システムソフトウェアと
オペレーティング・システム 71−21
マルチメディア通信と分散処理 73−21
（1995. 12. 1）

MIO (Multimedia Information Organizer) システムの設計

大門英明† 岩崎量† 萩野達也††
† 慶應義塾大学 政策・メディア研究科
†† 慶應義塾大学 環境情報学部

我々が開発を進めている MIO (Multimedia Information Organizer) システムは、情報処理教育を行なう様々な場面での利用が可能な基盤システムである。MIO システムはシステムの核となる部分に、LISP 処理系を持ち、さらに、マルチメディアのデータも扱うことのできるオブジェクトの機構を持っている。そのため、MIO システムを使えば、マルチメディアのデータを使ったコースウェアの作成が容易に行なうことができ、さらに、要求に応じて自由に拡張することができる。本論文ではこの MIO システムの設計および実装の現状について説明する。

Design of MIO (Multimedia Information Organizer) System

Hideaki Ohmon†, Ryo Iwasaki†, Tatsuya Hagino††
†Keio University, Graduate School of Media and Governance
††Keio University, Faculty of Environmental Information

MIO (Multimedia Information Organizer) system is designed as a base system for education of information processing courses. MIO system has a LISP interpreter as its core. The interpreter includes object-oriented mechanism to handle multimedia data. Using MIO system, users can easily make multimedia courseware and add new functions to courseware on demand. In this paper, we explain the design of MIO system and current state of implementation.

1 はじめに

慶應義塾大学湘南藤沢キャンパス (Shonan Fujisawa Campus 以下 SFC と略す) は、1990 年に開設された比較的新しいキャンパスである。SFC では、開設当時より、その教育の柱として、コンピュータの使い方やプログラミングを教える情報処理教育を行っている。MIO (Multimedia Information Organizer) システムは、この SFC における情報処理教育を行う上での基盤となるようなシステムとして、1994 年の秋頃から開発を進めている。

本論文では、まず、このようなシステムを開発するに至った経緯について述べ、その後 MIO システムの設計概要について説明する。そして、現在のプロトタイプを説明し、最後に今後の課題についてまとめる。

2 SFC における情報処理教育

最初に述べたように、我々は MIO システムを SFC の情報処理教育を行なうためのシステムとして開発している。そこで、まず SFC における情報処理教育について簡単に説明する。

2.1 カリキュラム

SFC における情報処理教育のカリキュラムには、全員の 1 年生が履修する情報処理 I、主に 2 年生が履修する情報処理 II がある。

SFC のすべての学生は情報処理の導入として、必修である情報処理 I を履修する。情報処理 I では、最初にキーボードの打ち方、電子メイルの使い方等のコンピュータリテラシーを学び、その後 C 言語を

使った簡単なプログラミングを学ぶ。

情報処理Iに続いて受講することのできる情報処理IIには、本格的なプログラミングやコンピュータグラフィックス、コンピュータミュージック、論理プログラミング、統計処理、ネットワークシステムなどのコースがあり、学生は興味に合わせて選択することができる。

2.2 コースウェア

SFC の情報処理教育には、積極的にオンラインコースウェアを導入している。まず、1992 年度春学期にコンピュータリテラシーの教材として iplnote を導入した。これはテキストファイルを more コマンドを使って、見るだけの簡単なものであった。続いて同年秋学期からは、C 言語プログラミングの教材 iplohp を導入した。iplohp は OHP (Over Head Projector) をオンライン化したもので、モノクロではあるが、ビットマップイメージやアニメーションを多用している。図 1 は iplohp の画面例である。

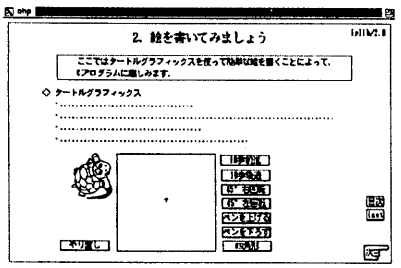

図 1: iplohp の画面例

その後数年は iplnote、iplohp の内容的な改良だけにとどめられていたが、1995 年春からはコンピュータリテラシーの教材として WWW を用いた iplht を導入した。テキストだけであった iplnote に代わって、グラフィックスや画面ダンプを多用し HTML で記述された教材を、学生が Mosaic で見るという形態にした。

