

(12) 特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2016年4月28日(28.04.2016)



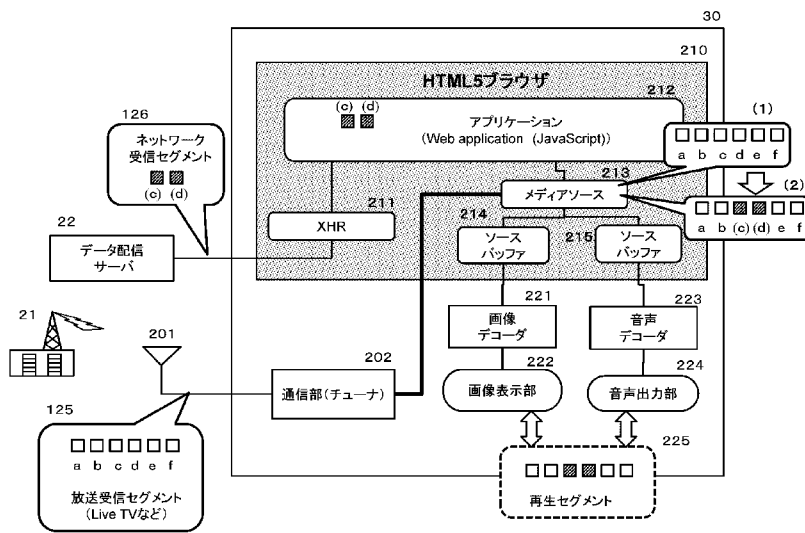
(10) 国際公開番号
WO 2016/063780 A1

- (51) 国際特許分類:
H04N 21/435 (2011.01) H04N 21/44 (2011.01)
H04N 21/431 (2011.01) H04N 21/442 (2011.01)
H04N 21/438 (2011.01)
- (21) 国際出願番号: PCT/JP2015/079098
- (22) 国際出願日: 2015年10月14日(14.10.2015)
- (25) 国際出願の言語: 日本語
- (26) 国際公開の言語: 日本語
- (30) 優先権データ:
特願 2014-213498 2014年10月20日(20.10.2014) JP
- (71) 出願人: ソニー株式会社 (SONY CORPORATION) [JP/JP]; 〒1080075 東京都港区港南1丁目7番1号 Tokyo (JP).
- (72) 発明者: 五十嵐 卓也 (IGARASHI, Tatsuya); 〒1080075 東京都港区港南1丁目7番1号 ソニー株式会社内 Tokyo (JP). 吉川 典史 (KIKKAWA, Norifumi); 〒1080075 東京都港区港南1丁目7番1号 ソニー株式会社内 Tokyo (JP). 出葉 義治 (DEWA, Yoshiharu); 〒1080075 東京都港区港南1丁目7番1号 ソニー株式会社内 Tokyo (JP). 山岸 靖明 (YAMAGISHI, Yasuaki); 〒1080075 東京都港区港南1丁目7番1号 ソニー株式会社内 Tokyo (JP).
- (74) 代理人: 宮田 正昭, 外 (MIYATA, Masaaki et al.); 〒1040032 東京都中央区八丁堀三丁目25番9号 KSKビル西館8階 特許業務法人 大同特許事務所 Tokyo (JP).
- (81) 指定国 (表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, 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, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.
- (84) 指定国 (表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, RU, TJ, TM), ヨーロッパ (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC,

[続葉有]

(54) Title: RECEIVING DEVICE, TRANSMITTING DEVICE, AND DATA PROCESSING METHOD

(54) 発明の名称: 受信装置、送信装置、およびデータ処理方法



- 22 Data distributing server
- 125 Broadcast receiving segment (live TV, etc.)
- 126 Network receiving segment
- 202 Communication unit (tuner)
- 210 HTML5 browser
- 212 Application (Web application (JavaScript))
- 213 Media source
- 214, 215 Source buffer
- 221 Image decoder
- 222 Image display unit
- 223 Sound decoder
- 224 Sound outputting unit
- 225 Reproducing segment

(57) Abstract: Provided are devices and a method that efficiently execute a process of synthesizing broadcast receiving data and network receiving data. An application programming interface (API) is applied to set the broadcast receiving data received by a receiving device via a communication unit to a media source object that is to be processed by an application executed by the receiving device. The application executes the synthesizing process of the broadcast receiving data with the network receiving data received via a network, as a process for the media source object. The application obtains a time offset equivalent to a time difference between an application time axis and a broadcast time axis through the API application process, thereby executing the highly-precise and short-delay data synthesizing process.

(57) 要約: 放送受信データとネットワーク受信データの合成処理を効率的に実行する装置、方法を提供する。受信装置が通信部を介して受信する放送受信データを、API

(Application Programming Interface) を適用して、受信装置の
実行するアプリケーションの処理オブジェクトであるメディアソースオブジェクトに設定する。アプリケーションは、メディアソースオブジェクトに対する処理として、前記放送受信データと、ネットワークを介して受信するネットワーク受信データとの合成処理を実行する。アプリケーションは、API 適用処理によってアプリケーション時間軸と放送時間軸の時間差に相当する時間オフセットを取得し、高精度、低遅延のデータ合成処理を実行する。

WO 2016/063780 A1

MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, 添付公開書類:
TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, — 國際調查報告 (條約第 21 條(3))
KM, ML, MR, NE, SN, TD, TG).

明 細 書

発明の名称：受信装置、送信装置、およびデータ処理方法

技術分野

[0001] 本開示は、受信装置、送信装置、およびデータ処理方法に関する。さらに詳細には例えば放送波やネットワークを介したデータの送信または受信を実行する受信装置、送信装置、および通信データに対するデータ処理方法に関する。

背景技術

[0002] 放送局やコンテンツサーバ等、コンテンツを提供する送信装置から、テレビ、PC、携帯端末等の受信装置に対して、放送波等による一方向通信、あるいは、インターネット等のネットワークを介した双方向通信、一方向通信によって、放送番組等のコンテンツを送受信するシステムについての開発や規格化が、現在、盛んに進められている。

[0003] しかし、ネットワークを介したデータ送信においては、ネットワークの輻輳等により、一定のビットレートでのデータ受信が困難となる場合がある。

このような問題を解決するため、複数のビットレートのデータの細分化ファイルであるセグメントをネットワーク上に送信し、ネットワーク状況に応じてクライアント側で最適なビットレートのセグメントを選択して再生するアダプティブ（適応型）ストリーミングが提案されている。

[0004] このアダプティブ（適応型）ストリーミングの規格として、DASH（Dynamic Adaptive Streaming over HTTP）がある。DASHは、HTTP（HyperText Transfer Protocol）プロトコルを使用したアダプティブ（適応型）ストリーミング技術の標準規格である。

[0005] アダプティブ（適応型）ストリーミングでは、コンテンツ配信サーバは様々なクライアントで配信コンテンツの再生を可能とするため、複数のビットレートの動画コンテンツの細分化ファイルとこれらの属性情報やURLを記

述したマニフェスト・ファイルを作成し保持する。

[0006] クライアントは、マニフェスト・ファイルをサーバから取得して、自装置の表示部のサイズや利用可能な通信帯域に応じた最適なビットレートコンテンツを選択し、選択コンテンツを受信して再生する。ネットワーク帯域の変動に応じてビットレートの動的な変更も可能であり、クライアント側では、状況に応じた最適なコンテンツを随時切り替えて受信することが可能となり、映像途切れの発生を低減した動画コンテンツ再生が実現される。なお、アダプティブ（適応型）ストリーミングについては、例えば特許文献1（特開2011-87103号公報）に記載がある。

[0007] MPEG (Moving Picture Expert Group) 符号化された動画や音声データを上記のDASHに従ってストリーミング配信するための仕様を定めた規格としてMPEG-DASH規格がある。

MPEG-DASH規格には、以下の2つの規格が含まれる。

(a) 動画や音声ファイルの管理情報であるメタデータを記述するためのマニフェスト・ファイル (MPD: Media Presentation Description) に関する規格。

(b) 動画コンテンツ伝送用のファイル・フォーマット (セグメント・フォーマット) に関する規格。

MPEGデータをDASHに従ってストリーミング配信する場合は、このMPEG-DASH規格に従った処理が行われる。

[0008] 一方、放送、インターネットからのコンテンツ配信を受信するクライアント (受信装置) には対話的サービスを行うためにHTML5ブラウザが搭載され、HTML5にて記述されたアプリケーションがクライアントで実行される。

上記アダプティブ (適応型) ストリーミングもHTML5アプリケーションのJavaScript (登録商標) によって制御が行われる。

WWW (World Wide Web) 利用技術の国際的標準化団体であるW3C (World Wide Web Consortium) は、

このためのAPI (Application Programming Interface) の標準規格として、MSE (Media Source Extensions) - APIを規定している。

このMSE-APIを利用することで、アダプティブストリーミングのみならず、アプリケーションがユーザの属性に合わせた広告をコンテンツに挿入を行うことも可能である。

[0009] しかし、例えば、テレビ等のクライアント（受信装置）が、放送波を介した受信データにネットワークを介して受信するデータをアプリケーションの制御により重畳、あるいは置き換えて出力しようとする場合、データ出力時間の遅延が発生するという問題がある。

これは、例えば、MSE-APIでは、アプリケーションがチューナー受信データとネットワーク受信データを処理し、さらに合成処理において実行される様々なデータ処理の処理時間に起因する。

また、広告挿入において複数のHTML5ビデオをタイミングを合わせて重畳表示する場合でも、アプリケーションの制御によって放送番組の映像、音声と広告の映像、音声の出力タイミングを正確に合わせることは難しい。

これは、HTML5ブラウザ等のアプリケーション実行環境では、アプリケーションが一秒以下の精度でタイミング合わせをするようなリアルタイム処理に適さないという問題に起因する。

[0010] さらに、放送システムの場合は、番組もしくはチャンネル毎に異なるアプリケーションを切り替えて実行するのが一般的である。従って、アプリケーションがMSE-APIを利用し放送波を介して受信した映像音声データとインターネットの映像音声データを制御するような場合は、番組の切り替え、チャンネルの切り替えで映像、音声再生の連続性が失われるという重大な問題がある。

先行技術文献

特許文献

[0011] 特許文献1：特開2011-87103号公報

発明の概要

発明が解決しようとする課題

[0012] 本開示は、例えば、上記問題に鑑みてなされたものであり、放送波、もしくは、ネットワークから受信した映像音声データの合成処理をアプリケーションが行うとともに、再生遅延の減少や、連続的な映像や音声データの再生を可能とする受信装置、送信装置、およびデータ処理方法を提供することを目的とする。

課題を解決するための手段

[0013] 本開示の第1の側面は、
受信装置の受信データのバッファ処理を、受信装置が実行するアプリケーションのメディア再生の処理オブジェクトであるメディアソースオブジェクトとして開示するデータ処理部を有する受信装置にある。

[0014] さらに、本開示の第2の側面は、
受信装置において利用されるアプリケーションを送信する通信部を有し、
前記アプリケーションは、受信装置における受信データのバッファ処理を開示するメディアソースオブジェクトを利用して、受信データとアプリケーションの受信データとの合成処理を実行させるアプリケーションである送信装置にある。

[0015] さらに、本開示の第3の側面は、
受信装置において実行するデータ処理方法であり、
前記受信装置のデータ処理部が、受信データを受信装置が実行するアプリケーションのメディア再生の処理オブジェクトであるメディアソースオブジェクトに設定するデータ処理方法にある。

[0016] さらに、本開示の第4の側面は、
送信装置において実行するデータ処理方法であり、
送信装置の通信部が、受信装置において利用されるアプリケーションを送信し、
前記アプリケーションは、受信装置における受信データのバッファ処理を

開示するメディアソースオブジェクトを利用して、受信データとアプリケーションの受信データとの合成処理を実行させるアプリケーションであるデータ処理方法にある。

[0017] 本開示のさらに他の目的、特徴や利点は、後述する本開示の実施例や添付する図面に基づくより詳細な説明によって明らかになるであろう。なお、本明細書においてシステムとは、複数の装置の論理的集合構成であり、各構成の装置が同一筐体内にあるものには限らない。

発明の効果

[0018] 本開示の一実施例の構成によれば、放送受信データとネットワーク受信データの合成処理を効率的に実行する装置、方法が実現される。

具体的には、例えば受信装置が通信部を介して受信する放送受信データの処理を、HTML 5 ブラウザのMSE API (Application Programming Interface) を適用して、受信装置の実行するアプリケーションの処理オブジェクトであるメディアソースオブジェクトとしてアプリケーションが制御可能なように設定する。アプリケーションは、メディアソースオブジェクトに対する処理として、放送受信データのメディアソースオブジェクトをHTML 5 のビデオオブジェクトに設定し、放送受信データの映像音声再生をアプリケーションから制御する。アプリケーションは放送受信データのメディアソースオブジェクトを利用して前記放送受信データと、ネットワークを介して受信するネットワーク受信データとの合成処理の制御を実行する。アプリケーションは、API適用処理によってアプリケーション時間軸と放送時間軸の時間差に相当する時間オフセットを取得し、高精度、低遅延のデータ合成処理の制御を実行する。

本構成により、放送受信データとネットワーク受信データの合成処理および出力処理を低遅延、高精度に行うことが可能となる。

なお、本明細書に記載された効果はあくまで例示であって限定されるものではなく、また付加的な効果があってもよい。

図面の簡単な説明

- [0019] [図1]本開示の処理を実行する通信システムの一構成例について説明する図である。
- [図2]送信装置の送信データについて説明する図である。
- [図3]送信装置および受信装置のプロトコルスタックの例を示す図である。
- [図4]サービスワーカー（SW）を利用した処理の具体例（ユースケース）について説明する図である。
- [図5]サービスワーカー（SW）を利用した処理の具体例（ユースケース）について説明する図である。
- [図6]サービスワーカー（SW）を利用した処理の一例を説明する図である。
- [図7]放送受信コンテンツと、ネットワーク受信コンテンツとの合成表示処理例について説明する図である。
- [図8]放送受信コンテンツと、ネットワーク受信コンテンツとの合成表示処理例について説明する図である。
- [図9]アダプティブ（適応型）ストリーミングについて説明する図である。
- [図10]アダプティブ（適応型）ストリーミングに従って、様々なビットレートのセグメントを受信して再生する受信装置の処理構成例を示す図である。
- [図11]放送波とネットワークを介する2つの経路の受信データを合成して出力する場合の処理と、遅延要因について説明する図である。
- [図12]本開示の受信装置（クライアント）における合成処理、すなわち放送波の受信データの一部を、ネットワーク受信データに置き換えて出力する処理を説明する図である。
- [図13]放送波の受信データの一部を、ネットワーク受信データに置き換えて出力するメディアソース更新処理について説明する図である。
- [図14]受信装置におけるデータ受信および合成処理に適用するハードウェア構成例について説明する図である。
- [図15]受信装置の実行する全体的な処理のシーケンスを説明するフローチャートを示す図である。
- [図16]受信装置のアプリケーションが実行する処理のシーケンスを説明する

フローチャートを示す図である。

[図17]受信装置のアプリケーションが実行する処理のシーケンスを説明するフローチャートを示す図である。

[図18]通信装置である送信装置と受信装置の構成例について説明する図である。

[図19]通信装置である送信装置と受信装置のハードウェア構成例について説明する図である。

発明を実施するための形態

[0020] 以下、図面を参照しながら本開示の受信装置、送信装置、およびデータ処理方法の詳細について説明する。なお、説明は以下の項目に従って行なう。

1. 通信システムの構成例について
2. データ通信プロトコルFLUTE、およびROUTEについて
3. 送信装置と受信装置の実行する通信処理例について
4. サービスワーカー（SW）について
5. 放送波とネットワーク双方の受信データの合成処理について
6. 放送波とネットワーク双方の受信データの合成処理における遅延を解消または減少させた実施例について
7. 受信装置におけるデータ受信および合成処理に適用するハードウェア構成例について
8. 受信装置の実行する処理のシーケンスについて
9. サービスワーカー（SW）とアプリケーションを利用した処理例について
10. 送信装置と受信装置の構成例について
11. 本開示の構成のまとめ

[0021] [1. 通信システムの構成例について]

まず、図1を参照して本開示の処理を実行する通信システムの一構成例について説明する。

図1に示すように、通信システム10は、画像データや音声データ等のコ

ンテンツを送信する通信装置である送信装置20と、送信装置20の送信するコンテンツを受信する通信装置である受信装置30を有する。

[0022] 送信装置20は、具体的には、例えば放送局21やコンテンツサーバ22等、コンテンツを提供する側の装置である。

一方、受信装置30は、一般ユーザのクライアント装置であり、具体的には、例えばテレビ31、PC32、携帯端末33等によって構成される。

[0023] 送信装置20と受信装置30間のデータ通信は、インターネット等のネットワークを介した双方向通信、一方向通信、あるいは、放送波等による一方向通信の少なくともいずれか、あるいは両者を利用した通信として行われる。

[0024] 送信装置20から受信装置30に対するコンテンツ送信は、例えばアダプティブ（適応型）ストリーミング技術の規格であるMPEG-DASH規格に従って実行する。

MPEG-DASH規格には、以下の2つの規格が含まれる。

(a) 動画や音声ファイルの管理情報であるメタデータを記述するためのマニフェスト・ファイル（MPD: Media Presentation Description）に関する規格、

(b) 動画コンテンツ伝送用のファイル・フォーマット（セグメント・フォーマット）に関する規格、

送信装置20から、受信装置30に対するコンテンツ配信は、上記のMPEG-DASH規格に従って実行する。

[0025] 送信装置20は、コンテンツデータを符号化し、符号化データおよび符号化データのメタデータを含むデータファイルを生成する。符号化処理は、例えばMPEGにおいて規定されるMP4ファイルフォーマットに従って行われる。なお、送信装置20がMP4形式のデータファイルを生成する場合の符号化データのファイルは「mdat」、メタデータは「moov」や「mof」等と呼ばれる。

[0026] 送信装置20が受信装置30に提供するコンテンツは、例えば音楽データ

や、映画、テレビ番組、ビデオ、写真、文書、絵画および図表などの映像データや、ゲームおよびソフトウェアなど、様々なデータである。

[0027] 送信装置20の送信データについて図2を参照して説明する。

MPEG-DASH規格に従ってデータ送信を実行する送信装置20は、図2に示すように、大きく分けて以下の複数種類のデータの送信を行う。

- (a) シグナリングデータ50
- (b) AVセグメント60
- (c) その他のデータ (ESG, NRTコンテンツ等) 70

[0028] AVセグメント60は、受信装置において再生する画像 (Video) や、音声 (Audio) データ、すなわち例えば放送局の提供する番組コンテンツ等によって構成される。例えば、上述したMP4符号化データ (mdat) や、メタデータ (moov, moof) によって構成される。

[0029] 一方、シグナリングデータ50は、番組表等の番組予定情報や、番組取得に必要なアドレス情報 (URL等)、さらにコンテンツの再生処理に必要な情報、例えばコーデック情報 (符号化方式など) などからなる案内情報、制御情報によって構成される。

受信装置30は、このシグナリングデータ50を、再生対象となる番組コンテンツを格納したAVセグメント60の受信に先行して受信することが必要となる。

このシグナリングデータ50は、例えばXML (Extensible Markup Language) 形式のデータとして、スマホやテレビ等のユーザ端末である受信装置 (クライアント) に送信される。

[0030] 前述したように、シグナリングデータは、随時、繰り返し送信される。例えば100msec毎など、頻繁に繰り返し送信される。

これは、受信装置 (クライアント) が、いつでも、即座にシグナリングデータを取得することを可能とするためである。

クライアント (受信装置) は、随時、受信可能なシグナリングデータに基づいて、必要な番組コンテンツのアクセス用アドレスの取得や、コーデック

設定処理など、番組コンテンツの受信および再生に必要な処理を遅滞なく実行することが可能となる。

[0031] その他のデータ70は、例えばESG (Electronic Service Guide)、NRTコンテンツ等が含まれる。

ESGは、電子サービスガイド (Electronic Service Guide) であり、例えば番組表等の案内情報である。

NRTコンテンツはノンリアルタイム型のコンテンツである。

[0032] NRTコンテンツには、例えば、クライアントである受信装置30のブラウザ上で実行される様々なアプリケーションファイル、動画、静止画等のデータファイル等が含まれる。

後述するアプリケーション等の制御プログラムとして利用されるサービスワーカー (Service Worker) も、NRTコンテンツに含まれる。

[0033] 図2に示す以下のデータ、すなわち、

(a) シグナリングデータ50

(b) AVセグメント60

(c) その他のデータ (ESG, NRTコンテンツ等) 70

これらのデータは、例えば、データ通信プロトコル: FLUTE (File Delivery over Unidirectional Transport) に従って送信される。

[0034] [2. データ通信プロトコルFLUTE、およびROUTEについて]

データ通信プロトコル: FLUTE (File Delivery over Unidirectional Transport) はマルチキャストにより伝送するコンテンツのセッション管理を行うプロトコルである。

例えば送信装置であるサーバ側で生成されるファイル (URLとバージョンで識別される) は、FLUTEプロトコルに従って、受信装置であるクライアントに送信される。

[0035] 受信装置（クライアント）30は、例えば受信装置（クライアント）30の記憶部（クライアントキャッシュ）に、受信ファイルのURLおよびバージョンとファイルを対応付けて蓄積する。

同じURLでバージョンが異なるものはファイルの中身が更新されているものとみなす。FLUTEプロトコルは一方向ファイル転送制御のみを行うもので、クライアントにおけるファイルの選択的なフィルタリング機能はないが、FLUTEで転送制御するファイルをそのファイルに紐づけられるメタデータを利用して、クライアント側で取捨選択することにより、選択的なフィルタリングを実現し、ユーザの嗜好を反映したローカルキャッシュを構成・更新管理することが可能となる。

なお、メタデータは、FLUTEプロトコルに拡張して組み込むこともできるし、別途ESG（Electronic Service Guide）等のプロトコルで記述することもできる。

[0036] なお、FLUTEは、当初マルチキャストにおけるファイル転送プロトコルとして仕様化された。FLUTEは、FDTと、ALCと呼ばれるスケラブルなファイルオブジェクトのマルチキャストプロトコル、具体的にはそのビルディングブロックであるLCTやFECコンポーネント、の組み合わせにより構成される。

[0037] 従来のFLUTEは、主に非同期型のファイル転送に利用するために開発されたが、現在、放送波およびネットワークを介したデータ配信システムに関する規格化団体であるATSC（Advanced Television System Committee）において、ブロードキャストライブストリーミングにも適用しやすくするための拡張を行っている。このFLUTEの拡張仕様がROUTE（Real-Time Object Delivery over Unidirectional Transport）と呼ばれる。

[0038] 放送波およびネットワークを介したデータ配信システムに関する規格の1つとして現在、標準化が進められている規格としてATSC（Advanc

ed Television System Committee) 3.0がある。このATSC 3.0は、ROUTEを従来のFLUTEプロトコルに置き換えて、シグナリングデータや、ESG、あるいは非同期ファイル、同期型ストリーム等の送信に採用したスタック構成を規定している。

[0039] [3. 送信装置と受信装置の実行する通信処理例について]

次に、送信装置と受信装置の実行する通信処理例について説明する。

図3は、送信装置および受信装置のプロトコルスタックの例を示す図である。

図3に示す例は、以下の2つの通信データの処理を行なうための2つのプロトコルスタックを有する。

(a) ブロードキャスト（マルチキャストも含む）通信（例えば放送型データ配信）

(b) ユニキャスト（ブロードバンド）通信（例えばHTTP型のクライアントサーバ型通信）

[0040] 図3の左側が（a）ブロードキャスト通信（例えば放送型データ配信）に対応するプロトコルスタックである。

図3の右側が、（b）ユニキャスト（ブロードバンド）通信（例えばHTTP型のクライアントサーバ型通信）に対応するプロトコルスタックである。

[0041] 図3左側に示す（a）ブロードキャスト通信（例えば放送型データ配信）に対応するプロトコルスタックは、下位レイヤから順に、以下のレイヤを持つ。

(1) ブロードキャスト物理レイヤ (Broadcast PHY)

(2) IPマルチキャストレイヤ (IP Multicast)

(3) UDPレイヤ

(4) ROUTE (=拡張型FLUTE) レイヤ

(5) ESG, NRT content, DASH (ISO BMFF) およびVideo/Audio/CC

(6) アプリケーションレイヤ (Applications (HTML 5))

[0042] なお、(2) IPマルチキャストレイヤ (IP Multicast) の上位レイヤとして Signaling (シグナリング) レイヤが設定される。

シグナリングレイヤは、先に図2を参照して説明したシグナリングデータ50の送受信に適用されるレイヤである。シグナリングデータには、番組表等の番組予定情報や、番組取得に必要なアドレス情報 (URL等)、さらにコンテンツの再生処理に必要な情報、例えばコーデック情報 (符号化方式など) などからなる案内情報、制御情報などが含まれる。

なお、(1) ブロードキャスト物理レイヤ (Broadcast PHY) の上位レイヤとして将来の新たなプロトコルの利用許容レイヤ (Future Extensibility) が設定されている。

[0043] (1) ブロードキャスト物理レイヤ (Broadcast PHY) は、ブロードキャスト通信を実行するための例えば放送系の通信部を制御する通信制御部によって構成される物理レイヤである。

(2) IPマルチキャストレイヤ (IP Multicast) は、IPマルチキャストに従ったデータ送受信処理を実行するレイヤである。

(3) UDPレイヤは、UDPパケットの生成、解析処理レイヤである。

[0044] (4) ROUTEレイヤは、拡張型FLUTEプロトコルであるROUTEプロトコルにしたがって転送データの格納や取り出しを行うレイヤである。

ROUTEは、FLUTEと同様、ALCと呼ばれるスケーラブルなファイルオブジェクトのマルチキャストプロトコルであり、具体的にはそのビルディングブロックであるLCTやFECコンポーネントの組み合わせにより構成される。

