

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B1)

(11) 特許番号

特許第5725242号
(P5725242)

(45) 発行日 平成27年5月27日 (2015. 5. 27)

(24) 登録日 平成27年4月10日 (2015. 4. 10)

| | |
|---------------------------|----------------|
| (51) Int. Cl. | F I |
| HO 4 N 21/235 (2011. 01) | HO 4 N 21/235 |
| HO 4 N 21/435 (2011. 01) | HO 4 N 21/435 |
| HO 4 N 21/6543 (2011. 01) | HO 4 N 21/6543 |

請求項の数 5 (全 56 頁)

| | | | |
|-----------|------------------------------|-----------|----------------|
| (21) 出願番号 | 特願2014-116283 (P2014-116283) | (73) 特許権者 | 000002185 |
| (22) 出願日 | 平成26年6月4日 (2014. 6. 4) | | ソニー株式会社 |
| 審査請求日 | 平成26年12月19日 (2014. 12. 19) | | 東京都港区港南1丁目7番1号 |
| 早期審査対象出願 | | (74) 代理人 | 100093241 |
| | | | 弁理士 官田 正昭 |
| | | (74) 代理人 | 100101801 |
| | | | 弁理士 山田 英治 |
| | | (74) 代理人 | 100095496 |
| | | | 弁理士 佐々木 榮二 |
| | | (74) 代理人 | 100086531 |
| | | | 弁理士 澤田 俊夫 |
| | | (74) 代理人 | 110000763 |
| | | | 特許業務法人大同特許事務所 |

最終頁に続く

(54) 【発明の名称】 送信装置及び送信方法、並びに受信装置並びに受信方法

(57) 【特許請求の範囲】

【請求項1】

データ放送で利用されるデータ・ファイルを、ペイロードにデータ・ファイルを含むことを示す第1のペイロード種別情報をパケット・ヘッダーに挿入した第1の伝送パケットで送信するデータ・ファイル送信部と、

前記データ放送に関わるシグナリング・メッセージを、ペイロードにシグナリング・メッセージを含むことを示す第2のペイロード識別情報をパケット・ヘッダーに挿入した第2の伝送パケットで送信するシグナリング・メッセージ送信部と、
を具備し、

前記シグナリング・メッセージ送信部は、データ放送の提示単位を識別する提示単位タグと提示単位毎のキャッシュへのロック対象となるノードの数とロック対象ノードを識別するノード・タグを示すロック・キャッシュ情報を記載するデータ・コンテンツ管理テーブルと、前記ノード・タグに関連するデータ・ファイルを識別するアイテム識別情報を記載するデータ・アセット管理テーブルを前記シグナリング・メッセージに含めて送信する、
送信装置。

【請求項2】

前記シグナリング・メッセージ送信部は、前記データ放送の提示単位からリンクされる他の提示単位を識別するリンク提示単位タグをさらに前記シグナリング・メッセージに含めて送信する、

10

20

請求項 1 に記載の送信装置。

【請求項 3】

送信装置が備えるデータ・ファイル送信部が、データ放送で利用されるデータ・ファイルを、ペイロードにデータ・ファイルを含むことを示す第 1 のペイロード種別情報をパケット・ヘッダーに挿入した第 1 の伝送パケットで送信するデータ・ファイル送信ステップと、

前記送信装置が備えるシグナリング・メッセージ送信部が、前記データ放送に関わるシグナリング・メッセージを、ペイロードにシグナリング・メッセージを含むことを示す第 2 のペイロード識別情報をパケット・ヘッダーに挿入した第 2 の伝送パケットで送信するシグナリング・メッセージ送信ステップと、

を有し、

前記シグナリング・メッセージ送信ステップでは、データ放送の提示単位を識別する提示単位タグと提示単位毎のキャッシュへのロック対象となるノードの数とロック対象ノードを識別するノード・タグを示すロック・キャッシュ情報を記載するデータ・コンテンツ管理テーブルと、前記ノード・タグに関連するデータ・ファイルを識別するアイテム識別情報を記載するデータ・アセット管理テーブルを前記シグナリング・メッセージに含めて送信する、

送信方法。

【請求項 4】

データ放送で利用されるデータ・ファイルをペイロードに含むとともに、ペイロードにデータ・ファイルを含むことを示す第 1 のペイロード種別情報をパケット・ヘッダーに挿入した第 1 の伝送パケットを受信するデータ・ファイル受信部と、

前記データ放送に関わるシグナリング・メッセージをペイロードに含むとともに、ペイロードにシグナリング・メッセージを含むことを示す第 2 のペイロード識別情報をパケット・ヘッダーに挿入した第 2 の伝送パケットを受信するシグナリング・メッセージ受信部と、

前記シグナリング・メッセージに含まれる、データ放送の提示単位を識別する提示単位タグと提示単位毎のキャッシュへのロック対象となるノードの数とロック対象ノードを識別するノード・タグを示すロック・キャッシュ情報を記載するデータ・コンテンツ管理テーブルと、前記ノード・タグに関連するデータ・ファイルを識別するアイテム識別情報を記載するデータ・アセット管理テーブルとに基づいて、データ・ファイルのキャッシュを制御する制御部と、

を具備する受信装置。

【請求項 5】

受信装置が備えるデータ・ファイル受信部が、データ放送で利用されるデータ・ファイルをペイロードに含むとともに、ペイロードにデータ・ファイルを含むことを示す第 1 のペイロード種別情報をパケット・ヘッダーに挿入した第 1 の伝送パケットを受信するデータ・ファイル受信ステップと、

前記受信装置が備えるシグナリング・メッセージ受信部が、前記データ放送に関わるシグナリング・メッセージをペイロードに含むとともに、ペイロードにシグナリング・メッセージを含むことを示す第 2 のペイロード識別情報をパケット・ヘッダーに挿入した第 2 の伝送パケットを受信するシグナリング・メッセージ受信ステップと、

前記受信装置が備える制御部が、前記シグナリング・メッセージに含まれる、データ放送の提示単位を識別する提示単位タグと提示単位毎のキャッシュへのロック対象となるノードの数とロック対象ノードを識別するノード・タグを示すロック・キャッシュ情報を記載するデータ・コンテンツ管理テーブルと、前記ノード・タグに関連するデータ・ファイルを識別するアイテム識別情報を記載するデータ・アセット管理テーブルとに基づいて、データ・ファイルのキャッシュを制御する制御ステップと、

を有する受信方法。

【発明の詳細な説明】

10

20

30

40

50

【技術分野】

【0001】

本明細書で開示する技術は、所定のトランスポート方式によりデータ放送用のファイルを送信する送信装置及び送信方法、並びに、所定のトランスポート方式により伝送されるデータ放送用のファイルを受信する受信装置並びに受信方法に関する。

【背景技術】

【0002】

現在の放送システムでは、メディアのトランスポート方式として、MPEG-2 TS (Moving Picture Experts Group-2 Transport Stream)方式やRTP (Real Time Protocol)方式が広く使用されている(例えば、特許文献1を参照のこと)。次世代のデジタル放送方式として、MPEGで新たなメディア・トランスポート方式として規格化されたMMT (MPEG Media Transport) (例えば、非特許文献1を参照のこと)が検討されている。MMTでは異なる伝送路の組み合わせで利用することが容易であり、放送や通信の複数の伝送路に共通に用いることができる。

10

【0003】

MMT方式によれば、MMTパケット上で、ストリーム・メディアである映像や音声などのタイムド・メディア (Timed media) と、ファイルのようなノンタイムド・メディア (Non timed media) の両方を伝送することが可能である。ここで言うタイムド・メディアは、ビデオやオーディオ、字幕などの放送番組本編のストリーム・データである。また、ノンタイムド・メディアは、例えばHTML (Hyper Text Markup Language) 文書のようなデータ放送アプリケーション (コンテンツ) のファイル・データである。

20

【0004】

放送番組に連動したデータ放送は、タイムリーな提示を行なうことが要求される。一方、データ放送で利用される各ファイルは、限られた放送伝送帯域で繰り返し伝送される。受信端末は、データ放送用のファイルをキャッシュしておくことによりデータ放送のタイムリーな提示を実現することができる。しかしながら、潤沢なキャッシュ・メモリーを装備しない受信端末においては、必要なファイルをキャッシュ・ミスすると、次の繰り返し周期まで待たなければならなくなり、データ放送を提示するまで例えば数十秒程度の遅延を生じてしまう。

30

【0005】

本出願時に運用されているBML (Broadcast Markup Language) によるデータ放送サービスでは、スクリプトから「LockModuleOnMemory ()」というAPI (Application Programming Interface) を呼び出すことにより、特定のファイルをあらかじめキャッシュ・メモリーにプリキャッシュして留めておくことが可能である(例えば、特許文献2を参照のこと)。

【先行技術文献】

【特許文献】

40

【0006】

【特許文献1】特開2013-153291号公報

【特許文献2】特開2007-274193号公報

【非特許文献】

【0007】

【非特許文献1】ISO/IEC FDIS 23008-1:2013(E) Information technology - High efficiency coding and media delivery in heterogeneous environments - Part 1: MPEG media transport (MMT)

50

【発明の概要】

【発明が解決しようとする課題】

【0008】

本明細書で開示する技術の目的は、所定のトランスポート方式によりデータ放送用のファイルを好適に送信することができる送信装置及び送信方法を提供することにある。

【0009】

本明細書で開示する技術のさらなる目的は、所定のトランスポート方式により伝送されるデータ放送用のファイルを好適に受信することができる受信装置並びに受信方法を提供することにある。

【課題を解決するための手段】

【0010】

本願は、上記課題を参酌してなされたものであり、請求項1に記載の技術は、データ放送で利用されるファイル・データを送信するファイル・データ送信部と、データ放送に関わるシグナリングにファイル・データの強制キャッシュを指定する強制キャッシュ情報を含めて送信するシグナリング・メッセージ送信部と、を具備する送信装置である。

【0011】

本願の請求項2に記載の技術によれば、請求項1に記載の送信装置の前記シグナリング・メッセージ送信部は、データ放送提示単位毎に、提示単位を構成する放送伝送ファイル・リストと中心となるファイル、及び、プリキャッシュの対象ファイル・リストの情報を記述したデータ・コンテンツ・マネジメント・テーブルを含んだデータ・トランスミッション・メッセージを送信するように構成されている。

【0012】

本願の請求項3に記載の技術によれば、請求項1に記載の送信装置の前記シグナリング・メッセージ送信部は、データ放送提示単位毎に、提示単位を構成する放送伝送ファイル・リストと中心となるファイル、キャッシュにロックする対象ファイル及びロック対象のうちアンロックする対象ファイルの情報を記述したデータ・コンテンツ・マネジメント・テーブルを含んだデータ・トランスミッション・メッセージを送信するように構成されている。

【0013】

本願の請求項4に記載の技術によれば、請求項1乃至3のいずれかに記載の送信装置は、データ放送が連動する放送番組本体のメディア・データを送信するメディア・データ送信部をさらに備えている。

【0014】

また、本願の請求項5に記載の技術は、データ放送で利用されるファイル・データを送信するファイル・データ送信ステップと、データ放送に関わるシグナリングにファイル・データの強制キャッシュを指定する強制キャッシュ情報を含めて送信するシグナリング・メッセージ送信ステップと、を有する送信方法である。

【0015】

また、本願の請求項6に記載の技術は、データ放送で利用されるファイル・データを送信するファイル・データ受信部と、データ放送に関わるシグナリングにファイル・データの強制キャッシュを指定する強制キャッシュ情報を含めて送信するシグナリング・メッセージ受信部と、前記強制キャッシュ情報に基づいて前記ファイル・データ受信部が受信するファイル・データのキャッシュ・メモリーへのキャッシュを制御する制御部と、を具備する受信装置である。

【0016】

本願の請求項7に記載の技術によれば、請求項6に記載の受信装置の前記シグナリング

10

20

30

40

50

・メッセージ受信部は、データ放送提示単位毎に、提示単位を構成する放送伝送ファイル・リストと中心となるファイル、及び、プリキャッシュの対象ファイル・リストの情報を記述したデータ・コンテンツ・マネジメント・テーブルを含んだデータ・トランSMission・メッセージを受信するように構成されている。

【0017】

本願の請求項8に記載の技術によれば、請求項7に記載の受信装置の前記制御部は、前記プリキャッシュの対象ファイル・リストに含まれているファイルを前記ファイル・データが受信すると、前記キャッシュ・メモリーにプリキャッシュするように構成されている。

【0018】

本願の請求項9に記載の技術によれば、請求項6に記載の受信装置の前記シグナリング・メッセージ受信部は、データ放送提示単位毎に、提示単位を構成する放送伝送ファイル・リストと中心となるファイル、キャッシュにロックする対象ファイル及びロック対象のうちアンロックする対象ファイルの情報を記述したデータ・コンテンツ・マネジメント・テーブルを含んだデータ・トランSMission・メッセージを受信するように構成されている。

【0019】

本願の請求項10に記載の技術によれば、請求項9に記載の受信装置の前記制御部は、前記ロック対象のファイルを前記ファイル・データが受信すると、前記キャッシュ・メモリーにプリキャッシュするように構成されている。

【0020】

本願の請求項11に記載の技術によれば、請求項9に記載の受信装置の前記制御部は、前記アンロック対象のファイルを前記キャッシュ・メモリーから削除するように構成されている。

【0021】

本願の請求項12に記載の技術によれば、請求項6又は9のいずれかに記載の受信装置の前記制御部は、現在のデータ放送提示単位を構成する放送伝送ファイル・リストと中心となるファイルを前記ファイル・データが受信すると、前記キャッシュ・メモリーにキャッシュするように構成されている。

【0022】

本願の請求項13に記載の技術によれば、請求項6乃至12のいずれかに記載の受信装置は、ファイル・データを利用してデータ放送を提示するデータ放送提示部をさらに備えている。

【0023】

本願の請求項14に記載の技術によれば、請求項6乃至13のいずれかに記載の受信装置は、データ放送が連動する放送番組本体のメディア・データを受信するメディア・データ受信部と、メディア・データに基づいて放送番組を提示する放送番組提示部をさらに備えている。

【0024】

また、本願の請求項15に記載の技術は、
データ放送で利用されるファイル・データを送信するファイル・データ受信ステップと、
データ放送に関わるシグナリングにファイル・データの強制キャッシュを指定する強制キャッシュ情報を含めて送信するシグナリング・メッセージ受信ステップと、
前記強制キャッシュ情報に基づいて前記ファイル・データ受信部が受信するファイル・データのキャッシュ・メモリーへのキャッシュを制御する制御ステップと、
を有する受信方法である。

【発明の効果】

【0025】

本明細書で開示する技術は、例えばHTML(Hyper Text Markup

10

20

30

40

50

L a n g u a g e) 5 によるデータ放送のファイルを M M T 方式で伝送することを想定している。本明細書で開示する技術によれば、放送局などの送信装置は、データ放送に関わるシグナリングに、強制キャッシュを指定する情報を含めて伝送することができる。また、本明細書で開示する技術によれば、家庭内に設置された S T B (S e t T o p B o x) やテレビ受信端末などの受信装置は、受信したデータ放送に関わるシグナリングに含まれている強制キャッシュ情報に基づいてデータ放送用の各ファイルのキャッシュ制御を好適に行ない、放送番組に連動したデータ放送をタイムリーに提示することができる。

【 0 0 2 6 】

なお、本明細書に記載された効果は、あくまでも例示であり、本発明の効果はこれに限定されるものではない。また、本発明が、上記の効果以外に、さらに付加的な効果を奏する場合もある。

10

【 0 0 2 7 】

本明細書で開示する技術のさらに他の目的、特徴や利点は、後述する実施形態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。

【 図面の簡単な説明 】

【 0 0 2 8 】

【 図 1 】 図 1 は、本明細書で開示する技術を適用したデジタル放送システム 1 0 の構成例を模式的に示した図である。

【 図 2 】 図 2 は、M M T を適用した放送信号のスタック・モデル 2 0 0 を示した図である。

20

【 図 3 】 図 3 は、図 2 に示した放送信号を送出する放送送出システム 1 1 の構成例を示した図である。

【 図 4 】 図 4 は、図 2 に示した放送信号を受信する受信機 1 2 の構成例を示した図である。

【 図 5 】 図 5 は、M M T 方式に従って放送送出システム 1 1 から R F 伝送路に送出される放送信号 (パッケージ) 5 0 0 のイメージを示した図である。

【 図 6 】 図 6 は、M M T パケットのヘッダーの構成例を示した図である。

【 図 7 】 図 7 は、ノンタイムド・メディアを伝送する M M T P パケットの場合の拡張ヘッダー 7 0 0 の構成例を示した図である。

【 図 8 】 図 8 は、M P U モードの場合の M M T P ペイロード 8 0 0 の構成例を示した図である。

30

【 図 9 】 図 9 は、ペイロードにタイムド・メディアを配置した M F U の D U _ H e a d e r 9 0 0 の構成例を示した図である。

【 図 1 0 】 図 1 0 は、ペイロードにノンタイムド・メディアを配置した M F U の D U _ H e a d e r 1 0 0 0 の構成例を示した図である。

【 図 1 1 】 図 1 1 は、ノンタイムド・メディアのデータを伝送する際のパケット構成例を示した図である。

【 図 1 2 】 図 1 2 は、P A メッセージ 1 2 0 1 と、P A メッセージに含まれる M P テーブル 1 2 0 2 の構成例を示した図である。

【 図 1 3 】 図 1 3 は、P A メッセージ 1 3 0 0 のシンタックス例を示した図である。

40

【 図 1 4 】 図 1 4 は、P A メッセージに含まれるパラメーターの説明を示した図である。

【 図 1 5 】 図 1 5 は、M P テーブル (M P T) のシンタックス例 (前半部分) を示した図である。

【 図 1 6 】 図 1 6 は、M P テーブルのシンタックス例を示した図である。

【 図 1 7 】 図 1 7 は、M P テーブルに含まれる各パラメーターを説明した図である。

【 図 1 8 】 図 1 8 は、M 2 セクション・メッセージ 1 8 0 0 の構成例を示した図である。

【 図 1 9 】 図 1 9 は、M 2 セクション・メッセージで伝送される M H A I (A p p l i c a t i o n I n f o r m a t i o n) テーブル (M H A I T) 1 9 0 0 の構成例を示した図である。

【 図 2 0 】 図 2 0 は、アプリケーション情報記述子 2 0 0 0 の構成例を示した図である。

50

【図 2 1】図 2 1 は、アプリケーション情報記述子に含まれるパラメータの説明を示した図である。

【図 2 2】図 2 2 は、伝送プロトコル記述子 2 2 0 0 の構成例を示した図である。

【図 2 3】図 2 3 は、HTTP / HTTPS、MMT、及びノンタイムド伝送に共通のセクター・バイトの構成例を示した図である。

【図 2 4】図 2 4 は、シグナリング・メッセージの 1 つであるデータ・トランスミッション・メッセージの構成例を示した図である。

【図 2 5】図 2 5 は、データ・アセット・マネジメント・テーブル (DAMT) 2 5 0 0 の構成例を示した図である。

【図 2 6】図 2 6 は、データ・ディレクトリー・マネジメント・テーブル (DDMT) 2 6 0 0 の構成例を示した図である。

【図 2 7】図 2 7 は、データ・コンテンツ・マネジメント・テーブル (DCMT) 2 7 0 0 の構成例を示した図である。

【図 2 8】図 2 8 は、データ・コンテンツ・マネジメント・テーブル (DCMT) 2 7 0 0 の構成例を示した図である。

【図 2 9】図 2 9 は、MMT 伝送されるデータ放送アプリケーション (コンテンツ) の伝送、ロケーションと提示を行なう仕組みを説明するための図である。

【図 3 0】図 3 0 は、MMT 伝送路からデータ放送アプリケーション (コンテンツ) を取得する際の、シグナリング情報として伝送される各テーブルの参照関係を説明するための図である。

【図 3 1】図 3 1 は、受信機内でデータ放送アプリケーション (コンテンツ) をキャッシュする仕組みを模式的に示した図である。

【図 3 2】図 3 2 は、受信機 1 2 における方式 1 に基づくファイル・データのキャッシュ制御手順を示したフローチャートである。

【図 3 3】図 3 3 は、受信機 1 2 における方式 1 に基づくファイル・データのプリキャッシュ動作例を示した図である。

【図 3 4】図 3 4 は、受信機 1 2 における方式 2 に基づくファイル・データのキャッシュ制御手順を示したフローチャートである。

【図 3 5】図 3 5 は、受信機 1 2 における方式 2 に基づくファイル・データのキャッシュのロック及びアンロック動作例を示した図である。

【発明を実施するための形態】

【0029】

以下、図面を参照しながら本明細書で開示する技術の実施形態について詳細に説明する。

【0030】

図 1 には、本明細書で開示する技術を適用したデジタル放送システム 1 0 の構成例を模式的に示している。図示のデジタル放送システム 1 0 は、放送送出システム 1 1 と、受信機 1 2 で構成される。

【0031】

放送送出システム 1 1 は、伝送メディアを含む IP (Internet Protocol) 方式の放送信号を送信する。放送信号の伝送メディアには、タイムド・メディアと、ファイルのようなノンタイムド・メディアの両方が含まれる。タイムド・メディアは、例えば、ビデオやオーディオ、字幕などの放送番組本編に関わるストリーム・データである。また、ノンタイムド・メディアは、例えば HTML 文書のような、データ放送に利用される各ファイル・データである。以下の説明では、HTML 5 によるデータ放送サービスを想定している。

【0032】

一方、受信機 1 2 は、放送送出システム 1 1 から送られてくる放送信号を受信する。そして、受信機 1 2 は、受信した放送信号からビデオやオーディオ、字幕などの伝送メディアを取得して、画像や音声を提示する。また、受信機 1 2 は、受信した放送信号からデー

10

20

30

40

50

タ放送用の各ファイル・データを取得すると、HTMLブラウザなどのアプリケーション・エンジンを起動して、放送番組に連動したデータ放送の提示を行なう。

【0033】

図1に示したデジタル放送システム10では、放送送出システム11から受信機12へ放送信号を伝送する際のトランスポート方式として、MMTを適用することを想定している。図2には、この場合の放送信号構成例をスタック・モデル200で示している。

【0034】

スタック・モデル200の最下層には、物理レイヤー(PHY)201がある。物理例201には、変調方式や誤り訂正方式などが含まれる。

【0035】

物理レイヤー201の上に、TLV(Type Length Value)の伝送パケットのレイヤー202がある。また、TLV202の上にはIPパケット203が載り、さらにその上にUDP(User Datagram Protocol)204が載る。また、TLVの伝送パケット202の上には、IP203とUDP204のヘッダーを圧縮したヘッダー圧縮IP205と、シグナリング(Signaling)情報としての伝送制御信号206も載る。

【0036】

UDP204の上には、MMTパケット207、現在時刻の情報を含むNTP(Network Time Protocol)パケット208などが載る。MMTプロトコル(MMTP)は、MMTPペイロード209をIPネットワーク上で伝送するためのアプリケーション・レイヤーのトランスポート・プロトコルである。

【0037】

MMTパケット207のMMTペイロード209には、MFU(MMT Fragment Unit)210あるいはデータ放送に関わるシグナリング・メッセージ(Signaling Message)211が含まれる。MFU210は、符号化されたタイムド・メディア並びにノンタイムド・メディアのコンテナであるMPU(Media Processing Unit)のフラグメントである。MFU210には、ビデオやオーディオ、字幕などのストリーム・データ(タイムド・メディア)212や、HTML文書データなどのファイル・データ(ノンタイムド・メディア)213が挿入される。

