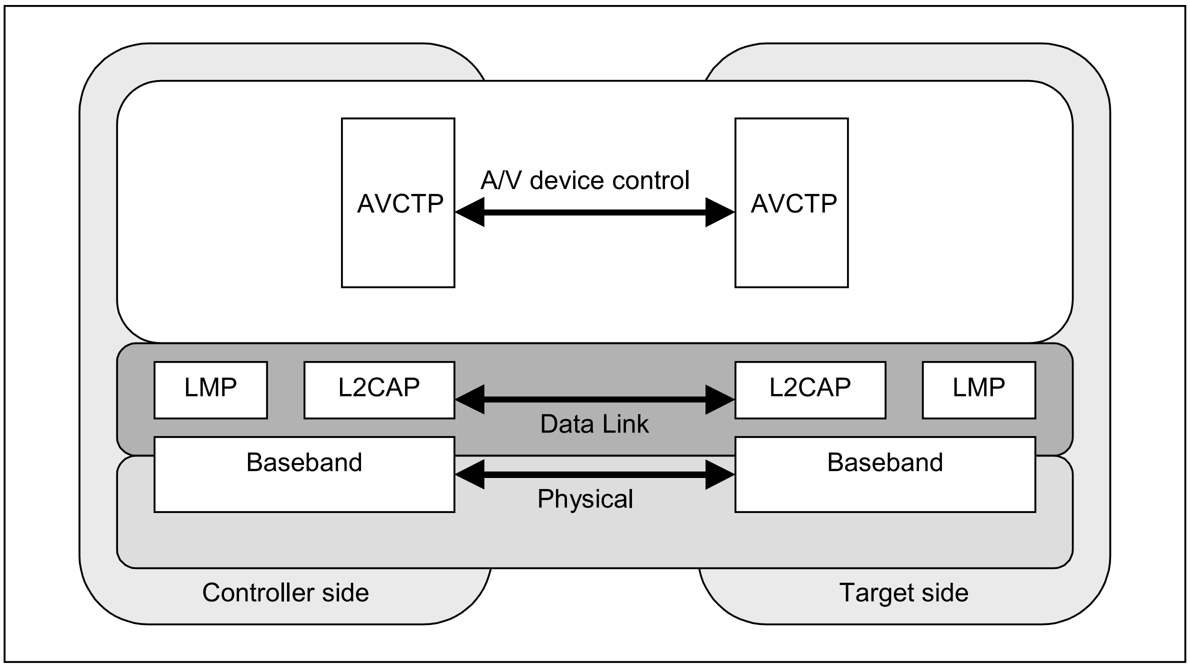


Figure 2-1: AVCTP as Control Layer Between Bluetooth Devices

AVCTP uses point-to-point signalling over connection-oriented L2CAP channels that have first to be set up between both devices. L2CAP channels are the most suitable for the support of A/V applications, which require dedicated transport services for A/V content streaming and feature control on the same link.

2.2 Operations Between Layers

AVCTP implementations shall be based on the general architecture described in the following figure.

