ソフトウエア工学 93−14
（１９９３．７．９）

３次元ＣＧをベースとしたＵＩＭＳの実現

朝日　宣雄　　田中　昭二　　李　殷碩　　前中　聡

三菱電機（株）情報電子研究所
E-mail: asahi@isl.melco.co.jp, gon@isl.melco.co.jp, lee@isl.melco.co.jp, maenaka@isl.melco.co.jp

３次元アニメーションを用いたユーザインタフェースを容易に構築し、さらに、任意の３次元オブジェクトを新たな部品として生成、追加を可能とするUIMSとしてMECOT（Metaphor Environment Construction Tools）を作成した。MECOTは、現状のGUIが持つ表現力の限界を解消する新しい枠組みを探究するための実験システムである。本稿では、MECOTのシステム概要、および、MECOTで採用した３次元インタフェースの部品化方式について説明し、３次元ユーザインタフェースの事例としてMECOT上に構築した３次元オフィス環境とその試作から得られた考察を述べる。

An Implementation of UIMS based on Three Dimensional Computer Graphics

Nobuo Asahi,　Shouji Tanaka,　Eun-Seok Lee　and　Akira Maenaka

Mitsubishi Electric Corporation, Computer & Information Systems Lab.
5-1-1 Ofuna, Kamakura, Kanagawa 247, Japan

We have implemented a UIMS, named MECOT (Metaphor Environment Construction Tools), on which 3D animated user interfaces can be easily constructed and any 3D objects can be generated as new user interface objects. MECOT is an experimental system designed to explore the new framework of 3D user interface which is much more expressive than the current GUI framework. In this paper, first, we describe the outline of MECOT system and the hierarchical construction method adopted to the system. Then, we show a 3D office environment which implemented on MECOT as an example, and discuss some advantages of MECOT and some issues on 3D user interfaces.

1．はじめに

情報化社会の進展に伴って、我々を取巻く情報環境は益々高度化・多様化し、情報機器の操作方法も複雑多岐に渡っている。また、情報機器のユーザ層の拡大に伴って、ユーザのニーズが多様化・個性化している。一般に、複雑な情報機器の操作方法や機能をユーザに直感的に提示するには、Xerox Starシステムのデスクトップメタファ[SMI,82]で代表されるようなインタフェースメタファの利用が有効である。近年、ユーザニーズの多様化・個性化に伴い、このインタフェースメタファも多様化・個性化することが望まれている。[ASA,90]

近年、グラフィカルユーザインタフェース（以下、GUI）が多くのシステムで使われるようになり、ユーザが直接操作によって直感的に操作しやすいインタフェースが一般的になってきている。これが可能となったのは、メニュー、アイコン、ボタンなどのGUI部品を持つプラットフォームが整備され、多くのアプリケーションプログラマが、これらGUI部品を組み合わせることによって、ユーザインタフェースを構築できる環境が整ったからであるといえる。しかし、一方で、このようなGUI環境で提供されるGUI部品は、計算機能力の飛躍的な向上にもかかわらずXerox Starシステムのころから大きな変化は見られない。この理由は、現在のGUIの形が、(1) 2次元表示に基づくユーザインタフェースの規格化、(2) アプリケーションからの独立性、および、(3) 設計者のための有限の部品セットの提供という制約の下で最適化したことによる限界であるといえる。

本研究の目的は、上記の3つの制約を取り払うことにより、現状のGUIが持つ表現力の限界を解消する新しい枠組みを探究することにある。このために、3次元アニメーションに基づいたユーザインタフェースが構築でき、さらに、任意の3次元オブジェクトをGUI部品として生成、追加可能とするような構築ツールおよび実行環境を備えた実験システムMECOT（Metaphor Environment Construction Tools）を作成した。以下、第2章において、新しいユーザインタフェースの枠組みとして我々が提唱する仮想操作環境を説明し、第3章において、MECOTで実現されたいくつかの技術的な内容を述べる。第4章では、MECOTを用いて構築した仮想操作環境の1事例である3次元オフィス環境を示し、第5章においてMECOTおよび3次元オフィス環境の試作から得られた考察をまとめる。

