

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ェアアーキテクチャ†

7R-1

徳田 英幸 萩野 達也 斎藤 信男

慶應義塾大学環境情報学部

1 はじめに

慶應義塾大学環境情報学部における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)は、1992年からスタートし、参加企業10社とともに、分散マルチメディア環境のソフトウェア基盤に関する基礎技術やその環境を実現するためのプロトタイプを開発し、評価実験を行なってきた。本プロジェクトでは、連続メディア情報を扱うための実時間カーネルや実時間プロトコルの基本機能、連続メディアストリームの同期や協調制御を行なうメディアサーバ、連続メディア情報を蓄えておくメディアベースの研究開発を行なっている。本論文では、Keio-MMPにおけるこれらのソフトウェアアーキテクチャについて概説する。

2 分散マルチメディア統合基盤環境

マルチメディアシステムは、単にパーソナルコンピュータにCD-ROMとスピーカやマイクロホンを接続しただけの構成から、ワークステーションなどをネットワークで接続し、遠隔地の機器を結んで構成されたシステムなど様々なアプリケーションが開発されてきている。ここでは、分散マルチメディアシステムとして、ネットワークに接続されたワークステーション、モバイルホスト、PDAなどがデジタルビデオ/オーディオなどの連続メディア情報を扱うシステムを考え、以下に、これらのシステムを構築していく上でいくつかの技術的な課題を述べる。

2.1 いくつかの技術的な課題

- 動的拡張/変更可能なアーキテクチャ
従来からの分散マルチメディアシステムは、ネットワークに接続されたワークステーションだけを主な対象としているが、モバイルホスト、PDAや組み込みシステムなどの幅広いハードウェアプラットフォームを考慮した拡張可能なシステムアーキテクチャが必要である。モバイルホストやPDAなどでは、アプリケーションに最適化されたシステムソフトウェアの構成をカスタマイズできなければならない。また、ネットワーク接続などもホストの使用環境に応じて高速なネットワークから無線LANなどに動的に切り換えることをサポートできることが要求される。
- 解析可能なリアルタイム処理
連続メディアを処理する上では、単にメディア処理の即答性や応答性を高めるだけでなく、解析可能なリアルタイムスケジューリング機能、リアルタイム同期機能、リアルタイム通信機能などを提供することが必要である。UNIX系のシステムでは、従来からタイムシェアリングポリシィが一般的に利用されてきたが、連続

メディアデータの処理中に、他のユーザプロセスの起動によってプロセッサが横取りされてしまうと、画像や音声の再生が正しく行なえないことが生じるだけでなく、周期的なアクティビティの制御や一時的な過負荷な状況への対応等も難しい。リアルタイム同期に関しては、単一メディアストリーム内での同期だけでなく、複数メディアストリーム間での同期を保証できなければならない。リアルタイム通信に関しても、単に高速のネットワークプロトコルに対応するというだけでなく、時間的な制約と同時にメディア処理に対するサービスの質(QOS, Quality of Service)を保証できる機能が要求される。

- QOSに基づく資源管理

分散マルチメディアシステムでは、アプリケーションが要求するサービスの質(QOS)に基づく資源管理をシステムソフトウェアの各レイヤが協調しながら行なう必要がある。例えば、ビデオ会議システムなどにおいて、ネットワークを介してビデオストリームなどを処理する場合、プロセッサのスケジューラやネットワークプロトコルモジュールなどが、指定されたQOSパラメータに基づいて、必要なCPU、メモリ、ネットワーク資源などを動的に制御する必要がある。また、QOSの指定に関しても、各レイヤ間でパラメータ数やそれらの単位が異なる場合が多く、パラメータのマッピングが必要であり、協調した資源制御が必要である。