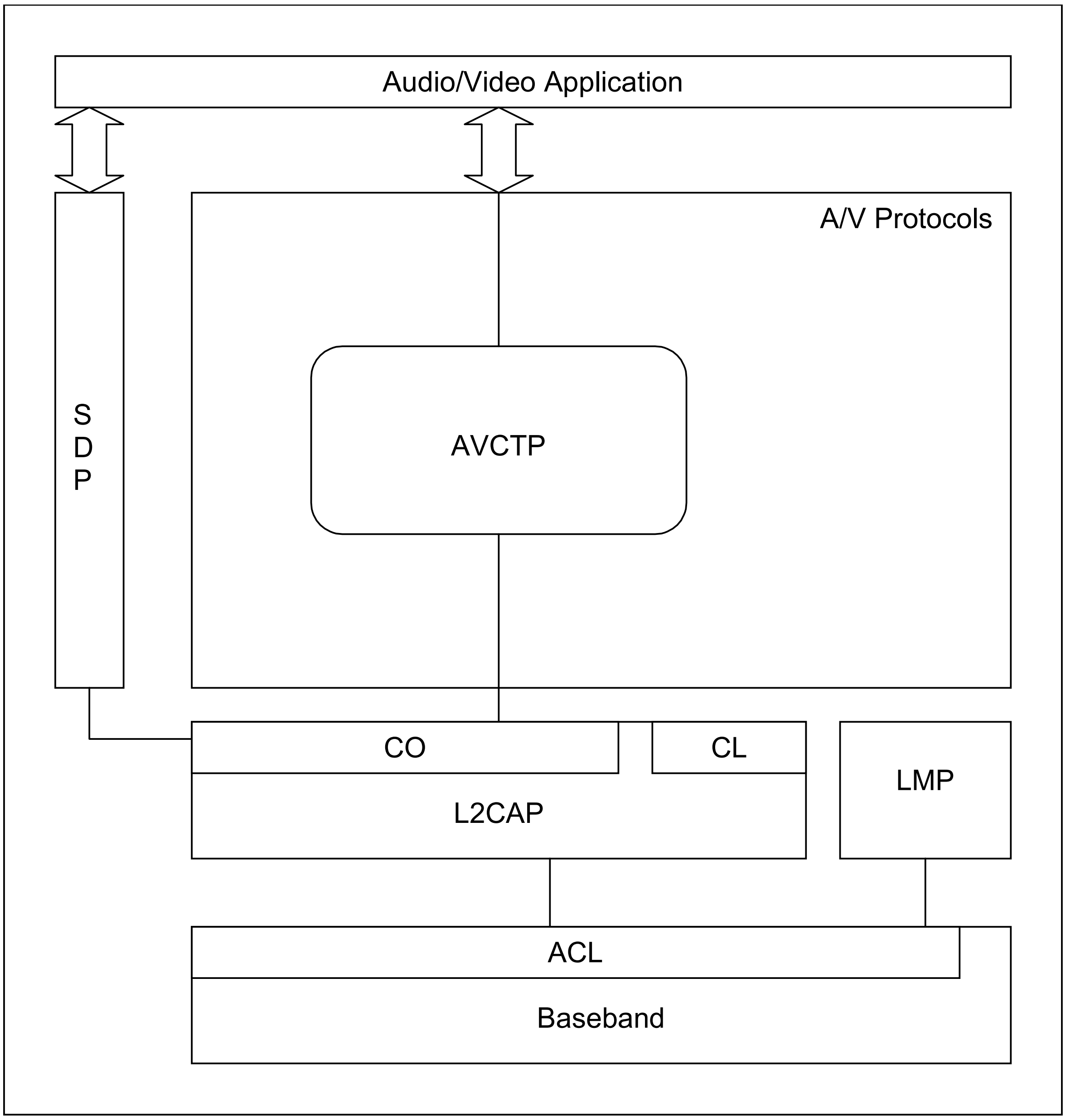


Figure 2-2: AVCTP Stack Architecture

In the above figures, AVCTP is one of the protocol specifications used in the context of audio and video applications. AVCTP specifies the protocol used to transport the control messages (command and response) between compliant devices in A/V applications.

All A/V protocols are specified independently so that a remote controller function <u>may</u> be implemented in a non-streaming device while an A/V streaming device <u>may</u> not support A/V controls. However, this protocol could coexist in the same device with other A/V protocols and <u>should</u> be able to share a common ACL link.

3 AVCTP Generic Transaction Model

This section specifies the transaction model used by AVCTP to exchange messages between two A/V-compliant devices.

3.1 Conceptual Element Definitions

AVCTP transactions are performed when a command message and possible response messages are exchanged between two compliant devices.

The device that initiates an AVCTP transaction by sending a command message is called the controller (CT). The command message is transmitted to the remote device that is called the target (TG) for this transaction. The target returns zero or more responses back to the controller according to the condition of the target or the type of the initiated transaction.

3.2 General Requirements

AVCTP transactions require that an ACL link has been initially set up between the pair of devices that interact for the related procedures. AVCTP transactions are performed on a single connection-oriented channel to be established between the devices: those transactions consist of bi-directional messages transported on this channel.

The L2CAP channel used for AVCTP transactions is set up, configured, and released using standard L2CAP services according to the procedures of [R1]. AVCTP provides full L2CAP channel service encapsulation and ensures that only one channel connection is used for this purpose between peer devices.

In AV devices controls may exist at both sides of connection. For this reason AVCTP shall be able to support both controller and target functionalities at both sides of the connection.

A specific PSM (Protocol/Service Multiplexer) value is required for the AVCTP protocol. This value is assigned by the Bluetooth SIG [R2].

3.3 Transaction Procedure

The controller initiates a transaction by sending a command to the remote device that is the target for this transaction. A complete AVCTP transaction consists of one message containing a command addressed to the target and zero or more messages containing a response returned to the controller by the target. AVCTP neither controls the command/response message sequence nor defines the rules that describe the controller and target behaviour; such rules are defined by the control profiles that use the AVCTP transport services.

An example of a basic transaction, in which a target performs an internal action on receipt of a command message and then returns a response, is shown in Figure 3-1: AVCTP Basic Transaction Example.