2．仮想操作環境

多くの場合、人間は具体的、空間的に表現されている対象の方が、抽象的、論理的な表現よりも記憶や理解が行いやすい[CAR,80]。この特性を利用したユーザインタフェースの試みとして、HI-VISUAL'89のオフィス情報処理[TAH,89]、Monsterの仮想オフィス[KAT,92]、TVメタファ[FRI,89]などがある。これらの共通の目的は、オフィスやテレビというメタファをできるだけ具体的なイメージでユーザに提示することによって、より分かりやすいユーザインタフェースを提供することにある。しかし、第1章で述べたように、このような直感的なメタファを表示するためには現状のGUIでは表現力に限界がある。

これを可能とするために我々は仮想操作環境として次の性質を備えた新しいGUIの枠組みを考えた。

(1) 3次元モデルで構成された任意のオブジェクトが3次元空間内に配置される。

(2) 各オブジェクトはユーザの操作またはアプリケーションからのイベントに応じて形状および位置等が変化し、それがアニメーションで表示される。

(3) 複数のアプリケーションがこれらの3次元オブジェクトの動作と連携して動作する。

(4) ユーザは3次元空間内で視点を自由に変更することが可能であり、3次元空間内に配置された3次元オブジェクトを利用することでタスクを行える。

以上のような性質を持つ仮想操作環境上で、様々なメタファをデザインし、それに対して様々なアプリケーションを構築することにより、直感的に分かりやすいインタフェースメタファの研究、3次元アニメーションの視覚効果を利用した新しい意味表現の研究、楽しさなどの感性を刺激するインタフェースの研究など、次世代のユーザインタフェースのための研究が行えると考えている。

MECOTは、これを可能とするために構築された仮想操作環境の実験システムであり、仮想操作環境上でのメタファの試作、評価のサイクルを容易にするものである。

3．MECOTシステム

3.2 システム構成

MECOTは、図1に示すようにクラスオブジェクトエディタ、操作環境ビルダ、操作環境マネジャの3つのツール群により構成される。

クラスオブジェクトエディタ及び操作環境ビルダ

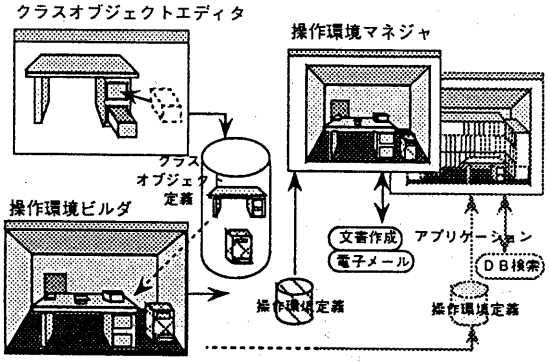

図1　MECOTのシステム構成

は構築ツールであり、これらの2つのツールを使ってメタファの部品化を図る。クラスオブジェクトエディタでは、3次元CADによって作られた3次元モデルとそれらの組み合わせや動きを記述したクラスオブジェクト定義によってインタフェースメタファのクラスを作る。操作環境ビルダでは、インタフェースメタファのクラスからそれらのインスタンスを生成し、3次元空間内に配置することによって仮想操作環境を構成する。

操作環境マネジャは実行ツールであり、構築ツールによって構成された操作環境定義を実際のアプリケーションと連動した形で利用できるようにする。

3.2　階層的部品化

MECOTでは、あらかじめ定義された少数の汎用的なクラスオブジェクトを組み合わせるという従来のGUIで採られている方法ではなく、新たなクラスオブジェクトもユーザインタフェース設計者が定義することを可能としている。このとき新たに定義するクラスオブジェクトは、形状だけではなく、その動きも同時に記述される必要があるが、その際に重要となるのは、再利用性の高いクラスオブジェクトの部品化の枠組みである。

