Keio-MMP におけるサウンドパフォーマ†

7R－4

平林 真実　萩野 達也

慶應義塾大学環境情報学部

1 はじめに

慶應義塾大学では，情報処理振興事業協会 (IPA) からの委託研究として 10 社の企業と共同研究として，分散マルチメディア統合環境プロジェクト (Keio-MMP プロジェクト) を行なっている [1].

本稿では，本プロジェクトで提案している実時間マルチメディアアーキテクチャであるコンダクタ/パフォーマモデルに基づいたサウンドサーバについて報告する.

2 コンダクタ/パフォーマモデル

本プロジェクトでは実時間 OS である Real-Time Mach3.0 [4] を使い，分散環境上のでマルチメディア処理のための実時間性の実現と QOS 制御を行なえるようなシステムの研究を行なっている.

この中でコンダクタ/パフォーマモデルに基づいたシステムの提案を行なっている [2]. コンダクタ/パフォーマモデルは，Real-Time Mach3.0 マイクロカーネル上でお互いに協調しながら動作するサーバ群によって構成される．サーバ群の中で全体を管理するためのサーバをコンダクタと呼び，各種のメディア処理を行なうためのサーバをパフォーマと呼ぶ．コンダクタはパフォーマが担当する各メディアの実時間制約を満たすようにパフォーマに対して処理の依頼を行ない，同時にパフォーマ間の通信のために必要な共有バッファ資源の管理を行なう.

サウンドパフォーマはコンダクタの指示にしたがった音声の入出力を担当するサーバである．したがってシステム全体としての実時間制約と QOS 制御についてはコンダクタによって実現されているため，パフォーマではデバイスに関連するようなローカルな QOS 制御を行なうことになる.

3 サウンドパフォーマの目的

音を扱うためのパフォーマとしての必要な条件として，

1. 他のメディアとの同期.
   タイムスタンプ付きのデータを指定した時刻に再生することにより他メディアとの同期をはかる.
2. 時間制約を満たす音の再生.
   指定した時間内に再生できなくなった場合への対応.
3. 滑らかな再生.
   音声の途切れが生じないようにバッファ内のデータ量の調整を行なう.
4. デバイス独立性.
   多くのサウンドボードに対して柔軟に対応できるようにデバイスに依存した制御を 1 つのモジュールにまとめる.
5. フォーマットの違いの吸収.
   実際のオーディオデバイスではサポートしているサンプリングレートやオーディオフォーマットが限定されるため，Performer 内でフォーマット変換を自動的に行なう.
6. 拡張性.
   リアルタイムに音声に対して加工を行なうためのモジュールの追加を可能にするための機構を用意する.

以上の点を考慮して開発を行なった.

4 構成

プログラム構成としては，図 1のような構成を行なうことにより，デバイス依存部とパフォーマとしての制御部を分離することで，他のサウンドボードやハードウェア構成にも柔軟に対応できるようにした.

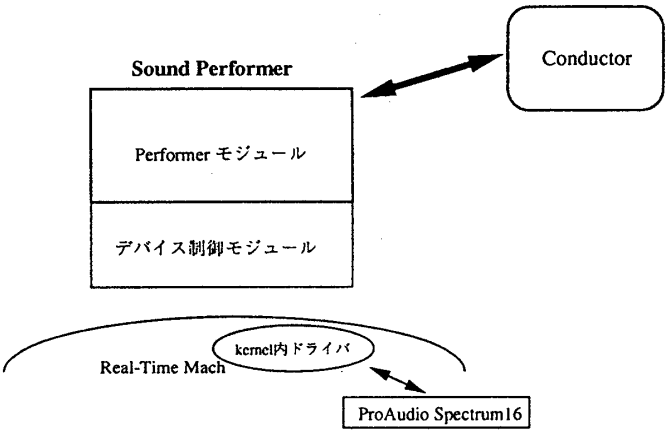

図 1: サウンドパフォーマの構成

デバイス制御モジュールは パフォーマ制御モジュールからの

- dev_open()
- dev_close()
- dev_read()
- dev_write()
- dev_get_status()
- dev_set_status()
- dev_command()

の各関数によって呼び出される.

呼び出された関数内でカーネル内デバイスドライバに対する入出力を行なう.

パフォーマの制御の主体はコンダクタ側にあり，全体の QOS 制御はコンダクタによって行なわれている．コ

"Sound Performer in Keio-MMP project"†
Masami Hirabayashi, Tatsuya Hagino
*Keio University, 5322, Endo, Fujisawa-shi, Kanagawa, 252 Japan

† この研究は，情報処理振興事業協会 (IPA) が実施している開放型基盤ソフトウェア研究開発評価事業「マルチメディア統合環境基盤ソフトウェア」プロジェクトのもとに行なわれた.