3 コースウェアの要件

このように、SFC では今まで専用のコースウェアや既存のソフトウェアを組み合わて情報処理教育を行なってきたが、我々は、このような方法に限界を感じ始めた。そこで、新しく共通の基盤となるシステムを開発しようと考えるに至った。

ここで、我々のシステムについて説明する前に、情報処理教育に用いるコースウェアとそれを作成するシステムに、一般的に必要とされる機能を挙げる。

3.1 表示形式

プレゼンテーションソフトウェアやオーサリングツールなどの表示形式としては、カード型、本型、スクロール型、環境型の 4 つが考えられる。

カード型: 決まった大きさの 1 ページに情報を収める形式である。ページ単位の情報に注目させることができるため、プレゼンテーションソフトやオーサリングツールで一般的に用いられている。

本型: カード型と同じくページの大きさは一定だが、ひとつの単位の情報が 1 ページに収まっているとはかぎらない。本を模したインターフェースなので初心者でも使いやすい。一部のオーサリングツールやスケジュール管理ソフトなどで用いられている。

スクロール型: ページの大きさが一定ではなく、表示しきれない情報はスクロールバーにより画面をスクロールさせて表示する。Mosaic や NetScape などの WWW ブラウザに用いられている。

環境型: ある仮想的な空間を用意し、ユーザーが自由に探索できるようになっている。3 次元のインターフェースをもつものもある。一部の語学学習コースウェアやゲームで用いられている。

3.2 利用形態

コースウェアには多様な利用形態がある。ここではそれを講義、自習、参照、ブラウジングの 4 つ場面に分け、それぞれの場面でコースウェアに必要な機能を挙げる。

講義: 教師が学生に対して講義を行う場合、コースウェアにはオンラインプレゼンテーションを行なう機能が必要とされる。例えば、テキストを箇条書にしたり、説明しているテキストを目立たせる等の機能が必要だ。

自習: 学生が主体的に自習を行う場合、学生が各人の進度、興味に合わせて、自由にコースウェアのどの部分でも見られるようでなければならない。しかし、逆に、教師がいない場合でもコースウェアがガイドラインになれるよう一定の到達点に達しない学生が先に進むのを制御することもできることが望ましい。

参照: 分からない単語などが出てきた時に、オンラインで参照できると便利である。この場合には、必要な情報を見やすくまとめてあるとよい。また、いろいろな方法での検索を行なう機能も必要だ。

ブラウジング: 学習に際して、学生は常に明確な目的を持っているとは限らない。特に目的を持たず、コースウェアを眺めながら、興味のある情報を探す場面もある。この場合、多くの情報をまとめて見られるとよい。また、他の情報を自由に見られる機能が特に必要とされる。

表 1: 表示形式と適する利用形態

	講義	自習	参照	ブラウジング
カード型	◎	○	○	△
本型	×	○	◎	○
スクロール型	×	△	○	◎
環境型	×	○	×	◎

前述した表示形式と利用形態の向き不向きの関係をまとめると表 1 のようになる。講義には、プレゼンテーションが行ないやすい形態であるカード型が向いている。自習の場合は、自由に学習を進めるのであれば、どの形態であっても構わない。しかし、ガイドラインとして進度を制御する場合には、境界が分かりにくいスクロール型は不向きである。参照の場合は、調べたいことが簡単に参照できる仕組みが必要であるため、環境型は不向きである。ブラウジングの場合は、明確な目的を持たずに、情報を探すのであるから、1 情報 1 ページとなっているカード型は不向きである。

3.3 必要な機能

次に、コースウェアを作成するシステムに必要な機能を列挙する。

テキスト: どのような目的に使うにも、必要不可欠なものはテキストの表示機能である。任意の位置に表示できるだけでなく、箇条書や通常の段落形式に簡単に整形できる機能が必要である。