後述する3次元オフィス環境を例にとると、我々が目指す仮想操作環境では、机、書類、ゴミ箱、部屋などの対象が3次元的に表示され、これらの対象は、それぞれがユーザの操作に応じて動作すると同時に、互いに関連する動作も行う。例えば、机の引き出しが開閉する動作は机の動作であるが、ドキュメントを机の引き出しに収納する動作は机とドキュメントの2つの対象が関与する動作である。そのため、クラスオブジェクトを部品化するためには、1つのオブジェクトに閉じた動作定義と複数のオブジェクトが関与する動作定義を切り分けた上で、前者を部品化することを考えなくてはならない。

MECOTでは、これをコンポーネントレベル、オブジェクトレベル、環境レベルという3つのレベルからなる階層化を採用することによって解決している（図2）。

コンポーネントレベルでは、オブジェクトの構成要素を定義する。この構成要素は、オブジェクトの部品となる3次元形状（ポリゴンデータ）とメニュー、ボタンなどの従来型GUIのユーザインタフェースオブジェクト（ウィジェット）である。

オブジェクトレベルでは、コンポーネントを所定の位置に配置することにより1つのクラスオブジェクトを定義する。クラスオブジェクトは、いくつかの状態をもち、状態遷移によってその動作を定義する。クラスオブジェクトの各状態には、それぞれ異なったコンポーネントの配置をとるグラフィカルな状態が対応しており、状態の遷移はグラフィカルな変化に対応する。

操作環境レベルでは、オブジェクトレベルで定義されたクラスオブジェクトからインスタンスを生成し、それを所定の位置に配置することによって定義

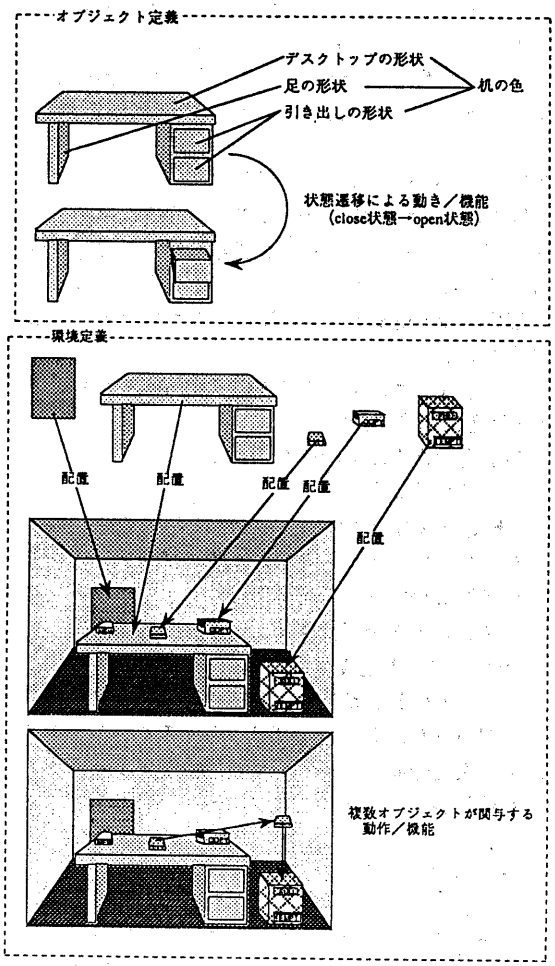

図2　オブジェクトと環境

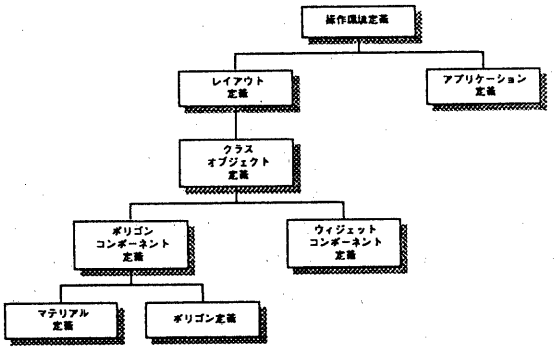

図3　記述言語の階層

する。操作環境レベルでは、複数のオブジェクトが関与するような動作、および、環境を考慮することによって可能となる動作が定義される。

このように、コンポーネントレベル、オブジェクトレベル、環境レベルという3つのレベルに定義を階層化することによって再利用性の高い部品化が可能となる。例えば、コンポーネントレベルで作られたコンポーネントは、様々なクラスオブジェクトの構成要素として利用可能であり、また、オブジェクトレベルで作られたクラスオブジェクトを利用して、様々な仮想操作環境を構成することができる。

