Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2695645 C 2017/05/23

(11)(21) 2 695 645

(12) BREVET CANADIEN
CANADIAN PATENT

(13) C

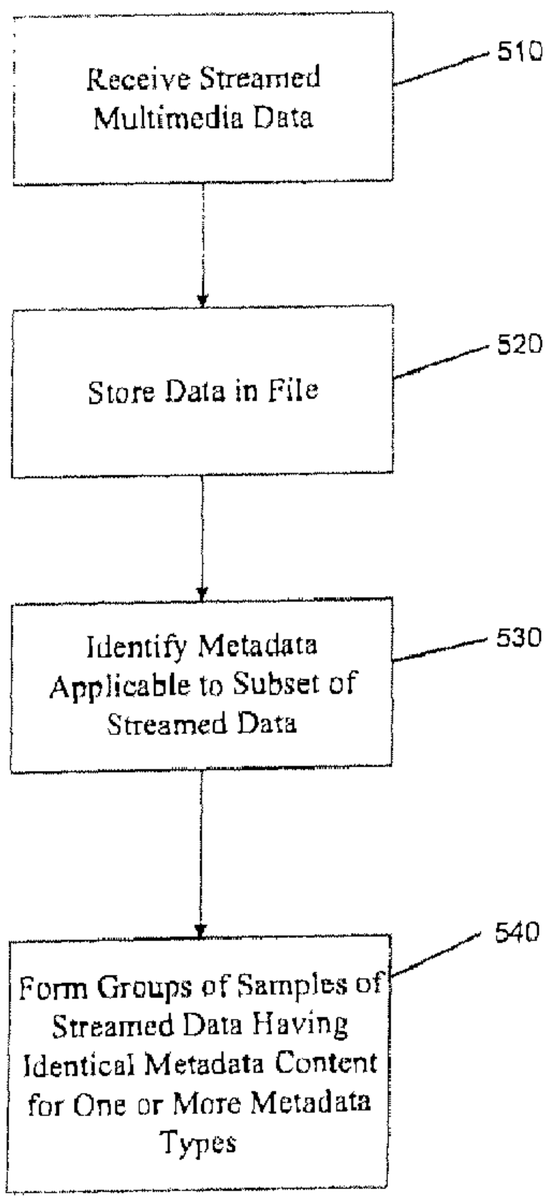
(86) Date de dépôt PCT/PCT Filing Date: 2008/08/19
(87) Date publication PCT/PCT Publication Date: 2009/02/26
(45) Date de délivrance/Issue Date: 2017/05/23
(85) Entrée phase nationale/National Entry: 2010/02/04
(86) N° demande PCT/PCT Application No.: IB 2008/053321
(87) N° publication PCT/PCT Publication No.: 2009/024926
(30) Priorités/Priorities: 2007/08/20 (US60/956,912); 2007/09/10 (US60/971,193); 2007/12/03 (US60/992,057)

(51) Cl.Int./Int.Cl. *G06F 17/30* (2006.01)
(72) Inventeur/Inventor:
HANNUKSELA, MISKA, FI

(73) Propriétaire/Owner:
NOKIA TECHNOLOGIES OY, FI

(74) Agent: SIM & MCBURNEY

(54) Titre : METADONNEES SEGMENTEES ET INDICES POUR DES DONNEES MULTIMEDIAS EN FLUX
(54) Title: SEGMENTED METADATA AND INDEXES FOR STREAMED MULTIMEDIA DATA



(57) Abrégé/Abstract:
A method of organizing streamed data includes storing streamed data in a file, identifying metadata applicable to a subset of the streamed data, and forming at least one group of one or more samples of the streamed data, each sample in a group having

CA 2695645 C 2017/05/23

(11)(21) 2 695 645

(13) C

(57) Abrégé(suite)/Abstract(continued):
identical metadata content for a metadata type. The file may be in accordance with ISO base media file format. The storing streamed data in a file may include storing in a reception hint track. The at least one group may be indicated in a sample group description box. The metadata type may be indicated by a grouping type and grouping instance data, the grouping type specifying semantics of the grouping instance data and the metadata content. The metadata content may comprise a metadata payload and zero or more metadata payload extensions, where the metadata payload is included in a first structure and the zero or more metadata payload extensions are included in a second structure.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
26 February 2009 (26.02.2009)

PCT

(10) International Publication Number
WO 2009/024926 A1

(51) International Patent Classification:
G06F 17/30 (2006.01)

(21) International Application Number:
PCT/IB2008/053321

(22) International Filing Date: 19 August 2008 (19.08.2008)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

60/956,912	20 August 2007 (20.08.2007)	US
60/971,193	10 September 2007 (10.09.2007)	US
60/992,057	3 December 2007 (03.12.2007)	US

(71) Applicant *(for all designated States except US)*: **NOKIA CORPORATION** [FI/FI]; Keilalahdentie 4, FIN-02150 Espoo (FI).

(71) Applicant *(for LC only)*: **NOKIA INC** [US/US]; 6000 Connection Drive, Irving, TX 75039 (US).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: **HANNUKSELA, Miska** [FI/FI]; Linnunlaulunkuja 9 B 5, FIN-36110 Ruutana (FI).

(74) Agent: **ALBERT, JR., G. Peter**; Foley & Lardner, LLP, P.O. Box 80278, San Diego, CA 92130 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI,

[Continued on next page]

(54) Title: SEGMENTED METADATA AND INDEXES FOR STREAMED MULTIMEDIA DATA

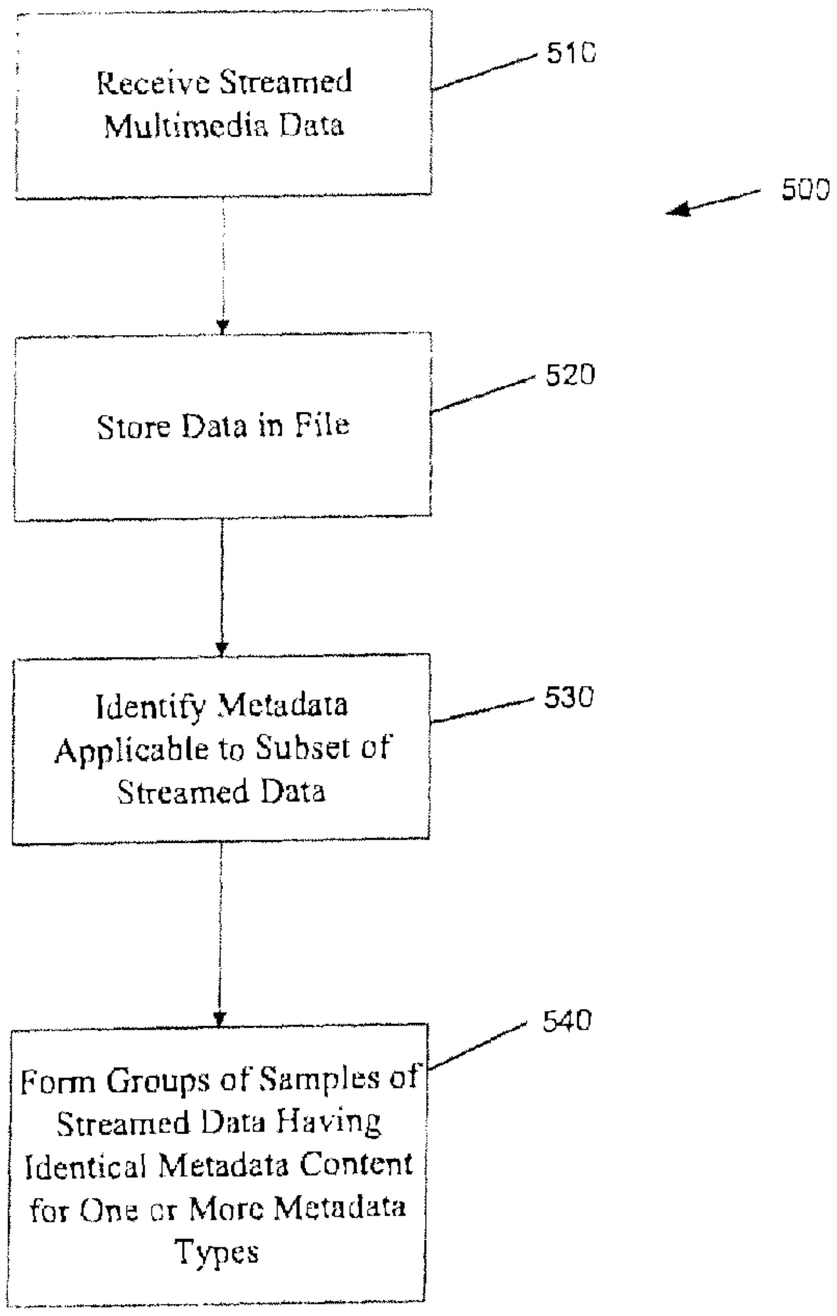


Figure 5A

(57) Abstract: A method of organizing streamed data includes storing streamed data in a file, identifying metadata applicable to a subset of the streamed data, and forming at least one group of one or more samples of the streamed data, each sample in a group having identical metadata content for a metadata type. The file may be in accordance with ISO base media file format. The storing streamed data in a file may include storing in a reception hint track. The at least one group may be indicated in a sample group description box. The metadata type may be indicated by a grouping type and grouping instance data, the grouping type specifying semantics of the grouping instance data and the metadata content. The metadata content may comprise a metadata payload and zero or more metadata payload extensions, where the metadata payload is included in a first structure and the zero or more metadata payload extensions are included in a second structure.

WO 2009/024926 A1

WO 2009/024926 A1

FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, NO, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

SEGMENTED METADATA AND INDEXES FOR STREAMED MULTIMEDIA DATA

FIELD OF THE INVENTION

[0001] The present invention relates generally to the field of streamed multimedia data and, more specifically, to the organization of such multimedia data with metadata and/or indexes.

BACKGROUND OF THE INVENTION

[0002] This section is intended to provide a background or context to the invention that is recited in the claims. The description herein may include concepts that could be pursued, but are not necessarily ones that have been previously conceived or pursued. Therefore, unless otherwise indicated herein, what is described in this section is not prior art to the description and claims in this application and is not admitted to be prior art by inclusion in this section.

[0003] The multimedia container file format is an important element in the chain of multimedia content production, manipulation, transmission and consumption. There are substantial differences between the coding format (a.k.a. elementary stream format) and the container file format. The coding format relates to the action of a specific coding algorithm that codes the content information into a bitstream. The container file format comprises means of organizing the generated bitstream in such way that it can be accessed for local decoding and playback, transferred as a file, or streamed, all utilizing a variety of storage and transport architectures. Furthermore, the file format can facilitate interchange and editing of the media as well as recording of received real-time streams to a file.

[0004] Available media file format standards include ISO base media file format (ISO/IEC 14496-12), MPEG-4 file format (ISO/IEC 14496-14, also known as the MP4 format), AVC file format (ISO/IEC 14496-15) and 3GPP file format (3GPP TS 26.244, also known as the 3GP format). Other formats are also currently in development.

[0005] It is desirable that such media file formats enable the use of metadata. In many applications, such as reception and storage of media streams, such as audio and video streams, it is desirable that the file format enable the use of metadata that is specific to segments of the content of the streamed data to facilitate organization and access to desired sections or segments of the streamed data.

SUMMARY OF THE INVENTION

[0006] Accordingly, in one aspect there is provided a method comprising: causing streamed data to be stored in a file; identifying metadata applicable to one or more samples of the streamed data; causing at least one timed metadata track to be created based on the identified metadata; and forming at least one group from the one or more samples of the streamed data, each sample in a group having identical metadata content for a metadata type.

[0007] In one embodiment, the at least one group is identified in a file.

[0008] In one embodiment, the file is in accordance with ISO base media file format. The streamed data is caused to be stored in a reception hint track. The at least one group may be indicated in a sample group description box.

[0009] In one embodiment, the metadata type includes titles of segments of the streamed data.

[0010] In one embodiment, a group includes samples from two or more tracks. The method may further comprise identifying the group including samples from two or more tracks with metadata associated with all related tracks. Alternatively, the method may comprise identifying the group including samples from two or more tracks with metadata associated with a master track.

[0011] In one embodiment, the at least one group of one or more samples of the streamed data is indicated in a sample group description box for the timed metadata track through associating the at least one group of one or more samples of the streamed data with respective timed metadata samples in the timed metadata track.

[0012] In one embodiment, the timed metadata track associates timing metadata with one or more timelines. The timed metadata track may associate timing metadata with two or more timelines.

[0013] According to another aspect there is provided a non-transitory computer-readable medium storing program code portions therein, the program code portions being configured to, upon execution, cause an apparatus to at least: cause streamed data to be stored in a file; identify metadata applicable to one or more samples of the streamed data; cause at least one timed metadata track to be created based on the identified metadata; and

form at least one group from the one or more samples of the streamed data, each sample in a group having identical metadata content for a metadata type.

[0014] According to another aspect there is provided an apparatus comprising a processor and a memory storing program code, the memory and program code being configured to, with the processor, cause the apparatus to at least: cause streamed data to be stored in a file; identify metadata applicable to one or more samples of the streamed data; cause at least one timed metadata track to be created based on the identified metadata; and form at least one group from the one or more samples of the streamed data, each sample in a group having identical metadata content for a metadata type.

[0014a] According to another aspect there is provided an apparatus, comprising: means for causing streamed data to be stored in a file; means for identifying metadata applicable to one or more samples of the streamed data; means for causing at least one timed metadata track to be created based on the identified metadata; and means for forming at least one group from the one or more samples of the streamed data, each sample in a group having identical metadata content for a metadata type.

[0015] In various embodiments, the metadata type may be indicated by a grouping type and grouping instance data, the grouping type specifying semantics of the grouping instance data and the metadata content. The metadata content may comprise a metadata payload and zero or more metadata payload extensions, where the metadata payload is included in a first structure and the zero or more metadata payload extensions are included in a second structure.

[0015a] According to still yet another aspect there is provided a method, comprising: storing streamed data in a file; identifying metadata applicable to a subset of the streamed data; forming at least one group of one or more samples of the streamed data, each sample in a group having metadata content for a metadata type; identifying the at least one group in a file; and indicating the metadata type by a grouping type and grouping instance data, the grouping type specifying semantics of the grouping instance data and the metadata.

[0016] These and other advantages and features of the invention, together with the organization and manner of operation thereof, will become apparent from the following detailed description when taken in conjunction with the accompanying drawings, wherein like elements have like numerals throughout the several drawings described below.

BRIEF DESCRIPTION OF THE DRAWINGS

[0017] Figure 1 illustrates a graphical representation of an exemplary multimedia communication system within which embodiments of the present invention may be implemented;

[0018] Figure 2 illustrates an exemplary box in accordance with the ISO base media file format;

[0019] Figure 3A is an exemplary box illustrating sample grouping;

[0020] Figure 3B illustrates an exemplary box containing a movie fragment including a SampletoToGroup box;

[0021] Figures 4 illustrates an exemplary box with segmented metadata according to grouping type according to an embodiment of the present invention;

[0022] Figure 5A is a flowchart illustrating an exemplary method of organizing streamed data according to an embodiment of the present invention;

[0023] Figure 5B is a flowchart illustrating an exemplary method of organizing streamed data according to another embodiment of the present invention;

[0024] Figure 6 is a block diagram of an exemplary device for receiving, storing and organizing streamed data according to an embodiment of the present invention;

[0025] Figure 7 is a representation of a simplified box hierarchy indicating the nesting structure for the DVBSampleToGroup box when group_description_index is not referred;

[0026] Figure 8 is an illustration of an example file containing a movie fragment including a DVBSampleToGroup box;

[0027] Figure 9 is an illustration of a simplified box hierarchy indicating the nesting structure for the DVBGroupingInstance, DVBSampleGroupDescription, DVBSampleToGroup, and DVBSampleToGroupExtension boxes; and

[0028] Figure 10 is an illustration of a file including two sample groupings according to the DVBSampleToGroup box for the timecode index.

DETAILED DESCRIPTION OF VARIOUS EMBODIMENTS

[0029] Figure 1 is a graphical representation of a generic multimedia communication system within which various embodiments of the present invention may be implemented. As shown in Figure 1, a data source *100* provides a source signal in an analog, uncompressed digital, or compressed digital format, or any combination of these formats. An encoder 110 encodes the source signal into a coded media bitstream. It should be noted that a bitstream to be decoded can be received

directly or indirectly from a remote device located within virtually any type of network. Additionally, the bitstream can be received from local hardware or software. The encoder 110 may be capable of encoding more than one media type, such as audio and video, or more than one encoder 110 may be required to code different media types of the source signal. The encoder 110 may also get synthetically produced input, such as graphics and text, or it may be capable of producing coded bitstreams of synthetic media. In the following, only processing of one coded media bitstream of one media type is considered to simplify the description. It should be noted, however, that typically real-time broadcast services comprise several streams (typically at least one audio, video and text sub-titling stream). It should also be noted that the system may include many encoders, but in Figure 1 only one encoder 110 is represented to simplify the description without a lack of generality. It should be further understood that, although text and examples contained herein may specifically describe an encoding process, one skilled in the art would understand that the same concepts and principles also apply to the corresponding decoding process and vice versa.

[0030] The coded media bitstream is transferred to a storage 120. The storage 120 may comprise any type of mass memory to store the coded media bitstream. The format of the coded media bitstream in the storage 120 may be an elementary self-contained bitstream format, or one or more coded media bitstreams may be encapsulated into a container file. Some systems operate "live", i.e. omit storage and transfer coded media bitstream from the encoder 110 directly to the sender 130. The coded media bitstream is then transferred to the sender 130, also referred to as the server, on a need basis. The format used in the transmission may be an elementary self-contained bitstream format, a packet stream format, or one or more coded media bitstreams may be encapsulated into a container file. The encoder 110, the storage 120, and the server 130 may reside in the same physical device or they may be included in separate devices. The encoder 110 and server 130 may operate with live real-time content, in which case the coded media bitstream is typically not stored permanently, but rather buffered for small periods of time in the content encoder 110

and/or in the server 130 to smooth out variations in processing delay, transfer delay, and coded media bitrate.

