コンテントオリエンティッド符号化実現のためのシステム提案

A Proposal of a New System for Content-oriented Coding

清水 直人 *1　宮澤 敏記 *2　亀山 渉 *1　渡辺 裕 *1　富永 英義 *1*2
Naoto SHIMIZU*1 Toshinori MIYAZAWA*2 Wataru KAMEYAMA*1 Hiroshi WATANABE*1 Hideyoshi TOMINAGA*1*2

†早稲田大学大学院 国際情報通信研究科
†*Global Info. and Tele. Institute, WASEDA Univ.*

‡早稲田大学 理工学部 電子・情報通信学科
‡*Dept. of Elec. Info. and Comm. Eng., WASEDA Univ.*

1 はじめに

コンテントオリエンティッド符号化は，各コンテンツの信号特性を考慮に入れた符号化である．自然画像と合成画像はコンテンツの特性からみれば，それぞれ別々の性質を持っている．MPEG に代表される DCT を基本とした符号化方式が広く利用されているが，これらの符号化方式はアニメーション画像に代表される人工画に対して最適化されていない [1]．この様なあらゆるコンテンツに対して一意の符号化方式を適用する方法は，コンテンツサービスプロバイダ・サービス利用者双方に様々な簡便性を提供する一方で，符号化効率の観点からはコンテントオリエンティッド符号化より劣る．したがって，この簡便性を保ちつつ，コンテントオリエンティッド符号化を実現することが可能ならば，より豊かなサービスを提供可能となる．本稿では，コンテントオリエンティッド符号化を実現する為のシステムを提案する．

2 Decoder Downloadable System

2.1 必要性

コンテントオリエンティッド符号化実現のためには，新たな符号化方式の開発だけでなく，それらを実装したデコーダをいかに配信するかということを考慮に入れる必要がある．インターネットストリーミングの分野 [2] においては，新たな符号化方式に対応する為に，Plug-in という機構を提供している．しかし，これらの機構をインターネット上のみならず，放送などの分野に拡張し，コンテントオリエンティッド符号化を実現するためには，表 1 に示す機能が要求される．

表 1　要求される機能

要求機能	従来提供済み
新しいデコーダの提供	○
オートデコーダダウンロード	○
コンテンツ間の連続再生	×
オートデコーダダウンロード + 連続再生	×

我々が提案する Decoder Downloadable System では，表 1 において従来提供されてない機能の提供を目指している．

2.2 システムアーキテクチャ

2.2.1 処理要求

コンテンツを連続再生する制約条件下では，各コンテンツに最適なデコーダをメディア再生以前にダウンロードしておく必要がある．そのためには，デコーダをコンテンツと共に配信するのではなく，柔軟なスケジューリングを基にダウンロード処理を実現する必要がある．

2.2.2 システム構成及び処理手順

システム構成図及びシステムの処理手順を図 1 に示す．

具体的な処理手順は以下の様に行う (図 1 の (1) から (4) がそれぞれの処理に対応している)．

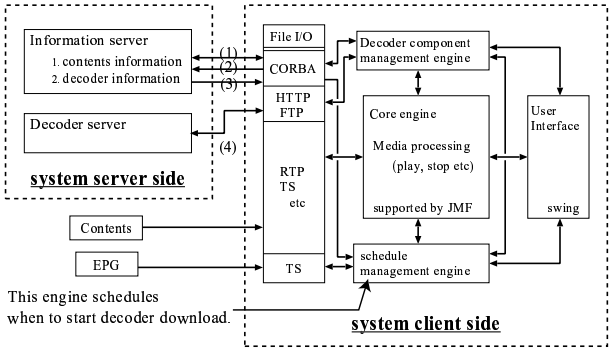

図 1　システム構成図・処理手順

(1) ダウンロード処理のスケジューリングを決定する際に利用する情報を取得する．クライアントシステムのスケジューラは，Information server にあるコンテンツ情報を解析し，クライアント側にデコーダが存在確認を行う．

(2) デコーダが存在しない場合，Information server に対して最適なデコーダがどこに存在するかデコーダ情報を問い合わせる．

(3) Information server は，デコーダ情報を検索し，クライアントに返す．

(4) Decoder information には，デコーダの URI が含まれている．クライアントは何時ダウンロードを始めれば良いかスケジューリングを行いながら，Decoder server よりデコーダをダウンロードする．

2.3 連続再生実現の為の課題

コンテンツの連続再生を実現するためには，配信モデル毎に様々な要因が絡んでいる．コンテンツ切り替え間で遅延が発生する原因を，次に示す．それぞれデコーダを十分に保存しておくだけの記憶容量が用意されているか否かで場合分けしている．

(1) Internet streaming(Pull 型)
  (a) PC 等 (記憶容量大)
  初期化遅延，ネットワーク遅延，バッファ遅延，デコーダ切り替え遅延
  (b) 携帯端末等 (記憶容量小)
  初期化遅延，ネットワーク遅延，バッファ遅延，デコーダ切り替え遅延，デコーダダウンロード遅延

(2) Internet streaming(Push 型)
  (a) PC 等 (記憶容量大)
  初期化遅延，ネットワーク遅延，バッファ遅延，デコーダ切り替え遅延
  (b) 携帯端末等 (記憶容量小)
  初期化遅延，ネットワーク遅延，バッファ遅延，デコーダ切り替え遅延，デコーダダウンロード遅延

(3) Broadcast(放送型)