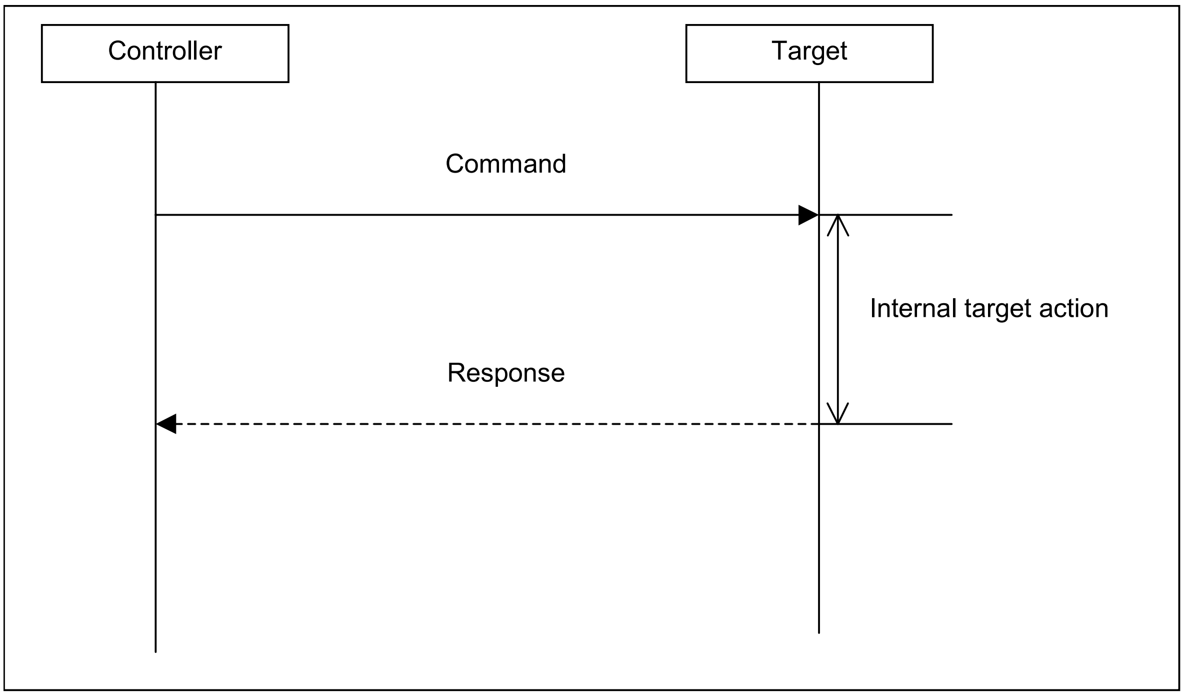


Figure 3-1: AVCTP Basic Transaction Example

3.4 Bit and Byte Ordering Conventions

In this specification the leftmost position in the drawings and tables represent the MSB of the corresponding byte or field.

When multiple bit fields are contained in a single byte and represented in a drawing, the more significant (high-order) bits are shown toward the left and the less significant (low-order) bits toward the right.

When drawn vertically multiple-byte fields are represented with the more significant bytes (high order) toward the top and the less significant bytes (low order) toward the bottom; when drawn horizontally the more significant bytes (high order) are represented toward the left and the less significant bytes (low order) toward the right.

3.5 Message service limitations

In this transaction model the response message is not compelled: whether the target sends back one or more response message depends on the application not on the reliability of the transport. The message service is a transport service that only ensures the integrity of a passed on message. On unreliable ACL links the transmission can fail: in such case it is up to the application to decide about retransmission of the complete message. In other words as the response message is not compelled actual acknowledgment of a command is not supervised by the protocol itself.

3.6 Byte-Constrained Packetization

The payload size is always adjusted to fit octet boundaries in packet transmission.

4 AVCTP Message Services

4.1 Overview

Each AVCTP command or response message is transmitted in one or several AVCTP packets that consist of a packet header part and a variable length message information part. As AVCTP packets does not contain a length information they rely on the L2CAP packet length field to delimit the transported packets: for this reason each AVCTP packet shall be transported on a single L2CAP packet.

The header part includes a transaction label that uniquely identifies the transaction with a limited scope in the command/response flow and packet type information that provides support for message fragmentation.

The message information part contains the command or response data with a flow direction indicator and a profile context information (see Section 6.2).

4.2 Transaction Labelling

Each AVCTP command/response message is identified by a *transaction label* field, which specifies a unique tag for each transaction. The transaction label field is a 4-bit field. All transaction label values are valid and all values shall be accepted by all AVCTP implementations. On the controller and the target side, handling of transaction labels is dependent on the application, and is therefore not defined in this specification. AVCTP uses the transaction label in the destination entity to identify AVCTP packets that belong to the same message: the application shall provide label values that permit differentiation between packets of different messages.

4.3 AVCTP Message Fragmentation

In the AVCTP transaction model, most command or response messages are transported as payload of a single L2CAP packet. Occasionally large messages need to be fragmented by AVCTP for the transport over more than one L2CAP packet: the number of required L2CAP packets depends on the Maximum Transmission Unit (MTU) the channels have negotiated according to the procedures in [R1]. The **Packet_Type** field (see Section 6.1.2) qualifies each L2CAP packet as either first (**Packet_Type**=01), continue (**Packet_Type**=10), or end packet (**Packet_Type**=11). In the case of a non-fragmented message (see Section 6.1.1), this field (**Packet_Type**=00) simply indicates that the message fits into a single packet and the number of packets is not inserted in the message.

As AVCTP packets can be transmitted on unreliable ACL connections, the number of packetsto be expected at the destination side shall be inserted by the AVCTP sender side in front of the first packet message information.

4.4 Multiple Profiles Support

AVCTP is used by applications to convey control messages transparently between applications. Depending on the configuration complexity, different profiles with different usage rules may be supported by controllers and targets. AVCTP uses the concept of a Profile Identifier to allow applications to discriminate messages related to different profiles.

In AVCTP transactions, a Profile Identifier field is used to represent the profile control mechanisms to which the transported message is referring. When a controller application wants to initiate a transaction according to some profile rules, it first selects the appropriate PID value for this profile and passes this value to AVCTP for insertion in front of the command message to be sent (see Section 6.2). If the received PID value is in the supported range, the target decodes the command message according to the related profile rules. In its response the target application shall indicate the received PID value of the matching command in order that the controller can decode the response message using the same rules.

4.5 AVCTP Message Type

This protocol provides an indication of the type of received message (command or response). As the device may support the dual role of controller and target in the same application, the message type information provides discrimination in command/response flows that relate to the assigned device roles.

4.6 AVCTP Message Size Information

As L2CAP supports variable payload length, no additional length information in AVCTP packets is required.

5 AVCTP Channel Management Services

5.1 Channel establishment and profile registration

In AVCTP configurations the device that supports the controller functionality is also responsible for initiating the L2CAP channel connection on request of the application. As both sides of the connection may support an active controller or controllers for different profiles all AVCTP implementations shall include a channel management entity to resolve conflicting requests and shall ensure that only one channel is connected between peer entities.