[0031] The server 130 sends the coded media bitstream using a communication protocol stack. The stack may include but is not limited to Real-Time Transport Protocol (RTP), User Datagram Protocol (UDP), and Internet Protocol (IP). When the communication protocol stack is packet-oriented, the server 130 encapsulates the coded media bitstream into packets. For example, when RTP is used, the server 130 encapsulates the coded media bitstream into RTP packets according to an RTP payload format. Typically, each media type has a dedicated RTP payload format. It should be again noted that a system may contain more than one server 130, but for the sake of simplicity, the following description only considers one server 130.

[0032] The server 130 may or may not be connected to a gateway 140 through a communication network. The gateway 140 may perform different types of functions, such as translation of a packet stream according to one communication protocol stack to another communication protocol stack, merging and forking of data streams, and manipulation of data stream according to the downlink and/or receiver capabilities, such as controlling the bit rate of the forwarded stream according to prevailing downlink network conditions. Examples of gateways 140 include multipoint conference control units (MCUs), gateways between circuit-switched and packet-switched video telephony, Push-to-talk over Cellular (PoC) servers, IP encapsulators in digital video broadcasting-handheld (DVB-H) systems, or set-top boxes that forward broadcast transmissions locally to home wireless networks. When RTP is used, the gateway 140 is called an RTP mixer or an RTP translator and typically acts as an endpoint of an RTP connection.

[0033] The system includes one or more receivers 150, typically capable of receiving, de-modulating, and de-capsulating the transmitted signal into a coded media bitstream. The coded media bitstream is transferred to a recording storage 155. The recording storage 155 may comprise any type of mass memory to store the coded media bitstream. The recording storage 155 may alternatively or additively comprise computation memory, such as random access memory. The format of the coded media bitstream in the recording storage 155 may be an elementary self-contained

bitstream format, or one or more coded media bitstreams may be encapsulated into a container file. If there are many coded media bitstreams, such as an audio stream and a video stream, associated with each other, a container file is typically used and the receiver 150 comprises or is attached to a container file generator producing a container file from input streams. Some systems operate "live," i.e. omit the recording storage 155 and transfer coded media bitstream from the receiver 150 directly to the decoder 160. In some systems, only the most recent part of the recorded stream, e.g., the most recent 10-minute excerption of the recorded stream, is maintained in the recording storage 155, while any earlier recorded data is discarded from the recording storage 155.

[0034] The coded media bitstream is transferred from the recording storage 155 to the decoder 160. If there are many coded media bitstreams, such as an audio stream and a video stream, associated with each other and encapsulated into a container file, a file parser (not shown in Figure 1) is used to decapsulate each coded media bitstream from the container file. The recording storage 155 or a decoder 160 may comprise the file parser, or the file parser is attached to either recording storage 155 or the decoder 160.

[0035] The codec media bitstream is typically processed further by a decoder 160, whose output is one or more uncompressed media streams. Finally, a renderer 170 may reproduce the uncompressed media streams with a loudspeaker or a display, for example. The receiver 150, recording storage 155, decoder 160, and renderer 170 may reside in the same physical device or they may be included in separate devices.

[0036] The Digital Video Broadcasting (DVB) organization is currently in the process of specifying the DVB File Format. The primary purpose of defining the DVB File Format is to ease content interoperability between implementations of DVB technologies, such as set-top boxes according to current (DVT-T, DVB-C, DVB-S) and future DVB standards, IP (Internet Protocol) television receivers, and mobile television receivers according to DVB-H and its future evolutions. The DVB File Format will allow exchange of recorded (read-only) media between devices from different manufacturers, exchange of content using USB mass memories or similar

read/write devices, and shared access to common disk storage on a home network, as well as much other functionality.

[0037] The ISO file format is the basis for most current multimedia container file formats, generally referred to as the ISO family of file formats. The ISO base media file format may be the basis for the development of the DVB File Format as well.

[0038] Referring now to Figure 2, a simplified structure of the basic building block 200 in the ISO base media file format, generally referred to as a "box", is illustrated. Each box 200 has a header and a payload. The box header indicates the type of the box and the size of the box in terms of bytes. Many of the specified boxes are derived from the "full box" (FullBox) structure, which includes a version number and flags in the header. A box may enclose other boxes, such as boxes 210, 220, described below in further detail. The ISO file format specifies which box types are allowed within a box of a certain type. Furthermore, some boxes are mandatory to be present in each file, while others are optional. Moreover, for some box types, more than one box may be present in a file. In this regard, the ISO base media file format specifies a hierarchical structure of boxes.

[0039] According to ISO family of file formats, a file consists of media data and metadata that are enclosed in separate boxes, the media data (mdat) box 220 and the movie (moov) box 210, respectively. The movie box may contain one or more tracks, and each track resides in one track box 212, 214. A track can be one of the following types: media, hint or timed metadata. A media track refers to samples formatted according to a media compression format (and its encapsulation to the ISO base media file format). A hint track refers to hint samples, containing cookbook instructions for constructing packets for transmission over an indicated communication protocol. The cookbook instructions may contain guidance for packet header construction and include packet payload construction. In the packet payload construction, data residing in other tracks or items may be referenced (e.g., a reference may indicate which piece of data in a particular track or item is instructed to be copied into a packet during the packet construction process). A timed metadata track refers to samples describing referred media and/or hint samples. For the presentation one media type, typically one media track is selected.

[0040] ISO base media file format does not limit a presentation to be contained in one file, and it may be contained in several files. One file contains the metadata for the whole presentation. This file may also contain all the media data, whereupon the presentation is self-contained. The other files, if used, are not required to be formatted to ISO base media file format, are used to contain media data, and may also contain unused media data, or other information. The ISO base media file format concerns the structure of the presentation file only. The format of the media-data files is constrained the ISO base media file format or its derivative formats only in that the media-data in the media files must be formatted as specified in the ISO base media file format or its derivative formats.

[0041] Movie fragments can be used when recording content to ISO files in order to avoid losing data if a recording application crashes, runs out of disk, or some other incident happens. Without movie fragments, data loss may occur because the file format insists that all metadata (the Movie Box) be written in one contiguous area of the file. Furthermore, when recording a file, there may not be sufficient amount of RAM to buffer a Movie Box for the size of the storage available, and re-computing the contents of a Movie Box when the movie is closed is too slow. Moreover, movie fragments can enable simultaneous recording and playback of a file using a regular ISO file parser. Finally, smaller duration of initial buffering is required for progressive downloading (e.g., simultaneous reception and playback of a file, when movie fragments are used and the initial Movie Box is smaller compared to a file with the same media content but structured without movie fragments).

[0042] The movie fragment feature enables splitting of the metadata that conventionally would reside in the moov box 210 to multiple pieces, each corresponding to a certain period of time for a track. Thus, the movie fragment feature enables interleaving of file metadata and media data. Consequently, the size of the moov box 210 can be limited and the use cases mentioned above be realized.

[0043] The media samples for the movie fragments reside in an mdat box 220, as usual, if they are in the same file as the moov box. For the meta data of the movie fragments, however, a moof box is provided. It comprises the information for a certain duration of playback time that would previously have been in the moov box

210. The moov box 210 still represents a valid movie on its own, but in addition, it comprises an mvex box indicating that movie fragments will follow in the same file. The movie fragments extend the presentation that is associated to the moov box in time.

[0044] The metadata that can be included in the moof box is limited to a subset of the metadata that can be included in a moov box 210 and is coded differently in some cases. Details of the boxes that can be included in a moof box can be found from the ISO base media file format specifications ISO/IEC International Standard 14496-12, Second Edition, 2005-04-01, including Amendments 1 and 2.

[0045] In addition to timed tracks, ISO files can contain any non-timed binary objects in a meta box, or "static" metadata. The meta box can reside at the top level of the file, within a movie box, and within a track box. At most one meta box may occur at each of the file level, movie level, or track level. The meta box is required to contain a 'hdlr' box indicating the structure or format of the "meta" box contents. The meta box may contain any number of binary items that can be referred and each one of them can be associated with a file name.

[0046] In order to support more than one meta box at any level of the hierarchy (file, movie, or track), a meta box container box ('meco') has been introduced in the ISO base media file format. The meta box container box can carry any number of additional meta boxes at any level of the hierarchy (file, move, or track). This allows, for example, the same meta-data to be presented in two different, alternative, meta-data systems. The meta box relation box ("mere") enables describing how different meta boxes relate to each other (e.g., whether they contain exactly the same metadata, but described with different schemes, or if one represents a superset of another).

[0047] Referring now to Figures 3A, 3B and 4, the use of sample grouping in boxes is illustrated. A sample grouping in the ISO base media file format and its derivatives, such as the AVC file format and the SVC file format, is an assignment of each sample in a track to be a member of one sample group, based on a grouping criterion. A sample group in a sample grouping is not limited to being contiguous samples and may contain non-adjacent samples. As there may be more than one

sample grouping for the samples in a track, each sample grouping has a type field to indicate the type of grouping. Sample groupings are represented by two linked data structures: (1) a SampleToGroup box (sbgp box) represents the assignment of samples to sample groups; and (2) a SampleGroupDescription box (sgpd box) contains a sample group entry for each sample group describing the properties of the group. There may be multiple instances of the SampleToGroup and SampleGroupDescription boxes based on different grouping criteria. These are distinguished by a type field used to indicate the type of grouping, as illustrated in Figure 4.

[0048] Figure 3A provides a simplified box hierarchy indicating the nesting structure for the sample group boxes. The sample group boxes (SampleGroupDescription Box and SampleToGroup Box) reside within the sample table (stbl) box, which is enclosed in the media information (minf), media (mdia), and track (trak) boxes (in that order) within a movie (moov) box.

[0049] The SampleToGroup box is allowed to reside in a movie fragment. Hence, sample grouping can be done fragment by fragment. Figure 3B illustrates an example of a file containing a movie fragment including a SampleToGroup box. As illustrated in Figure 4, the SampleGroupDescription Box and SampleToGroup Box may identify a grouping_type based on an index or metadata type.

[0050] A key feature of the DVB file format is known as reception hint tracks, which may be used when one or more packet streams of data are recorded according to the DVB file format. Reception hint tracks indicate the order, reception timing, and contents of the received packets among other things. Players for the DVB file format may re-create the packet stream that was received based on the reception hint tracks and process the re-created packet stream as if it was newly received. Reception hint tracks have an identical structure compared to hint tracks for servers, as specified in the ISO base media file format. For example, reception hint tracks may be linked to the elementary stream tracks (i.e., media tracks) they carry, by track references of type 'hint'. Each protocol for conveying media streams has its own reception hint sample format.

[0051] Servers using reception hint tracks as hints for sending of the received streams should handle the potential degradations of the received streams, such as transmission delay jitter and packet losses, gracefully and ensure that the constraints of the protocols and contained data formats are obeyed regardless of the potential degradations of the received streams.

[0052] The sample formats of reception hint tracks may enable constructing of packets by pulling data out of other tracks by reference. These other tracks may be hint tracks or media tracks. The exact form of these pointers is defined by the sample format for the protocol, but in general they consist of four pieces of information: a track reference index, a sample number, an offset, and a length. Some of these may be implicit for a particular protocol. These 'pointers' always point to the actual source of the data. If a hint track is built 'on top' of another hint track, then the second hint track must have direct references to the media track(s) used by the first where data from those media tracks is placed in the stream.

[0053] Conversion of received streams to media tracks allows existing players compliant with the ISO base media file format process DVB files as long as the media formats are also supported. However, most media coding standards only specify the decoding of error-free streams, and consequently it should be ensured that the content in media tracks can be correctly decoded. Players for the DVB file format may utilize reception hint tracks for handling of degradations caused by the transmission, i.e., content that may not be correctly decoded is located only within reception hint tracks. The need for having a duplicate of the correct media samples in both a media track and a reception hint track can be avoided by including data from the media track by reference into the reception hint track.

[0054] Currently, three types of reception hint tracks are being specified: MPEG-2 transport stream (MPEG2-TS), Real-Time Transport Protocol (RTP), and Real-Time Transport Control Protocol (RTCP) reception hint tracks. Samples of an MPEG2-TS reception hint track contain MPEG2-TS packets or instructions to compose MPEG2-TS packets from references to media tracks. An MPEG-2 transport stream is a multiplex of audio and video program elementary streams and some metadata information. It may also contain several audiovisual programs. An RTP reception hint

track represents one RTP stream, typically a single media type. An RTCP reception hint track may be associated with an RTP reception hint track and represents the RTCP packets received for the associated RTP stream.

[0055] RTP is used for transmitting continuous media data, such as coded audio and video streams in networks based on the Internet Protocol (IP). The Real-time Transport Control Protocol (RTCP) is a companion of RTP, i.e. RTCP should be used to complement RTP always when the network and application infrastructure allow. RTP and RTCP are usually conveyed over the User Datagram Protocol (UDP), which, in turn, is conveyed over the Internet Protocol (IP). There are two versions of IP, IPv4 and IPv6, differing by the number of addressable endpoints among other things. RTCP is used to monitor the quality of service provided by the network and to convey information about the participants in an on-going session. RTP and RTCP are designed for sessions that range from one-to-one communication to large multicast groups of thousands of endpoints. In order to control the total bitrate caused by RTCP packets in a multiparty session, the transmission interval of RTCP packets transmitted by a single endpoint is proportional to the number of participants in the session. Each media coding format has a specific RTP payload format, which specifies how media data is structured in the payload of an RTP packet.

[0056] The metadata requirements for the DVB file format can be classified to four groups based on the type of the metadata: 1) sample-specific timing metadata, such as presentation timestamps; 2) indexes; 3) segmented metadata; and 4) user bookmarks (e.g., of favorite locations in the content).

[0057] For sample-specific timing metadata, there can be different timelines to indicate sample-specific timing metadata. Timelines may be required to cover the entire length of the recorded streams. Further, timelines may be paused. For example, timeline A may be created in the final editing phase of a movie. A service provider or another entity may insert commercials and provided timeline B for the commercials. Timeline A may be paused while commercials are ongoing. Timelines may also be transmitted after the content itself. In one embodiment, a timeline samples may be carried within MPEG-2 program elementary streams (PES). A PES conveys an elementary audio or video bitstream, and hence, timelines are accurately

synchronized with audio and video frames. In this regard, reference is made to ETSI TS 102 823, "Specification for the carriage of synchronized auxiliary data".

[0058] Indexes may include, for example, video access points and trick mode support (e.g., fast forward/backward, slow-motion). Such operations may require, for example, indication of self-decodable pictures, decoding start points, and indications of reference and non-reference pictures.

[0059] In the case of segmented metadata, the DVB services may be described with a service guide according to a specific metadata schema, such as Broadcast Content Guide (BCG), TV-Anytime, or Electronic Service Guide (ESG) for IP datacasting (IPDC). The description may apply to a part of the stream only. Hence, the file may have several descriptive segments (e.g., a description about that specific segment of the program, such as "Holiday in Corsica near Cargese") information.

[0060] In addition, the metadata and indexing structures of the DVB file format are required to be extensible and user-defined indexes are required to be supported. Additionally, there should be a method for fast access to elements in the file (e.g., via index tables). Furthermore, it should be possible to signal the accuracy of any index. For example, the index might be perfectly accurate, accurate to ± a specified amount or a heuristic guess only.

[0061] Various techniques for performing indexing and implementing segmented metadata have been proposed. Such techniques include, for example, sample events and sample properties. The sample event mechanism enables the listing of samples associated with a particular event type and event description index in one data structure, referred to as the SampleToEvent box. Each entry in the list may be additionally accompanied by a value or description, having a determined size in terms of bytes. Samples are listed with reference to their sample number within the associated track. It is not required to have all samples listed for a particular event type. The allowed values of the event type are predetermined and specified, e.g., in the DVB File Format specification. For each used event type in any SampleToEvent box, there is a respective SampleEventDescription box associated with the same event type. A SampleEventDescription box contains a list of sample event description

entries, each associated with a unique event description index (staring from 1 and *incremented by 1 per each entry in the list*). The syntax and semantics of sample event description entries are determined by the event type and are specified e.g. in the DVB File Format specification. The semantics specified for a particular sample description index and event type apply to those samples that are associated with that sample description index and event type in SampleToEvent boxes. The SampleToEvent box is allowed to reside in a movie fragment. Hence, a listing of sample events can be performed fragment by fragment. There may be multiple instances of the SampleToEvent and SampleEventDescription boxes for different event types, but there can be at most one SampleToEvent box of a particular event type per a movie fragment and at most one SampleEventDescription box of a particular event type within a file.

[0062] Syntax of the SampleToEvent box and the SampleEventDescription box is specified as follows:

```
aligned(8) class SampleToEventBox extends FullBox(„steb“) {
  unsigned int(32) event_type;
  unsigned int(32) entry_count;
  for(i=1; i<=entry_count; i++) {
   unsigned int(32) event_desc_index;
   unsigned int(32) sample_count;
   for(j=1; j<=sample_count; j++) {
    unsigned int(32) sample_number;
    unsigned int(8) value[event_data_length];
   }
  }
}
aligned(8) class SampleEventDescriptionBox extends FullBox(„sedb“) {
  unsigned int(32) event_type;
  unsigned int(32) entry_count;
  for(i=1; i<=entry_count; i++) {
   SampleEventDescriptionEntry();
```

}
}

[0063] Semantics of the syntax elements used in the SampleToEvent box and the SampleEventDescription box are specified as follows:
event_type is a 32-bit unsigned integer which shall contain the four-character code of an event type, built of ASCII characters.
entry_count is a 32-bit unsigned integer with the number of entries in the following loop.
event_desc_index is a 32-bit unsigned integer with the index of the SampleEventDescriptionEntry.
sample_count is a 32-bit unsigned integer with the number of entries in the following loop which contains the sample numbers of the samples the event applies to.
value is an array of event-specific data. The length of this field is specified by event_data_length in the corresponding SampleEventDescriptionEntry. This field is constant size for all indexes in this loop and padding shall be applied for variable-length indexes.