[0045] (5) ESG, NRTcontent, DASH (ISO BMFF) およびVideo/Audio/CCは、ROUTEプロトコルに従って転送

されるデータである。

[0046] DASH規格に従った同報型配信サービスは、MBMS (Multimedia Broadcast Multicast Service) と呼ばれる。このMBMSをLTEで効率的に実現させる方式としてeMBMS (evolved Multimedia Broadcast Multicast Service) がある。

MBMSやeMBMSは、同報型配信サービスであり、特定のエリア内に位置する受信装置である複数のユーザ端末 (UE) に対して共通のベアラで一斉に同一データ、例えば映画コンテンツなどを配信するサービスである。MBMSやeMBMSに従った同報配信により、配信サービス提供エリアに位置する多数のスマホやPC、あるいはテレビ等の受信装置に、同じコンテンツを同時に提供することができる。

[0047] MBMS、およびeMBMSは、3GPPファイルフォーマット (ISO-BMFFファイル、MP4ファイル) に従ったファイルを、転送プロトコルROUTE、またはFLUTEに従ってダウンロードする処理について規定している。

[0048] 先に図2を参照して説明した以下のデータ、すなわち、

- (a) シグナリングデータ50
- (b) AVセグメント60
- (c) その他のデータ (ESG、NTRコンテンツ等) 70

これらのデータの多くはROUTEプロトコル、またはFLUTEプロトコルに従って送信される。

[0049] (5) ESG, NRTcontent, DASH (ISO BMFF) およびVideo/Audio/CCは、ROUTEプロトコルに従って転送されるデータである。

[0050] ESGは、電子サービスガイド (Electronic Service Guide) であり、例えば番組表等の案内情報である。

[0051] NRTcontentはノンリアルタイム型のコンテンツである。

前述したように、NRTコンテンツには、例えば、クライアントである受信装置のブラウザ上で実行される様々なアプリケーションファイル、動画、静止画等のデータファイル等が含まれる。さらに、後述するアプリケーション等の制御プログラムとして利用されるサービスワーカー（Service Worker（SW））も、NRTコンテンツに含まれる。

Video/Audio/CCは、DASH規格に従って配信されるビデオやオーディオ等、再生対象となる実データである。

[0052] (6) アプリケーションレイヤ（Applications（HTML5））は、ROUTEプロトコルに従って転送するデータの生成、あるいは解析、その他、様々なデータの出力制御等を実行するアプリケーションレイヤであり、例えばHTML5を適用したデータ生成、解析、出力処理等を行う。

[0053] 一方、図3の右側に示す、(b)ユニキャスト（ブロードバンド）通信（例えばHTTP型のP2P通信）に対応するプロトコルスタックは、下位レイヤから順に、以下のレイヤを持つ。

- (1) ブロードバンド物理レイヤ（Broadband PHY）
- (2) IPユニキャストレイヤ（IP Unicast）
- (3) TCPLレイヤ
- (4) HTTPレイヤ
- (5) ESG, Signaling, NRTcontent, DASH（ISO BMFF）およびVideo/Audio/CC
- (6) アプリケーションレイヤ（Applications（HTML5））

[0054] (1) ブロードバンド物理レイヤ（Broadband PHY）は、ブロードバンド通信を実行する例えばネットワークカード等の通信部を制御するデバイスドライバ等の通信制御部によって構成される物理レイヤである。

(2) IPユニキャストレイヤ（IP Unicast）は、IPユニキャスト送受信処理を実行するレイヤである。

(3) HTTPレイヤは、HTTPパケットの生成、解析処理レイヤである。

この上位レイヤは、図3左側の(a)ブロードキャスト通信(例えば放送型データ配信)のスタック構成と同様である。

[0055] なお、送信装置(サーバ)20、受信装置(クライアント)30は、図3の2つの処理系、すなわち、

(a)ブロードキャスト通信(例えば放送型データ配信)

(b)ユニキャスト(ブロードバンド)通信(例えばHTTP型のP2P通信)

これら2つの通信プロトコルスタックの少なくともいずれかに従った処理を行なう。

[0056] 図3に示すプロトコルスタックにおいて、ROUTE(FLUTE)に従ってマルチキャスト転送されるファイル群の属性(ファイルの識別子であるURLを含む)は、ROUTE(FLUTE)の制御ファイル内に記述することもできれば、ファイル転送セッションを記述するシグナリング(Signaling)データ中に記述することもできる。また、ファイル転送セッションのさらなる詳細属性を(エンドユーザへの提示用途にも適用可能な)ESGにより記述することもできる。

[0057] [4. サービスワーカー(SW)について]

次に、送信装置(サーバ)20が提供し、主に受信装置(クライアント)30において利用されるサービスワーカー(SW:Service Worker)について説明する。

サービスワーカー(SW)は、放送サーバ21や、データ配信サーバ22等の送信装置20から受信装置に提供される。

[0058] サービスワーカー(SW)は、受信装置(クライアント)30において実行されるアプリケーション(=アプリケーションプログラム)や、アプリケーションの実行時に利用されるデータファイル等の取得処理や、記憶部(キャッシュ)に対する格納処理、さらに更新処理、削除処理等を実行するプロ

グラムである。具体的には、例えばJavaScript（登録商標）によって構成される。

[0059] サービスワーカー（SW）は、例えば放送サーバ21や、データ配信サーバ22等の送信装置20が提供する放送番組（放送コンテンツ）に対応して設定され、送信装置20から受信装置30に提供されるアプリケーションの制御および管理プログラムとして、受信装置30に提供される。

[0060] サービスワーカー（SW）、アプリケーション、およびアプリケーションの実行時に利用されるデータファイル、これらは、例えば先に図2、図3を参照して説明したNRTコンテンツ（ノンリアルタイムコンテンツ）として、送信装置20から受信装置30に提供される。

あるいは、放送番組を配信するサーバとは別のデータ提供サーバが、サービスワーカー（SW）、アプリケーション、およびアプリケーションの実行時に利用されるデータファイルを受信装置30に提供する構成としてもよい。

[0061] サービスワーカー（SW）は、例えば、受信装置30においてWebページ等の閲覧処理を実行するために利用されるプログラムであるブラウザを利用して情報表示を実行するアプリケーションのリソース管理（取得、保持、更新、削除等）処理などを実行する。

[0062] サービスワーカー（SW）を利用した処理の具体例（ユースケース）について、図4、図5を参照して説明する。

図4は、受信装置30が、放送サーバ21等の送信装置20から、ある番組コンテンツを受信し、受信装置30の表示部に表示している状態を示している。

[0063] 放送サーバ21等の送信装置20は、番組配信に併せて、NRTコンテンツ（ノンリアルタイムコンテンツ）として、天気情報を表示するアプリケーション、およびこの天気情報表示アプリケーションに利用される様々なデータファイル、例えば動画、静止画、音声等の様々なデータを含むデータファイルを受信装置30に提供する。

以下では、これらのアプリケーションおよびデータファイルを「リソース」と呼ぶ。

放送サーバ21は、さらに、これらの「リソース」を管理するリソース管理プログラムとして、サービスワーカー（SW）を、やはりNRTコンテンツ（ノンリアルタイムコンテンツ）として受信装置30に提供する。

[0064] 受信装置30は、送信装置20から受信した「リソース」、すなわちアプリケーションおよびデータファイルを利用して、図4に示すように、番組表示に併せて、天気情報の表示を行うことができる。

このような、アプリケーションを利用したデータ表示は、これまでのデータ配信構成では、アプリケーションの提供される番組の終了とともに実行できなくなってしまう。

[0065] これは、天気情報表示アプリケーション等のリソースは、番組受信中は、受信装置30において利用可能な設定、例えば、一時記憶キャッシュに格納され、利用可能な状態に設定されるが、番組終了、あるいはユーザがチャンネルを切り替えると、これらのキャッシュデータが消去、あるいはアクセス不可能な状態に設定されてしまうためである。

[0066] サービスワーカー（SW）は、このような番組対応のアプリケーションやデータを、番組終了後や、チャンネル切り替え後、あるいは放送の非受信状態、ネットワーク非接続状態等のオフライン状態であっても利用可能とするためのリソース管理プログラムとして機能する。

[0067] 図5に示すように、天気情報表示アプリケーションを、このアプリを提供した番組の終了後や、他のチャンネルに切り替え後、あるいはデータ受信を実行していないオフライン状態であっても、利用することが可能となる。すなわち天気情報を受信装置30の表示部に表示して閲覧することが可能となる。

なお、天気情報表示アプリケーションは、例えばブラウザ上で表示されるプログラムである。

[0068] この天気情報表示アプリケーションは、サービスワーカー（SW）の制御

によって、受信装置30の記憶部（永続キャッシュ）に格納される。例えばユーザによる表示要求等のリクエスト（イベント）があると、サービスワーカー（SW）の制御によって、の記憶部（永続キャッシュ）から読み出され、表示部に表示される。

なお、アプリケーション等のリソースを格納する記憶部（永続キャッシュ）は、受信装置30の電源がオフとなっても格納データが消去されない不揮発性メモリとすることが好ましい。

このように、サービスワーカー（SW）を利用することで、様々な番組対応アプリケーションを、番組の表示、非表示と無関係に利用することが可能となる。

[0069] なお、サービスワーカー（SW）は、例えばある番組対応のリソース（アプリケーション、およびアプリ関連データ）単位ごとに設定され、リソースに併せて、あるいはリソース送信に前後して送信装置20から受信装置30に提供される。

サービスワーカー（SW）は、各番組対応の設定とすることもできるが、複数番組を含む特定のチャンネル対応のリソースに対して、共通に利用可能としたサービスワーカー（SW）を設定することもできる。

サービスワーカー（SW）と、サービスワーカー（SW）によって管理されるリソース（アプリケーションおよびアプリ関連データ）は、受信装置30の記憶部（永続キャッシュ）に格納される。

[0070] 図6は、サービスワーカー（SW）を利用した処理の一例を説明する図である。

図6には、受信装置30が送信装置20からリソースとしてのWebページ（例えば図4、図5に示す天気情報表示ページ）を取得して受信装置30の記憶部（永続キャッシュ）に格納して利用するシーケンスの一例を示している。

なお、Webページは、所定のWebページ表示アプリケーションと、表示用データによって構成されるリソースを利用して表示される。

図6には、受信装置内表示制御部90の構成要素として表示処理部91、サービスワーカー（SW）92、キャッシュ（記憶部）93を示している。

[0071] ステップS101～S102は、受信装置30による送信装置20に対する初回アクセス処理によるリソース（Webページ）取得処理である。

これは、例えば放送サーバ等の送信するNRTコンテンツから取得される。

この取得処理後、表示処理部91によって、Webページ95が、受信装置30の表示部に表示される。この表示は、このWebページを提供している番組に併せて表示されている状態であり、先に図3を参照して説明した表示状態に相当する。

[0072] この表示期間において、例えばユーザによる指示としてリソース（Webページ）の登録（インストール）要求がなされると、ステップS103において、サービスワーカー（SW）92が、リソース（Webページ）の登録（インストール）処理を開始する。

具体的には、ステップS104に示すようにリソースをキャッシュ93に渡し、記憶部（永続キャッシュ）に格納する処理を行なう。

[0073] その後、番組終了後、あるいはチャンネル切り替え後、あるいはオフライン設定状態において、ステップS105において、ユーザがWebページの閲覧要求を行う。

サービスワーカー（SW）92は、この閲覧要求の入力をフェッチイベントとして検出し、フェッチイベント検出に応じて、ステップS106において、リソース（Webページ）を記憶部（永続キャッシュ）から取得する。

表示処理部91は、ステップS107において、Webページ96を表示する。

[0074] このWebページ表示処理は、番組終了後、あるいはチャンネル切り替え後、あるいはオフライン設定状態における表示処理であり、先に図5を参照して説明した表示状態に相当する。

[0075] このように、サービスワーカー（SW）を利用することで、様々な番組対

応アプリケーションを、番組の表示、非表示と無関係に利用することが可能となり、例えば、番組付属の表示情報として設定されたWebページを番組と無関係に、任意のタイミングで表示する等の処理が可能となる。

[0076] このように、サービスワーカー（SW）は、例えば、Webページ、HTMLページ、JavaScript（登録商標）等を構成要素としたアプリケーションや、アプリケーションに利用されるデータ等からなるリソースの取得、保存、更新、削除等の、リソース管理を実行する。

[0077] リソースの保存、あるいはキャッシュは、永続的なキャッシュであり、通常のローカル／テンポラリなキャッシュとは異なり、アプリケーションが稼働していなくても、データが保存される。

Webページ表示プログラムであるブラウザに一種のプロキシサーバが実装され、いつでも、必要なときにプロキシサーバ（永続キャッシュ）をアクセスしてWebページを取得して表示可能としたイメージである。

[0078] なお、サービスワーカー（SW）自身も永続キャッシュに格納（インストール）される。サービスワーカー（SW）が、受信装置にインストールされると、このサービスワーカー（SW）の管理対象とするリソースについて、様々な制御が可能となる。

例えば、リソースへのアクセスリクエスト（リソースへのフェッチリクエスト）に応じて、ブラウザ側の処理（ローカルキャッシュやネットワークからのリソースの取得）が始まる前に、サービスワーカー（SW）の処理が開始され、永続キャッシュからのリソースの提供が行われる。

また、サービスワーカー（SW）は、JavaScript（登録商標）で提供されるため、さまざまな手続きを組み込むことが可能であり、永続キャッシュのリソースの一部の更新等キャッシュ制御についての柔軟な処理記述が可能となっている。

[0079] なお、サービスワーカー（SW）自身も更新可能である。サービスワーカー（SW）は、送信装置20から提供されるが、このサービスワーカー（SW）のヘッダ情報（HTTP Cache-Control）に更新日時情

報や更新データのアクセス情報等、更新処理に必要となる各種情報が記録され、このヘッダ情報に基づいて更新処理が実行される。

例えば、ヘッダに設定された使用期限等に基づいて、使用期限が来ると、受信装置30は、新しいバージョンのサービスワーカー（SW）の取得処理を実行して、キャッシュに格納された旧バージョンのSWを置き換える更新処理を行なう。

[0080] [5. 放送波とネットワーク双方の受信データの合成処理について]

受信装置30は、送信装置である放送サーバ21から放送波を介して番組コンテンツを受信し、さらに、データ配信サーバ22からネットワークを介して様々なコンテンツを受信することが可能である。受信装置30は、これらの2つの通信経路を併用して、2つの通信経路から受信したデータを合成して表示することができる。

[0081] 例えば、受信装置において実行するアプリケーション、例えば上述したサービスワーカー（SW）の管理アプリケーション等、受信装置30において実行されるアプリケーションを適用して、2つの通信経路から受信したデータを合成する処理を実行させることが可能となる。

[0082] 例えば、アプリケーションは、

(a) 放送波を介して受信する番組コンテンツ

(b) ネットワークを介して受信する広告コンテンツ

これらの2つのデータを合成して表示する処理を行なう。

[0083] 具体的な処理例について図7、図8を参照して説明する。

図7には、受信装置（クライアント）30の表示部に表示されるデータの時間経過に伴う推移を示している。

(1) 時間 $t_0 \sim t_1$ では、放送波を介して受信する番組コンテンツを表示している。

(2) 時間 $t_1 \sim t_2$ では、放送波を介して受信する番組コンテンツに、ネットワークを介して受信する広告コンテンツを重畳して表示している。

(3) 時間 $t_2 \sim t_3$ では、放送波を介して受信する番組コンテンツを表

示している。

[0084] (2) の時間 $t_1 \sim t_2$ においてネットワークを介して受信する広告コンテンツは、例えば特定の地域や特定のユーザを対象とした広告コンテンツ（ターゲット広告）としてユーザに提供することができる。

受信装置 30 が、(2) の時間 $t_1 \sim t_2$ においてネットワークを介して受信する広告コンテンツは、受信装置 30 においてアプリケーションを実行している場合に受信可能なコンテンツとなる。

アプリケーションを実行していない受信装置では、放送波を介して受信するデータのみが表示される。

[0085] 図 7 に示す構成は、時間 $t_1 \sim t_2$ において、放送波の受信データにネットワーク受信データを併せて表示（並列表示）する例である。この場合は、広告コンテンツを番組コンテンツに重畳し再生するために複数の映像復号化装置が必要である。

この他、受信装置 30 のアプリケーションの処理態様としては、2つの受信データの並列表示ではなく、放送波を介する受信データの一部期間のデータを、ネットワーク受信データに完全に置き換えて出力する処理も可能である。この場合は、一つの映像復号化装置にて再生可能である。

この置換表示例を図 8 に示す。

[0086] (1) 時間 $t_0 \sim t_1$ では、放送波を介して受信する番組コンテンツを表示している。

(2) 時間 $t_1 \sim t_2$ では、ネットワークを介して受信する広告コンテンツを表示している。

(3) 時間 $t_2 \sim t_3$ では、放送波を介して受信する番組コンテンツを表示している。

[0087] (2) の時間 $t_1 \sim t_2$ においてネットワークを介して受信する広告コンテンツは、例えば特定の地域や特定のユーザを対象とした広告コンテンツ（ターゲット広告）としてユーザに提供することができる。

受信装置 30 が、(2) の時間 $t_1 \sim t_2$ においてネットワークを介して

受信する広告コンテンツは、受信装置30がアプリケーションを実行している場合において受信可能なコンテンツとなる。

アプリケーションを実行していない受信装置では、時間 $t_1 \sim t_2$ の間に、放送波を介して受信する不特定多数向けの広告コンテンツを受信して表示する設定とすることができる。

[0088] このように、アプリケーションの処理によって、放送受信データとネットワーク受信データとの合成表示処理が実行できる。

しかし、この、放送受信データとネットワーク受信データの合成処理の処理時間が長くなると、再生遅延を発生させてしまう場合がある。

特に、例えば図8に示すように、放送受信データをネットワーク受信データに置き換えて表示する場合、合成処理に時間を要すると、図8に示す時間 t_1 において、広告表示が実行されず、画像が途切れてしまうといった事態が発生する可能性がある。

以下、この問題点について説明する。

[0089] この問題点の説明の前に、ネットワーク状況に応じてクライアント側で最適なビットレートのセグメントを選択して再生するアダプティブ（適応型）ストリーミング処理について説明する。

前述したように、ネットワークを介したデータ送信においては、ネットワークの輻輳等により、一定のビットレートでのデータ受信が困難となる場合がある。

このような問題を解決するため、複数のビットレートのデータの細分化ファイルであるセグメントをネットワーク上に送信し、ネットワーク状況に応じてクライアント側で最適なビットレートのセグメントを選択して再生するアダプティブ（適応型）ストリーミングが提案されている。

[0090] 受信装置30は、ネットワーク受信データに限らず、放送受信データについてもネットワーク状況や、地震の表示装置の解像度情報に応じて最適なビットレートのデータを選択取得して出力可能な構成を持つ。

なお、受信装置30は、どのようなビットレートのデータが送信されるか

についての情報を、送信装置 20 が提供するマニフェストファイルに基づいて取得することができる。

[0091] 図 9 は、アダプティブ（適応型）ストリーミングについて説明する図である。

データ配信サーバ 22 は、配信データとして、例えばライブ映像からなるコンテンツを配信する場合、予め複数の異なるビットレートのデータを用意する。

図に示す例では、

高ビットレート（2 Mbps）データ 101、

中ビットレート（1.5 Mbps）データ 102、

低ビットレート（1 Mbps）データ 103、

これらのデータを配信データとする。

[0092] これらのデータは細分化されたデータであるセグメント単位でネットワークを介して受信装置 30 に提供される。A1～An, B1～Bn, C1～Cn の各々がセグメントデータである。

A1, B1, C1 はすべて同一シーンのデータであり、ビットレートが異なるデータ（セグメント）として構成されている。

各セグメントは、1つのセグメント単位で復号可能なデータ、たとえば MP4 ファイルを含むセグメントである。

さらに、セグメントの属性情報や URL を記述したマニフェスト・ファイル 104 を、ネットワークを介して受信装置 30 に提供する。

[0093] 受信装置（クライアント）30 は、マニフェスト・ファイル 104 をサーバから取得して、自装置の表示部のサイズや利用可能な通信帯域に応じた最適なビットレートデータ（セグメント）を選択し、選択データを受信して再生する。ネットワーク帯域の変動に応じてビットレートの動的な変更も可能であり、クライアント側では、状況に応じた最適なデータを随時切り替えて受信することが可能となり、映像途切れの発生を低減した動画コンテンツ再生が実現される。

図に示す再生データ105が受信装置30における再生データであり、様々なビットレートのデータ（セグメント）が混在して再生される。

例えば、図7、図8を参照して説明した広告コンテンツもこのようなセグメントデータの組み合わせによって構成される。

[0094] 図10は、アダプティブ（適応型）ストリーミングに従って、様々なビットレートのセグメントを受信して再生する受信装置30の処理構成例を示す図である。

図10に示す構成は、例えば受信装置30において実行されるブラウザによって利用されるAPI（Application Programming Interface）として実装可能なMSE（Media Source Extensions）-APIの処理構成である。

[0095] メディアソース110は、例えば外部サーバから受信する再生対象データを示すオブジェクトである。このオブジェクトには、複数のビットレートのセグメントデータが含まれる。

なお、オブジェクトとは、アプリケーションによる処理やアクセスが可能な要素を意味し、受信装置30の受信するデータや、記憶部に格納されるデータ、あるいは受信装置の記憶部や通信部等のハードウェアがオブジェクトに設定可能な要素となる。

メディアソース110は、例えば受信装置30の記憶部に格納されたビデオや音声データに相当する。

[0096] MSE-APIは、メディアソース110をそれぞれのビットレートのデータ（セグメント）単位で振り分け、個別のソースバッファ111~113を設定する。

これら個別のソースバッファ111~113は、メディアソース110の構成要素であり、いずれもアプリケーションによる処理対象となるオブジェクトである。

ソースバッファ111は、ビデオデータと2chオーディオデータの各セグメントを含むデータのソースバッファである。

ソースバッファ 112 は、ビデオデータのみ、ソースバッファ 113 はオーディオデータのみを格納するソースバッファである。

[0097] これらの各データは、それぞれ再生時間を同期させたトラックバッファ 114 に振り分けられ、ビデオ、オーディオ個別のデコーダ 115 においてデコード処理がなされた後、画像出力部 117、音声出力部 116 に出力される。

なお、画像出力部 117 には、いずれか 1 つのビデオデコード結果が出力され、音声出力部 116 にはいずれか 1 つのオーディオデコード結果が出力される。

どのデコード結果、すなわちどのビットレートデータを選択して出力するかは、受信装置（クライアント）30 側において、ネットワーク状況や受信装置の出力部機能等に応じて決定することができる。

[0098] このように、受信装置は、ネットワークを介して受信する様々なビットレートのデータを適宜、選択して出力することが可能となる。

しかし、このようなアプリケーションが MSE API を利用してネットワーク受信データを放送波に合成して出力する処理に時間を要すると合成画像の出力に遅延が発生する場合がある。

[0099] 遅延の具体的要因としては、例えば以下のような要因があると考えられる。

(A) ライブ放送での再生遅延の要因

合成処理を実行するアプリケーションは、放送波によって送信される放送ストリームを、XHR (XMLHttpRequest) によってセグメント単位で読み込む。読み込みセグメントは、図 10 を参照して説明したネットワーク経由の受信セグメントの格納領域を想定したオブジェクトとしてのソースバッファに追加 (Append) する。この一連の処理により遅延が生じる。

[0100] (B) 番組切り替え時、チャンネル切り替え時における遅延

合成処理を実行するアプリケーションは、番組単位、あるいはチャンネル

単位で設定されており、番組切り替え時、チャンネル切り替え時にはアプリケーションも切り替える必要がある。この切り替え処理によって遅延が発生し、また、連続して再生できなく映像が静止したり黒画像が挿入される問題がある。

[0101] このように、アプリケーションがネットワーク受信データを放送波に合成して出力する場合は上記のような問題がある。

図11を参照して、放送波とネットワークを介する2つの経路の受信データを合成して出力する処理と、遅延要因について説明する。

なお、以下では、放送波の受信データの一部をネットワーク受信データに置き換えて出力する場合の処理例について説明する。すなわち、図8を参照して説明したように、放送受信データの一部期間のデータをネットワーク受信データに置き換えて出力する処理例について説明する。

[0102] 図11は、受信装置（クライアント）30における合成処理、すなわち放送受信データの一部を、ネットワーク受信データに置き換えて出力する処理を説明する図である。

受信装置（クライアント）30は、放送サーバ21からの放送波を、アンテナ131を介して受信する。

さらに、データ配信サーバ22からデータ受信を行う。

[0103] アンテナ131を介して受信するデータを放送受信セグメント125、データ配信サーバ22からの受信データをネットワーク受信セグメント126とする。

放送受信セグメント125は、例えば映画等の番組コンテンツであり、ネットワーク受信セグメント126は、番組コンテンツの決められたタイミングに表示する広告コンテンツであるとする。

[0104] 図に示す例では、放送受信セグメント125の一部セグメントa, b, c, d, e, fの6つのセグメントを示している。

一方、ネットワーク受信セグメント126として(c), (d)の2つのセグメントを示している。

なお、各セグメントは、各セグメント単位で復号可能な例えばMP E G圧縮されたデータ（MP 4データ）である。

データの置き換えはセグメント単位で実行する。

[0105] ネットワーク受信セグメントである2つの広告セグメント（c）、（d）を、放送受信セグメント125のセグメントa、b、c、d、e、f中のセグメントc、dに置き換えて出力する。