3.3　インタフェース記述言語

以上の部品化方式を反映して、仮想操作環境全体を定義するための記述言語を開発した。本記述言語は後述する操作環境マネジャによって解釈され、仮想操作環境の構築および制御するためのデータとして扱われる。

本記述言語は、図3に示すような階層を持ち、それぞれ独立に記述する。

各定義の概要は次の通りである。

(1)操作環境定義

操作環境定義では、ユーザおよびアプリケーションからのイベントによる各オブジェクト間の相互作用動作など、構築するインタフェース全体のメカニズムを記述する。

(2)レイアウト定義

レイアウト定義では、クラスオブジェクト定義で記述されたクラスオブジェクトのインスタンスを3次元空間内にどのように配置するかによって初期状態におけるインタフェースの外観を定義する。

(3)クラスオブジェクト定義

クラスオブジェクト定義では、操作対象となるクラスオブジェクトを構成するコンポーネントの配置、クラスオブジェクトの属性および固有の動作などを記述する。

(4)アプリケーション定義

アプリケーション定義では、インタフェースと独立して動作するアプリケーションとの通信プロトコルを記述する。

(5)ポリゴンコンポーネント定義

3次元CADで作成された3次元コンポーネントのポリゴン情報およびその色を定義する。

(6)ウィジェットコンポーネント定義

ウィジェットの配置、色などを定義する。

(7)ポリゴン定義

頂点座標、ポリゴンの法線ベクトルなどによってポリゴンを定義する。

(8)マテリアル定義

環境光の反射、拡散反射、鏡面反射、透過率によってポリゴンコンポーネントの色を定義する。

以上のようにインタフェース全体の記述を階層的に分割してそれぞれ独立に記述することから、インタフェースの仕様変更に伴う変更が局所的に行え、またメタファとアプリケーションの組み合わせに自由度を持たせることができる。

4．動作例

MECOTを用いて仮想操作環境の1例として3次元オフィス環境を作成した。現在、MECOTはシリコングラフィックス社IRIS4D 340VGX上で動作しており、仮想操作環境は通常のディスプレイに表示されたものを3ボタンマウスによって操作するようになっている。

4.1　3次元オフィス環境

仮想操作環境の1つの事例として、電子メール、文書作成、文書検索を行う3次元オフィスルーム環境を作成した。この仮想操作環境は以下のようなコンセプトに基づいている。

まず、この仮想操作環境は、パーソナルな作業空間としての部屋（パーソナルルーム）、会議を行うパブリックな空間としての部屋（パブリックルーム）、たくさんの文書を格納しその検索するための部屋（ライブラリルーム）という3つの部屋により異なった種類の仕事を空間的に配置するというコンセプトで作られている。（図4）

パーソナルルームでは、個人的なドキュメント作成およびドキュメントの整理を行う空間である。机の上のドキュメントを選択すると、そのドキュメントそのものがエディタとなり編集可能となる。このドキュメントは電子メールとして送信することも可