[0064] In unicast, multicast, and broadcast streaming applications the available streams are announced and their coding formats are characterized to enable each receiver to conclude if it can decode and render the content successfully. Sometimes a number of different format options for the same content are provided, from which each receiver can choose the most suitable one for its capabilities and/or end-user wishes. The available media streams are often described with the corresponding media type and its parameters that are included in a session description formatted according to the Session Description Protocol (SDP). In unicast streaming applications the session description is usually carried by the Real-Time Streaming Protocol (RTSP), which is used to set up and control the streaming session. In broadcast and multicast streaming applications, the session description may be carried as part of the electronic service guide (ESG) for the service. The SDP information for a broadcast stream may be updated during the broadcast. Hence, SDP for a recorded stream should be stored in such a manner that it can be updated later.

[0065] Referring now to Figure 5A, a method 500 according to an embodiment of the invention by which streamed data is advantageously stored and organized is illustrated. According to an embodiment of the invention, streamed media is received at, for example, a receiving unit or another device (block 510). The streamed data, which may be at least one media or multimedia stream, is stored in a file (block 520). In terms of the ISO base media file format, the at least one received media or multimedia stream may be stored in a reception hint track in a file. Metadata which is applicable to a subset of the streamed data is identified (block 530). For example, metadata applicable to certain samples of the reception hint track may be identified. Groups of samples of the streamed data having identical metadata content of a certain metadata type are formed and identified in a file (block 540). In this regard, sample grouping based on a certain metadata type may be indicated with the sample group description box, and samples can be mapped to certain metadata content included in the sample group description box with the sample-to-group box. In another embodiment, a certain metadata type is considered to comprise both the type indicating the syntax and semantics of the metadata content and a part of the metadata content that remains unchanged in the grouping. For example, the metadata type can indicate timeline metadata, and the part of the metadata remaining unchanged in the grouping can comprise an identifier of the timeline.

[0066] Referring now to Figure 5B, a method 505 according to another embodiment of the invention is illustrated. In the embodiment of Figure 5B, streamed media is received at, for example, a receiving unit or another device (block 515). The streamed data, which may be at least one media or multimedia stream, is stored in a file and associated with a reception hint track in a file (block 525). At least one metadata sample applicable to a reception hint sample is identified (block 535). The at least one metadata sample is then stored in the file and associated with a timed metadata track (block 545). Groups of samples of the streamed data having identical metadata content of a certain metadata type are formed and identified in a file (block 555). Sample grouping based on a certain metadata type is indicated with the sample group description box, and metadata samples can be mapped to certain metadata content included in the sample group description box with the sample-to-group box.

As above, a certain metadata type can also be considered to comprise both the type indicating the syntax and semantics of the metadata content and a part of the metadata content that remains unchanged in the grouping. It is noted that the sequential processing blocks suggested in Figure 5B can be applied to different granularities of multimedia data and metadata, ranging from a subset of a sample to an entire stream. Different blocks can also apply to a different granularity of data. In one arrangement, blocks 515, 525, 535, 545 are iteratively applied to individual reception hint samples and metadata samples.

[0067] One grouping type is defined per metadata or index type in the DVB file format specification. Examples of potential grouping types include the following:

- Any metadata specified by handler_type used for the meta box. Each sample group is associated with one item of the metadata scheme. The metadata scheme may be one of those specified in DVB specifications (BCG, TV-Anytime, IPDC ESG);
- Any index type (e.g., those proposed in DVB document TM-FF0035, such as indication of self-decodable pictures, decoding start points, and indications of reference and non-reference pictures); and
- SDP information for received RTP streams.

[0068] Referring now to Figure 6, an exemplary device according to an embodiment of the present invention is illustrated. The device 600, such as a receiver, is provided with a DVB file generator 610, a storage unit 620, and a processor 630. Of course, those skilled in the art will understand that the various components may be separated into further components or combined into fewer components. Further, the various components may include executable code for performing their functions. Upon receipt of streamed data, the DVB file generator 610 creates a DVB file from received packets. In this regard, the DVB file generator 610 receives the packet stream(s) themselves and associated metadata (such as BCG, TV-Anytime, or IPDC ESG). The device 600 stores the received packets to a file (in an mdat section of a file, if the moov and mdat boxes reside in the same file) as samples of a reception hint track in the storage unit 620. The receiver 600 may also derive file metadata that will be later stored within the moov box or the moof box, depending on whether or not movie fragments are in use. For any received piece of metadata, the receiver 600 (e.g., the

processor 630) analyzes its type and creates a new sample group description box, if the type of metadata did not exist before. The receiver 600 also checks whether or not the content of the piece of metadata differs from all metadata stored earlier in the file. If the content is novel, then a new entry is created for the sample group description box and the metadata content is copied into the entry. The SampleToGroup box is maintained for each grouping type and samples of the reception hint track are mapped to one of the entries in the respective sample group description box.

[0069] In another embodiment, the DVB file generator 610 of the device 600 creates one or more reception hint tracks and file metadata for the moov box or the moof box. The device 600 also derives indexes, such as a randomly accessible intra picture indication, from received packets. This may be performed by the DVB file generator 610 or by another component of the device 600, such as the processor 630 or another component not shown in Figure 6. If one or more indexes or pieces of metadata are derived or received for a received packet, the device 600 creates a timed metadata sample containing the indexes or the pieces of metadata. Again, this may be performed by a component of the device 600, such as the DVB file generator 610 or the processor 630. The timed metadata sample is associated with a timed metadata track, and the timed metadata track is associated with the reception hint track. For any received or derived piece of metadata or index, the processor 630 of the device 600 analyzes its type and creates a new sample group description box, if the type of metadata did not exist before. The processor 630 also checks whether or not the content of the piece of metadata differs from all metadata stored earlier in the file. If the content is novel, then a new entry is created for the sample group description box and the metadata content is copied into the entry. The SampleToGroup box is maintained for each grouping type and samples of the timed metadata track are mapped to one of the entries in the respective sample group description box.

[0070] In this embodiment, one metadata sample can be created for every reception hint sample, regardless of whether or not an index or a piece of metadata is associated with that reception hint sample. The payload of the corresponding metadata sample indicates if no index or piece of metadata is associated with a reception hint sample. Further, potential track fragments of a timed metadata track can be aligned to the

corresponding reception hint track in terms of the first and last included sample in the fragments. It can be indicated within a file in a track-header-level box or structure whether or not there is one timed metadata sample per each reception hint sample, and whether or not track fragments of a timed metadata track are identical to those of the associated reception hint track. When one metadata sample is created per each reception hint sample, corresponding track fragments of a timed metadata track and a reception hint track may contain samples having identical sample numbers.

[0071] It is noted that the creation of sample grouping may be implemented in a number ways. For example, the creation of sample grouping may not happen simultaneously with the reception and storage of the samples themselves. Instead, sample grouping may be performed off-line, based on an intermediate format for storing metadata or based on the timed metadata track. Referring to Figure 5A, blocks 530 and 540 would happen after the reception of the stream has been completed. Referring to Figure 5B, block 555 would happen after the reception of the stream has been completed. Referring again to Figure 1, the off-line creation of sample grouping would result into two additional blocks getting their input from the recording storage 155 and outputting to the decoder 160. The first block in processing order can be referred to as a file re-writer, which inputs a file without sample grouping for a particular metadata type and outputs a file with that type of sample grouping. The second block in processing order can be referred to as a second recording storage, which may have similar properties to the recording storage 155.

[0072] The file re-writer and the second recording storage referred to above may reside in the same device as the receiver 150, the recording storage 155, the decoder 160, or a different device. Moreover, the file re-writer and the second recording storage may reside in the same device or different devices from each other.

[0073] Received streams can be of different nature – they may contain several media types and programs in a multiplexed format (MPEG-2 transport streams) or they may contain a single media type of a single program (RTP streams). Consequently, the receiver operation and file format structures are adapted accordingly as follows.

[0074] It should be noted that the various embodiments of the present invention are not limited to any particular type of sample grouping, such as the sample grouping specified in ISO/IEC 14496-12. The following embodiment describes a sample grouping mechanism that can be applied with various embodiments of the invention.

[0075] In addition to the grouping type, a sample grouping may be characterized by zero or more global parameters. Examples of global parameters include a program identifier (PID) within an MPEG-2 transport stream and a timeline identifier. When parsing sample groups, only a particular value of the global parameter is of interest at a time, e.g., a desired timestamp within a particular timeline. In addition to enumerated sample group description entries, it can advantageous to allow local parameters that may change for each entry in the SampleToGroup box. For example, a timecode can be associated with an entry in the SampleToGroup box. When parsing a sample grouping, different values of local parameters are interleaved into the same structure, i.e. in the same SampleToGroup box.

[0076] Syntax of a sample grouping mechanism of this embodiment is presented below.

```
aligned(8) class DVBSampleToGroupBox
extends FullBox('dstg', version = 0, flags)
{
        unsigned int(32) grouping_type;
        if (flags & 1) {
                unsigned int(8) info_length;
                unsigned int(8) instance_info[info_length];
        }
        if (flags & 16) {
                unsigned int(8) static_length;
                unsigned int(8) static_info[static_length];
}
if (flags & 2)
        unsigned int(8) payload_length;
unsigned int(32) entry_count;
```

```
for (i=1; i <= entry_count; i++) {
        if (flags & 4)
                unsigned int(32) sample_number;
        if (!(flags & 8))
                unsigned int(32) sample_count;
        if (flags & 2)
                unsigned int(8) payload[payload_length];
        else
                unsigned int(32) group_description_index;
    }
}
```

[0077] The presented DVBSampleToGroup box is structurally compatible with the SampleToGroup box, i.e., if flags (of the box header) are 0, then the syntax is identical to the SampleToGroup box. Global parameters as specified above are represented by grouping instance information (instance_info). static_info represents static values remaining unchanged for all grouped samples. The semantics of static_info depend on grouping_type. For example, static_info may be formatted according to the payload of the DVBIndexBox (presented subsequently), giving timing and sample number inaccuracy ranges. Local parameters are represented by payload or group_description_index. The DVBSampleToGroup box can be parsed without semantic knowledge of grouping_type thanks to info_length, static_length, and payload_length. A file editor software program or device that is unaware of a particular grouping_type can edit the box (e.g. as response of removal or insertion of samples), even if it did not have knowledge about the semantics of a particular grouping_type. sample_number indicates the first sample for which the payload or group_description_index apply, and the persistence of the payload or group_description_index in terms of number of consecutive samples is indicated by sample_count. If sample_number is not present, then the first sample for which the payload or group_description_index apply is subsequent to the last sample for which the previous entry in the loop applied. Values of sample_count in a DVBSampleToGroup box is increasing in appearance order. A sample can be

included only once in a DVBSampleToGroup box, i.e., the sum of previous values of sample_number and sample_count is less than the present value of sample_number. It is not required that each sample in a track is mapped in a DVBSampleToGroup box. The DVBSampleGroupDescription box and group_description_index can be used for enumerated indexes. The value of payload or group_description_index for those samples that are not mapped in a DVBSampleToGroup box is unspecified. The semantics of instance_info and payload are specified for each grouping_type.

[0078] An alternative definition for the syntax of the DVBSampleToGroup box is presented below. A primary difference compared to the previous case is that varying-length payloads are enabled.

```
aligned(8) class DVBSampleToGroupBox
extends FullBox('dstg', version = 0, flags)
{
        unsigned int(32) grouping_type;
    if (flags & 1) {
            unsigned int(8) info_length;
            unsigned int(8) instance_info[info_length];
    }
    if (flags & 16) {
            unsigned int(8) static_length;
            unsigned int(8) static_info[static_length];
    }
    if (flags & 2)
            unsigned int(8) payload_length;
            unsigned int(32) entry_count;
    for (i=1; i <= entry_count; i++) {
            if (flags & 4)
                    unsigned int(32) sample_number;
            if (!(flags & 8))
                    unsigned int(32) sample_count;
```

```
if (flags & 2)
        unsigned int(8) payload[payload_length];
else if (flags & 32) {
        unsigned int(8) varying_length;
        unsigned int(8) payload[varying_length];
    }
    else
        unsigned int(32) group_description_index;
  }
}
```

[0079] Figure 7 provides a simplified box hierarchy indicating the nesting structure for the DVBSampleToGroup box 700 when group_description_index is not referred ((flags & 2) is equal to 1). DVBSampleToGroup boxes 700 reside within the sample table (stbl) box 710, which is enclosed in the media information (minf) 720, media (mdia) 730, and track (trak) boxes 740 (in that order) within a movie (moov) box 210.

[0080] The DVBSampleToGroup box 700 is allowed to reside in a movie fragment. Hence, sample grouping can be done fragment by fragment. Figure 8 illustrates an example of a file containing a movie fragment 800 including a DVBSampleToGroup box 700.

[0081] Syntax of DVBSampleGroupDescription box is presented below.

```
aligned(8) class DVBSampleGroupDescriptionBox
(unsigned int(32) handler_type)
extends FullBox('dsgd', version, flags)
{
        unsigned int(32) grouping_type;
    if (flags & 1) {
        unsigned int(8) info_length;
        unsigned int(8) instance_info[info_length];
    }
    if (version>=1)
```

```
        unsigned int(32) default_length;
    unsigned int(32) entry_count;
    int i;
    for (i = 1 ; i <= entry_count ; i++){
    if (version>=1 && default_length==0)
        unsigned int(32) description_length;
    switch (handler_type){
        case 'vide': // for video tracks
            VisualSampleGroupEntry ();
            break;
        case 'soun': // for audio tracks
            AudioSampleGroupEntry();
            break;
        case 'hint': // for hint tracks
            HintSampleGroupEntry();
            break;
        }
    }
}
```

[0082] The structure of the DVBSampleGroupDescription box is compatible with the SampleGroupDescription box, i.e., if flags (in the box header) are set to 0, the syntax is identical to the SampleGroupDescription box. Global parameters as defined above are represented by grouping instance information (instance_info). The inclusion of instance_info in the DVBSampleGroupDescription box enables different enumerations (i.e., sample group entries) for different values of the global parameters.

[0083] The syntax presented above for the DVBSampleToGroup box only allows a fixed-length payload or group_description_index being specified for each entry in the loop. It may be desirable for some grouping types to have varying-length payloads for any entry specified in a particular DVBSampleToGroup box. The following extension mechanism, referred to as DVBSampleToGroupExtension box is depicted below for this purpose.

```
aligned(8) class DVBSampleToGroupExtensionBox
extends FullBox('dsge', version = 0, flags)
{
        unsigned int(32) grouping_type;
if (flags & 1) {
        unsigned int(8) info_length;
        unsigned int(8) instance_info[info_length];
}
if (flags & 2)
        unsigned int(32) extension_type;
if (!(flags & 4))
        unsigned int(16) ext_length;
unsigned int(32) entry_count;
for (i=1; i<=entry_count; i++) {
        if (flags & 4)
                unsigned int(16) ext_length;
        unsigned int(8) extension[ext_length];
        }
}
```

[0084] Each entry in the loop corresponds to the respective entry (having the same loop counter value) in the DVBSampleToGroup box having identical grouping_type and global parameter values (i.e., identical values of info_length and instance_info). In other words, extension applies to the same samples as payload when the loop counter value is the same in respective DVBSampleToGroupExtension and DVBSampleToGroup boxes. If extension_type is not present, then the default semantics for an extension of a grouping_type are applied. When extension_type is present, then it determines the semantics of extension. Similarly to the DVBSampleToGroup box, a DVBSampleToGroupExtension box of a particular global parameter and extension_type can be included for each movie fragment.

[0085] In addition to varying-length local parameters for sample groups, the DVBSampleGroupExtension box can be used for providing new pieces of information

for a sample grouping while maintaining compatibility with existing sample group definitions and parsers. In other words, a parser not capable of interpreting the DVBSampleToGroupExtension box or a particular extension_type can still parse the corresponding DVBSampleToGroup box successfully.

[0086] In one embodiment, the instance_info used above in the syntax for DVBSampleToGroup, DVBSampleGroupDescription, and DVBSampleToGroupExtension boxes is an index to the loop included in the DVBGroupingInstance box having an identical value of grouping_type as the aforementioned boxes. The syntax of the DVBGroupingInstance box can be specified as follows. The semantics of instance_descr are determined by grouping_type.

```
aligned(8) class DVBGroupingInstanceBox
extends FullBox('dgri', version = 0, flags)
{
        unsigned int(32) grouping_type;
        unsigned int(32) default_length;
        unsigned int(32) entry_count;
        for (i=1; i<=entry_count; i++) {
                if (default_length == 0) {
                        unsigned int(32) descr_length;
                        unsigned int(8) instance_descr[descr_length];
                }
                else
                        unsigned int(8) instance_descr[default_length];
        }
}
```

[0087] In one embodiment, extension_type can be parameterized similarly to global and local parameters discussed previously. In other words, rather than identifying an extension based on its extension_type only, it can be associated with a parameter further specifying its intent and scope. The associated parameter can also be an index to a loop in another box, which contains the actual parameter values.

[0088] Figure 9 provides a simplified box hierarchy indicating the nesting structure for the DVBGroupingInstance box 900, DVBSampleGroupDescription box 910, DVBSampleToGroup box 700, and DVBSampleToGroupExtension box 920. The above-identified boxes reside within the sample table (stbl) box 710, which is enclosed in the media information (minf) 720, media (mdia) 730, and track (trak) boxes 740 (in that order) within a movie (moov) box 210. Arrows 950, 960 and 970 indicate a logical connection. Arrows 950 indicate that DVBSampleGroupDescription box 910, DVBSampleToGroup box 700, and DVBSampleToGroupExtension box 920, if present, refer to the DVBGroupingInstance box 950 (with certain values of grouping_type). Arrow 960 indicates that with certain grouping_type and value of flags a DVBSampleToGroup box 700 refers to the respective DVBSampleGroupDescription box 910. Arrow 960 indicates that a DVBSampleToGroupExtension box 920 always refers to the respective DVBSampleToGroup box 700.

[0089] When sample grouping is applied to MPEG2-TS reception hint track (or any other multiplexed stream), the following holds:

- Sample groups associated with the reception hint track are capable of representing metadata for the audio-video multiplex, e.g., toextual descriptive metadata;
- Indices and metadata for a single media type can be represented by specific grouping types;
- Approximate indices can be represented with specific grouping types. Alternatively or complementarily, the accuracy of the indices can be indicated by including more than one contiguous sample into a sample group, which indicates that the indexed feature should be true in one of the contiguous samples. For example, an approximate random access point could be indicated by including N contiguous samples to a sample group of type random access point, indicating that one of the N samples should be a decoding start position.