すなわち、受信装置30における出力コンテンツを以下のセグメント列とする。

a、b、（c）、（d）、e、f

上記セグメント列中、セグメントa、b、e、fは、放送受信セグメントであり、セグメント（c）、（d）は、ネットワーク受信セグメントである。

[0106] 受信装置30は、ブラウザ（HTML5ブラウザ）140を有し、ブラウザ140上でアプリケーション（Web application（JavaScript（登録商標））142）を実行する。

アプリケーション142は、例えばAPI（先に説明した図10に示すMSE-API）等を利用した処理により、放送受信セグメント125と、ネットワーク受信セグメント126との合成処理を行なう、

[0107] 放送受信セグメント125は、受信装置に設定されるプロキシサーバとしてのHTTPサーバ133に一旦、格納される。

HTTPサーバ133にデータを格納することで、アプリケーション142は、HTTPサーバ133から、XHR（XMLHttpRequest）141を適用してセグメント単位のデータ読み込みを行うことが可能となる。

[0108] アプリケーション142は、HTTPサーバ133に格納された放送受信セグメント125を、XHR（XMLHttpRequest）141によってセグメント単位で読み込む。

アプリケーション142によるセグメント読み込みにおいて、ネットワー

ク受信セグメント (c), (d) によって置き換え対象となる放送受信セグメント c, d は読み込まず、出力対象となる放送受信セグメント a, b, e, f のみの読み込み処理を行なう。

[0109] その後、アプリケーション 142 は、読み込んだ放送受信セグメント a, b, e, f と、データ配信サーバ 22 から受信した放送受信セグメント (c), (d) を合成する。

アプリケーション 142 は、先に図 8 を参照して説明した放送波として受信する番組コンテンツを非表示として、ネットワークを介して受信する広告コンテンツを番組コンテンツに置き換えたセグメント列、すなわち、

a, b, (c), (d), e, f

上記セグメント列を生成する。

[0110] アプリケーション 142 は、放送受信セグメント 125 の一部のセグメントをネットワーク受信セグメント 126 に置き換えたセグメント列、

a, b, (c), (d), e, f

上記セグメント列からなるメディアソース 143 の生成処理を実行する。

このメディアソース 143 は、図 10 に示すメディアソース 110 に相当する。

なお、図 10 に示す例では、メディアソース 110 には様々なビットレートのデータが混在するものとして説明したが、図 11 に示す例では、説明を簡略化するため、すでに出力予定の 1 つのビットレートが決定されており、そのビットレートのセグメントに対する処理を行うものとして説明する。

[0111] このように、アプリケーション 142 は、放送受信セグメント 125 の一部のセグメントをネットワーク受信セグメント 126 に置き換えたメディアソース 143 の生成処理を実行する。

その後、メディアソース 143 は、データの種別に応じて振り分けられ、画像対応ソースバッファ 144 と、音声対応ソースバッファ 145 が生成される。

[0112] 画像対応ソースバッファ 144 は、画像デコーダ 134 によってデコード

され、画像表示部135に表示される。

音声対応ソースバッファ145は、音声デコーダ136によってデコードされ、音声出力部137に出力される。

この結果として再生されるセグメントが図に示す再生セグメント127であり、放送受信セグメント125の一部のセグメントがネットワーク受信セグメント126に置き換えて再生される。

例えば図8に示す時間 $t_0 \sim t_3$ の一連のデータ表示が実行される。

[0113] 図11を参照して説明したアプリケーション142によって実行される放送受信セグメント125とネットワーク受信セグメント126との合成処理は、先に図10を参照して説明したMSE (Media Source Extensions) -APIを利用して実行される。

このAPIの処理アルゴリズムは例えば、以下のアルゴリズムとなる。

```
<script>
var video=document.getElementById
('v');
var mediaSource=new MediaSource()
;
mediaSource.addEventListener('sourceopen',
onSourceOpen.bind(this, video));
video.src=window.URL.createObject
URL(mediaSource);
var videoSourceBuffer=mediaSource
.addSourceBuffer('video/mp4;codecs=
"avc1.4d401f");
var audioSourceBuffer=mediaSource
.addSourceBuffer('audio/mp4;codecs=
"mp4a.40.2")
```

</script>

[0114] 上記のAPIの処理の概要は以下の通りである。

(S1) 新規のメディアソース143を作成する。

(S2) メディアソース143対応のビデオソースバッファ144と、オーディオソースバッファ145を追加し、HTML5のビデオオブジェクトにメディアバッファを設定する。

アプリケーション142は、これらのAPI処理によって生成されたメディアソース143、ビデオソースバッファ144、オーディオソースバッファ145に上述した合成セグメントを設定する処理を行なう。

[0115] しかし、図11を参照して説明したセグメント合成処理において、問題となるのが再生遅延である。具体的には、例えばネットワーク受信コンテンツの表示タイミングが、規定の時間より遅れてしまう場合があるということである。特にライブイベントの番組コンテンツやチャンネル切り替え場合にこの遅延の問題が大きい。

これは、図11を参照して説明した一連の合成処理に時間を要してしまうということが大きな要因である。

[0116] 特に、以下の処理の処理時間が遅延の要因として考えられる。

(a) 放送受信セグメント125のHTTPサーバ133に対する格納と、アプリケーション142によるHTTPサーバ133からのセグメント単位の読み込み処理

(b) アプリケーション142によるメディアソース143へセグメントデータの置き換えや追加する処理、

これらの一連の処理が遅延を発生させる大きな要因であると考えられる。

[0117] (a)の処理は、前述したように、放送受信セグメント125は、HTTPサーバ133に一旦、格納され、その後、アプリケーション142が、HTTPサーバ133に格納された放送受信セグメント125を、XHR (XMLHttpRequest) 141によってセグメント単位で読み込む処理である。

(b) の処理は、アプリケーション 142 が、放送受信セグメント 125 の一部のセグメントをネットワーク受信セグメント 126 に置き換えてメディアソース 143 を生成する処理である。

これらの処理には、所定の時間を要するため、再生遅延が避けられない状況となっていると考えられる。

さらに、アプリケーション 142 は、番組切り替えやチャンネル切り替え時に切り替えることが必要となる場合が多く、このアプリケーション切り替え処理時間も、さらに遅延を大きくする要因となる。HTTPサーバ 133 の代わりに、WebSocketサーバを利用して、アプリケーション 125 が WebSocket API によってプッシュ型のデータ送受信をすることで遅延を低減する方法も考えられるが、最低でも一つのセグメントを受信して、送信するという処理に要する時間は低減することができない。

[0118] [6. 放送波とネットワーク双方の受信データの合成処理における遅延を解消または減少させた実施例について]

次に、放送波とネットワーク双方の受信データの合成処理における遅延を解消または減少させた実施例について説明する。

[0119] 図 12 は、本開示の受信装置（クライアント）30における合成処理、すなわち放送波の受信データの一部を、ネットワーク受信データに置き換えて出力する処理を説明する図である。

なお、図 12 に示す処理も図 11 と同様、放送波の受信データの一部をネットワーク受信データに置き換えて出力する場合の処理、すなわち、図 8 を参照して説明したデータ置き換え処理を行なう場合の処理例である。

[0120] 受信装置（クライアント）30は、放送サーバ 21からの放送波を、アンテナ 131を介して受信する。

さらに、データ配信サーバ 22からデータ受信を行う。

アンテナ 131を介して受信するデータを放送受信セグメント 125、データ配信サーバ 22からの受信データをネットワーク受信セグメント 126とする。

放送波セグメント125は、例えば映画等の番組コンテンツであり、ネットワーク受信セグメント126は、番組コンテンツの決められたタイミングに表示する広告コンテンツであるとする。

[0121] 図に示す例では、放送受信セグメント125の一部セグメントa, b, c, d, e, fの6つのセグメントを示している。

一方、ネットワーク受信セグメント126として(c), (d)の2つのセグメントを示している。

なお、各セグメントは、各セグメント単位で復号可能な例えばMP EG圧縮されたデータ(MP4データ)である。

データの置き換えはセグメント単位で実行する。

[0122] ネットワーク受信セグメントである2つの広告セグメント(c), (d)を、放送受信セグメント125のセグメントa, b, c, d, e, f中のセグメントc, dと置き換えて出力する。

すなわち、受信装置30における出力コンテンツを以下のセグメント列とする。

a, b, (c), (d), e, f

上記セグメント列中、セグメントa, b, e, fは、放送受信セグメントであり、セグメント(c), (d)は、ネットワーク受信セグメントである。

[0123] 受信装置30は、ブラウザ(HTML5ブラウザ)210を有し、ブラウザ210上でアプリケーション(Web application (JavaScript (登録商標)))212を実行する。

このアプリケーション212が放送受信セグメント125と、ネットワーク受信セグメント126との合成処理を行なう、

[0124] この図12に示す構成では、放送受信セグメント125は、そのままメディアソース213として構成される。

先に図11を参照して説明した構成では、放送受信セグメント125は、受信装置に設定されるプロキシサーバとしてのHTTPサーバ133に一旦

、格納され、その後、アプリケーションによるサーバからのセグメント単位の読み出しが実行されていた。

この図12に示す構成では、放送受信セグメント125に対するHTTPサーバに対する格納処理や読み出し処理が実行されず、放送受信セグメント125は、そのままメディアソース213として構成される。

この時点で、メディアソース213は、図に示すセグメント列(1)のように、放送受信セグメント、a, b, c, d, e, fによって構成される。つまり、放送受信セグメントはアプリケーション212が実行されているか否かに関わらず、画像デコーダ、音声デコーダに送られ、再生されている。

[0125] その後、アプリケーション212は放送波で伝送されたシグナリングにより放送の番組に連動して起動され、ネットワーク受信セグメント126であるセグメント(c), (d)のみを取得し、このネットワーク受信セグメント(c), (d)を、放送受信セグメント、a, b, c, d, e, fのみによって構成されたメディアソース213の一部のセグメントc, dと置き換える処理を実行する。

[0126] すなわち、アプリケーション212は、放送受信セグメント、a, b, c, d, e, fのみによって構成されたメディアソース(1)更新処理を実行して、更新されたメディアソース(2)、すなわち、

a, b, (c), (d), e, f

上記セグメント列からなるメディアソース(2)となるように制御する。

。

[0127] このように、アプリケーション212は、放送受信セグメント列のみからなるメディアソースの構成セグメントの一部をネットワーク受信セグメントに置き換えるメディアソース更新処理の制御を実行する。

この具体的処理については、後段で説明する。

[0128] その後、

a, b, (c), (d), e, f

上記セグメント列を有するメディアソース(2)213は、データの種類

に応じて振り分けられ、画像対応ソースバッファ214と、音声対応ソースバッファ215が生成される。

[0129] 画像対応ソースバッファ214は、画像デコーダ221によってデコードされ、画像表示部222に表示される。

音声対応ソースバッファ215は、音声デコーダ223によってデコードされ、音声出力部224に出力される。

この結果として再生されるセグメントが図に示す再生セグメント225であり、放送受信セグメント125の一部のセグメントがネットワーク受信セグメント126に置き換えて再生される。

例えば図8に示す時間 $t_0 \sim t_3$ の一連のデータ表示が実行される。

[0130] 図12を参照して説明したアプリケーション212によって実行される放送受信セグメント125とネットワーク受信セグメント126との合成処理は、先に図10を参照して説明したMSE (Media Source Extensions) -APIをベースとして改良した新API (Application Programming Interface) を利用して実行される。

この新APIの処理アルゴリズムは例えば、以下のアルゴリズムとなる。

```
<script>
```

```
var video=document.getElementById('v');  
var tuner=navigator.tv.currentTuner();  
var mediaSource=.tuner.getMediaSource();  
mediaSource.addEventListener('sourceopen', onSourceOpen.bind(this, video));  
video.src=window.URL.createObject
```

```
URL (mediaSource) ;  
    var videoSourceBuffer=mediaSource  
.sourceBuffers [0] ;  
    var audioSourceBuffer=mediaSource  
.sourceBuffers [1] ;  
</script>
```

[0131] 上記の新APIの処理の概要は以下の通りである。

(S1) 通信部(チューナー)202受信セグメント格納用のメディアソース213を取得し、アプリケーション142による参照を可能な設定とする。

(S2) メディアソース213対応のビデオソースバッファ214と、オーディオソースバッファ215を取得し、アプリケーション142による参照を可能な設定とする。メディアソース213は、HTML5のビデオオブジェクトに設定する。

[0132] アプリケーション142は、これらのAPI処理によって参照可能となったメディアソース212、ビデオソースバッファ214、オーディオソースバッファ215を参照して、これらのオブジェクトに格納されている通信部(チューナー)202受信セグメントに対して、ネットワーク受信セグメントによるセグメント置き換えの制御を実行する。

[0133] このように、アプリケーション142は、新APIの処理によって参照可能となった放送受信セグメント用のメディアソースと、ソースバッファに対する放送受信セグメントのデータ格納状況を確認して、置き換え対象の放送受信セグメントがバッファに格納されていることを確認すると、広告コンテンツの再生コンテンツを挿入する時間位置の放送受信セグメントをネットワーク受信セグメントに置き換える処理を実行する。

[0134] 図12を参照して説明した処理例では、放送受信メディアセグメントはアプリケーションで処理されないため、先に図11を参照して説明した再生遅延の要因が解消される。

すなわち、以下の処理の処理時間が不要となるため、処理遅延が解消される。

(a) 放送受信セグメントのHTTPサーバに対する格納と、アプリケーションによるHTTPサーバからのセグメント単位の読み込み処理

(b) アプリケーションによるメディアソースへの放送受信セグメントの置き換えや追加処理

図12を参照して説明した処理例では、これらの一連の遅延発生要因となる処理が不要となり、この結果再生遅延を解消することができる。

[0135] 図12に示す処理構成では、放送受信セグメント125は、そのままメディアソース213として設定されるので、放送受信セグメント125のHTTPサーバへの格納処理や、アプリケーションによる読み込み処理が不要となる。また、アプリケーションによるメディアソースの生成処理も不要となる。

また、アプリケーションは、番組切り替えやチャンネル切り替え時に切り替えることが必要となるが、放送受信セグメントによって構成されるメディアソースは、アプリ切り替えと無関係に生成されるのでアプリケーション切り替え処理時間によって、メディアソース生成の遅延や、アプリケーションの切り替え時に放送セグメントの再生が中断される問題は発生しない。

[0136] 図12に示すアプリケーション212の実行するメディアソース更新処理の詳細について、図13を参照して明する。

図13には、以下の3つのデータを時系列（左から右に時間が経過）に示している。

(A) アプリケーション受信セグメント（ネットワーク受信セグメント）

(B) チューナー受信セグメント（放送受信セグメント）

(C) デコーダ入力セグメント（出力セグメント）

[0137] 時間軸として、以下の2つの時間軸を示している。

下段に放送時間軸（WallClock（UTC））

上段にHTML5ビデオオブジェクト時間軸（CurrentTime）

[0138] 放送時間軸 (Wall Clock (UTC)) は、放送サーバ 21 が、放送波を介して送信するデータの時間管理に適用する時間を示す時間軸 (受信映像音声再生時間軸) である。

ここでは、いわゆる実時間情報に相当するウォールクロック (Wall Clock (UTC)) を適用した例を示す。

放送波を介して受信する放送受信セグメントには、放送時間軸 (Wall Clock (UTC)) に従ったタイムスタンプが設定されている。

[0139] 一方、HTML 5 ビデオ時間軸 (Current Time) は、アプリケーション 212 の処理時間や、アプリケーション 212 による生成データの管理用の時間を規定したアプリケーション時間軸 (アプリケーションメディア再生時間軸) である。

アプリケーション 212 は、HTML 5 ビデオ時間軸 (Current Time) に従って処理を実行する。

図 13 において、HTML 5 ビデオ時間軸 (Current Time) 上の各時間 $t_3h \sim t_5h$ は、HTML 5 ビデオ時間軸 (Current Time) 上の時間であることを示すため (h) を付記している。

[0140] 以下、各時間軸に示す $t_1 \sim t_5$ 、 $t_3h \sim t_5h$ における処理について、順次、説明する。

(時間 t_1)

まず、時間 t_1 において、受信装置は、チューナー選局を行う。例えばユーザの選択した特定のチャンネルの番組の受信を開始する。

チューナーによる番組受信の開始に併せて、

(B) チューナー受信セグメント (放送受信セグメント) が、放送サーバ 21 から送信され、受信装置 30 において順次、受信される。

[0141] この時点では、アプリケーション 212 によるセグメントの合成処理は開始されておらず、

(B) チューナー受信セグメント (放送受信セグメント)

は、そのまま、

(C) データコーダ入力セグメント (出力セグメント)

として設定され、チューナー受信セグメント (放送受信セグメント) がデコードされて出力 (表示、音声出力) される。

[0142] (時間 t_2)

時間 t_2 において、チューナー受信番組に対応して設定されたアプリケーション (HTML5 アプリ) 212 が起動する。

アプリケーション 212 の起動は、例えばチューナー受信番組、またはそのメタデータに含まれるトリガー情報に基づいて実行される。

ユーザによるアプリケーション起動指示に基づいて起動する設定も可能である。

[0143] (時間 t_3 ($= t_3 h$))

時間 t_3 は、時間 t_2 におけるアプリケーション起動後、アプリケーション 212 による処理が開始可能となった時間である。

アプリケーション 212 は、この処理可能開始時間を、HTML5 ビデオ時間軸の起点 ($current\ Time = 0$) に設定され、HTML5 ビデオ時間軸の $Current\ Time$ はチューナー受信番組の再生とともに更新される。。

アプリケーションの処理タイミングは、HTML5 ビデオ時間 ($current\ Time$) に基づいて決定される。

[0144] 時間 t_3 ($t_3 h = Current\ Time = 0$) において、アプリケーション 212 は、処理対象オブジェクトとしてメディアソースやソースバッファを確保する。

これは、先に図 12 を参照して説明した新 API の処理によって行われる。前述したように、新 API の処理は以下の通りである。

(S1) 通信部 (チューナー) 202 受信セグメント格納用のメディアソース 213 を取得し、アプリケーション 142 による参照を可能な設定とする。

(S2) メディアソース 213 対応のビデオソースバッファ 214 と、オ

オーディオソースバッファ 215 を取得し、アプリケーション 142 による参照を可能な設定とする。

これらの A P I 処理によって、アプリケーション 212 は、メディアソースやソースバッファを参照して、バッファ状況の確認に基づく処理が可能となる。

[0145] なお、図 12 に示すメディアソース 213、ソースバッファ 214、215 は、アプリケーション 212 による処理対象となるオブジェクトであり、具体的には、受信装置 30 の記憶部（バッファ）に格納される放送受信セグメント列に相当する。

ビデオソースバッファ 214 は、受信装置 30 の記憶部（バッファ）に格納されるビデオデータのみからなるセグメント列に相当する。

オーディオソースバッファ 215 は、受信装置 30 の記憶部（バッファ）に格納されるオーディオデータのみからなるセグメント列に相当する。

メディアソース 213 は、受信装置 30 の記憶部（バッファ）に格納されるビデオデータとオーディオデータを含む放送受信セグメント列に相当する。

[0146] この時間 t_3 ($= t_3 h$) 以降、アプリケーション 212 は、メディアソース 213、ソースバッファ 214、215（記憶部内に格納される放送受信セグメント列）を参照し、様々な処理を行なうことが可能となる。

[0147] アプリケーションは、まず、時間 t_3 ($= t_3 h$) 以降、新たにメディアソース 213 に設定されたチューナー受信セグメント、すなわち、受信装置 30 の記憶部（バッファ）に新たに格納された放送受信セグメントに設定されたタイムスタンプを取得する。

図に示す例では、セグメント [Seg (tuner1)] が、時間 t_3 ($= t_3 h$) 以降に最初にメディアソース 213 に入力したチューナー受信セグメントである。

アプリケーション 213 は、このセグメント [Seg (tuner1)] に設定されたタイムスタンプを取得する。

[0148] 放送受信セグメントには、放送時間系のタイムスタンプ、すなわちワールドクロック (WallClock (UTC)) に従ったタイムスタンプが設定されている。

アプリケーションは、このタイムスタンプ (WallClock (UTC)) を用いて、放送時間軸 (WallClock (UTC)) と、HTML 5ビデオ時間軸 (currentTime) との差分 (時間オフセット) である放送タイムオフセット (broadcastTimeOffset) を算出する。

すなわち、時間 t_3 ($= t_3 h$) 以降に最初にメディアソース 213 に入力したチューナー受信セグメント [Seg (tuner1)] に設定されたタイムスタンプの示す時間を、放送タイムオフセット (broadcastTimeOffset) とする。

[0149] 具体例として、セグメント [Seg (tuner1)] に設定されたタイムスタンプが、

UTC 2014年10月3日午前7時3分40秒

であったとする。

この場合、アプリケーションは、放送タイムオフセット (broadcastTimeOffset) を上記時間に設定する。すなわち、

$broadcastTimeOffset = \text{UTC 2014年10月3日午前7時3分40秒}$

とする。

[0150] この時間オフセットは、放送時間軸 (WallClock (UTC)) と、HTML 5ビデオ時間軸 (currentTime) との差分 (時間オフセット) に相当する。

すなわち、時間 t_3 ($= t_3 h$) において、

HTML 5ビデオ時間 : $currentTime = 0$ 、

放送時間 : $WallClock (UTC) = 014年10月3日午前7時3分40秒$

であり、

時間オフセット： $\text{broadcastTimeOffset} = \text{UTC}2014\text{年}10\text{月}3\text{日}午前7時3分40秒$ は、放送時間軸（WallClock（UTC））と、HTML5ビデオ時間軸（currentTime）との差分（時間オフセット）に相当する。

[0151] なお、上述した時間オフセット算出処理例は、アプリケーション212の処理対象オブジェクトであるメディアソース、すなわちメディアソースとしての記憶部に格納された放送受信セグメントのタイムスタンプを用いて算出する処理例であり、前述した新API（MSE-API）によってメディアソースから取得するデータをそのまま利用することができる。

すなわち、APIによる時間オフセット情報の取得処理として実行することが可能である。

[0152] また、時間オフセット情報の取得処理は、その他のオブジェクトから取得することも可能である。例えば、メディアソース以外のアプリケーションの処理オブジェクトとして設定可能な通信部（チューナー）202や、ビデオデータ等のオブジェクトを介したオフセット情報の取得または算出を行う構成としてもよい。

[0153] （時間 $t4h$ ）

その後、アプリケーションは、時間 $t4h$ において、ネットワークを介した合成用セグメントの受信を開始する。すなわち、放送受信セグメントに置き換えて出力するための例えば広告コンテンツ等からなるネットワーク受信セグメントの受信処理を開始する。

[0154] なお、ネットワークを介した合成用セグメントの受信開始時間 $t4h$ は、ネットワーク受信セグメント（広告：Ad）の出力開始時間 $t5$ （BroadcastAdStartTime）（ $=t5h = \text{VideoAdStartTime}$ ）から予め規定した時間 [マージン（Margin）] 前の時間とする。

マージンは、例えば、インターネットを介したデータ取得に必要な時間等

によって決定することができる。

[0155] アプリケーション212は、ネットワークを介した合成用セグメントの受信開始時間 t_4h を、以下の式によって算出する。

$$\begin{aligned} t_4h &= \text{VideoAdStartTimeBefore} \\ &= t_5h - \text{Margin} \\ &= \text{VideoAdStarttime} - \text{Margin} \\ &= t_5 - \text{broadcastTimeOffset} \\ &= \text{BroadcastAdStartTime} - \text{broadcastTimeOffset} \end{aligned}$$

[0156] なお、上記式において、

$$\begin{aligned} t_4h &: \text{VideoAdStartTimeBefore} \\ t_5h &: \text{VideoAdStarttime} \\ \text{Margin} & \end{aligned}$$

これらの時間は、HTML5ビデオ時間軸 (currentTime) に従った時間情報である。

$$t_5 : \text{BroadcastAdStartTime}$$

この時間は、放送時間軸 (WallClock (UTC)) に従った時間情報である。

[0157] なお、アプリケーション212は、(広告: Ad) の出力開始時間 t_5 ($\text{BroadcastAdStartTime}$) について、コンテンツ (番組) の属性情報 (メタデータ) から取得することが可能である。

[0158] また、

$$\text{broadcastTimeOffset}$$

は、時間 t_3 ($= t_3h$) において算出した時間オフセットであり、

$$\text{broadcastTimeOffset} = \text{UTC} 2014 \text{年} 10 \text{月} 3 \text{日} \\ \text{午前} 7 \text{時} 3 \text{分} 40 \text{秒}$$

である。

アプリケーション212は、ネットワークを介した合成用セグメントの受

信開始時間 t_4 h を、これらの各データを用いて算出する。

[0159] この時間 t_4 h 以降、アプリケーション 212 は、ネットワークを介して、ネットワーク受信セグメント（広告：Ad）を順次、受信する。

図 13 に示すセグメント Sa1 ~ Sa5 である。

アプリケーションは、これらのネットワーク受信セグメント Sa1 ~ Sa5 を、チューナー受信セグメント S1 ~ S5 の位置に設定する処理を実行する。

すなわち、チューナー受信セグメント S1 ~ S5 をネットワーク受信セグメント Sa1 ~ Sa5 に置き換える処理を行なう。

[0160] アプリケーション 212 は、前述した新 API（新 MSE-API）の処理によって参照可能とされたメディアソース 213、ソースバッファ 214、215 に格納された放送受信セグメントの格納状況を確認してセグメント置き換え処理を実行する。

[0161] このセグメント置き換え処理は、置き換え後のネットワーク受信セグメントの出力に間に合うように実行すればよい。

例えば、ネットワーク受信セグメント Sa1 は、チューナー受信セグメント S1 の位置に置き換えて設定され、チューナー受信セグメント S1 の出力時間 (t_5) に出力される。

従って、ネットワーク受信セグメント Sa1 の置換処理は、広告開始時間 t_5 (= Broadcast Ad Start Time = 2014 年 10 月 3 日 午前 7 時 16 分 20 秒) までに完了しておくことが必要となる。