【0038】

図3には、図2に示した放送信号を送出する放送送出システム11の構成例を示している。図示の放送送出システム11は、時計部301と、信号送出部302と、ビデオ・エンコーダー303と、オーディオ・エンコーダー304と、キャプション・エンコーダー305と、シグナリング・エンコーダー306と、ファイル・エンコーダー307と、情報システム308と、TLVシグナリング・エンコーダー309と、IPサービス・マルチプレクサー(MUX)310と、TLVマルチプレクサー(MUX)311と、変調・送信部312を備えている。

【0039】

時計部301は、NTPサーバー(図示しない)から取得した時刻情報に同期した時刻情報を生成し、この時刻情報を含むIPパケットをIPサービス・マルチプレクサー310に送る。

【0040】

信号送出部302は、例えばTV放送局のスタジオやVTRなどの記録再生機であり、タイムド・メディアであるビデオ、オーディオ、字幕などのストリーム・データや、ノンタイムド・メディアであるデータ放送用のHTML文書データなどのファイル・データをそれぞれ、ビデオ・エンコーダー303、オーディオ・エンコーダー304、キャプション・エンコーダー305、ファイル・エンコーダー307に送る。また、情報システム308は、TV放送局のスケジューラー並びにファイルの供給源であり、ノンタイムド・メディアであるHTML文書データ、シグナリング情報をそれぞれ、ファイル・エンコーダー307、シグナリング・エンコーダー306に送る。

10

20

30

40

50

【 0 0 4 1 】

ビデオ・エンコーダー 3 0 3 は、信号送出部 3 0 2 から送出されるビデオ信号を符号化し、さらにパケット化して、ビデオの M M T パケットを含む I P パケットを I P サービス・マルチプレクサー 3 1 0 に送る。また、オーディオ・エンコーダー 3 0 4 は、信号送出部 3 0 2 から送出されるオーディオ信号を符号化し、さらにパケット化して、オーディオの M M T パケットを含む I P パケットを I P サービス・マルチプレクサー 3 1 0 に送る。また、キャプション・エンコーダー 3 0 5 は、信号送出部 3 0 2 から送出される字幕信号を符号化し、さらにパケット化して、字幕の M M T パケットを含む I P パケットを I P サービス・マルチプレクサー 3 1 0 に送る。

【 0 0 4 2 】

シグナリング・エンコーダー 3 0 6 は、情報システム 3 0 8 から送出される情報に基づいてデータ放送に関わるシグナリング・メッセージを生成し、ペイロード部にこのシグナリング・メッセージが配置された M M T パケットを含む I P パケットを I P サービス・マルチプレクサー 3 1 0 に送る。本実施形態では、データ放送に関わるシグナリング・メッセージは、P A メッセージ、M 2 セクション・メッセージ、データ・トランスミッション・メッセージの 3 種類に大別される。本実施形態では、データ放送で利用される各ファイルの強制キャッシュを指定する情報がデータ・トランスミッション・メッセージ内に含まれる。各シグナリング・メッセージの詳細については後述に譲る。

【 0 0 4 3 】

ファイル・エンコーダー 3 0 7 は、信号送出部 3 0 2 又は情報システム 3 0 8 から送出されるファイル・データを、必要に応じて分割して、ファイル・データを含む M M T パケットを生成し、この M M T パケットを含む I P パケットを I P サービス・マルチプレクサー 3 1 0 に送る。なお、ファイル・データは、データ放送コンテンツ（データ放送用アプリケーション）を構成するものである。

【 0 0 4 4 】

放送送出システム 1 1 は、送出するチャンネル（放送番組）毎に I P サービス・マルチプレクサー 3 1 0 を装備する。1 つのチャンネルの I P サービス・マルチプレクサー 3 1 0 は、各エンコーダー 3 0 3 ~ 3 0 7 から送られてくるビデオ、オーディオ、字幕、シグナリング・メッセージ、及びファイル・データの各々を含む I P パケットをマルチプレクスして、1 つのチャンネルを構成する T L V パケットを生成する。

【 0 0 4 5 】

T L V シグナリング・エンコーダー 3 0 9 は、情報システム 3 0 8 から送出されるシグナリング情報をエンコードして、ペイロード部に配置する T L V パケットを生成する。

【 0 0 4 6 】

T L V マルチプレクサー 3 1 1 は、各 I P サービス・マルチプレクサー 3 1 0 - 1 ~ 3 1 0 - N 及び T L V シグナリング・エンコーダー 3 0 9 で生成される T L V パケットをマルチプレクスして、放送ストリームを生成する。

【 0 0 4 7 】

変調・送信部 3 1 2 は、T L V マルチプレクサー 3 1 1 で生成された放送ストリームに対して R F 変調処理を行なって、R F 伝送路に送出する。

【 0 0 4 8 】

図 3 に示した放送送出システム 1 1 の動作について説明しておく。

【 0 0 4 9 】

時計部 3 0 1 では、N T P サーバーから取得した時刻情報に同期した時刻情報が生成され、この時刻情報を含む I P パケットが生成される。

【 0 0 5 0 】

信号送出部 3 0 2 から送出されるビデオ信号は、ビデオ・エンコーダー 3 0 3、に供給される。ビデオ・エンコーダー 3 0 3 では、ビデオ信号が符号化され、さらにパケット化されて、ビデオの M M T パケットを含む I P パケットが生成される。この I P パケットは、I P サービス・マルチプレクサー 3 1 0 に送られる。

10

20

30

40

50

【 0 0 5 1 】

また、信号送出部 3 0 2 から送出されるオーディオ信号、字幕信号に対しても、同様の処理が行なわれる。そして、また、オーディオ・エンコーダー 3 0 4 で生成されるオーディオの M M T パケットを含む I P パケットが I P サービス・マルチプレクサー 3 1 0 に送られ、キャプション・エンコーダー 3 0 5 で生成される字幕の M M T パケットを含む I P パケットが I P サービス・マルチプレクサー 3 1 0 に送られる。

【 0 0 5 2 】

また、シグナリング・エンコーダー 3 0 6 では、情報システム 3 0 8 から送出される情報に基づいてデータ放送に関わるシグナリング・メッセージを生成され、ペイロード部にこのシグナリング・メッセージが配置された M M T パケットを含む I P パケットが生成される。この I P パケットは、I P サービス・マルチプレクサー 3 1 0 に送られる。

10

【 0 0 5 3 】

また、信号送出部 3 0 2 又は情報システム 3 0 8 から送出されるファイル・データは、ファイル・エンコーダー 3 0 7 に供給される。ファイル・エンコーダー 3 0 7 では、ファイル・データが必要に応じて分割され、ファイル・データを含む M M T パケットが生成され、この M M T パケットを含む I P パケットが生成される。この I P パケットは、I P サービス・マルチプレクサー 3 1 0 に送られる。

【 0 0 5 4 】

各 I P サービス・マルチプレクサー 3 1 0 では、各エンコーダー 3 0 3 ~ 3 0 7 から送られてくるビデオ、オーディオ、字幕、シグナリング・メッセージ、及びファイル・データの各々を含む I P パケットがマルチプレクスされて、1 つのチャンネルを構成する T L V パケットが生成される。

20

【 0 0 5 5 】

T L V シグナリング・エンコーダー 3 0 9 では、情報システム 3 0 8 から送出されるシグナリング情報がエンコードされて、ペイロード部に配置する T L V パケットを生成する。

【 0 0 5 6 】

T L V マルチプレクサー 3 1 1 では、各 I P サービス・マルチプレクサー 3 1 0 - 1 ~ 3 1 0 - N 及び T L V シグナリング・エンコーダー 3 0 9 で生成される T L V パケットがマルチプレクスされて、放送ストリームが生成される。変調・送信部 3 1 2 では、T L V マルチプレクサー 3 1 1 で生成された放送ストリームに対して R F 変調処理が行なわれ、その R F 変調信号が R F 伝送路に送出される。

30

【 0 0 5 7 】

また、図 4 には、図 2 に示した放送信号を受信する受信機 1 2 の構成例を示している。図示の受信機 1 2 は、チューナー・復調部 4 0 1 と、デマルチプレクサー (D E M U X) 4 0 2 と、時計部 4 0 3 と、ビデオ・デコーダー 4 0 4 と、オーディオ・デコーダー 4 0 5 と、キャプション・デコーダー 4 0 6 と、アプリケーション・データ制御部 4 0 7 と、キャッシュ・メモリー 4 0 8 と、データ放送アプリケーション・エンジン 4 0 9 と、システム制御部 4 1 0 と、合成部 4 1 1 と、I P インターフェース 4 1 2 を備えている。

【 0 0 5 8 】

チューナー・復調部 4 0 1 は、R F 変調信号を受信して、復調処理を行なって、放送ストリームを得る。デマルチプレクサー 4 0 2 は、この放送ストリームに対して、デマルチプレクス処理及びパケット化処理を行なって、N T P 時刻情報、P T S (P r e s e n t a t i o n T i m e S t a m p : 提示時刻情報)、シグナリング情報、放送番組本編に関わるビデオ、オーディオ、キャプションの各符号化信号、放送番組に連動するデータ放送に利用されるファイル・データ、並びにシグナリング情報を出力する。なお、データ放送に利用されるファイル・データは、例えば H T M L 5 形式で記述されたデータ放送アプリケーションである。

40

【 0 0 5 9 】

ビデオ・デコーダー 4 0 4 は、デマルチプレクサー 4 0 2 で得られる符号化ビデオ信号

50

をデコードして、ベースバンドのビデオ信号を得る。また、オーディオ・デコーダー 405 は、デマルチプレクサー 402 で得られる符号化オーディオ信号をデコードして、ベースバンドのオーディオ信号を得る。また、キャプション・デコーダー 406 は、デマルチプレクサー 402 で得られる符号化字幕信号をデコードして、字幕の表示信号を得る。

【0060】

アプリケーション・データ制御部 407 は、データ放送で利用される各ファイル・データの処理を行なう。本実施形態では、データ放送で利用される各ファイル・データは放送信号並びに IP ネットワークの 2 系統から伝送されることを想定し、前者はチューナー・復調部 401 及びデマルチプレクサー 402 経由で、後者は IP インターフェース 412 経由で、それぞれアプリケーション・データ制御部 407 が取得する。アプリケーション・データ制御部 407 は、デマルチプレクサー 402 から出力されるシグナリング情報に基づいて、取得したファイル・データの処理を制御する。具体的には、アプリケーション・データ制御部 407 は、現在いずれのデータ放送提示単位 (Presentation Unit: PU) にいるかを管理し、該当するファイル・データ (HTML5 などのデータ放送アプリケーション) の処理を HTML ブラウザーなどのデータ放送アプリケーション・エンジン 409 に指示する。データ放送アプリケーション・エンジン 409 は、適宜キャッシュ・メモリー 408 に事前キャッシュ又はプリキャッシュされたファイル・データを用いて、データ放送アプリケーションを処理する。

【0061】

アプリケーション・データ制御部 407 は、PA メッセージ、M2 セクション・メッセージ、データ・トランスミッション・メッセージの各々に含まれるシグナリング・テーブルを参照して、データ放送を提示するために必要なアクセス範囲を特定し、データ放送アプリケーション・エンジン 407 でキャッシュ可能なファイル・データを事前にキャッシュ・メモリー 408 にキャッシュするためのフィルタリング動作を制御する。ファイル・データの事前キャッシュの詳細については後述に譲る。

【0062】

また、データ放送で利用される各ファイルの強制キャッシュを指定する情報がデータ・トランスミッション・メッセージ内に含まれている。本実施形態では、アプリケーション・データ制御部 407 は、データ・トランスミッション・メッセージ内に含まれている強制キャッシュ情報に基づいて、データ放送をタイムリーに提示するために必要なファイル・データをキャッシュ・メモリー 408 にプリキャッシュする。ファイル・データのプリキャッシュの詳細については後述に譲る。

【0063】

なお、放送ストリームでは、同一コンテンツのファイル・データが繰り返し送られてくる。システム制御部 410 は、デマルチプレクサー 402 におけるフィルタリング動作を制御して、繰り返し送られてくるファイル・データ群の中からデマルチプレクサー 402 において必要なもののみがアプリケーション・データ制御部 407 で取得されるようにする。

【0064】

システム制御部 410 は、デマルチプレクサー 402 で得られるシグナリング情報や、ユーザー操作部 (図示しない) を介したユーザーからの操作情報などに基づいて、当該受信機 12 の各部の動作を制御する。時計部 403 は、デマルチプレクサー 402 で得られる NTP 時刻情報に基づいて、この時刻情報に同期した時刻情報を生成する。

【0065】

また、システム制御部 410 は、各デコーダー 404 ~ 406 におけるデコード・タイミングを PTS に基づいて制御し、ビデオ、オーディオ、字幕の提示タイミングを調整する。合成部 411 は、ベースバンドのビデオ信号に、字幕の表示信号及びデータ放送の表示信号を合成し、映像表示用のビデオ信号を得る。また、オーディオ・デコーダー 405 で得られるベースバンドのオーディオ信号は、音声出力用のオーディオ信号となる。ビデオ信号及びオーディオ信号からなる放送番組本編は、図示しないモニター・ディスプレイ

から映像及び音声出力される。また、データ放送アプリケーション・エンジン 409 が処理したデータ放送も、モニター・ディスプレイ上で放送番組本編の画面に重畳して表示される。

【0066】

図 4 に示した受信機 12 の動作について説明しておく。

【0067】

チューナー・復調部 401 では、RF 変調信号が受信され、復調処理が行なわれて、放送ストリームが得られる。デマルチプレクサー 402 では、この放送ストリームに対して、デマルチプレクス処理及びパケット化処理を行なわれ、NTP 時刻情報、PTS、シグナリング情報、ビデオ、オーディオ、キャプションの各符号化信号、並びに、ファイル・データが抽出される。

10

【0068】

デマルチプレクサー 402 で抽出された NTP 時刻情報は、時計部 403 に送られる。時計部 403 では、NTP 時刻情報に基づいて、この時刻情報に同期した時刻情報が生成される。つまり、時計部 403 では、放送送出システム 11 側の時計部 301 で生成された時刻情報に合った時刻情報が生成される。

【0069】

デマルチプレクサー 402 で抽出された符号化ビデオ信号は、ビデオ・デコーダー 404 に送られてデコードされ、ベースバンドのビデオ信号が得られる。また、デマルチプレクサー 402 で抽出された符号化字幕信号はキャプション・デコーダー 406 に送られてデコードされ、字幕の表示信号が得られる。また、デマルチプレクサー 402 で抽出されたファイル・データはデータ放送アプリケーション・エンジン 407 に送られて処理され、データ放送の表示信号が得られる。なお、システム制御部 410 によってデマルチプレクサー 402 におけるフィルタリング動作が制御されて、必要なファイル・データのみがデマルチプレクサー 402 で取得されるようにする。

20

【0070】

そして、合成部 411 では、ベースバンドのビデオ信号に、字幕の表示信号及びデータ放送の表示信号が合成され、映像表示用のビデオ信号が得られる。

【0071】

また、デマルチプレクサー 402 で抽出された符号化オーディオ信号はオーディオ・デコーダー 405 に送られてデコードされ、音声出力用のベースバンドのオーディオ信号が得られる。ビデオ信号及びオーディオ信号からなる放送番組本編は、図示しないモニター・ディスプレイから映像及び音声出力される。

30

【0072】

一方、アプリケーション・データ制御部 407 は、現在いずれのデータ放送提示単位にいるかを管理し、該当するファイル・データ (HTML5 などのデータ放送アプリケーション) の処理を HTML ブラウザなどのデータ放送アプリケーション・エンジン 409 に指示する。データ放送アプリケーション・エンジン 409 は、適宜キャッシュ・メモリー 408 に事前キャッシュ又はプリキャッシュされたファイル・データを用いて、データ放送アプリケーションを処理する。データ放送アプリケーション・エンジン 409 が処理したデータ放送も、モニター・ディスプレイ上で放送番組本編の画面に重畳して表示される。

40

【0073】

また、アプリケーション・データ制御部 407 は、PA メッセージ、M2 セクション・メッセージ、データ・トランスミッション・メッセージの各々に含まれるシグナリング・テーブルを参照して、キャッシュ・メモリー 408 の空き容量に応じたファイル・データの事前キャッシュや、放送番組に連動してデータ放送をタイムリーに提示するための強制キャッシュを制御する (後述)。

【0074】

図 1 に示したデジタル放送システム 10 では、放送送出システム 11 から受信機 12

50

へ放送信号を伝送する際のトランスポート方式として、MMTを適用することを想定している。図5には、MMT方式に従って放送送出システム11からRF伝送路に送出される放送信号500のイメージを示している。

【0075】

1つのチャンネル(放送番組)の放送信号は、ビデオ、オーディオ、字幕などの放送番組本編に関わるタイムド・メディアと、放送番組に連動するデータ放送に利用されるファイル・データのようなノンタイムド・メディアで構成され、これらをエンコードしたメディア・データをMPUに格納して伝送する。また、これらの放送信号の伝送制御などに関する情報を、シグナリング情報として伝送する。MMTでは、1つのチャンネル(放送番組)を構成するタイムド・メディア及びノンタイムド・メディアのデータを異なる伝送路の組み合わせで利用することが容易である。図5に示す例では、放送信号500として、ビデオ、オーディオ、字幕、ファイル・データ、シグナリング情報など、データのタイプ毎のMMT伝送路501~504が利用されている。なお、図中、字幕データ用の伝送路は便宜上、図示を省略している。

10

【0076】

1つのチャンネル(放送番組)は、ビデオ、オーディオ、字幕、ファイル・データ(データアプリケーション)などタイプの異なる複数のアセットで構成される「パッケージ」と言うことができる(パッケージは、MMT伝送路を使って伝送されるメディア・データの論理集合である)。各アセットは、同じasset_id(アセット識別子)を共有する1又はそれ以上のMPUの集合(論理グループ)であり、それぞれ専用のES(Elementary Stream)すなわちMMT伝送路上で伝送される(アセットは、固有の識別子に関連付けられ、マルチメディアのプレゼンテーションを構成するために使用されるデータのエンティティである)。すなわち、伝送路501では、共通のasset_idを持つMPU論理グループからなるビデオのMMTパケット(MMTP)が伝送され、伝送路502では、共通のasset_idを持つMPU論理グループからなるオーディオのMMTパケットが伝送され、伝送路503では、共通のasset_idを持つMPU論理グループからなるファイル・データのMMTパケットが伝送される。MPUは、asset_idと、該当する伝送路上でのMPUのシーケンス番号で特定される。また、各メディアを伝送するMMT伝送路は、asset_idで識別することができる。

20

30

【0077】

付言すれば、1つのパッケージ(放送番組)で、タイプが同じ複数の(すなわち、asset_idが異なる)アセットが伝送されることもある。例えば、同じ放送番組に対して、2以上のファイル・コンテンツ(データ放送アプリケーション)が提供される場合である。このような場合、異なるファイル・コンテンツには別々のasset_idが割り振られ、別々のMPU論理グループとして異なるMMT伝送路上で伝送されることになる。図5では、簡素化のため、ファイル・データ用の伝送路503を1本しか描いていない。

【0078】

また、MMTは、放送や通信の複数の伝送路に共通に用いることができる。HTML文書データのようなノンタイムド・メディアは、図5に示したように放送の伝送路でタイムド・メディアとともに伝送される以外に、IPネットワークなど通信の伝送路を介して提供することもできる。

40

【0079】

また、伝送路504では、同じシグナリング・メッセージを含んだMMTパケットが、繰り返し伝送される。本明細書で開示する技術を実現する上で、伝送されるシグナリング・メッセージは、PAメッセージ510、M2セクション・メッセージ520、データ・トランスミッション・メッセージ530の3種類のシグナリング・メッセージが関連する。各種シグナリング・メッセージで、シグナリング・テーブルが伝送される。例えば、PAメッセージ510内には、MP(MMT Package)テーブル511が含まれて

50

いる。また、M2セクション・メッセージ520内には、MH AI (Application Information) テーブル521が含まれている。また、データ・トランスミッション・メッセージ530は、データの伝送方法やデータ管理の制御方法を通知するためのメッセージであり、データ・ディレクトリー・マネジメント・テーブル531、データ・アセット・マネジメント・テーブル532、データ・コンテンツ・マネジメント・テーブル533の各シグナリング・テーブルが含まれている。各テーブルの詳細については後述に譲る。

【0080】

上述したように、MMTPでは、ビデオ、オーディオ、字幕などのタイムド・メディアや、ファイル・データのようなノンタイムド・メディアが伝送される。図6には、MMTPパケット600の構成例を示している。MMTPパケットは、MMTPプロトコルを用いて伝送されるようにフォーマットされたメディア・データのユニットである。詳細については、例えば非特許文献1を参照されたい。

10

【0081】

参照番号601で示すパケット・カウンター・フラグ「C」に1が代入されていると、参照番号602で示すパケット・カウンターのフィールドがこのMMTPパケット内に存在することが表される。パケット・カウンター602は、MMTPパケットをカウントした整数値を書き込む32ビット長のフィールドであり、MMTPパケットを送信する度に1ずつインクリメントされる。

【0082】

参照番号603で示す拡張フラグ「X」に1が代入されていると、参照番号604で示す拡張ヘッダー604が存在することが表される。図6の下には、拡張ヘッダー604の構成例を併せて示している。拡張ヘッダー604は、参照番号604-1で示す16ビット長のtypeフィールドと、参照番号604-2で示すlengthフィールドと、参照番号604-3で示すheader_extensin_valueフィールドで構成される。lengthフィールドには、header_extensin_valueフィールドのバイト長が書き込まれる。header_extensin_valueフィールドには、MMTの仕様から外れた拡張情報を書き込むことができる。

20

【0083】

参照番号606で示すtypeフィールドには、当該MMTPパケットのペイロード・データのタイプを表すタイプ値が書き込まれる。タイプ値の定義を以下の表1に示しておく。

30

【0084】

【表1】

| Value | Data type | Definition of data unit |
|-------|-------------------|--|
| 0x00 | MPU | a media-aware fragment of the MPU |
| 0x01 | Generic object | A generic object such as a complete MPU or an object of another type |
| 0x02 | Signaling message | one or more signaling messages or a fragment of a signaling message |
| 0x03 | Repair symbol | a single complete repair symbol |

40

【0085】

参照番号605で示すRAP (Random Access Point) フラグに1が代入されていると、当該MMTPパケットのペイロードが当該データ・タイプのデータ・ストリームへのRandom Access Pointを含んでいることを表す。

50

【0086】

参照番号607で示す、16ビット長のpacket_idフィールドには、アセットを区別するための整数値が書き込まれる。このフィールドの値は、当該MMTPパケットが属するアセットのasset_idに由来する。packet_idとasset_idのマッピングは、シグナリング・メッセージの一部であるMMTPパッケージ(MP)テーブルで示されている。

【0087】

参照番号608で示す、32ビット長のtimestampフィールドには、当該MMTPパケットの送信時間が、NTPプロトコルで規定されているshort-formatで記載される。

10

【0088】

参照番号609で示す、32ビット長のpacket_sequence_numberフィールドには、同一のpacket_idを持つパケットを識別するための整数値(MMT伝送路上でのシーケンス番号)が記載される。

