http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 10 (11): 2147-2153, 2011
ISSN 1812-5638 / DOI: 10.3923/itj.2011.2147.2153

Test and Evaluation of Flute Protocol Client Program

H. Jin, F. Zhiyi, Z. Yang, C. Ruicheng and Z. Lu
College of Computer Science and Technology, Jilin University, Changchun, China

Abstract: In broadcast and mobile systems, large-scale and scalable protocols are needed to guarantee reliability and provide function of congestion control. ALC (Asynchronous Layered Coding) was designed to meet that, Nonetheless, the drawbacks of it are obviously such as the end-to-end reliability can`t be guaranteed, the document to be sent and its attributes are separated. So, FLUTE (File Delivery over Unidirectional Transport) is defined based on ALC which can avoid the shortcomings of ALC by the mechanism of FDT. In this study, we tested and evaluated the client-side programs based on FLUTE to confirm that it can meet the user's needs. The correctness of the software exploited in this study also has been proved by applying in the European Union`s INSTINCT- WORLD project.

Key words: FLUTE, ALC, FDT, LCT

INTRODUCTION

In the mobile multimedia broadcasting system, the file transfer protocol used to support transmission, download and updates of a variety of file, images, audio and video clips, software, services broadcasting information and of various metadata. As a large audience broadcast content, file transfer protocol must meet the massive scalability. In addition, the transport protocol also face the difference of channel and terminal, diversity of the transmission content, transmission reliability, congestion control and security issues and other challenges.

EU INSTINCT-WORLD (DVB, 2005; ETSI, 2004) (IP-based Networks, Services and Terminals for Converging Systems, Worldwide) integrated projects is to study the integration of DVB-H/T (Digital Video Broadcast-Handheld /Terrestrial) digital TV network and GPRS/UMTS (General Packet Radio Service/Universal Mobile Telecommunication System) mobile communication network. Our research projects are the Work Package 5 of INSTINCT-Terminal side API of the mobile digital TV and middleware. The project purpose is to develop proprietary middleware product for the domestic manufacturers of mobile digital broadcasting in the coming interactive TV services to provide field production and to seize the initiative.

The content of this study is the transmission protocol module of the contents in the Terminal side API of the mobile digital TV and middleware R and D projects. We focus on the characteristics, working principle, system design and code process of FLUTE File Transfer Protocol, final show the operating results.

FLUTE PROTOCOL ANALYSIS

FLUTE (Paila *et al*., 2004; Chu and Wong, 2008) (File Delivery over Unidirectional Transport) protocol is the preferred option of IP multicast transport protocol under the conditions of large-scale scalable, IPDC (IP Datacast) solutions in the DVB-H and MBMS (Multimedia Broadcast Multicast Service) solution in the 3G cellular networks both use FLUTE protocol as its content delivery protocol, aims to support handheld devices flexible and efficient mobile multimedia broadcasting applications.

FLUTE is a multicast-oriented one-way file transfer protocol defined in 3926 RFC. The aim of this protocol is to achieve reliable file transfer by one or more of the sender to one or more of the receiving end in the Internet or other one-way data transmission system (such as the DVB-H) based on multicast (also supports unicast).

FLUTE in DVB-H protocol stack position as shown in Fig. 1. DVB-H standard bases on IP protocol, realizes the full IP optimization in the network layer, the application layer sends data content to numerous users by IP multicast technology, FLUTE protocol in the transport layer protocol UDP (User Datagram Protocol) above, suitable for one-way large-scale reliable transmission of IP multicast system files and ESG metadata.

FDT-file attributes mapping mechanism: FDT (File Delivery Table) is a new file properties mapping mechanism which is introduced to FLUTE based on ALC (Asynchronous Layered Coding) session management function, it is the essence of FLUTE protocol. ALC protocol only provides any binary object transmission but

Corresponding Author: He Jin, College of Computer Science and Technology, Jilin University, Changchun, China