ンダクタ側との通信は，aus_open() によってサウンドパフォーマとの通信のためのポートを確保し，得られたポートに対する read/write をコンダクタ側の制御によって行なうことになる．

再生時にはコンダクタからはタイムスタンプ付きの音声データを受取り，パフォーマ内で時間を参照しながらサウンドボードへの出力を行なう．音声データには一連のデータであることを示すためにストリーム ID が付けられている．パフォーマ内でのタイムスタンプ参照の基準となる時間は，データ開始時からの相対時間とする．タイムスタンプがないデータについては，その場で再生される．

タイムスタンプが同じデータがある場合にはパフォーマ内でミキシングされ再生されることになる．このような処理はサウンドボードがそのための機能を持っている場合にはボードを使って行なうが，無い場合にはソフトウェアによって行なうことになる．

音声入力時にも，ある時刻からの相対時間をタイムスタンプとして付加しデータを作成する．

フォーマットの自動変換に使われるものと同様のインターフェイスを持つ関数群をリンクすることによって，パフォーマ内でリアルタイムに音声の加工を行なうことも可能である．

5 パフォーマ内の QOS 制御

コンダクタによるシステム全体の QOS 制御とは別に，デバイスを管理しているパフォーマにもローカルな QOS 制御が必要となる．デバイスの状態を監視しながらの内部的な QOS 制御とその状態をコンダクタに通知するのがパフォーマで必要とされる QOS 制御機構の主な点である．

パフォーマ内での QOS 制御は，内部バッファの状態を監視しながらバッファが溢れそうになったときに聴覚上違和感が生じない範囲での音声データの間引きを行ないタイムスタンプの値と実再生時刻との差を或る程度の範囲内に抑えるための機構によって行なう．また，同時に時間制約を維持できなくなったことをコンダクタに通知する．更に新たなリクエストが生じたときに，パフォーマでこれ以上の処理を行なえないと判断した場合には，そのリクエストを拒否できるようなアドミッション制御も行なう．

6 実装

コンダクタ/パフォーマモデルに基づいたサウンドサーバであり，PC/AT 互換機に ProAudio Spectrum16 サウンドボードを使い実現した．

リクエスト処理用と音声の入出力用に 2 つの実時間スレッドを持ち，各スレッドはそれぞれのポートによって入出力を行なう．2つのスレッドによりリクエスト処理と音声データ入出力を分担することによって迅速な処理が要求されるリクエスト処理をデータ量が多い音声入出力に影響を与えずに処理することができる (図2参照)．

リクエスト処理スレッドではポート経由でコマンドを受取り必要な処理を行なう．データ入出力スレッドではコンダクタから音声データを受取り，タイムスタンプを参照しながらサウンドボードへデータを送る．データ入力時にはサウンドボードからの音声データにタイムスタンプを付けてコンダクタに渡す．

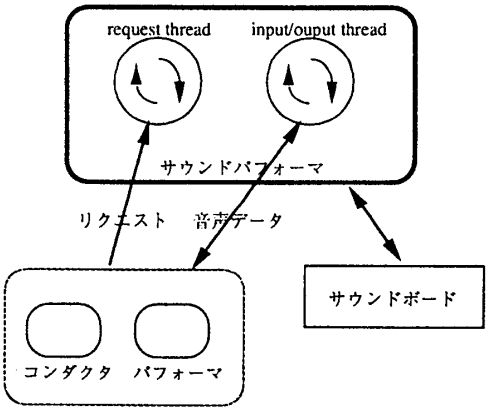

図 2: 内部構成

出力用データにはデータの情報を示すためのヘッダが付けられているため，そのヘッダ情報から出力するために必要な処理を判断し，フォーマット変換などが行なわれる．

7 今後の予定

サウンドパフォーマへの機能追加として，リアルタイムで音のエフェクトを行なえるモジュールの開発と他のサウンドボードへの対応を行なう予定である．また，サウンド関連パフォーマとして QuickTime 2.0 で実現されているような自分自身でサンプリング音源を持つような音源パフォーマの開発をする予定である．

謝辞

本研究を行なうにあたり協力／助言していただいている慶應義塾大学「マルチメディア統合環境基盤ソフトウェア」プロジェクトの皆様に感謝いたします．

参考文献

[1] 徳田，萩野，斎藤: “分散マルチメディア統合環境 Keio-MMP プロジェクトにおける連続メディア処理のためのソフトウエアアーキテクチャ,” 第 49 回情処全大 7R-01 (1994).

[2] 西尾，多田，徳田，萩野，斎藤: “Keio-MMP における Conductor/Performer アーキテクチャの協調性能評価,” 第 49 回情処全大 7R-07 (1994).

[3] 河内谷，緒方: “Real-Time Mach 用のオーディオドライバ,” 第 49 回情処全大 7R-03 (1994).

[4] H. Tokuda, T. Nakajima and P. Rao: “Real-Time Mach: Towards a Predictable Real-Time System,” USENIX Mach Workshop, pp.73–82 (1990).

[5] 持田，他: “Real-Time Mach 3.0 における I/O サーバの構成と評価,” 第 48 回情処全大論文集, 1H-2, pp. 4-19–4-20 (1994).