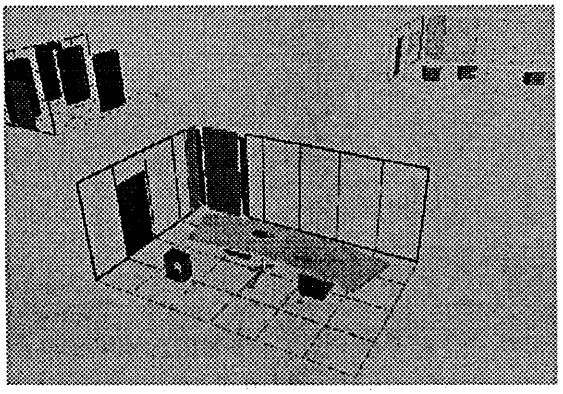

図4　3次元オフィス環境全景

能である。電子メールとして送信したドキュメントは部屋の外に飛んでいくというアニメーションによって直感的にメール送信が理解できるようになっている。メールが来ると逆に部屋の外からドキュメントが飛んでくる。飛んできたメールはドキュメントと同様に編集／保存が可能である。机の上は作業中のドキュメントが置かれており、また、一次格納のためには引き出しという格納場所が用意されている。ゴミ箱は不要なドキュメントを捨てるためのメタファある。机が一次的なドキュメントの格納を表わしている一方、部屋の後ろには個人のファイルキャビネットが置かれており、ここで個人のファイルを検索し、編集のために取り出すことができる。

パーソナルルームでは個人ファイルの編集、検索を行うのに対して、ライブラリルームは、多くの人と共有しているさらに大きなデータベースからのファイルの検索を行う場所である。ここにはパーソナルルームで使われているファイルキャビネットメタファが再利用されて置かれている。

また、パブリックルームは、将来、ネットワーク上の他のユーザとの電子会議を行うという利用を想定した会議室メタファとなっているが、現在はアプリケーションとのリンクは行っていない。

4.2　構築例

4.2.1　クラスオブジェクトエディタ

クラスオブジェクトエディタは、3次元CADなどによって作られた3次元モデルとそれらの組み合わせや動きを記述したクラスオブジェクト定義によってインタフェースメタファのクラスを作成するためのツールである。

クラスオブジェクトエディタは、次に示す機能を有する。

・3次元形状(ポリゴン)ファイルのロード
・コンポーネントの色定義機能
・コンポーネントの配置機能
・コンポーネントの階層化機能
・メタファの属性定義機能
・状態遷移に基づくメタファの動作定義機能

上記機能を図を用いて説明する。

(1)　ファイルロード、配置、階層化、色定義

操作ウィンドウに読み込まれたコンポーネントは、メニュー内のマテリアルデータベースから適切な色を選ぶことによって色付けされる。また、要求する色がデータベース内になければ、マテリアルエディタと呼ばれる色編集ツールによって新たに色を作成し、データベースに追加することもできる。

配置および階層化はマウス操作によって行われる。配置はまた数値パネルによって詳細に設定することもできる。(図5)

(2)　状態遷移に基づくメタファの動作定義

動作定義は、まず状態定義パネルによって論理状態を設定し、その論理状態に対応したグラフィカル状態を設定することによって定義することができる。（図6）

4.2.2　操作環境ビルダ

操作環境ビルダは、仮想操作環境を構築するためのツールである。

操作環境ビルダでは、クラスオブジェクトからそれらのインスタンスを生成し、3次元空間内に配置することによって仮想操作環境の外観を定義し、この配置情報と環境内のインタフェースメタファの動作記述を合わせて操作環境定義を生成する。