【0089】

図7には、ノンタイムド・メディアを伝送するMMTPパケットの場合の拡張ヘッダー700の構成例を示している。図示のように、この場合、lengthフィールド701には、header_extensin_valueフィールドのバイト長として4が書き込まれる。header_extensin_valueフィールド702には、download_idが4バイトで記載される。

20

【0090】

MMTPプロトコルを使ってMPUを伝送する際、送信側及び受信側ではそれぞれパケット化、デパケット化が必要である。パケット化により、MPUはMMTPペイロードに挿入され、MMTPパケットで伝送される。MMTPペイロードのフォーマットは、大きなペイロードの伝送が可能のように、MMTPペイロードのフラグメンテーションを許容する。また、MPUペイロードのフォーマットは、小さなデータ・ユニットに対応して、複数のMMTPペイロードを単一のMMTPペイロードに挿入するアグリゲーションも許容する。受信側では、デパケット化して、元のMPUデータを復元する。

【0091】

図8には、MPUモードの場合のMMTPペイロード800の構成例を示している。詳細については、例えば非特許文献1を参照されたい。MPUモードは、MMTPヘッダーのtypeフィールド606に「0x00」が書き込まれている場合である。MPUモードのMMTPパケットは、放送番組本編に関わるビデオ、オーディオ、並びに、放送番組に連動するファイル・データ(データ放送アプリケーション)の伝送に使用される。

30

【0092】

参照番号801で示すMPU Fragment Type(FT)フィールドには、フラグメントのタイプが4ビットの値で示される。FT値の定義を以下の表2に示しておく。

【0093】

【表 2】

| FT | Description | Content |
|----|-------------------------|--|
| 0 | MPU metadata | contains the ftyp, mmpu, moov, and meta boxes as well as any other boxes that appear in between. |
| 1 | Movie fragment metadata | contains the moof box and the mdat box, excluding all media data inside the mdat box. |
| 2 | MFU | contains a sample or sub-sample of timed media data or an item of non-timed media data. |

10

【0094】

参照番号802で示すTimed(T)フラグに1が記入されているときには、タイムド・メディアを伝送するMPUがフラグメントされていることを示し、0が記入されているときには、ノンタイムド・メディアを伝送するMPUがフラグメントされていることを示す。

【0095】

参照番号803で示すFragmentation Identifier(f_i)フィールドは、ペイロード内のデータ・ユニットのフラグメンテーションに関する情報を、2ビットで表す。f_iの4つの値の定義を以下の表3に示しておく。

20

【0096】

【表 3】

| Value | Definition of data unit |
|-------|-----------------------------|
| 00 | ペイロードは整数個のデータ・ユニットを含む。 |
| 01 | ペイロードは最初のフラグメントのデータ・ユニットを含む |
| 10 | ペイロードは中間のフラグメントのデータ・ユニットを含む |
| 11 | ペイロードは最後のフラグメントのデータ・ユニットを含む |

30

【0097】

当該ペイロードが複数のデータ・ユニットをアグリゲートしたものであるときには、参照番号804で示すaggregation(A)フラグに1が記入される。

【0098】

参照番号805で示す、8ビット長のfragment_counterフィールドには、当該MMTPペイロードが続く同じデータ・ユニットのフラグメントを含んでいるペイロードの数が記載される。

40

【0099】

参照番号806で示す、16ビット長のDU_lengthフィールドには、当該フィールドに続くデータ(DU: Data Unit)の長さが記載される。但し、Aフラグ804が0のときは、DU_lengthフィールド806はない。

【0100】

参照番号807で示すDU_Headerは、データ・ユニットのヘッダーである。但し、FT値801が0又は1のとき(言い換えれば、MFUでないとき)には、DU_H

50

header 807はない。MFUは、タイムド・メディアのサンプル若しくはサブサンプル、又は、ノンタイムド・メディアのアイテムを含んでいる。

【0101】

図9には、ペイロードにタイムド・メディアを配置したMFUのDU__Header 900の構成例を示している。また、図10には、ペイロードにノンタイムド・メディアを配置したMFUのDU__Header 1000の構成例を示している。図10に示すように、ノンタイムド・メディアの場合のDU__Header 1000は、当該MFUの一部として伝送されるアイテムの識別子である32ビット長のitem__IDで構成される。アイテムは、HTML文書データや、HTML文書から参照されるモノメディア・データなどの、アプリケーションを構成するリソースである。asset__idで指定されたMMT伝送路上では、上述したMMTPパケットのヘッダー内のpacket__id及び拡張ヘッダー内のdownload__idと、DUヘッダー内のitem__IDの組み合わせで、アイテムを一意に特定することができる。

10

【0102】

図11には、ノンタイムド・メディアのデータを伝送する際のパケット構成例を示している。

【0103】

図11(a)には、元のファイル・データの状態を示している。同図中、F1、F2はそれぞれ1つのファイル・データである。ファイル・データは、例えばHTML文書であり、1以上のアイテムを含んでいる。また、HTML文書自体も1つのアイテムである。

20

【0104】

図11(b)には、各ファイル・データF1、F2をMFUに配置した様子を示している。ファイル・データF1は、ファイル・サイズが大きくないので、そのまま1つのMFUのペイロードに配置される。一方、ファイル・データF2は、ファイル・サイズが大きいため、複数個に分割され、それぞれがMFUのペイロードに配置される。図示の例では、ファイル・データF2は、F2-1とF2-2に2分割され、それぞれが別のMFUのペイロードに配置されている。

【0105】

ここで、HTML文書データやモノメディアなどのノンタイムド・メディアがペイロードに配置されるMFUには、そのアイテムを一意に示すitem__IDが記載されたDU__Header(図10を参照のこと)がそれぞれ付けられる。

30

【0106】

次いで、図11(c)に示すように、各MFUには、MMTペイロードのヘッダー(図8を参照のこと)が付けられて、MMTペイロードとなる。ここで、MMTペイロードのヘッダーのFragment Type(FT)フィールドに値2を記載して、フラグメントのタイプがMFUであることを示す。また、Timed(T)フラグに値0を記載して、ノンタイムド・メディアを伝送するMPUであることを示す。また、フラグメントしていないノンタイムド・メディアを配置したMFUには、Fragmentation Identifier(f__i)フィールドに値0を記載する。一方、フラグメントしたノンタイムド・メディアを配置したMFUには、Fragmentation Identifier(f__i)フィールドに値1を記載するとともに、fragment__counterフィールドに該当するカウント値を記載する。

40

【0107】

次いで、図11(d)に示すように、各MMTペイロードに、MMTPパケットのヘッダー及び拡張ヘッダー(図6を参照のこと)が付けられて、MMTパケット・ストリームとなる。ここで、MMTPヘッダーのtypeフィールドには、0を記載して、ペイロード・データのタイプがMPUであることを記載し、packet__idフィールドにはアセットを区別するための整数値が書き込まれる。また、拡張ヘッダーには、download__idが記載される。したがって、asset__idで指定されたMMT伝送路上では、上述したMMTPパケットのヘッダー内のpacket__id及び拡張ヘッダー内の

50

`download_id`と、DUヘッダー内の`item_ID`の組み合わせで、アイテムを一意に特定することができる。

【0108】

さらに、図11(e)に示すように、各MMTパケットにIPヘッダー及びUDPヘッダーが付けられて、IPパケット・ストリームとなる。図示を省略したが、各IPパケットにTLVヘッダーを付けることで、放送ストリームを構成するTLVパケットが生成される。

【0109】

なお、図11では図示を省略したが、MMTパケットには、シグナリング・メッセージをペイロードに含むMMTパケットも存在する。シグナリング・メッセージは、PAメッセージ、M2セクション・メッセージ、データ・トランスミッション・メッセージがある（前述並びに図5を参照のこと）。MMTPペイロードにタイムド・メディアやノタイムド・メディアなどの伝送メディアが含まれるか、あるいは、シグナリング・メッセージが含まれるかは、MMTPヘッダー内の`type`フィールドの値を参照して識別することができる。

10

【0110】

続いて、本明細書で開示する技術を実現する上で関連する、MMTプロトコルで使用されるシグナリング・メッセージの構成について説明する。シグナリング・メッセージは、パッケージの伝送制御やパッケージの使用に必要なシグナリング情報であり、各種のシグナリング・テーブルを伝送する。

20

【0111】

MMTのシグナリング・メッセージは、3つの共通するフィールドと、シグナリング・メッセージ・タイプ毎の特定の1つのフィールドと、メッセージ・ペイロードからなる一般的なフォーマットを使用する。メッセージ・ペイロードは、シグナリング情報を伝送する。以下、PAメッセージ、M2セクション・メッセージ、データ・トランスミッション・メッセージの順に説明する。

【0112】

PA(Package Access)メッセージは、Package Accessに必要なすべてのシグナリング・テーブル上の情報を持つPAテーブルを伝送する。PAテーブルには、MMTPackage(MP)テーブルが含まれる。図12には、シグナリング・メッセージの1つであるPAメッセージ1201と、PAメッセージに含まれるMPテーブル1202の構成例を示している。また、図13には、PAメッセージ1300のシンタックス例を示し、図14には、PAメッセージに含まれるパラメーターの説明を示している。

30

【0113】

`message_id`は、各種シグナリング情報において、PAメッセージを識別する16ビットの固定値である。`version`は、PAメッセージのバージョンを示す、8ビットの整数値のパラメーターである。例えばMPテーブルを構成する一部のパラメーターでも更新した場合には、`version`は+1だけインクリメントされる。`length`は、このフィールドの直後からカウントされる、当該PAメッセージのサイズをバイト単位で示す、32ビット長のパラメーターである。

40

【0114】

`extension`フィールドには、`payload`のフィールドに配置されるMPテーブル(MPT)のインデックス情報が配置される。このフィールドには、8ビットの`table_id`と、8ビットの`table_version`と、16ビットの`table_length`が配置される。`table_id`は、MPテーブルを識別する固定値である。`table_version`は、MPテーブルのバージョンを示す。`table_length`は、MPテーブルのサイズをバイト単位で示す。

【0115】

PAメッセージの`payload`フィールドには、MPテーブルが配置される。MPテ

50

ーブルは、すべてのアセットのリストを含むパッケージに関連する情報を格納する。

【0116】

図15及び図16には、MPテーブルのシンタックス例を示している(図16は、図15の続く後半部分である)。また、図17には、MPテーブルに含まれるパラメータの説明を示している。以下、MPテーブルの構成について説明する。

【0117】

`table_id`は、各種シグナリング情報においてMPテーブルであることを識別する8ビットの固定値である。`version`は、MPテーブルのバージョンを示す8ビットの整数値である。例えば、MPテーブルを構成する一部のパラメータでも更新した場合には、`version`は+1だけインクリメントされる。`length`は、このフィールドの直後からカウントされる、MPテーブルのサイズをバイト単位で示す、32ビット長のパラメータである。

10

【0118】

`MMT_package_id`は、放送信号で伝送されるすべての信号(ビデオ、オーディオ、字幕)、並びにファイル・データなどのアセットを構成要素とする全体のパッケージとしての識別情報である。この識別情報は、テキスト情報である。`MMT_package_id_length`は、そのテキスト情報のサイズをバイト単位で示す。

【0119】

`MP_table_descriptors`のフィールドは、パッケージ全体に関わる記述子の格納領域である。`MP_table_descriptor_length`は、そのフィールドのサイズ N_2 をバイト単位で示す、16ビット長のパラメータである。そして、`MP_table_descriptor`は、さまざまな目的の記述子を規定した上で、 N_2 バイト分(1つ又は複数配置)することを想定している。

20

【0120】

`number_of_assets`は、パッケージを構成する要素としてのアセット(信号、ファイル)の数を示す、8ビットのパラメータである。`number_of_assets`の数分(N_3)だけ、以下の`Asset_loop`が配置される。

【0121】

1つの`Asset_loop`内には、個々のアセットの情報としての`asset_id_len`、`asset_id`、`gen_loc_info`、`asset_desc_len`、`asset_descriptor`の各パラメータが配置される。

30

【0122】

`asset_id`は、アセットをユニークに識別するテキスト情報である。`asset_id_len`は、`asset_id`のサイズをバイト単位で示す。`gen_loc_info`は、アセットの取得先のロケーションを示す情報である。本実施形態では、`gen_loc_info`は、アセットの取得先となる伝送路上の`packet_id`の形式で記述される。したがって、MPテーブル上で`asset_id`を引いて、MMT伝送路上の該当する`packet ID`を取り出すことができる。

【0123】

`asset_descriptor`のフィールドは、アセットに関わる記述子の格納領域である。`asset_descriptor_length`は、`asset_descriptor`フィールドのサイズ N_5 をバイト単位で示す。そして、`asset_descriptor`は、さまざまな目的の記述子を規定した上で、 N_5 個(1つ又は複数)配置することを想定している。

40

【0124】

M2セクション・メッセージは、MPEG-2 Systemのセクション拡張形式をそのまま伝送するために用いるシグナリング・メッセージである。図18には、M2セクション・メッセージ1800の構成例を示している。以下、M2セクション・メッセージの各パラメータの意味について説明する。

【0125】

50

message_id (メッセージ識別) は、各種シグナリング情報において、M2セクション・メッセージを識別する16ビットの固定値であり、本実施形態では0x8000とする。version (バージョン) は、M2セクション・メッセージのバージョンを示す、8ビットの整数値のパラメータである。length (メッセージ長) は、このフィールドの直後からカウントされる、当該M2セクション・メッセージのサイズをバイト単位で示す、16ビット長のパラメータである。table_id (テーブル識別) は、当該セクションが属するテーブルの識別のために使用する領域である。section_syntax_indicator (セクション・シンタクス指示) は、拡張形式を示す「1」とする。section_length (セクション長) は、セクション長領域より後に続くデータのバイト数を書き込む領域である。table_id_extension (テーブル識別拡張) は、テーブル識別の拡張を行なう領域である。version_number (バージョン番号) は、テーブルのバージョン番号を書き込む領域である。current_next_indicator (カレント・ネクスト指示) は、テーブルが現在使用可能である場合は「1」とし、テーブルが現在使用不可であり次に有効となることを示す場合は「0」とする。section_number (セクション番号) は、テーブルを構成するセクション番号を書き込む領域である。last_section_number (最終セクション番号) は、テーブルを構成する最後のセクション番号を書き込む領域である。CRC32 (CRC)、ITU-T勧告H.222.0に従う巡回冗長符号とする。

10

【0126】

20

図19には、M2セクション・メッセージで伝送されるMH AI (Application Information) テーブル (MH AIT) 1900の構成例を示している。以下、MH AI テーブルの各パラメータの意味について説明する。

【0127】

table_id (テーブル識別) は、各種シグナリング情報においてアプリケーション情報 (AI) テーブルであることを識別する8ビットの固定値であり、本実施形態では0x89とする。section_syntax_indicator (セクション・シンタクス指示) は、1ビットのフィールドで、常に「1」とする。section_length (セクション長) は、12ビットのフィールドで、先頭の2ビットは常に「00」とする。これは、セクション長フィールドからCRC32を含むセクションの最後までのセクションのバイト数を規定する。この値は1021 (16進数で0x3FD) を超えないものとする。application_type (アプリケーション形式) は、16ビットのフィールドで、AITで伝送しているアプリケーションの値を示す。DVBでは、DVB-Jアプリケーションに対して0x0001が割り当てられている。ARIB-Jアプリケーションにおいても0x0001とする。version_number (バージョン番号) は、5ビットのフィールドで、サブテーブルのバージョン番号である。version_number は、当該MH AI テーブルのバージョン番号であり、サブテーブル内の情報に変化があった場合に+1だけインクリメントされる。また、バージョン番号の値が「31」になったとき、その次は「0」に戻る。current_next_indicator (カレント・ネクスト指示) は、常に「1」とする。section_number (セクション番号) は、8ビットのフィールドで、セクションの番号を表す。サブテーブル内で最初のセクションのセクション番号は0x00である。セクション番号は、同一のテーブル識別及びアプリケーション形式を持つセクションが追加される度に+1だけインクリメントされる。last_section_number (最終セクション番号) は、8ビットのフィールドであり、そのセクションが属するサブテーブルにおける最後のセクション番号を規定する。

30

40

【0128】

common_descriptor_length (共通記述子ループ長) は、8ビットのフィールドで、後続のdescriptor (記述領域内記述子) のバイト長を規定する。このdescriptor (記述領域内記述子) は、common_desc

50

`iptor_length`の数分のループからなる一連の領域に、記述子(`descriptor`)の情報を格納する。`descriptor`は、A I Tサブテーブル内のすべてのアプリケーションに適用される。例えば、伝送プロトコル記述子が`descriptor`フィールドに書き込まれる。

【0129】

`application_loop_length`は、このMH A Iテーブルに含まれるアプリケーション情報の数を書き込む領域である。そして、`application_loop_length`が示す数分だけ、アプリケーション情報のループが配置される。

【0130】

1つのアプリケーション情報のループ内には、`application_identifier`(アプリケーション識別子)と、`application_control_code`(アプリケーション制御コード)と、`application_descriptor_loop_length`(アプリケーション情報記述子ループ長)の数分のループからなる一連の領域に記載される`descriptor`(アプリケーション情報記述子)が配置される。この記述子領域内の記述子は、指定したアプリケーションのみに適用される。

【0131】

`application_identifier`(アプリケーション識別子)は、アプリケーションを識別するパラメータである。`application_control_code`(アプリケーション制御コード)は、8ビットのフィールドで、アプリケーションの状態を制御する制御コードを規定する。このフィールドのセマンティックスは、アプリケーション形式の値に依存する。`application_control_code`として“`autostart`”が指示されていたら、このMH A Iテーブルを参照した受信機は、`application_identifier`で指定されたアプリケーションを起動開始する。また、`application_control_code`として“`prefetch`”が指示されていたら、このMH A Iテーブルを参照した受信機は、`application_identifier`で指定されたアプリケーションを先読みする。また、`application_control_code`として“`kill`”が指示されていたら、このMH A Iテーブルを参照した受信機は、`application_identifier`で指定されたアプリケーションの実行を停止する。CRC 32(CRC)、ITU-T勧告H.222.0に従う巡回冗長符号とする。

【0132】

要するに、MH A Iテーブルは、MMT伝送路で送られてくるアプリケーション(ファイル・データ)の処理方法や、伝送方法(`transport_protocol`)、ロケーション(URL)を指定するテーブルである。受信機は、M2セクション・メッセージで送られてくるMH A Iテーブルを受信すると、`application_control_code`で指定された処理を実行するために、指定されたロケーションから指定された`transport_protocol`でアプリケーションを取得する。

【0133】

図20には、MH A Iテーブルのアプリケーション情報のループ内に格納される、アプリケーション情報記述子2000の構成例を示している。また、図21には、アプリケーション情報記述子2000に含まれるパラメータの説明を示している。以下、アプリケーション情報記述子2000の各パラメータの意味について説明する。

【0134】

`descriptor_tag`は、当該記述子2000を識別する、8ビットの整数値である。`descriptor_length`は、このフィールドより後に続く当該記述子2000のデータのバイト数を書き込む領域である。

【0135】

`application_profile_length`の数分のループからなる一連

10

20

30

40

50

の領域には、`application_profile`の情報が書き込まれる。`application_profile`は、本アプリケーションが実行可能である受信機のプロファイルであり、受信機に要求する機能毎のビットマップで要求機能を示す。但し上位3ビットは機能ビットマップ切り替えを示す。上記ビットマップはバージョン毎に規定する。また、`version_major`、`version_minor`、`version_micro`はそれぞれ、アプリケーション・プロファイル規定のバージョンである。

【0136】

`service_bound_flag`は、本アプリケーションが現在のサービスのみで有効かどうかを示すフラグである。`visibility`は、アプリケーション可視か否かを示す。`application_priority`は、このサービス内で告知されているアプリケーション間の相対優先度である。`transport_protocol_label`は、アプリケーションを伝送するプロトコルを示す。`transport_protocol_label`の値としては、`0x0003`はHTTP/HTTPS伝送、`0x0005`はMMT並びにノンタイムド伝送を規定する。

10

【0137】

また、図22には、伝送プロトコル記述子2200の構成例を示している。以下、伝送プロトコル記述子2200の各パラメーターの意味について説明する。

【0138】

`descriptor_tag`は、当該記述子2200を識別する、8ビットの整数値である。`descriptor_length`は、このフィールドより後に続く当該記述子2200のデータのバイト数を書き込む、8ビットの領域である。

20

【0139】

`protocol_id`(プロトコル識別子)は、アプリケーションを伝送するプロトコルを示す。値としては、`0x0003`はHTTP/HTTPS伝送、`0x0005`はMMT並びにノンタイムド伝送を規定する。`transport_protocol_label`(伝送プロトコル・ラベル)は、1つのアプリケーションを複数の経路で伝送する場合にその伝送手段を一意に識別する値であり、アプリケーション情報記述子の同名のフィールドに対応する。`selector_byte`(セレクター・バイト)は、プロトコルID毎にシンタックスが規定される領域であり、取得場所が書き込まれる。

【0140】

図23には、HTTP/HTTPS、MMTノンタイムド伝送に共通のセレクター・バイト2300の構成例を示している。

30

【0141】

`url_base_byte`は、`url_base_length`の数分のループからなる一連の領域に、URL文字列のうち、`url_base`を示すテキスト情報を格納する。

【0142】

`url_extension_count`は、`url_base`に続く`url_extension`の数を示し、`url_extension_count`の数分だけ`url_extension`のループが配置される。そして、1つの`url_extension`のループ内では、`url_extention_byte`は、`url_extension`の長さを規定する`url_extention_length`の数分のループからなる一連の領域に、個々の`url_extention`を示すテキスト情報を格納する。各`url_extention`は、`url_base`に続くURL文字列である。例えば、`url_base`が“`http://www.xbc.com`”で、`url_extension`が“`index.html`”であれば、これらの文字列を連結して、完全なURL“`http://www.xbc.com/index.html`”を得ることができる。

40

【0143】

要するに、MH AIテーブルのアプリケーション情報のループ内のアプリケーション情報記述子並びに伝送プロトコル記述子を参照することで、アプリケーションの伝送手段

50

(M M T 伝送か、 H T M L 伝送か)、並びに、ロケーション情報 (U R L) を取得することができる。

【 0 1 4 4 】

図 2 4 には、シグナリング・メッセージの 1 つであるデータ・トランスミッション・メッセージ 2 4 0 0 の構成例を示している。以下、データ・トランスミッション・メッセージの各パラメーターの意味について説明する。

【 0 1 4 5 】

`message_id` (メッセージ識別) は、各種シグナリング情報において、データ・トランスミッション・メッセージを識別する 1 6 ビットの固定値であり、本実施形態では `0xF000` とする。`version` (バージョン) は、データ・トランスミッション・メッセージのバージョン番号を書き込む領域である。`length` (メッセージ長) は、このフィールドより後に続く当該メッセージのデータのサイズをバイト単位で示す、3 2 ビットのパラメーターである。

【 0 1 4 6 】

`num_of_tables` (テーブル数) は、このデータ・トランスミッション・メッセージに格納するテーブルの数を示す。データ・トランスミッション・メッセージに格納するテーブルとして、そして、`num_of_tables` が示す数分だけ、テーブル情報のループが配置される。

