VOD向けのネットワークAPIの提案

6Bb−5

河内谷 清久仁　根岸 康　田胡 和哉

日本アイ・ビー・エム(株) 東京基礎研究所

1 はじめに

ATMなどに代表される通信媒体の性能向上により，通信全体におけるプロトコル処理の割合が大きくなってきている．特に，ビデオ・オン・デマンド(VOD)システムでは，転送されるデータ量が大きいため，ネットワーク処理においてなるべく不要な処理が行なわれないことが重要である．そのための方式として，通信機構内での不要なデータコピーを減らしたいわゆる「コピー無し通信方式」が提案されている．しかし，従来の方式では，送受信データの格納場所がシステム指定のものであることなどにより，アプリケーションが使用するために結局データコピーが必要になる場合がみられた．この問題を解決するため我々は，アプリケーション側が送受信データの格納場所を指定可能な，新しいネットワークAPIを考案した．本稿では，このAPIについて報告する．

2 データコピーによるオーバヘッド

信頼性のある通信を実現するためには，通信でエラーが生じた際に，エラーがあったデータを再転送できるよう，通信の正常終了が確認されるまで，送信側で通信データを保持する必要がある．また，使用される通信プロトコルによっては，ヘッダの操作などのために通信機構に都合のよい形式でデータを保持したいなどの要求もある．そのため一般に，送信要求時に通信データをアプリケーションのバッファから通信機構のバッファにコピーすることが行なわれる．しかし最近では通信メディア自体の速度がかなり向上しており，通信全体でこのデータコピーが占めるオーバヘッドが無視できなくなってきている[1]．この問題を解決するために，図1に示すような，アプリケーションと通信機構との間で通信用バッファを共有しデータコピーを無くした「コピー無し通信方式」が提案されている[2, 3, 4]．

しかし，この場合の問題は，データの格納場所である「通信用共有バッファ」の確保の方式である．このバッファ領域をアプリケーション側から指定できない場合，結局データコピーが必要になる場合が発生する．例えば，図2は，我々が試作中のVODシステム[5]におけるデータ転送を示したものである．このシステムにおけるビデオデータの流れは，以下のようになる．

1. ディスクサブシステムからデータを読み出す．
2. 読み出したデータを，通信機構に渡す．
   (通信機構によりデータが転送される)
3. 通信機構からデータを受け取る．
4. 受け取ったデータを，ビデオ表示機構に渡す．

上記のステップ1において，データは必ずしも任意のメモリ領域に読み出せるとは限らない．例えば，ディスクサブシステムが確保したメモリ領域にしか読み出せない場合が考えられる．また，ステップ4でも，ビデオ表示のためには特定のメモリ領域(たとえば，ビデオ表示ハードウェア上のメモリ)にデータを置かなければならないことが多い．つまりこのような状況では，アプリケーション側が指定したメモリ領域のデータを直接送受信できない限り，結局アプリケーションによる「通信用共有バッファ」に対するデータコピーが必要になってしまうということである(図2の2, 3の矢印)．

A New Network API for VOD
Kiyokuni KAWACHIYA, Yasushi NEGISHI, and Kazuya TAGO
IBM Research, Tokyo Research Laboratory
1623-14, Shimotsuruma, Yamato, Kanagawa 242, Japan
E-Mail: <kawatiya@trl.ibm.co.jp>