操作環境ビルダは、次に示す機能を有する。

・クラスオブジェクトファイルのロードおよびインスタンス生成機能
・インスタンスの配置機能

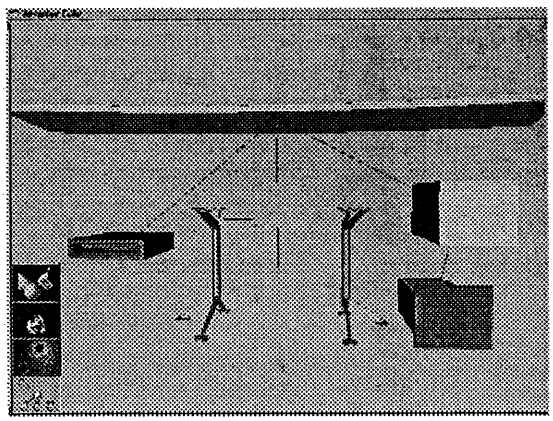

図5　コンポーネント配置

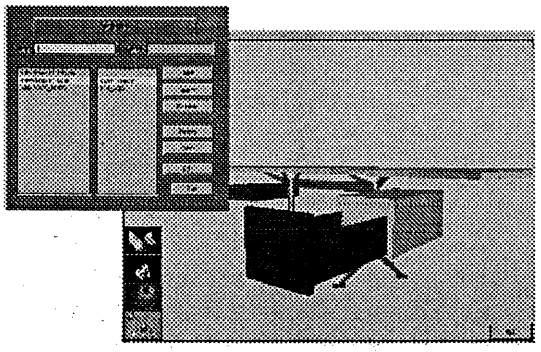

図6　オブジェクトの動作定義

・仮想エリア配置機能
・視点設定機能

ここでは仮想エリア配置機能および視点設定機能に関して説明する。

(1)仮想エリアの設定

仮想エリアは環境内のオブジェクトの空間的な移動に際してその移動先を名前で指定するための機能である。図7に示すようにエリアをワイヤーフレームで表示し、それをマウスあるいは設定パネルによって、位置、大きさ、セルの大きさを設定することによって定義できる。またエリア内の重力方向は、塗りつぶされた面によって示されているため、設定した重力の方向が一目で分かるようになっている。

(2)視点設定

視点は、図8に示すように視点の位置と向きを表す立体を表示し、それをマウスや設定パネルを用いて、視点の位置と向き、視野、クリッピング平面を設定することによって定義する。

4.3　実行例

操作環境マネジャが仮想操作環境定義を解釈し、実行する様子を説明する。

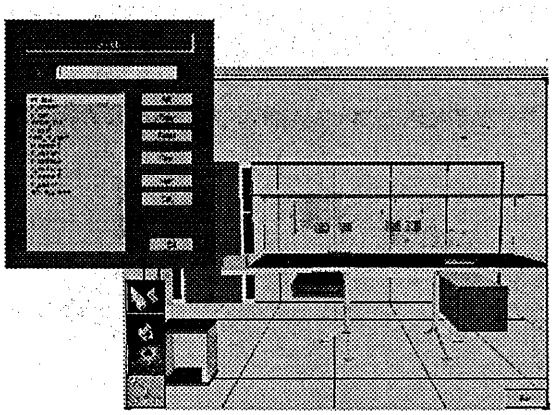

図7　仮想エリアの設定

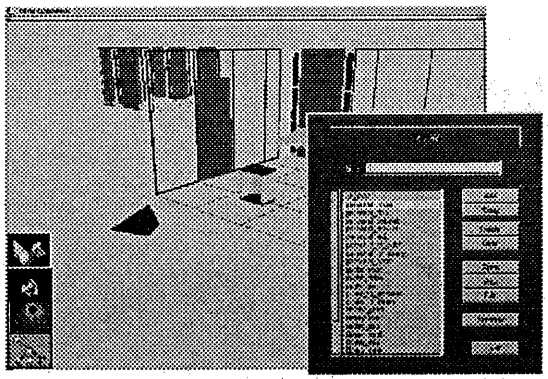

図8　視点の設定

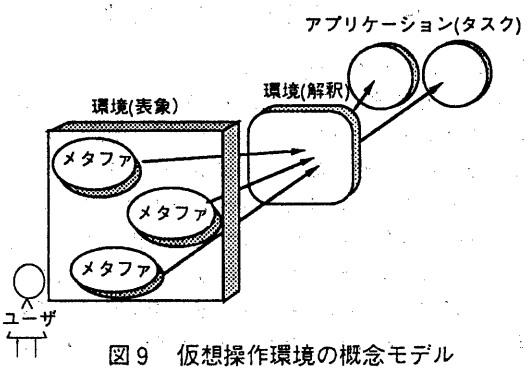

図9　仮想操作環境の概念モデル

図9は、仮想操作環境の概念的な実行モデルを表わしたものである。クラスオブジェクトは、操作環境マネジャ上では、そのインスタンスオブジェクトが表示され、アプリケーション機能の表象としてユーザに提示される。これらのインスタンスオブジェクトの動作は、オブジェクトレベルの動作についてはクラスオブジェクト定義、また、操作環境レベルの動作については仮想操作環境定義を解釈することにより実行される。これらの動作は、基本的にはイベントとそれに対する動作記述という形式で書かれている。アプリケーションからのイベントおよびアプリケーションへの処理要求は操作環境レベルの動作記述の中に書かれる。

