

オブジェクトサーバを中心としたマルチメディア・アプリケーションのための 開発および実行支援環境

1 R-6

小坂 一也, 何 千山, 福田 剛志, 森本 康彦

日本アイ・ビー・エム株式会社 東京基礎研究所

1 はじめに

ネットワークで相互に結合された計算機間で自由にマルチメディアの情報を交換し合えるような世界が実現可能になってきたが、従来のようなデータフォーマットを規定するデータ交換の手法をこの世界にそのまま適用することには問題が生じてきた。データ交換は、いくつかの共通データフォーマットを規定して各アプリケーションが個別に対応することで実現される。ところがマルチメディアのような複雑なデータを扱い始めると、単なるデータ交換ではそれを処理するアプリケーション作成の負担が大きくなり過ぎて対応し切れなくなって来たのである。つまり、従来のデータ交換に代わる新しい枠組が必要となってきたのである。

現在、筆者らはこのような環境におけるマルチメディア・アプリケーションの開発および実行を支援するための環境 COSMOS (COmmon Service for Multimedia ObjectS) [1] を構築している。本稿では、本研究の第一段階として作成した最初のプロトタイプについて報告する。

2 オブジェクト指向アプローチ

前章で指摘したマルチメディアに対するデータ交換の問題点を整理すると以下の二点に集約される。

1. データ構造が複雑で共通フォーマットが存在しない
2. 各メディアに対して個々のアプリケーションが個別に対応するのが負担

COSMOS はオブジェクト指向によるアプローチでこの問題を解決しようと試みている。すなわち、データとそれを処理するための手続きをカプセル化したオブジェクトを情報の単位として交換することにより個々のアプリケーションの負担を減らし、また、マルチメディア用のクラス・ライブラリを提供することにより、より抽象度の高い交換規則を提供するのである。この方式は、新たなメディアの出現に対してもサブクラスの定義などで柔軟に対処できるという利点も持っている。さらに、それら交換可能なオブジェクトを格納するためにオブジェクトサーバを提供しており、マルチメディア・オブジェクトの統一的な管理を可能としている。

本アプローチの有効性を確認するために、筆者らは研究の第一段階として簡単なプロトタイプ・システムを作成した。この段階での主な目的は、実用性を重視した詳

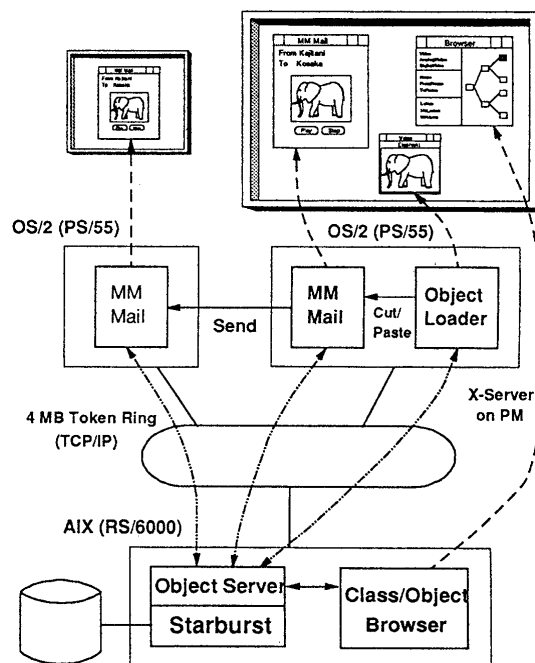


図 1: プロトタイプ・システムの構成

細な設計および実現ではなく、必要最小限の構成で迅速な実現を行ない、多数のアプリケーション開発者および使用者に提示してその反応を調べることである。

3 アプリケーションの選択

単なるデータ交換では対応が困難であり、かつ、我々のアプローチが効果的であるアプリケーションとして、マルチメディア電子メールを採用した。通常の電子メールは文字数値のみを対象にしているので、メールの受信者はだれでも容易にその内容を表示することができる。ところがマルチメディア電子メールを考えた時、データ交換では受信側のシステムが考えられる限りのメディアに対応することが現実的ではないことから、送信者が処理可能なメディアと受信者が処理可能なメディアが必ずしも一致するとは限らないのである。つまり、この種の問題に対処するためにはデータのみならずその処理プログラムも含めたオブジェクトを転送することが有効なのである。