【 0 1 4 7 】

1 つのテーブル情報のループ内には、テーブル情報として、`table_id` (テーブル識別)、`table_version` (テーブル・バージョン)、並びに、`table_length` (テーブル長) が格納される。`table_id` (テーブル識別) は、このデータ・トランスミッション・メッセージに格納するテーブルの識別のための使用する領域である。データ・トランスミッション・メッセージでは、データ・アセット・マネジメント・テーブル (D A M T)、データ・ディレクトリー・マネジメント・テーブル (D D M T)、データ・コンテンツ・マネジメント・テーブル (D C M T) の 3 種類のシグナリング・テーブルが伝送されるが (前述)、`table_id` (テーブル識別子) はこれらのうちいずれのテーブルであるかを識別する。`table_version` (テーブル・バージョン) は、このデータ・トランスミッション・メッセージに格納するテーブルのバージョンを示す。`table_length` (テーブル長) は、このデータ・トランスミッション・メッセージに格納するテーブルの大きさをバイト単位で示す。

【 0 1 4 8 】

また、`num_of_tables` が示す数分だけ、テーブルのループが配置される。1 つのテーブルのループ内には、`table_id` で識別されるテーブルの中身の情報が格納される。`table` (テーブル) は、このデータ・トランスミッション・メッセージに格納するテーブルを示す。

【 0 1 4 9 】

図 2 5 には、データ・トランスミッション・メッセージで伝送されるデータ・アセット・マネジメント・テーブル 2 5 0 0 の構成例を示している。データ・アセット・マネジメント・テーブルは、M M T P パケットとして伝送されるファイル・データのアセットの情報と、ファイル・データの各アセットに含まれるアイテムの情報を管理するテーブルである。以下、このデータ・アセット・マネジメント・テーブルの各パラメーターの意味について説明する。

【 0 1 5 0 】

`table_id` (テーブル識別) は、各種シグナリング情報においてデータ・アセット・マネジメント・テーブルであることを示す 8 ビットの固定値であり、本実施形態では `0xA2` とする。`version` (バージョン) は、このデータ・アセット・マネジメント・テーブルのバージョンを示す 8 ビットの整数値のパラメーターである。例えばデータ・アセット・マネジメント・テーブルを構成する一部のパラメーターでも更新した場合には、`version` は + 1 だけインクリメントされる。`length` は、このフィール

10

20

30

40

50

ドの直後からカウントされる、このデータ・アセット・マネジメント・テーブルのサイズをバイト単位で示す、16ビット長のパラメーターである。

【0151】

`number_of_asset`は、パッケージに含まれるファイル・データのアセットの数を示す、8ビットのパラメーターである。`number_of_asset`の数分だけ、以下のアセット情報のループが配置され、アセット毎のファイル・データの情報が格納される。

【0152】

1つのアセット情報のループ内には、`download_id`と、アセット(ファイル・データ)自体に関する情報と、そのアセットに含まれる各アイテムに関する情報が含まれる。`download_id`は、ノンタイムド・メディア(ファイル・データ)を伝送するMMTPパケットの拡張ヘッダーに書き込まれる識別情報である(図7を参照のこと)。

【0153】

アセット情報のループ内に格納されるアセット自体に関する情報として、`asset_ID_scheme`、`asset_ID_length`、`asset_ID_length`と、`asset_ID_byte`を含む。`asset_ID_scheme`は、`asset_ID`の形式を示す。`asset_ID`の形式として、例えばUUID(Universal Unique Identifier)、URI(Uniform Resource Identifier)、GURL(General URL)を割り当てることができる。`asset_ID_length`は、`asset_ID_byte`の長さをバイト単位で表す。`asset_ID_byte`は、`asset_ID_length`の数分のループからなる一連の領域に、`asset_ID_scheme`で指定された形式で、`asset_ID`を示す。ちなみに、この情報は、本実施形態では、MPテーブル、データ・アセットマネジメント・テーブル共通にアセットを識別する情報として用いられるが、データ量が大きいので他の代用可能なアセット識別情報を用いてもよい。例えば、MPテーブルにおいて`asset_ID`に対応する情報として16ビットの`component_tag`を定義し、データ・アセット・マネジメント・テーブルにおいては`asset_ID`の代わりに`component_tag`を利用することが想定される。

【0154】

`number_of_items`は、該当するファイル・データのアセットを構成するアイテムの数を書き込む領域である。そして、`number_of_items`の数分だけアイテムのループが配置され、アセット(ファイル・データ)を構成する各アイテムに関する情報が書き込まれる。

【0155】

1つのアイテムのループ内には、アイテムに関する情報として、`item_ID`、`node_tag`、`item_size`、`item_version`、`item_checksum`、`item_info`の各パラメーターが記述される。`item_ID`は、ノンタイムドMFUで伝送されるアイテムを識別するIDを示す32ビットの値である。`node_tag`は、同様にアイテムを識別する情報であり、16ビットの値である。シグナリング情報としては、32ビットの`item_ID`に代えて16ビットの`node_tag`を使用することで、アイテムの識別に必要なビット・サイズを削減することができる。なお、`node`は、データ放送アプリケーション(コンテンツ)を構成するディレクトリー構造上のノードとなるディレクトリー並びにアイテムの各々を指す。アイテムだけでなくディレクトリーも`node_tag`で指定することができる。`item_size`は、アイテムのサイズをバイト単位で表す。`item_version`は、アイテムのバージョンを示し、アイテムの内容が更新される度に`version`は+1だけインクリメントされる。`item_checksum`は、アイテムのチェックサムを示す。なお、チェックサムは、すべてのファイルに対して必ず設定するのは情報量が多いと考えられる。よって、そのような考慮により、例えば1ビットの`checksum_flag`を設定し、こ

10

20

30

40

50

れに1が代入された場合にのみ32ビットの `item_check_sum` が現れるようにしてもよい。あるいは、シグナリングではなく、図7に示したMMTPパケットの拡張ヘッダーとして `type` としてチェックサムを示し、`length` の後に32ビットのチェックサムを配置してもよい。`item_info_length` は、`item_info_byte` の情報領域のサイズをバイト単位で表す。そして、`item_info_byte` は、`item_info_length` の数分のループからなる一連の領域に、当該アイテムに関する情報 (`item_info()`) を格納する。

【0156】

`descriptor_loop_length` は、`descriptor` の全バイト長を示す。`descriptor` は、`descriptor_loop_length` の数分のループからなる一連の領域に記述子の情報 (`descriptor()`) を格納する。格納される記述子は別途定義する。

10

【0157】

要するに、データ・アセット・マネジメント・テーブル2500は、1つのパッケージに含まれるファイル・データ(コンテンツ)のアセット並びにアセットに含まれるアイテムに関する情報を管理するテーブルである。アイテムに関する情報として、アイテムのバージョン情報も管理する。データ・アセット・マネジメント・テーブル2500を参照して、`node_tag` (若しくは、`Item_ID`) から該当する `asset_id` やアセットを伝送するMMT拡張ヘッダーに記載された `downloadID` や `item_info` を引いたり、シグナリング情報の伝送路上で扱う `node_tag` からファイル・データの伝送路上の `item_ID` や `item_info` を引いたりすることができる。

20

【0158】

図26には、データ・トランスミッション・メッセージで伝送されるデータ・ディレクトリー・マネジメント・テーブル(DDMT)2600の構成例を示している。データ・ディレクトリー・マネジメント・テーブルは、データ放送アプリケーション(コンテンツ)を構成するディレクトリー、並びに、ディレクトリーに含まれる各ノード(下位のディレクトリーやアイテム(ファイル・データ))のロケーション情報を管理するテーブルである。以下、このデータ・ディレクトリー・マネジメント・テーブルの各パラメータの意味について説明する。

【0159】

`table_id` (テーブル識別) には、各種シグナリング情報においてデータ・ディレクトリー・マネジメント・テーブルであることを示す8ビットの固定値が書き込まれる。`version_` (バージョン) は、このデータ・ディレクトリー・マネジメント・テーブルのバージョンを示す8ビットの整数値のパラメータである。例えばデータ・ディレクトリー・マネジメント・テーブルを構成する一部のパラメータでも更新した場合には、`version` は+1だけインクリメントされる。`length` は、このフィールドの直後からカウントされる、このデータ・ディレクトリー・マネジメント・テーブルのサイズをバイト単位で示す、16ビット長のパラメータである。

30

【0160】

`base_folder_path_length` は、`base_folder_path_byte` の情報領域のサイズをバイト単位で表す。`base_folder_path_byte` は、`base_folder_path_length` の数分のループからなる一連の領域に、`base_folder` (上位のディレクトリー) へのパス名を格納する。`base_folder_path_byte` は、例えば、対応するディレクトリーへアクセスするための絶対的なURL形式で表記される。

40

【0161】

`num_of_folder_nodes` は、データ・ディレクトリー・マネジメント・テーブルに記載されるフォルダー・ノードの数を示す。そして、`num_of_folder_nodes` の数分だけフォルダー・ノードのループが配置される。

【0162】

50

1つのフォルダー・ノードのループ内には、データ・ディレクトリー・マネジメント・テーブルに記載される各フォルダー・ノードの情報と、ベース・フォルダーに含まれる各ファイル・データの情報が格納される。

【0163】

`folder__node__path__length`は、`folder__node__path__byte`の情報領域のサイズをバイト単位で表す。`folder__node__path__byte`は、`folder__node__path__length`の数分のループからなる一連の領域に、`folder__node`へのパス名を格納する。`folder__node__path__byte`は、例えば、対応するディレクトリーへアクセスするための、`base__folder__path`からの相対的なURL形式で表記される。図示しないが、フォルダー・ノードの情報として`folder__node__version`(フォルダー・ノードのバージョン情報)を含んでいてもよい。例えば、`base__folder`のパス名(URL)が“`http://www.xbc.com`”で、ある`folder__node`のパス名(URL)が“`index.html`”であれば、これらの文字列を連結して、完全なURL“`http://www.xbc.com/index.html`”を得ることができる。

10

【0164】

`num__of__files`は、データ・ディレクトリー・マネジメント・テーブルに記載されるファイルの数を示す。そして、`num__of__files`の数分だけファイルのループが配置される。

20

【0165】

1つのファイルのループ内には、ベース・フォルダーに含まれる各ファイル・データの情報として、`node__tag`と、`file__name__byte`(ファイル名)が格納される。`node__tag`は、ノンタイムドMFUで伝送されるアイテムを識別する情報を、32ビットの`item__ID`よりも短い16ビットで表す。なお、`node`は、データ放送アプリケーション(コンテンツ)を構成するディレクトリー構造上のノードとなるディレクトリー並びにアイテムの各々を指す。アイテムだけでなくディレクトリーも`node__tag`で指定することができる(前述)。`file__name__byte`は、`file__name__length`の数分のループからなる一連の領域に格納される。

【0166】

要するに、データ・ディレクトリー・マネジメント・テーブル2600は、1つのパッケージに含まれるディレクトリー並びにディレクトリーに含まれるサブディレクトリーやファイル(アイテム)に関するディレクトリー構造を管理するテーブルである。データ・ディレクトリー・マネジメント・テーブル2600により、データ放送アプリケーションのファイル構成と、ファイル伝送のための構成を分離することができる。また、データ・ディレクトリー・マネジメント・テーブル2600を参照して、`node__tag`から該当するアイテムのパス名(URL)を引いたり、逆にパス名(URL)から該当する`node__tag`を引いたりすることができる。なお、本構成例では、データ・ディレクトリー・マネジメント・テーブルにおいて、ファイルが存在するディレクトリーのロケーション情報を`folder__path__byte`として設定するとともに各ディレクトリーを`node__tag`として識別情報を与え、アイテム毎の情報としてはファイル名と`node__tag`のみを指定するので、`folder__path__byte`の情報量が大きくなり過ぎるということはない。

30

40

【0167】

続いて、データ・トランスミッション・メッセージで伝送されるデータ・コンテンツ・マネジメント・テーブル(DCMT)について説明する。

【0168】

データ・コンテンツ・マネジメント・テーブル(DCMT)は、ノンタイムド・メディアとして伝送されるファイル・データすなわちコンテンツ(データ放送アプリケーション)の情報を管理するテーブルである。本実施形態では、データ・コンテンツ・マネジメン

50

ト・テーブルに強制キャッシュを指定する情報を含めて伝送する。放送番組に連動したデータ放送は、タイムリーな提示を行なうことが要求される。このような場合、放送送出システム11側からは、データ・コンテンツ・マネジメント・テーブル(DCMT)に強制キャッシュを指定する情報を含めることで、受信機12側では、放送番組に連動したデータ放送で利用される各ファイルをプリキャッシュしておき、データ放送のタイムリーな提示を実現することができる。

【0169】

データ・コンテンツ・マネジメント・テーブル(DCMT)による強制キャッシュ情報の伝送方式として、以下の2通りを挙げることができる。

【0170】

(方式1)データ・コンテンツ・マネジメント・テーブルにおいて、データ放送提示単位(PU)毎に、提示単位を構成する放送伝送ファイル(member item)リストと中心となるファイル(primary item)、及びプリキャッシュの対象ファイルが存在する場合はその対象ファイル・リスト(pre-cache item)の情報を記述する。

【0171】

ここで言うプリキャッシュの対象ファイルは、例えば、現在のデータ放送提示単位(PU)から次に参照するデータ放送提示単位(PU)を構成する放送ファイル・リストで構成される。放送番組の制作側では、このようにデータ放送のシグナリング情報でプリキャッシュの対象ファイル・リストを提示することで、受信機側では次に遷移するデータ放送提示単位に必要なファイル・データをプリキャッシュしておくという、プリキャッシュによるキャッシュ制御動作を行なうことができる。その結果、放送番組に連動したタイムリーなデータ放送サービスを実現することができる。

【0172】

(方式2)データ・コンテンツ・マネジメント・テーブルにおいて、データ放送提示単位(PU)毎に、提示単位を構成する放送伝送ファイル(member item)リストと中心となるファイル(primary item)、及びキャッシュにロックする対象ファイル(lock cache item)及びロック対象のうちアンロックする対象ファイル(unlock cache item)の情報を記述する。

【0173】

ここで言うロック対象ファイルは、例えば、現在のデータ放送提示単位(PU)から次に参照するデータ放送提示単位(PU)で使用する放送ファイル・リストで構成される。放送番組の制作側では、このようにデータ放送のシグナリング情報でロック並びにアンロックの対象ファイル・リストを提示することで、受信機側ではロック並びにアンロックによるキャッシュ制御動作を行なうことができる。例えば、受信機側では、次に遷移するデータ放送提示単位に必要なファイル・データをキャッシュにロックしておくことができる。その結果、放送番組に連動したタイムリーなデータ放送サービスを実現することができる。また、アンロック対象ファイルは、例えば、現在のデータ放送提示単位(PU)では不要となる放送ファイル・リストで構成される。アンロック対象ファイルを指定することで、不要なファイルをキャッシュ・メモリー408から削除することができ、メモリー・サイズを節約することができる。

【0174】

図27には、方式1を実現するデータ・トランスミッション・メッセージで伝送されるデータ・コンテンツ・マネジメント・テーブル(DCMT)2700の構成例を示している。

【0175】

table__id(テーブル識別)には、各種シグナリング情報においてデータ・コンテンツ・マネジメント・テーブルであることを示す8ビットの固定値が書き込まれる。version__(バージョン)は、このデータ・コンテンツ・マネジメント・テーブルのバージョンを示す8ビットの整数値のパラメータである。例えばデータ・コンテンツ・マ

10

20

30

40

50

ネジメント・テーブルを構成する一部のパラメーターでも更新した場合には、`version`は+1だけインクリメントされる。`length`は、このフィールドの直後からカウントされる、このデータ・コンテンツ・マネジメント・テーブルのサイズをバイト単位で示す、16ビット長のパラメーターである。

【0176】

`number_of_content`は、パッケージに含まれるコンテンツの数を示す、8ビットのパラメーターである(コンテンツは、例えば、データ放送アプリケーションを記述したHTML文書などのファイル・データである)。`number_of_content`の数分だけ、以下のコンテンツのループが配置され、コンテンツ毎の情報が格納される。

10

【0177】

1つのコンテンツのループ内には、コンテンツに関する情報として、`content_ID`と、`content_version`と、`content_cache_size`と、当該コンテンツに含まれるデータ放送提示単位(Presentation Unit: PU)に関する情報が書き込まれる。`content_ID`は、コンテンツの識別情報である。`content_version`は、コンテンツのバージョンを示す。`content_cache_size`は、コンテンツをキャッシュするサイズを示す。

【0178】

`number_of_PU`は、コンテンツに含まれるデータ放送提示単位PUの数であり、`number_of_PU`の数分だけPUのループが配置される。

20

【0179】

1つのPUのループ内には、PUの識別情報である`PU_tag`と、PUをキャッシュするサイズを示す`PU_cache_size`と、当該データ放送提示単位(PU)の中心となるファイル(primary item)を識別する`PU_primary_item_node_tag`が書き込まれる。

【0180】

また、PUのループ内には、当該データ放送提示単位(PU)を構成する放送伝送ファイル(member item)リストが書き込まれる。具体的には、`number_of_PU_member_nodes`は、当該データ放送提示単位(PU)に含まれる(すなわち、PUのメンバーとなる)ノードの数を示す。その後、この`number_of_PU_member_nodes`の数分だけのPUメンバー・ノードのループが配置され、各PUメンバー・ノードのループ内には、PUメンバー・ノードの`node_tag`が書き込まれる。PUメンバー・ノードは、ディレクトリーのノードとアイテムのノードを含む。

30

【0181】

また、PUのループ内には、該当するPUにおいてプリキャッシュの対象ファイルが存在する場合はその対象ファイル・リスト(pre-cache item)の情報を記述する。具体的には、`number_of_pre_cache_nodes`はプリキャッシュの対象となるノードの数であり、`number_of_pre_cache_nodes`の数分だけのプリキャッシュ・ノードのループが配置される。1つのプリキャッシュ・ノードのループ内には、プリキャッシュ・ノードを識別する`pre_cache_node_tag`が書き込まれる。

40

【0182】

ここで言うプリキャッシュの対象ファイルは、例えば、現在のデータ放送提示単位(PU)から次に参照するデータ放送提示単位(PU)を構成する放送ファイル・リストで構成される。放送番組の制作側では、このようにデータ放送のシグナリング情報でプリキャッシュの対象ファイル・リストを提示することで、受信機側では次に遷移するデータ放送提示単位に必要なファイル・データをプリキャッシュしておくことができる。その結果、放送番組に連動したタイムリーなデータ放送サービスを実現することができる。

【0183】

50

また、1つのPUのループ内には、このPUからリンクされる他のPUの数を示す `number_of_linked_PU` と、 `number_of_linked_PU` の数分だけの `linked_PU` のループが配置される。1つの `linked_PU` のループ内では、 `linked_PU` の識別情報である `linked_PU_tag` が書き込まれる。

【0184】

図28には、方式2を実現するデータ・トランSMIッション・メッセージで伝送されるデータ・コンテンツ・マネジメント・テーブル(DCMT)2800の構成例を示している。

【0185】

`table_id` (テーブル識別)には、各種シグナリング情報においてデータ・コンテンツ・マネジメント・テーブルであることを示す8ビットの固定値が書き込まれる。`version` (バージョン)は、このデータ・コンテンツ・マネジメント・テーブルのバージョンを示す8ビットの整数値のパラメータである。例えばデータ・コンテンツ・マネジメント・テーブルを構成する一部のパラメータでも更新した場合には、`version` は+1だけインクリメントされる。`length` は、このフィールドの直後からカウントされる、このデータ・コンテンツ・マネジメント・テーブルのサイズをバイト単位で示す、16ビット長のパラメータである。

【0186】

`number_of_content` は、パッケージに含まれるコンテンツの数を示す、8ビットのパラメータである(コンテンツは、例えば、データ放送アプリケーションを記述したHTML文書などのファイル・データである)。`number_of_content` の数分だけ、以下のコンテンツのループが配置され、コンテンツ毎の情報が格納される。

【0187】

1つのコンテンツのループ内には、コンテンツに関する情報として、`content_ID` と、`content_version` と、`content_cache_size` と、当該コンテンツに含まれるデータ放送提示単位(Presentation Unit: PU)に関する情報が書き込まれる。`content_ID` は、コンテンツの識別情報である。`content_version` は、コンテンツのバージョンを示す。`content_cache_size` は、コンテンツをキャッシュするサイズを示す。

【0188】

`number_of_PU` は、コンテンツに含まれるデータ放送提示単位PUの数であり、`number_of_PU` の数分だけPUのループが配置される。

【0189】

1つのPUのループ内には、PUの識別情報である `PU_tag` と、PUをキャッシュするサイズを示す `PU_cache_size` と、当該データ放送提示単位(PU)の中心となるファイル(primary item)を識別する `PU_primary_item_node_tag` が書き込まれる。

【0190】

また、PUのループ内には、当該データ放送提示単位(PU)を構成する放送伝送ファイル(member item)リストが書き込まれる。具体的には、`number_of_PU_member_nodes` は、当該データ放送提示単位(PU)に含まれる(すなわち、PUのメンバーとなる)ノードの数を示す。その後、この `number_of_PU_member_nodes` の数分だけのPUメンバー・ノードのループが配置され、各PUメンバー・ノードのループ内には、PUメンバー・ノードの `node_tag` が書き込まれる。PUメンバー・ノードは、ディレクトリーのノードとアイテムのノードを含む。

【0191】

また、PUのループ内には、該当するPUにおいて、キャッシュ・メモリー408への

10

20

30

40

50

キャッシュをロックする対象となるファイル (`lock_cache_item`) 及びロック対象のうちアンロックする対象となるファイル (`unlock_cache_item`) の情報を記述する。具体的には、`number_of_lock_cache_nodes` はプリキャッシュの対象となるノードの数であり、その後 `number_of_lock_cache_nodes` の数分だけのロック対象ノードのループが配置される。1つのロック対象ノードのループ内には、ロック対象ノードを識別する `lock_cache_node_tag` が書き込まれる。また、`number_of_unlock_cache_nodes` はプリキャッシュの対象となるノードの数であり、その後 `number_of_unlock_cache_nodes` の数分だけのアンロック対象ノードのループが配置される。1つのアンロック対象ノードのループ内には、アンロック対象ノードを識別する `unlock_cache_node_tag` が書き込まれる。

10

【0192】

ここで言うロック対象ファイルは、例えば、現在のデータ放送提示単位 (PU) から次に参照するデータ放送提示単位 (PU) で使用する放送ファイル・リストで構成される。放送番組の制作側では、このようにデータ放送のシグナリング情報でロック対象ファイル・リストを提示することで、受信機側では次に遷移するデータ放送提示単位に必要なファイル・データをキャッシュにロックしておくことができる。その結果、放送番組に連動したタイムリーなデータ放送サービスを実現することができる。また、アンロック対象ファイルは、例えば、現在のデータ放送提示単位 (PU) では不要となる放送ファイル・リストで構成される。アンロック対象ファイルを指定することで、不要なファイルをキャッシュ・メモリー 408 から削除することができ、メモリー・サイズを節約することができる。

20

【0193】

また、1つのPUのループ内には、このPUからリンクされる他のPUの数を示す `number_of_linked_PU` と、`number_of_linked_PU` の数分だけの `linked_PU` のループが配置される。1つの `linked_PU` のループ内では、`linked_PU` の識別情報である `linked_PU_tag` が書き込まれる。