[0162] 図 13 に示す例では、時間 t_5 h (= t_5) が、広告開始時間である。

この時間は、ソースバッファのセグメント (S1) が出力される時間であり、この時間 (t_5) のタイムスタンプの設定されたセグメント (S1) を置換対象セグメント列の開始位置として、ネットワーク受信セグメント Sa1 から順次、置き換えればよい。

[0163] アプリケーションは、この処理のために、放送時間系であるウォールクロック (Wall Clock (UTC)) に従った広告開始時間 t_5 (= Br

`broadcastAdStartTime` = 2014年10月3日午前7時16分20秒) を、アプリケーションの時間軸 [HTML5ビデオ時間軸 (`currentTime`)] における時間 `t5h` (`VideoAdStartTime`) に変換する。

変換処理は以下の式に従って実行される。

$$\begin{aligned}
 [0164] \quad t5h(\text{VideoAdStartTime}) & \\
 &= t5 - (\text{broadcastTimeOffset}) \\
 &= (\text{BroadcastAdStartTime}) - (\text{broadcastTimeOffset}) \\
 &= (2014年10月3日午前7時16分20秒) - (2014年10月3日午前7時3分40秒) \\
 &= 12分40秒
 \end{aligned}$$

[0165] なお、上記式において、

`t5h`: `VideoAdStarttime`

この時間は、HTML5ビデオ時間軸 (`currentTime`) に従った時間情報である。

`t5`: `BroadcastAdStartTime`

この時間は、放送時間軸 (`WallClock (UTC)`) に従った時間情報である。

[0166] なお、アプリケーション212は、(広告: `Ad`) の出力開始時間 `t5` (`BroadcastAdStartTime`) について、コンテンツ (番組) の属性情報 (メタデータ) から取得することが可能である。

また、記憶部 (バッファ) に格納された放送受信セグメント `S1` のタイムスタンプから取得することも可能である。

セグメント `S1` には出力時間を示すタイムスタンプとして、放送時間系であるウォールクロック (`WallClock (UTC)`) に従った広告開始時間 `t5` (`=BroadcastAdStartTime=2014年10月3日午前7時16分20秒`) が設定されている。

[0167] また、上記式において、

`broadcastTimeOffset`

は、時間 t_3 ($=t_3h$) において算出した時間オフセットであり、

`broadcastTimeOffset` = UTC 2014年10月3日
午前7時3分40秒

である。

[0168] 上記算出式によって算出された

12分40秒

は、HTML5ビデオ時間軸に従った時間 (`currentTime`)、
すなわちアプリケーション起動時間: $t_3h=0$ からの経過時間に相当する。
。

アプリケーションは、この算出時間、

`currentTime` = 12分30秒までに、受信装置の記憶部 (バッファ) に格納された放送受信セグメント S_1 をネットワーク受信セグメント S_{a1} に置き換える処理を実行する。

[0169] アプリケーションは、HTML5ビデオ時間軸に従った時間 (`currentTime`) に出力対象として設定されたソースバッファの放送受信セグメント、すなわちセグメント S_1 をネットワーク受信セグメント S_{a1} に置き換える処理を実行する。

以降のセグメント $S_2 \sim S_5$ についても、各セグメントに設定された出力時間を示すタイムスタンプの経過前に同様の処理を実行して、ネットワーク受信セグメント $S_{a2} \sim S_{a5}$ に置き換える処理を実行する。

[0170] (時間 t_5h ($=t_5$))

受信装置 30 の記憶部に格納された放送受信セグメントのセグメント $S_1 \sim S_5$ は、アプリケーション 212 の処理によって、順次、ネットワーク受信セグメント $S_{a1} \sim S_{a5}$ に置き換えられ、広告開始時間 t_5 ($=BroadcastAdStartTime = 2014$ 年10月3日午前7時16分20秒) 以降、置き換え後のネットワーク受信セグメント $S_{a1} \sim S_{a5}$

が、順次、出力される。

[0171] 図13に示す(c)デコーダ入力セグメントは、置き換え後のセグメント列を示している。このセグメント列がデコーダに入力され、デコード処理後、表示部等の出力部に出力されることになる。

この処理によって、放送波受信セグメントの一部がネットワーク受信セグメントに置き換えられ、先に図8を参照して説明したようなデータ切り換え表示が可能となる。

すなわち、放送受信データである番組コンテンツを、所定タイミングでネットワーク受信データである広告コンテンツに切り替え、その後、再度、放送受信データである番組コンテンツに切り替えるデータ出力処理が可能となる。

[0172] 時間 t_3 h 以後、アプリケーション 212 は、アプリケーション 212 の参照可能なオブジェクトに設定された図13(B)チューナー受信セグメント、すなわち記憶部(バッファ)に格納された放送受信セグメントを参照して、バッファされた放送受信セグメントの出力時間に相当するタイムスタンプを取得し、それぞれのタイムスタンプに至る前に、放送受信セグメント S1~S5 をネットワーク受信セグメント Sa1~Sa5 に置き換える処理を完了させる。

[0173] 記憶部(バッファ)に格納された放送受信セグメントは、前述の新APIの処理により、アプリケーション 212 による処理対象オブジェクトであるメディアソース 213 や、ソースバッファ 214, 215 として設定され、アプリケーションは、バッファ状況を確認することが可能である。

アプリケーション 212 は、バッファされた放送受信セグメント S1~S5 のタイムスタンプを確認し、それぞれのタイムスタンプ設定時間までに、置換処理を完了させる。

[0174] なお、放送受信セグメント S1~S5 のタイムスタンプは、放送時間軸(Wall Clock(UTC))に従った時間情報である。

アプリケーション 212 は、タイムスタンプ設定時間を、前述の時間(t

3 h) において算出した時間オフセット: `broadcastTimeOffset` = UTC 2014年10月3日午前7時3分40秒を考慮して、HTML5ビデオ時間軸 (`currentTime`) に従った時間情報に変換する。

この変換処理によって、生成した時間に基づいて、各セグメントの置換処理を各セグメントの出力開始時間までに完了させる。

[0175] このように、アプリケーション212は、受信装置30の記憶部（バッファ）に格納された放送受信セグメントの一部をネットワーク受信セグメントによって置き換えるメディアソース更新処理を実行する。

この処理によって、放送波受信セグメントの一部がネットワーク受信セグメントに置き換えられ、先に図8を参照して説明したようなデータ切り換え表示が可能となる。

すなわち、放送受信データである番組コンテンツを、所定タイミングでネットワーク受信データである広告コンテンツに切り替え、その後、再度、放送受信データである番組コンテンツに切り替えるデータ出力処理が可能となる。

[0176] この実施例では、チューナー受信セグメントは、HTTPサーバへの格納、読み出し処理が行われず、直接、アプリケーション212の処理対象オブジェクトとしてのソースバッファとして設定される。

アプリケーション212は、ソースバッファのセグメントの置き換えを伴うソースバッファ更新処理によって、放送波受信セグメントのみからなるソースバッファの一部のセグメントを、ネットワーク受信セグメントに置き換える。

このソースバッファ更新処理は、HTTPサーバに対するデータ格納、読み出し処理に比較して極めて短時間の処理として実行することが可能であり、置き換えられたセグメントの出力を遅延なく実行することが可能となる。

[0177] [7. 受信装置におけるデータ受信および合成処理に適用するハードウェア構成例について]

次に、図 14 を参照して受信装置におけるデータ受信および合成処理に適用するハードウェア構成例について説明する。

[0178] 図 14 には、受信装置 30 において実行する以下の処理、すなわち、

- (a) ネットワークおよび放送波を介したデータ受信処理、
- (b) 受信データに基づくデータ合成処理（セグメント合成処理）を実行して表示部等の出力部に出力するデータの生成処理、

これらの処理を実行する受信装置 30 のハードウェア構成例を示す図である。

[0179] 図 14 に示すように、受信装置 30 は、

放送受信データ処理部 310、ネットワーク受信データ処理部 330、さらに、メタデータ（シグナリング）処理部 321、バッファ管理部 322、合成部 341 を有する。

放送受信データ処理部 310 は、チューナー 311、記憶部（バッファ）312、デコーダ&レンダラー 313 を有する。

また、ネットワーク受信データ処理部 330 は、ネットワーク I/F 331、HT アプリケーション実行部（HTML5 ブラウザ）332、グラフィックシステム（HTML ページ出力部）333 を有する。

[0180] 放送波を介して送信されるデータは、アンテナを介して放送受信データ処理部 310 のチューナー 311 によって受信される。受信データは、記憶部（バッファ）312 に格納された後、デコーダ&レンダラー 313 において、デコード処理および描画処理がなされ、合成部 341 において、ネットワーク受信データ処理部の生成する HTML ページとの合成処理がなされて合成画像の出力が実行される。

[0181] ネットワークを介して送信されるデータは、ネットワーク受信データ処理部 330 のネットワーク I/F 331 によって受信される。受信データは、アプリケーション実行部 332 に渡され、必要に応じて、前述した放送受信セグメントとの合成処理が行われる。

アプリケーション実行部 332 の生成したデータが、グラフィックシス

テム（HTMLページ生成部）333に渡され、生成されたHTMLページが合成部341に出力される。

合成部341において、放送受信データ処理部の生成した出力データとの合成処理がなされて合成画像の出力が実行される。

[0182] ここで、アプリケーション実行部332が、前述した放送受信セグメントとネットワーク受信セグメントとの合成処理を行なう場合の処理例について説明する。

放送受信セグメントは、すべて記憶部（バッファ）312に格納される。

[0183] この記憶部（バッファ）312に格納された放送受信セグメントのバッファ格納情報は、バッファ管理部322を介してアプリケーション実行部332が取得することができる。

記憶部（バッファ）312の格納データは、図12に示すアプリケーション212の処理対象オブジェクトであるメディアソース213、ソースバッファ214、215を構成するデータである。

前述したように、アプリケーション212は、新APIの処理によって、記憶部（バッファ）312の格納データを参照することが可能となる。

図14に示すハードウェア構成では、アプリケーション実行部332は、バッファ管理部322を介してバッファ情報を取得する処理を実行する。

[0184] また、アプリケーション実行部332は、チューナー311を介して受信される各種のメタデータ（シグナリングデータ）をメタデータ（シグナリング）処理部321を介して取得することができる。

このメタデータ（シグナリング）には、例えば、先に図13を参照して説明した広告開始時間（t5）等の情報が含まれる。

[0185] アプリケーション実行部332は、バッファ管理部322を介して取得するバッファ情報、例えば、記憶部（バッファ）312に格納された放送受信セグメントのタイムスタンプ情報等を取得し、さらに、メタデータ（シグナリング）処理部321を介して例えば、先に図13を参照して説明した広告開始時間（t5）等の情報を取得する。

アプリケーション実行部 212 は、これらの情報を適用して、先に図 13 を参照して説明したセグメントの置き換え処理、すなわち、記憶部（バッファ）312 に格納された放送受信セグメントの一部をネットワーク受信セグメントに置き換える処理をバッファ管理部 322 に指示して実行させる。

このハードウェア構成例によれば、放送受信セグメントのメディアソースのバッファリング情報は API としては公開されているが、放送セグメント自体はソフトウェアで実装されたブラウザに伝送されることはなく、放送受信データ処理部で処理されるためハードウェア化できるため、ソフトウェアの負荷を低減できるため、高性能な CPU や大量のメモリを必要しない低コストな実装や低消費電力化の実装が可能である。

[0186] [8. 受信装置の実行する処理のシーケンスについて]

次に、図 15 以下に示すフローチャートを参照して、受信装置 30、および図 12 に示すアプリケーション 212（＝図 14 に示すアプリケーション実行部 332）の実行する処理のシーケンスについて説明する。

[0187] まず、図 15 に示すフローチャートを参照して、受信装置 30 の実行する全体的な処理についてのシーケンスを説明する。

図 15 に示すフローは、例えば受信装置 30 の記憶部に記憶されたプログラムに従って、受信装置のプログラム実行機能を持つ CPU 等を備えたデータ処理部の制御の下に実行される。

以下、図 15 に示すフローの各ステップの処理について、順次、説明する。

[0188] (ステップ S201)

まず、ステップ S201 において、受信装置 30 側のユーザによって受信チャンネルの選択がなされ、選択されたチャンネルのコンテンツ（番組）の受信処理と再生処理が開始される。

このコンテンツは、放送サーバ 21 の送信する放送波から取得される。

なお、受信装置 30 は、コンテンツ（番組）の受信に併せて、コンテンツ対応のアプリケーションや、コンテンツ対応の様々なメタデータを受信する

。

[0189] (ステップS202)

次に、受信装置は、ステップS101において選択され、受信、再生を開始したコンテンツ（番組）に対応して設定されたアプリケーションを起動する。例えば図12に示すアプリケーション212である。

アプリケーションの起動は、例えばチューナー受信番組、またはそのメタデータに含まれるトリガー情報に基づいて実行される。

ユーザによるアプリケーション起動指示に基づいて起動する設定としてもよい。

[0190] (ステップS203)

次に、受信装置30は、起動したアプリケーションによる処理を実行する。具体的には、ネットワークを介して受信した広告コンテンツ等のネットワーク受信データを放送受信データに置き換え、あるいは重畳するデータ合成処理を実行する。

これは、先に図12～図14を参照して説明した処理に相当する。

このステップS203のアプリケーション処理の詳細シーケンスについては、図16～図17に示すフローチャートを参照して後段で説明する。

[0191] (ステップS204～S205)

ステップS204は、チャンネル切り替えの有無判定処理である。

チャンネル切り替えが発生すると、ステップS205に進み、バングミ連動のアプリケーションは終了する。アプリケーション終了後、ステップS201に戻り、切り替え後のチャンネル対応のコンテンツ（番組）の受信、再生処理を開始する。

一方、ステップS204においてチャンネル切り替え無しの判定の場合は、ステップS206に進む。その他、チャンネル切り替えが発生していない場合でも、先に示したように受信番組のメタデータに含まれるトリガー情報に基づいて、別のアプリケーションが起動される場合もある。

[0192] (ステップS206～S207)

ステップS 206は、ユーザによる受信装置30の電源オフ動作がなされたか否かの判定処理である。

電源がオフされた場合、ステップS 207に進み、コンテンツ受信、再生を終了する。併せてアプリケーションの処理も終了する。

一方、電源がオフされていない場合は、ステップS 203におけるアプリケーションの処理を継続して実行する。

[0193] 次に、ステップS 203において実行されるアプリケーション処理の詳細シーケンスについて、図16、図17に示すフローチャートを参照して説明する。

図16のフローのステップS 301から、各ステップの処理の詳細について、順次、説明する。

なお、アプリケーションは、図12を参照して説明したアプリケーション212であり、具体的には、例えばブラウザ上で動作するHTML5アプリケーションである。

[0194] (ステップS 301)

まず、アプリケーションは、ステップS 301において、アプリケーションの処理対象（オブジェクト）としてHTML5ビデオオブジェクトを設定する。

これは、放送受信データとネットワーク受信データとの合成処理を可能とするための準備処理であり、アプリケーションの生成する合成データ用の処理オブジェクトとしてHTML5ビデオデータを設定する処理である。

[0195] (ステップS 302)

次に、アプリケーションは、ステップS 302において、アプリケーションの処理対象（オブジェクト）としてチューナーを設定する。図12に示す放送受信セラグメント125を受信する通信部（チューナー）202である。

これも、放送受信データとネットワーク受信データとの合成処理を可能とするための準備処理であり、放送受信データを受信する通信部（チューナー

) 202を処理対象(オブジェクト)に設定する処理である。Current Tunerとあるのは、受信機が複数チューナーを搭載していた場合、テレビに表示されているチューナーのオブジェクトを取得するという意味である。このチューナーオブジェクトはアプリケーションがチャンネル切り替えを指示するAPIとしても利用される。

なお、この処理は、前述した新APIを適用した処理として実行される。

前述した新API(MSE-API)の実行する処理アルゴリズム中の以下の処理に相当する。

```
tuner=navigator.tv.currentTuner()  
;
```

[0196] (ステップS303)

次に、アプリケーションは、ステップS303において、ステップS320において処理対象オブジェクトに設定したチューナーオブジェクトから、チューナーメディアソースオブジェクトを取得する。

この処理は、図に示すメディアソース213の取得処理に相当する。

図12に示す通信部(チューナー)202は、放送サーバ21の送信する放送受信セグメント125を受信し、受信セグメントをメディアソースとして記憶部(バッファ)に格納された設定となる。ただし、ハードウェア構成として図14に示す様に、チューナーからの受信セグメントはアプリケーションの実行部に関わらず、記憶部(バッファ)312に放送受信セグメント125を格納し、デコーダ&レンダラ313を経て再生されている事になる。

[0197] アプリケーションは、この記憶部(バッファ)312に格納された放送受信セグメントのバッファリング情報を提供するアプリケーションの処理対象オブジェクトであるメディアソース、すなわち、図12に示すメディアソース213として取得する。

なお、この処理も、前述した新APIを適用した処理として実行される。

前述した新APIの実行する処理アルゴリズム中の以下の処理に相当する

。

```
mediaSource = tuner.getSource()  
();
```

[0198] (ステップS304)

次に、アプリケーションは、ステップS304において、ステップS301においてアプリケーションの処理対象オブジェクトに設定したHTML5ビデオオブジェクトに、ステップS303において取得したチューナーメディアソースオブジェクトを設定する。

すなわち、アプリケーションによる合成処理の準備処理として、放送ジュシンセグメントのみから構成されるメディアソースを、HTML5ビデオオブジェクトに設定する処理である。

これは、例えば、図13に示す(B)チューナー受信セグメントのセグメントS1等を含むセグメント列(=更新前メディアソース)をHTML5ビデオオブジェクトに設定する処理である。

なお、この処理も、前述した新APIを適用した処理として実行される。

前述した新APIの実行する処理アルゴリズム中の以下の処理に相当する。ただし、この処理は元のMSE APIで行われる処理と同じである。

```
video.src = window.URL.createObjectURL(mediaSource);
```

[0199] (ステップS305)

次に、アプリケーションは、ステップS305において、ステップS303において取得したチューナーメディアソースオブジェクトから、放送時間オフセットを取得する。

この時間オフセット取得処理は、先に図13を参照して説明した放送タイムオフセット(broadcastTimeOffset)の取得処理である。

[0200] チューナーメディアソースオブジェクトに含まれる放送受信セグメントには、放送時間系のタイムスタンプ、すなわちウォールクロック(WallC

lock (UTC)) に従ったタイムスタンプが設定されている。

アプリケーションは、このタイムスタンプ (WallClock (UTC)) を用いて、放送時間軸 (WallClock (UTC)) と、HTML 5ビデオ時間軸 (currentTime) との差分 (時間オフセット) である放送タイムオフセット (broadcastTimeOffset) を決定する。

[0201] 先に、図13を参照して説明したように、例えば、図13に示すセグメント [Seg (tuner1)] に設定されたタイムスタンプが、

UTC 2014年10月3日午前7時3分40秒

であったとする。

この場合、アプリケーションは、放送タイムオフセット (broadcastTimeOffset) を上記時間に設定する。すなわち、

$\text{broadcastTimeOffset} = \text{UTC 2014年10月3日午前7時3分40秒}$

とする。

[0202] この時間オフセットは、放送時間軸 (WallClock (UTC)) と、HTML 5ビデオ時間軸 (currentTime) との差分 (時間オフセット) に相当する。

すなわち、時間 $t_3 (= t_3 h)$ において、

HTML 5ビデオ時間: $\text{currentTime} = 0$ 、

放送時間: $\text{WallClock (UTC)} = 014\text{年}10\text{月}3\text{日午前}7\text{時}3\text{分}40\text{秒}$

であり、

時間オフセット: $\text{broadcastTimeOffset} = \text{UTC 2014年10月3日午前7時3分40秒}$ は、放送時間軸 (WallClock (UTC)) と、HTML 5ビデオ時間軸 (currentTime) との差分 (時間オフセット) に相当する。

[0203] なお、上述した時間オフセット算出処理例は、アプリケーションの処理対

象オブジェクトであるメディアソース、すなわちメディアソースとしての記憶部に格納された放送受信セグメントのタイムスタンプを用いて算出する処理例であり、前述した新API（MSE-API）によってメディアソースから取得するデータをそのまま利用することができる。

すなわち、以下のAPIによる時間オフセット情報の取得処理として実行することが可能である。

```
broadcastTimeOffset = mediaSource.broadcastTimeOffset
```

[0204] また、時間オフセット情報の取得処理は、その他のオブジェクトから取得することも可能である。例えば、以下のようにメディアソース以外のアプリケーションの処理オブジェクトとして設定可能な通信部（チューナー）や、ビデオデータ等のオブジェクトを介したオフセット情報の取得または算出を行う構成としてもよい。

```
broadcastTimeOffset = navigator.tv.currentTuner().broadcastTimeOffset
broadcastTimeOffset = video.broadcastTimeOffset
```

[0205] (ステップS306)

次に、アプリケーションは、ステップS306において、ステップS303で取得したチューナーメディアソースオブジェクトから、ビデオ、オーディオの各ソースバッファを取得する。

ビデオソースバッファは、図12に示すソースバッファ214に相当し、オーディオソースバッファは、図12に示すソースバッファ215に相当する。

これらのソースバッファ214、215は、メディアソース213の一部の構成要素から構成されるオブジェクト、すなわちアプリケーションが処理可能なオブジェクトである。

[0206] これらは、受信装置30の記憶部（バッファ）に格納される放送受信セグ

メント列に相当する。

ビデオソースバッファ214は、図14に示す受信装置30の記憶部（バッファ）312に格納されるビデオデータのみからなるセグメント列に相当する。

オーディオソースバッファ215は、受信装置30の記憶部（バッファ）312に格納されるオーディオデータのみからなるセグメント列に相当する。

メディアソース213は、受信装置30の記憶部（バッファ）312に格納されるビデオデータとオーディオデータを含む放送受信セグメント列に相当する。

[0207] アプリケーションは、ステップS306において、ステップS303で取得したチューナーメディアソースオブジェクトから、ビデオ、オーディオの各ソースバッファを取得する。

なお、この処理も、前述した新APIを適用した処理として実行される。

前述した新APIの実行する処理アルゴリズム中の以下の処理に相当する。元のMSE APIではアプリケーションが新規にバッファを作成していたが、新APIではチューナーメディアソースオブジェクトは放送受信データに含まれるビデオ、オーディオの数やコーデックの種類に合わせてブラウザにより生成される

```
videoSourceBuffer=mediaSource.sourceBuffers[0];
```

```
audioSourceBuffer=mediaSource.sourceBuffers[1];
```

[0208] (ステップS307)

次に、アプリケーションは、ステップS307において、放送波を介して受信するメタデータ（シグナリングデータ）から、ネットワーク受信データ（例えばMP4広告からなるセグメント列）の挿入の開始時間（BroadcastAdStartTime）と、ネット受信セグメントファイルのU

R L リスト、セグメントファイルの数を取得する。

[0209] これは、図 1 4 に示すチューナー 3 1 1 から、メタデータ（シグナリング）処理部 3 2 1 を介してアプリケーション実行部 3 3 2 に至るデータ経路を介して実行される処理である。

放送サーバ 2 1 は、コンテンツ（番組）に併せて、様々なメタデータをシグナリングデータとして受信装置 3 0 に提供しており、受信装置 3 0 は、これらのメタデータをチューナー（図 1 2 の通信部（チューナー） 2 0 2 = 図 1 4 のチューナー 3 1 1））を介して受信する。

アプリケーションは、このメタデータ（シグナリングデータ）から、ネットワーク受信データ（例えば MP 4 広告からなるセグメント列）の挿入の開始時間（Broadcast Ad Start Time）と、ネット受信セグメントファイルの URL リスト、セグメントファイルの数を取得する。これらの情報は、MPEG DASH のマニフェスト・ファイル（MPD : Media Presentation Description）を用いて記述することもでき、したがってアダプティブストリーミングによりコンテンツ再生することもできる。また、サービスワーカーをつかうことによって、広告コンテンツのセグメントを予め取得し、受信機の永続的キャッシュに保存しておくことで、ネットワークの帯域に影響されずに、放送番組と同等の高画質、高音質にて広告を再生することが可能である。

[0210] なお、ネットワークを介して受信し、放送データに合成するコンテンツには様々な種類があるが、例えば広告データの場合、提供地域やユーザに応じて異なる広告とする設定が可能である。いわゆるターゲット広告の設定が可能となる。

このような場合、受信装置やユーザの属性によって提供される広告動画は異なることになり、ネット受信セグメントファイルの URL リスト、セグメントファイルの数等は、受信装置毎に異なる設定となる場合がある。

[0211] （ステップ S 3 0 8 ~ S 3 0 9）

次に、アプリケーションは、ステップ S 3 0 8 において、ステップ S 3 0

1で処理対象として設定し、ステップS304でチューナーメディアソースを設定したHTML5ビデオオブジェクト、すなわち、放送受信セグメントのみによって構成されるHTML5ビデオオブジェクトのカレントタイム (currentTime) を取得する。カレントタイム (currentTime) は、アプリケーションの時間軸 [HTML5ビデオ時間軸 (currentTime)] に従った時間情報である。

[0212] さらに、ステップS309において、ネットワークを介して、ネットワーク受信セグメント、例えば広告コンテンツからなるMP4ファイルを格納したセグメントの受信を開始する。

このステップS308～S309の処理は、先に図13を参照して説明した時間 (t3h～t4h) の処理に相当する。

[0213] すなわち、先に図13を参照して説明したように、アプリケーションは、ネットワークを介した合成用セグメントの受信開始時間t4hを、以下の式によって算出する。

$$\begin{aligned}
 t4h &= \text{VideoAdStartTimeBefore} \\
 &= t5h - \text{Margin} \\
 &= \text{VideoAdStarttime} - \text{Margin} \\
 &= t5 - \text{broadcastTimeOffset} \\
 &= \text{BroadcastAdStartTime} - \text{broadcastTimeOffset}
 \end{aligned}$$