図10はオープンしたドキュメントをメールとして送信する画面の例である。まず、ユーザのドキュメントインスタンスに対する操作は、ドキュメントクラスの以下の動作ルールを発火する。

```
mail_open -> mail_closed
trigger: { select(doc_base_component) }
effect: {
    send_mail();
    fly_mail(integer 10);
}
```

ここで発火された動作ルールには、電子メールア

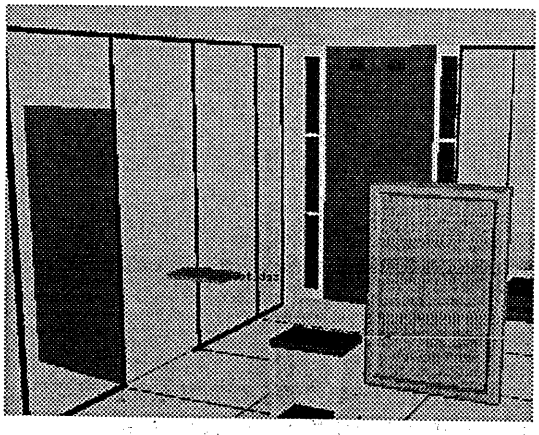

図１０　メール送信画面例

プリケーションに、ドキュメントインスタンスが表わしているファイルを送信する要求を出すアクション（send_mail）とドキュメントインスタンスが部屋の外に飛んでいくと同時に部屋のドアが開閉するアニメーションを表示するアクション（fly_mail）が記述されている。

send_mailは、アプリケーション定義に書かれているメッセージプロトコルにしたがってプロセス間通信によって処理要求を送信する。fly_mailは、操作環境定義においてアクション関数として次のように定義されている。

```
fly_mail(integer step) {
    event(room1, open_door);
    fly(CALLER, behind_door, *step);
    event(room1, close_door);
    delete(CALLER);
}
```

ここでevent(room1, open_door)はroom1というインスタンスにopen_doorというイベントを送ることであり、room1はこのイベントを受け、部屋クラス定義に従って、もしドアが閉じていたらドアを開くアクションを実行する。fly(CALLER, behind_door, *step)において、flyは操作環境マネジャが持つアニメーション表示のための組み込み関数で、飛ばす対象となるオブジェクトインスタンス、飛ばす行き先を表わす仮想エリア，アニメーションのステップ数を引数とする。CALLERはこの関数を呼び出したオブジェクトインスタンスを示す特別な変数である。最後にroom1にclose_doorイベントを送信し、ドキュメントインスタンス自身はdelete関数により消滅する。

