

MMS : メディア管理システム

谷正之 山足公也 谷越浩一郎 谷藤真也

(株) 日立製作所 日立研究所

教育やプレゼンテーションの分野ではグラフィックス、映像、音声を駆使した対話型マルチメディアシステムの開発が盛んである。これらのシステムでは、従来のウインドウ、アイコン、メニューといったグラフィックユーザインタフェース(GUI)だけでなく、マルチメディアを利用したユーザインタフェース(MUI)の開発が必要である。そこで、従来のGUIに加え、MUIの開発支援を目的とするメディア管理システムMMSを提案する。MMSでは、応用プログラムからMUIを分離し、MUIの仕様を定義するための記述言語及び直接操作による定義環境を提供する。

Media Management System

Masayuki Tani Kimiya Yamaashi Kouichirou Tanikoshi Shinya Tanifuji

Hitachi Research Laboratory, Hitachi, Ltd.

4026 Kuji-cho, Hitachi-shi Ibaraki-ken, 319-12 Japan

This paper proposes Media Management System (MMS), a user interface management system extended to support multimedia user interface (MUI) which utilizes sound and video, as well as conventional graphical user interface (GUI) techniques. MMS supports: (1) a separation between a MUI and an application program, (2) a user interface description language extended to define synchronizations of media, and (3) an interactive development environment where the designer can build a MUI by direct manipulation.

はじめに

本研究の目的は、マルチメディアユーザインタフェース(MUI: Multimedia User Interface)の開発を支援することにある。ここで、MUIは、ビデオ映像、アニメーション、音声などのマルチメディアを使ったユーザインタフェース、及びそれらのメディアを操作するためのユーザインタフェースのことである。現在、マルチメディア応用システムの開発が盛んになっているが、そのMUIの実現に必要なメディア処理プログラムは個々の応用システムごとに開発されている。メディア処理プログラムでは、各種メディア機器の制御、メディア間の同期、メディアと計算機処理との同期など煩雑な処理を行なう必要があり、その開発に多大な時間を要している。

ウインドウ、アイコン、メニュー等のグラフィックユーザインタフェース(GUI)の開発支援を目的としてユーザインタフェース管理システム(UIMS)が提案されている[7]。UIMSでは、(1) GUIを実現するプログラムを応用プログラム本体から分離し、GUIを応用プログラム本体と独立に開発、変更できるようにすること、(2) メニューやアイコンなどの良く使われるプログラムをツールとして用意し、それらを組み合わせてユーザインタフェースを構築すること、(3) ユーザインタフェース記述言語(UIDL: User Interface Description Language)や、直接操作によるユーザインタフェース定義環境を提供すること、によりユーザインタフェースの開発効率を向上する。すでに多くのUIMSが開発され実用化されている[8]。しかし、既存のUIMSには、MUIの開発支援機能がなく、MUIはGUIとは別個に応用システムごとに開発する必要がある。

そこで、UIMSを拡張したメディア管理シス

テム(MMS: Media Management System)を提案する。MMSでは下記アプローチによりMUIの開発を支援する。

(1) MUIを応用プログラムから分離

MUIを応用プログラムから分離することにより、MUIを応用プログラムとは独立に開発、変更できる。例えば、画面上での映像の表示位置や、表示タイミングを応用プログラムを変更することなく自由に設定できる。

(2) MUIの開発支援環境の提供

MUIの開発支援環境として、記述言語(M-UIDL)と直接操作による定義環境を提供する。M-UIDLは、メディアに同期して実行すべき処理をイベントと動作の対として記述する。直接操作による定義環境では、各メディアの制御、メディア間の同期などを画面上での対話操作によって定義できる。

MUIとは

小文ではマルチメディアユーザインタフェースMUIを、(1) メディアを操作するためのユーザインタフェース、及び(2) メディアを利用したユーザインタフェースと定義する(ここでメディアは再生型の映像・音声に限定して考える)。

(1) メディアを操作するためのユーザインタフェース

編集システムでは編集対象のメディアを操作するための何らかのユーザインタフェースを備えている。例えば、ビデオ編集システムではビデオ機器に通常備わっている操作ボタンや計器をGUIで模擬したユーザインタフェースがよく使われる。また「代理旅行(surrogate travel)」と呼ばれるシステムでは、ビデオ映像の上に合成表示されたGUIを