マルチメディア: 文字だけではなく、いわゆるマルチメディアの素材も扱える必要がある。簡単な直線や幾何図形、ビットマップファイルや画像ファイル、さらには音声や動画も必要である。

アニメーションや連動: 文字や図形のオブジェクトを静的に並べるだけではなく、オブジェクトが動くアニメーションや、オブジェクト間の連動などもできなければならない。このためには、オブジェクトの動作を細かく記述できる言語も必要である。

コラボレーション: 教育での利用では、教師と学生のコースウェアの連動や、学生同士の共同作業といったコラボレーションを支援する機能が必要である。さらに、サーバーによる成績や教材などの一元管理の機能も必要である。

汎用性: コースウェアの構成は、カード型でページ単位のもの、本のように連続した数ページに渡るもの、ハイパーリンクをもつものなどいろいろである。これらのどのような構成にも対応できなければならない。個別の目的には優秀な専用ソフトウェアが存在するが、すべてを同一の基盤で行えるものはない。

拡張性: どのように優れたコースウェアであっても、すべての利用者を満足させることはできない。そこで、利用者に応じた自由な拡張が行なえることが求められる。

価格: 教育現場では多くの台数のコンピュータにコースウェアをインストールすることになる。したがって、価格はできるだけ安いか、もしくは無料であることが望ましい。

4 MIO システムの設計概要

これらの要件を踏まえ、我々は次のような特徴を持つシステムとして MIO システムを設計した。

- 高い拡張性をもつ
- マルチメディアデータを扱うことができる

- ネットワークを利用したコラボレーションが行なえる

ここではその設計概要について述べる。

4.1 MIO LISP

高い拡張性を持つために MIO システムでは、拡張機能を記述することのできるプログラミング言語をシステムの基盤として提供する。これが MIO LISP と呼ばれる LISP 処理系である。MIO LISP は LISP 処理系としての機能に加えて、マルチメディアのデータやネットワークを利用するための機能を追加することのできる機構を持っている。

MIO システムと同様に LISP 系の言語を拡張機能の記述言語として採用しているシステムの代表的なものに GNU プロジェクトの emacs がある。emacs は電子メイル、電子ニュースの読み書きを行なったり、デバッガとして利用することができるが、このような拡張機能は emacs LISP と呼ばれる LISP のプログラムとして書かれている。つまり emacs は emacs LISP を拡張機能記述言語として持っているので、非常に拡張性が高く、単なるエディタとしてだけでなく、さまざまな用途で利用することができる。

4.2 オブジェクト機構

emacs の例を見てもわかるように、機能拡張言語として LISP 系のプログラミング言語を提供するのは、良いことである。しかし、LISP の場合処理の速度が問題になる。そこで、システムの拡張性と実行速度という相反するものをうまく調和させるために、MIO LISP にオブジェクト機構を追加している。

MIO システムのすべてのオブジェクトはクラスに属するインスタンスとして生成される。MIO におけるクラスとはオブジェクトがどのような性質を持つかを記述したものであり、実際には、メッセージを受けとった時にどのように振舞うかが定義してあるメソッドの集合である。

新しいインスタンスを生成するには、組み込み関数 make-instance を利用して以下のように記述する。

```
(make-instance クラス名 引数1 引数2 …)
```

次にクラスの定義方法について説明する。MIO のオブジェクトのクラスには MIO Core Object クラスと MIO User Object クラスの 2 種類のクラスが存在する。MIO Core Object クラスはシステムに組み込まれるため C 言語で記述し、MIO User Object クラスは MIO LISP で記述する。MIO User Object クラスを定義するためには組み込み関数 define-class を利用して以下のように記述する。

```
(define-class
  (クラス名 引数名1 引数名2 …)
  本体1 本体2 …)
```

make-instance を行なうと引数がここで定義された引数名にバインドされ、その後本体が順番に実行されて新しいインスタンスが作られる。

クラスのメソッドを追加するには組み込み関数 define-method を利用して以下のように記述する。

```
(define-method クラス名
  (メソッド名 引数名1 引数名2 …)
  本体1 本体2 …)
```