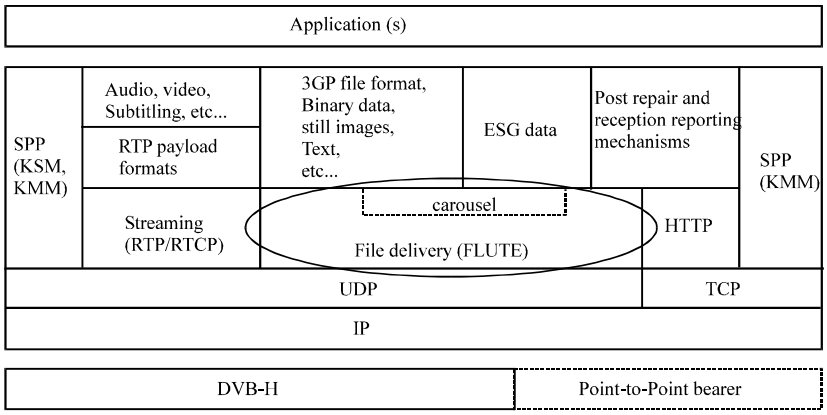

Fig. 1: FLUTE in DVB-H protocol stack position

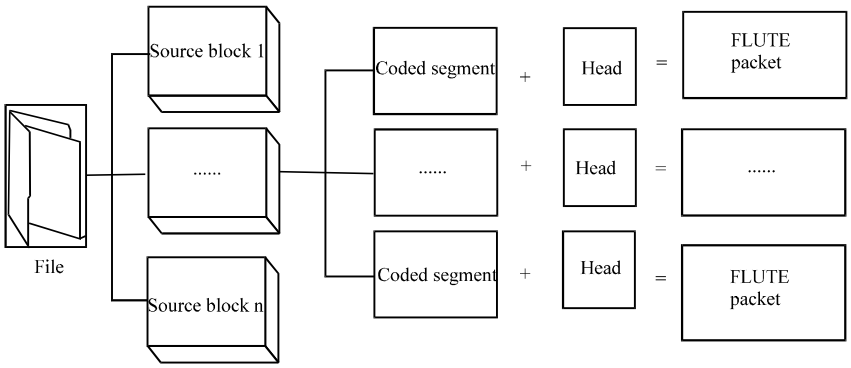

Fig. 2: Working principles of FLUTE

in the actual application, the receiver application should also know what the received object represents, or the related attributes of file object is what. FDT provides a mechanism transmitting various file related attributes information in the file transfer session, the attributes include filename, file ID, file type, file size, etc. Logically, FDT can be identified as a collection of some file described entries. Each entry represents a mapping relationship between a file and its attributes and must contain the URI which describes the files and the TOI which assigned to the file when the files are sending. Each file transfer session must have a FDT to describe all files to its attribute the corresponding relationship of the session.

FDT is transmitted in the form of FDT instance, the FDT instance can be seen as a special file object in transmission process and transmitted together with other file object in a transmission conversation. FLUTE advices transmitting FDT instance before the contents of the file, so the receiver application can get the descriptive information about the document before the contents of the file arrive, decide whether receive the file in advance to avoid waste of resources caused by receive file that do not want to accept or cannot be handled.

The receiver maintains a FDT database according to the received FDT instance. At any moment, the contents of the receiver FDT database represent the receiver's cognition about current file transfer session. Due to the independence of different receiver, their FDT database may be different at the same time.

Working principles of FLUTE protocol: FLUTE is a unidirectional transport protocol which inherited the entire characteristics of ALC. Simultaneously, its innovation compared with ALC is the mechanism of file delivery table been shorted for FDT which is the core of FLUTE. The function of FDT is providing mapping between files and their attribute based on the function of session management in ALC. The working principles of FLUTE are shown in Fig. 2.

The sender who runs FLUTE can initiate one or more transmission sessions, each session was identified by (IPS, TSI), in which the IPS represents IP address of the sender and TSI represents Transport Session Identifier

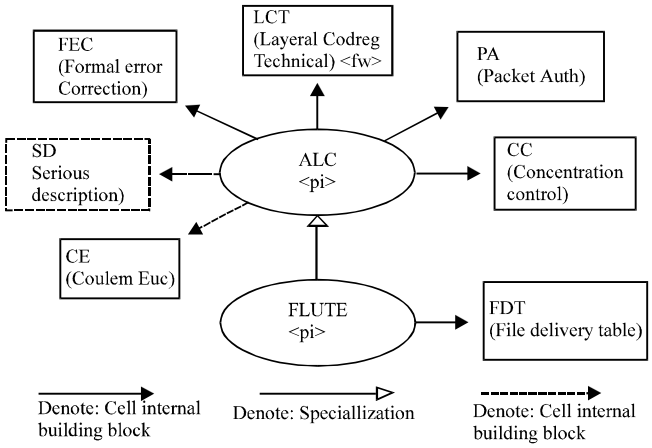

Fig. 3: Protocol component

(TSI). The file to be transmitted in a certain session called transmission object, one or more objects with unique identifier called TOI can be transmitted in one session, so, the working processes of ALC are mainly three steps as shown in Fig. 4, the details are as follows:

- **Step 1:** The file to be transmitted was separated in many blocks, each block called a source block and they have similar sizes
- **Step 2:** Coding for each source block in order to guarantee the reliability of the whole transmission
- **Step 3:** Adding the head which contains information such as the size of primitive block and the length of the code added in step 2 to the coded primitive blocks, in order to guarantee the receiver can receive the blocks in right sequence

The relevance of FLUTE, ALC, LCT: Before FLUTE, ITIF's RMT (Reliable Multicast Transport) workgroup has developed ALC (Luby *et al*., 2002a) (Asynchronous Layered Coding, defined in RFC3450) and LCT (Luby *et al*., 2002b) (Layered Coding Transport, defined in RFC 3451) protocol modules and the FEC protocol modules which are large-scale reliable multicast transport protocol and related content. LCT module is responsible for defining FLUTE / ALC protocol data packet formats and the working mechanism and process of file transfer protocol in common sense; FEC (Luby *et al*., 2002c) is responsible for forward error correction encoding and decoding, ALC modules are integrated to form a complete Protocol Instantiation, to achieve a large-scale, scalable, reliable transmission of arbitrary binary objects. FLUTE inherits the ALC and introduces the FDT concept based on ALC, in order to achieve the file attributes associated with the file mapping mechanism itself, making the file and file attribute information can be carried out with "in-band transmission".

ALC (Asynchronous Layered Coding) defines arbitrary binary object of multicast transfer protocol. The protocol is designed based IP multicast technology, it supports 'extensive scalable' to achieve reliable transmission of binary object.

FLUTE inherited ALC protocol in the basic content of transmission, so we can understand that the ALC is a base protocol or father protocol of FLUTE, their component parts can see Fig. 3.

DESIGN AND IMPLEMENTATION OF CLIENT PROGRAM

FLUTE is a subsystem of IP Datacast system, IP Datacast function is divided into three different levels:

- **Bearers layer:** Bearers provide IP data transport mechanism. In the IPDC/DVB-H, DVB-H transmits multicast and broadcast traffic efficiently in a multipoint way, thus forming the basis of IP Datacast business
- **Delivery layer:** When the application process of the receiver sends the file content, it needs to call one or more file delivery methods. Two delivery r methods defined in the IPDC are known as file delivery and streaming media. FLUTE protocol implementation methods are file delivery
- **User service layer:** The user service layer provides users with applications. Different applications will have different requirements for file delivery; also can call a different delivery method

FLUTE protocol stack application environment shown in Fig. 1.

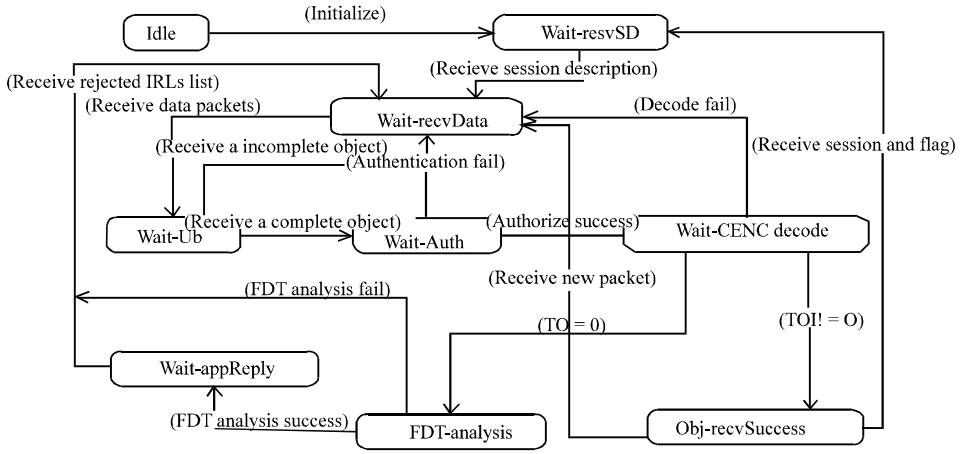

Fig. 4: State machine of FLUTE

Design of state machine: Through the analysis and understanding of the protocol RFC documents, finite state machines, state machine migration process and the corresponding trigger conditions exist in the FLUTE protocol client system are summarized as show in Fig. 4:

System state transition process is as follows:

- **Idle (Idle) state:** System in Idle state after start, then through the initialization process, into Wait_SD state
- **Wait_SD (wait for session descriptions) states:** In this state, the system waits for the application layer to send session descriptions and refuses to receive any data packets at the same time. When receive the session descriptions, go to the Wait_recvData state
- **Wait_recvData (waiting for data packets) state:** In this state, the system waits to receive data packets, when receive data packet, go to the Wait_recvObj state
- **Wait_recvObj (waiting to receive complete object) state:** In this state, the system determine whether an object is received completely, if the reception has been completed, go to Wait_Auth module, or go to Wait_recvData state
- **Wait_Auth (wait for authentication) states:** In this state, wait for authentication to received object, when receive the authentication through information, go to Wait_CENCdecode state; when receive authentication failure signal, abandon the current object and go to Wait_recvData state
- **Wait_CENCdecode (waiting for content decoding) state:** In this state, system wait for content decoding of the received objects, when the content decoding is successful, if the object TOI = 0, then go to FDT_analysis state, if TOI! = 0, go to Obj_recvSuccess state; when receive a content decoding failure signal, abandon the current object and go to Wait_recvData state
- **FDT_analysis (FDT analysis) state:** In this state, analysis he received FDT object t object, if successfully analyzes, transfer the analytical results to the application layer and go into the Wait_AppReply state; if the analysis fails, then discard the FDT instance, go to Wait_recvData state
- **Wait_AppReply (waiting for the application layer response) state:** In this state, waiting for the URI list of rejected file return from the application layer. After receiving this signal, go to Wait_recvData to waiting to receive data packets
- **Obj_recvSuccess (object successfully received) state:** The state indicates an object has been successfully received, if the new received data packets do not belong to the object, go to Wait_recvData receiving a new object of this session; if receive a end indicator of the session, then end this session, go to Wait_SD state waiting for the next session description information

Design of modules: According to the system's functionality requires, the system design is divided into:

- **Packet receive module (PR):** Call the socket API, from the lower UDP module and obtain destination IP address and port number from the control management module, establish Socket and receive packets to the specified buffer
- **Packet analysis module (PP):** Receive the pointer to a buffer and packet length information from PR module, extract TSI, TOI from the control management module and flit packet based on TSI, TOI, while inform the control and management

module FDT Instance ID, FEC-related information, content encoding scheme which extracted for FEC decoding and file reconstruction module

- **FEC decoding and file reconstruction module (FC):** FEC decoding and file reconstruction module calculates the source symbols based on the relevant information of packet analysis and assemble it into a transfer target, according to TOI to judge the difference between ordinary files and FDT Instance, according to extended header and the FDT length information Check the length of transmission objects and inform the control management module the file handle and TOI of the transfer object
- **Data Authentication module (DA):** Obtain relevant authentication information from the control management module using TOI of the object and receive transmit objects from the FC module, do "application level" data authentication based on MD5 algorithm, if authentication is passed, then provided content decoding module the file handle and TOI of the transfer object, if the authentication fails, then discard the object
- **Content decoding module (CD):** Receive the file handle of the transmission objects that through the authentication from DA module and receive content encoding information from the control management module using the TOI to determine whether the FDT need for content decoding, do contents decoding to the object in accordance with the agreed content encoding and decoding scheme of FDT, provides the decoded file handle to the upper interaction module and provides the decoded FDT instance and FDT Instance ID to the control management module
- **Upper interaction Module (UM):** Before the start of session, the upper interaction module receives the session descriptions and the instructions whether receive a file or not from the top layer and inform the control management module, receives the file handle from the CD module, while receives files attribute information of FDT table from the control management module, pass the file handle and documents related property to the upper
- **Control and management module:** Determine whether the file transfer is end Based on correlation algorithm

Their mutual relations, as shown in Fig. 5:

Implementation: FLUTE is a subsystem of the system of IP Datacast which is a part of project INSTINCT-WORLD, the architecture of IP Datacast system is divided into three layers: User service layer, Delivery Method layer and Bearers layer.

FLUTE is a protocol of the middle layer (delivery method layer), there are two kinds of transmission method: file transmission and streaming media transmission. FLUTE is a file transmission method.

FLUTE was implemented by C programming language in Linux platform on the premises of the three layer architecture.