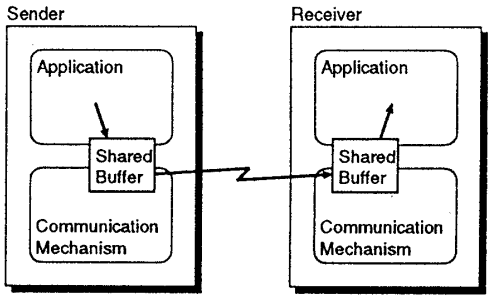

図1: コピー無しの通信機構

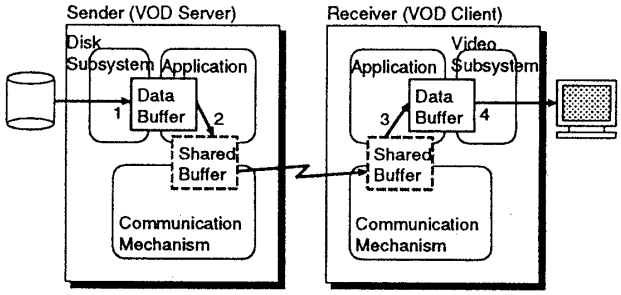

図2: VODシステムにおけるデータ転送

3 VOD向けのネットワークAPI

我々は，VODでの使用を第一目標とした，新しい通信機構の開発を行なってきている[6, 7]．この通信機構では用途をVODに限定しており，従来のネットワークAPIに縛られる必要がない．そこで，上述の問題点を解決できるような新しいAPIを採用し，性能の向上を図っている．このAPIにより，通信用共有バッファの管理を「アプリケーション側が用意した関数」により行なえるようになる．これにより，アプリケーションに都合のよいデータ構造，同期手段のままでコピー無しにデータ通信を行なえるようになる．通信機構側からは，これらのバッファ管理関数が必要に応じて呼び出される[1]．

表1に，このAPIで使用される関数を示す．BUFALLOC()とBUFFREE()はアプリケーションが用意する，バッファ管理のための関数で，必要に応じて通信機構側からも呼び出される．BUFALLOC()は，*sizepで指定された大きさのバッファを確保し，*bufpにそのアドレスを返す．*sizepに−1を指定した場合，適当な大きさのバッファが確保される．実際に確保されたバッファのサイズは*sizepに返され，確保したバッファのIDが*bufidpに返される．BUFFREE()は，bufidで指定されたバッファを解放するための関数で，これにより，通信完了の同期が行なわれることになる．これらの関数により，アプリケーションが自分に都合のよいバッファ管理を行なうことができる．送信側と受信側でそれぞれ異なるバッファ管理を用いることもできる．

BUFSEND()とBUFRECV()は通信機構が提供する，データ送受信のためのAPI関数である．BUFSEND()は，アプリケーション

[1] このようにシステムがアプリケーション内の関数を呼び出す場合には通常，呼び出しのオーバヘッドやセキュリティ上の問題を解決する必要がある．しかし，我々が提案する通信機構は核外のライブラリプログラムで実装されており，これらの問題は生じない．

アプリケーションが用意する関数

バッファ割り当て	int BUFALLOC(char **bufp, int *sizep, int *bufidp)
バッファ解放	int BUFFREE(int bufid)

通信機構が提供する API 関数

バッファ送信	int BUFSEND(char *buf, int size, int bufid)
バッファ受信	int BUFRECV(char **bufp, int *sizep, int *bufidp)

表 1: VOD 向けの API で使用される関数

が BUFALLOC() で確保したバッファbuf 中の大きさ size のデータを送信する．この関数は送信完了を待たずにリターンし，送信完了は通信機構から対応する bufid を引数とした BUFFREE() が呼ばれることで通知される．BUFRECV() は，受信したデータをアプリケーションが受け取るための関数である．通信機構はあらかじめ BUFALLOC() により受信用のバッファを確保しており，データは直接このバッファに受信される．データが受信されると，そのバッファのアドレスが*bufp に，サイズが*sizep に返される．

4 通信機構の動作

これらの関数を用いた，通信機構の動作の様子は以下のとおりである (図 3)．
まず，送信側は以下のように動作する (図 3左)．

1. アプリケーションは，コネクション確立後，通信機構に対し BUFFREE() 関数を登録する．
2. アプリケーションは，(BUFALLOC() で) 送信用共有バッファを確保し，そこにデータを用意する[2]．
3. アプリケーションは，BUFSEND() で通信機構に対しバッファ内のデータの送信を指示する．この関数は，送信完了を待たずにリターンする．
4. 通信機構は，渡されたバッファから直接 (コピー無しに) データをネットワークに送り出す．
5. 送信完了後，通信機構は BUFFREE() を呼び出す．
6. アプリケーションは，この呼び出しでバッファの解放タイミング (送信完了) を知る．