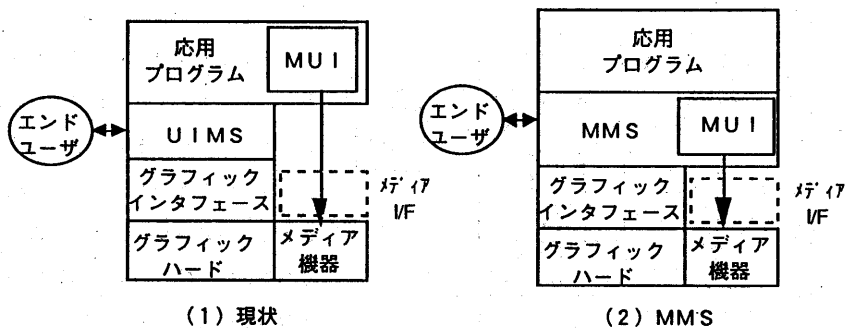


図1 MUIと応用プログラムとの分離

使って、ユーザが行きたい方向や、速度を指示するとそれによって再生するビデオフレームや再生速度が設定される[4][9]。こうしたシステムではメディアの操作にGUIを使う場合が多い。

(2) メディアを利用したユーザインタフェース

直感的に理解し易いユーザインタフェースを作成するために映像や音声を利用する。例えば、ファイルアイコンをドラッグするときにファイルの大きさと種類に合わせた音をフィードバックする[6]、画面上に表示されたビデオ映像内の事物にユーザがタッチするとそれに関連する情報を表示する[4]、アニメーションでヘルプを行なう[10]、といったユーザインタフェースが考案されている。また複数のメディアを同時に使うことも多い。語学学習システムでは会話の映像と台詞のテキストとを同時に表示したり、台詞を指定して映像を検索したりといった異種メディアを組み合わせ、相互に検索できるユーザインタフェースが有効である[5]。

MUI開発の問題点

MUIの実現には、メディアの制御、GUIとの組み合わせ、メディア間の同期などの処理が必要である。従来のマルチメディア応用プログラムでは、こうした処理を図1(1)に示すように各応用プログラムごとに開発していた。ビデオディスクなどのメディア機器に直接アクセスする処理が応用プログラム内に組み込まれる場合も多かった。最近ではウインドウシステムなどにメディア機器を制御するための標準のインタフェースを組み込むことが検討されている[1]。そうしたインタフェースを利用する場合でも映像の画面レイアウトや、グラフィックスなどの他のメディアとの合成、同期の処理などは応用プログラム内に組み込まれる。このため、MUIの作成が応用プログラムとは独立にできず、MUIを変更するには応用プログラムを変更する必要がある。

そこで、MMSでは図1(2)に示すようにGUIだけでなくMUIも応用プログラムから分離し、応用プログラムとは独立にMUIを作成、変更できるようにする。

MMSの構成

図2にMMSの構成を示す。MMSではアイコン、メニューなどのGUIツール、矩形、円などの図形

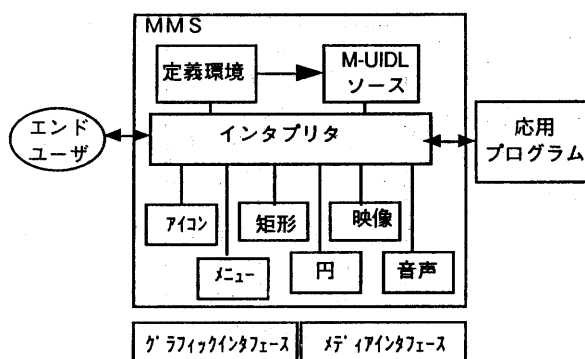


図2 MMSの構成

要素、映像、音声などのメディアをオブジェクトとして提供する。それらのオブジェクトを生成、操作するための基本的なメッセージが各オブジェクトごとに提供される。グラフィックシステムや、メディア機器へのインタフェースはオブジェクトの中に隠蔽される。ある映像を再生する機器とか、記憶媒体内の位置などの情報はオブジェクトの内部に設定する。オブジェクトの利用者は再生したい音楽の題名や映像の題名を知っていればよい。オブジェクトを利用する人はそのオブジェクトがどのデバイスによって再生されるとか、CDの何トラック目に入っているとかを知る必要がない。