[0090] When sample grouping is applied to a media track or RTP reception hint track, indices and metadata for a single media type can be represented by specific grouping types. However, as sample grouping mechanism operates only within one

track, metadata that applies to more than one media track or reception hint track (later referred to as "multi-track" metadata) should be represented. This issue can materialize in two cases. First, both media tracks and reception hint track(s) may be created from the received stream. Second, there may be multiple RTP reception hint tracks. It is suggested that the sample grouping mechanism of the invention is associated with reception hint tracks whenever they are present in the file. Thus, the first case above can be avoided. For the second case, several options are contemplated.

[0091] First, "multi-track" metadata may be associated with any related track. When the metadata is associated to a reception hint track or a media track, the metadata implicitly applies to all reception hint tracks or media tracks, respectively, within the same period of decoding time. This alternative requires parsers to check all tracks for potential "multi-track" metadata.

[0092] Alternatively, "multi-track" metadata may be associated with a "master" track. The master track may be explicitly indicated with a new box within the moov box, for example. The new box may also list all the inter-linked tracks for the "multi-track" metadata.

[0093] In another embodiment, timed metadata track may be associated with many media tracks or RTP reception hint tracks. Sample grouping may be applied to the timed metadata track. The referred media and reception hint tracks may be listed as track references of type 'cdsc'. This serves as an explicit indication which tracks the metadata pertains to. If used, sample duration can be set to match the persistence of each metadata item.

[0094] In one embodiment of the present invention, in the embodiment of the "master" track described above, the new box is referred to as a track relation box and indicates relationship between tracks. The track relation box is defined as follows:

Box Type: 'trel'
Container: Movie Box ('moov')
Mandatory: No
Quantity: Zero or One

[0095] The syntax for the track relation box is as follows:

```
aligned(8) class TrackRelationBox
        extends FullBox('trel', version = 0, flags) {
        int i,j;

        if(flags & 0x000001 == 1) {
                unsigned int(16) num_reception_groups;
                for(i=1; i<=num_reception_groups; i++) {
                        unsigned int(16) num_tracks_in_reception_group;
                        for(j=1; j<=num_tracks_in_reception_group; j++)
                                unsigned int(32) track_id;
                }
        }
}
```

[0096] In the above syntax, "version" is an integer that specifies the version of the track relation box (0 as described above), and "flags" is a 24-bit integer with flags. The following bits are defined, where bit 0 is the least significant bit, bit 1 is the second least significant bit, etc. When bit 0 is equal to 1, this indicates that information of reception groups is present in this box. When bit 0 is equal to 0, this indicates that information of reception groups is not present in this box. It is obvious for a person skilled in the art that other types of track relations could also be specified and indicated with a different bit number than bit 0.

[0097] "num_reception_groups" indicates the number of reception groups that are signaled. A reception group comprises reception hint tracks that are received simultaneously or are otherwise part of the same recording. In addition, a reception group may comprise media tracks that are derived based on reception hint tracks. "track_id" indicates the track identifier of the j-th track in the i-th reception group that is signaled. The first track_id of a reception group is a master track that may contain

segmented metadata that is indicated with sample groups and is associated jointly with all the tracks in the reception group. No other track than a master track may be allowed to contain segmented metadata associated with all the tracks in the reception group.

[0098] When timed metadata tracks for indexes or segmented metadata are created, in one embodiment, the following practices may be followed.

[0099] One timed metadata track is created for program-specific indexes and metadata of a single-program MPEG-2 transport stream. Program-specific indexes and metadata apply equally to audio and video streams of a program and to any other potential components of the program, such as subtitle streams.

[0100] One timed metadata track per program is created for program-specific indexes and metadata of a multi-program MPEG-2 transport stream. In this regard, a timed metadata track may contain metadata of only one program. The program can be identified by its program_number value, which is a 16-bit unique identifier for programs within an MPEG-2 transport stream, used in program association and program map tables of an MPEG-2 transport stream, for example. In one embodiment, the parameter program_number can be included in the sample entry structure for timed metadata tracks associated with MPEG2-TS reception hint tracks.

[0101] One timed metadata track is created for media-specific indexes of each elementary stream of an MPEG2-TS program. Media-specific indexes may apply only to a particular media type. For example, they can be indications of reference and non-reference frames of video or indications of the temporal scalability level of video.

[0102] One timed metadata track may be created for media-specific indexes for an RTP stream. Further, one timed metadata track may be created for program-specific indexes of multiple RTP streams. The timed metadata track is associated with the RTP reception hint tracks using track references. In another embodiment, the timed metadata track is associated with the "master" reception hint track with a track reference and the other associated reception hint tracks are indicated through the TrackRelationBox as presented above.

[0103] Even though one program-specific timed metadata track and one media-specific timed metadata track per elementary media stream may often be preferable,

more than one timed metadata tracks can be created. For example, if an alternative timeline for the program is provided subsequently to the program itself, it is more practical from the file arrangement point of view to create a new timed metadata track for the provided timeline.

[0104] For embodiments using timed metadata tracks, a receiver in accordance with an embodiment of the present invention can operate as follows as a response to each received packet:

- Convert the received packet to a reception hint sample in the mdat box.
- Derive indexes and segmented metadata. Write associated metadata sample(s), if any, to the mdat box (immediately after the corresponding reception hint sample).
- Update boxes within the track header of the reception hint track.
- Update boxes within the track header of the timed metadata track.
- If the memory reserved for track header is about to be fully occupied (and cannot be dynamically re-allocated), start a new movie fragment.

[0105] A receiver with a greater amount of buffer memory may arrange several metadata samples and reception hint samples in continuous chunks of memory and, therefore, save in the storage space required for the sample to chunk box and the chunk offset box.

[0106] Indexes and segmented metadata have the following characteristics when it comes to reception hint samples that are associated with them. An index may indicate a characteristic to be valid from the associated reception hint sample onwards, usually until the next index of the same type. For example, an index may indicate a polarity change of scrambling in MPEG-2 transport stream. An index may indicate a characteristic of a single reception hint sample or an event that is synchronized with a reception hint sample. A bookmark is an example of such an index.

[0107] An index may indicate a characteristic of the stream in between the associated reception hint sample and the previous reception hint sample. An indication of missing packets is such an index.

[0108] An index may indicate a characteristic of a coded media sample. It is noted that timed metadata tracks proposed in this invention are associated to reception hint

samples, reception hint samples do not usually contain exactly one media sample, and data for one media sample may or may not reside in contiguous reception hint samples (e.g., because elementary audio and video streams are multiplexed in an MPEG-2 transport stream). Consequently, there are at least two options how to index media samples: (1) an index is associated only with the first reception hint sample containing data for a media sample; and (2) an index is associated with all reception hint samples containing data for a media sample.

[0109] Sample groups associated with timed metadata tracks that contain indexes and segmented metadata can be created in any of the following ways:

- A sample group can indicate the value of the index. For example, video sample characteristics can be indicated (random access intra picture, temporal level of the picture). In one embodiment, the sample group description box contains the allowed or used values for the index. In another embodiment, the payload or the extension fields included in a DVBSampleToGroup box or DVBSampleToGroupExtension box, respectively, can indicate the values of indexes.
- A sample group can indicate whether or not the index of a particular type changed compared to the previous sample, i.e., the persistence of indexes. For example, changes of protection key for encrypted content can be indicated. In one embodiment, the DVBSampleToGroupExtension box can be used to contain the values of indexes when the sample to group box only contained an indication whether the index value changed.
- A sample group can indicate bookmarks. The sample group description box contains the bookmarks themselves.
- A sample group can indicate titles and other information for segmented metadata. The sample group description box contains the titles. One mechanism for linking structured metadata according to any particular metadata schema to sample groups is as follows. The metadata section formatted according to any particular metadata schema is included as an item in a meta box associated with the timed metadata track, the movie, or the whole file. Rather than including the metadata section as such in the sample group description box, the sample group description box contains links (typically as item identifiers) to the items containing the associated

metadata section.

° A sample group can indicate the value of a set (or tuple) of indexes or segmented metadata of any characteristic. For example, a sample group can indicate the value of a pair of indexes, such as temporal level in a hierarchical temporal scalability structure and picture type (intra, inter, or bi-predictive picture) for coded video elementary stream.

[0110] If an index is associated with a coded media sample, a sample group may indicate only the first reception hint sample containing data for the media sample or all reception hint samples containing data for the media sample.

[0111] Exemplary sample entry formats may be provided as follows:

```
abstract class IndexSampleEntry() extends MetadataSampleEntry ('ixse') {
        unsigned int(16) entry_count;
        for (i = 1; i <= entry_count; i++)
                unsigned int(32) index_type_4cc;
}

class MPEG2TSIndexSampleEntry() extends IndexSampleEntry ('m2ix') {
        unsigned int(16) program_number;
}

class MPEG2TSMediaIndexSampleEntry() extends IndexSampleEntry ('m2mi') {
        unsigned int(3) reserved;
        unsigned int(13) pid;
}

class RTPIndexSampleEntry() extends IndexSampleEntry ('rtix') {
}

class RTPReceptionGroupIndexSampleEntry() extends IndexSampleEntry ('rgix') {
}
```

[0112] "IndexSampleEntry" indicates the types of indexes that may be present in samples associated with this sample entry. If "entry_count" is equal to 0, any indexes may be included in samples associated with this sample entry. If "entry_count" is greater than 0, a loop of "index_type_4cc" values is given and each value of "index_type_4cc" indicates a four-character code for a box that may be present in samples associated with this sample entry. The sample entries for a timed metadata track therefore indicate the types of metadata that may occur in the timed metadata track. It is noted that other mechanisms to indicate the types of metadata present or allowed in a timed metadata track can be envisioned, e.g., as extensions of track header boxes.

[0113] "MPEG2TSIndexSampleEntry" is used for timed metadata tracks containing program-specific indexes and metadata and associated with an MPEG2-TS reception hint track. "program_number" identifies a program within the MPEG-2 transport stream.

[0114] "MPEG2TSMediaIndexSampleEntry" is used for timed metadata tracks containing media-specific indexes and metadata and associated with an elementary stream contained in an MPEG2-TS reception hint track. pid is an identifier for an elementary stream within an MPEG-2 transport stream.

[0115] "RTPIndexSampleEntry" is used for timed metadata tracks containing media-specific indexes and metadata and associated with an RTP reception hint track.

[0116] "RTPReceptionGroupIndexSampleEntry" is used for program-specific indexes and metadata and associated to more than one RTP reception hint track.

[0117] An exemplary sample format for a timed metadata track containing indexes and segmented metadata is provided below:

```
aligned(8) class IndexSample {
        box index_box[];
}
```

[0118] "IndexSample" contains zero or more index boxes. The four-character code for the included index boxes must be among those indicated by the associated sample entry.

[0119] Exemplary index boxes which can be used with the invention and which were originally proposed in DVB document TM-FF0058r0 are provided as follows:

```
abstract aligned(8) class DVBIndexBox (type) extends Box(type) {
        unsigned int(4)             time_accuracy;
        unsigned int(4)             sample_accuracy;
        if(time_accuracy >= 8)
                unsigned int(32)    max_timing_inaccuracy;
        if(sample_accuracy >= 8)
                unsigned int(32)    max_sample_accuracy;
}
```

[0120] The following values are specified for "time_accuracy" and "sample_accuracy": 0x0: accurate, 0x1: unspecified, 0x2: heuristic, 0x3: reserved (no maximum provided), 0x4 – 0x7: application-specific (no maximum provided), 0x8: maximum inaccuracy specified, 0x9: reserved (maximum inaccuracy provided), 0xA – 0xF: application-specific (maximum inaccuracy provided).

```
aligned(8) class DVBVideoIndex extends DVBIndexBox('idvi') {
        unsigned int(8)             video_event_mask;
        unsigned int(24)        video_event_length;
};
```

[0121] "video_event_mask" is a bit mask indicating the video event(s) that start in the indicated sample, as per Table 1 below.

Table 1: Mask values used for "video_event_mask"

Mask	Meaning

0x01	video decode start point (e.g. a Random Access Point)
0x02	Self decodable picture (e.g. I frame)
0x04	Reference Picture
0x08	P Picture
0x10	B Picture

[0122] "video_event_length" is the number of samples (transport packets) that make up this video picture, including the current packet. The value '0' shall be used to mean "unknown". The Sync Sample Box can also carry the indexes to the events of type 0x01.

```
aligned(8) class DVBPCRIndex extends DVBIndexBox('idpi') {
        unsigned int(1)          PCR_discontinuity_flag;
        unsigned int(5)          reserved_0;
        unsigned int(42)     PCR_Value;
}
```

[0123] "PCR_discontinuity_flag" is a field that is set to '1' if there is a program clock reference (PCR) discontinuity in the associated PCR event; otherwise it is set to '0'.

[0124] "PCR_value" is the 27-MHz value extracted from the PCR that is indexed, for example, as per equation (2-1) in ISO/IEC International Standard 13818-1.

```
aligned(8) class DVBPolarityChange extends DVBIndexBox('idpc') {
        unsigned int(8)          polarity;
}
```

[0125] "polarity" is the polarity of the associated event, as per Table 2 below:

Table 2: Interpretation of Polarity values

Value	Meaning

0	Clear
1	Odd polarity
2	Even polarity

[0126] This value indicates the new polarity value which applies, and the timed metadata sample corresponds to the first reception hint sample with this new polarity. A polarity change index shall only be deemed to occur when the polarity of a stream of packets on a given PID changes, and not when it changes between packets of different PIDs.

```
aligned(8) class DVBCAIndex extends DVBIndexBox('idci') {
        unsigned int(8)         polarity;
        unsigned int(8)         ca_event_data[];
}
```

[0127] "ca_event_data" is the bytes that comprise the packet carrying the conditional access (CA) event; typically, this will be an ECM. The "ca_event_data" continues until the end of the box. The length of the "ca_event_data" can be determined from the length of the box.

```
aligned(8) class DVBTimecodeIndex extends DVBIndexBox('idtc') {
        unsigned int(8)                 timeline_id;
        unsigned int(2)                 reserved_0;
        unsigned int(6)                 tick_format; // as per table 6 in TR 102 823
        unsigned int(32)        absolute_ticks;
}
```

[0128] "timeline_id" is the identifier of the timeline, as per ETSI TS 102 823, clause 5.2.2.4, and "tick_format" is a field which specifies the format that the "absolute_ticks" field takes, as per the Synchronized Auxiliary Data (SAD)

specification, ETSI TS 102 823. "absolute_ticks" is the timecode, coded as indicated by the field "tick_format".

```
aligned(8) class DVBSectionUpdateIndex extends DVBIndexBox('idsu') {
        unsigned int(8)          table_id;
        unsigned int(16)      table_id_extension;
        unsigned int(8)          section_no;
        unsigned int(n*8)     section_data;              // optional
}
```

[0129] "DVBIDTable" box can be included, for example, in a sample table box for a timed metadata track. "table_id" is the table id of the section version update that is being indexed. "table_id_extension" (or "program_number" for PMT, or "transport_stream_id" for PAT) from the section version update is being indexed. "section_no" is the section number to which this update applies. "section_data" is the field may not be present. If it is present it contains the section data of the new version. The section data shall continue until the end of the box. The length of the "section_data" can be determined from the length of the box.

```
aligned(8) class DVBIDIndex extends DVBIndexBox(`didi`) {
        unsigned int(5)          reserved;
        unsigned int(3)          running_status;          // As per table 105
in 102 323
        unsigned int(24)      ID_Table_index;
}
```

[0130] "running_status" is the field which indicates the status of the ID that is referenced by the "ID_Table_index" field (e.g., if the ID is running or paused). The value of this field is define in table 105 of TS 102 323. "ID_Table_index" is an index into the "DVBIDTableBox", indicating the ID that applies at this location with the indicated running_status.

```
aligned(8) class DVBIDTable extends FullBox(`didt`, version = 0, 0) {
        unsigned int(32)        ID_count;
        for(i=0;i<ID_count;i++) {
                string                  ID;     //in URI Format
        }
}
```

[0131] "ID_count" is the number of IDs that follow in the "DVBIDTable". "ID" is the URI formatted ID.

[0132] Further examples of index boxes are provided below:

```
aligned(8) class SDPUpdate extends DVBIndexBox('idsd') {
        string  sdp_text;
}
```

[0133] "sdp_text" is a null-terminated string containing an SDP description that is valid starting from the indicated sample.

```
aligned(8) class KeyUpdate extends DVBIndexBox('idkm') {
        string  key_message;
}
```

[0134] "key_message" contains the cryptographic key to be used for deciphering the packet payloads starting from the related reception hint sample.

```
aligned(8) class ErrorIndex extends DVBIndexBox('idei') {
        unsigned int(2) packet_header_error;
        unsigned int(2) packet_payload_error;
        unsigned int(2) packet_sequence_gap;
        unsigned int(2) reserved;
```

```
}
```

[0135] For "packet_header_error", a value 0x0 indicates that the packet header contains no errors. Value 0x1 indicates that the packet header may or may not contain errors. Value 0x2 indicates that the packet header contains errors. Value 0x3 is reserved.

[0136] For "packet_payload_error" a value 0x0 indicates that the packet payload contains no errors. Value 0x1 indicates that the packet payload may or may not contain errors. Value0x2 indicates that the packet payload contains errors. Value 0x3 is reserved.

[0137] For "packet_sequence_gap", a value 0x0 indicates that the packet immediately follows the previous packet in the reception hint track in transmission order. Value 0x1 indicates that the packet may or may not immediately follow the previous packet in the reception hint track in transmission order. Value 0x2 indicates that the packet does not immediately follow the previous packet in the reception hint track in transmission order, for example, that a there is at least one missing packet preceding this packet. Value 0x3 is reserved.

