PDE-II における実時間同期のための周期スレッドの提案

1H－8　吉川耕平†　岡村耕二　荒木啓二郎
奈良先端科学技術大学院大学　情報科学研究科
（†シャープ株式会社技術本部より派遣留学中）

1. はじめに

PDE(Parallel&Distributed Environment)-II では、複数の表現メディアデータを交換する際の処理の連続性や複数処理間の同期を、処理の質 (QoS) と考え、その保証機構を、ハードウェアに依存しないマルチメディア処理機能として OS に取り込むことを目的としている[1]。同期としては、テレビ電話における口の動きと声との同期、マルチメディアサーバからのデータ再生における背景音、効果音、映像、テロップ間の同期等を考えており、処理の質の意味付けと、同期機構の実装を目指している。

筆者らは先に、ネットワークをはさんだ複数のエンティティ（アプリケーションやデバイス）間に張られたコネクション上を、表示の連続性及び同期という時間的制約を持つ表現メディアデータを伝送し、コネクション上に置かれた QoS ポイント（図1）でその制約を管理するモデルを提案している[2]。

複数の連続メディアデータの再生時における同期機構については、大域的な論理時刻 (Logical Time) を中心に据えたものが提案されている[3]が、我々は連続メディア／非連続メディア統合の際の拡張性を考え、各コネクションとそこを流れるデータに基づく同期機構を採用している。

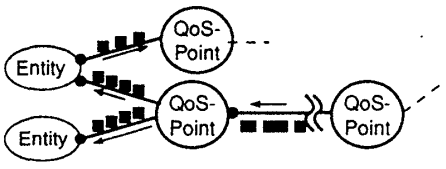

図 1: QoS ポイント

本稿では、提案したモデルの構成要素である QoS ポイントを実現する「周期スレッド」を提案し、そのプログラムインタフェースについて述べる。これはプログラマに、周期的な処理を容易に記述させるとともに、各処理の連続性や同期の QoS を管理するものである。

2. 周期スレッドの動作

QoS ポイントは、エンティティが持つコネクションごとの通信ポート（エンドポイント）とは別の、コネクション上に独立に存在するポートに相当するもので、相手エンティティ側の QoS ポイントとの間の通信、またはファイル／メモリ／デバイスなどのハンドリング

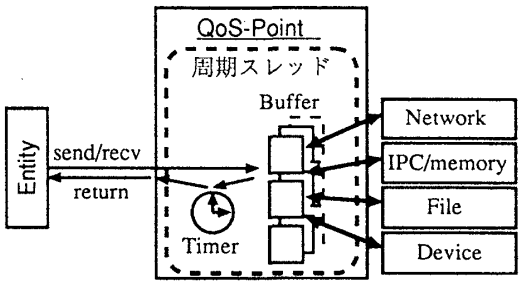

図 2: 同期ポイントの周期スレッドによる実現

を行ない、かつ連続性の監視や同期の修正を行なう自律的なプロセスという側面を持つ。我々はこれを、固有のバッファ領域と動作プログラム、タイマを持つ周期スレッドにより実現した（図2）。

周期スレッドの基本的な動作を説明する。

- 相手となる QoS ポイントとの間での通信や、ファイル, メモリ, デバイス等をハンドリングする。
- データの連続性と同期を監視するとともに同期のアルゴリズム[4]を用いて可能な範囲でその調整を行なう。
- 処理のサイクル時間を持ち、エンティティからのコール（読み出し／書き込み）に対し、次章に述べる動作を行なう。
- エンティティからのコールの返り値として連続性と同期の程度を報告する。
- 読み出し/書き込みのデータが足りない場合、必要に応じてダミーデータや補間データを生成する。

3. 周期スレッドのインタフェース

周期スレッドは以下の形で呼び出される。ここではメッセージパッシングの標準化案[5]を参考とした。

$$\begin{bmatrix} \sqcup \\ a \\ v \end{bmatrix} \begin{bmatrix} p \\ b \\ n \end{bmatrix} \begin{bmatrix} send \\ recv \end{bmatrix}$$

それぞれの動作を下で説明する。

i) 読み出し／書き込み時のデータ構造

- ⊔ 指定なし。バッファに順番に書き込む。
- a 配列指定。同じ型、サイズのデータを等間隔（引数指定）を取ってバッファに書き込む。
- v ベクトル指定。任意の構造（引数指定）に従って、バッファに書き込む。