[0214] なお、上記式において、

$$\begin{aligned}
 t4h &: \text{VideoAdStartTimeBefore} \\
 t5h &: \text{VideoAdStarttime} \\
 \text{Margin} &
 \end{aligned}$$

これらの時間は、HTML5ビデオ時間軸 (currentTime) に従った時間情報である。

$$t5 : \text{BroadcastAdStartTime}$$

この時間は、放送時間軸 (WallClock (UTC)) に従った時間

情報である。

[0215] なお、アプリケーションは、(広告: Ad) の出力開始時間 t_5 (Broadcast Ad Start Time) を、ステップ S307 においてコンテンツ (番組) の属性情報 (メタデータ) から取得済みである。

[0216] また、

`broadcastTimeOffset`

は、ステップ S305 において算出した時間オフセットであり、

図 13 を参照して説明した例では、

`broadcastTimeOffset = UTC 2014 年 10 月 3 日
午前 7 時 3 分 40 秒`

である。

アプリケーションは、ネットワークを介した合成用セグメントの受信開始時間 (Video Ad Start Time Before)、すなわち、図 13 に示す時間 t_4h を、これらの各データを用いて算出する。

[0217] (ステップ S310 ~ S311)

アプリケーションは、ステップ S310 において、HTML5 ビデオ時間軸 (`currentTime`) に従った時間 (`currentTime`) が、ネットワークを介した合成用セグメントの受信開始時間 (Video Ad Start Time Before) になったか否かを判定する。

時間になったと判定した場合は、ステップ S311 に進み、ネットワークを介した合成用セグメント、例えば MP4 広告コンテンツを格納したセグメントの受信を開始する。

時間になっていない場合は、ステップ S308 に戻り、ステップ S308 ~ S309 の処理を繰り返す。

[0218] (ステップ S312)

アプリケーションは、ステップ S312 において、ステップ S311 でネットワークを介して受信したネットワーク受信セグメント (例えば MP4 広告) を、ビデオ、オーディオの各ソースバッファの (Video Ad Sta

rtTime) の位置に追加する処理を行なう。

[0219] アプリケーションの処理対象オブジェクトとして設定されたビデオ、オーディオの各ソースバッファに設定されたデータ (セグメント) は、すべてアプリケーションの時間軸 [HTML5ビデオ時間軸 (currentTime)] に従った処理を行なうオブジェクトとして設定される。

[0220] ネットワーク受信セグメントの追加位置を規定する (VideoAdStartTime) は、HTML5ビデオ時間軸 (currentTime)] における時間である。これは、先に図13を参照して説明した時間 (t5h) に相当する。

[0221] 先に図13を参照して説明したように、時間 (t5h = VideoAdStartTime) は、放送時間系であるウォールクロック (WallClock (UTC)) に従った広告開始時間 t5 (= BroadcastAdStartTime) を利用して算出する。

[0222] 算出処理は以下の式に従って実行される。図13を参照して説明した時間情報を適用して説明する。

$$\begin{aligned}
 t5h (\text{VideoAdStartTime}) & \\
 &= t5 - (\text{broadcastTimeOffset}) \\
 &= (\text{BroadcastAdStartTime}) - (\text{broadcastTimeOffset}) \\
 &= (2014年10月3日午前7時16分20秒) - (2014年10月3日午前7時3分40秒) \\
 &= 12分40秒
 \end{aligned}$$

[0223] なお、上記式において、

t5h : VideoAdStarttime

この時間は、HTML5ビデオ時間軸 (currentTime) に従った時間情報である。

t5 : BroadcastAdStartTime

この時間は、放送時間軸 (WallClock (UTC)) に従った時間

情報である。

[0224] なお、アプリケーションは、(広告: Ad) の出力開始時間 t_5 (Broadcast Ad Start Time) をステップ S307 において、コンテンツ (番組) の属性情報 (メタデータ) から取得済みである。

[0225] アプリケーションは、このようにして算出した HTML5 ビデオ時間軸 (current Time) に従った時間情報である広告開始時間 (Video Ad Start Time) の時間位置に、ステップ S311 でネットワークを介して受信したネットワーク受信セグメント (例えば MP4 広告) を追加する処理を行なう。デコード&レンダラー 313 の処理による遅延のために実際は HTML5 ビデオの Current Time の値とチューナーのメディアソースのとのバッファに格納されているセグメントのバッファ範囲を示す Buffered プロパティにはこの遅延分のズレがあるので、より緻密なタイミング制御には Buffered プロパティの値を利用して、セグメントの置換処理を行うのが良い。

[0226] (ステップ S313~S314)

次に、アプリケーションは、ステップ S313 において、すべてのネットワーク受信セグメントの処理を完了したか否かを判定する。

処理が完了していない場合は、ステップ S311 に戻り、未処理セグメントについてステップ S311 以下の処理を実行する。

すべてのセグメントに対する処理が完了したと判定した場合は、ステップ S314 に進み、次のネットワーク受信セグメント列 (例えば次の広告) の処理のために、ステップ S307 に戻る。

[0227] 以上、アプリケーションの実行する処理の詳細について、図 16、図 17 に示すフローチャートを参照して説明した。

これらの説明から理解されるように、アプリケーションは、図 14 に示す受信装置 30 の記憶部 (バッファ) に格納された放送受信セグメントの一部をネットワーク受信セグメントによって置き換えるメディアソース更新処理を実行する。

この処理によって、放送波受信セグメントの一部がネットワーク受信セグメントに置き換えられ、先に図8を参照して説明したようなデータ切り換え表示が可能となる。

すなわち、放送受信データである番組コンテンツを、所定タイミングでネットワーク受信データである広告コンテンツに切り替え、その後、再度、放送受信データである番組コンテンツに切り替えるデータ出力処理が可能となる。

[0228] 本実施例では、チューナー受信セグメントは、先に図11を参照して説明したようなHTTPサーバへの格納、読み出し処理が行われない。

本実施例では、チューナー受信セグメントは、直接、アプリケーションの処理対象オブジェクトであるソースバッファとして設定される。

アプリケーションは、ソースバッファのセグメントの置き換えを伴うソースバッファ更新処理によって、放送波受信セグメントのみからなるソースバッファの一部のセグメントを、ネットワーク受信セグメントに置き換える。

このソースバッファ更新処理は、HTTPサーバに対するデータ格納、読み出し処理に比較して極めて短時間の処理として実行することが可能であり、置き換えられたセグメントの出力を遅延なく実行することが可能となる。

[0229] [9. サービスワーカー (SW) とアプリケーションを利用した処理例について]

上述したアプリケーションは、先に説明したサービスワーカー (SW) の管理アプリケーションとして設定することが可能となる。

サービスワーカー (SW) の管理アプリケーションとして設定することで、アプリケーションは、アプリケーションの提供されたコンテンツ (番組) 終了後も、受信装置の記憶部に格納され、他の番組再生中や、ネットワークが接続されていないオフライン状態であっても任意のタイミングで実行することが可能となる。

[0230] サービスワーカー (SW) は、例えば放送波やネットワークを介して送信されるデータを選択的に取得する処理や、アプリケーションの起動タイミン

グを制御する処理などを実行する機能を有する。

[0231] このサービスワーカー（SW）とアプリケーションを組み合わせることで、例えば、以下のような処理を行なうことが可能となる。

（処理例1）

アプリケーションが、ユーザの視聴履歴を取得し、取得した視聴履歴情報をメタデータとしてサービスワーカー（SW）に通知する。

サービスワーカー（SW）は、メタデータ（視聴履歴情報）に基づいて、放送波によって送信される様々な種類の広告データから、ユーザの興味に適合した広告を選択取得する。

アプリケーションは、サービスワーカー（SW）が選択的に取得した広告コンテンツの表示処理を実行する。

このような処理を行なうことで、ユーザの興味に一致した広告を集中的に提供することが可能となる。

[0232] （処理例2）

サービスワーカー（SW）が、事前に広告データと、その出力時間を記録したメタデータを取得し、受信装置の記憶部に格納しておく。

アプリケーションは、受信装置の記憶部に格納された広告データと、メタデータを取得して、メタデータに記録された出力時間に併せて取得した広告データを出力する処理を実行する。

[0233] サービスワーカー（SW）とアプリケーションを組み合わせることで、例えば、上記のような処理を行なうことが可能となる。

[0234] また、上述した実施例では、図8に示した置換表示により広告を表示する例を示したが、図7のように広告コンテンツを重畳する場合も容易に実現することが可能である。重畳表示する場合は、放送の受信データを格納したメディアソースに加え、重畳するネットワーク受信したデータを再生するメディアソースの2つのメディアソースを利用し、HTML5のビデオオブジェクトも2つ作成し、各々のメディアソースを設定する。

[0235] ネットワーク受信したデータを再生するメディアソースを通常のMSEー

APIで作成する。本実施例で示したとおり、放送データのメディアソースが設定されたHTML5ビデオの時間軸と放送の時間軸を合わせることは可能であるので、ネットワーク受信のデータの再生するHTML5ビデオ時間軸を放送データのHTML5ビデオの時間軸合わせることができれば、放送時間の任意のタイミングにて広告コンテンツを重畳して表示させることが可能である。

[0236] 具体的にはHTML5のビデオオブジェクトにはMediaGroupプロパティーが定義されており、複数のビデオ再生をCurrentTimeを同時に進む様に同期させることが可能である。重畳表示の場合は図16に示した設定のフローは、以下に示すJavaScript（登録商標）に従った処理として行われる。

```
[0237] <script>
var videoTuner=document.getElementById('vTuner');
var tuner=navigator.tv.currentTuner();
var mediaSourceTuner=.tuner.getMediaSource();
mediaSourceTuner.addEventListener('sourceopen',onSourceOpen.bind(this,video));
videoTuner.src=window.URL.createObjectURL(mediaSourceTuner);
var videoSourceBufferTuner=mediaSourceTuner.sourceBuffers[0];
var audioSourceBufferTuner=mediaSourceTuner.sourceBuffers[1];
var videoAd=document.getElementById
```

```
d ('vAd') ;  
    var mediaSourceAd=new MediaSource  
    () ;  
    mediaSourceAd. addEventListener ('so  
urceopen', onSourceOpen. bind (this, v  
ideo)) ;  
    videoAd. src=window. URL. createObj  
ectURL (mediaSourceAd) ;  
    var videoSourceBufferAd=mediaSou  
rceAd. addSourceBuffer ('video/mp4;cod  
ecs="avc1.4d401f"') ;  
    var audioSourceBufferAd=mediaSou  
rceAd. addSourceBuffer ('audio/mp4;cod  
ecs="mp4a.40.2"')  
    videoTuner.mediaGroup = "SYNC";  
    videoAd.mediaGroup = "SYNC";  
</script>
```

[0238] 広告コンテンツのセグメントの追加処理は図17に示したフローは重畳表示でもほぼ、同じである。重畳表示ではアプリケーションは放送データのメディアソースではなく、ネットワークのメディアソース (mediaSourceAd) のソースバッファ (audioSourceBufferAd、videoSourceBufferAd) にネットワークから受信した広告のセグメントデータを広告開始時間の位置に挿入するのが違いである。

[0239] 同じMediaGroupプロパティの値を持つビデオオブジェクトのCurrentTimeは同期して進行し、同期しているいずれかのMediaSourceに再生位置のセグメントデータが格納されていた場合はCurrentTimeが更新され再生が継続するものとする。この場合はMediaSourceAdのソースバッファには広告区間の時間範囲のみセ

グメントデータを追加すれば良い。一方、Media Soucerを設定せずにHTML 5ビデオの同期をとる場合には、お互いのHTML 5ビデオ時間軸の同期時間のオフセット(例、video.mediaGroupTimeOffset)を設定するようにHTML 5ビデオを拡張することで実現できる。

[0240] [10. 送信装置と受信装置の構成例について]

次に、通信装置である送信装置(サーバ)20と、受信装置(クライアント)30の装置構成例について、図18、図19を参照して説明する。

[0241] 図18には、送信装置(サーバ)20と、受信装置(クライアント)30の構成例を示している。

送信装置(サーバ)20は、データ処理部751、通信部752、記憶部753を有する。

受信装置(クライアント)30は、データ処理部771、通信部772、記憶部773、入力部774、出力部775を有する。

データ処理部には通信データ処理部771a、再生処理部771bが含まれる。

[0242] 送信装置(サーバ)20のデータ処理部751は、データ配信サービスを実行するための各種のデータ処理を実行する。例えばデータ配信サービスの構成データの生成や送信制御を行う。さらに、データ処理部751は、受信装置(クライアント)30に提供するアプリケーション、サービスワーカー(SW)、その他の様々なデータや、シグナリングデータの生成、送信処理を行う。

[0243] 通信部752は、AVセグメントの他、アプリケーション、サービスワーカー(SW)、その他の様々なデータ、シグナリングデータ等の配信等の通信処理を行う。

記憶部753は配信対象とするAVセグメント、アプリケーション、サービスワーカー(SW)、アプリケーションによって利用されるデータ、シグナリングデータなどが格納される。

さらに、記憶部 753 は、データ処理部 751 の実行するデータ処理のワークエリアとして利用され、また各種パラメータの記憶領域としても利用される。

[0244] 一方、受信装置（クライアント）30 は、データ処理部 771、通信部 772、記憶部 773、入力部 774、出力部 775 を有する。

通信部 772 は、送信装置（サーバ）20 から配信されるデータ、例えば AV セグメントやアプリケーション、サービスワーカー（SW）、アプリケーションによって利用されるデータ、シグナリングデータ等を受信する。

[0245] データ処理部 771 は、通信データ処理部 771a、再生処理部 771b を有し、例えば先に説明した実施例に従った処理等を実行する。

具体的には、アプリケーションや、API、さらに、サービスワーカー（SW）を利用したデータ処理等を実行する。

[0246] ユーザの指示コマンド、例えばチャンネル選択、アプリケーション起動、インストール等の様々なコマンドは入力部 774 を介して入力される。

再生データは表示部やスピーカ等の出力部 775 に出力される。

記憶部 773 は AV セグメント、サービスワーカー（SW）、アプリケーション、アプリケーションによって利用されるデータ、シグナリングデータなどが格納される。

さらに、記憶部 773 は、データ処理部 771 の実行するデータ処理のワークエリアとして利用され、また各種パラメータの記憶領域としても利用される。

[0247] 図 19 は、送信装置 20、受信装置 30 として適用可能な通信装置のハードウェア構成例を示している。

[0248] CPU (Central Processing Unit) 801 は、ROM (Read Only Memory) 802、または記憶部 808 に記憶されているプログラムに従って各種の処理を実行するデータ処理部として機能する。例えば、上述した実施例において説明したシーケンスに従った処理を実行する。RAM (Random Access Memory)

803には、CPU801が実行するプログラムやデータなどが記憶される。これらのCPU801、ROM802、およびRAM803は、バス804により相互に接続されている。

[0249] CPU801はバス804を介して入出力インタフェース805に接続され、入出力インタフェース805には、各種スイッチ、キーボード、マウス、マイクロホンなどよりなる入力部806、ディスプレイ、スピーカなどよりなる出力部807が接続されている。CPU801は、入力部806から入力される指令に対応して各種の処理を実行し、処理結果を例えば出力部807に出力する。

[0250] 入出力インタフェース805に接続されている記憶部808は、例えばハードディスク等からなり、CPU801が実行するプログラムや各種のデータを記憶する。通信部809は、インターネットやローカルエリアネットワークなどのネットワークを介したデータ通信の送受信部、さらに放送波の送受信部として機能し、外部の装置と通信する。

[0251] 入出力インタフェース805に接続されているドライブ810は、磁気ディスク、光ディスク、光磁気ディスク、あるいはメモリカード等の半導体メモリなどのリムーバブルメディア811を駆動し、データの記録あるいは読み取りを実行する。

[0252] なお、データの符号化あるいは復号は、データ処理部としてのCPU801の処理として実行可能であるが、符号化処理あるいは復号処理を実行するための専用ハードウェアとしてのコーデックを備えた構成としてもよい。

[0253] [11. 本開示の構成のまとめ]

以上、特定の実施例を参照しながら、本開示の実施例について詳解してきた。しかしながら、本開示の要旨を逸脱しない範囲で当業者が実施例の修正や代用を成し得ることは自明である。すなわち、例示という形態で本発明を開示してきたのであり、限定的に解釈されるべきではない。本開示の要旨を判断するためには、特許請求の範囲の欄を参酌すべきである。

[0254] なお、本明細書において開示した技術は、以下のような構成をとることが

できる。

(1) 受信装置の受信データのバッファ処理を、受信装置が実行するアプリケーションのメディア再生の処理オブジェクトであるメディアソースオブジェクトとして開示するデータ処理部を有する受信装置。

[0255] (2) 前記データ処理部は、API (Application Programming Interface) を適用して、前記受信データを前記メディアソースオブジェクトに設定し、アプリケーションが前記バッファの状態の取得やバッファに格納されている受信データの置き換え、または追加の処理を実行する(1)に記載の受信装置。

[0256] (3) 前記データ処理部は、前記アプリケーションによる処理時間を規定するアプリケーションのメディア再生時間軸と、前記受信データにおいて利用される映像音声再生時間軸の時間差に相当する時間オフセットを取得する(1)または(2)に記載の受信装置。

[0257] (4) 前記データ処理部は、前記アプリケーション、または前記APIの適用処理により、前記時間オフセットを取得する(3)に記載の受信装置。

[0258] (5) 前記アプリケーションは、
前記メディアソースオブジェクトに対する処理として、前記データ処理部に格納された受信データと、アプリケーション受信データとの合成処理を実行する(1)～(4)いずれかに記載の受信装置。

[0259] (6) 前記アプリケーションは、
前記アプリケーションによる処理時間を規定するアプリケーションメディア再生時間軸と、前記受信データにおいて利用される映像音声再生時間軸の時間差に相当する時間オフセットを取得し、取得した時間オフセットを利用して、前記受信データに対する前記アプリケーションが受信したデータの挿入時間位置を決定する(5)に記載の受信装置。

[0260] (7) 前記アプリケーションは、
前記時間オフセットを利用して、前記アプリケーション受信データの受信

開始時間を決定する（６）に記載の受信装置。

[0261] （８） 前記アプリケーションは、

前記受信データを構成するセグメントを、前記アプリケーション受信データを構成するセグメントに置き換えるセグメント置換処理を実行する（５）～（７）いずれかに記載の受信装置。

[0262] （９） 受信装置において利用されるアプリケーションを送信する通信部を有し、

前記アプリケーションは、受信装置における受信データのバッファ処理を開示するメディアソースオブジェクトを利用して、受信データとアプリケーションの受信データとの合成処理を実行させるアプリケーションである送信装置。

[0263] （１０） 前記アプリケーションを実行する受信装置は、API（Application Programming Interface）を適用して、前記受信データを前記メディアソースオブジェクトに設定する処理を実行し、

前記アプリケーションは、前記APIによって設定されたメディアソースオブジェクトを利用した処理を実行するアプリケーションである（９）に記載の送信装置。

[0264] （１１） 受信装置において実行するデータ処理方法であり、

前記受信装置のデータ処理部が、受信データを受信装置が実行するアプリケーションのメディア再生の処理オブジェクトであるメディアソースオブジェクトに設定するデータ処理方法。

[0265] （１２） 送信装置において実行するデータ処理方法であり、

送信装置の通信部が、受信装置において利用されるアプリケーションを送信し、

前記アプリケーションは、受信装置における受信データのバッファ処理を開示するメディアソースオブジェクトを利用して、受信データとアプリケーションの受信データとの合成処理を実行させるアプリケーションであるデー

タ処理方法。

[0266] また、明細書中において説明した一連の処理はハードウェア、またはソフトウェア、あるいは両者の複合構成によって実行することが可能である。ソフトウェアによる処理を実行する場合は、処理シーケンスを記録したプログラムを、専用のハードウェアに組み込まれたコンピュータ内のメモリにインストールして実行させるか、あるいは、各種処理が実行可能な汎用コンピュータにプログラムをインストールして実行させることが可能である。例えば、プログラムは記録媒体に予め記録しておくことができる。記録媒体からコンピュータにインストールする他、LAN (Local Area Network)、インターネットといったネットワークを介してプログラムを受信し、内蔵するハードディスク等の記録媒体にインストールすることができる。

また、実施例においては放送からブロードバンド受信した番組コンテンツのセグメントにネットワークから受信した広告コンテンツのセグメントで置き換えることを示したが、受信経路については限定するものではなく、番組コンテンツ、広告コンテンツの双方をネットワーク経由で受信する場合は、もしくは、双方を放送経由で受信する場合でも適応することが可能である。

[0267] なお、明細書に記載された各種の処理は、記載に従って時系列に実行されるのみならず、処理を実行する装置の処理能力あるいは必要に応じて並列的にあるいは個別に実行されてもよい。また、本明細書においてシステムとは、複数の装置の論理的集合構成であり、各構成の装置が同一筐体内にあるものには限らない。

産業上の利用可能性

[0268] 以上、説明したように、本開示の一実施例の構成によれば、放送受信データとネットワーク受信データの合成処理を効率的に実行する装置、方法が実現される。

具体的には、例えば受信装置が通信部を介して受信する放送受信データを、API (Application Programming Inter

face) を適用して、受信装置の実行するアプリケーションの処理オブジェクトであるメディアソースオブジェクトに設定する。アプリケーションは、メディアソースオブジェクトに対する処理として、前記放送受信データと、ネットワークを介して受信するネットワーク受信データとの合成処理を実行する。アプリケーションは、API適用処理によってアプリケーション時間軸と放送時間軸の時間差に相当する時間オフセットを取得し、高精度、低遅延のデータ合成処理を実行する。

本構成により、放送受信データとネットワーク受信データの合成処理および出力処理を低遅延、高精度に行うことが可能となる。これにより放送された番組コンテンツに対してネットワークからの広告コンテンツを挿入することが容易となり、放送サービスとインターネットサービスの融合が進むことが期待される。

符号の説明

- [0269] 10 通信システム
- 20 送信装置
- 21 放送サーバ
- 22 データ配信サーバ
- 30 受信装置
- 31 TV
- 32 PC
- 33 携帯端末
- 50 シグナリングデータ
- 60 AVセグメント
- 70 その他のデータ
- 90 受信装置内表示制御部
- 91 表示処理部
- 92 サービスワーカー (SW)
- 93 キャッシュ

- 95, 96 Webページ
- 101 高ビットレートデータ
- 102 中ビットレートデータ
- 103 低ビットレートデータ
- 104 マニフェスト・ファイル
- 105 再生データ
- 110 メディアソース
- 111~113 ソースバッファ
- 114 トラックバッファ
- 125 放送受信セグメント
- 126 ネットワーク受信セグメント
- 127 再生セグメント
- 131 アンテナ
- 132 通信部 (チューナー)
- 133 HTTPサーバ
- 134 画像デコーダ
- 135 画像表示部
- 136 音声デコーダ
- 137 音声出力部
- 140 ブラウザ
- 141 XHR
- 142 アプリケーション
- 143 メディアソース
- 144, 145 ソースバッファ
- 201 アンテナ
- 202 通信部 (チューナー)
- 210 ブラウザ
- 212 アプリケーション

- 2 1 3 メディアソース
- 2 1 4, 2 1 5 ソースバッファ
- 2 2 1 画像デコーダ
- 2 2 2 画像表示部
- 2 2 3 音声デコーダ
- 2 2 4 音声出力部
- 2 2 5 再生セグメント
- 3 1 0 放送受信データ処理部
- 3 1 1 チューナー
- 3 1 2 記憶部 (バッファ)
- 3 1 3 デコーダ&レンダラー
- 3 2 1 メタデータ (シグナリング) 処理部
- 3 2 2 バッファ管理部
- 3 3 0 ネットワーク受信データ処理部
- 3 3 1 ネットワーク I/F
- 3 3 2 アプリケーション実行部
- 3 3 3 グラフィックシステム
- 3 4 1 合成部
- 7 5 1 データ処理部
- 7 5 2 通信部
- 7 5 3 記憶部
- 7 7 1 データ処理部
- 7 7 2 通信部
- 7 7 3 記憶部
- 7 7 4 入力部
- 7 7 5 出力部
- 8 0 1 CPU
- 8 0 2 ROM

- 803 RAM
- 804 バス
- 805 入出カインタフェース
- 806 入力部
- 807 出力部
- 808 記憶部
- 809 通信部
- 810 ドライブ
- 811 リムーバブルメディア

請求の範囲

- [請求項1] 受信装置の受信データのバッファ処理を、受信装置が実行するアプリケーションのメディア再生の処理オブジェクトであるメディアソースオブジェクトとして開示するデータ処理部を有する受信装置。
- [請求項2] 前記データ処理部は、API (Application Programming Interface) を適用して、前記受信データを前記メディアソースオブジェクトに設定し、アプリケーションが前記バッファの状態の取得やバッファに格納されている受信データの置き換え、または追加の処理を実行する請求項1に記載の受信装置。
- [請求項3] 前記データ処理部は、前記アプリケーションによる処理時間を規定するアプリケーションのメディア再生時間軸と、前記受信データにおいて利用される映像音声再生時間軸の時間差に相当する時間オフセットを取得する請求項1に記載の受信装置。
- [請求項4] 前記データ処理部は、前記アプリケーション、または前記APIの適用処理により、前記時間オフセットを取得する請求項3に記載の受信装置。
- [請求項5] 前記アプリケーションは、
前記メディアソースオブジェクトに対する処理として、前記データ処理部に格納された受信データと、アプリケーション受信データとの合成処理を実行する請求項1に記載の受信装置。
- [請求項6] 前記アプリケーションは、
前記アプリケーションによる処理時間を規定するアプリケーションメディア再生時間軸と、前記受信データにおいて利用される映像音声再生時間軸の時間差に相当する時間オフセットを取得し、取得した時間オフセットを利用して、前記受信データに対する前記アプリケーションが受信したデータの挿入時間位置を決定する請求項5に記載の受信装置。
- [請求項7] 前記アプリケーションは、