5.2 Channel release and profile de-registration

By nature control transactions between AV devices are sporadic; therefore a target could presumably request for releasing the channel while it is still in use by a controller application at the remote side or when the channel is still in use for another control profile. This could cause the target to be unresponsive for a while at a next user command because the channel needs to be re-established first by the controller side. Therefore each AVCTP entity is responsible to register/deregister the channel connection usage on a per profile basis locally.

6 AVCTP Message Format

AVCTP messages are transmitted in one or several AVCTP packets.

6.1 AVCTP Packet Headers

The packet header format depends on the possible fragmentation of the message.

6.1.1 Non-Fragmented AVCTP Message

The following table describes the generic format of an AVCTP message encapsulated in a single L2CAP packet.

7	6	5	4	3	2	1	0	
Transaction label				Packet_type (00)		C/R	IPID	Octet 0
Profile Identifier (PID)								Octet 1
								Octet 2
AVCTP Command/Response Message Information								Octet 3
								:
								Octet n+3

Table 6-1: Not-Fragmented AVCTP Packet Format

The **Transaction label** field (octet 0, bits 7-4) value is provided by the application.

The **Packet_type** field (octet 0, bits 3 and 2) is set to zero (00) to indicate that the command/response message is transmitted in a single L2CAP packet.

C/R (octet 0, bit 1) indicates whether the message conveys a command frame (0) or a response frame (1). It is provided by the application.

The **IPID** bit (octet 0, bit 0) is set in a response message to indicate an invalid Profile Identifier received in the command message of the same transaction; otherwise this bit is set to zero. In command messages this bit is set to zero.

The **Profile Identifier** (PID) field indicates that the command/response frame is coded according to the rules defined by the identified profile. The value shall be identical to the 16-bits UUID of the service class defined for this profile in the Bluetooth Assigned Numbers document [R2].

The next figure depicts the encapsulation process for Command/Response messages that fit the MTU requirements for transport in a single L2CAP packet.