【0194】

要するに、データ・コンテンツ・マネジメント・テーブルは、1つのパッケージで各コンテンツ (データ放送アプリケーション) をデータ放送提示単位 (PU) で管理するテーブルであり、データ・コンテンツ・マネジメント・テーブルを参照して、`node_tag` から、そのノードを含むデータ放送提示単位の `PU_tag` を取得することができる。また、データ・コンテンツ・マネジメント・テーブル受信機 12 側でのデータ放送用ファイルの強制キャッシュを制御するという側面を持つ。但し、データ・コンテンツ・マネジメント・テーブルを利用したキャッシュ制御動作の詳細については、後述に譲る。

30

【0195】

図 29 には、MMT 伝送されるデータ放送アプリケーション (コンテンツ) の伝送、コンテンツのロケーションと、アプリケーションの提示を行なう仕組みを図解している。

【0196】

図 29 (A) には、コンテンツのディレクトリー構造を示している。各コンテンツ `content 1, 2, ...` は、データ放送アプリケーション (`app`) と材料で構成される。データ放送アプリケーションや材料は、それぞれファイル・データを実体とするリソースである。各リソースは、MMT 伝送路上ではアセットの構成要素であるアイテムに相当し、32ビットの `item_ID` で識別することができる。また、シグナリング情報内では、アイテムは16ビットの `node_tag` で識別することができる。図 29 (C) に示すように、各リソースは、該当するアセットのMMT伝送路上でアイテムとして伝送される (後述)。アプリケーションは、コンテンツの実行時 (データ放送の提示時) において参照される1以上のHTML文書からなる。また、材料は、HTML文書から参照される `jpeg` 画像やテキストなどのモノメディア・データなどである。1

40

50

つのHTML文書と、そこから参照されるマテリアルで、1つのデータ放送提示単位PUを構成する。図29(A)に示す例では、content1は、A11.html、A12.html、A13.htmlなどの1以上のHTML文書をデータ放送アプリケーションのリソースとして持つ。このうち、A11.htmlは、コンテンツの実行時に直接参照されるリソースとする。

【0197】

図29(B)には、コンテンツの実行時(データ放送の提示時)におけるリソース間の参照関係を示している。図示の例では、コンテンツの実行時に直接参照されるアプリケーションA11とこれが参照するマテリアルB11、B02が1つのデータ放送提示単位PUを構成するリソース・グループ2801であり、PU_tagとしてp1が割り当てられている(なお、B14は、放送によりMMT伝送されるのではなく通信によるHTTP伝送で随時取得することができるマテリアルであり、以下では、データ放送提示単位のリソース・グループには含まないものとして扱う)。

10

【0198】

同様に、アプリケーションA12とこれが参照するマテリアルB12、B02、B13が1つのデータ放送提示単位PUを構成するリソース・グループ2802であり、PU_tagとしてp2が割り当てられている(なお、B07は、放送によりMMT伝送されるのではなく通信によるHTTP伝送で随時取得することができるマテリアルであり、以下では、データ放送提示単位のリソース・グループには含まないものとして扱う)。同様に、アプリケーションA01とこれが参照するマテリアルB03、B01、B04が1つのデータ放送提示単位PUを構成するリソース・グループ2803であり、PU_tagとしてp3が割り当てられている。

20

【0199】

また、複数のHTML文書間でリンク参照関係を持つことができる(周知)。図29(B)に示す例では、リソースA11.htmlは、コンテンツの実行時に直接参照され、最初に表示されるアプリケーション提示画面を記述するHTML文書である。これに対し、同じcontent1に含まれリソースA12.htmlと、content1外のcommonに含まれるリソースA01.htmlは、A11.htmlを実行して提示される画面から遷移するアプリケーション提示画面を記述するHTML文書であり、A11.htmlとリンク参照関係を持つ。各リソースA11.html、A12.html、A01.htmlは、それぞれ1つのデータ放送提示単位PUを構成するリソース・グループ2801、2802、2803を形成する。そして、リンクし合うデータ放送提示単位2801、2802、2803同士で、さらに上位の大きなリソース・グループ2810を構成する。リソースA11.html、A12.html、A01.htmlは、各々のデータ放送提示単位PUにおいて中心となるファイル・データ(primary item_node)である。

30

【0200】

また、パッケージ(1つの放送番組)に含まれるアプリケーション全体となるさらにコンテンツ全体で大きなリソース・グループすなわちデータ・コンテンツ全体を構成する。データ・コンテンツ全体とは、共通のcontent_IDを持つデータ放送提示単位PUの範囲である。データ・コンテンツ・マネジメントテーブルで、該当するcontent_IDのPUのループを回すことにより、コンテンツに含まれるすべてのデータ放送提示単位PUを一括して特定することができる。図29(B)に示す例では、content1とcommonに含まれるアプリケーションでコンテンツ全体のリソース・グループ2820を形成している。

40

【0201】

図29(C)には、コンテンツをMMT伝送する様子を模式的に示している。コンテンツの構成要素であるアプリケーションやマテリアルは、それぞれファイル・データが実体であり、「リソース」とも呼ぶ。各リソースは、MMT伝送路上ではアセットの構成要素であるアイテムに相当する。MMT伝送では、パッケージに含まれる各コンテンツは1つ

50

のアセットとして扱われ、それぞれ `Asset_ID` が割り当てられる。図示の例では、`content1` には `asset_ID` として `a1` が割り当てられている。また、MMT 伝送では、HTML 文書データやマテリアルなどの個々のリソースは、1つのアイテムとして扱われ、それぞれ `Item_ID` が割り当てられる。図示の例では、`content1` に含まれる各リソースには、それぞれ `Item_ID` として `i11`、`i12`、`i13`、`i14` が割り当てられている。

【0202】

また、同じコンテンツに含まれるリソースは同じ `asset_ID` を共有し、同じ MMT 伝送路上で伝送される。図 29 (C) に示す例では、`Item_ID` が `i11`、`i12`、`i13`、`i14` の各アイテムは、同じ `Asset_ID` として `a1` を共有しており、
10

前述したデータ・ディレクトリー・マネジメント・テーブルは図 29 (A) で表現され、データ・コンテンツ・マネジメント・テーブルは図 29 (B) で表現され、データ・アセット・マネジメント・テーブルは図 29 (C) で表現され、これらの間を `item_ID` 若しくは `node_tag` により関係付けられることになる。

【0203】

MMT 伝送路からデータ放送アプリケーション(コンテンツ)を取得する際の、シグナリング情報として伝送される各テーブルの参照関係について、図 30 を参照しながら説明する。

【0204】

受信機 12 は、M2 セクション・メッセージで、MH-AI テーブル (MH-AIT) 2901 を取得すると、`application_control_code` を参照して、アプリケーションの状態がどのように制御されているかを確認する。そして、“`autostart`” が指示されている場合には、テーブル内の `transport_protocol_label` を参照して、MMT 伝送が指定されていることを確認すると、このアプリケーションの提示時に直接参照されるアイテム(ファイル・データ)の URL 情報を伝送プロトコル記述子から取り出す。そして、受信機は、データ・トランスミッション・メッセージで送られてくるデータ・ディレクトリー・マネジメント・テーブル (DDMT) 2902 を参照して、その `base_folder_path_byte`、`folder_node_path_byte`、及び `file_name_byte` の組み合わせ
30

【0205】

次いで、受信機 12 は、データ・トランスミッション・メッセージで送られてくるデータ・アセット・マネジメント・テーブル (DAMT) 2903 を参照して、取得した `node_tag` を MMT 伝送路上の `item_ID` に戻すとともに、対応するアセットを特定して、その `asset_ID` と `download_id` を取得する。

【0206】

そして、受信機は、PA メッセージで送られてくる MP テーブル (MPT) 2904 を参照して、取得した `asset_ID` に対応する `packet_id` を取得すると、ファイル・データの MMT 伝送路上で、MMTP パケットのヘッダー内の `packet_id` と、拡張ヘッダー内の `download_id` と、DU ヘッダー内の `item_ID` に基づいてフィルタリングして、所望する(アプリケーションの提示時に直接参照する)アイテムを取得することができる。
40

【0207】

また、受信機 12 は、データ・トランスミッション・メッセージで送られてくるデータ・コンテンツ・マネジメント・テーブル (DCMT) 2905 内で、データ・ディレクトリー・マネジメント・テーブル 2902 から取得した `node_tag` を引いて、該当するアプリケーション提示単位の `PU_tag` を取り出すことができる。また、この `PU_tag` の `PU` のループ内で `linked_PU` のループを回すことにより、これにリンクする他のアプリケーション提示単位の `PU_tag` を一括して取り出すことができる。
50

【0208】

図31には、受信機内で、データ放送に利用されるファイル・データをキャッシュする仕組みを模式的に示している。ここで言うキャッシュには、実行するファイル・データをキャッシュすることとプリキャッシュすることを含む。

【0209】

アプリケーション・データ制御部407は、デマルチプレクサー402で放送ストリームからデマルチプレクスされたシグナリング・メッセージを解析して、受信機内の動作を制御する。コンテンツのプリキャッシュに関しては、アプリケーション・データ制御部407は、データ・コンテンツ・マネジメント・テーブルに含まれている強制キャッシュ情報に基づいて、放送番組に連動したデータ放送で利用される各ファイルをプリキャッシュする。

10

【0210】

具体的には、アプリケーション・データ制御部407は、データ・トランスミッション・メッセージで伝送されるデータ・コンテンツ・マネジメント・テーブルでプリキャッシュすることが指定されたノード(ディレクトリー、又はファイル・データ)の`cache_node_tag`を取得する。`node_tag`から該当するMMTPパケットを特定できることは、図30を参照しながら説明した通りである。

【0211】

アプリケーション・データ制御部407は、プリキャッシュしたいファイル・データの`node_tag`を、データ・トランスミッション・メッセージで伝送されるデータ・アセット・マネジメント・テーブルで引いて、そのノードが属するアセットの`asset_ID`を取得し、次いで、`asset_ID`をPAメッセージで伝送されるMPテーブルで引いて、アセットが伝送されるMMTPパケットの`packet_id`を取得する。また、システム制御部408は、データ・アセット・マネジメント・テーブルから、所望するアイテムを伝送するMMTPパケットの拡張ヘッダーに記載される`download_id`を取得すると、ファイル・データのMMT伝送路上で、MMTPパケットのヘッダー内の`packet_id`と、拡張ヘッダー内の`download_id`と、DUヘッダー内の`item_ID`に基づいてフィルタリングして、所望するアイテムのエンティティを取得して、キャッシュ・メモリー408にプリキャッシュする。

20

【0212】

データ放送アプリケーション・エンジン409は、アプリケーションを実行する際、必要なアイテム(ファイル・データ)が既にキャッシュ・メモリー408に事前にキャッシュされていれば、デマルチプレクサー402で放送ストリームからデマルチプレクスされたファイル・データが届くのを待つことなく、キャッシュ・メモリー408から取り出して、迅速に応答して、データ放送用表示信号を生成することができる。一方、必要なアイテムがキャッシュ・メモリー408内に存在しないときには、データ放送アプリケーション・エンジン407は、放送ストリームからデマルチプレクスされたファイル・データが届くのを待って応答して、データ放送用表示信号を生成する。

30

【0213】

続いて、受信機12において、データ放送で利用されるファイル・データをキャッシュする制御動作について詳解する。

40

【0214】

データ・コンテンツ・マネジメント・テーブル(DCMT)による強制キャッシュ情報の伝送方式として、方式1及び方式2の2通りがあることは既に述べた。まず方式1を利用したキャッシュ制御動作について説明する。

【0215】

方式1では、放送送出システム11側からは、図27に示したデータ・コンテンツ・マネジメント・テーブル(DCMT)2700がデータ・トランスミッション・メッセージで伝送される。データ・コンテンツ・マネジメント・テーブル(DCMT)2700は、データ放送提示単位(PU)毎に、提示単位を構成する放送伝送ファイル(member

50

item) リストと中心となるファイル(primary item)、及びプリキャッシュの対象ファイルが存在する場合はその対象ファイル・リスト(pre-cache item)の情報を記述する。したがって、受信機12側では、以下に示すような、プリキャッシュによるキャッシュ制御動作を行なうことができる。

【0216】

(動作01) アプリケーション・データ制御部407は、MMT伝送路504上で伝送されるデータ・トランスミッション・メッセージを適宜検出して更新を行ないつつ、最新の情報を取得する。

(動作02) アプリケーション・データ制御部407が、データ放送アプリケーション制御エンジン409などからの指示によりprimary_itemに指定された(データ放送提示単位の中心となる)アプリケーション・ファイルにアクセスすることにより、対応するデータ放送提示単位(PU)の提示状態に入ったことを認識する。

(動作03) アプリケーション・データ制御部407は、データ・トランスミッション・メッセージに含まれるデータ・コンテンツ・マネジメント・テーブル(DCMT)において、該当するデータ放送提示単位(PU)に含まれる各メンバー・ファイル(PU_member_node)も同時に取得して、キャッシュ・メモリー408に保持する。

(動作04) さらに、データ・コンテンツ・マネジメント・テーブル内で、該当するデータ放送提示単位(PU)に対してプリキャッシュの対象が指定されている場合には、アプリケーション・データ制御部407は、プリキャッシュ対象の各ファイル(item)も取得して、キャッシュ・メモリー408にプリキャッシュする。

(動作05) データ放送アプリケーション・エンジン409は、primary_itemに指定されたアプリケーション・ファイルを、キャッシュ・メモリー408から実行する。

(動作06) その後、データ・コンテンツ・マネジメント・テーブル(DCMT)の更新により、現在提示状態にあるデータ放送提示単位(PU)に含まれるメンバー・ファイル(PU_member_node)やプリキャッシュ対象のファイルの構成が変化した場合、又は、データ放送アプリケーション・エンジン409におけるアプリケーション動作により他のデータ放送提示単位(PU)の提示状態に遷移した場合などには、現在提示状態にあるデータ放送提示単位(PU)において、上記の(動作03)~(動作05)の処理を行なう。以前の状態でファイルをキャッシュ・メモリー408に保持していても、アプリケーション・データ制御部407は、現在の状態で不要なファイルをキャッシュ・メモリー408から削除する。

【0217】

図32には、受信機12における方式1に基づくファイル・データのキャッシュ制御手順をフローチャートの形式で示している。

【0218】

データ放送アプリケーション制御エンジン409がアプリケーション動作すなわちデータ放送の提示を行なっているときに(ステップS3201のYes)、アプリケーション・データ制御部407は、データ・コンテンツ・マネジメント・テーブル内でprimary_item(データ放送提示単位の中心)に指定されたアプリケーション・ファイルにアクセスすることにより(ステップS3202のYes)、対応するデータ放送提示単位(PU)の提示状態に入ったことを認識する。

【0219】

アプリケーション・データ制御部407は、キャッシュ・メモリー408をリセットして、データ・コンテンツ・マネジメント・テーブル(DCMT)において、現在提示しているデータ放送提示単位(PU)のprimary_item並びに各メンバー・ファイル(PU_member_node)を取得して、キャッシュ・メモリー408に保持する(ステップS3203)。データ放送アプリケーション・エンジン409は、primary_itemに指定されたアプリケーション・ファイルを、キャッシュ・メモリー408から実行する。

【 0 2 2 0 】

また、アプリケーション・データ制御部 4 0 7 は、データ・コンテンツ・マネジメント・テーブル内で、該当するデータ放送提示単位 (P U) に対してプリキャッシュの対象が指定されているかどうかをチェックする (ステップ S 3 2 0 4)。そして、プリキャッシュの対象が指定されている場合には (ステップ S 3 2 0 4 の Y e s)、アプリケーション・データ制御部 4 0 7 は、プリキャッシュ対象の各ファイル (i t e m) も取得して、キャッシュ・メモリー 4 0 8 にプリキャッシュする (ステップ S 3 2 0 5)。

【 0 2 2 1 】

その後、データ・コンテンツ・マネジメント・テーブル (D C M T) の更新により、現在提示状態にあるデータ放送提示単位 (P U) に含まれるメンバー・ファイル (P U _ m e m b e r _ n o d e) やプリキャッシュ対象のファイルの構成が変化した場合には (ステップ S 3 2 0 6 の Y e s)、ステップ S 3 2 0 3 に戻り、現在提示状態にあるデータ放送提示単位 (P U) において、上記の処理を繰り返し実行する。

10

【 0 2 2 2 】

また、データ放送アプリケーション・エンジン 4 0 9 におけるアプリケーション動作により他のデータ放送提示単位 (P U) の提示状態に遷移し、そのデータ放送提示単位の p r i m a r y _ i t e m に指定されたアプリケーション・ファイルにアクセスした場合には (ステップ S 3 2 0 7 の Y e s)、ステップ S 3 2 0 3 に戻り、遷移した先のデータ放送提示単位 (P U) において、上記の処理を繰り返し実行する。

【 0 2 2 3 】

以上の処理を、データ放送アプリケーション・エンジン 4 0 9 がアプリケーション動作を終了するまで (ステップ S 3 2 0 8 の N o)、繰り返し実行する。

20

【 0 2 2 4 】

図 3 3 には、受信機 1 2 における方式 1 に基づくファイル・データのプリキャッシュ動作例を示している。

【 0 2 2 5 】

アプリケーション・データ制御部 4 0 7 は、MMT 伝送路 5 0 4 上で受信する各種シグナリング・メッセージを解析している。シグナリング・メッセージの 1 つであるデータ・トランスミッション・メッセージには、データ・コンテンツ・マネジメント・テーブル (D C M T) が含まれている。

30

【 0 2 2 6 】

データ放送アプリケーション制御エンジン 4 0 9 が P U _ i d = 1 で識別されるデータ放送提示単位 (P U) のアプリケーション動作を行なっているとき、アプリケーション・データ制御部 4 0 7 は、データ・コンテンツ・マネジメント・テーブルを参照して、現在提示状態にあるデータ放送提示単位 (P U _ i d = 1) の p r i m a r y _ i t e m (データ放送提示単位の中心) に指定されたアプリケーション・ファイル「 A 0 1 . h t m l 」、並びに、各メンバー・ファイル (P U _ m e m b e r _ n o d e) 「 B 0 1 」、「 B 0 2 」それぞれの n o d e _ t a g をデータ・コンテンツ・マネジメント・テーブルから取得して、データ・アセットを伝送する MMT 伝送路 5 0 3 からこれらの n o d e _ t a g に対応するファイル・データのエンティティを取得すると、参照番号 3 3 0 1 で示すように、キャッシュ・メモリー 4 0 8 にキャッシュする。なお、 n o d e _ t a g から MMT 伝送路 5 0 3 上で伝送されるファイルにアクセスする方法については、図 3 0 を参照しながら、既に説明した通りである (以下、同様)。但し、このデータ放送提示単位 (P U) に対してプリキャッシュの対象が指定されていないので、プリキャッシュ動作は行なわない。

40

【 0 2 2 7 】

次いで、データ放送アプリケーション・エンジン 4 0 9 におけるアプリケーション動作により、 P U _ i d = 3 で識別される他のデータ放送提示単位 (P U) の提示状態に遷移したとする。アプリケーション・データ制御部 4 0 7 は、上記と同様に、遷移した先のデータ放送提示単位 (P U _ i d = 3) の p r i m a r y _ i t e m (データ放送提示単位

50

の中心)に指定されたアプリケーション・ファイル「A11.html」、並びに、各メンバー・ファイル(PU_member_node)「B11」、「B12」、「B13」それぞれのnode_tagをデータ・コンテンツ・マネジメント・テーブルから取得すると、MMT伝送路503からこれらのnode_tagに対応するファイル・データのエンティティを取得して、参照番号3302で示すように、キャッシュ・メモリー408にキャッシュする。但し、このデータ放送提示単位(PU)に対してプリキャッシュの対象が指定されていないので、プリキャッシュ動作は行なわない。

【0228】

アプリケーション・データ制御部407は、MMT伝送路504上で受信する各種シグナリング・メッセージを常に解析している。そして、参照番号3303で示すように、データ・コンテンツ・マネジメント・テーブルのバージョンが1から2に更新されたことを検出すると、アプリケーション・データ制御部407は、現在提示状態にあるデータ放送提示単位(PU_id=3)のprimary_item並びに各メンバー・ファイル、プリキャッシュ対象のファイルに変更がないかどうかをチェックする。今回はメンバー・ファイルに変更がないので、以前キャッシュしておいた各メンバー・ファイルはキャッシュ・メモリー408に保持したままとする。また、データ放送提示単位(PU_id=3)のプリキャッシュ対象のファイルとして「A12」、「B14」、「B15」が追加されているので、アプリケーション・データ制御部407は、MMT伝送路503からこれらのnode_tagに対応するファイル・データのエンティティを取得して、参照番号3304で示すように、キャッシュ・メモリー408にキャッシュする。

【0229】

次いで、参照番号3305で示すように、データ放送の提示の更新を指示するイベント・メッセージをMMT伝送路504から受信すると、データ放送アプリケーション・エンジン409は、実行中のA11ファイルからA12ファイルへのHTML文書の遷移を行なう。一方でほぼ同時に、アプリケーション・データ制御部407は、バージョンが2から3に更新されたデータ・コンテンツ・マネジメント・テーブルを参照して、現在提示状態にあるデータ放送提示単位(PU_id=3)のprimary_itemが「A12.html」に変更するとともに、メンバー・ファイルが「B14」、「B15」、「B16」に変更したことを検出する。上述したように、primary_item「A12.html」並びにメンバー・ファイルが「B14」、「B15」は既にキャッシュ・メモリー408にプリキャッシュされているので、データ放送アプリケーション・エンジン409は、キャッシュ・メモリー408に保持されているファイルを利用してデータ放送を迅速に表示することができる。すなわち、放送番組に連動したタイムリーなデータ放送の提示を行なうことができる。また、データ放送提示単位(PU_id=3)のプリキャッシュ対象のファイルとして「B12」が追加されているので、アプリケーション・データ制御部407は、MMT伝送路503からそのnode_tagに対応するファイル・データのエンティティを取得して、参照番号3306で示すように、キャッシュ・メモリー408にキャッシュする。

【0230】

続いて、方式2を利用したキャッシュ制御動作について説明する。

【0231】

方式2では、放送送出システム11側からは、図28に示したデータ・コンテンツ・マネジメント・テーブル(DCMT)2800がデータ・トランスミッション・メッセージで伝送される。データ・コンテンツ・マネジメント・テーブル(DCMT)2800は、データ放送提示単位(PU)毎に、提示単位を構成する放送伝送ファイル(member_item)リストと中心となるファイル(primary_item)、及びキャッシュにロックする対象ファイル(lock_cache_item)及びロック対象のうちアンロックする対象ファイル(unlock_cache_item)の情報を記述する。したがって、受信機12側では、以下に示すような、キャッシュのロック並びにアンロックによるキャッシュ制御動作を行なうことができる。

【0232】

(動作11) アプリケーション・データ制御部407は、MMT伝送路504上で伝送されるデータ・トランスミッション・メッセージを適宜検出して更新を行ないつつ、最新の情報を取得する。

(動作12) アプリケーション・データ制御部407が、データ放送アプリケーション制御エンジン409などからの指示により `primary_item` に指定された(データ放送提示単位の中心となる)アプリケーション・ファイルにアクセスすることにより、対応するデータ放送提示単位(PU)の提示状態に入ったことを認識する。

(動作13) アプリケーション・データ制御部407は、データ・トランスミッション・メッセージに含まれるデータ・コンテンツ・マネジメント・テーブル(DCMT)において、該当するデータ放送提示単位(PU)に含まれる各メンバー・ファイル(`PU_member_node`)も同時に取得して、キャッシュ・メモリー408に保持する。