前記時間オフセットを利用して、前記アプリケーション受信データの受信開始時間を決定する請求項6に記載の受信装置。

[請求項8]

前記アプリケーションは、

前記受信データを構成するセグメントを、前記アプリケーション受信データを構成するセグメントに置き換えるセグメント置換処理を実行する請求項5に記載の受信装置。

[請求項9]

受信装置において利用されるアプリケーションを送信する通信部を有し、

前記アプリケーションは、受信装置における受信データのバッファ処理を開示するメディアソースオブジェクトを利用して、受信データとアプリケーションの受信データとの合成処理を実行させるアプリケーションである送信装置。

[請求項10]

前記アプリケーションを実行する受信装置は、API (Application Programming Interface) を適用して、前記受信データを前記メディアソースオブジェクトに設定する処理を実行し、

前記アプリケーションは、前記APIによって設定されたメディアソースオブジェクトを利用した処理を実行するアプリケーションである請求項9に記載の送信装置。

[請求項11]

受信装置において実行するデータ処理方法であり、

前記受信装置のデータ処理部が、受信データを受信装置が実行するアプリケーションのメディア再生の処理オブジェクトであるメディアソースオブジェクトに設定するデータ処理方法。

[請求項12]

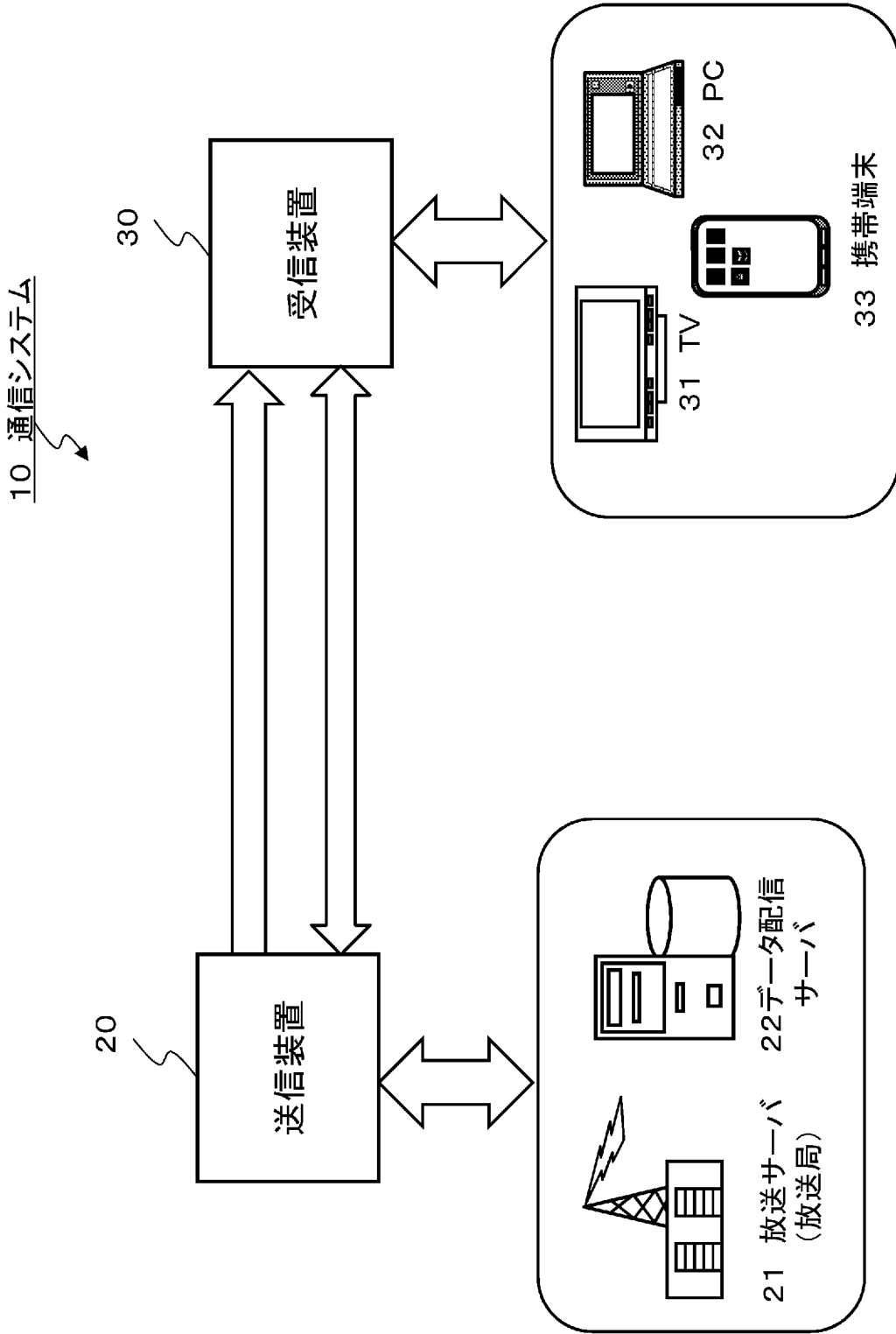
送信装置において実行するデータ処理方法であり、

送信装置の通信部が、受信装置において利用されるアプリケーションを送信し、

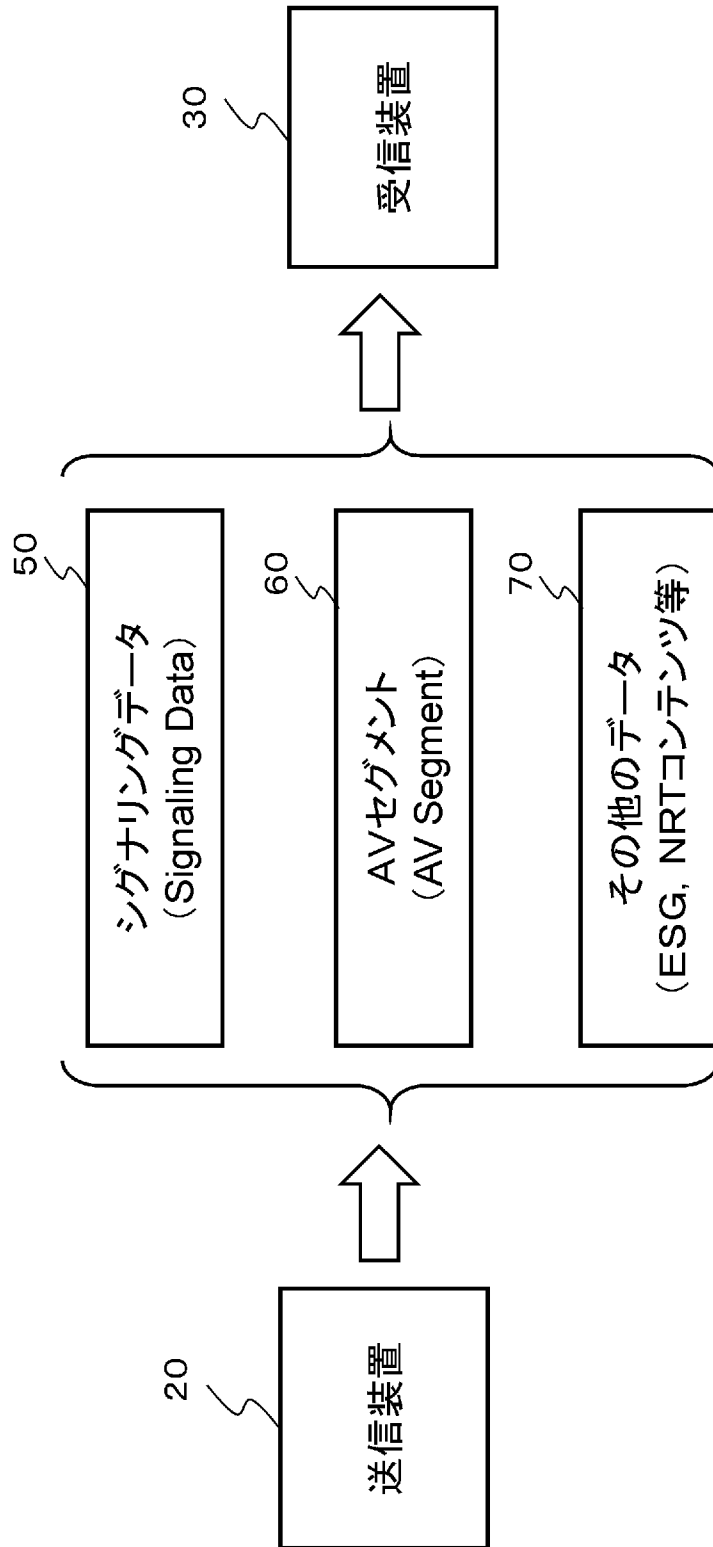
前記アプリケーションは、受信装置における受信データのバッファ処理を開示するメディアソースオブジェクトを利用して、受信データ

とアプリケーションの受信データとの合成処理を実行させるアプリケーションであるデータ処理方法。

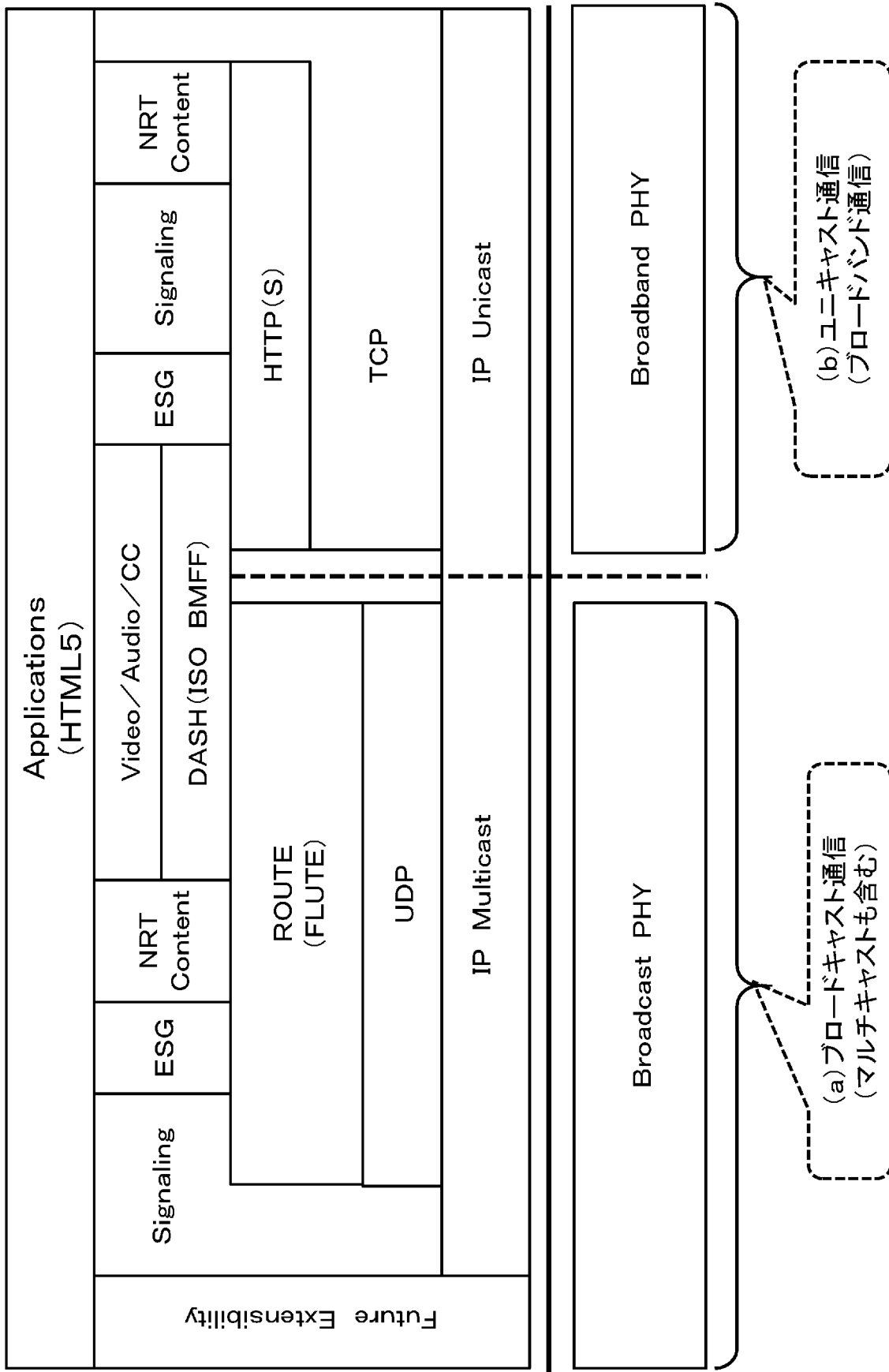
[図1]



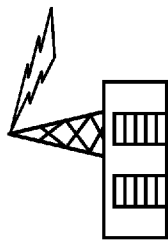
[図2]



[図3]

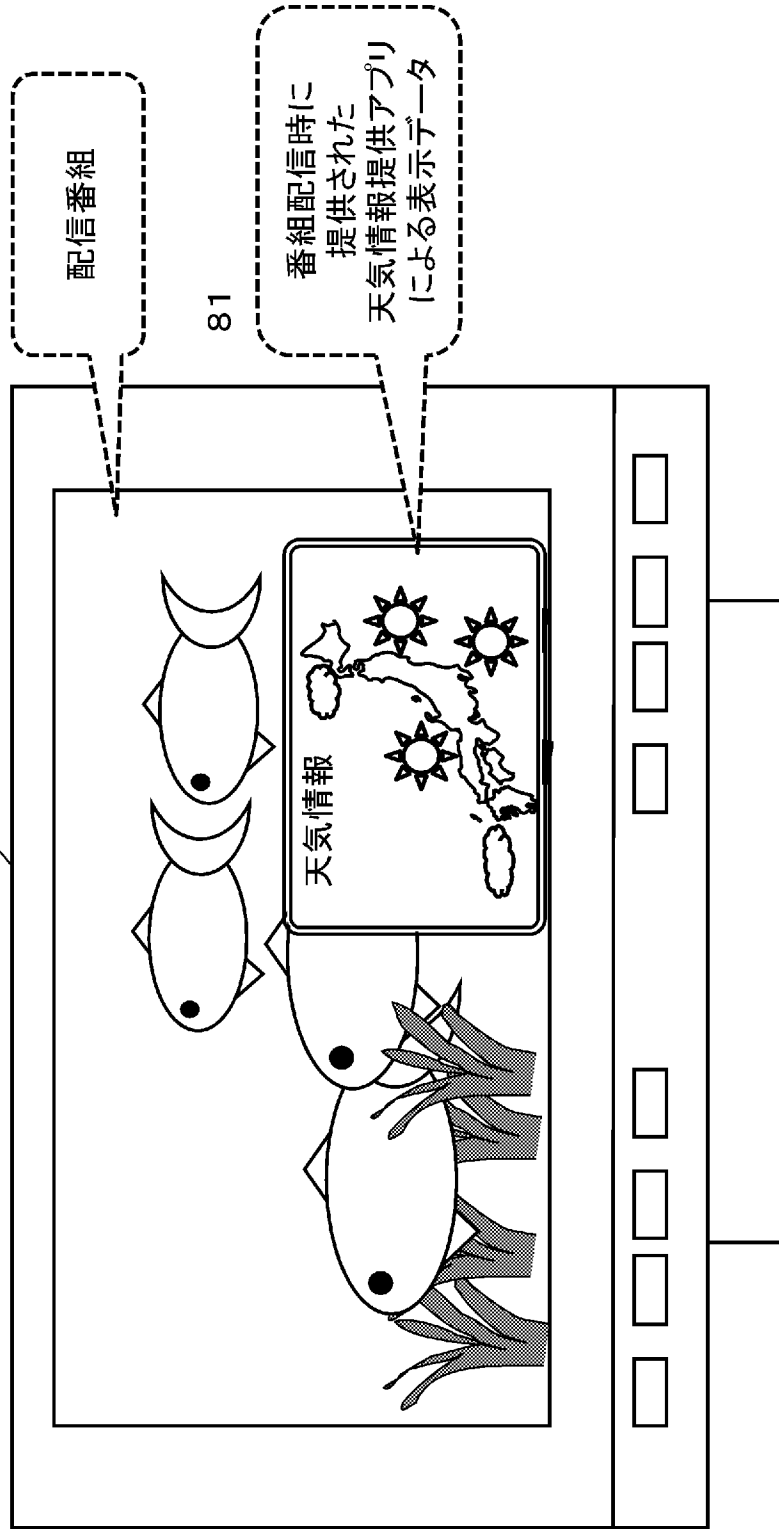


[図4]

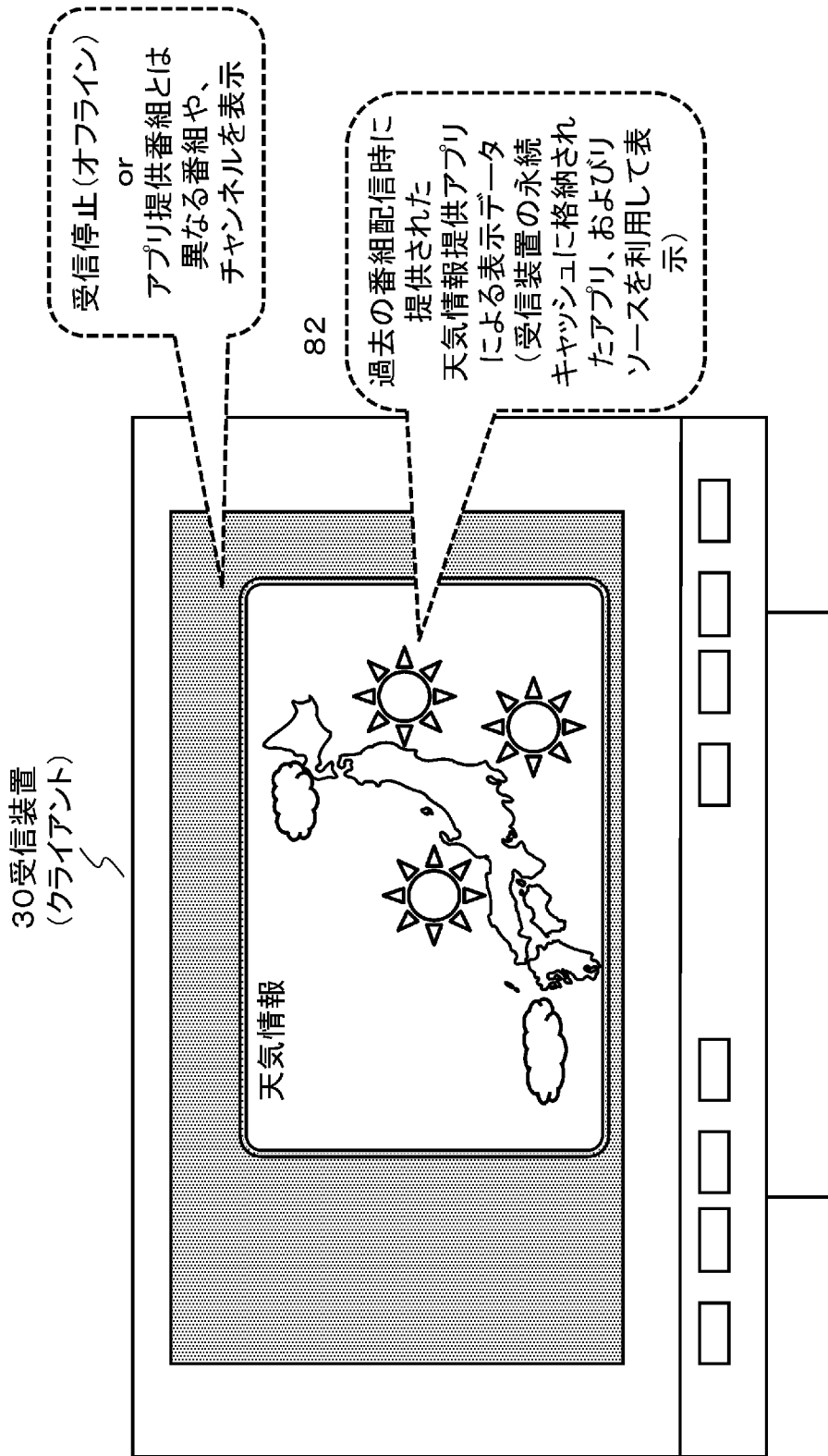


21 放送サーバ
(放送局)

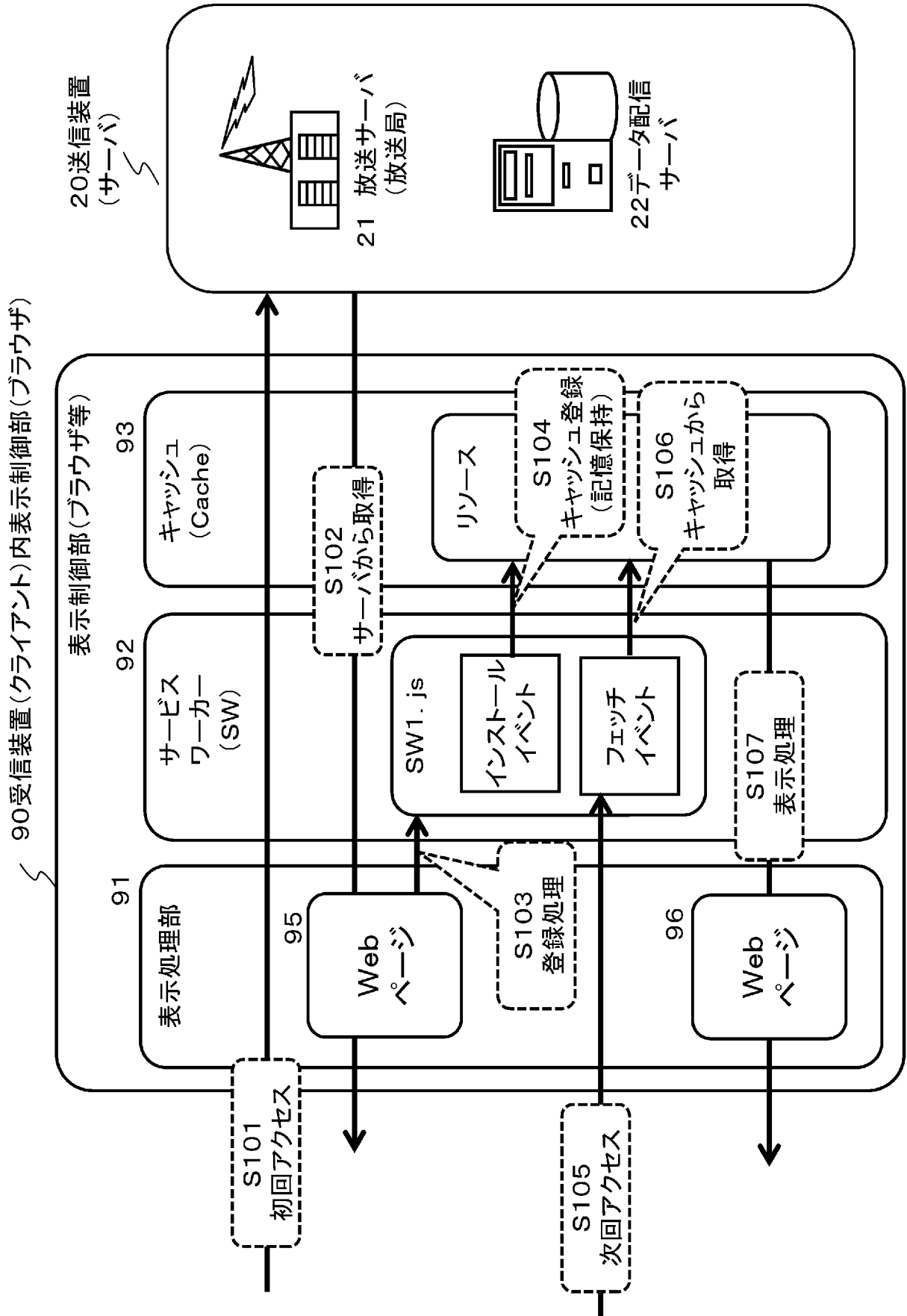
30 受信装置
(クライアント)