ユーザインタフェースはこれらのオブジェクトを組み合わせて構築する。オブジェクトの組み合わせや属性の設定はユーザインタフェース記述言語M-UIDLと定義環境を併用して設定する。定義環境では画面上での直接操作によってオブジェクトを配置したり、属性を設定できる。定義環境によって設定した内容はM-UIDLソースに反映され、実行時にインタプリタにより解釈実行される。

イベントルール

ユーザインタフェースにおける対話処理はイ

ベントと動作の対として記述できる[2][3]。イベントと動作の対をイベントルールと呼ぶことにする。イベントルールはあるイベントが発生したときに、どんな動作を実行すべきかを定義する。イベントとは何かが起きたときに何が起きたかを知らせる合図である。GUIではイベントはユーザの入力機器の操作（マウスの移動、ボタンやキーの押し下げ、解放）に対応する。アイコンをクリックしてプログラムを起動するという対話処理は、「アイコンの上でマウスの

ボタンが押し下げられた」というイベントと、「所定のプログラムを起動する」という動作の対からなるイベントルールとして定義できる。

MUIでは前述のように複数のメディアを同時に使用する場合が多くあり、ユーザ操作に対応する処理だけでなく、メディア間の同期も記述する必要がある。MMSではGUIだけでなくメディア間の同期もイベントルールを用いて定義する。例えば、映像に同期してグラフィックスを表示する処理は、「所定のフレームが表示された」というイベントと、「所定のグラフィックスを表示する」という動作からなるイベントルールとして定義できる。また、ある音楽の再生時にイントロが終わったら、ビデオの再生を開始するという処理は、「イントロが終わった」というイベントと、「ビデオの再生を開始する」という動作の対からなるイベントルールとして定義できる。

「所定のフレームが表示された」、「イントロが終わった」などのイベントはMMSではマーカ機能によって実現する。マーカ機能は音声や映像の任意の部分に印を付ける機能で、マーカを付けた部分が再生されるときにマーカイベントが発生する。マーカ機能によって本質的に連続的なメディアに対する処理もイベントルールで記述可能にな

る。ビデオや音声は、計算機の処理とは独立に、予め決められた速度で再生される場合が多く、再生の進行状況をイベントとして扱うことによってメディア間の同期を記述し易くなる。

MMSではMUIとGUIを定義するための記述言語M-UIDLを提供する。M-UIDLはイベントルールを用いてメディア間の同期を定義できるように我々がすでに開発したGUI記述言語[11]を拡張したものである。M-UIDLでは下記形式によりイベントルールを記述する。

event イベント種別(引数並び) [動作]

「イベント種別」には、マウスやキーの操作によって発生する各種のユーザ操作イベントと、映像や音声の再生によって発生するマーカイベントがある。マーカイベントは、予めマークした映像フレーム、または予めマークした音声の位置(時間軸上)が再生されるときに発生するイベントである。「引数並び」はイベントの付加的な情報を指定する。ユーザ操作イベントの場合はそのイベントを発生するデバイス(どのボタンか、どのキーか)を指定する[11]。マーカイベントの場合は、そのイベントを発生するオブジェクトと、マーカの識別番号を指定する。「動作」はイベントが発生したときに実行したい処理を指定する。処理内容はオブジェクトに定義されているルーチンの実行、他のオブジェクトへのメッセージ送信、応用プログラム内のルーチンの呼び出しを使って記述する。

実行管理

イベントルールはオブジェクトごとに定義される。MMSのインタプリタはユーザ操作や使用中

のメディア機器を監視し、イベントが発生するとそのイベントを適当なオブジェクトに渡し、該当するイベントルールを実行する。メディアイベントとユーザ操作イベントではオブジェクトへの渡し方が異なる。ユーザ操作イベントはイベント発生時にカーソル位置にあったオブジェクトだけに渡される。また、そのイベントによって実行されるイベントルールも一つだけである。一方、メディアイベントはカーソル位置とは無関係に、そのイベントに対するイベントルールが定義されているオブジェクト全てに渡され、そのイベントにマッチするイベントルールはすべて実行される。