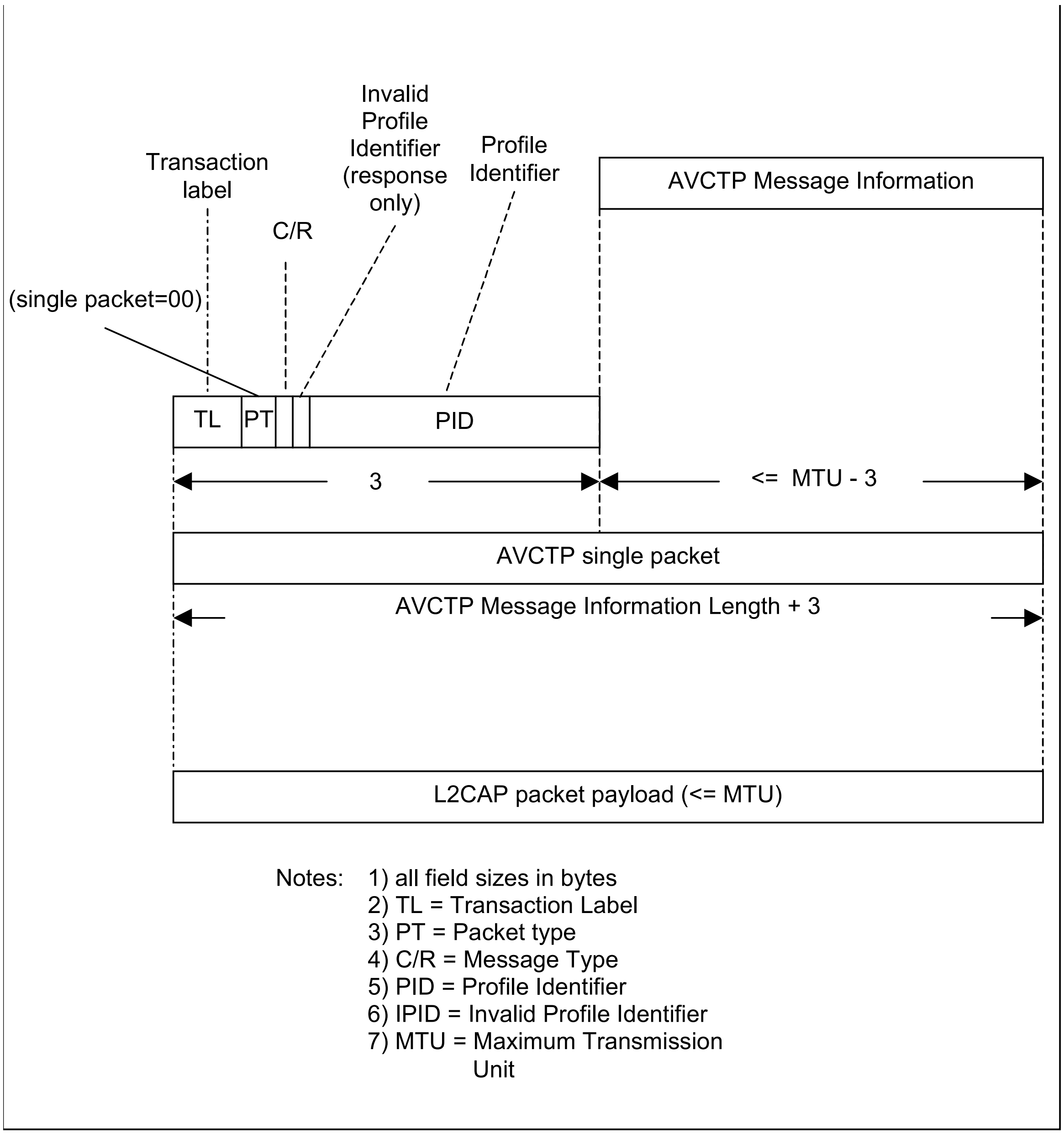

Transaction label
C/R
Invalid Profile Identifier (response only)
Profile Identifier
(single packet=00)
AVCTP Message Information
TL
PT
PID
3
<= MTU - 3
AVCTP single packet
AVCTP Message Information Length + 3
L2CAP packet payload (<= MTU)
Notes: 1) all field sizes in bytes
2) TL = Transaction Label
3) PT = Packet type
4) C/R = Message Type
5) PID = Profile Identifier
6) IPID = Invalid Profile Identifier
7) MTU = Maximum Transmission Unit

Figure 6-1 : Unfragmented AVCTP Message: Encapsulation

6.1.2 Fragmented AVCTP Message

The following section describes the format of AVCTP packets used to transport AVCTP command/response messages that cannot fit in a single L2CAP packet.

Fragmentation shall occur in the transmitting AVCTP entity to divide a too large message information into AVCTP packets less than or equal to the L2CAP negotiated MTU limit before sending them to the L2CAP layer. Fragmented messages are marked by a specific packet type code in each packet of the sequence. There are 3 possible types used in fragmented message packets:

- The **start packet** type for the first packet in the sequence.

- The **continue packet** type is used for all subsequent packets that are not the last packet in the sequence.

- The **end packet** type qualifies the last packet in the sequence.

At the receiving side the AVCTP entity reassembles the complete message by identifying packets that belong to the same transaction and by inspecting the type of each packet.

The L2CAP channel guarantees individual AVCTP packet integrity and delivery in the right order, not the integrity of the entire message. As the channel can be unreliable, some AVCTP packets of the sequence can be lost. As the start packet provides the number of AVCTP packets needed to reassemble the complete message, the AVCTP receiving entities shall implement a consistency check and discard any incomplete message.

The receiver of a fragmented message may allocate the resources for message reassembly based on the start packet length and the number of packets information elements provided by the first packet. For this reason:

- The payload of the L2CAP packets that encapsulate the start and continue packets of a fragmented AVCTP message shall have the same length

- The payload of the L2CAP packet that encapsulates the end packet of an AVCTP message shall not be larger than the start and continue packets belonging to the same message.

The following tables describe the detailed format for each AVCTP packet type used in fragmented messages.

7	6	5	4	3	2	1	0	
Transaction label				Packet_type (01)		C/R	IPID	Octet 0
Number of AVCTP Packets								Octet 1
Profile Identifier (PID)								Octet 2 Octet 3
AVCTP Command/Response Message Information (AVCTP start packet)								Octet 4 : Octet n+4

Table 6-2: Fragmented AVCTP Message Packet Format: Start Packet

7	6	5	4	3	2	1	0	
Transaction label				Packet_type (10)		C/R	RFA	Octet 0
AVCTP Command/Response Message Information (AVCTP continue packet)								Octet 1 : Octet n+1

Table 6-3: Fragmented AVCTP Message Packet: Continue Packet

7	6	5	4	3	2	1	0	
Transaction label				Packet_type (11)		C/R	RFA	Octet 0
AVCTP Command/Response Message Information (AVCTP end packet)								Octet 1 : Octet n+1

Table 6-4: Fragmented AVCTP Message Packet: End Packet

The **Transaction label** field (octet 0, bits 7-4) value is provided by the application and is replicated by the sender of the message in each packet of the sequence. It is used at the receiver side to identify packets that belong to the same message.

The **Packet_ type** (octet 0, bits 3 and 2) field qualifies the AVCTP packet as a start (01), continue (10), or end (11) packet.

C/R (octet 0, bit 1) indicates whether the message conveys a command frame (0) or a response frame (1). This field is provided by the application and is present in each packet of the message.