[0138] In another embodiment, an Index Sample Entry format for a timed metadata track containing indexes is as follows:

```
class IndexSampleEntry() extends MetadataSampleEntry ('ixse') {
        unsigned int(16) program_number;
        unsigned int(16) entry_count;
        int(32) sample_number_offset;
        for (i = 1; i <= entry_count; i++) {
                unsigned int(32) index_type_4cc;
                unsigned int(8) instance_info_length;
                unsigned int(8) payload_length;
                unsigned int(8) extension_count;
        }
}
```

[0139] An IndexSampleEntry indicates the types of indexes that may be present in samples associated with this sample entry. program_number identifies a program within the MPEG-2 transport stream. If entry_count is equal to 0, then any indexes may be included in samples associated with this sample entry, and default values (according to the first release of the DVB File Format specification) of instance_info_length, payload_length, and extension_count are used.
sample_number_offset specifies an offset to be added to the sample_number in the associated timed metadata samples to obtain the sample number in the referred track.
index_type_4cc indicates an index type that may be present in samples associated with this sample entry. instance_info_length, payload_length, and extension_count indicate the values of these fields used in index samples. A DVB File Format parser ignores those bytes index_instance_info, index_payload and index_extension that are beyond the byte range specified in this document for the respective index_type_4cc. This mechanism allows appending fields in a backward-compatible manner in later index format versions. If there are many timed metadata tracks for a reception hint track, then index_type_4cc values can be used to locate the track containing the desired indexes.

[0140] An index sample contains one or more index events, each having the following abstract syntax. The semantics of index_instance_info and index_payload are determined by the 32-bit index_type and the semantics of index_extension is determined by extension_type as specified subsequently. index_type corresponds to grouping_type in sample grouping.

```
abstract class IndexEvent(index_type, info_length, payload_length, extension_count)
{
        unsigned int(8) index_instance_info[info_length];
        unsigned int(8) index_payload[payload_length];
    for (i=1; i<=extension_count; i++) {
            unsigned int(32) extension_type;
            unsigned int(16) extension_length;
            unsigned int(8) index_extension[extension_length];
}
```

```
        unsigned int(8) pad[];
}
abstract class IndexInstanceInfo(index_type, info_length)
{
}
abstract class IndexPayload(index_type, payload_length)
{
}
abstract class IndexExtension(extension_type, extension_length)
{
}
```

[0141] The semantics of index_instance_info, index_payload, extension_type, and index_extension in the IndexEvent structure are identical to the semantics of index_instance_info, index_payload, extension_type, and index_extension, respectively, used in DVBSampleToGroup, DVBSampleGroupDescription, and DVBSampleToGroupExtension boxes. This match enables the straightforward conversion between indexes included in timed metadata tracks and indexes indicated by DVBSampleToGroup boxes.

[0142] Syntax of some index events are provided below, where the semantics of the syntax elements is the same as described in the embodiment presented previously.

```
class VideoIndex extends IndexEvent("idvi", 0, 2, 0);
class VideoIndexPayload extends IndexPayload("idvi", 2) {
        unsigned int(2)         reserved = 0;
        unsigned int(2)         sample_depends_on;
        unsigned int(2)         sample_is_depended_on;
        unsigned int(2)         sequence_depends_on;
        unsigned int(8)         dependency_level;
}

class PCRIndex extends IndexEvent("idpc", 0, 6, 0);
class PCRIndexPayload extends IndexPayload("idpc", 6) {
```

```
        unsigned int(1)  PCR_discontinuity_flag;
        unsigned int(5)  reserved_0;
        unsigned int(42) PCR_Value;
}

class PolarityIndex extends IndexEvent("idca", 2, 1, 1);

class PolarityInstanceInfo extends IndexInstanceInfo("idca", 2) {
        unsigned int(13) pid;
        unsigned int(3) reserved;
}

class PolarityIndex extends IndexEvent("idca", 0, 1, 1);

class PolarityIndexPayload extends IndexPayload("idca", 1) {
        unsigned int(6) reserved_0;
        unsigned int(2) polarity;
}

class PolarityIndexExtension extends IndexExtension("idca", extension_length) {
        unsigned int(8) ca_event_data[extension_length];
}

class TimecodeIndex extends IndexEvent("idtc", 1, 5, 0);

class TimecodeIndexInstanceInfo extends IndexInstanceInfo("idtc", 1) {
        unsigned int(8) timeline_id;
}

class TimecodeIndexPayload extends IndexPayload("idtc", 5) {
        unsigned int(2) reserved_0;
```

```
        unsigned int(6) tick_format; // as per table 6 in TR 102 823
        unsigned int(32) absolute_ticks;
}

class TimeIndex extends IndexEvent("idti", 0, 0, 0);

class SectionUpdateIndex extends IndexEvent("idsu", 0, 4, 1);

class SectionUpdateIndexPayload extends IndexPayload("idsu", 4) {
        unsigned int(8)  table_id;
        unsigned int(16) table_id_extension;
        unsigned int(8)  section_no;
}

class SectionUpdateIndexExtension extends IndexExtension("idsu",
extension_length) {
        unsigned int(8) section_data[extension_length];
}
```

[0143] There are at least two types of usage for descriptive metadata. First, when a broadcast is recorded continuously (e.g., as a response of user initiating an instantaneous record operation rather than a scheduled recording of a particular program), content from more than one program may be recorded to a single file. Thus, Electronic Service Guide (ESG) information is also stored for more than one program within the file. An indexing mechanism can be used to map a group of samples comprising a program with the respective fragments of ESG information. Second, different "storylines", such as summaries or special cuts, can be provided by broadcasters. For example, a broadcaster can provide information, in particular the time periods and descriptions, about the goals of a soccer game, general highlights of the game, or highlights of a particular player during the game. These "storylines" can be provided after the recording and may use a specific timeline (as discussed in the

timecode index presented earlier). An indexing mechanism can be used for indicating "storylines" in a recorded file.

[0144] The DVBSampleToGroup box can be used for descriptive or segmented metadata as follows. The value of grouping_type can be differentiate ESG and "storyline" metadata, even though they may be structurally identical and the same grouping_type could also be used. The global parameter, instance_info, is used to differentiate between storylines. For ESG metadata, instance_info can contain the item_id of the metadata item for initiating the ESG metadata (such as giving information about the ESG provider) or it may be used to differentiate between different programs of ESG metadata. In practice, instance_info can contain the item_id of a metadata item containing the information about storylines or ESG. Alternatively, instance_info may point to the DVBGroupingInstance box, which can e.g. contain a textual name of each storyline or URIs pointing to descriptions. The entries in the loop included in the DVBSampleToGroup box can be used as follows. If the constant-size payload field is used, it may contain the item_id of the metadata item containing a description of the grouped samples, such as respective ESG fragment or descriptive data about an event in the storyline. If group_description_index is used within the loop, the DVBSampleGroupDescription box may contain the URIs of the respective ESG fragments or descriptive XML fragments about an event in the storyline. Alternatively, the DVBSampleGroupDescription box may contain textual description of the storyline events. When the syntax for DVBSampleToGroup box that allows varying-length payloads is used, the payload can be an URI used for purposes mentioned above or a textual description of an event.

[0145] Figure 10 illustrates a file including two sample groupings according to the DVBSampleToGroup box for the timecode index. As the syntax above specifies, the value of timeline_id serves as the global parameter for a timecode index sample grouping. In 1010, the timeline_id is equal to 100, whereas in 1020, the timeline_id is equal to 200.

[0146] When sample groups are included in a reception hint track or a media track, seeking to a position in a file based on a desired segmented metadata or index property can be done by a file parser with the following steps:

1. Select an appropriate sample group description box (based on its grouping type).
2. Select a desired group description index within the sample group description box.
3. Seek for the desired group description index from the SampleToGroup box(es) (of the selected grouping type). This reveals the sample number of the desired metadata/index property.
4. Use the sample table box to locate the sample in a file based on its sample number.

[0147] When sample groups according to the DVBSampleToGroup box are included in a reception hint track or a media track and payload rather than group_description_index is used, seeking a position in a file based on a desired segmented metadata or index property can be done by a file parser with the following processes:

1. Select an appropriate DVBSampleToGroup box (based on its grouping type and instance_info).
2. Seek for the desired value of payload in the DVBSampleToGroup box(es). This reveals the sample number having the desired metadata/index property.
3. Use the sample table box to locate the sample in a file based on its sample number.

[0148] When sample groups are included in a timed metadata track, seeking to a position in a file based on a desired segmented metadata or index property can be done by a file parser with the following processes:

1. Select an appropriate sample group description box (based on its grouping type).
2. Select a desired group description index within the sample group description box.
3. Seek for the desired group description index from the SampleToGroup

box(es) (of the selected grouping type). This reveals the sample number of the desired metadata/index property within the timed metadata track.

4. Map the sample of the timed metadata track to a sample of the associated reception hint track. If one metadata sample is created for each reception hint sample, the desired sample number in the reception hint track is identical to that of the timed metadata track. Otherwise, derive the decoding timestamp of the timed metadata sample (using the decoding time to sample box) and seek for a reception hint sample having an identical decoding timestamp (or a reception hint sample that pertains at the time of the decoding timestamp)

5. Use the sample table box of the reception hint track to locate the sample in a file based on the sample number of the reception hint sample identified in the previous step.

[0149] When sample groups are included in a reception hint track or a media track, obtaining the indexes and segmented metadata valid for a particular reception hint sample or a media sample, respectively, can be done by a file parser with the following processes:

1. Select each sample to group box (one at a time) associated with the reception hint track or the media track and find the sample group description index of the of the particular reception hint sample or media sample, respectively.

2. Obtain the index or segmented metadata corresponding to the sample group description index from the respective sample group description box.

[0150] When sample groups are included in a timed metadata track, obtaining the indexes and segmented metadata valid for a particular reception hint sample can be done by a file parser with the following processes:

1. Map the reception hint sample to a metadata sample as follows. If one metadata sample is created for each reception hint sample, the desired sample number in the reception hint track is identical to that of the timed metadata track. Otherwise, derive the decoding timestamp of the reception hint sample (using the decoding time to sample box) and seek for a timed metadata sample having an identical decoding timestamp (or a timed metadata sample that pertains at the time of the decoding timestamp).

2. Select each sample to group box (one at a time) associated with the timed metadata track and find the sample group description index of the of the particular timed metadata sample, respectively.

3. Obtain the index or segmented metadata corresponding to the sample group description index from the respective sample group description box.

[0151] A player, typically comprising a file parser, a decoder 160, and a renderer 170, can provide a list of indices, bookmarks, or other types of metadata to the end-user by analyzing and re-producing some part of the content of the sample group description boxes.

[0152] A player may provide bookmarking operation to end-users. Bookmarks can be stored to a file using sample grouping similarly as described above for the receiver operation

[0153] Thus, in accordance with embodiments of the present invention, sample groups can be used with movie fragments seamlessly, because SampleToGroup box can reside in a track fragment box. Hence, simultaneous recording and playback of a single file is fully supported. Further, the sample group mechanism is extensible, as new grouping types can be specified. Persistence of metadata is clear: contiguous samples with the same value of group_description_index (of the same grouping type). Further, getting a table of contents of the file based on SampleGroupDescription boxes is straightforward. Thus, when storing metadata associated with single samples, e.g. bookmarks, the sample grouping mechanism can be better than timed metadata tracks.

[0154] While embodiments of the invention have mostly concerned a receiver 150, a recording storage 155, a file parser, and a decoder 160, the presented file structures can also be used by an encoder 110 when creating a file. In particular, when a file is provided for playback without converting its media contents to a packetized format, it is often useful to provide segmented metadata and indices structured as proposed in the invention. In accordance with the ISO base media file format, the encoder 110 derives file metadata that will be later stored within the moov box or the moof box, depending on whether or not movie fragments are in use. For any received piece of metadata, the encoder 110 identifies its type and creates a new sample group

description box, if the type of metadata did not exist before. The encoder 110 also checks whether or not the content of the piece of metadata differs from all metadata stored earlier in the file. If the content is novel, then a new entry is created for the sample group description box and the metadata content is copied into the entry. The SampleToGroup box is maintained for each grouping type and samples of a media track or a hint track are mapped to one of the entries in the respective sample group description box.

[0155] It is noted that the above-described creation of sample grouping may be implemented in a number ways. For example, the creation of sample grouping may not happen simultaneously with the encoding and storage 120 of the samples themselves. Instead, sample grouping may be performed off-line, based on an intermediate format for storing metadata. Referring again to Figure 1, the off-line creation of sample grouping would result into two additional blocks getting their input from the storage 120 and outputting to the server 130. The first block in processing order can be referred to as a server file re-writer, which inputs a file without sample grouping for a particular metadata type and outputs a file with that type of sample grouping. The second block in processing order can be referred to as a second storage, which may have similar properties to the storage 120.

[0156] The server file re-writer and the second storage referred to above may reside in the same device as the encoder 110, the storage 120, the server 130, or a different device. Moreover, the server file re-writer and the second storage may reside in the same device or different devices from each other.

[0157] In one embodiment, the server 130 typically includes a file parser (not shown in Figure 1). The file parser follows the instructions provided in one or more hint tracks to generate a packet stream from coded media bitstreams included in a container file. The file parser further processes the information given in the sample group description boxes associated with the hint tracks. The processing may include, for example, creation of SDP information or parts of an electronic service guide based on the information given in the sample group description boxes. The processing may also include processing of the SampleToGroup boxes to resolve the relation of pieces of metadata (associated with a certain group description index) given in the sample

group description boxes with media or hint samples, and the persistence of metadata provided in the sample group description boxes. For example, the server may check when new SDP information is required for decoding of the transmitted packet stream and send the new SDP information such a way that it is available for a receiver when it is needed for decoding packets. In one embodiment, the file provided to the server is created by a receiver 150 and stored in a recording storage 155, which is operationally connected to the server 130.

[0158] While particular embodiments of the present invention have been disclosed, it is to be understood that various different modifications and combinations are possible and contemplated within the scope of the appended claims. There is no intention, therefore, of limitations to the exact abstract and disclosure herein presented.

[0159] Various embodiments of present invention described herein are described in the general context of method steps, which may be implemented in one embodiment by a program product including computer-executable instructions, such as program code, executed by computers in networked environments. A computer-readable medium may include removable and non-removable storage devices including, but not limited to, Read Only Memory (ROM), Random Access Memory (RAM), compact discs (CDs), digital versatile discs (DVD), etc. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Computer-executable instructions, associated data structures, and program modules represent examples of program code for executing steps of the methods disclosed herein. The particular sequence of such executable instructions or associated data structures represents examples of corresponding acts for implementing the functions described in such steps. Various embodiments of the present invention can be implemented directly in software using any common programming language, e.g. C/C++ or assembly language.

[0160] Software and web implementations of the present invention could be accomplished with standard programming techniques with rule based logic and other logic to accomplish the various database searching steps, correlation steps,

comparison steps and decision steps. It should also be noted that the words "component" and "module," as used herein and in the claims, is intended to encompass implementations using one or more lines of software code, and/or hardware implementations, and/or equipment for receiving manual inputs.

[0161] Individual and specific structures described in the foregoing examples should be understood as constituting representative structure of means for performing specific functions described in the following the claims, although limitations in the claims should not be interpreted as constituting "means plus function" limitations in the event that the term "means" is not used therein. Additionally, the use of the term "step" in the foregoing description should not be used to construe any specific limitation in the claims as constituting a "step plus function" limitation. To the extent that individual references, including issued patents, patent applications, and non-patent publications, are described or otherwise mentioned herein, such references are not intended and should not be interpreted as limiting the scope of the following claims.

[0162] The foregoing description of embodiments of the present invention have been presented for purposes of illustration and description. It is not intended to be exhaustive or to limit the present invention to the precise form disclosed, and modifications and variations are possible in light of the above teachings or may be acquired from practice of the present invention. The embodiments were chosen and described in order to explain the principles of the present invention and its practical application to enable one skilled in the art to utilize the present invention in various embodiments and with various modifications as are suited to the particular use contemplated. The features of the embodiments described herein may be combined in all possible combinations of methods, apparatus, modules, systems, and computer program products.

What is claimed is:

1. A method comprising:

causing streamed data to be stored in a file;

identifying metadata applicable to one or more samples of the streamed data;

causing at least one timed metadata track to be created based on the identified metadata; and

forming at least one group from the one or more samples of the streamed data, each sample in a group having identical metadata content for a metadata type.

2. The method of claim 1, further comprising causing the at least one group to be identified in a file.

3. The method of claim 1, wherein the file is in accordance with International Organization for Standardization (ISO) base media file format.

4. The method of claim 3, wherein causing the streamed data to be stored in a file includes causing the streamed data to be stored in a reception hint track.

5. The method of claim 3, wherein the metadata type is indicated by a grouping type and grouping instance data, the grouping type specifying semantics of the grouping instance data and the metadata content.

6. The method of claim 5, wherein the metadata content comprises a metadata payload and zero or more metadata payload extensions, wherein the metadata payload is included in a first structure, and wherein the zero or more metadata payload extensions are included in a second structure.

7. The method of claim 5, wherein the at least one group is indicated in a sample group description box.

8. The method of claim 3, wherein the at least one group is indicated in a sample group description box.

9. The method of claim 1, wherein the metadata type includes titles of segments of the streamed data.

10. The method of claim 1, wherein a group includes samples from two or more tracks.

11. The method of claim 10, further comprising:

identifying the group including samples from two or more tracks with metadata associated with all related tracks.

12. The method of claim 10, further comprising:

identifying the group including samples from two or more tracks with metadata associated with a master track.

13. The method of claim 8, wherein the at least one group of one or more samples of the streamed data is indicated in a sample group description box for the timed metadata track through associating the at least one group of one or more samples of the streamed data with respective timed metadata samples in the timed metadata track.

14. The method of claim 1, wherein timed metadata track associates timing metadata with one or more timelines.

15. The method of claim 14, wherein timed metadata track associates timing metadata with two or more timelines.

16. The method of claim 1, wherein causing the at least one timed metadata track to be created comprises causing at least two timed metadata tracks to be created.

17. The method of claim 16, wherein causing at least two timed metadata tracks to be created comprises causing a media-specific timed metadata track to be created and causing a program-specific metadata track to be created.