TEST AND EVALUATION

The user-oriented command of FLUTE is flute but the parameters vary from service to service, users distinguish different service according to the parameter of the command. There are mainly three kinds of parameters in terms of different functions they can perform; the first one is sharing parameter that can be used both by senders or receivers, the second one is sender's parameter which is only used by senders; the other one is receiver's parameter which is only used by receivers. The sharing parameters and their functions are shown in Table 1. The

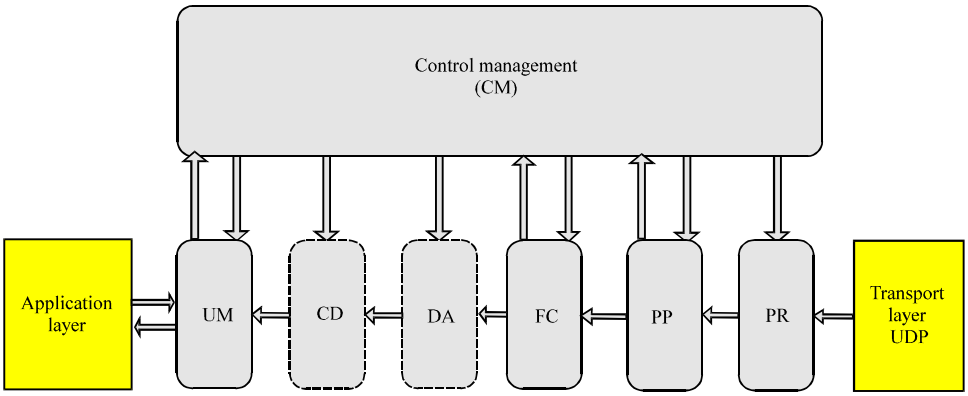

Fig. 5: Modules of FLUT

Table 1: Sharing parameters

-S	The terminal sending data acts as a server; default: acts as a client
-U	Type of address is unicast; default: multicast
-m:str	IP address of channel (unicast or multicast) default: 226.10.40.1
-p:int	Port of channel; default: 4001
-i:str	Local port, default: INADDR_ANY
-t:ull	Transfer Session Identifier, default: 1
-w:int	Methods of congestion control: 0:no congestion control, 1:RLC method; default: 0
-V:str	Print the log to the file named "str"; default: Standard output
-d:str	Session description file, default: none
-h	Print online help

Table 2: Sender's parameters

-c:int	Number of channel used, default: 1
-z:int ZLIB,	0: no coding 1: code with ZLIB for all files, 2: code FDT with code general file with GZIP; 3: code with PAD: default: 0
-D:ull	Session duration (unit: second); default: 604800
-f:str	Send files based on FDT; default name of the file to be sent: fdt.xml
-K	The complete scheme of FDT instance (D = 0)
-l:int	Length of the code (unit: byte); default 1428
-r:int	Transmission rate of the channel unit (kbit/s); default: 250
-T:int	The maximum hops packet can be retransmitted; default: 15
-e:int	0: in FDT file, 1: in the extend head; default: 0;
-F:str	The path of the file
-n:int	Times of transmission; default: 1
-C	Continuous transmission; default: not used
-x:int	0: no FEC code, 1: use simple xor FEC code, 2: use Reed-Solomon FEC code; default: 0
-X:int	Probability coding with FEC in the head of packet; default: 50%
-L:int	Max length of source block
-B:str	Base path of the file to be sent; default the current directory

parameters for senders are shown in Table 2. The parameters for receivers are shown in Table 3.

Test environment: Protocol client program running on the host named "hbzh @ debian" (host R), IP address "59.64.153.37", server-side application running on the host named "hejin @ wangyao" (Host S), IP address "59.64.186.193". On the host program's working directory of FLUTE protocol in host S, there is a base directory called files where the outgoing document in which contains two test files: testA.a and testB.b (Fig. 6).

- Protocol client receive the specified file on "file list mode".
- Protocol client receive the file from the server side on "automatic mode" (server side sends the file multi-channel, the channel number is 3) (Fig. 7)

Table 3: Receiver's parameters

-A	Receive files automatically
-F:str,str,...	file list excepted to be received
-c:int	Maximum of channels, default: 1
-B:str	base directory used to download files, default: flute-downloads
-s:str	Source IP address, default: IP address of the first packet.
-o:ull	TOI of the object excepted to be received
-E	Still receive FDT object which is overtime
-N	The object with obscure name