イベントルールの実行時には、計算機のオーバーロードを考慮する必要がある。例としてビデオ映像の再生時に、各フレームの再生に同期して、計算機で生成されたグラフィックスを合成表示する場合を考える。NTSCのビデオ映像の場合、1/30秒ごとにマーカイベントが生成されることになる。ビデオとグラフィックスが完全に同期させるには各マーカイベントに対応して定義されたイベントルールが1/30秒以内に処理される必要がある。これが達成されるか否かはグラフィック処理の内容、計算機的能力などに依存しており、イベントルールの定義時にオーバーロードを予見するのは難しい。

MMSではオーバーロード時の影響を最小限するため下記の実行管理方針を用意し、応用ごとに選択できるようにする。

(1) 遅れてもいいから全てのイベントルールを実行する。

(2) 処理が遅れたイベントルールは実行しない。

(3) 処理が遅れたイベントルールのうち優先度が一定以上のものだけは必ず実行する。

実行管理方針は基本的には同期を重視するか、

処理の質を重視するかの特ラードオフによって決まる。前記の例でいえば、同期を最重視する場合には遅れてしまったグラフィック処理は実行しないという方針になる。この場合はグラフィックスの表示は飛び飛びになり処理の質が落ちる。グラフィック処理の質を重視する場合は、どんなに遅れても全てのグラフィック処理を実行する。この場合、ビデオとグラフィック処理との同期は保証されない。応用ごとに最適な方針を選択する必要がある。

直接操作による定義環境

MMSでは、画面上での直接操作によって、個々のオブジェクトの定義（オブジェクトの配置や表示形態などの定義）、イベントルールの定義（オブジェクト間の同期の定義）を行なう定義環境を提供する。

オブジェクトの定義

オブジェクトの定義手順はオブジェクトの種類によらない。まず使用したいオブジェクトの種類をメニューから選択し、次に選択したオブジェクトの属性をマウスによる直接操作や、オブジェクトごとに用意されている設定シートを使って定義する。設定シートはオブジェクトを所定のマウスボタンを使ってクリックすると表示される。選択できるオブジェクトの種類は、メニュー、アイコンなどのGUIツール、文字や矩形などの図形要素、映像、音声などのメディアに分けられる。

オブジェクトとしてメニューを選択すると、画面上にデフォルトのメニューが表示される。メニューの位置はマウスを使って設定する。メニューの項目名や、項目が選択されたときに実行する処理の内容は設定シートによって指定する。同様に、

オブジェクトとして映像を選ぶと、映像を表示する領域が矩形として表示される。表示領域の位置や大きさはマウスを使って変更する。表示領域をマウスで選択すると、映像オブジェクトの属性を設定するための設定シートが表示される。設定シートにより、使用デバイス、再生範囲、クロマキーなどの再生方法、マーカイメントなどを指定する。

オブジェクトとして音声を指定すると、音声オブジェクトを表すアイコンが表示される。音声アイコンは定義時にもみ表示され実行時には表示されない。このアイコンを所定のマウスボタンでクリックすると設定シートが開く。設定シートにより使用デバイス、再生範囲、音量などの再生方法、マーカイメントなどを指定する。

イベントルールの定義

すでに述べたようにイベントルールは記述言語M-UIDLによって定義する。記述言語でイベントルールを定義する際には、マーカの識別番号を人間が直接指定する必要があり不便である。画面上に表示した映像を直接指定してマーカの識別番号を定義したり、音声の波形を見ながらマーカを指定できるような開発支援環境が必要である。

MMSでは下記手順によりイベントルールを定義する。

(1) オブジェクトの選択（ルールを定義する対象）

イベントルールを定義するオブジェクトを選択する。オブジェクトはマウスで画面上の表示物をピックして選択する。定義環境では定義対象となるオブジェクトは何らかの形で画面上に表示されている。映像は矩形もしくは静止イメージとして表示される。映像の特定のフレームにイベントルールを定義する場合には、実際に映像を再生しな