インスタンスにメッセージを送ると定義されたメソッドの中から適当なものを探し出し、引数を引数名にバインドし、本体を順番に実行する。インスタンスそのものを引数にしたい場合には self という引数を利用することができる。

前述したように MIO システムには MIO Core Object クラスと MIO User Object クラスという 2 種類のオブジェクトのクラスがある。以下ではそれぞれについて説明する。

4.3 MIO Core Object クラス

MIO Core Object クラスは、システム組み込みのオブジェクトのクラスである。オブジェクトがオペレーティングシステムやウィンドウシステムの機能を直接利用しなければならない場合や、そうでなくても高速性が重要な場合には、そのオブジェクトのクラスは MIO Core Object クラスのひとつとして実装される。

MIO Core Object クラスのオブジェクトには、ウィンドウシステムのためのものや、グラフィックス、イメージ、文字など描画に関するオブジェクトなどがある。

ネットワークを利用して通信するための機能も MIO Core Object クラスのオブジェクトとして実

装する。複数のマシン上で MIO システムが動作し、その MIO システム同士が通信する事により、MIO のアプリケーションを利用するユーザがコミュニケーションを行なう事ができるようにする。

4.4 MIO User Object クラス

MIO User Object クラスというのは MIO Core Object クラスを利用し MIO LISP を使って記述するオブジェクトのクラスである。このクラスは、文字通りユーザーによって定義したり、変更することができる。この機構により MIO システムは高い拡張性をもつのである。

簡単な MIO User Object クラスの例として string-button クラスについて説明する。string-button クラスのインスタンスを生成するには、引数として、ボタンに描かれる文字列を渡す。そうすると、文字列とその文字列を囲むのにちょうどいい大きさの長方形、ボタンのへこみ等を表現するための修飾用の長方形が作られ、これをグループ化したものがインスタンスとして返される。

string-button クラスは文字列を描画するための string クラスのインスタンスと長方形を描画するための fill-rectangle クラスのインスタンスという MIO Core Object クラスのインスタンスを組み合わせで定義されている。このようにして MIO User Object クラスを定義するのには MIO Core Object クラスのインスタンスを利用することができる。

イメージや文字など描画に関するオブジェクトを自由に配置するための canvas クラスも MIO User Object クラスのひとつである。さらに string-button クラスや canvas クラスのインスタンスを組み合わせて、より複雑な MIO User Object のクラスを定義することも可能である。

4.5 MIO システムの構成

情報処理教育等で利用される MIO システムのコースウェアは、今説明した MIO LISP を利用して記述する。このコースウェアを含めたシステムの構成は図 2 のようになる。

MIO システムの最下層のレイヤーには、システムの核となる LISP 処理系 MIO LISP とオペレーティングシステム、ウィンドウシステムの機能を利用する組み込みオブジェクトのクラス MIO Core Object クラスが存在する。そして MIO Core Object クラスの機能を利用して記述される MIO User Object クラスがその上のレイヤーに存在する。MIO システムのコースウェアはシステム階層の最上位のレイヤーに存在することになる。

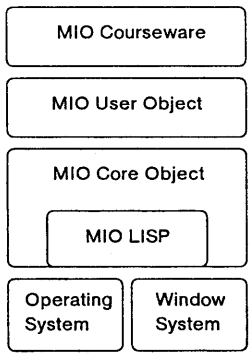

図 2: MIO システムの構成図

5 プロトタイプ

現在、我々は MIO システムの実装を進めているが、すでにシステムのプロトタイプが完成した。ここではそのプロトタイプについて説明する。

5.1 人体ウォークスルー

人体ウォークスルーとは中学、高校程度の学生が人体の仕組みを学ぶためのコースウェアである。このコースウェアには、人体の各部所、例えば、目や肺などの説明があり、説明図をクリックすると別プログラムが起動され、3 次元グラフィックスで描かれた人体の各部分をウォークスルーすることができる。