18. A non-transitory computer-readable medium storing program code portions therein, the program code portions being configured to, upon execution, cause an apparatus to at least:

cause streamed data to be stored in a file;

identify metadata applicable to one or more samples of the streamed data;

cause at least one timed metadata track to be created based on the identified metadata; and

form at least one group from the one or more samples of the streamed data, each sample in a group having identical metadata content for a metadata type.

19. The computer-readable medium of claim 18, wherein the program code portions are further configured to, upon execution, cause the apparatus to cause the at least one group to be identified in a file.

20. The computer-readable medium of claim 18, wherein the file is in accordance with International Organization for Standardization (ISO) base media file format.

21. The computer-readable medium of claim 20, wherein the program code portions are further configured to, upon execution, cause the apparatus to cause the streamed data to be stored in a file by causing the streamed data to be stored in a file in a reception hint track.

22. The computer-readable medium of claim 20, wherein the metadata type is indicated by a grouping type and grouping instance data, the grouping type specifying semantics of the grouping instance data and the metadata content.

23. The computer-readable medium of claim 21, wherein the metadata content comprises a metadata payload and zero or more metadata payload extensions, wherein the metadata payload is included in a first structure, and wherein the zero or more metadata payload extensions are included in a second structure.

24. The computer-readable medium of claim 20, wherein the at least one group is indicated in a sample group description box.

25. The computer-readable medium of claim 18, wherein the metadata type includes titles of segments of the streamed data.

26. The computer-readable medium of claim 18, wherein a group includes samples from two or more tracks.

27. The computer-readable medium of claim 26, wherein the program code portions are further configured to, upon execution, cause the apparatus to identify the group including samples from two or more tracks with metadata associated with all related tracks.

28. The computer-readable medium of claim 26, wherein the program code portions are further configured to, upon execution, cause the apparatus to identify the group including samples from two or more tracks with metadata associated with a master track.

29. The computer-readable medium of claim 24, wherein the at least one group of one or more samples of the streamed data is indicated in a sample group description box for the timed metadata track through associating the at least one group of one or more samples of the streamed data with respective timed metadata samples in the timed metadata track.

30. The computer-readable medium of claim 18, wherein timed metadata track associates timing metadata with one or more timelines.

31. The computer-readable medium of claim 30, wherein timed metadata track associates timing metadata with two or more timelines.

32. The computer-readable medium of claim 18, wherein the program code portions are further configured to, upon execution, cause the apparatus to cause the at least one timed metadata track to be created by causing at least two timed metadata tracks to be created.

33. The computer-readable medium of claim 32, wherein the program code portions are further configured to, upon execution, cause the apparatus to cause the at least two

timed metadata tracks to be created by causing a media-specific timed metadata track to be created and causing a program-specific metadata track to be created.

34. An apparatus comprising a processor and a memory storing program code, the memory and program code being configured to, with the processor, cause the apparatus to at least:

cause streamed data to be stored in a file;

identify metadata applicable to one or more samples of the streamed data;

cause at least one timed metadata track to be created based on the identified metadata; and

form at least one group from the one or more samples of the streamed data, each sample in a group having identical metadata content for a metadata type.

35. The apparatus of claim 34, wherein the apparatus is further caused to identify the at least one group in a file stored in the storage unit.

36. The apparatus of claim 34, wherein the file is in accordance with International Organization for Standardization(ISO) base media file format.

37. The apparatus of claim 36, wherein the apparatus is caused to cause the streamed data to be stored in a reception hint track.

38. The apparatus of claim 36, wherein the metadata type is indicated by a grouping type and grouping instance data, the grouping type specifying semantics of the grouping instance data and the metadata content.

39. The apparatus of claim 38, wherein the metadata content comprises a metadata payload and zero or more metadata payload extensions, wherein the metadata payload is included in a first structure, and wherein the zero or more metadata payload extensions are included in a second structure.

40. The apparatus of claim 36, wherein the at least one group is indicated in a sample group description box.

41. The apparatus of claim 34, wherein the metadata type includes titles of segments of the streamed data.

42. The apparatus of claim 34, wherein a group includes samples from two or more tracks.

43. The apparatus of claim 34, wherein the apparatus is further caused to identify the group including samples from two or more tracks with metadata associated with all related tracks.

44. The apparatus of claim 42, wherein the apparatus is further caused to identify the group including samples from two or more tracks with metadata associated with a master track.

45. The apparatus of claim 40, wherein the at least one group of one or more samples of the streamed data is indicated in a sample group description box for the timed metadata track through associating the at least one group of one or more samples of the streamed data with respective timed metadata samples in the timed metadata track.

46. The apparatus of claim 34, wherein timed metadata track associates timing metadata with one or more timelines.

47. The apparatus of claim 46, wherein timed metadata track associates timing metadata with two or more timelines.

48. The apparatus of claim 34, wherein the apparatus is caused to cause the at least one timed metadata track to be created by causing at least two timed metadata tracks to be created.

49. The apparatus of claim 48, wherein the apparatus is caused to cause at least two timed metadata tracks to be created by causing a media-specific timed metadata track to be created and causing a program-specific metadata track to be created.

50. An apparatus, comprising:

means for causing streamed data to be stored in a file;

means for identifying metadata applicable to one or more samples of the streamed data;

means for causing at least one timed metadata track to be created based on the identified metadata; and

means for forming at least one group from the one or more samples of the streamed data, each sample in a group having identical metadata content for a metadata type.

51. The apparatus of claim 50, wherein the file is in accordance with International Organization for Standardization (ISO) base media file format.

52. The apparatus of claim 51, wherein the storing streamed data in a file includes storing in a reception hint track.

53. The apparatus of claim 51, wherein the metadata type is indicated by a grouping type and grouping instance data, the grouping type specifying semantics of the grouping instance data and the metadata content.

54. The apparatus of claim 53, wherein the metadata content comprises metadata payload and zero or more metadata payload extensions, wherein the metadata payload is included in a first structure, and wherein the zero or more metadata payload extensions are included in a second structure.

55. The apparatus of claim 51, wherein the at least one group is indicated in a sample group description box.

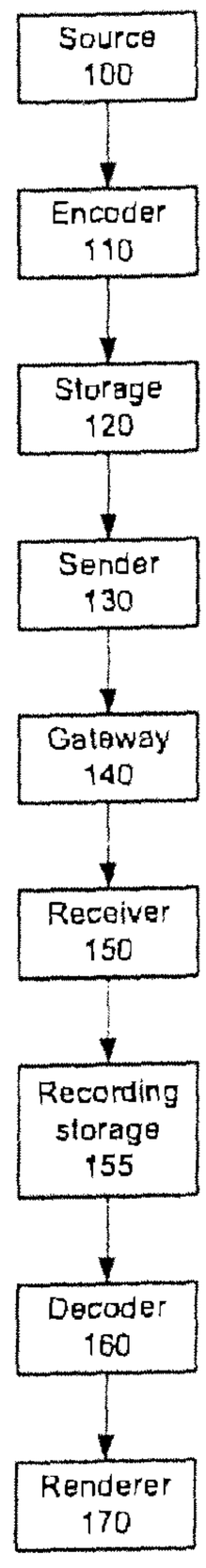

Source
100
Encoder
110
Storage
120
Sender
130
Gateway
140
Receiver
150
Recording
storage
155
Decoder
160
Renderer
170

Figure 1

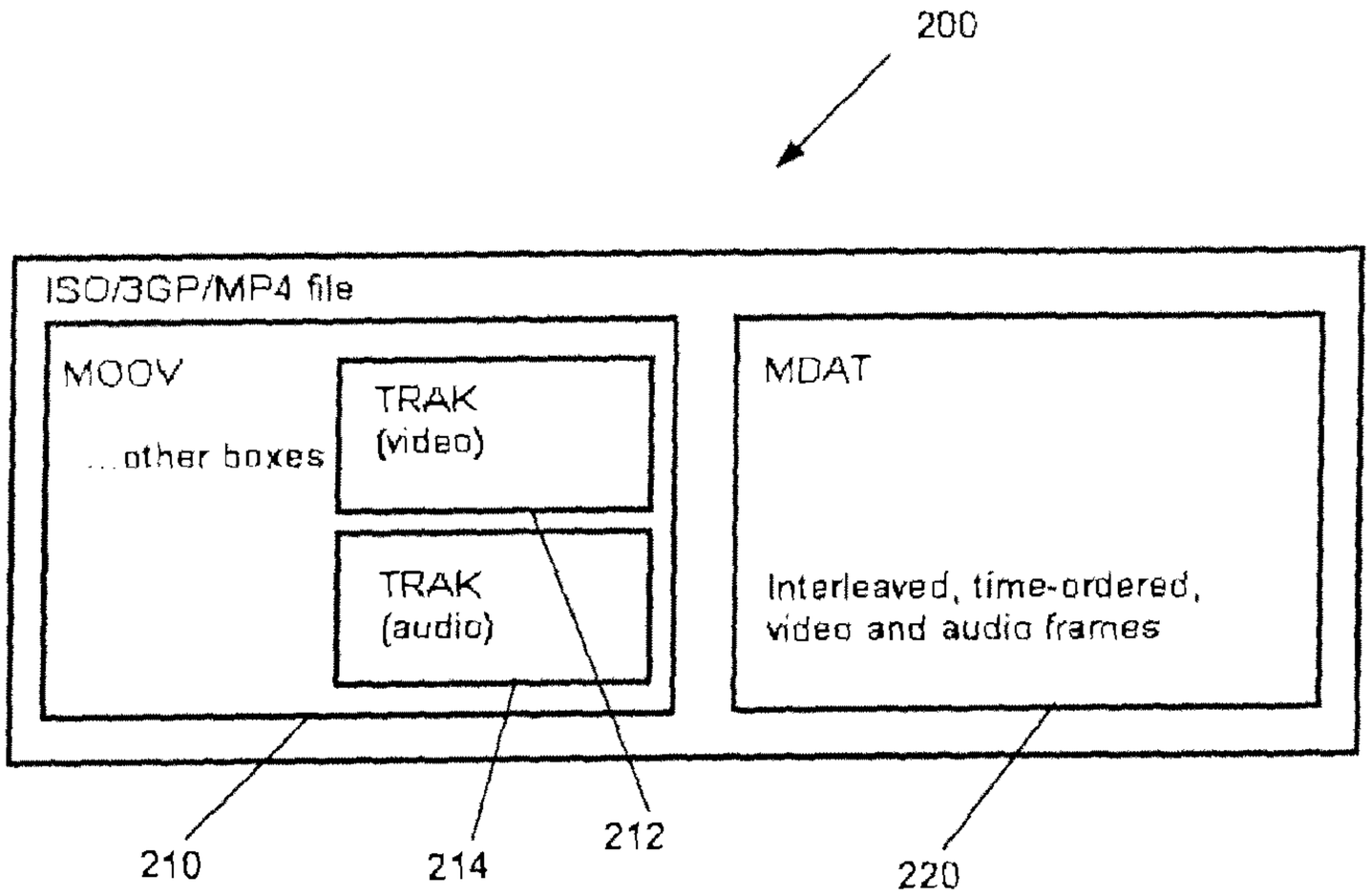

200
ISO/3GP/MP4 file
MOOV
...other boxes
TRAK
(video)
TRAK
(audio)
MDAT
Interleaved, time-ordered,
video and audio frames
210
214
212
220