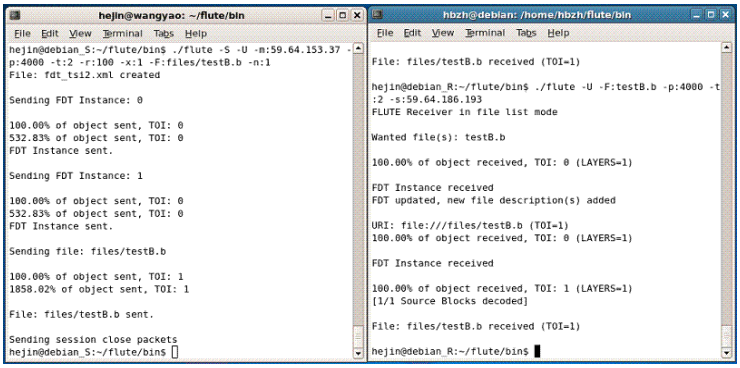

Fig. 6: File list mode

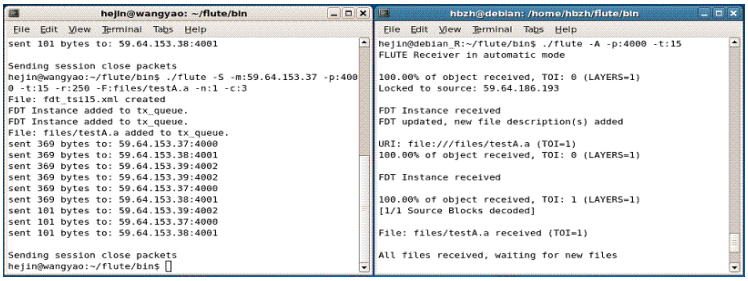

Fig. 7: Automatic mode A

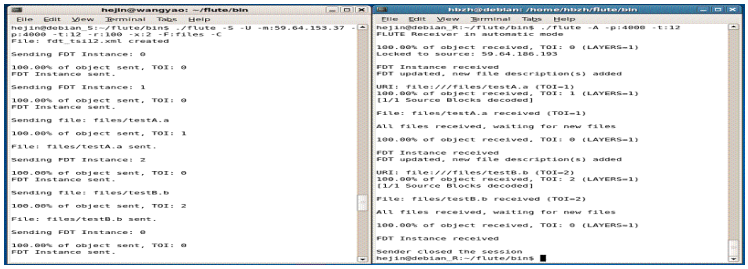

Fig. 8: Automatic mode B

- Protocol client receive the file from the server side on "automatic mode" (server side send the file cycle and do FEC coding for sending file based on Reed-Solomon algorithm) (Fig. 8)

CONCLUSION

FLUTE is introduced systematically and the working principles of the modules are analyzed in this study, simultaneously, the protocol of FLUTE is designed and finally implemented by C programming language under the Linux platform. The implementation and application of FLUTE show that it is a suitable protocol for IP multicast, unidirectional file transfer system. The intraband transmission of file attributes is realized though FDT mechanism; the reliability is guaranteed by FEC coding; In a word, FLUTE is an appropriate protocol in IP multi-cast systems because of its characteristic of simply control, scalability and considerable performance.

REFERENCES

Chu, C. and Y.C. Wong, 2008. FLUTE: Fast lookup table based rectilinear steiner minimal tree algorithm for VLSI design. Comput. Aided Design Integrated Circuits Syst., 27: 70-83.

DVB, 2005. IP datacast over DVB-H: Content Delivery Protocols (CDP). Digital Video Broadcasting Document A101, December 2005. http://www.dvb-h-online.org/PDF/a101.tm3350r3.cbms1167r12.IPDC_Content_Delivery_Protocols.pdf

ETSI, 2004. Digital Video Broadcasting (DVB): Transmission System for Handheld Terminals (DVB-H). ETSI standard, EN 302 304 V1.1.1,http://www.dvb.org/

Luby, M., J. Gemmell, L. Vicisano, L. Rizzo and J. Crowcroft, 2002a. Asynchronous Layered Coding (ALC) protocol instantiation. Cambridge University, Cambridge, http://tools.ietf.org/html/rfc3450

Luby, M., J. Gemmell, L. Vicisano, L. Rizzo, M. Handley and J. Crowcroft, 2002b. Layered Coding Transport (LCT) building block. http://ftp.zcu.cz /pub/doc/rfc /rfc3451.txt.pdf

Luby, M., L. Vicisano, J. Gemmell, L. Rizzo, M. Handley and J. Crowcroft, 2002c. Forward Error Correction (FEC) building block. RFC 3452. http://tools.ietf.org/pdf/rfc3452.pdf

Paila, T., M. Luby, R. Lehtonen, V. Roca and R. Walsh, 2004. FLUTE-file delivery over unidirectional transport. The Internet Society, October 2004. http://tools.ietf.org/html/rfc3926