(動作14) さらに、データ・コンテンツ・マネジメント・テーブル内で、該当するデータ放送提示単位(PU)に対してロックキャッシュの対象が指定されている場合には、アプリケーション・データ制御部407は、ロックキャッシュ対象のうち未取得の各ファイル(`item`)も取得して、キャッシュ・メモリー408にキャッシュし、且つ、ロックキャッシュ対象ファイルとして別途管理する。

(動作15) 逆に、データ・コンテンツ・マネジメント・テーブル内で、該当するデータ放送提示単位(PU)に対してアンロックキャッシュの対象に指定されている場合には、アプリケーション・データ制御部407は、アンロック対象ファイルがキャッシュ・メモリー408にキャッシュされていれば、これを削除するとともに、別途管理していたロックキャッシュ対象ファイルからも削除する。

(動作16) データ放送アプリケーション・エンジン409は、`primary_item` に指定されたアプリケーション・ファイルを、キャッシュ・メモリー408から実行する。

(動作17) その後、データ・コンテンツ・マネジメント・テーブル(DCMT)の更新により、現在提示状態にあるデータ放送提示単位(PU)に含まれるメンバー・ファイル(`PU_member_node`)やプリキャッシュ対象のファイルの構成が変化した場合には、上記の(動作13)~(動作16)の処理を行なう。

(動作18) データ放送アプリケーション・エンジン409におけるアプリケーション動作により他のデータ放送提示単位(PU)の提示状態に遷移した場合には、ロックキャッシュ対象のファイルは一旦キャッシュ・メモリー408に保持した上で、上記の(動作13)~(動作16)の処理を行なう。ロックキャッシュ対象以外のファイルは、キャッシュ・メモリー408から削除してもよい。

【0233】

図34には、受信機12における方式2に基づくファイル・データのキャッシュ制御手順をフローチャートの形式で示している。

【0234】

データ放送アプリケーション制御エンジン409がアプリケーション動作すなわちデータ放送の提示を行なっているときに(ステップS3401のYes)、アプリケーション・データ制御部407は、データ・コンテンツ・マネジメント・テーブル内で `primary_item` (データ放送提示単位の中心)に指定されたアプリケーション・ファイルにアクセスすることにより(ステップS3402のYes)、対応するデータ放送提示単位(PU)の提示状態に入ったことを認識する。

【0235】

アプリケーション・データ制御部407は、キャッシュ・メモリー408をリセットして、データ・コンテンツ・マネジメント・テーブル(DCMT)において、現在提示しているデータ放送提示単位(PU)の `primary_item` 並びに各メンバー・ファイル(`PU_member_node`)を取得して、キャッシュ・メモリー408に保持する(ステップS3403)。データ放送アプリケーション・エンジン409は、`prim`

10

20

30

40

50

`array__item`に指定されたアプリケーション・ファイルを、キャッシュ・メモリー 408 から実行する。

【0236】

また、アプリケーション・データ制御部 407 は、データ・コンテンツ・マネジメント・テーブル内で、該当するデータ放送提示単位 (PU) に対してロックキャッシュの対象が指定されているかどうかをチェックする (ステップ S3404)。そして、ロックキャッシュの対象が指定されている場合には (ステップ S3404 の Yes)、アプリケーション・データ制御部 407 は、ロックキャッシュ対象のうち未取得の各ファイル (`item`) も取得して、キャッシュ・メモリー 408 にキャッシュし、且つ、ロックキャッシュ対象ファイルとして別途管理する (ステップ S3405)。

10

【0237】

また、アプリケーション・データ制御部 407 は、データ・コンテンツ・マネジメント・テーブル内で、該当するデータ放送提示単位 (PU) に対してアンロックキャッシュの対象が指定されているかどうかをチェックする (ステップ S3406)。そして、アンロックキャッシュの対象が指定されている場合には (ステップ S3406 の Yes)、アプリケーション・データ制御部 407 は、アンロック対象ファイルがキャッシュ・メモリー 408 にキャッシュされていれば、これを削除するとともに、別途管理していたロックキャッシュ対象ファイルからも削除する (ステップ S3407)。

【0238】

その後、データ・コンテンツ・マネジメント・テーブル (DCMT) の更新により、現在提示状態にあるデータ放送提示単位 (PU) に含まれるメンバー・ファイル (`PU__member__node`) やプリキャッシュ対象のファイルの構成が変化した場合には (ステップ S3408 の Yes)、ステップ S3403 に戻り、現在提示状態にあるデータ放送提示単位 (PU) において、上記の処理を繰り返し実行する。

20

【0239】

また、データ放送アプリケーション・エンジン 409 におけるアプリケーション動作により他のデータ放送提示単位 (PU) の提示状態に遷移し、そのデータ放送提示単位の `primary__item` に指定されたアプリケーション・ファイルにアクセスした場合には (ステップ S3409 の Yes)、ステップ S3403 に戻り、遷移した先のデータ放送提示単位 (PU) において、上記の処理を繰り返し実行する。

30

【0240】

以上の処理を、データ放送アプリケーション・エンジン 409 がアプリケーション動作を終了するまで (ステップ S3410 の no)、繰り返し実行する。

【0241】

図 35 には、受信機 12 における方式 2 に基づくファイル・データのキャッシュのロック及びアンロック動作例を示している。

【0242】

アプリケーション・データ制御部 407 は、MMT 伝送路 504 上で受信する各種シグナリング・メッセージを解析している。シグナリング・メッセージの 1 つであるデータ・トランスミッション・メッセージには、データ・コンテンツ・マネジメント・テーブル (DCMT) が含まれている。

40

【0243】

データ放送アプリケーション制御エンジン 409 が `PU__id = 1` で識別されるデータ放送提示単位 (PU) のアプリケーション動作を行なっているとき、アプリケーション・データ制御部 407 は、データ・コンテンツ・マネジメント・テーブルを参照して、現在提示状態にあるデータ放送提示単位 (`PU__id = 1`) の `primary__item` (データ放送提示単位の中心) に指定されたアプリケーション・ファイル「A01.html」、並びに、各メンバー・ファイル (`PU__member__node`) 「B01」、「B02」それぞれの `node__tag` をデータ・コンテンツ・マネジメント・テーブルから取得して、データ・アセットを伝送する MMT 伝送路 503 からこれらの `node__ta`

50

g に対応するファイル・データのエンティティを取得すると、参照番号 3501 で示すように、キャッシュ・メモリー 408 にキャッシュする。なお、node_tag から MMT 伝送路 503 上で伝送されるファイルにアクセスする方法については、図 30 を参照しながら、既に説明した通りである（以下、同様）。但し、このデータ放送提示単位（PU）に対してロックキャッシュの対象が指定されていないので、キャッシュ動作は行なわない。

【0244】

次いで、データ放送アプリケーション・エンジン 409 におけるアプリケーション動作により、PU_id = 3 で識別される他のデータ放送提示単位（PU）の提示状態に遷移したとする。アプリケーション・データ制御部 407 は、上記と同様に、遷移した先のデータ放送提示単位（PU_id = 3）の primary_item（データ放送提示単位の中心）に指定されたアプリケーション・ファイル「A11.html」、並びに、各メンバー・ファイル（PU_member_node）「B11」、「B12」、「B13」それぞれの node_tag をデータ・コンテンツ・マネジメント・テーブルから取得すると、MMT 伝送路 503 からこれらの node_tag に対応するファイル・データのエンティティを取得して、参照番号 3502 で示すように、キャッシュ・メモリー 408 にキャッシュする。但し、このデータ放送提示単位（PU）に対してロックキャッシュの対象が指定されていないので、ロックキャッシュ動作は行なわない。

【0245】

アプリケーション・データ制御部 407 は、MMT 伝送路 504 上で受信する各種シグナリング・メッセージを常に解析している。そして、参照番号 3503 で示すように、データ・コンテンツ・マネジメント・テーブルのバージョンが 1 から 2 に更新されたことを検出すると、アプリケーション・データ制御部 407 は、現在提示状態にあるデータ放送提示単位（PU_id = 3）の primary_item 並びに各メンバー・ファイル、ロックキャッシュ対象のファイルに変更がないかどうかをチェックする。今回はメンバー・ファイルに変更がないので、以前キャッシュしておいた各メンバー・ファイルはキャッシュ・メモリー 408 に保持したままとする。また、データ放送提示単位（PU_id = 3）のロックキャッシュ対象のファイルとして「A12」、「B14」、「B12」、「B15」が追加されているので、アプリケーション・データ制御部 407 は、MMT 伝送路 503 からこれらの node_tag に対応するファイル・データのエンティティを取得して、参照番号 3504 で示すように、キャッシュ・メモリー 408 にキャッシュするとともに、ロックキャッシュ対象ファイルとして別途管理する。図中、ロックキャッシュ対象として管理されているファイルを下線で示している（以下、同様）。

【0246】

次いで、参照番号 3505 で示すように、データ放送の提示の更新を指示するイベント・メッセージを MMT 伝送路 504 から受信すると、データ放送アプリケーション・エンジン 409 は、実行中の A11 ファイルから A12 ファイルへの HTML 文書の遷移を行なう。一方でほぼ同時に、アプリケーション・データ制御部 407 は、バージョンが 2 から 3 に更新されたデータ・コンテンツ・マネジメント・テーブルを参照して、現在提示状態にあるデータ放送提示単位（PU_id = 3）の primary_item が「A12.html」に変更するとともに、メンバー・ファイルが「B14」、「B15」、「B16」に変更したことを検出する。上述したように、primary_item「A12.html」並びにメンバー・ファイルが「B14」、「B15」は既にキャッシュ・メモリー 408 にロックキャッシュされているので、データ放送アプリケーション・エンジン 409 は、キャッシュ・メモリー 408 に保持されているファイルを利用してデータ放送を迅速に表示することができる。すなわち、放送番組に連動したタイムリーなデータ放送の提示を行なうことができる。また、このデータ放送提示単位（PU）に対してロックキャッシュの対象が指定されていないので、ロックキャッシュ動作は行なわない。参照番号 3506 で示すように、ロックキャッシュされたファイル「A12」、「B14」、「B12」、「B15」がそのままキャッシュ・メモリー 408 に保持される一方、不要に

10

20

30

40

50

なったファイル「A11」が削除されている。

【0247】

さらに、データ放送の提示の更新を指示するイベント・メッセージをMMT伝送路504から受信すると、アプリケーション・データ制御部407は、データ・コンテンツ・マネジメント・テーブルを参照して、現在提示状態にあるデータ放送提示単位(PU_id=3)に対して、ロックキャッシュ対象のファイルとして「B17」が追加されているとともに、アンロック対象ファイルとして「B14」が追加されている。そこで、参照番号3507で示すように、アプリケーション・データ制御部407は、MMT伝送路503からファイル「B17」のnode_tagに対応するファイル・データのエンティティを取得してキャッシュ・メモリー408にロックキャッシュするとともに、ロックキャッシュしていたファイル「B12」をキャッシュ・メモリー408から削除しロック対象から外す。

10

【0248】

現在運用されているBML(Broadcast Markup Language)によるデータ放送サービスでは、スクリプトから「LockModuleOnMemory()」というAPI(Application Programming Interface)を呼び出すことにより、特定のファイルをあらかじめキャッシュ・メモリーにプリキャッシュして留めておくことが可能である(例えば、特許文献2を参照のこと)。この方法は、スクリプトなどのアプリケーションの仕様に、放送運用を前提とする特殊な仕様を盛り込む必要がある。

20

【0249】

これに対し、本明細書で開示する技術によれば、放送局などの送信側からは、データ放送に関わるシグナリングに、強制キャッシュを指定する情報を含めて伝送し、受信機側では、受信したデータ放送に関わるシグナリングに含まれている強制キャッシュ情報に基づいてデータ放送用の各ファイルのキャッシュ制御を行なうようになっている。したがって、本明細書で開示する技術によれば、新しいHTML5によるデータ放送において、スクリプトなどのアプリケーションの仕様に、放送運用を前提とする特殊な仕様を盛り込むことなく、汎用性の高いフォーマットを維持することができる。

【産業上の利用可能性】

【0250】

以上、特定の実施形態を参照しながら、本明細書で開示する技術について詳細に説明してきた。しかしながら、本明細書で開示する技術の要旨を逸脱しない範囲で当業者が該実施形態の修正や代用を成し得ることは自明である。

30

【0251】

本明細書で開示する技術は、トランスポート方式としてMMTを採用するさまざまな放送システムに適用することができる。また、本明細書で開示する技術は、放送番組に連動するデータ放送に利用されるファイル・データをMMT方式又はその他のトランスポート方式により伝送するさまざまなデータ放送システムに適用することができる。

【0252】

要するに、例示という形態により本明細書で開示する技術について説明してきたのであり、本明細書の記載内容を限定的に解釈するべきではない。本明細書で開示する技術の要旨を判断するためには、特許請求の範囲を参酌すべきである。

40

【0253】

なお、本明細書の開示の技術は、以下のような構成をとることも可能である。

(1) データ放送で利用されるファイル・データを送信するファイル・データ送信部と、データ放送に関わるシグナリングにファイル・データの強制キャッシュを指定する強制キャッシュ情報を含めて送信するシグナリング・メッセージ送信部と、を具備する送信装置。

(2) 前記シグナリング・メッセージ送信部は、データ放送提示単位毎に、提示単位を構成する放送伝送ファイル・リストと中心となるファイル、及び、プリキャッシュの対象フ

50

ファイル・リストの情報を記述したデータ・コンテンツ・マネジメント・テーブルを含んだデータ・トランスミッション・メッセージを送信する、
上記(1)に記載の送信装置。

(3) 前記シグナリング・メッセージ送信部は、データ放送提示単位毎に、提示単位を構成する放送伝送ファイル・リストと中心となるファイル、キャッシュにロックする対象ファイル及びロック対象のうちアンロックする対象ファイルの情報を記述したデータ・コンテンツ・マネジメント・テーブルを含んだデータ・トランスミッション・メッセージを送信する、

上記(1)に記載の送信装置。

(4) データ放送が連動する放送番組本体のメディア・データを送信するメディア・データ送信部をさらに備える、

10

上記(1)乃至(3)のいずれかに記載の送信装置。

(5) データ放送で利用されるファイル・データを送信するファイル・データ送信ステップと、

データ放送に関わるシグナリングにファイル・データの強制キャッシュを指定する強制キャッシュ情報を含めて送信するシグナリング・メッセージ送信ステップと、
を有する送信方法。

(6) データ放送で利用されるファイル・データを送信するファイル・データ受信部と、
データ放送に関わるシグナリングにファイル・データの強制キャッシュを指定する強制キャッシュ情報を含めて送信するシグナリング・メッセージ受信部と、

20

前記強制キャッシュ情報に基づいて前記ファイル・データ受信部が受信するファイル・データのキャッシュ・メモリーへのキャッシュを制御する制御部と、
を具備する受信装置。

(7) 前記シグナリング・メッセージ受信部は、データ放送提示単位毎に、提示単位を構成する放送伝送ファイル・リストと中心となるファイル、及び、プリキャッシュの対象ファイル・リストの情報を記述したデータ・コンテンツ・マネジメント・テーブルを含んだデータ・トランスミッション・メッセージを受信する、

上記(6)に記載の受信装置。

(8) 前記制御部は、前記プリキャッシュの対象ファイル・リストに含まれているファイルを前記ファイル・データが受信すると、前記キャッシュ・メモリーにプリキャッシュする、

30

上記(7)に記載の受信装置。

(9) 前記シグナリング・メッセージ受信部は、データ放送提示単位毎に、提示単位を構成する放送伝送ファイル・リストと中心となるファイル、キャッシュにロックする対象ファイル及びロック対象のうちアンロックする対象ファイルの情報を記述したデータ・コンテンツ・マネジメント・テーブルを含んだデータ・トランスミッション・メッセージを受信する、

上記(6)に記載の受信装置。

(10) 前記制御部は、前記ロック対象のファイルを前記ファイル・データが受信すると、前記キャッシュ・メモリーにプリキャッシュする、

40

上記(9)に記載の受信装置。

(11) 前記制御部は、前記アンロック対象のファイルを前記キャッシュ・メモリーから削除する、

上記(9)に記載の受信装置。

(12) 前記制御部は、現在のデータ放送提示単位を構成する放送伝送ファイル・リストと中心となるファイルを前記ファイル・データが受信すると、前記キャッシュ・メモリーにキャッシュする、

上記(6)又は(9)のいずれかに記載の受信装置。

(13) ファイル・データを利用してデータ放送を提示するデータ放送提示部をさらに備える、

50

上記(6)乃至(12)のいずれかに記載の受信装置。

(14)データ放送が連動する放送番組本体のメディア・データを受信するメディア・データ受信部と、メディア・データに基づいて放送番組を提示する放送番組提示部をさらに備える、

上記(6)乃至(13)のいずれかに記載の受信装置。

(15)データ放送で利用されるファイル・データを送信するファイル・データ受信ステップと、

データ放送に関わるシグナリングにファイル・データの強制キャッシュを指定する強制キャッシュ情報を含めて送信するシグナリング・メッセージ受信ステップと、

前記強制キャッシュ情報に基づいて前記ファイル・データ受信部が受信するファイル・データのキャッシュ・メモリーへのキャッシュを制御する制御ステップと、を有する受信方法。

10

【符号の説明】

【0254】

10...デジタル放送システム

11...放送送出システム、12...受信機

301...時計部、302...信号送出部、303...ビデオ・エンコーダー

304...オーディオ・エンコーダー、305...キャプション・エンコーダー

306...シグナリング・エンコーダー、307...ファイル・エンコーダー

308...情報システム、309...TLVシグナリング・エンコーダー

20

310...IPサービス・マルチプレクサー

311...TLVマルチプレクサー、312...変調・送信部

401...チューナー・復調部、402...デマルチプレクサー

403...時計部、404...ビデオ・デコーダー

405...オーディオ・デコーダー、406...キャプション・デコーダー

407...アプリケーション・データ制御部、408...キャッシュ・メモリー

409...データ放送アプリケーション・エンジン

410...システム制御部、411...合成部

412...IPインターフェース

【要約】

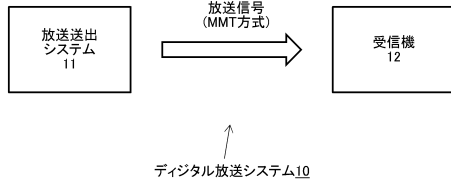
30

【課題】HTML5によるデータ放送のファイルをMMT方式で伝送する。

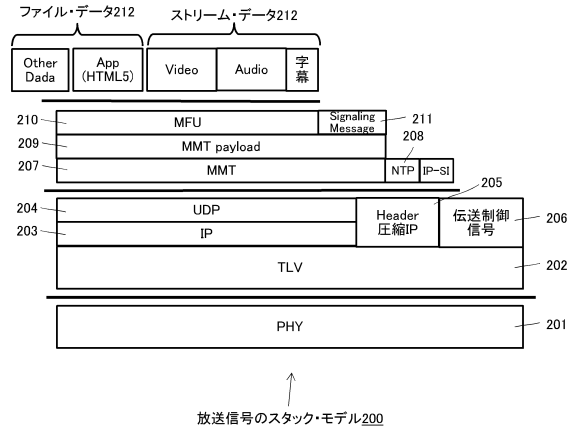
【解決手段】データ・コンテンツ・マネジメント・テーブルにおいて、データ放送提示単位毎に、提示単位を構成する放送伝送ファイル・リストと中心となるファイル、及びプリキャッシュの対象ファイル・リストの情報を記述する。現在のデータ放送提示単位から次に参照するデータ放送提示単位を構成する放送ファイル・リストをプリキャッシュの対象に指定すると、受信機側では次に遷移するデータ放送提示単位に必要なファイル・データをプリキャッシュしておくことができ、放送番組に連動したタイムリーなデータ放送サービスを実現できる。