Figure 2

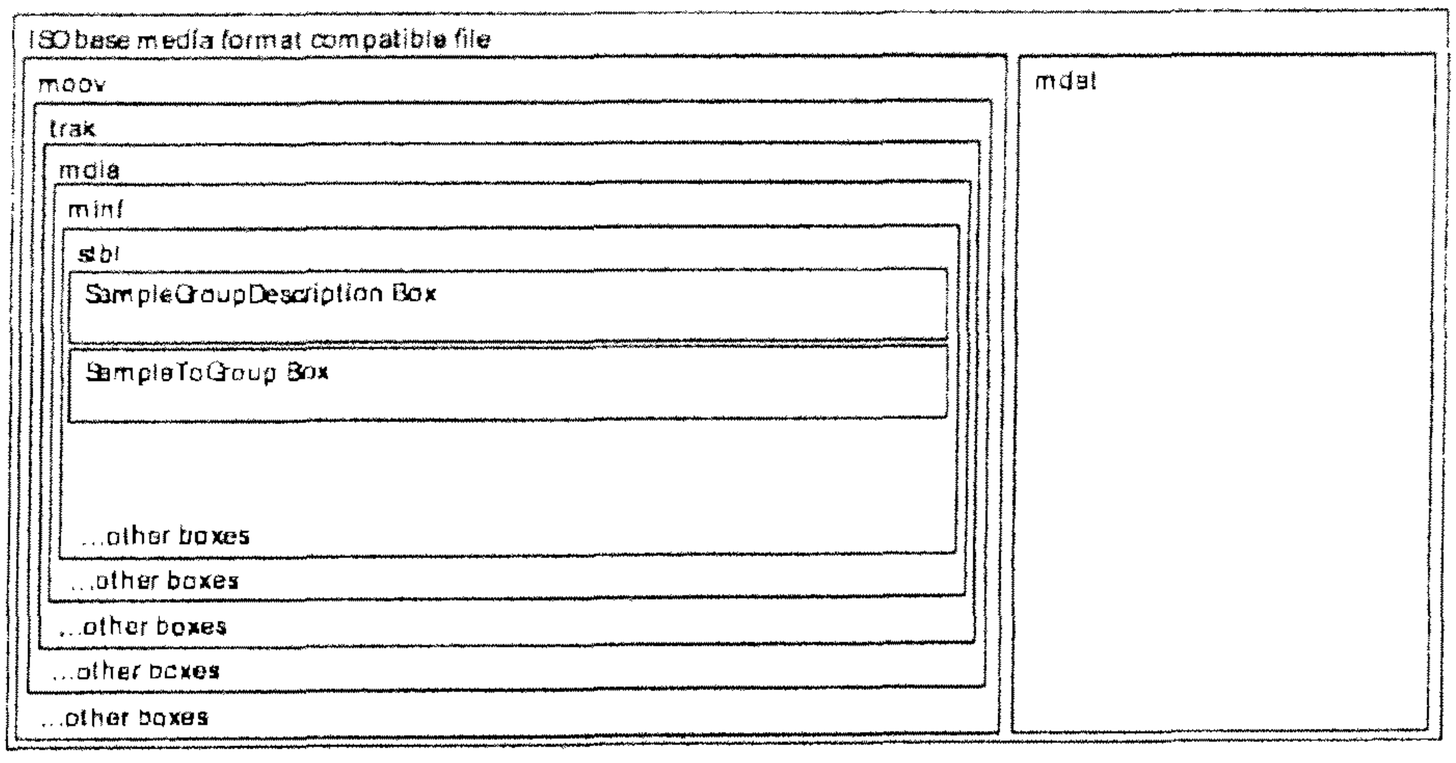


Figure 3A

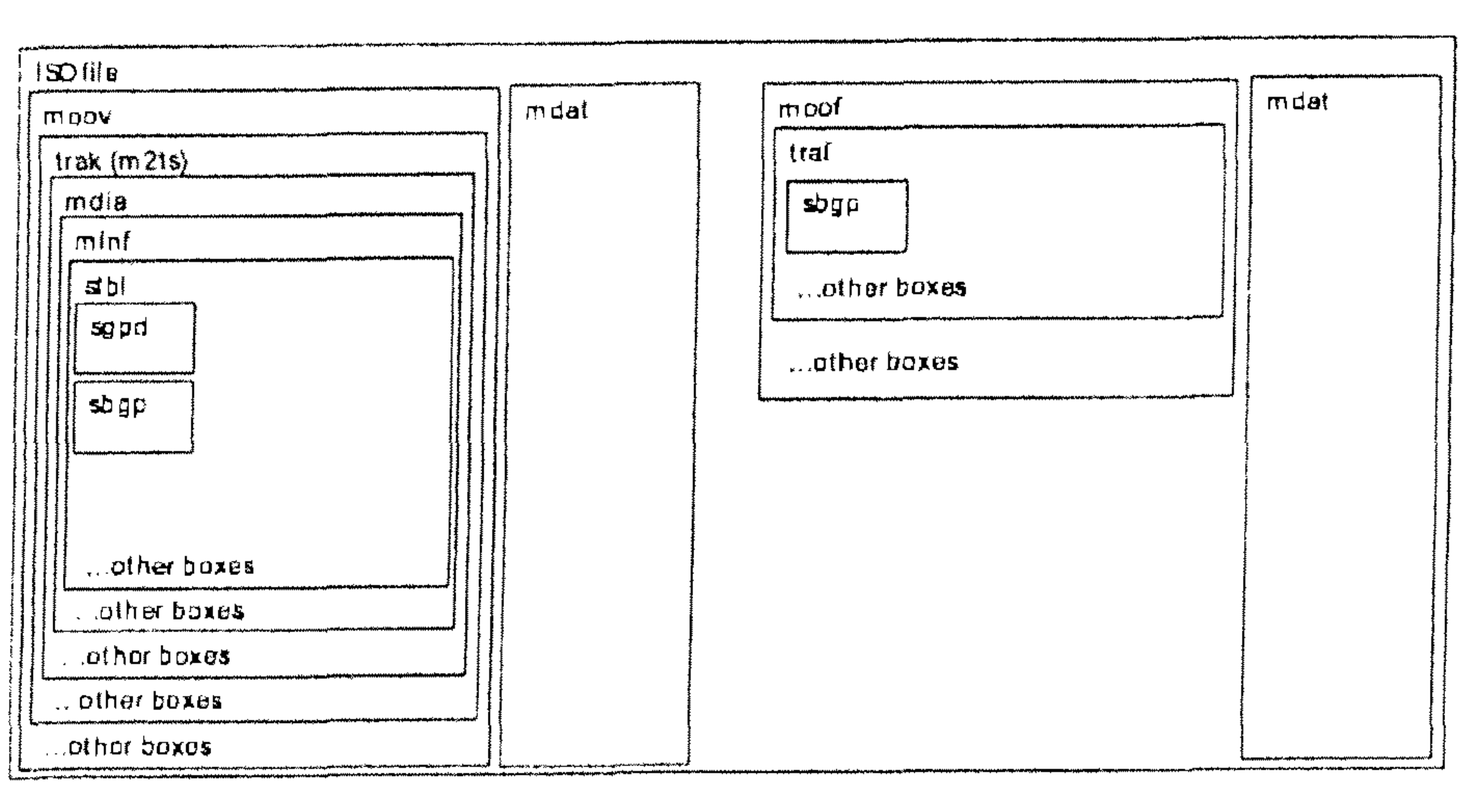


Figure 3B

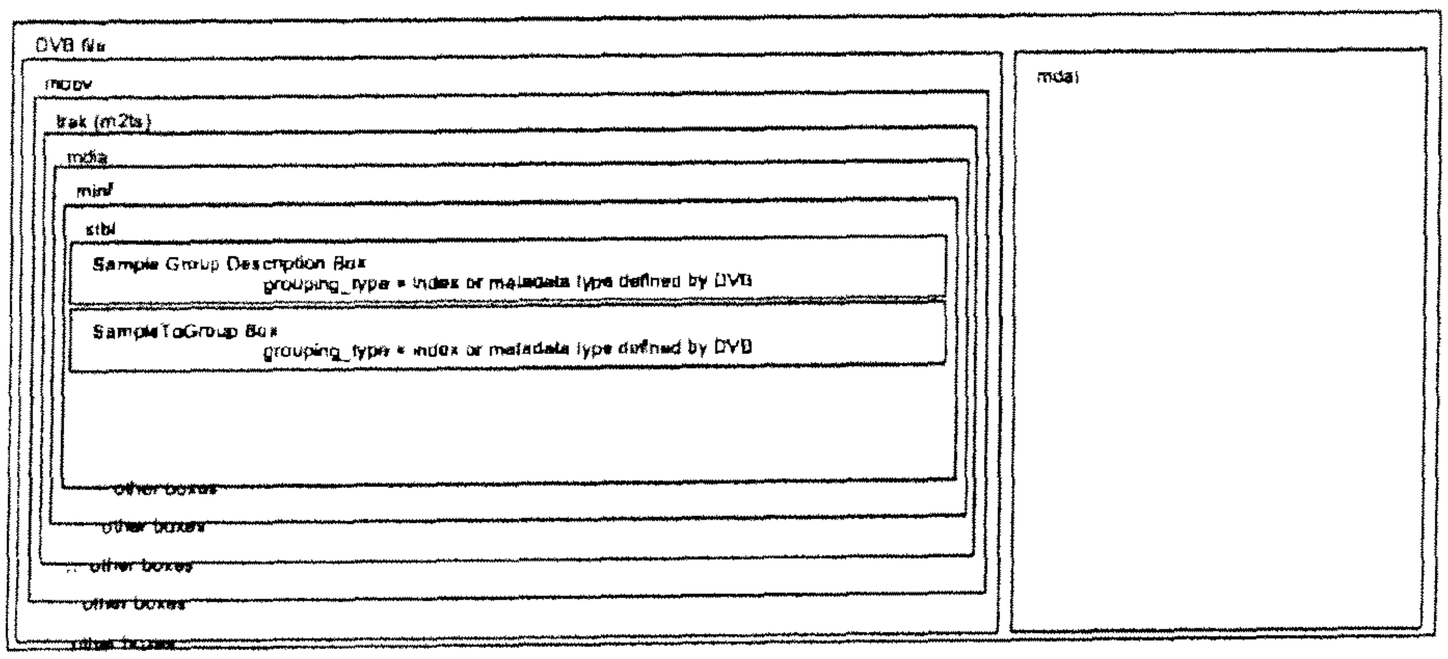

DVB file
moov
trak (m2ts)
mdia
minf
stbl
Sample Group Description Box
grouping_type = index or metadata type defined by DVB
SampleToGroup Box
grouping_type = index or metadata type defined by DVB
other boxes
other boxes
other boxes
other boxes
mdat

Figure 4

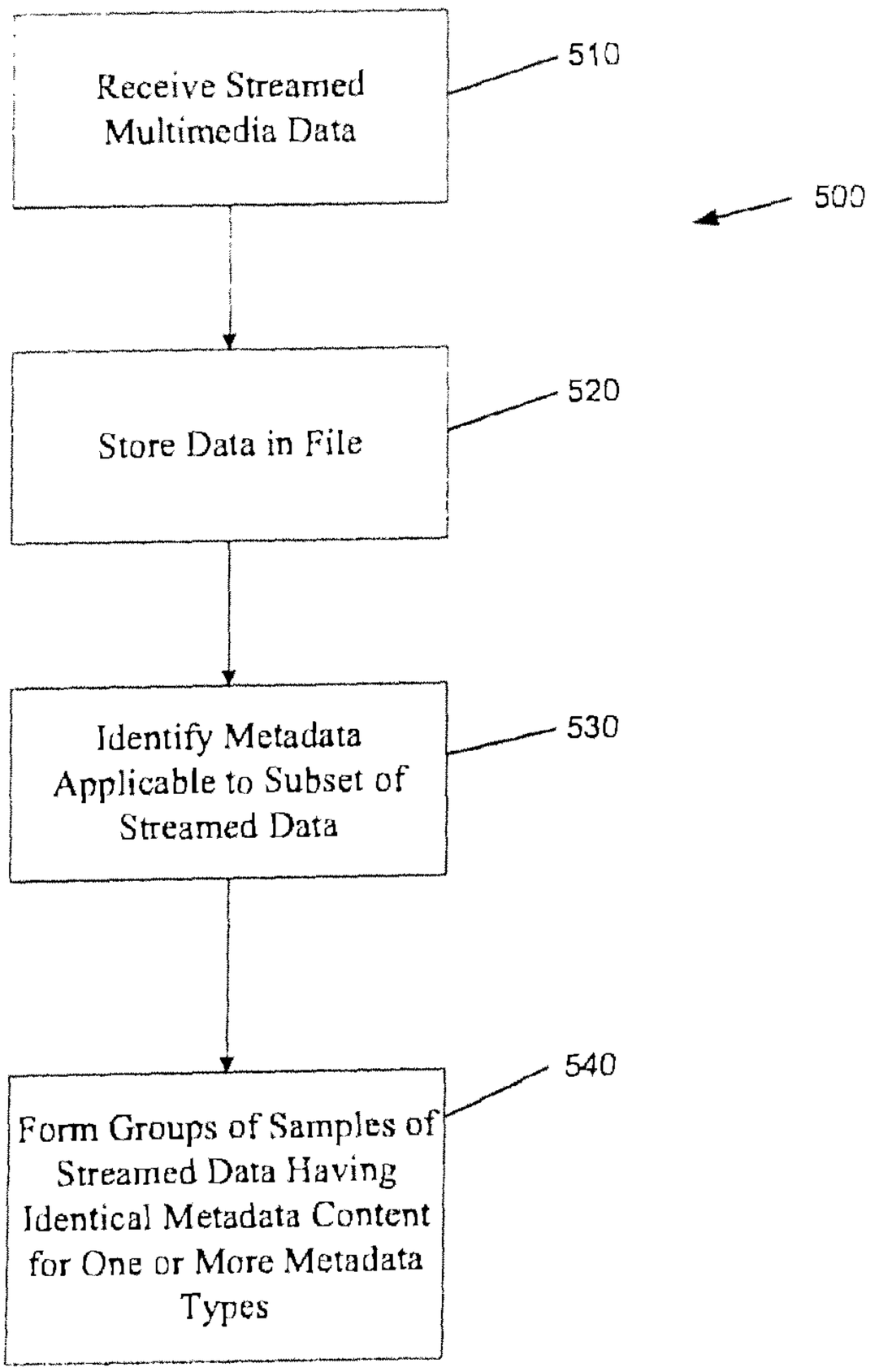

Receive Streamed Multimedia Data
510
500
Store Data in File
520
Identify Metadata Applicable to Subset of Streamed Data
530
Form Groups of Samples of Streamed Data Having Identical Metadata Content for One or More Metadata Types
540

Figure 5A

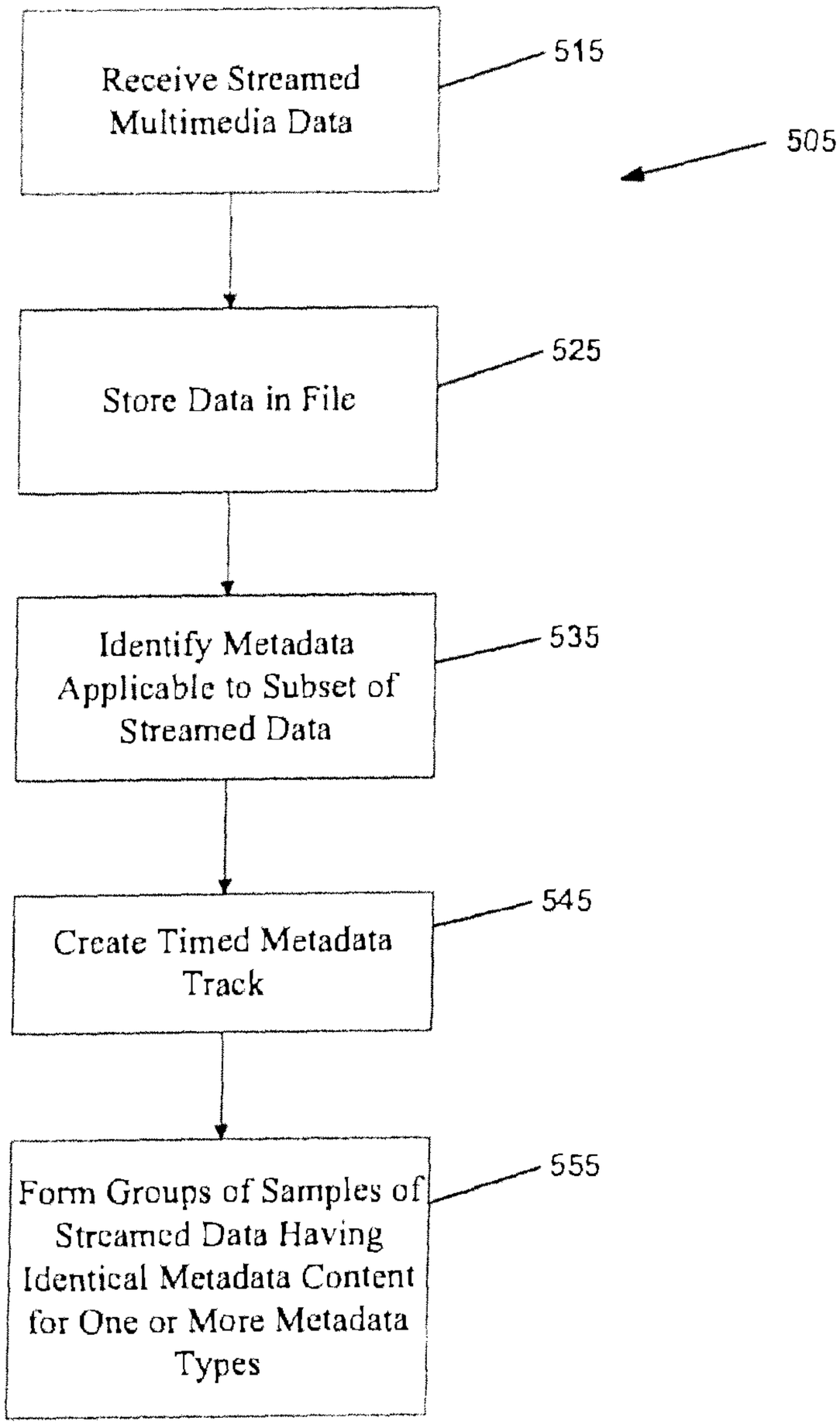

Receive Streamed Multimedia Data
515
505
Store Data in File
525
Identify Metadata Applicable to Subset of Streamed Data
535
Create Timed Metadata Track
545
Form Groups of Samples of Streamed Data Having Identical Metadata Content for One or More Metadata Types
555

Figure 5B

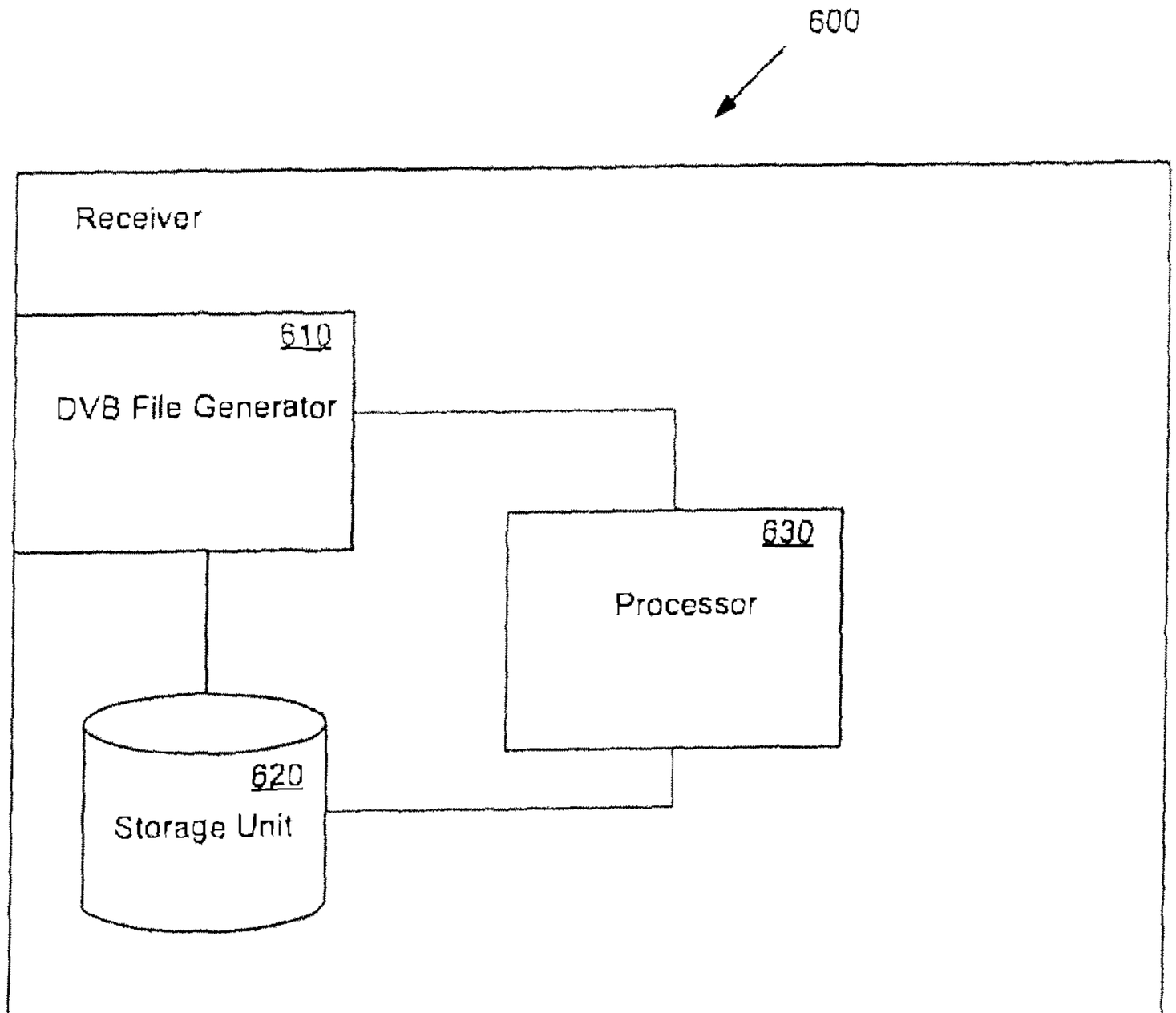

600
Receiver
610
DVB File Generator
630
Processor
620
Storage Unit

Figure 6

FIG. 7

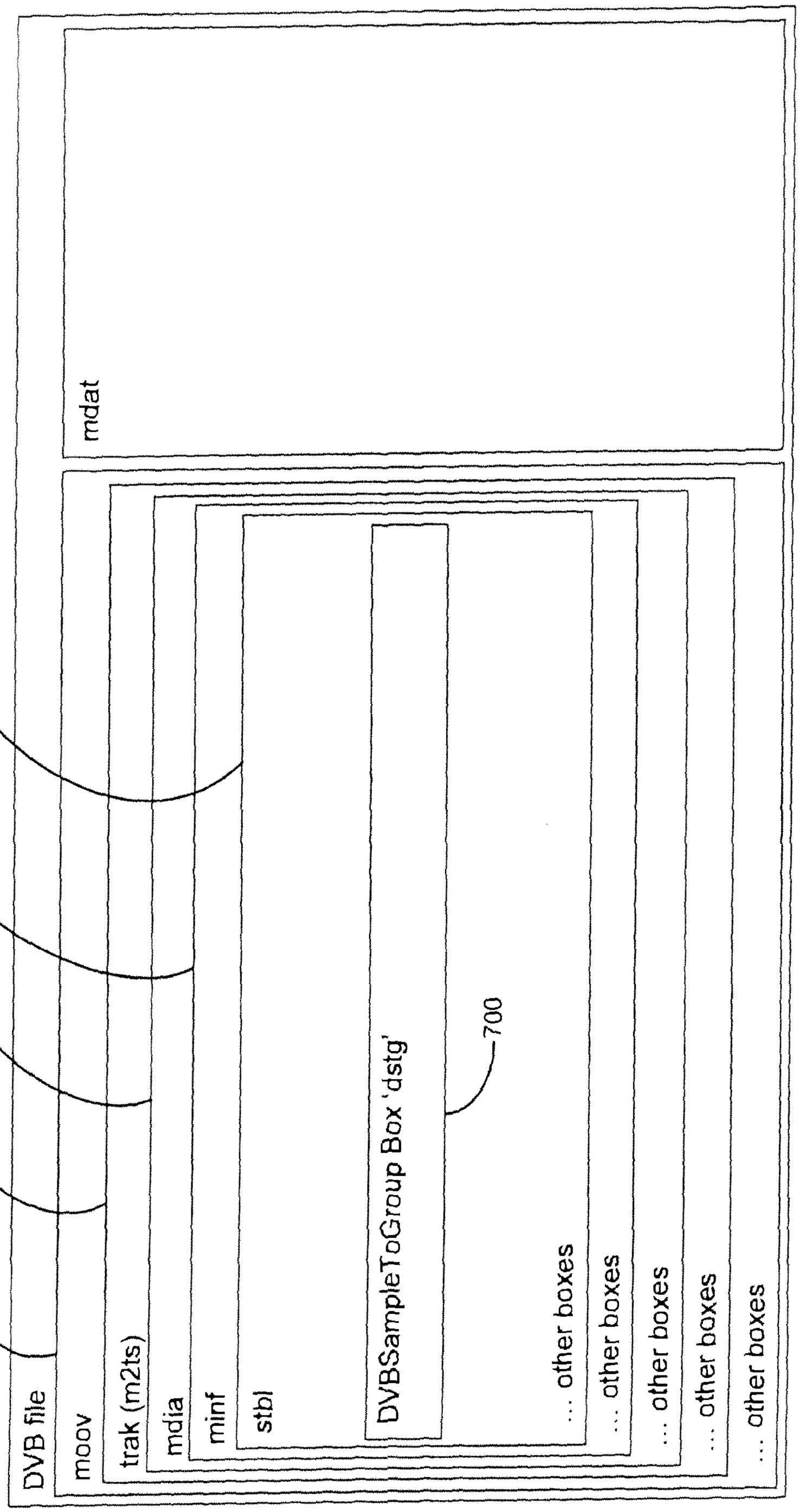

DVB file
moov
trak (m2ts)
mdia
minf
stbl
DVBSampleToGroup Box 'dstg'
... other boxes
... other boxes
... other boxes
... other boxes
... other boxes
mdat
210
710
720
730
740
700