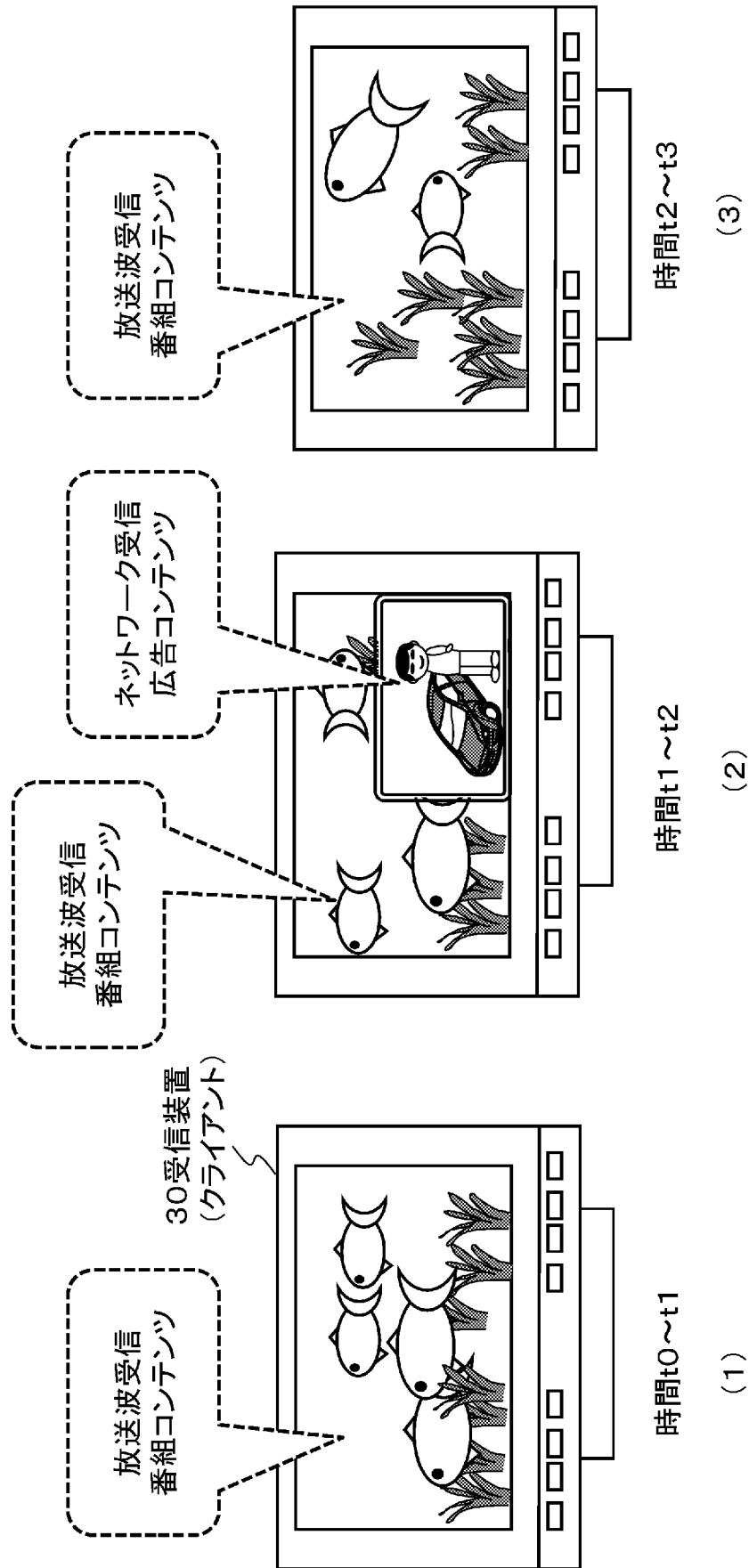
[図5]



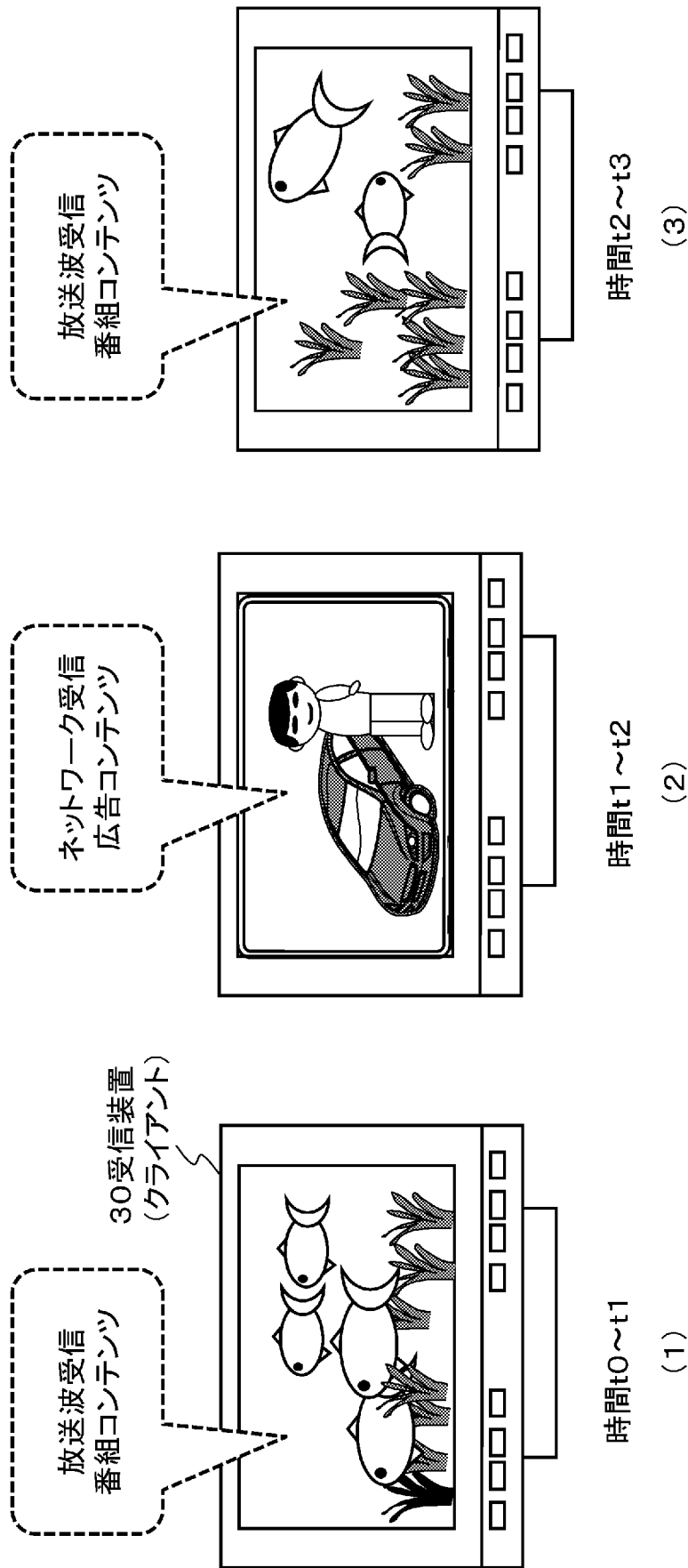
[図6]



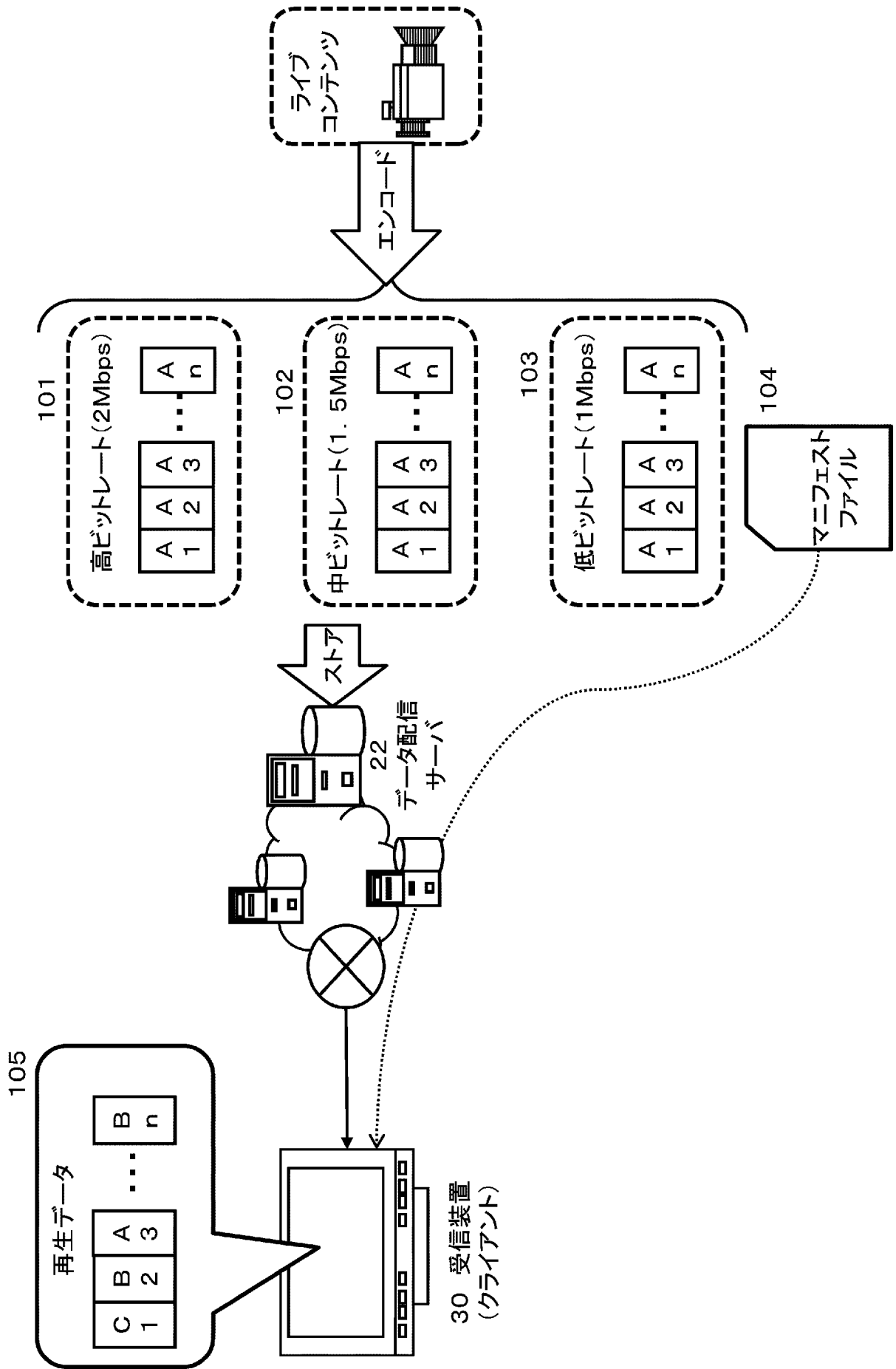
[図7]



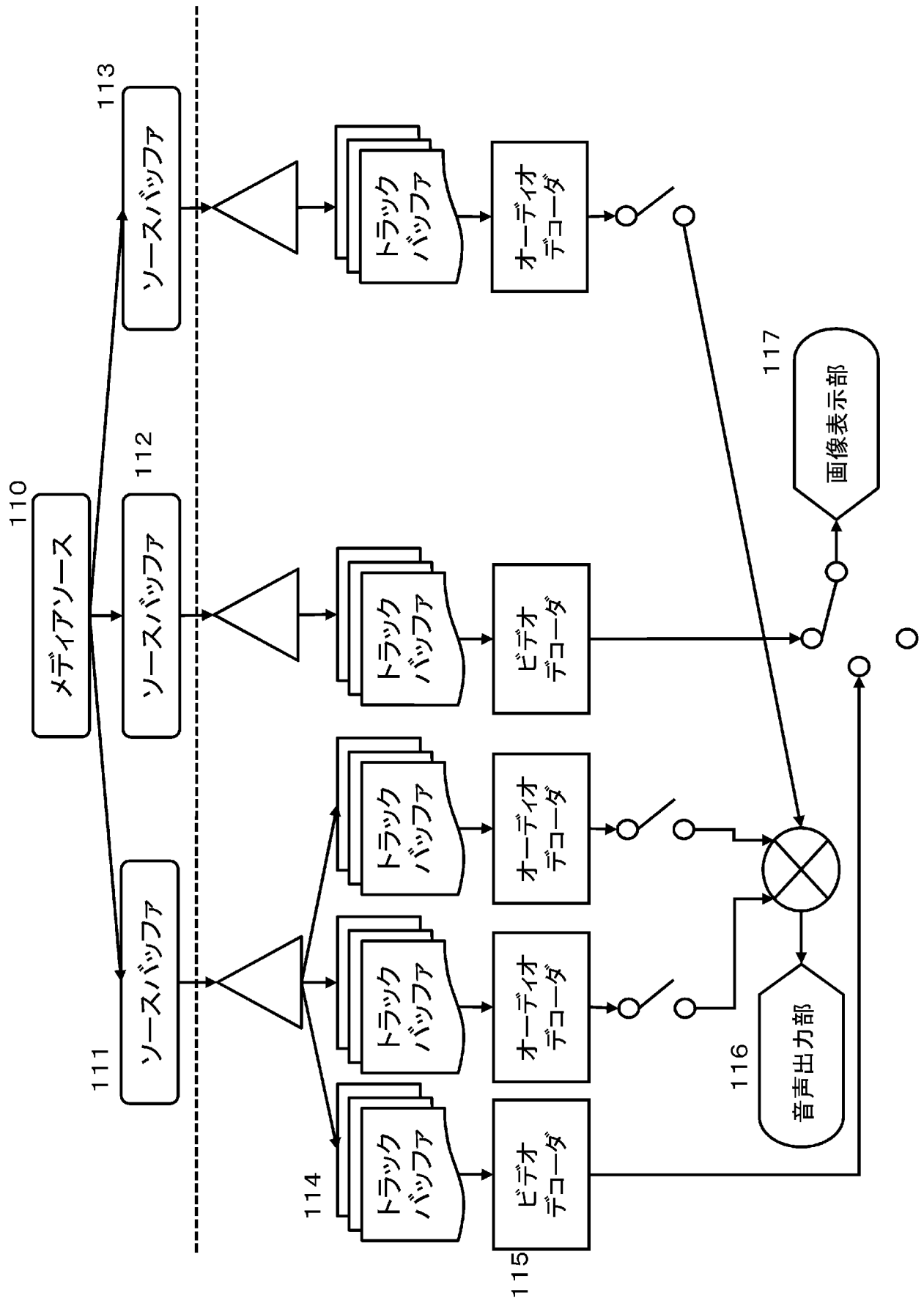
[図8]



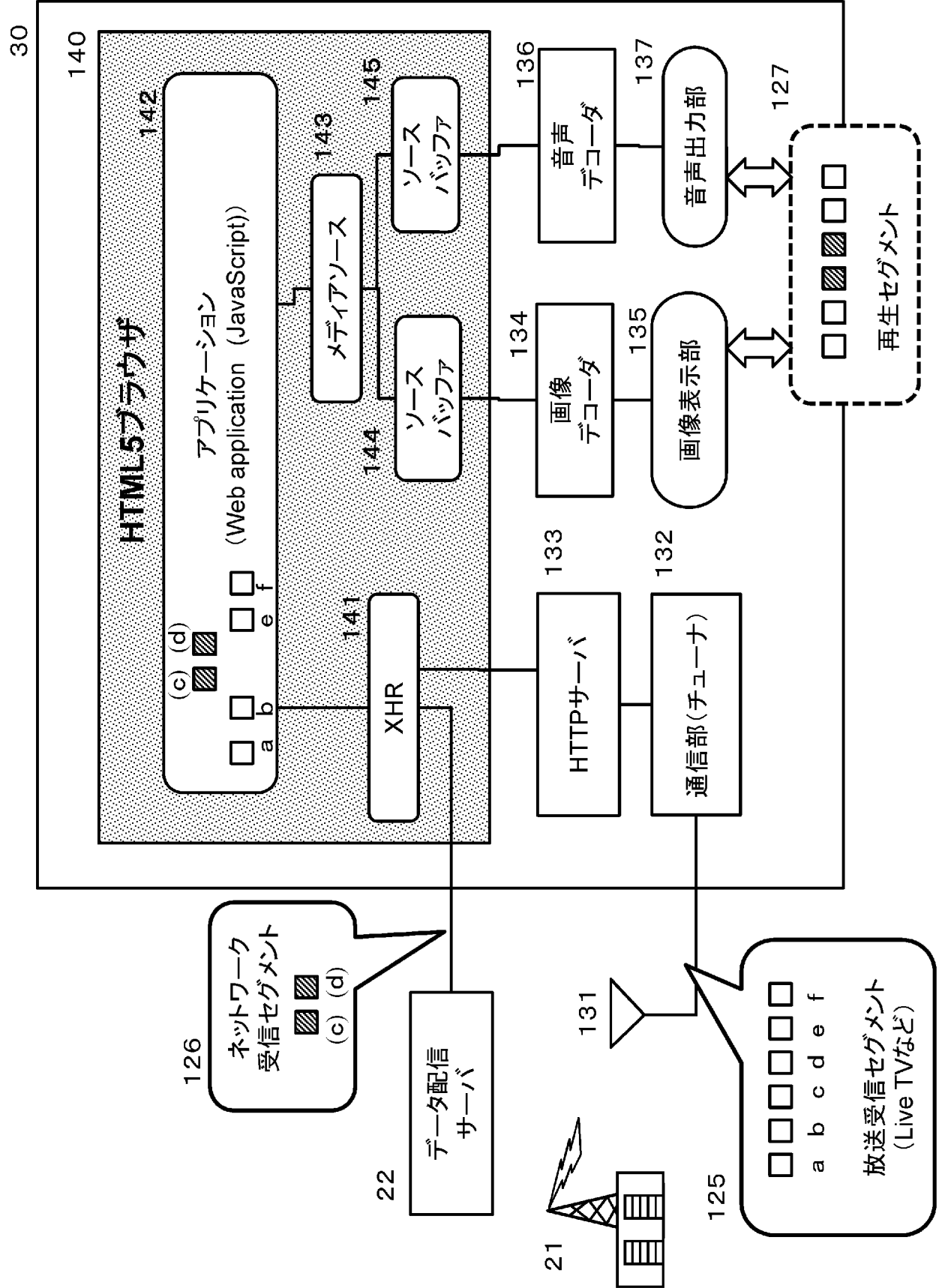
[図9]



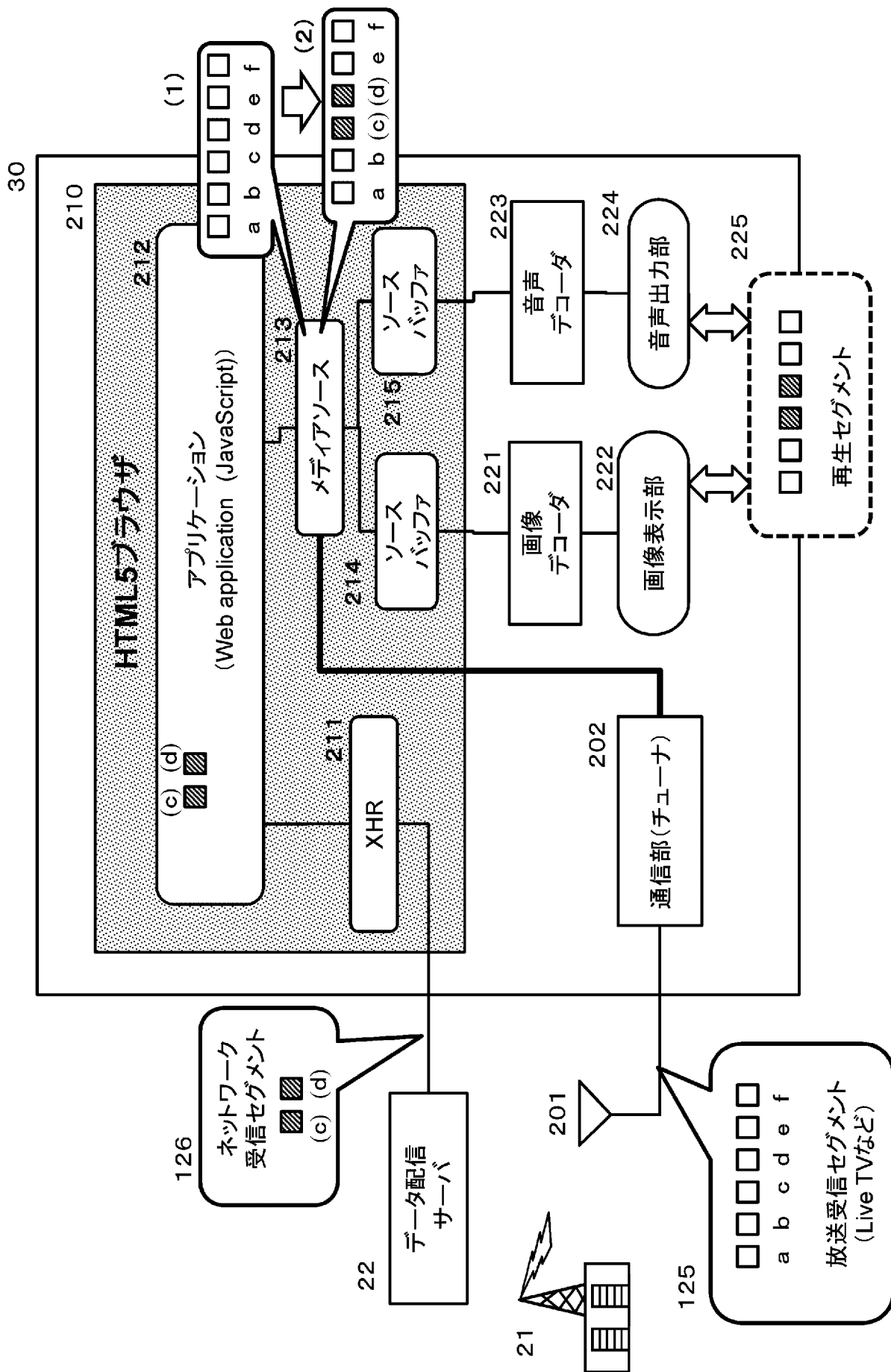
[図10]



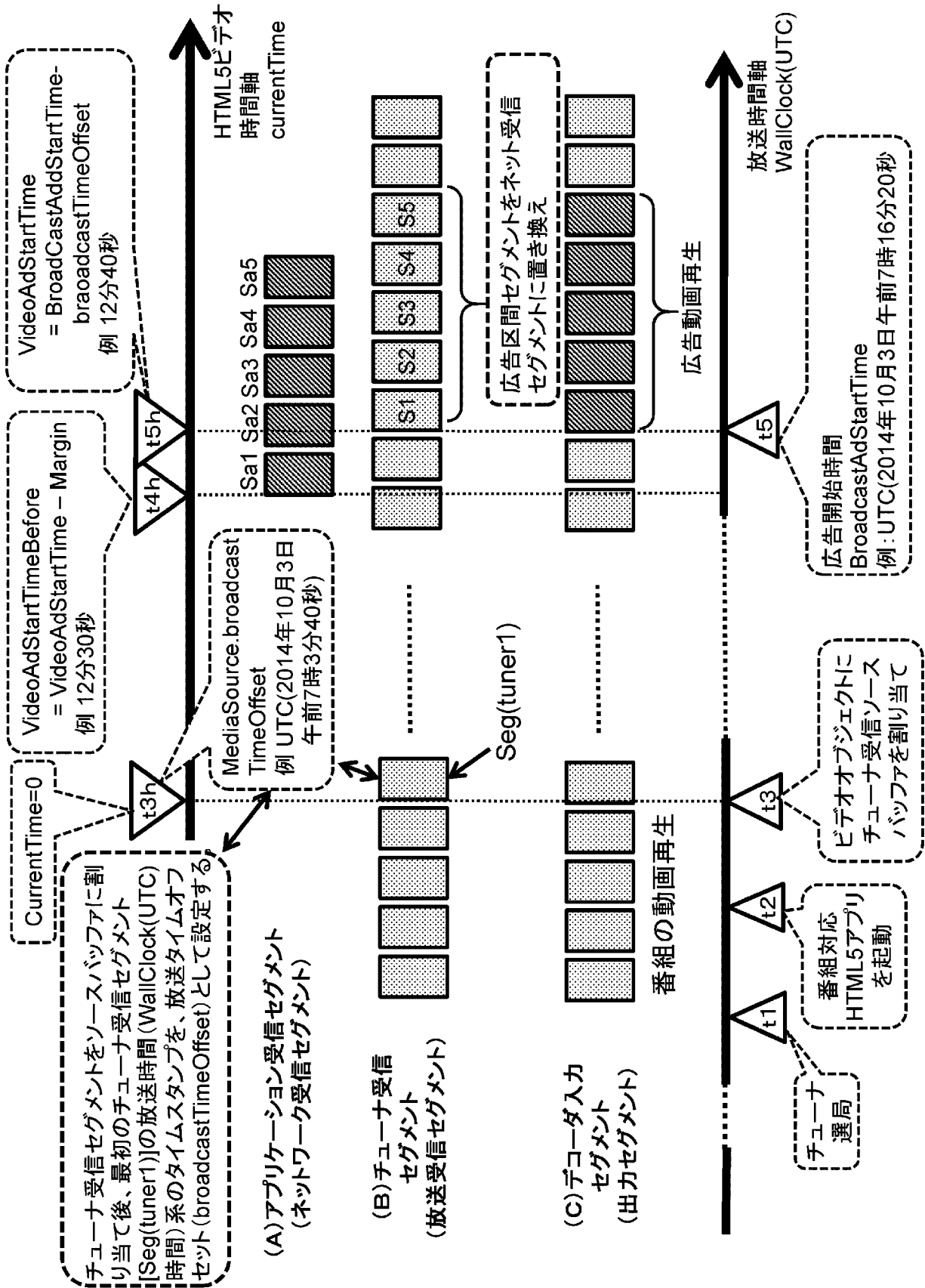
[図11]