この人体ウォークスルーは慶應義塾大学の千代倉研究室と合同で開発したものであり、人体の説明をする 3 次元グラフィックスは千代倉研究室により作成された。我々が、開発したのはインターフェース部分である。

このインターフェースのために MIO User Object クラスのひとつとして mio-book クラスを実装した。このクラスのインスタンスは本型のインターフェースを提供している。ユーザは本をめくっているような感覚でコースウェアを操作することができる。mio-book クラスは、前述した canvas クラスのインスタ

ンスを利用して定義している。

図3は人体ウォークスルーの画面例である。

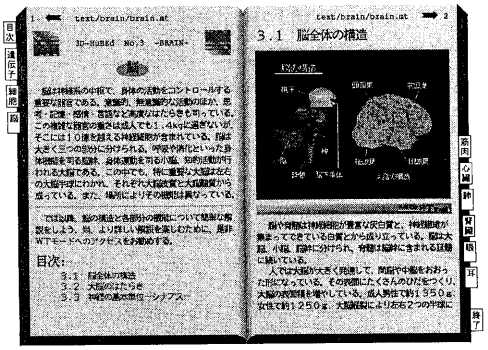

図 3: 人体ウォークスルーの画面例

5.2 iplmio

iplmio とは iplohp の後継として SFC の情報処理教育を行なうためのコースウェアである。このコースウェアを使えば、プログラミング教育を行なうことができる。

iplmio のために MIO User Object クラスのひとつとして mio-card クラスを実装した。このクラスのインスタンスは、プレゼンテーションを行ない易いように、カード型のインターフェースを持っている。また、プログラミング教育を行ない易いように、プログラムをクリックすると次のステップを実行するような、疑似ステップ実行の機能も提供している。mio-card クラスも mio-book クラスと同様に、canvas クラスのインスタンスを利用して定義している。

図 4は iplmio の画面例である。

6 今後の課題

以上が MIO システムの設計概要と現状である。最後に、我々の今後の課題について述べる。

- オブジェクトのクラスの充実

  MIO システムを充実させるためにより多くの MIO Core Object クラスと MIO User Object クラスを定義していかなければならない。特に、現在ネットワークを利用した通信や動画・音声のサポートができていないのでこれらを扱うことができる MIO Core Object の実装は重要な課題である。

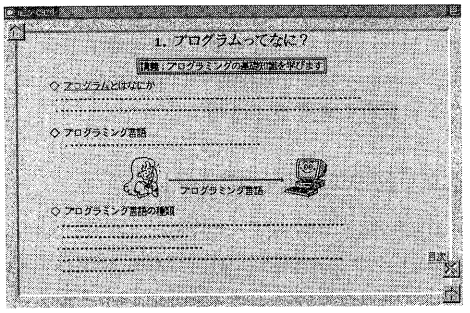

図 4: iplmio の画面例

- システムの利用を容易にするアプリケーションの開発

  現段階では MIO システムのコースウェアを準備する場合には MIO LISP で直接記述しなければならない。この方法では、特に、コンピュータを専門にしない先生がコースウェアを作る場合に非常に大変である。そこで、この手間を軽減するために、我々は WYSIWYG のハイパーテキストエディターやインターフェースビルダーも MIO LISP で記述したアプリケーションとして提供しようと考えている。これらのアプリケーションを開発することも今後の課題である。

参考文献

[1] 安村通晃, 有沢誠, 斎藤信男: コンピュータリテラシー教育の一事例, 情報処理, Vol. 32, No. 12, pp.1310-1317, (1991).

[2] Sun Microsystems: The Java Language Specification, (1995).

[3] 長崎祥, 田中譲: シンセティック・メディア・システム: IntelligentPad, コンピュータソフトウェア, Vol.11, No.1, pp.36-48, (1994).

[4] 市村哲, 前田典彦, 工藤正人, 松下温: 本とハイパーテキストを融合したグループ指向作業環境の実現, 情報処理学会マルチメディア通信と分散処理研究会, Vol.55, NO.10, pp.71-78, (1992).