【選択図】 図33

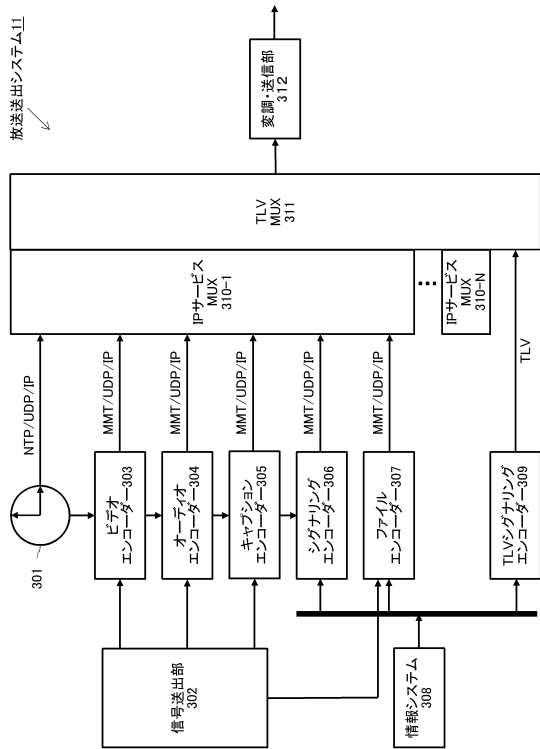
【図1】



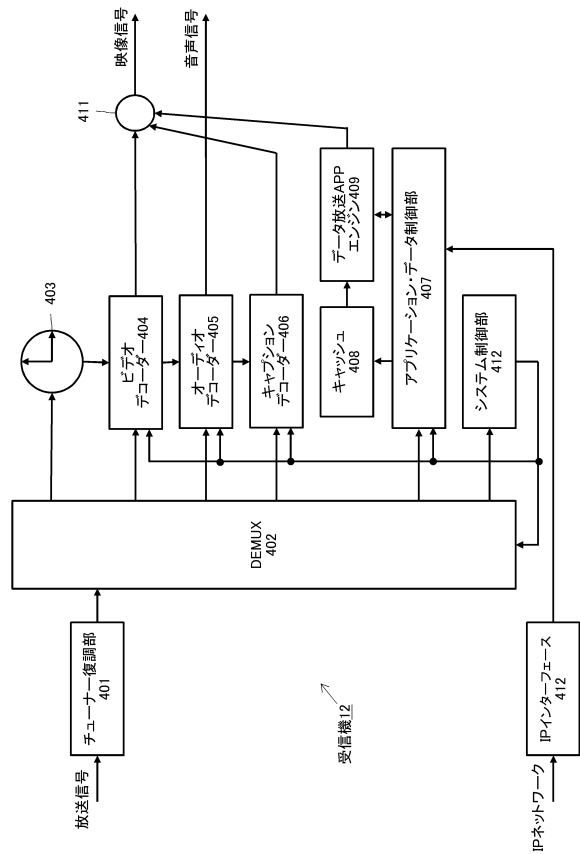
【図2】



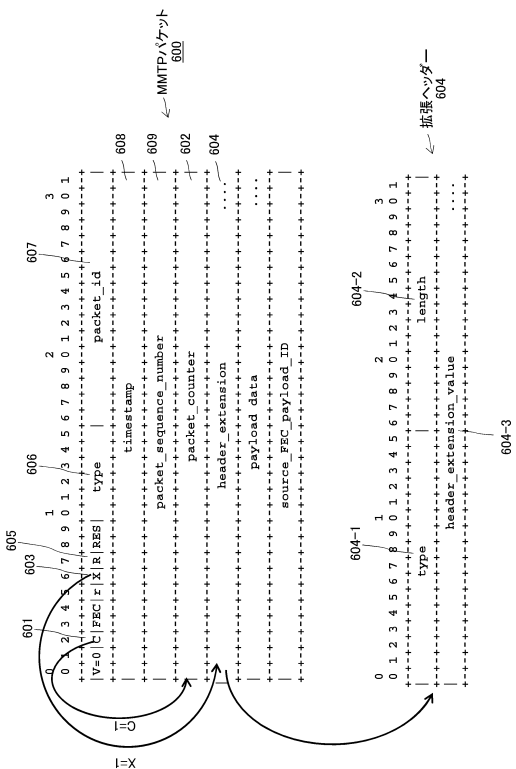
【図3】



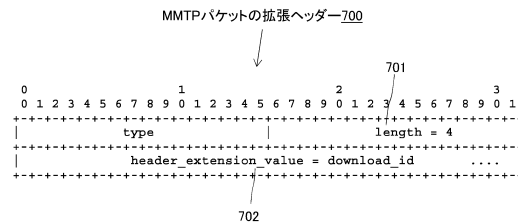
【図4】



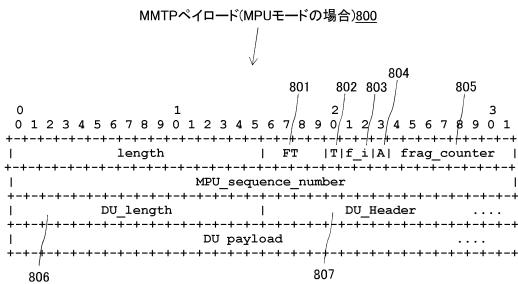
【 図 6 】



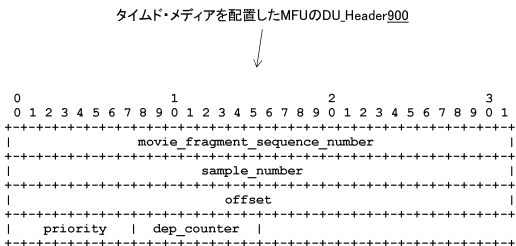
【 図 7 】



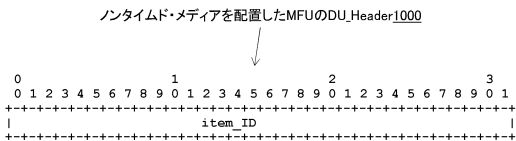
【 図 8 】



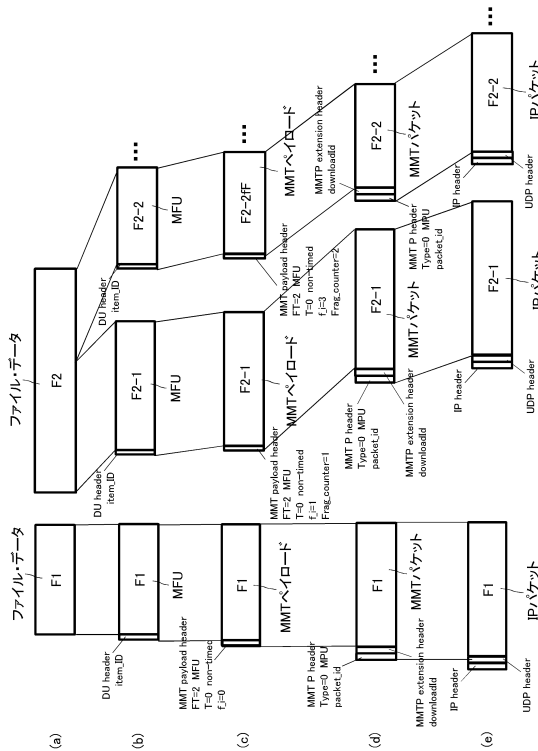
【 図 9 】



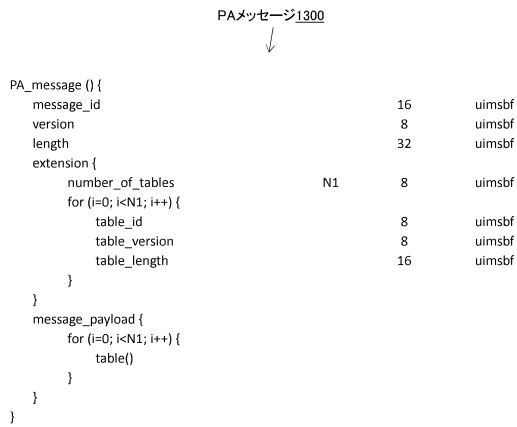
【 図 10 】



【 図 11 】



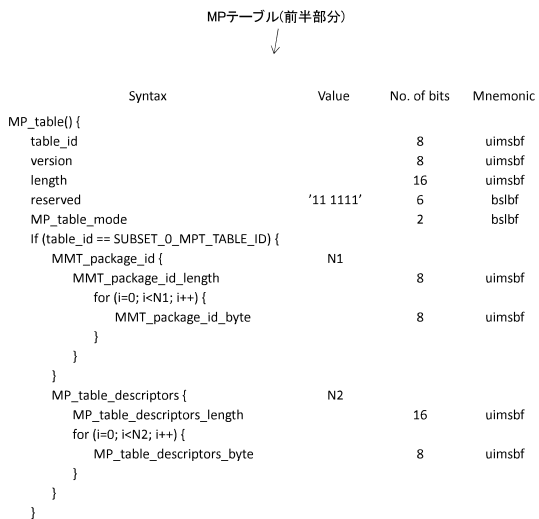
【 図 1 3 】



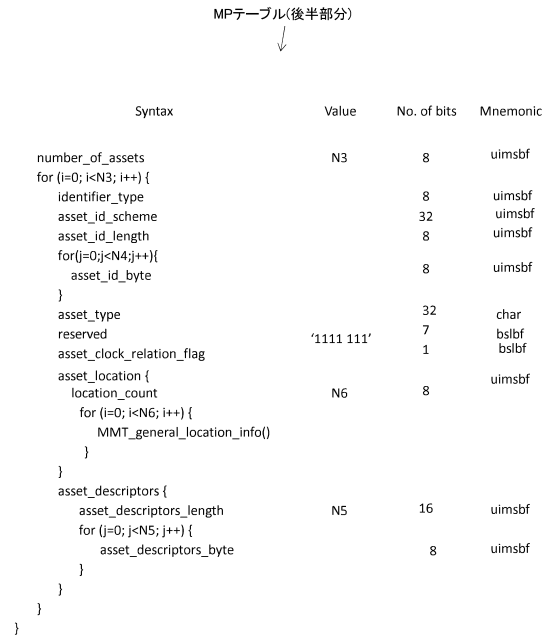
【 図 1 4 】

| 項目 | 日本語項目名 | 説明 |
|------------|---------|--|
| message_id | メッセージID | 各種シグナリング情報においてPA messageを識別する固定値 |
| version | バージョン | PA Messageのバージョンを示す。8bit整数値MPTを構成する一部のパラメータでも更新した場合には+1インクリメントされる。 |
| length | テーブル長 | PA Messageのバイト数。このフィールドの直後からカウントする。 |

【 図 1 5 】



【 図 1 6 】



【 図 1 7 】

| 項目 | 日本語項目名 | 説明 |
|-------------------|------------|---|
| table_id | テーブルID | 各種シグナリング情報においてMP tableを識別する固定値 |
| version | バージョン | MPTのバージョンを示す。8bit整数値MPTを構成する一部のパラメータでも更新した場合には+1インクリメントされる。 |
| length | テーブル長 | MP tableのバイト数。このフィールドの直後からカウントする。 |
| package_id | パッケージID | 放送信号で伝送される全ての信号、ファイルを構成要素とする全体のパッケージとしての識別情報。 |
| MPT_descriptors | MPT記述子領域 | パッケージ全体に関わる記述子の格納領域。記述子はさまざまな目的の記述子を規定した上で1つ又は複数配置する想定。 |
| number_of_assets | アセット数 | パッケージを構成する要素としての信号(アセット)の数。この数分だけ以下のアセットループが配置される。 |
| asset_id | アセットID | アセットをユニークに識別するID |
| gen_loc_info | 一般ロケーション情報 | アセットの取得先のロケーションを示す |
| asset_descriptors | Asset記述子領域 | アセットに関わる記述子の格納領域。記述子はさまざまな目的の記述子を規定した上で1つ又は複数配置する想定。 |

【 図 1 8 】

M2セクション・メッセージ1800

↓

| Syntax | No. of bits | Mnemonic |
|--------------------------|-------------|----------|
| M2section_message(){ | | |
| message_id | 16 | uimsbf |
| version | 8 | uimsbf |
| length | 16 | uimsbf |
| table_id | 8 | uimsbf |
| section_syntax_indicator | 1 | bslbf |
| '1' | 1 | bslbf |
| '11' | 2 | bslbf |
| section_length | 12 | uimsbf |
| table_id_extension | 16 | uimsbf |
| '11' | 2 | bslbf |
| version_number | 5 | uimsbf |
| current_next_indicator | 1 | bslbf |
| section_number | 8 | uimsbf |
| last_section_number | 8 | uimsbf |
| for(i=0; i<N; i++){ | | |
| signaling_data_byte | 8 | bslbf |
| } | | |
| CRC_32 | 32 | rpcof |
| } | | |
| | | uimsbf |
| | | uimsbf |

【 図 1 9 】

MH AIテーブル(MH AIT)1900

↓

| Syntax | No. of bits | Mnemonic |
|------------------------------------|-------------|----------|
| MH-Applicatin_Information_Table(){ | | |
| table_id | 8 | uimsbf |
| section_syntax_indicator | 1 | bslbf |
| reserved_future_use | 1 | bslbf |
| reserved | 2 | bslbf |
| section_length | 12 | uimsbf |
| application_type | 16 | uimsbf |
| reserved | 2 | bslbf |
| version_number | 5 | uimsbf |
| current_next_indicator | 1 | bslbf |
| section_number | 8 | uimsbf |
| last_section_number | 8 | uimsbf |
| reserved_future_use | 4 | bslbf |
| common_descriptor_length | 8 | uimsbf |
| for(i=0; i<N; i++){ | | |
| descriptor () | | rpcof |
| } | | |
| reserved_future_use | 4 | uimsbf |
| application_loop_length | | |
| for(i=0; i<N; i++){ | | |
| application_identifier () | | |
| application_control_code | 8 | uimsbf |
| reserved_future_use | 4 | bslbf |
| application_descriptor_loop_length | 12 | uimsbf |
| for (j=0; j<M; j++) { | | |
| descriptor () | | |
| } | | |
| } | | |
| CRC32 | 32 | rpcof |
| } | | |

【 図 2 0 】

アプリケーション情報記述子2000

↓

| Syntax | No. of bits | Mnemonic |
|----------------------------|-------------|----------|
| application_descriptor(){ | | |
| descriptor_tag | 8 | uimsbf |
| descriptor_length | 8 | uimsbf |
| application_profile_length | 8 | uimsbf |
| for(i=0; i<N; i++){ | | |
| application_profile | 16 | uimsbf |
| version_major | 8 | uimsbf |
| version_minor | 8 | uimsbf |
| version_micro | 8 | uimsbf |
| } | | |
| service_bound_flag | 1 | bslbf |
| visibility | 2 | bslbf |
| reserved_future_use | 5 | bslbf |
| application_priority | 8 | uimsbf |
| for(i=0; i<N; i++){ | | |
| transport_protocol_label | 8 | uimsbf |
| } | | |
| } | | |

【 図 2 1 】

| データ | 意味 |
|---|---|
| application_profile | 本アプリケーションが実行可能である受信機のプロファイル。受信機に要求する機能毎のビットマップで要求機能を示す。但し上位3bitは機能ビットマップ切り替えを示す。上記ビットマップはversion毎に規定する。 |
| version_major version_minor version_micro | アプリケーションプロファイル規定のバージョン。 |
| service_bound_flag | 本アプリケーションが現在のサービスのみで有効かどうかを示す。 |
| visibility | アプリケーション可視か否かを示す。 |
| application_priority | このサービス内で告知されているアプリケーション間の相対優先度 |
| transport_protocol_label | アプリケーションを送送するプロトコルを示す。 |

【 図 2 2 】

伝送プロトコル記述子2200

| Syntax | No. of bits | Mnemonic |
|----------------------------------|-------------|----------|
| transport_protocol_descriptor(){ | | |
| descriptor_tag | 8 | uimsbf |
| descriptor_length | 8 | uimsbf |
| protocol_id | 16 | uimsbf |
| transport_protocol_label | 8 | uimsbf |
| for(i=0;i<N;i++){ | | |
| selector_byte | 8 | uimsbf |
| } | | |
| } | | |

【 図 2 3 】

セレクター・バイト2300

| Syntax | No. of bits | Mnemonic |
|--------------------------------------|-------------|----------|
| for(i=0;i<N;i++){ | | |
| URL_base_length | 8 | uimsbf |
| for(j=0;j<N;j++){ | | |
| URL_base_byte | 8 | uimsbf |
| } | | |
| URL_extension_count | 8 | uimsbf |
| for(j=0;j<URL_extension_count;j++){ | | |
| URL_extension_length | 8 | uimsbf |
| for(k=0;k<URL_extension_length;k++){ | | |
| URL_extension_byte | 8 | uimsbf |
| } | | |
| } | | |
| } | | |

【 図 2 4 】

データ・トランスミッション・メッセージ2400

| Syntax | No. of bits | Mnemonic |
|-----------------------------------|-------------|----------|
| Data_Transmission_message(){ | | |
| message_id | 16 | uimsbf |
| version | 8 | uimsbf |
| length | 32 | uimsbf |
| num_of_tables | 8 | uimsbf |
| for (i=0; i<num_of_tables;i++) { | | |
| table_id | 8 | uimsbf |
| table_version | 8 | uimsbf |
| table_length | 16 | uimsbf |
| } | | |
| for (i=0; i<num_of_tables; i++) { | | |
| table() | | |
| } | | |
| } | | |

【 図 2 5 】

データ・アセット・マネジメント・テーブル(DAMT)2500

| Syntax | No. of bits | Mnemonic |
|--|-------------|----------|
| Data_Asset_Management_Table(){ | | |
| table_id | 8 | uimsbf |
| version | 8 | uimsbf |
| length | 16 | uimsbf |
| number_of_asset | 8 | Uimsbf |
| for(k=0;k<number_of_asset;k++){ | | |
| download_id | 32 | uimsbf |
| asset_ID_scheme | 32 | uimsbf |
| asse_ID_length | 8 | Uimsbf |
| for(i=0;i<asset_ID_length;i++){ | | |
| asset_ID_byte | 8 | uimsbf |
| } | | |
| number_of_items | 8 | uimsbf |
| for(i=0;i<number_of_items;i++){ | | |
| item_ID | 32 | uimsbf |
| node_tag | 16 | uimsbf |
| item_size | 32 | uimsbf |
| item_version | 8 | uimsbf |
| item_checksum | 32 | uimsbf |
| item_info_length | 8 | uimsbf |
| for(j=0;j<item_info_length;j++){ | | |
| item_info() | | |
| } | | |
| } | | |
| descriptor_loop_length | 16 | uimsbf |
| for(i=0;i<descriptor_loop_length;i++){ | | |
| descriptor() | | |
| } | | |
| } | | |

【 図 2 6 】

データ・ディレクトリ・マネジメント・テーブル(DDMT)2600

| Syntax | No. of bit | Mnemonic |
|---|------------|----------|
| Data_Directory_Management_Table(){ | | |
| table_id | 8 | uimsbf |
| version | 8 | uimsbf |
| length | 16 | uimsbf |
| base_folder_path_length | 8 | uimsbf |
| for(i=0;i<base_folder_path_length;i++){ | | |
| base_folder_path_byte | 8 | uimsbf |
| } | | |
| number_of_folder_nodes | 8 | uimsbf |
| for(i=0;i<number_of_folder_nodes;i++){ | | |
| node_tag | 16 | uimsbf |
| folder_node_path_length | 16 | uimsbf |
| for(j=0;j<folder_node_path_length;j++){ | | |
| forder_node_path_byte | 8 | char |
| } | | |
| } | | |
| num_of_files | 16 | uimsbf |
| for(j=0;j<num_of_files;j++){ | | |
| node_tag | 16 | uimsbf |
| file_name_length | 8 | uimsbf |
| for(k=0;k<file_name_length;k++){ | | |
| file_name_byte | 8 | char |
| } | | |
| } | | |
| } | | |

【 図 2 7 】

データ・コンテンツ・マネジメント・テーブル(DCMT)2700

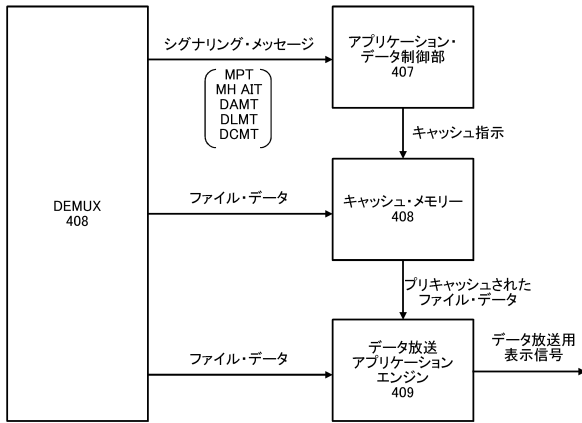
| Syntax | No. of bit | Mnemonic |
|---|------------|----------|
| Data_Content_Management_Table(){ | | |
| table_id | 8 | uimsbf |
| version | 8 | uimsbf |
| length | 16 | uimsbf |
| number_of_content | 8 | uimsbf |
| for(i=0;i<number_of_items;i++){ | | |
| contentID | 16 | uimsbf |
| content_version | 8 | uimsbf |
| content_cache_size | 32 | uimsbf |
| number_of_PU | 8 | uimsbf |
| for(j=0;j<number_of_PU;j++){ | | |
| PU_tag | 8 | uimsbf |
| PU_cache_size | 32 | uimsbf |
| PU_primary_item_node_tag | 16 | uimsbf |
| number_of_PU_member_nodes | 16 | uimsbf |
| for(k=0;k<number_of_nodes;j++){ | | |
| PU_member_node_tag | 16 | uimsbf |
| } | | |
| number_of_pre_cache_nodes | 8 | uimsbf |
| for(k=0;k<number_of_pre_cache_nodes;k++){ | | |
| pre_cache_node_tag | 16 | uimsbf |
| } | | |
| number_of_linked_PU | 8 | uimsbf |
| for(k=0;k<number_of_linkedPU;k++){ | | |
| linked_PU_tag | 8 | uimsbf |
| } | | |
| } | | |
| } | | |
| } | | |

【 図 2 8 】

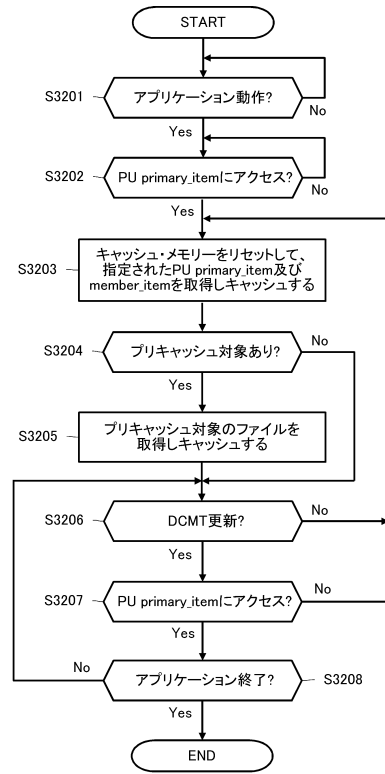
データ・コンテンツ・マネジメント・テーブル(DCMT)2800

| Syntax | No. of bit | Mnemonic |
|--|------------|----------|
| Data_Content_Management_Table(){ | | |
| table_id | 8 | uimsbf |
| version | 8 | uimsbf |
| length | 16 | uimsbf |
| number_of_content | 8 | uimsbf |
| for(i=0;i<number_of_items;i++){ | | |
| contentID | 16 | uimsbf |
| content_version | 8 | uimsbf |
| content_cache_size | 32 | uimsbf |
| number_of_PU | 8 | uimsbf |
| for(j=0;j<number_of_PU;j++){ | | |
| PU_tag | 8 | uimsbf |
| PU_cache_size | 32 | uimsbf |
| PU_primary_item_node_tag | 16 | uimsbf |
| number_of_PU_member_nodes | 16 | uimsbf |
| for(k=0;k<number_of_nodes;j++){ | | |
| PU_member_node_tag | 16 | uimsbf |
| } | | |
| number_of_lock_cache_nodes | 8 | uimsbf |
| for(k=0;k<number_of_lock_cache_nodes;k++){ | | |
| lock_cache_node_tag | 16 | uimsbf |
| } | | |
| number_of_unlock_cache_nodes | 8 | uimsbf |
| for(k=0;k<number_of_unlock_cache_nodes;k++){ | | |
| unlock_cache_node_tag | 16 | uimsbf |
| } | | |
| } | | |
| number_of_linked_PU | 8 | uimsbf |
| for(k=0;k<number_of_linkedPU;k++){ | | |
| linked_PU_tag | 8 | uimsbf |
| } | | |
| } | | |