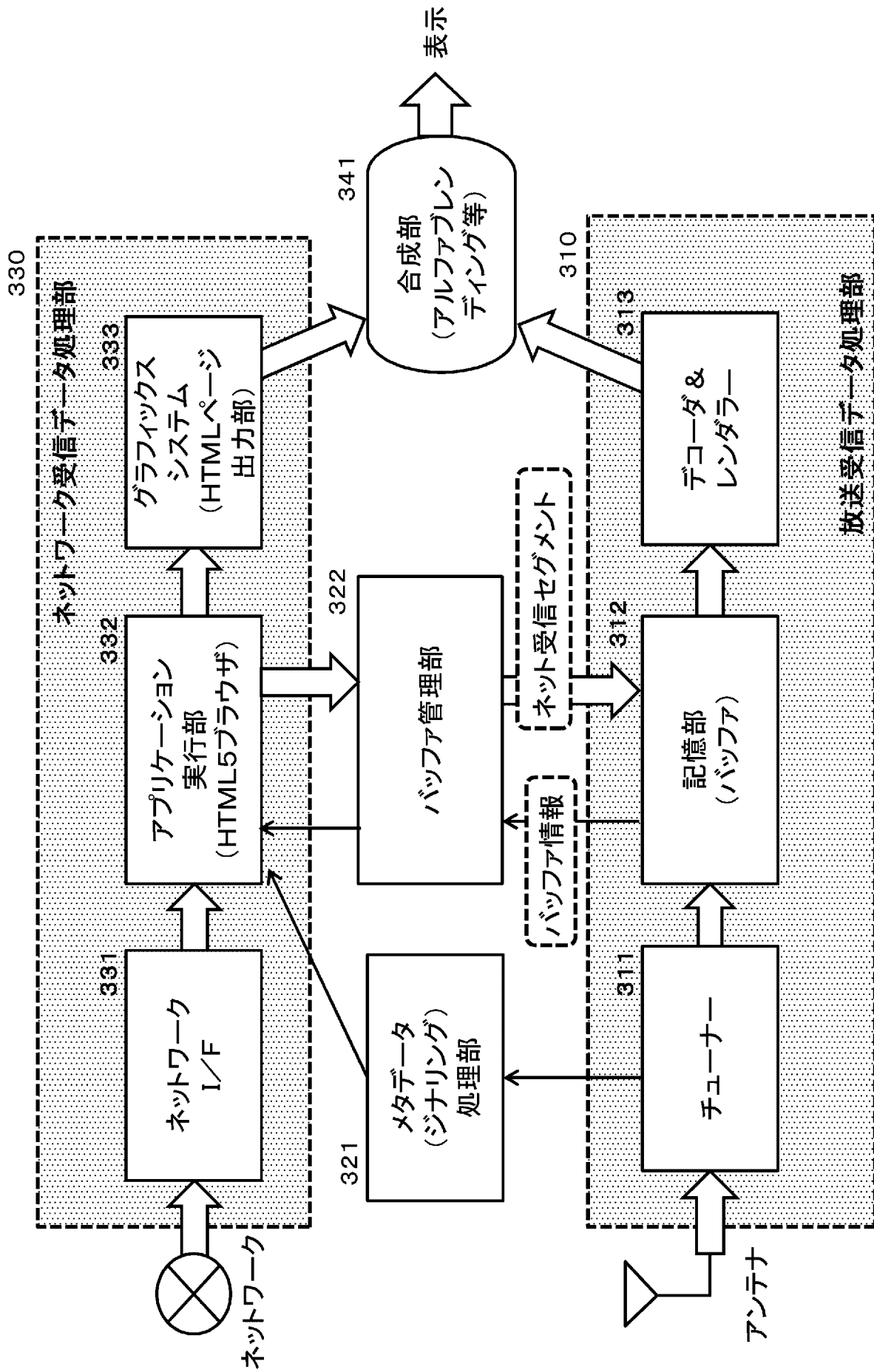
[図12]



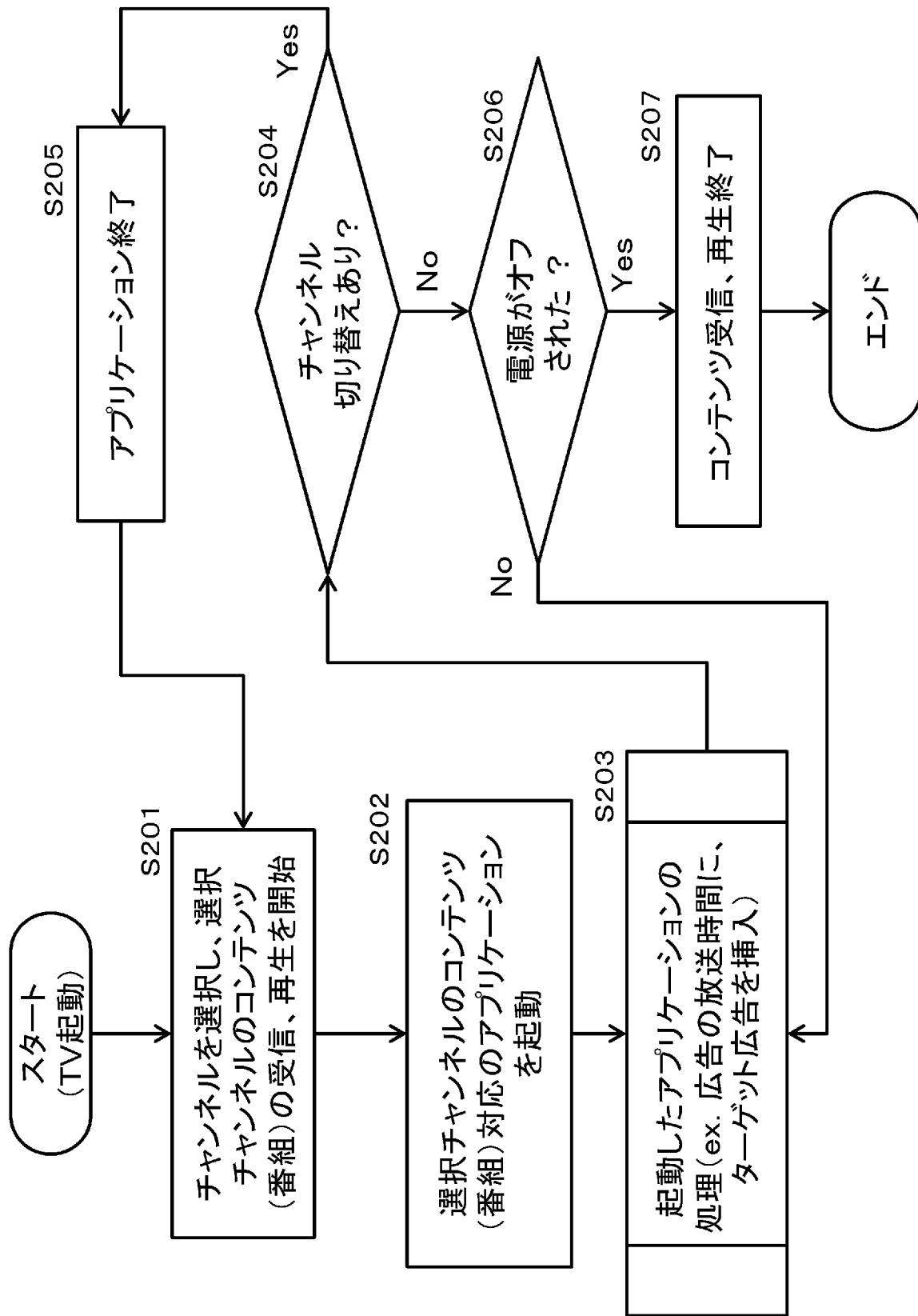
[図13]



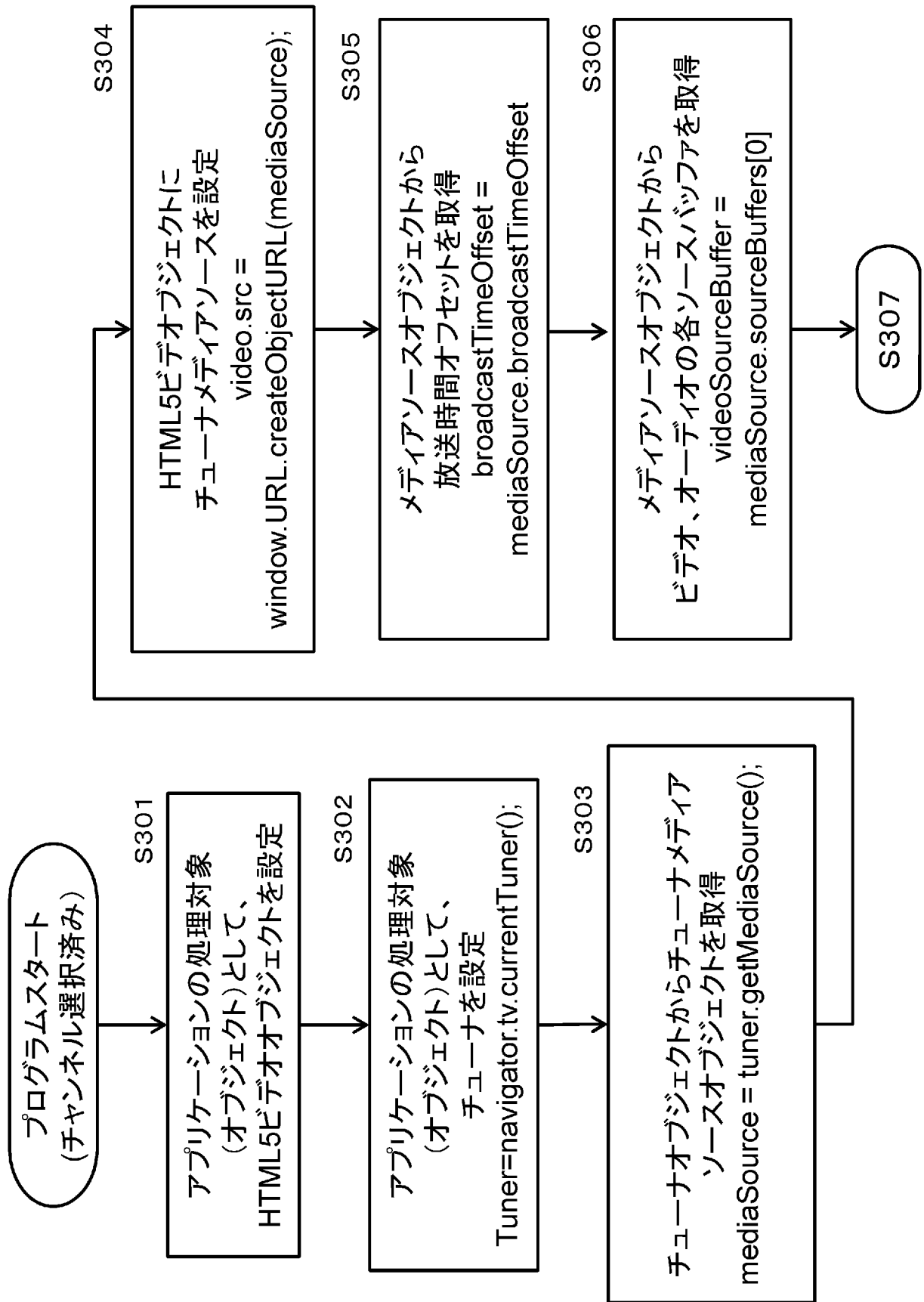
[図14]



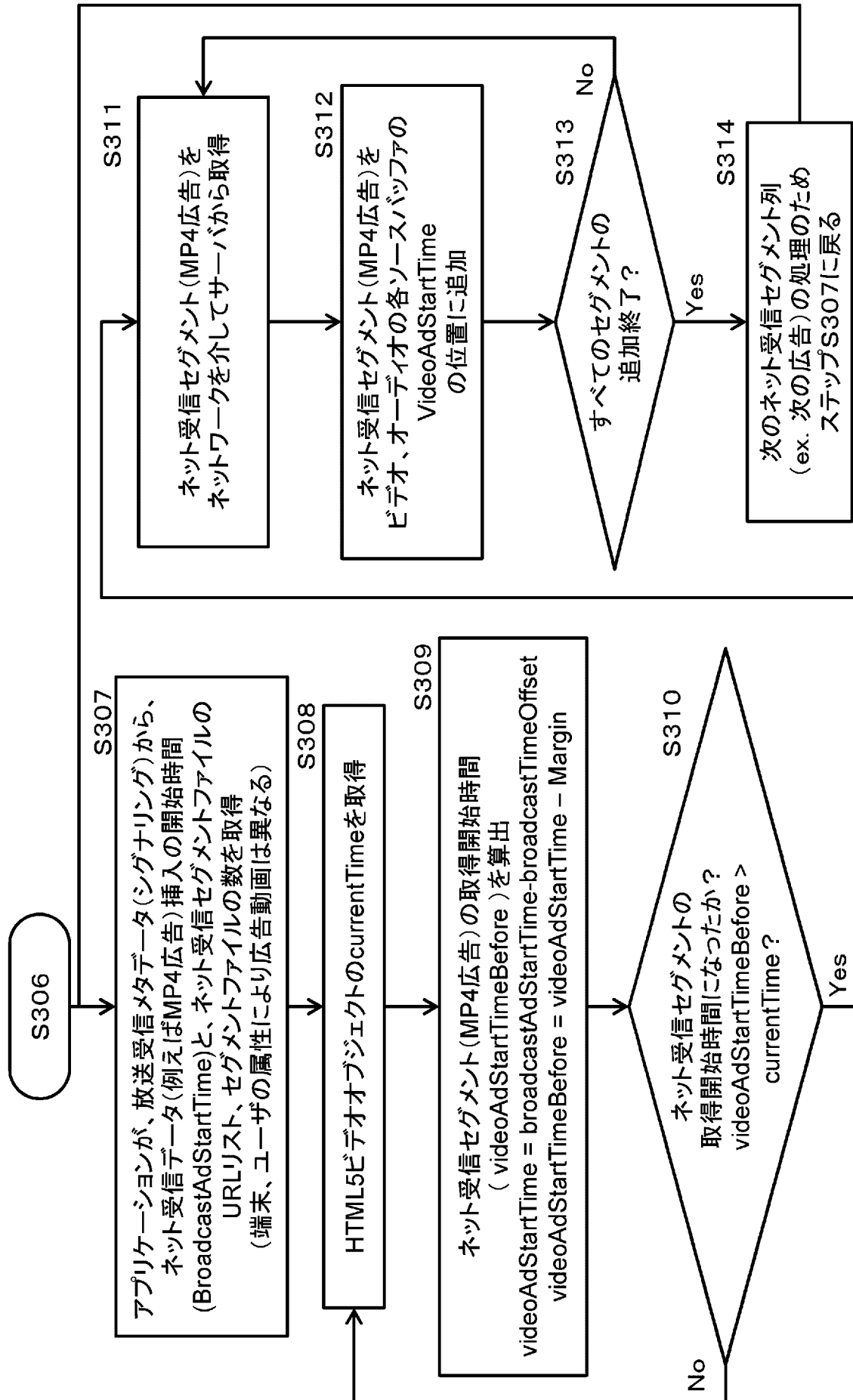
[図15]



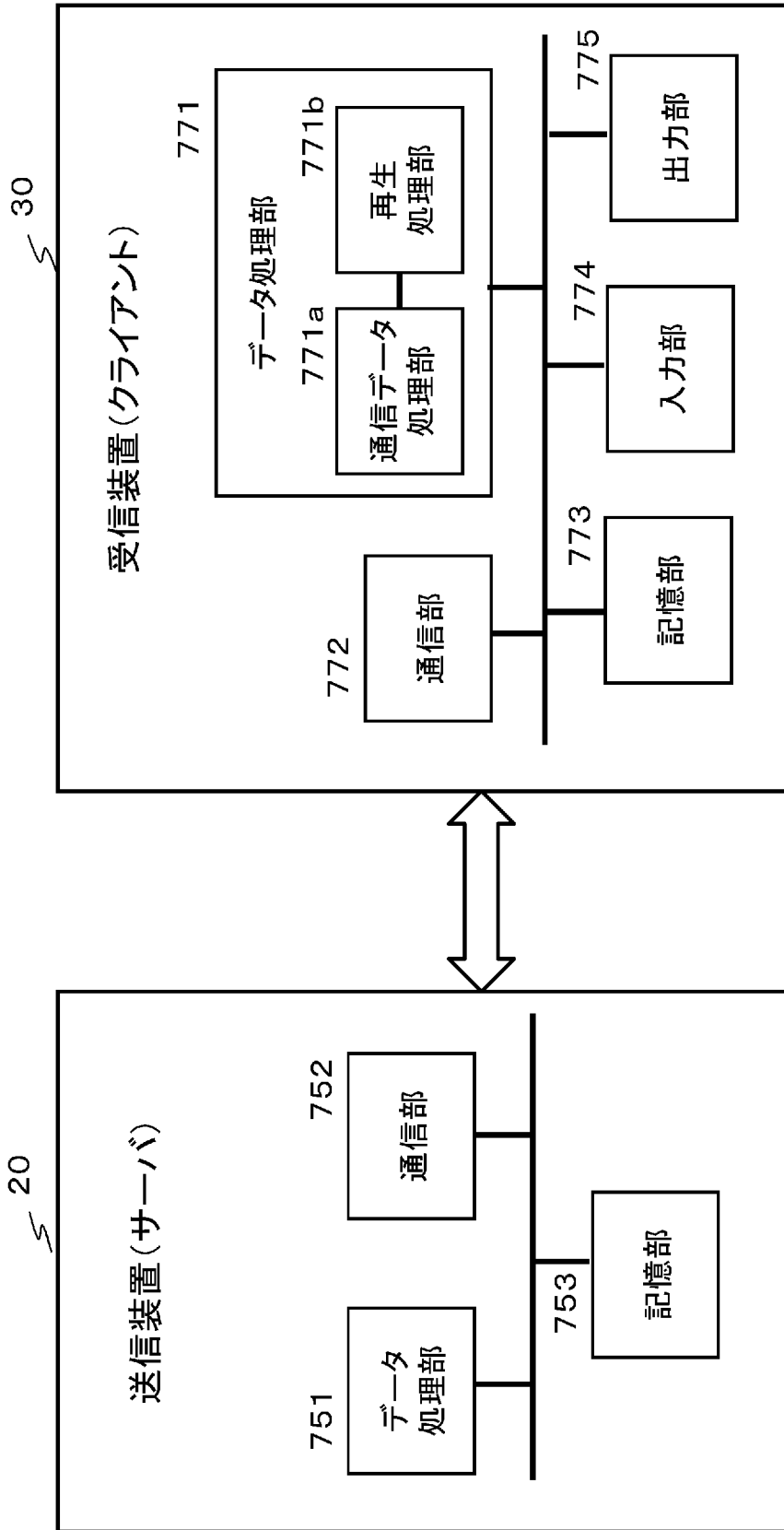
[図16]



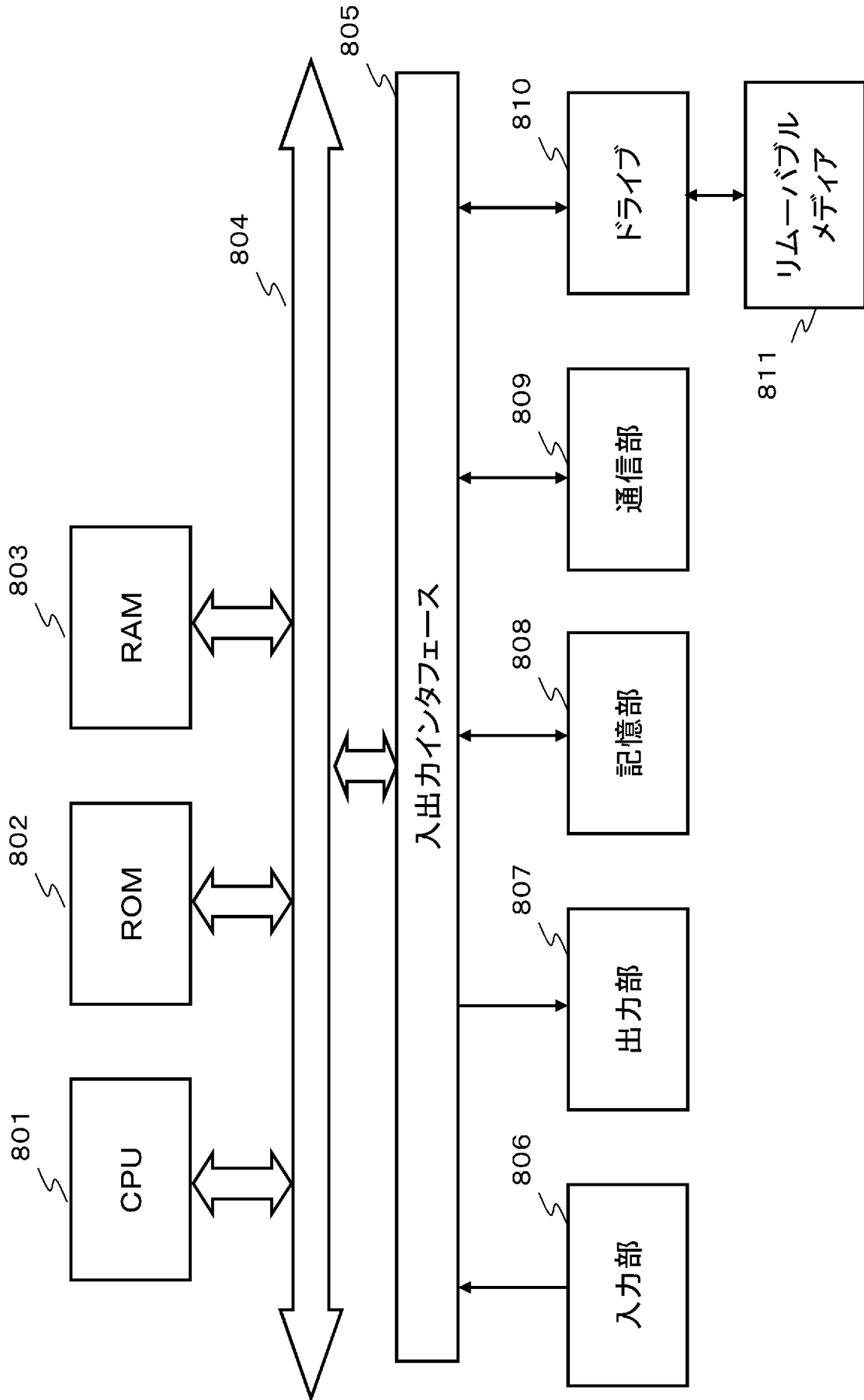
[図17]



[図18]



[図19]



INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2015/079098

A. CLASSIFICATION OF SUBJECT MATTER
H04N21/435(2011.01)i, H04N21/431(2011.01)i, H04N21/438(2011.01)i, H04N21/44(2011.01)i, H04N21/442(2011.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
 H04N21/435, H04N21/431, H04N21/438, H04N21/44, H04N21/442

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Jitsuyo Shinan Toroku Koho	1996-2016
Kokai Jitsuyo Shinan Koho	1971-2016	Toroku Jitsuyo Shinan Koho	1994-2016

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	JP 2013-9332 A (Nippon Hoso Kyokai), 10 January 2013 (10.01.2013), paragraphs [0006] to [0007] (Family: none)	1-12
A	WO 2014/162748 A1 (Panasonic Intellectual Property), 09 October 2014 (09.10.2014), paragraphs [0005], [0010] (Family: none)	1-12
A	WO 2014/115389 A1 (Toshiba Corp.), 31 July 2014 (31.07.2014), paragraphs [0006] to [0007] & US 2015/0043885 A1 paragraph [0028]	1-12

Further documents are listed in the continuation of Box C. See patent family annex.

* Special categories of cited documents:	"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A" document defining the general state of the art which is not considered to be of particular relevance	"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E" earlier application or patent but published on or after the international filing date	"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&" document member of the same patent family
"O" document referring to an oral disclosure, use, exhibition or other means	
"P" document published prior to the international filing date but later than the priority date claimed	

Date of the actual completion of the international search 07 January 2016 (07.01.16)	Date of mailing of the international search report 19 January 2016 (19.01.16)
---	--

Name and mailing address of the ISA/ Japan Patent Office 3-4-3, Kasumigaseki, Chiyoda-ku, Tokyo 100-8915, Japan	Authorized officer Telephone No.
--	---

A. 発明の属する分野の分類（国際特許分類（IPC）） Int.Cl. H04N21/435(2011.01)i, H04N21/431(2011.01)i, H04N21/438(2011.01)i, H04N21/44(2011.01)i, H04N21/442(2011.01)i		
B. 調査を行った分野 調査を行った最小限資料（国際特許分類（IPC）） Int.Cl. H04N21/435, H04N21/431, H04N21/438, H04N21/44, H04N21/442		
最小限資料以外の資料で調査を行った分野に含まれるもの 日本国実用新案公報 1922-1996年 日本国公開実用新案公報 1971-2016年 日本国実用新案登録公報 1996-2016年 日本国登録実用新案公報 1994-2016年		
国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）		
C. 関連すると認められる文献		
引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求項の番号
A	JP 2013-9332 A（日本放送協会）2013.01.10, 段落 [0006] - [0007] （ファミリーなし）	1-12
A	WO 2014/162748 A1 （パナソニック インテレクチュアル プロパティ）2014.10.09, 段落 [0005]、[0010] （ファミリーなし）	1-12
<input checked="" type="checkbox"/> C欄の続きにも文献が列挙されている。 <input type="checkbox"/> パテントファミリーに関する別紙を参照。		
* 引用文献のカテゴリー 「A」特に関連のある文献ではなく、一般的技術水準を示すもの 「E」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの 「L」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す） 「O」口頭による開示、使用、展示等に言及する文献 「P」国際出願日前で、かつ優先権の主張の基礎となる出願日の後に公表された文献 「T」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの 「X」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの 「Y」特に関連のある文献であって、当該文献と他の1以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの 「&」同一パテントファミリー文献		
国際調査を完了した日 07.01.2016	国際調査報告の発送日 19.01.2016	
国際調査機関の名称及びあて先 日本国特許庁（ISA/J P） 郵便番号100-8915 東京都千代田区霞が関三丁目4番3号	特許庁審査官（権限のある職員） 矢野 光治 電話番号 03-3581-1101 内線 3541	5C 3783

C (続き) . 関連すると認められる文献		
引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求項の番号
A	WO 2014/115389 A1 (株式会社東芝) 2014. 07. 31, 段落 [0006] - [0007] & US 2015/0043885 A1 [0028]	1 - 12