AN IMPLEMENTATION OF BLUETOOTH AUDIO VIDEO DISTRIBUTION TRANSPORT PROTOCOL

Ilija Basicevic, Miroslav Popovic, Faculty Of Technical Sciences, University Of Novi Sad

I INTRODUCTION
Bluetooth [1] is a wireless communication protocol suite. It is targeted at small hand-held devices, cooperating in close proximity. Examples of use of Bluetooth are transfer of files between personal computer (PC) and MP3 player, synchronization of personal digital assistant (PDA) with PC, exchange of virtual visit cards, and reading of gas-meters. Bluetooth profiles define several use-cases for usage of Bluetooth devices. In order to support audio and video streaming applications, protocol suite includes Audio Video Distribution Transport Protocol (AVDTP) [2]. Generic part of protocols and procedures that realize distribution of audio/video content is defined in Generic Audio Video Distribution Profile (GAVDP) [3]. Section II presents GAVDP. In section III some aspects of AVDTP are presented. Section IV contains discussion regarding our implementation of AVDTP. Section V contains conclusion remarks.

II GAVDP
GAVDP is used in combination with other profiles. There are two roles for devices that implement this profile: Initiator and Acceptor.
Initiator is the device which initiates signaling procedure, "active opener". Acceptor is the device that responds to request from Initiator, "passive opener". The roles can be switched between devices when new procedure is initiated. For example, a portable player is Initiator, and headphones are the Acceptor. Figure I presents Bluetooth protocol stack in GAVDP profile.

Connection establishment starts with both devices in Idle state. It consists of several operations: Stream End Point (SEP) Discovery, Get Capabilities, Stream Configuration and Stream Establishment. Eventually, both devices are set to be in Open state.
SEP Discovery and Get Capabilities procedures are started by Initiator - to collect SEP information and service capabilities of Acceptor. Start Stream command sets both devices to Streaming state. Stream can be suspended (transition from Streaming to Open) and reconfigured. Figure 2 presents reconfiguration procedure. Devices can start Connection Release when in Open or Streaming state.

III AVDTP
AVDTP provides three types of service: Basic Media Transport, Reporting, and Error Correction service. Basic service ensures transport of media packets of each session over individual transport channels. The Reporting service provides statistical information about time alignment of

Stream represents logical end-to-end connection of unidirectional streaming media data. Peer devices assume the roles of either the Source (SRC) or Sink (SNK).
The role of Source/Sink is independent of device's role as Initiator/Acceptor. GAVDP recognizes three states in AVDTP entity: Idle, Open, Streaming.

Figure 1: Bluetooth protocol stack in GAVDP profile

Figure 2: Change Streaming Parameters Procedure
media streams and packet losses. Reporting messages are used to achieve appropriate media streams synchronization. Error Correction (Recovery) service uses media packets of one transport session at the SRC side to generate additional coded packets, these packets can be used on SNK side to reconstruct media packets that have been lost on transmission path. This service is useful in applications that have huge bandwidth requirements and limited retransmission capabilities. Recovery service implements flexible on-demand error correction. Application can decide to cover only most sensible parts of media stream. All Recovery packets are conveyed through separate transport channel. Media Transport is based on RTP Data Transfer Protocol (RTP), while Reporting is based on RTP Control Protocol (RTCP) of RTP/RTCP suite [4]. While RTP/RTCP relies on User Datagram Protocol (UDP), AVDTP changes that: it relies on L2CAP which provides connection oriented communication. L2CAP channels can be flexibly configured to enable bandwidth share between AVDTP data streams. Several L2CAP channels are opened during AVDTP session. First channel opened is used for AVDTP signaling. Then, depending on the AVDTP mode used, one, two, or three channels are opened for media transport, reporting and error correction, subsequently. Figure 3 presents AVDTP interfaces. AVDTP state machine, according to [2], contains the following group of states: Idle, Configured, Open, Streaming, Closing, Aborting.

IV IMPLEMENTATION

RTP/RTCP standard [4] recommends application level framing for implementation of protocol. The application level framing approach allows for closer cooperation of transport level and application level code. Application level code is integrated with transport level code, so it has closer control over transport level mechanisms and features. Since AVDTP has more responsibilities than RTP/RCP we have decided to use more strict layered architecture (separate the code of two levels).

Transport level AVDTP entity has Service Access Points (SAP) interface towards application level. Standard set of features (connect, read, write, disconnect operations) is extended with some specific operations of AVDTP: SEP discovery, get capabilities, open stream, start streaming, etc.