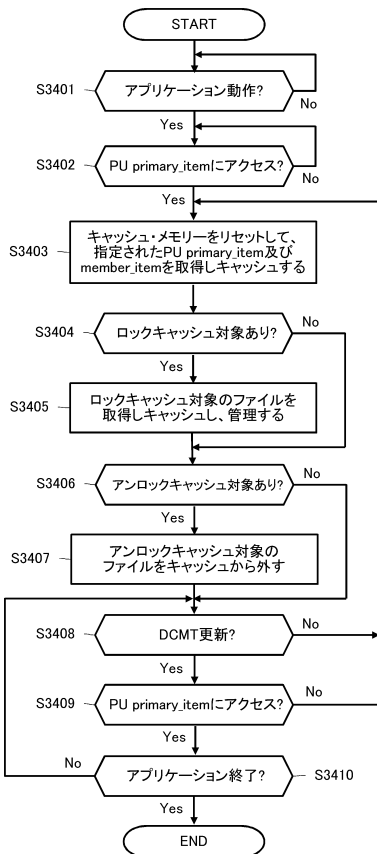
【図31】



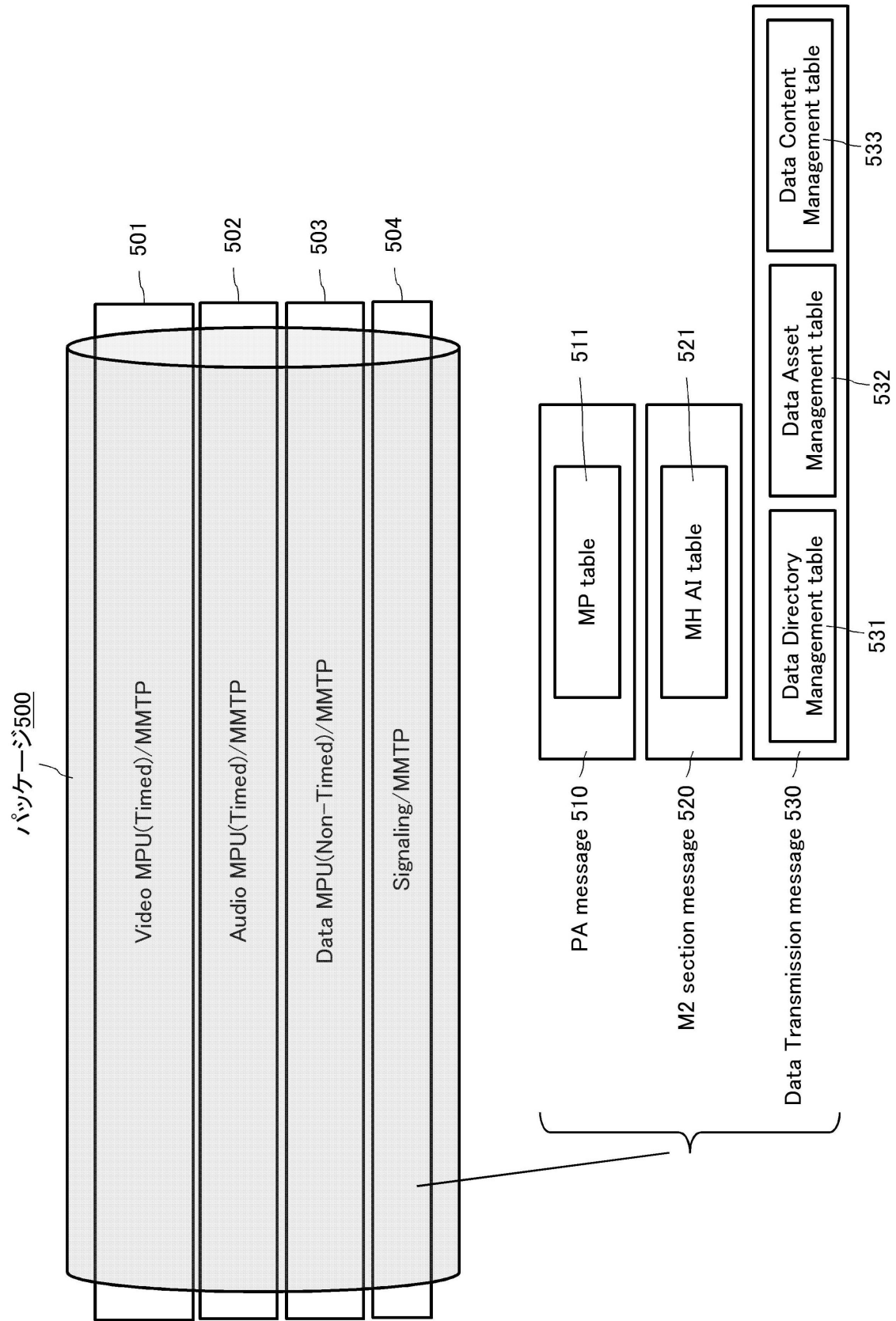
【図32】



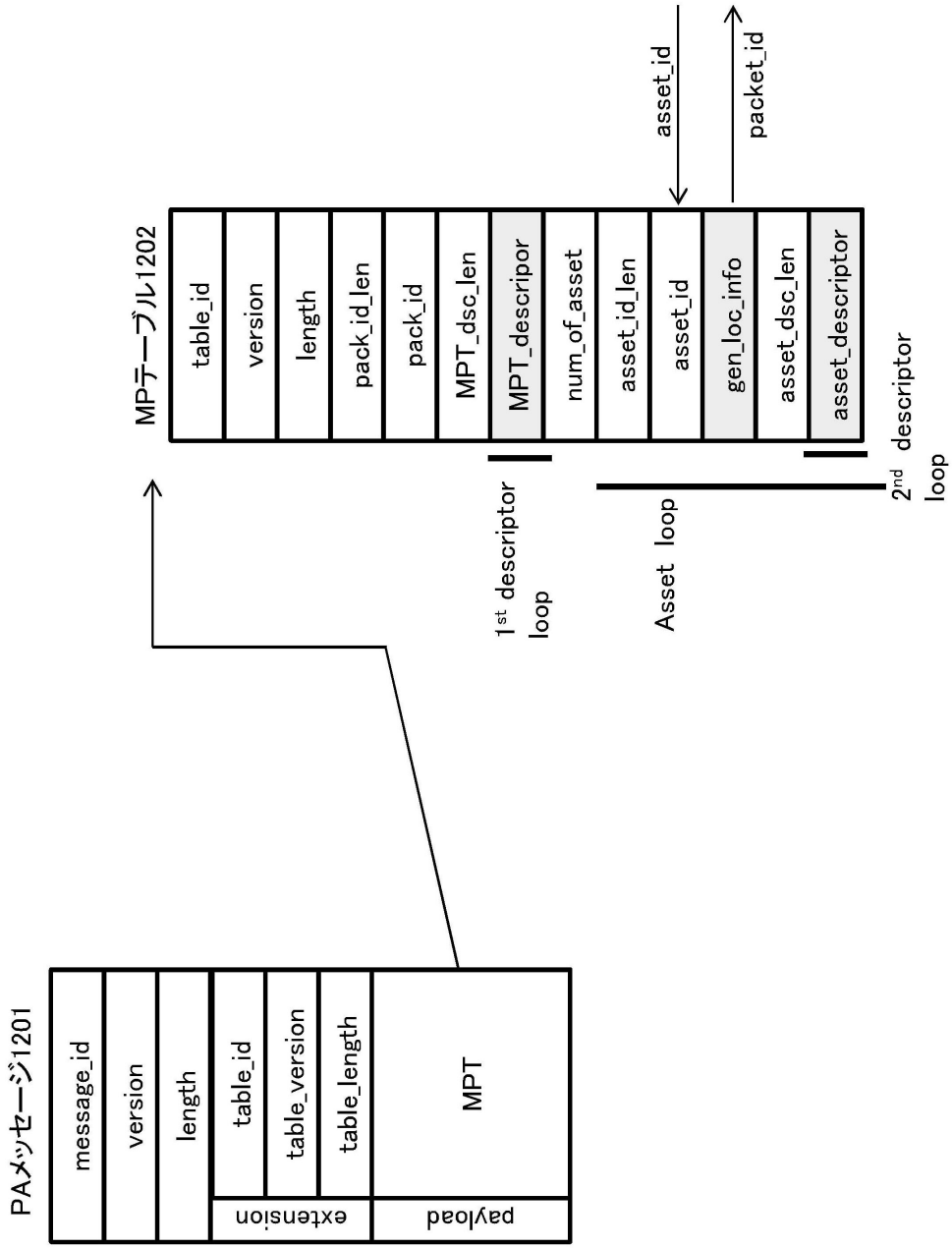
【図34】



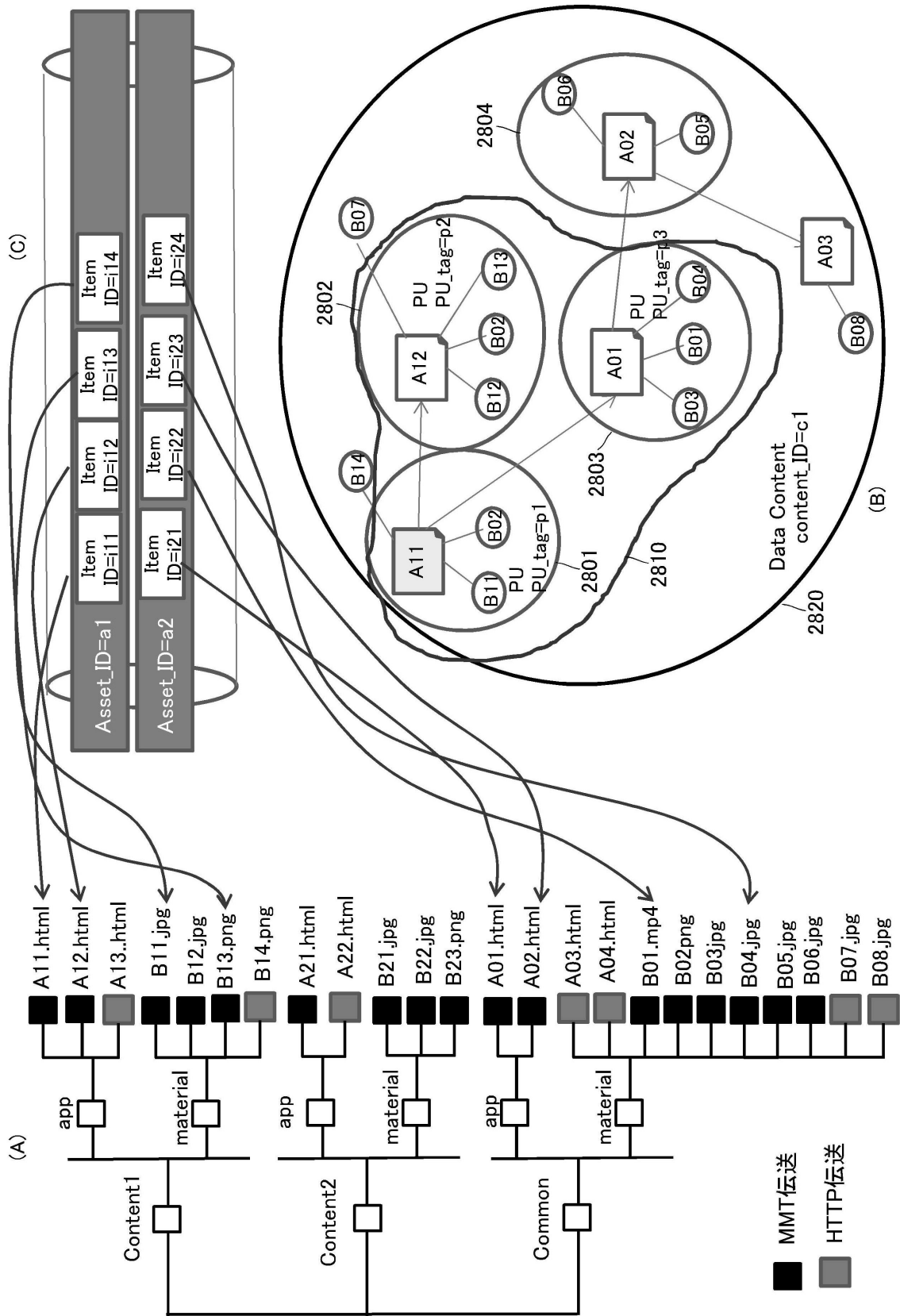
【 図 5 】



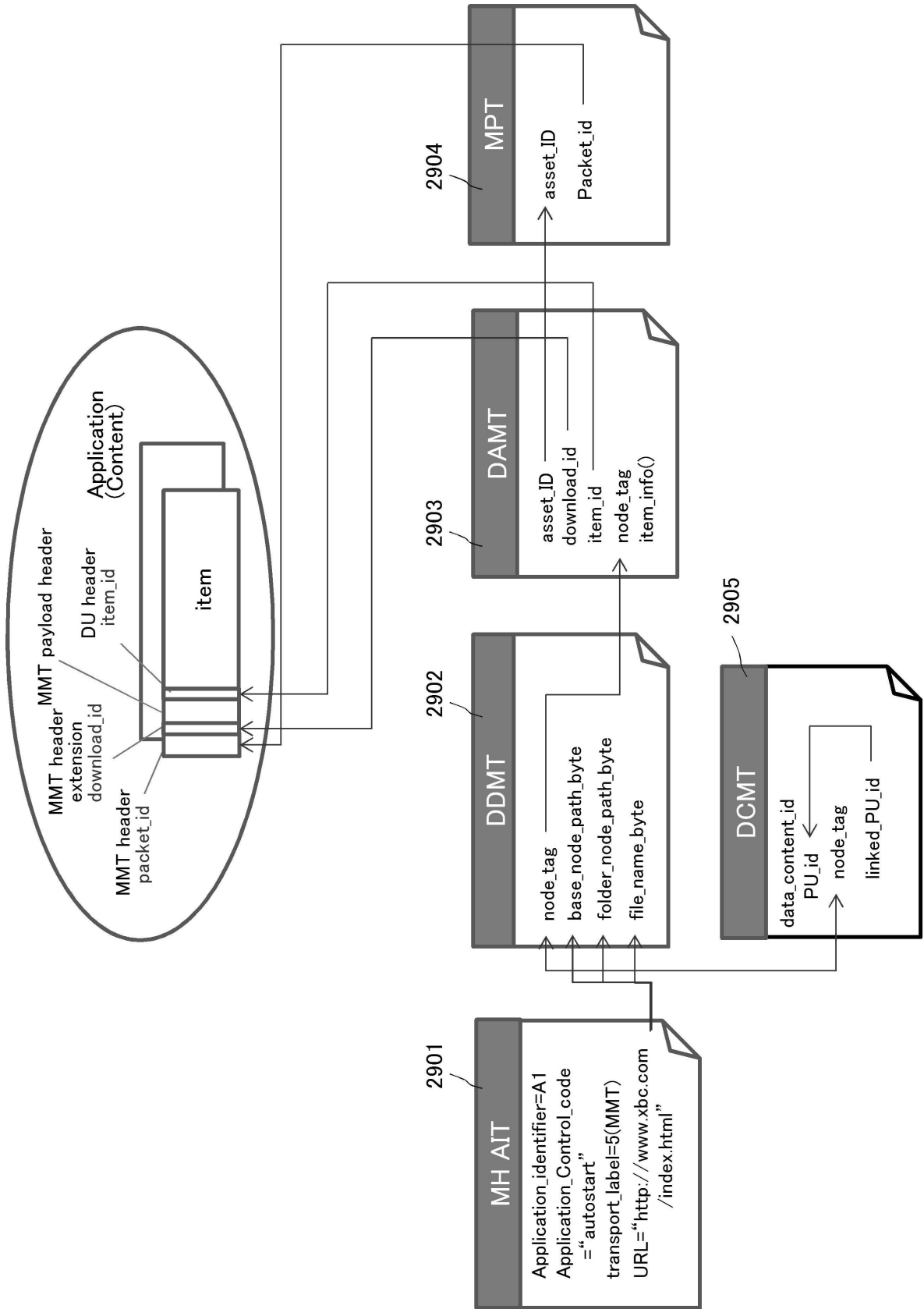
【 図 1 2 】



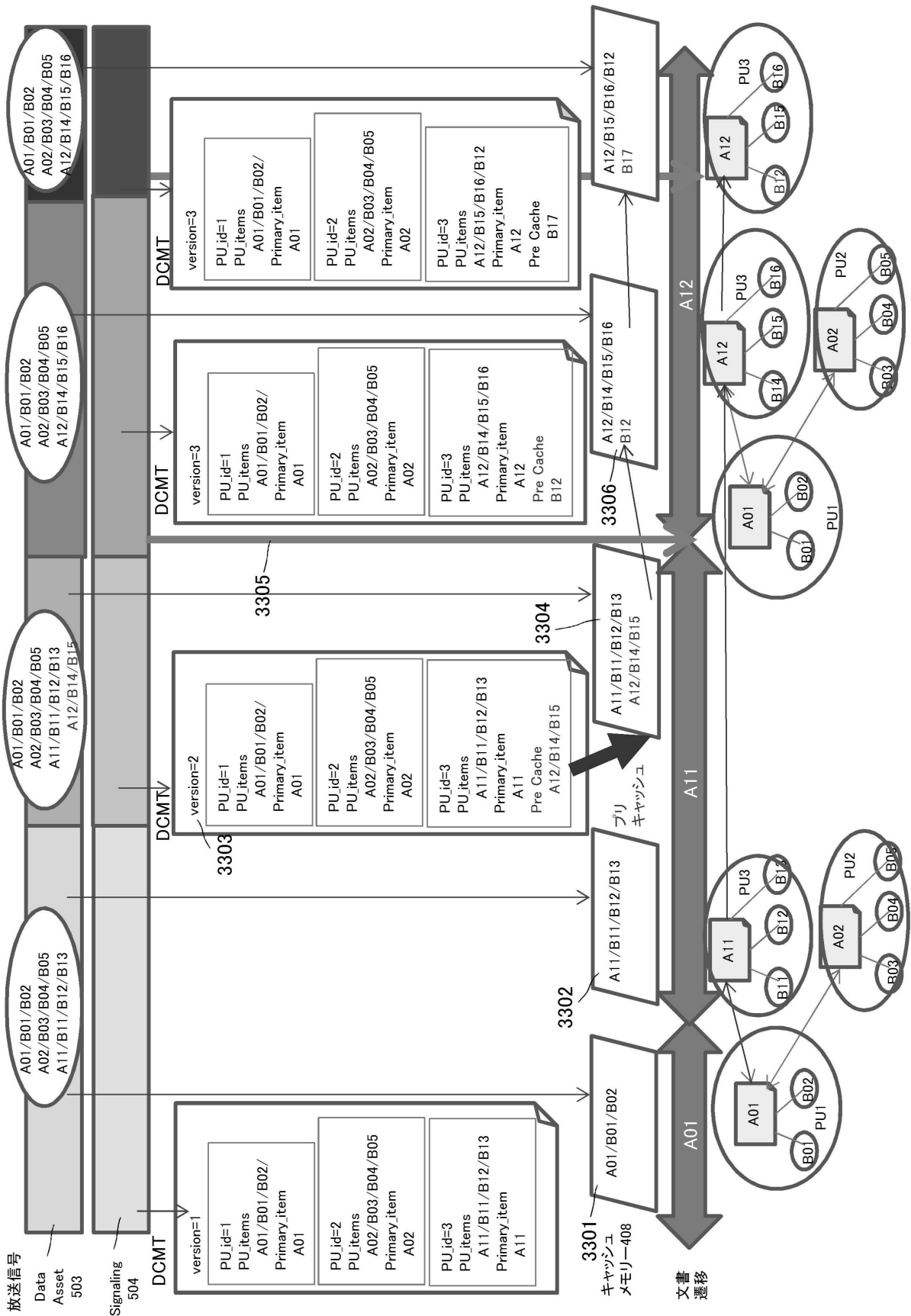
【 図 2 9 】



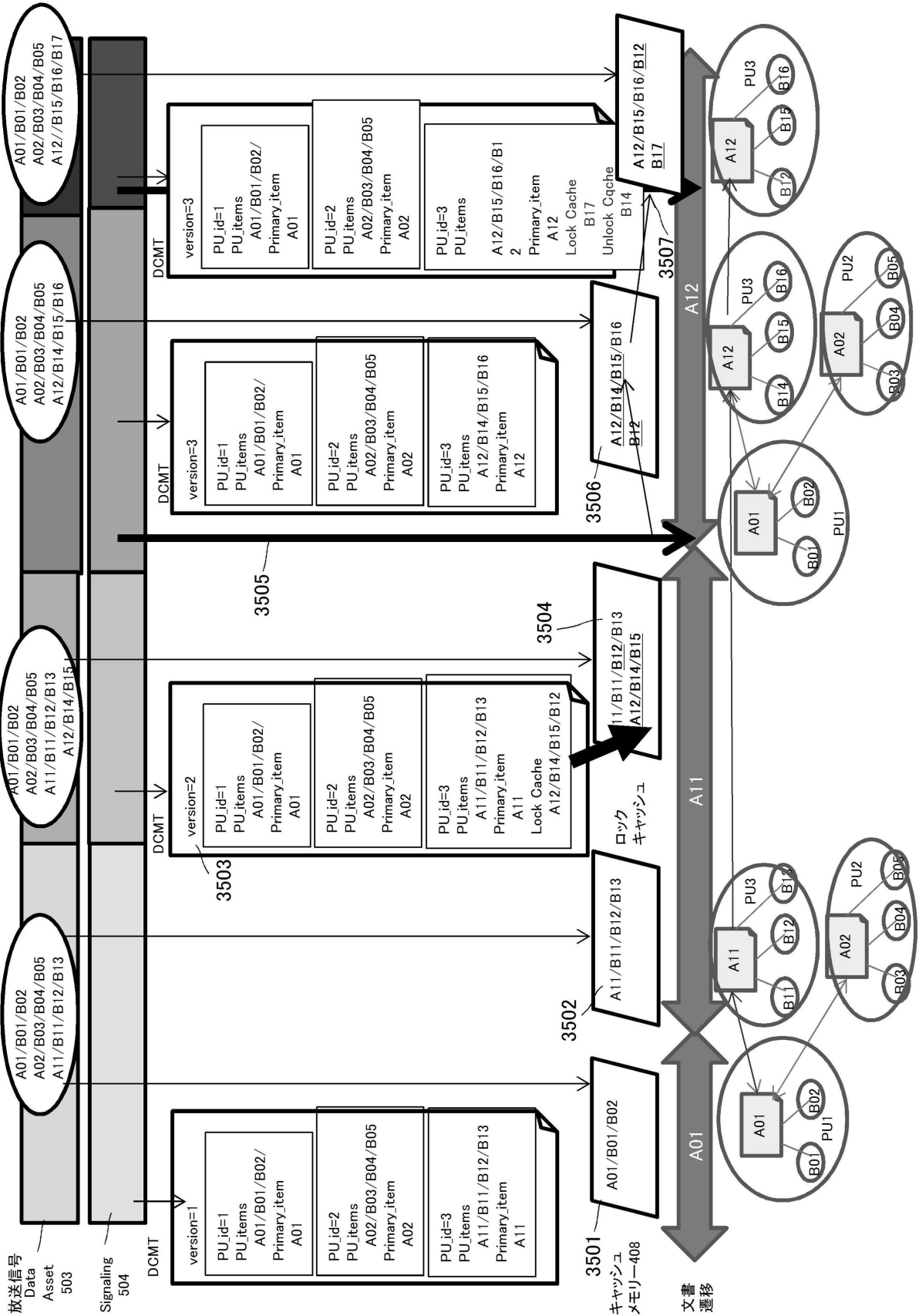
【 図 3 0 】



【 図 3 3 】



【 図 3 5 】



放送信号
Data
Asset
503

Signaling
504

DCMT

version=1

version=2

version=2

version=3

version=3

version=3

version=3

version=3

version=3

version=3

version=3

version=3

version=3

version=3

version=3

version=3

version=3

version=3

version=3

3501

3502

3503

3504

3505

3506

3507

3508

3509

3510

3511

3512

3513

3514

3515

3516

3517

3518

3519

3520

3521

3522

3523

3524

3525

3526

3527

3528

キャッシュ
メモリー-408

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

キャッシュ
メモリー-408

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

キャッシュ
メモリー-408

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

ロック
キャッシュ

フロントページの続き

- (72)発明者 北里 直久
東京都港区港南1丁目7番1号 ソニー株式会社内
- (72)発明者 出葉 義治
東京都港区港南1丁目7番1号 ソニー株式会社内

審査官 後藤 嘉宏

- (56)参考文献 大槻ほか, スーパーハイビジョン衛星放送システムにおけるMMTを用いたデータ伝送方式の検討, 映像情報メディア学会技術報告, 日本, (一社)映像情報メディア学会, 2014年 2月 28日, Vol.38 No.14, p.29-32
- Digital Video Broadcasting (DVB); IP Datacast: Content Delivery Protocols (CDP) Implementation Guidelines; Part 1: IP Datacast over DVB-H, 2010年 2月, ETSI TS 102 591-1 V1.3.1 (2010-02), p.63-65, URL, http://www.etsi.org/deliver/etsi_ts/102500_102599/10259101/01.03.01_60/ts_10259101v010301p.pdf

- (58)調査した分野(Int.Cl., DB名)
- H04N 21/00 - 21/858