4 プロトタイプ・システムの構成

システム全体の構成を図 1 に示し、主な構成要素を以下に説明する。

オブジェクト・サーバ クラス情報(メソッドを含む)とオブジェクトの内部情報を格納するために、拡張関係データベース Starburst[2] 上に TCP/IP の RPC を利用した簡単なオブジェクト・マネージャを実現した。将来的には C++ で記述したオブジェクトをそのまま永続的オブジェクトとして直接格納できるような言語インタフェースを提供する予定であるが、今回は後述するマルチメディア・ライブラリ中で直接永続性を持たせるためのコードを組み込んだ。

クラス・ブラウザ サーバ上で作成し、ユーザ・インタフェースの部分は OS/2* の X-server を利用した。オブジェクト・サーバと密接に関連しており、クラス階層や指定したクラスに属するオブジェクトのリストが表示できる。また、簡単な条件検索も可能になっている。オブジェクト・ローダ クラス・ブラウザからの指令によって、実際にオブジェクト・サーバからオブジェクトをクライアントにロードし実行(表示)する。また、カット操作により実行中のオブジェクトの ID を OS/2 ウィンドウ・システムのペースト・バッファに転送する機能を持つ。

マルチメディア・クラス・ライブラリ メディア・オブジェクトとして、アナログビデオと 3 次元グラフィックスを C++ で記述した。実行時にサーバから取り出してアプリケーションに結合出来るように、OS/2 のダイナミック・リンク・モジュールとして実現している。また、各クラスに固有の操作を行なうためのユーザインタフェースも実装した。

マルチメディア電子メール OS/2 上で動作する Smalltalk を利用して作成した。システムのペースト・バッファからオブジェクト ID を受取り、オブジェクト・サーバからオブジェクトをロードし実行することができる。また、ボタンを配置することができ、さらにそれぞれのボタンにスクリプト言語(Smalltalk)を記述できる。この機能によりオブジェクト・サーバに対して直接オブジェクトを検索したりすることができる。実現上特に工夫したのは、オブジェクト・サーバからロードしてきたオブジェクトを、あたかも Smalltalk のオブジェクトのようにスクリプト言語で記述できるようにしたところである。

実際に送信されるメールには、宛先や文章などの通常のデータに加えてペースト操作により張り込まれたオブジェクトの ID とそれらが配置される領域の情報のみが含まれており、オブジェクトの本体は受信したシステムが改めてオブジェクト・サーバからロードするような仕組みになっている。従って、受信者は受け取ったメールに含まれるオブジェクトにオブジェクト固有のユーザインタフェース等を通じて自由にメッセージを送ることができる。例えば、ビデオを再生したり、3 次元グラフィックスのオブジェクトを回転したりすることができる(図 2)。

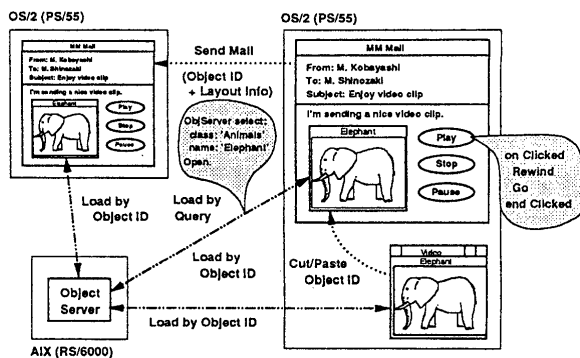


図 2: マルチメディア電子メール

5 むすび

ネットワーク環境におけるマルチメディア情報の処理に対して、我々はオブジェクト指向によるアプローチを採用し、そのアプローチの有効性を示すために迅速なプロトタイプを行なった。そして、実際に多数のアプリケーション開発者および使用者に提示してその反応を調べたところ以下の点について好評を得た。

- マルチメディアに対するオブジェクト指向アプローチの有効性
- オブジェクト・サーバによるメソッドまで含めたオブジェクトの共有管理
- 各オブジェクトに組み込まれたユーザインタフェース
- オブジェクトのスクリプト言語による簡単な操作

ただし、細部までの設計および実現がなされていないためシステム全体のパフォーマンスに関する評価はできなかった。また、メディアの種類が少な過ぎてクラス階層についての議論もできなかった。

本研究は現在第二段階にあり、最初のプロトタイプからの経験を生かしつつ、特に COSMOS 環境において中心的な役割を担うオブジェクト・サーバとマルチメディア・ライブラリに焦点を絞って詳細設計および実現を行っている。

参考文献

- [1] 小坂、梶谷、何、森本、福田: オブジェクトサーバとその応用, 情報処理学会データベースシステム研究会報告, 89-3, 1992.
- [2] G. Lohman et al, "Extensions to Starburst: Objects, Types, Functions, and Rules," Communications of the ACM, vol. 34, no. 10, pp. 94-109, Oct. 1991.