"Periodic Thread for realtime data processing with synchronization in PDE-II"
K.Yoshikawa, K.Okamura and K.Araki
Graduate School of Information Science, Nara Institute of Science and Technology

ii) 同期

p periodic: サイクル終了時刻までリターンしない。必要なデータ量に対する供給し又は供給されたデータ量の割合と、他のコネクションとの同期の程度をリターン値として返す。

b block: 通常のソケットインタフェースと同様、サイクル時間とは無関係にスレッドの処理終了までリターンしない。通信の場合は相手からの ack を待つ。

n non-block: サイクル時間とは無関係に、コール直後にリターンする。

いずれも、コールの有無に関わらずサイクルは進むものとする。

iii) 読み出し／書き込み

send: エンティティから QoS ポイントへのデータの書き込み。

recv: エンティティの QoS ポイントからのデータの読み出し。

図 3に periodic 指定による同期を取ったデータ読み出し (p_recv) の動作例を示す。図は、コールの時点に関わらず、サイクル終了時点にのみ return することで、結果としてエンティティも周期的に動作し、またサイクル内に p_recv が来なかった場合はそのサイクルのデータが捨てられることを表している。

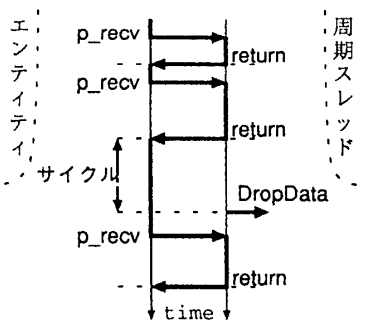

図 3: p_recv の動作

この他に、block/non-block 指定の処理（通信）により、従来の非連続メディアの処理も記述できる。

上のインタフェースにより、以下の利点が得られる。

i) ハードウェアに依存しない実時間同期処理の枠組を提供する。アプリケーションプログラムが、ハードの進歩による処理の高速化の影響を受けない。

ii) 実時間同期処理の記述が容易である。

- アプリケーションが QoS を管理する必要がない。プログラマは、実時間性が守られているか、他との同期が守られているかを、send/recv の返り値により得ることができる。
- 周期スレッドが、サイクル終了時刻までリターンしないため、上位プログラムも周期的な動作を行なう。

iii) 従来の非連続メディアの処理との統合ができる。

4. QoS ポイントの管理

モデル上、QoS ポイントは実際のデータ転送の有無とは独立に存在しうる。例えば、デバイスからの出力に周期性を与えるものや、受信者の有無に関わらず放送型サービスの提供を行なうもの等について、その存在を大域的なアドレスを用いて知らせておく。アプリケーション（エンティティ）は、相手となる QoS ポイントを指定し、自身の動作に必要な QoS ポイント（周期スレッド）の生成を要求することになる。

次に QoS ポイント生成の要求に対し、周期的な動作を可能な限り保証するため、相応のシステムリソース（CPU 時間, ネットワークバンド幅, バッファ領域等）を確保[6]する。確保に成功した場合、エンドポイントが返り値として渡される。

5. おわりに

マルチメディアに必須のデータ転送の連続性及びその同期の管理を行なう周期スレッドを提案した。これによりハードウェアに依存しない実時間同期機構が提供でき、またプログラマの時間的制約記述の負担を大きく軽減することができる。

今後は、周期スレッドの中で行なう通信以外の処理（ダミーデータ／補間データの作成等）について、処理時間の予測をいかに行ない、またいかにそれらをライブラリ化し、提供するかを検討していく。

さらに、連続／非連続メディアについて、より柔軟な同期機構の実現を目指していく。

参考文献

[1] 岡村, 吉川, 稲垣, 荒木, “PDE-II の概要 ～QOS に基づいたマルチメディア処理モデル～”, 情報処理学会第 4 8 回全国大会,1H-5, Mar. 1994.

[2] 岡村, 吉川, 稲垣, 荒木, “QOS 指定可能なマルチメディアモデルの提案”, 情報処理学会マルチメディア通信と分散処理ワークショップ, Nov. 1993.

[3] D. P.Anderson, G. Homsy, “A Continuous Media I/O Server and Its Synchronization Mechanism”, IEEE, Computer, Oct. 1991.

[4] 稲垣, 岡村, 荒木, “PDE-II におけるメディア間同期機構の実現に対する考察”, 情報処理学会第 4 8 回全国大会, 1H-6, Mar. 1994.

[5] Gropp, W., and Lusk, E., A test implementation of the MPI draft message-passing standard”, TR ANL-92/47, Argonne National Laboratory, Argonne, IL 60439, Dec. 1992.

[6] 松尾, 岡村, 荒木, “PDE-II におけるリソースのリザベーションに関する考察”, 情報処理学会第 4 8 回全国大会, 1H-7, Mar. 1994.