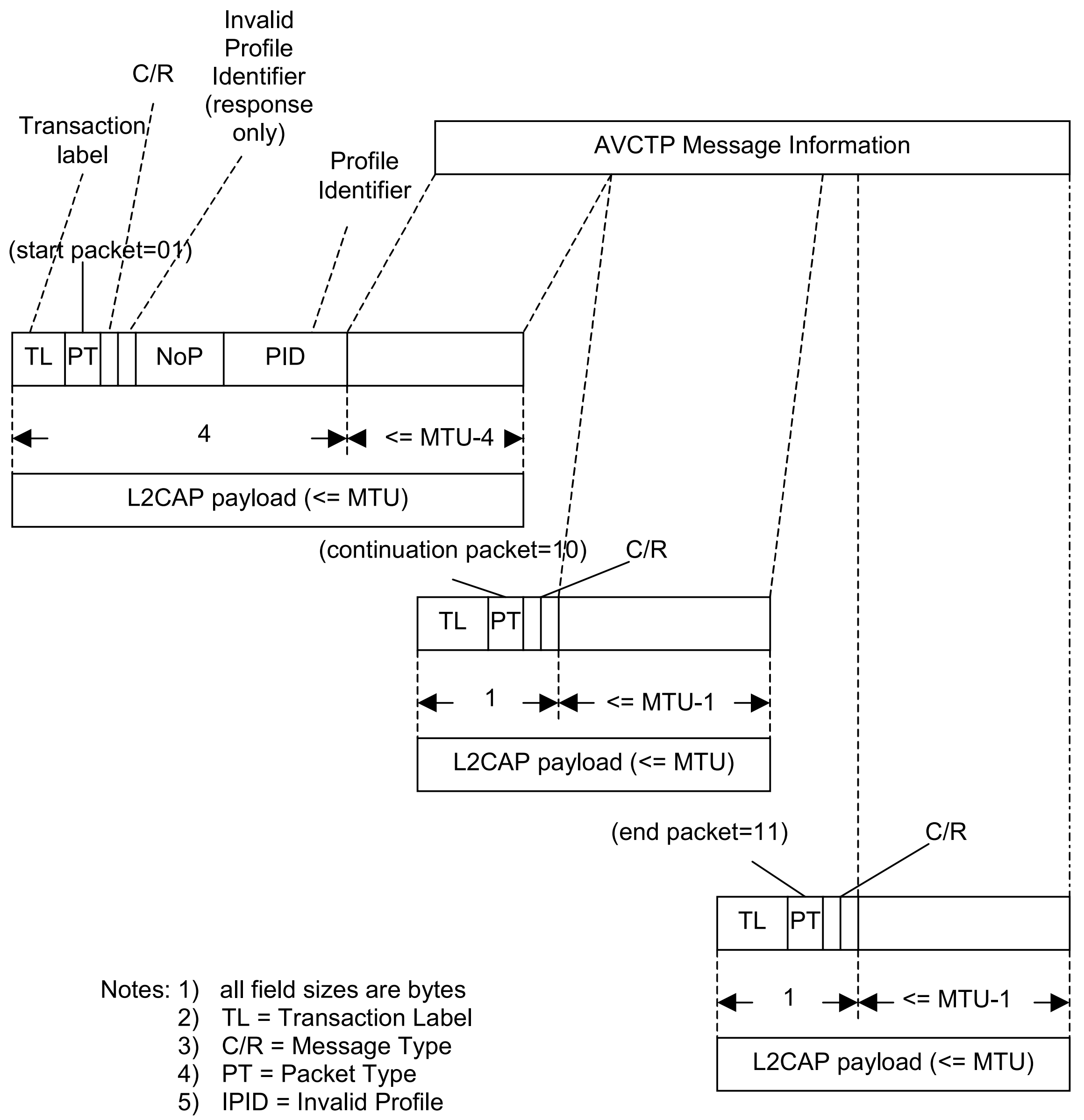
The **IPID** bit (octet 0, bit 0) is set in a response message to indicate an invalid Profile Identifier received in the command message of the same transaction; otherwise this bit is set to zero. In command messages this bit is set to zero. This field is only present in the start packet of the message.

The **Number of AVCTP Packets** is present in every start packet (octet 2) to indicate the total number of AVCTP packets that belong to the same message. As the start packet is also counted, this value is always greater than 1.

The **RFA** field (octet 0, bit 0) replaces the **IPID** field in continue or end packets. It is reserved for future additions and shall be set to zero (0).

The **Profile Identifier** (PID) field indicates that the message information part is coded according to the rules defined by the identified profile. The value shall be identical to the 16-bits UUID of the service class defined for this profile in the Bluetooth Assigned Numbers document [R2]. This field is only present in the start packet of the message.

The next figure depicts the encapsulation process for Command/Response messages that do not fit within the MTU requirements for transport in a single L2CAP packet.



Notes:
1) all field sizes are bytes
2) TL = Transaction Label
3) C/R = Message Type
4) PT = Packet Type
5) IPID = Invalid Profile Identifier
6) NoP = Number of Packets
7) PID = Profile Identifier
8) MTU = Maximum Transmission Unit

Figure 6-2 : Fragmented AVCTP Message: Encapsulation

6.2 AVCTP Message Information Part

The AVCTP message information part consists of a variable length command/response frame.

The format, coding, and usage rules of the command/response frames depend on the PID value indicated by the header. Those requirements are specified in the relevant profile.

7 Profile Identifier Usage Rules

For reasons of interoperability, this protocol specifies how the PID values shall be defined and used by applications. All profiles that use AVCTP services shall comply with these rules.

7.1 Minimum Requirements for Control Profiles

Each control profile that mandates AVCTP in a protocol descriptor list has an associated Profile Identifier (PID) for use by compliant devices to exchange AVCTP messages in the context of this profile: actually the PID value refers to the message coding and usage rules used by the indicated profile and shall be set to the Service Class UUID of the relevant profile.

For each discoverable control profile registration of the PID value for use by the local AVCTP transport service is mandatory. A description of the registration service can be found in [R1]. Each AVCTP implementation shall provide resources to support the required number of PID's the local device exposes.

7.2 Handling of messages for not registered profiles

A target AVCTP entity shall not hand down command messages that do not carry a registered PID: in this case AVCTP shall directly return a response message made of the 3-octets packet header with the PID value copied from the command and the IPID bit set to 1. The message shall not contain any response frame data.

8 References

[R1] Bluetooth SIG, Specification of the Bluetooth System, Core, Version 1.1

[R2] Bluetooth Assigned Numbers, Version 1.1 and over

9 List of Figures

10 List of Tables

11 APPENDIX A – AVCTP Upper Interface

This section presents an abstract description of the services offered by AVCTP in terms of service primitives and parameters. The service interfaces are specified for testing purposes only and may be used as a basis for other application specific implementations. The interface is described independently of any platform specific implementation.