FIG. 8

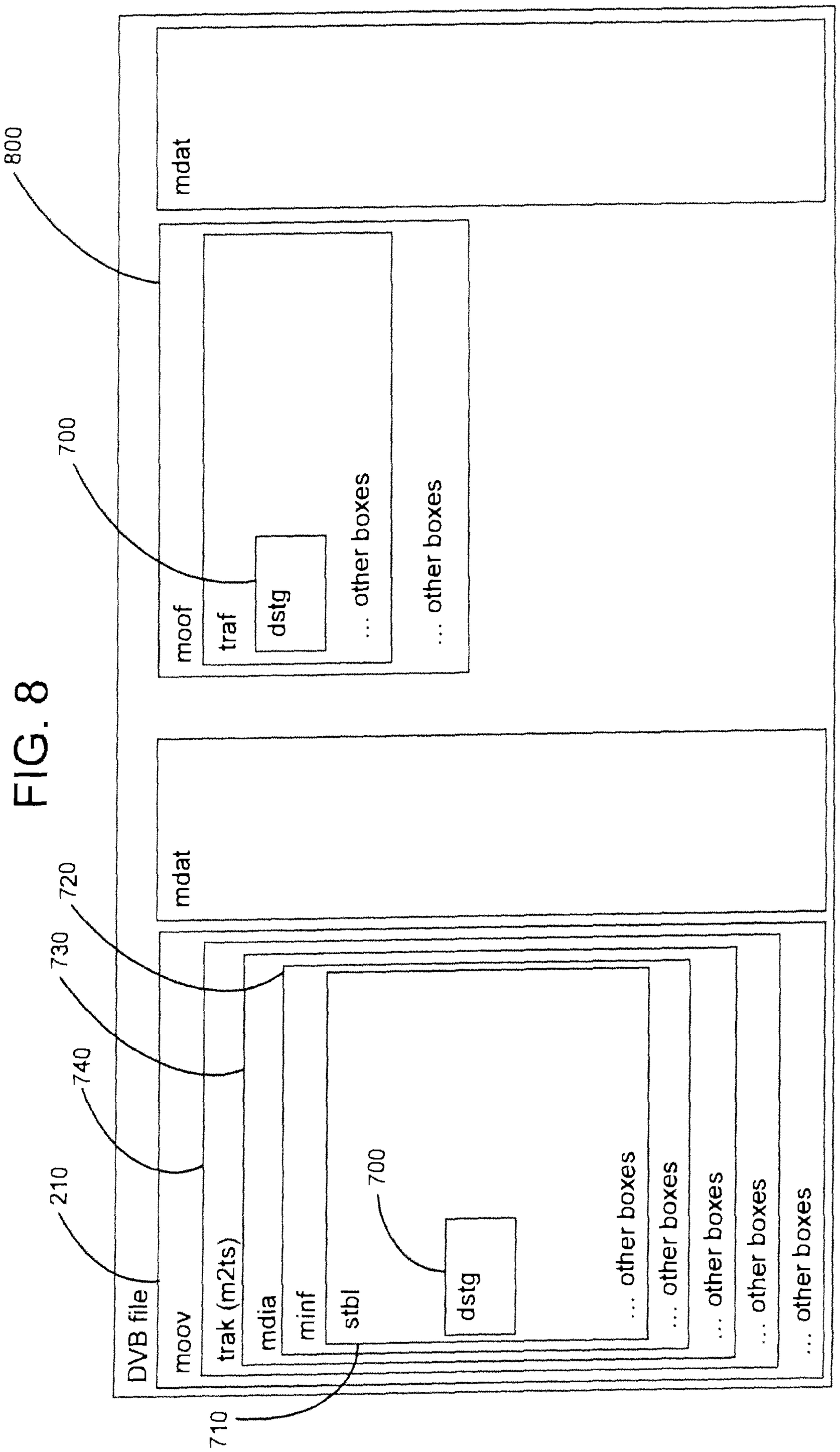

800
mdat
moof
traf
dstg
... other boxes
... other boxes
700
720
730
740
210
DVB file
moov
trak (m2ts)
mdia
minf
stbl
dstg
... other boxes
... other boxes
... other boxes
... other boxes
... other boxes
700
710

FIG. 9

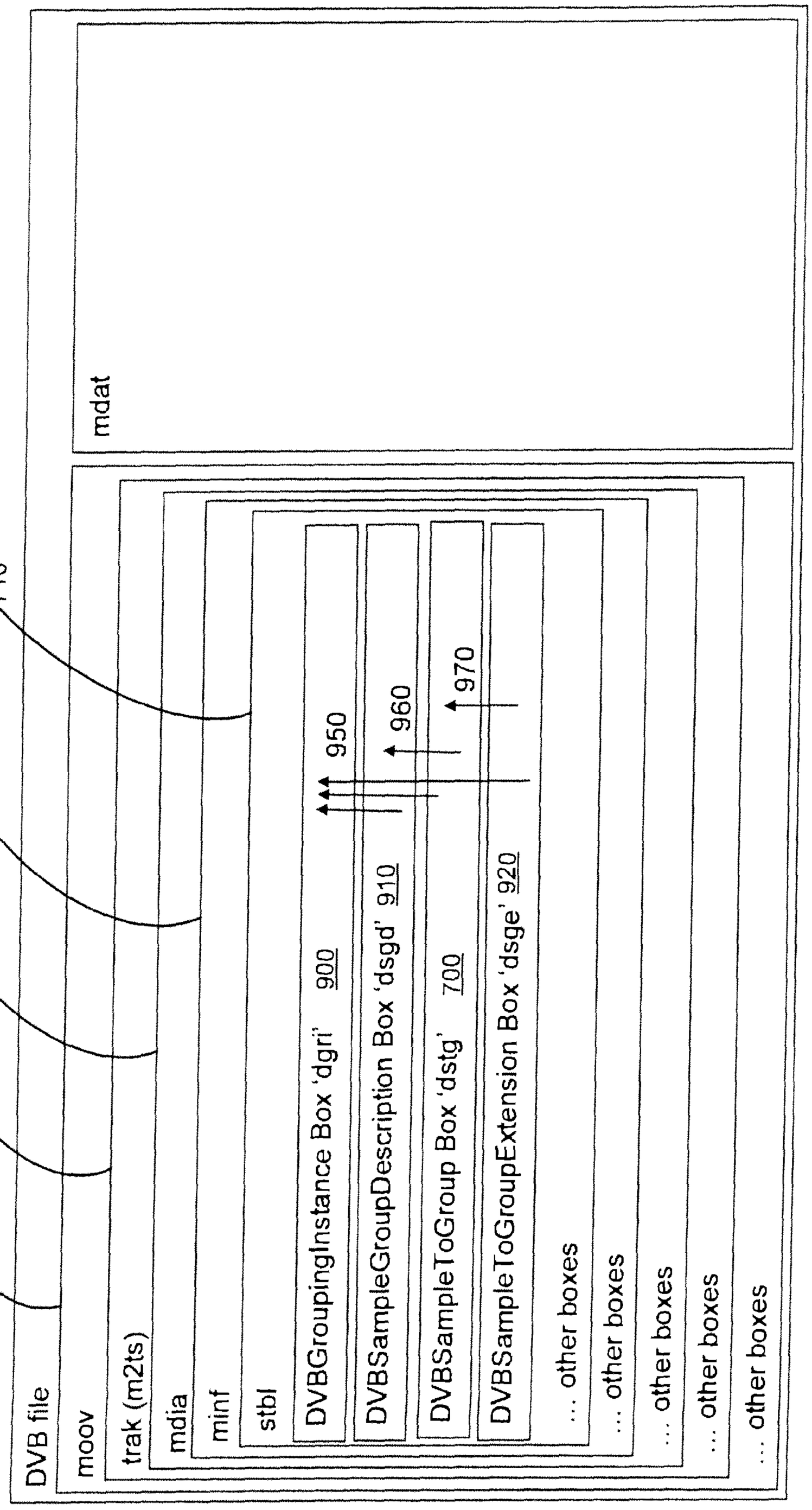
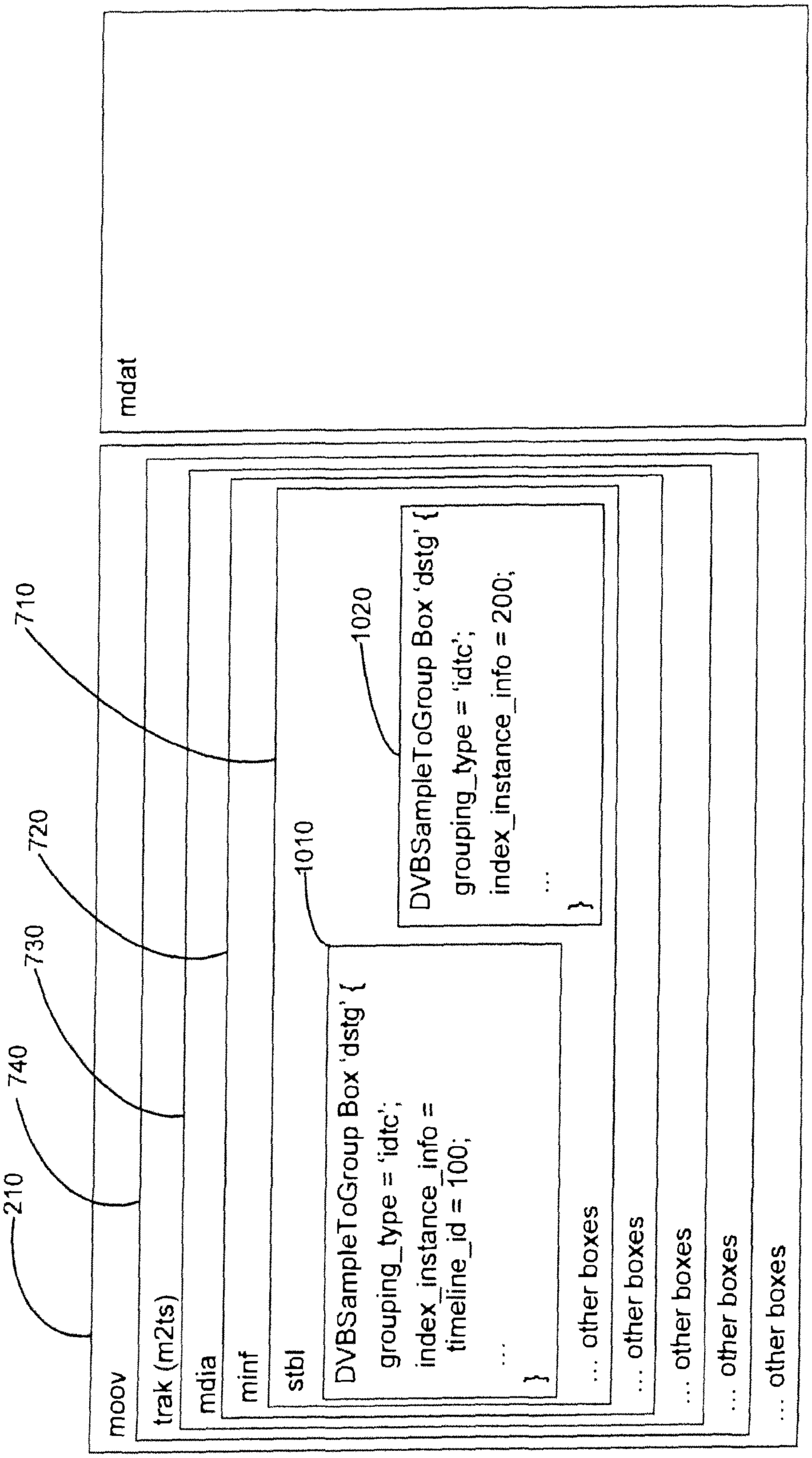
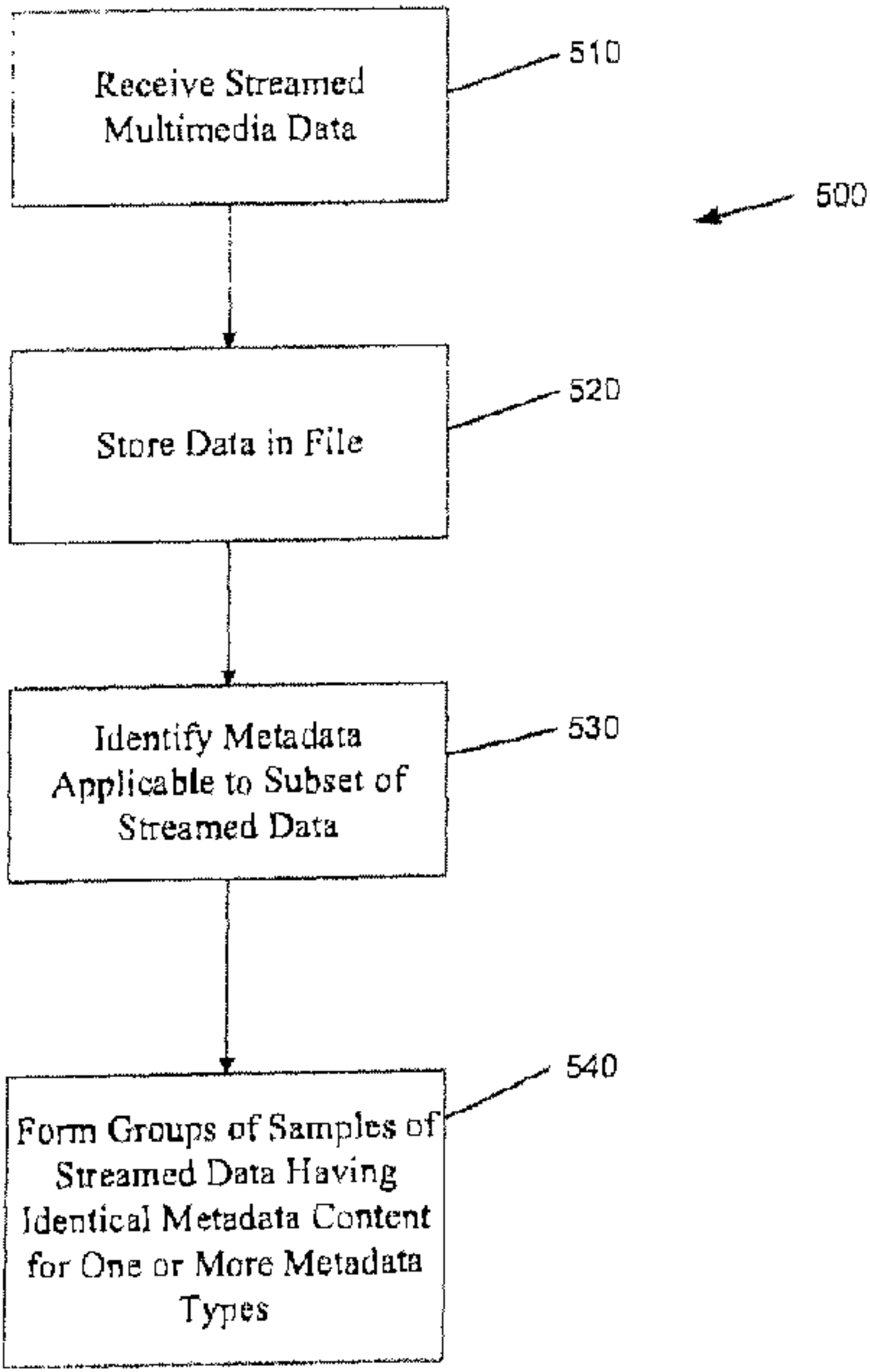

DVB file
moov
trak (m2ts)
mdia
minf
stbl
DVBGroupingInstance Box 'dgri' 900
DVBSampleGroupDescription Box 'dsgd' 910
DVBSampleToGroup Box 'dstg' 700
DVBSampleToGroupExtension Box 'dsge' 920
... other boxes
... other boxes
... other boxes
... other boxes
... other boxes
mdat
210
710
720
730
740
950
960
970

FIG. 10


710
720
730
740
210
1010
1020
moov
trak (m2ts)
mdia
minf
stbl
DVBSampleToGroup Box 'dstg' {
grouping_type = 'idtc';
index_instance_info =
timeline_id = 100;
...
}
DVBSampleToGroup Box 'dstg' {
grouping_type = 'idtc';
index_instance_info = 200;
...
}
... other boxes
... other boxes
... other boxes
... other boxes
... other boxes
mdat


Receive Streamed Multimedia Data
510
500
Store Data in File
520
Identify Metadata Applicable to Subset of Streamed Data
530
Form Groups of Samples of Streamed Data Having Identical Metadata Content for One or More Metadata Types
540