がら所望のフレームを選択する。音声はアイコンとして表示される。

(2) イベント種別の指定

メニューを選択して、イベント種別を選択する。イベントの引数の指定方法はイベントごとに異なる。マウスやキーのイベントではデバイスをキー入力によって指定する。映像イベントの場合は、そのイベントを発生するオブジェクトをメニューより選択し、次にマーカー番号を指定する。マーカー番号は実際に映像を再生しながら所望のフレームを選択して指定する。選択したフレームにすでにマーカーが定義されていればその番号が自動的にイベントの引数として定義される。マーカーが定義されていなければ、新たにマーカーが選択したフレームに定義され、その番号がイベントの引数に設定される。オブジェクトとして音声を選択した場合は、波形やスコア上でマーカーを選択または定義する。

(3) 動作の定義

動作は個々のオブジェクトに定義された基本動作を組み合わせて定義する。基本動作は動作エディタを用いて組み合わせる。動作エディタは動作の逐次実行、並列実行を表形式で階層的に組み合わせることができる。動作エディタの一つのセルには一つの動作が定義される。基本動作は動作主となるオブジェクトを選択し、次に動作主に定義されている基本動作をメニューから選択するという手順で定義する(図3)。

おわりに

マルチメディアを駆使したユーザインタフェースの開発を支援するメディア管理システムMMSを提案した。現在、MMSのプロトタイプを我々の開発したUIMS、MU[11]上に実現中である。今

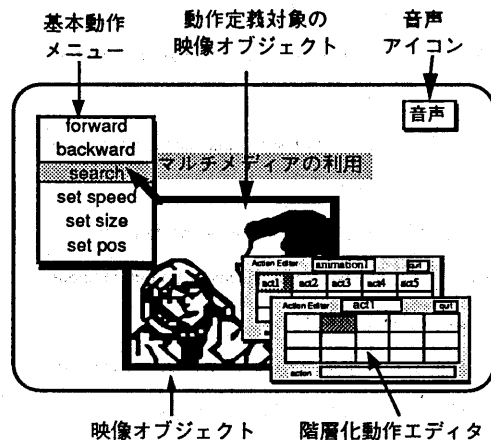


図3 直接操作による動作定義

後、MMSを使ってプレゼンテーションシステムなどの応用システムを開発しMMSの実用性を評価していく予定である。また、小文では再生型のメディアだけを考察の対象としたが、カメラやマイクなどからのリアルタイムに入力されるメディアを使ったユーザインタフェースについても今後検討する。

参考文献

1. Brunhoff, T. Pleasing the eye. Unix Review, Oct. (1989)
2. Green, M. A survey of three dialogue models. ACM Trans. on Graphics, 5, 3, July (1986), 244-275.
3. Hill, R.D. Supporting concurrency, communication, and synchronization in human-computer interaction - the Sassafras UIMS. ACM Trans. on Graphics, 5, 3, July (1986), 179-210.
4. Lippman, A. Movie-Maps: An application of the optical videodisc to computer graphics. SIGGRAPH'80 Conference Proceedings. July (1980), 32-42.
5. Mackay, W. E. and Davenport, G. Virtual Video Editing in Interactive Multimedia Application. CACM, 3

2, 7, July (1989), 802-810.

6. Nielsen, J. CHI'88 Trip report. SIGCHI Bulletin, Oct. (1988), 58-66.

7. Pfaff, G.E. User interface management systems. Springer-Verlag (1985)

8. Prime, M. User interface management systems - a current product review. North-Holland Computer Graphics Forum 9 (1990), 53-76.

9. Ripley, G.D. DVI - A digital multimedia Technology. CACM, 32, 7, July (1989), 811-822.

10. Sukaviriya, P. Dynamic construction of animated help from application context. Proc. of the ACM SIGGRAPH symposium on user interface software. Oct. (1988), 190-202.

11. 谷, 他. メタユーザインタフェースを有するユーザインタフェース構築支援システム. 情報処理学会論文誌, 30, 9 (1989), 1200-1210.