(a) PC 等 (記憶容量大)
デコーダ切り替え遅延

(b) 携帯端末等 (記憶容量小)
デコーダ切り替え遅延，デコーダダウンロード遅延

それぞれの遅延は次の通りである．

- 初期化遅延 サーバがセッションを提供するまでに要する遅延
- ネットワーク遅延 コンテンツが配信されてからクライアントまで到達するのに要する遅延
- バッファ遅延 インターネットにおけるジッタを吸収する為のバッファに蓄えるのに要する遅延
- デコーダ切り替え遅延 コンテンツにあわせてデコーダを変更するのに要する遅延
- デコーダダウンロード遅延 デコーダをダウンロードするのに要する遅延

コンテントオリエンティッド符号化環境下において連続再生を実現する為には，以上の遅延を考慮しなければならない．

3 Flexible decoder

3.1 デコーダダウンロード時間削減

デコーダダウンロード時間は，コンテンツ間の連続再生を実現する上で 1 つの問題である．その解決手法として，デコーダ単位ではなく，デコーダを構成する機能単位で交換できる仕組みの提供が求められる．デコーダ間で機能を共有化することで，ダウンロードしなければならないデコーダファイルサイズ量を削減し，デコーダダウンロード時間を減少させることが可能である．

3.2 デコーダ構成

デコーダのモジュール化を図るためには，再利用性が向上するように機能分離する必要がある．デコーダ構成の分離については Syntax Parsing (シンタックス解析部), Data Restorer (データ復元部)，Image Reconstruction (イメージ構成部) の 3 つの処理に分離する方法が提案されている [4]．この方法では，シンタックスが変更された場合は Syntax Parsing，復号アルゴリズムが変更された場合は Image Reconstruction を個別に更新すれば良く，柔軟なデコーダを開発できる．

3.3 MSDL-S コンパイラ

MSDL-S[5] のコンパイラの例としてとして，Flavor[6] があるが，現在一般に開発されているこれらのプログラムはシンタックス解析部に対応したソースコードだけが生成される．したがって，MSDL-S を用いてデコーダを設計する場合，生成されたコードに対して，独自にデータ復号処理を追加する必要がある．しかし，これによりシンタックス解析部とデータ復元部とのインターフェースが独自に定義されてしまい，シンタックス解析部を他のデコーダから共有化することが不可能となる．

3.4 Flexible Decoder 作成手法

シンタックス解析部とデータ復元部のインターフェースを一意に定めるには，コンパイル時にシンタックス解析部とデータ復号処理部間のインターフェースを生成するようにコンパイラを改善する．すなわち，MSDL-S を単にシンタックス解析モジュールのためだけでなく，データ復元部のインターフェースを定義する IDL(Interface definition language) として利用する．

MSDL-S では，関連した情報の集まりである 1 つの層を 1 つの class として定義する．Flavor では class 毎に対応した C++/Java におけるシンタックス解析モジュールを生成する．したがって，データ復元部も各層に対応するように生成される (図 2)．

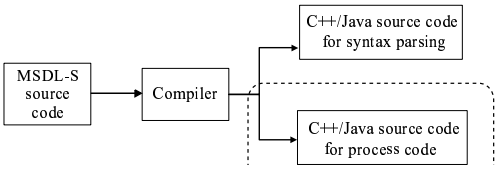

図 2　Compiler model for Flexible decoder

この時，デコーダ開発者は既に符号化で使用したアルゴリズムに対応したイメージ構成部が提供されている場合，それを利用してデータ復元部を作成する．もし，提供していない場合は各自アルゴリズムに対応したイメージ構成部モジュールを提供する．

3.5 問題点

上記手法を適応することで，データ復元部に実際のデコード処理でなく，検索システムのアルゴリズムを記述すれば，デコーダと検索システムで同じシンタックス解析モジュールの共有化が実現できる．

問題点として上記枠組みだけでは実時間処理を記述できない点が挙げられる．実際のデコーディング処理はメディア同期などの実時間処理を考慮に入れなければならない．そのためには，低レベルシステムを用意し，そこから呼び出す API(フレーム出力等) を一意に決定しておく方法が考えられる．

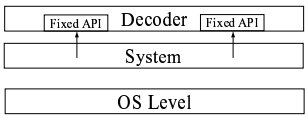

図 3　実時間処理の導入

4 まとめ

コンテントオリエンティッド符号化実現のためのシステムとして，Decoder Downloadable System を提案した．また，コンテンツ間の連続性実現の為の問題点について指摘し，デコーダダウンロードに要する遅延削減を目的として，Flexible decoder のデコーダ構成法について述べた．

参考文献

[1] O. Nakagami, N. Shimizu, T. Miyazawa, W. Kameyama, H. Watanabe, H. Tominaga, “A study on content-oriented coding scheme and decoder downloadable system,” World Multiconference on Systemics, Cybernetics and Informatics (SCI) 2002, July 2002, to appear.

[2] D. Wu, T. Hou, W. Zhu, Y.-Q. Zhang, J. M. Peha, “Streaming Video over the Internet: Approaches and Directions,” IEEE Trans. on Circuits and Systems for Video Technology, Feb. 2001.

[3] 宮澤，清水，亀山，渡辺，富永，“Pull 形サービスにおけるデコーダダウンローダブルシステムの構成及び評価,” PCSJ，2001

[4] H.Arakawa, T.Maeda, M.Etoh, “Software architecture for flexible and extensible image decoding,” Signal Processing: Image Communication 10, No.1-3, July 1997, pp.235-248

[5] ISO/IEC JTC1/SC29/WG11，“ISO/IEC 14496-1”

[6] A.Eleftheriadis, “Flavor: A Language for Media Representation,” Proceedings, ACM Multimedia 97 Conference, Seattle, WA, November 1997