Figure 3 presents AVDTP architecture as in [2]. Adaptation layer provides the following functionality:
- robust header compression
- multiplexing of several sessions on one transport channel

Stream is identified using Stream End Point Identifier (SEID) and Stream Handle (SH). SEID is a cross-device reference to a specific stream. SH is device local identifier. A stream can be decomposed into one, two, or three transport sessions between peer AVDTP entities. The Transport Session Identifier (TSID) represents a reference to a transport session.

Telecommunications software has to meet several requirements, regarding high performance, robustness, and code maintainability. An important characteristic of telecommunications software is modularity. Layers pattern [5] promotes modularity by separating code of different layers.

Bluetooth protocol stack software has layered structure. It is a set of cooperating message processing entities. Each entity is built around a finite state machine (FSM). We have implemented two services of AVDTP: Basic Media Transport and Reporting.

There are two types of finite state machine in AVDTP implementation. The first type of FSM is responsible for AVDTP signaling. It contains the states listed in section III, AVDTP. The list of states is extended to include:
- A_Initial
- A_Connecting_L
These states cover initialization procedure: connection and configuration of L2CAP channel used for signaling.

The second type of FSM is responsible for opening, configuration, data transmission and closing of L2CAP channel – used for media transport and reporting. We refer to this FSM as L2CAP channel management FSM. Each AVDTP session starts with instantiation of signaling FSM. When signaling FSM reaches Open state, two L2CAP channel management FSMs are instantiated – for Media Transport and Reporting respectively. Figure 4 presents above mentioned entities in AVDTP, including message dispatcher. There is one instance of message dispatcher in one instance of AVDTP entity.

L2CAP channel management FSM has six states:
- C_Initial
- C_Connecting_L
- C_Connecting_L2
- C_Connecting_R
- C_Connecting_R2
- C_Active

There are two transitive connecting states for both Initiator and Acceptor. Initiator states are identified with suffix L (stands for Local Initiator). Acceptor states are identified with suffix R (stands for Remote Initiator). Since L2CAP channel connection is immediately followed by channel configuration, there are two transitive connecting states in both cases (local and remote) – to cover two channel establishment phases – connection and configuration.

Both channel types (media transport and reporting) use the common L2CAP channel management FSM for channel establishment. The only difference is in Active state. This is the only state when FSM needs to determine if it runs at media transport or reporting channel.

[6] presents a system of design patterns (Patterns for Protocol System Architecture, PPSA) for implementation of communication protocol stacks. There are three patterns in this system of patterns. In our opinion, one of them, Protocol System pattern is an architectural level pattern. Two other patterns, Protocol Entity and Protocol Behavior pattern, are design level patterns. Protocol System pattern contains layered architecture of Layers pattern [5]. Bluetooth protocol stack software supports Protocol System pattern. Each message processing entity (protocol implementation) in Bluetooth system represents an implementation of Protocol Entity pattern. It contains Protocol Entity, Protocol Behavior and Storage, and it uses Entity Interfaces and Peer Interfaces. Protocol Behavior, as defined in [6], contains the active parts of protocol entity. It contains zero or more Routers, zero or one Communication Manager, and zero or more Communication Sessions. PPSA assumes two types of Protocol Behavior pattern: connectionless and connection-oriented. Protocol Behavior is more complex in Bluetooth AVDTP than in PPSA, so its structure can not be easily mapped to the structure of Protocol Behavior in PPSA. In Bluetooth Protocol Behavior implementation for AVDTP, besides routing and session management, there are two types of session: signaling procedures session and transport session.

As Figure 3 shows, in AVDTP there are two sublevel entities (Signaling, Stream Manager) at least.

[7] presents design of Versatile Message Transaction Protocol (VMTP), which is a transport level protocol. VMTP avoids connection establishment redundancy in communications protocol stack. In common protocol stack implementations, connections are established at both transport and application level. In VMTP there is no notion of connection at transport level. VMTP uses the notion of message transactions. Message transaction is a sequence of request and response message. This approach lessens message processing overhead. Bluetooth establishes connections at L2CAP, AVDTP and application level.

V CONCLUSION

Bluetooth protocol stack is implemented as a set of cooperating entities. Software implements layered architecture. Each layer of Bluetooth protocol stack is implemented around finite state machine. AVDTP implementation contains two types of finite state machine. The first one controls Signaling channel. The second one controls operation of L2CAP channel. It is used for Media Transport and Reporting channel. Only when in Active state, L2CAP channel management FSM has to determine whether it runs as Media Transport or Reporting channel. This makes reuse of FSM code possible.

REFERENCES


AN IMPLEMENTATION OF BLUETOOTH AUDIO VIDEO DISTRIBUTION TRANSPORT PROTOCOL, Ilija Basic, Miroslav Popovic.