AVCTP provides two types of service interfaces:

- Event registration service call: an application can perform registration for being notified when some asynchronous events are detected by AVCTP. At registration time the application shall indicate what event shall be reported; the application also provides an entry point where it can be called back in case such event occurs. Additional input and output parameters can be required at registration time for appropriate service configuration. The callback entry point is specified with all the input parameters that shall be notified by the service at the time the event is signalled.

- Application direct calls for service: this interface allows an application to request the execution of a peculiar on-demand service. A direct service call is made through a unique entry point; it requires a number of input parameters and a number of output parameters to be exchanged at execution time. An example of such direct call is the request of a compliant device to send a control message to the peer device.

In this section bracketed parameters are optional or conditional parameters that depend on the service context: for instance in the case of reject responses some parameters are not relevant as they are not transferred back to the requesting side.

When parameter types, values or ranges are not specified the AVCTP signalling section shall be used as reference.

The support level for each described primitive is specified in [**Error! Reference source not found.**]. Service primitives that are needed for testing are mandatory. All other primitives are optional.

11.1 Event Registration service call

Service	Input Parameters	Output Parameters
AVCT_EventRegistration	Event, Callback, PID	Result

Description:

The aim of this primitive is to request an application callback when the selected indication Event occurs. Each profile shall register for being called back separately.

Input Parameters:

Event *Type:uint* *Size: 2 octets*

Value	**Description**
0x0000	Forbidden
0x0001	AVCT_Connect_Ind
0x0002	AVCT_Connect_Cfm
0x0003	AVCT_Disconnect_Ind
0x0004	AVCT_Disconnect_Cfm
0x0005	AVCT_MessageRec_Ind
Other	RFD (Reserved for Future Definition)

Event definitions:

- AVCT_Connect_Ind: This event is sent by an AVCTP entity to the local application when a L2CAP connection (re)attempt is made by a remote device and no channel connection exists between both entities.

- AVCT_Connect_Cfm: This event is sent by an AVCTP entity to the local application when the requested channel connection is made available for the application.

- AVCT_Disconnect_Ind: This event is sent by an AVCTP entity to the local application when the signalling channel between peer AVCTP entities becomes unavailable on request of the remote side.

- AVCT_Disconnect_Cfm: This event is sent by an AVCTP entity to the local application when the when the requested channel disconnection procedure is complete.

- AVCT_MessageRec_Ind: This event is sent by an AVCTP entity to the local application to signal a received message from a distant device. At registration time an application indicates the profile ID to be considered by AVCTP to identify the message handler to be called back. An AVCTP implementation shall support registration of at least one profile ID; only one registration is allowed per profile ID.

Callback *Type:function* *Size: N/A*

Event	Callback Function Input Parameters
AVCT_Connect_Ind	BD_ADDR
AVCT_Connect_Cfm	BD_ADDR, Connect Result, Config Result, Status
AVCT_Disconnect_Ind	BD_ADDR
AVCT_Disconnect_Cfm	BD_ADDR, Disconnect Result
AVCT_MessageRec_Ind	BD_ADDR, Transaction, Type, Data, Length
Other	RFD (Reserved for Future Definition)

PID *Type:uint* *Size: 2 octets*

Value	Description
0xXXXX	Profile Identifier used to identify messages from/to the calling application

This parameter is provided by the application at AVCT_MessageRec event registration time.

Output Parameters:

Result *Type:uint* *Size: 2 octets*

Value	Description
0x0000	Event successfully registered
0x0001	Event registration failed
Other	RFD (Reserved for Future Definition)

Callback Input Parameters:

BD_ADDR *Type:uint* *Size: 6 octets*

Value	Description
0xXXXXXXXXXXXX	Unique Bluetooth address of the signalling remote device

RSP *Type:uint* *Size: 2 octets*

Value	**Description**
0x0000	Channel successfully connected
0x0001	L2CAP channel connection failure – see *Res* and *St* parameters if relevant
0x0002	No resources available
Other	RFD (Reserved for Future Definition)

Connect Result *Type:uint* *Size: 2 octets*

Value	**Description**
0xXXXX	L2CAP Connect Request Result (See [R1])

Config Result *Type:uint* *Size: 2 octets*

Value	**Description**
0xXXXX	L2CAP Configure Result (See [R1])

Status *Type:uint* *Size: 2 octets*

Value	**Description**
0xXXXX	L2CAP Connect Request Status (See [R1])

Disconnect Result *Type:uint* *Size: 2 octets*

Value	**Description**
0xXXXX	L2CAP Disconnect Result (See [R1])

Transaction *Type:uint* *Size: 1 octet*

Value	**Description**
0xX	Transaction identifier as provided by the remote application. Range is 0x0 to 0xF

Type *Type:uint* *Size: 1 octet*

Value	**Description**
0x01	Command Message Type
0x02	Response Message Type
Other	RFD (Reserve for Future Definition)

Data *Type:pointer* *Size: N/A*

Value	**Description**
N/A	Address of the AVCTP data buffer containing the received message information.

Length *Type:uint* *Size: 2 octets*

Value	**Description**
0xXXXX	Length of the received message information.

11.2 Application Service calls

In order to fulfill the requirements for Channel Management Services (see § 5), the following behaviour is expected for the channel establishment:

- AVCTP registration is automatically performed when a controller application calls the local connect service for a dedicated profile.
- If there is no channel already established at the time a profile registers the local AVCTP shall initiate the channel connection.
- Only one channel connection may exist between peer AVCTP entities at a time.
- During the establishment of the L2CAP channel, the remote side cannot be aware of the PID on the local side. Therefore, all profiles registered in AVCTP on the remote side shall be informed of the connection request.