- 過負荷状態における安定性

分散マルチメディアシステムでは、システム内で起動している複数のアプリケーション間でいろいろな資源の競合が発生し、一時的な過負荷状態に陥る場合が多い。このような過負荷状態においては、時間的制約を持つ連続メディアの処理が正しく実行されず、ビデオやオーディオの再生などに不都合が生じてしまう。QOS制御と同様に、新しいアプリケーションやセッションの起動などに対するアドミッション制御や時間的な制約が満たされなかった場合の対処機能を提供し、システム全体の安定性を確保する必要がある。

3 Keio-MMPのソフトウェアアーキテクチャ

3.1 システム構成モジュール

Keio-MMPプロジェクトでは、分散マルチメディア統合環境を種々のハードウェアプラットフォーム上で構築し、システム構成要素間での垂直統合をめざすために、Real-Time Mach3.0マイクロカーネル[7]をベースに図1に示されるようなソフトウェアアーキテクチャを開発している。

Keio-MMPのアーキテクチャでは、UXサーバ(UNIX)、MDOSサーバ(MS-DOS)などのOSパーソナリティモジュールだけでなく、モバイルホストやPDAなどをサポートするためのRTS(Real-Time Server)やNPS(Network Protocol Server)サーバが実装されている。プロトコルに関しては、STIIやMPCといった連続メディアストリー

†Software Architecture for Distributed Multimedia Computing Environment in the Keio-MMP Project†

Hideyuki Tokuda, Tatsuya Hagino, Nobuo Saito
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。

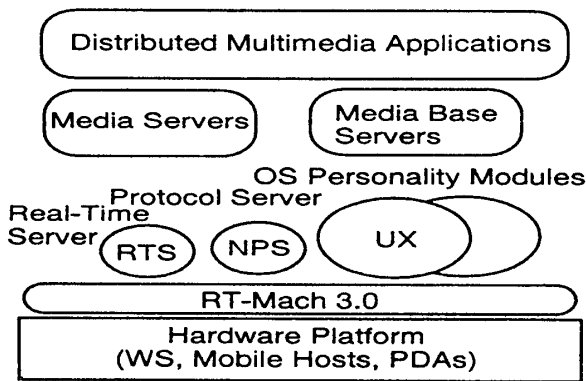


図 1: Software Architecture for the Keio-MMP project

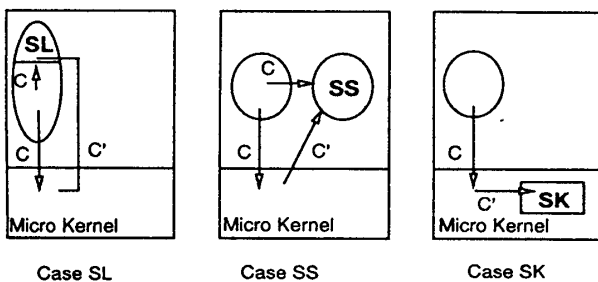


図 2: Placement of System Services

ム向きプロトコルを開発している [1]。また、連続メディア処理を支援するための統合メディアサーバ、メディアベースサーバなどもサーバ群として実現されている。サーバ間でのデータのコピーによるオーバーヘッドを抑えるために、サイクリック共有バッファ方式 [2] を実装している。

一般に、マイクロカーネルに基づくシステムアーキテクチャでは、柔軟にシステム機能の拡張が実現できる利点がある。拡張すべき機能を実装する場所としては、基本的には、図 2 に示すように、ライブラリ内 (SL 方式)、サーバ内 (SS 方式)、カーネル内 (SK 方式) の 3 つが考えられる [8]。

SL は、新しい機能をライブラリとして実装する方式で、SS に比べて高速な実装が可能であるが、ユーザプロセスと同一アドレス空間であるので信頼性が低い。SS は、ユーザプロセスとは独立したアドレス空間に実現されるので信頼性が高く、変更も容易であるが、コンテキストスイッチや IPC のオーバーヘッドが増加する。SK は、従来からの方式で、速度的には、SS より有利であるが、カーネル肥大化をさせてしまう点やカーネルに組み込まれてしまうため、動的な変更が難しくなる。