次に，受信側は以下のように動作する (図 3右)．

1. アプリケーションは，コネクション確立後，通信機構に対し BUFALLOC() 関数を登録する．
2. 通信機構は，BUFALLOC() を呼び出して受信用共有バッファを確保し，データの到着を待つ．
3. データが来たら，通信機構は確保したバッファに直接 (コピー無しに) データを受け取る．
4. アプリケーションは，BUFRECV() でデータの受信を要求する[3]．
5. 通信機構は，受信用バッファにデータが用意できるまで待って，それを返す．
6. アプリケーションは，バッファ中のデータを使用し，使い終わったら (BUFFREE() で) バッファを解放する．

図 4に，この通信機構を用いた場合の VOD システムのデータ転送を示す．提案する通信機構では，アプリケーションが指定した任意のメモリ領域に対して直接データの送受信が行なえるため，図 2と比較して，アプリケーション内でのデータコピーも不要になっていることがわかる．

[2]この部分は，「データが用意されたメモリ領域を得る」というまとまった処理でもかまわない
[3]受信側のステップ 2, 3とステップ 4の順序は受信要求のタイミングによって入れ替わる場合がある．

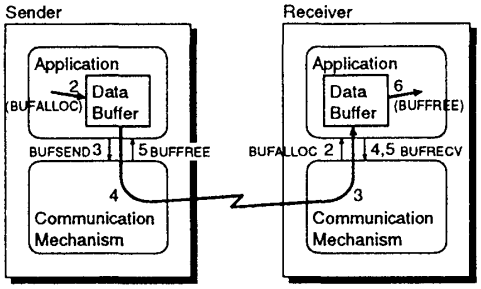

図 3: 提案する API による通信機構の動作

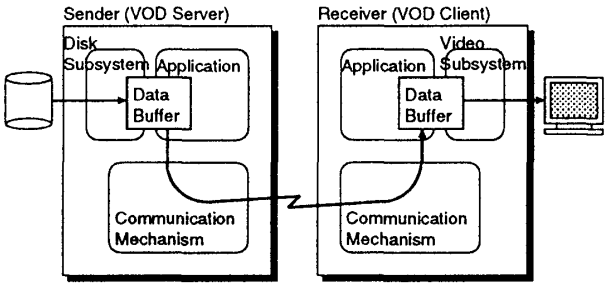

図 4: 提案する通信機構による VOD システム

5 おわりに

本稿では，VOD システムでの使用を考慮した新しいネットワーク API の提案を行なった．この API により，アプリケーションにとって都合のよいバッファで直接データの送受信が行なえるようになる．この方式では，通信機構との間で不要なデータコピーが行なわれないため，オーバヘッドが少ない通信が行なえる．これは，転送されるデータ量が大きい VOD システムにおいては特に有効である．また，バッファ管理の機構が送信，受信側ともにユーザに開放されており，汎用性と拡張性にもすぐれている．

参考文献

[1] D. D. Clark et al.: "An Analysis of TCP Processing Overhead," *IEEE Communications Magazine*, Vol. 27, No. 6, pp. 23-29 (1989).

[2] 加藤 他: "OSI プロトコル実装のためのユーザデータをコピーしないバッファ制御方式," 情処研報 93-DPS-62, pp. 95-102 (1993).

[3] 根岸: "通信インタフェースとオーバヘッドに関する考察," 第 48 回情処全大論文集, 4F-3, pp. 4-75–4-76 (1994).

[4] H. Kitamura et al.: "A New OS Architecture for High Performance Communication over ATM Networks — Zero-copy architecture —," *Proc. 5th Intl. Workshop on Network and Operating System Support for Digital Audio and Video*, pp. 87-90 (1995).

[5] 河内谷 他: "ATM を使った VOD システムの試作," 第 51 回情処全大論文集, 6F-7, pp. 1-227–1-228 (1995).

[6] 田胡 他: "既存の通信プロトコルを利用して実現した VOD 向け通信機構," 第 7 回コンピュータシステム・シンポジウム論文集, pp. 149-156 (1995).

[7] Y. Negishi et al.: "A Portable Communication System for Video-on-Demand Applications using the Existing Infrastructure," *Proc. IEEE INFOCOM '96*, to appear (1996).