The following behaviour is expected for the channel release:

- AVCTP deregistration is automatically performed when a controller or a target application calls the local disconnect service for a dedicated profile.
- The channel is released by the local AVCTP entity as soon as the last profile is de-registered locally by calling the disconnect service.
- When a channel is released by a remote entity and there are still one or more registered profiles in the local entity AVCTP shall re-establish the channel connection immediately.

11.2.1 CONNECT REQUEST

Service	**Input Parameters**	**Output Parameters**
AVCT_Connect_Req	BD_ADDR, PID, [Local side configuration requests]	RSP

Description:

This AVCTP primitive is used to request a channel connection to a distant AVCTP entity. Registration of the indicated profile is performed by AVCTP locally. If no channel connection exists this L2CAP channel connection is established.

Input Parameters:

BD_ADDR *Type:uint* *Size: 6 octets*

Value	Description
0xXXXXXXXXXXXX	Unique Bluetooth address of the distant device

[Local side configuration requests] *Type:N/A* *Size: N/A*

Value	Description
N/A	L2CAP configuration options as requested by the local application (InMTU, OutFlow, OutFlushTO, LinkTO - see L2CAP relevant specification)

Output Parameters:

RSP *Type:uint* *Size: 2 octets*

Value	Description
0x0000	Request accepted
0x0001-0xFFFF	Request rejected (value is implementation dependent)

11.2.2 CONNECT RESPONSE

Service	Input Parameters	Output Parameters
AVCT_Connect_Rsp	BD_ADDR, Connect Result, Status, [Local side configuration response]	Config Result

Description:

This AVCTP primitive is used to acknowledge an AVCT_Connect_Ind event received from the AVCTP local entity.

Input Parameters:

BD_ADDR *Type:uint* *Size: 6 octets*

Value	Description
0xXXXXXXXXXXXX	Unique Bluetooth address of the distant device

Connect Result *Type:uint* *Size: 2 octets*

Value	Description
0xXXXX	L2CAP Connect Response Result (See [R1])

Status *Type:uint* *Size: 2 octets*

Value	Description
0xXXXX	L2CAP Connect Response Status (See [R1])

[Local side configuration response] *Type:N/A* *Size: N/A*

Value	Description
N/A	L2CAP configuration options that are accepted by the local application (OutMTU, InFlow - see L2CAP relevant specification). Those values are returned to the requesting side

Output Parameters:

Config Result *Type:uint* *Size: 2 octets*

Value	Description
0xXXXX	L2CAP ConfigureResponse Result (See [R1])

11.2.3 DISCONNECT REQUEST

Service	Input Parameters	Output Parameters

AVCT_Disconnect_Req	BD_ADDR, PID	RSP

Description:

This AVCTP primitive is used to request the disconnection of an existing channel between the local entity and a peer device. The profile of the calling application is de-registered locally. If there is no more registered profile locally AVCTP shall release the L2CAP channel connection.

Input Parameters:

BD_ADDR *Type:uint* *Size: 6 octets*

Value	**Description**
0xXXXXXXXXXXXX	Unique Bluetooth address of the distant device

Output Parameters:

RSP *Type:uint* *Size: 2 octets*

Value	**Description**
0x0000	Request accepted
0x0001-0xFFFF	Request rejected (value is implementation dependent)

11.2.4 SEND_MESSAGE

Service	**Input Parameters**	**Output Parameters**
AVCT_SendMessage	BD_ADDR, Transaction, Type, PID, Data, Length	Result

Description:

This AVCTP primitive is used by the local application to a send a next message to a peer entity. Only valid messages are transferred by the service to the peer entity.

Input Parameters:

BD_ADDR *Type:uint* *Size: 6 octets*

Value	Description
0xXXXXXXXXXXXX	Unique Bluetooth address of the distant device

Transaction *Type:uint* *Size: 1 octet*

Value	Description
0xXX	Transaction identifier provided by the local application

Type *Type:uint* *Size: 1 octet*

Value	Description
0x01	Command Message Type
0x02	Response Message Type
Other	RFD (Reserve for Future Definition)

PID *Type:uint* *Size: 2 octets*

Value	Description
0xXXXX	Profile ID used in this transaction

Data *Type:pointer* *Size: N/A*

Value	Description
N/A	Address of application data buffer where the service can get the message data to be transferred by the service to the peer entity.

Length *Type:uint* *Size: 2 octets*

Value	Description
0xXXXX	Length of the message to be transferred by the service to the peer entity

Output Parameters:

Result *Type:uint* *Size: 2 octets*

Value	**Description**
0x0000	Request accepted
0x0001-0xFFFF	Request rejected (value is implementation dependent)

11.3 Service Primitives Support Level

11.3.1 Event Registration Service

Registration of all events shall be supported.

11.3.2 Application Services

Support of all application service calls is mandatory.

12 APPENDIX B – Acronyms and Abbreviations

Acronym	Description
A/V	Audio/Video
AVCTP	Audio/Video Control Transport Protocol
C/R	Command/Response
CT	Controller
IPID	Invalid Profile Identifier
MSB	Most Significant Bit (Byte)
MTU	Maximum Transmission Unit
PID	Profile Identifier
PSM	Protocol/Service Multiplexer
RFA	Reserved for Future Addition
RFD	Reserved for Future Definition
TG	Target
TL	Transaction Label