現在の実装では、種々のスケジューリングポリシーなどは、スケジューリングメカニズムのモジュールと分離し、マイクロカーネル内に実現されている。また、各種のプロトコルモジュールは、プロトコルサーバとして実現されている。リアルタイムスレッドなどの機能は、カーネル内で実現されている機能であるが、ユーザレベルでのリアルタイムスレッドパッケージとしても実装している。

3.2 Real-Time Mach3.0 マイクロカーネル

Real-Time Mach3.0 マイクロカーネルは、CMU で開発された MK83 版に様々なマルチメディア対応の拡張を

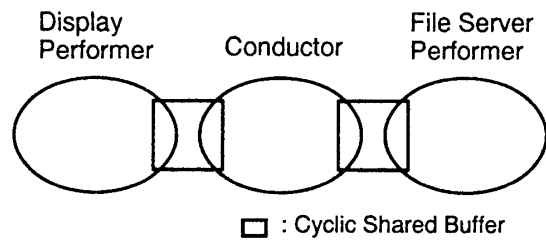


図 3: An Example of Conductor and Performers

行なっている版である。特に、リアルタイムスレッドの拡張 [3, 4] やモバイルホストに対応したパワーマネジメントサーバ [5] などの拡張が行なわれている。また、CPU 資源管理に関しては、ユーザスレッドごとに一定容量を予約できるプロセスリザベーションの機能も開発されている。

3.3 メディアサーバ

統合メディアサーバは、コンダクタ・パフォーマンスモデル [2] に基づいて、RT-Mach 上のサーバ群として構成されている。コンダクタは、セッションの生成、メディアストリームの同期制御、動的 QOS 制御を行ない、パフォーマンスは、各メディアに対応した個別の処理を行なう。図 3 に簡単な構成例を示す。

3.4 メディアベースサーバ

メディアベースサーバは、マイクロカーネルの提供している外部メモリマネージャの機能を利用し、複数のユーザレベルサーバとして実装している。連続メディアファイルユーザ空間内にマップし、アプリケーションのアクセスパターンに基づいて先読み処理を行って処理の効率化を図っている [6]。

4 おわりに

Keio-MMP における分散マルチメディア統合環境を実現するためのソフトウェアアーキテクチャについて述べた。このアーキテクチャは、RT-Mach3.0 マイクロカーネルをベースに、マルチメディア対応の新しいシステム機能をサーバ群として実装している。システムの拡張性、柔軟性という面で優れているが、マイクロカーネル自体の高速化や、メディアサーバやメディアベースサーバなどサーバ群の効率良い実装に関しては、今後も改良する必要がある。

参考文献

- [1] 南部, 他: 連続メディアストリーム向きプロトコル, 信学技報, CS94-83 (1994).
- [2] 西尾, 他: "Keio-MMP における Conductor/Performer アーキテクチャの協調性能評価", 第 49 回情処全大論文集 (7R-7) (1994).
- [3] 河内谷, 他: "連続メディア処理のためのリアルタイムスレッドモデルの拡張", 第 48 回情処全大論文集 (1H-1) (1993).
- [4] 追川, 他: "連続メディア処理のためのスレッドメカニズム", 第 49 回情処全大論文集 (7R-6) (1994).
- [5] 緒方, 他: "モバイルコンピュータを使った分散マルチメディア処理環境の実現", 第 49 回情処全大論文集, (6S-9) (1994).
- [6] 多田: "実時間オペレーティングシステムにおける連続メディアファイルサーバの実験", システムソフトウェアとオペレーティングシステム, マルチメディア通信と分散処理合同研究報告 (1994).
- [7] H. Tokuda, T. Nakajima and P. Rao: "Real-Time Mach: Towards a Predictable Real-Time System", USENIX Mach Workshop, pp.73-82 (1990).
- [8] H. Tokuda, et al., "Towards a New Operating System Architecture: Micro Kernel vs. Reflective Architecture", 日本ソフトウェア学会全国大会 (1992).