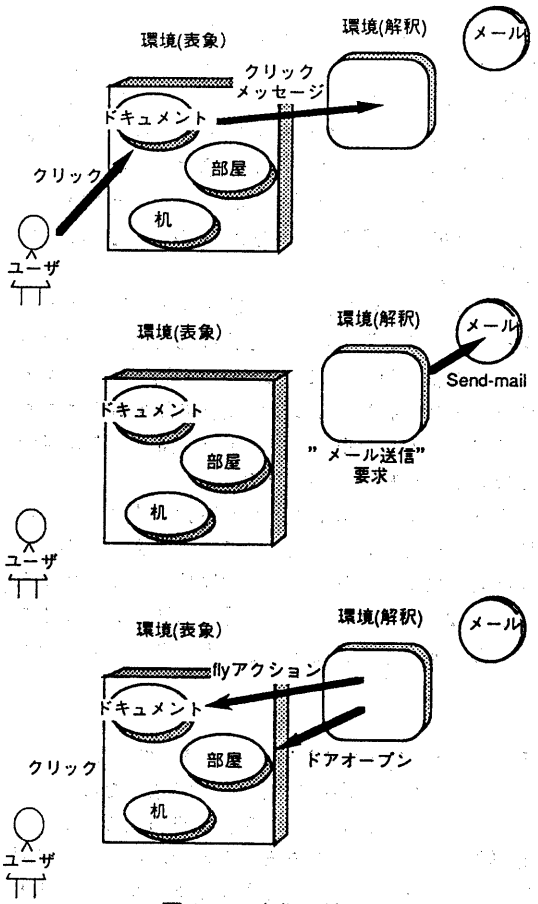

図１１　実行の流れ

５．考察

5.1　階層的部品化について

形状、質感、動き、アプリケーションの記述を分離し、階層的な依存関係を持たせることにより、様々なレベルでの交換や再利用が可能となっている。このため、形状や質感はデザイナにまかせる一方で簡単な形状を用いて動きの部分やアプリケーションを作成し、あとでそれらを組み合わせることなどが可能となる。このように細かいプログラミングの問題をユーザインタフェースのデザインと切り離すことが可能である性質は、ユーザインタフェース構築にとって重要な側面である。また、これを仮想操作環境の可能性を探る実験システムとして利用する際には、ユーザの不満の原因が形状や質感のデザインにあるのか、動きにあるのか、また、メタファとして用意するオブジェクトにあるのか、環境上でのメタファの不整合にあるのかなどの問題を切り出して評価することも可能であると考えている。

5.2 記述能力と効率について

MECOTでは、様々な仮想操作環境を容易に作成するために、独自の定義言語を作成し、簡単な記述で動きのある3次元ユーザインタフェースが作成できるようにした。この効果を調べるために、簡単な仮想操作環境のコーディング量をC言語＋グラフィックライブラリで行った場合、全て定義言語で記述した場合、構築ツールを用いて行ったものとで比較した。

この結果、定義言語では、C言語＋グラフィックライブラリで開発するものと比較して約1/20のコーディング量で行えることが分かった。また、構築ツールを用いることにより、配置に関する定義が直接操作で行えるため、コーディング量はさらに半分となるという結果が得られた。

このように簡単に記述可能な定義言語とするために、その記述能力を制限したことも事実である。記述能力と効率はトレードオフの関係にあると考えられるが、設計者の能力や要求に応じて詳細レベルまで変更できるような枠組みとしておくことが必要であろう。

5.3 3次元ユーザインタフェースについて

3次元オフィス環境の試作から痛感したことは、多くの人がマウスによって3次元空間上のドラッグを操作できないことである。この問題は、多くの原因を含む。まず、画面の描画サイクルが遅くなると操作のフィードバックが遅れるため、あるところからとたんに操作性が悪くなる。一般的に、1秒10画面以上の描画サイクルが快適な操作には必要とされている[CAD,91]が、オブジェクト同士の衝突をチェックしたり、複雑な形状のオブジェクトの表示を行う場合は困難である。また、2次元の画面では奥行きが認識しにくいという問題もある。現在は表示が遅くなることを防ぐために、影（シャドウ）を表示していないが、なんらかの奥行きを表わすキュー[LEO,92]を提示する必要がある。さらには、3次元操作用の入力デバイスを用いることも必要であろう。

また、3次元オフィス環境のような具体的なメタファによってアプリケーションを表示し、さらに複数のメタファによって環境を構築する場合、全体の整合性を保ちながらストーリーを構成する作業が予想以上に難しいことを痛感した。このため、この種のユーザインタフェースを設計する際には、映画やまたは最近のロールプレーイングゲームなどを作成する場合のように脚本や演出のセンスも必要となるだろう。

6．まとめ

3次元仮想操作環境を構築、実行するUIMSであるMECOTについて、そのシステム概要、および、仮想操作環境の1事例として作成した3次元オフィス環境を述べた。3次元仮想操作環境は従来のGUIと比較して潜在的な表現力を持つが、この表現力を有効に利用するためには試作、評価の繰り返しを地道に行うことが重要であると考えている。

7．謝辞

なお本研究の一部は通産省のFRIEND21プロジェクトの一環として実施されたものである。

参考文献

[ASA,90] 朝日: コンテクスチャルメタファのプロトタイプ, ヒューマンインタフェースN&R, Vol.5, pp279-288, 1990.

[CAD,91] Card, S.K., et.al.: The Information Visualizer, An Information Workspace, ACM SIGCHI'91 Proceedings, pp181-188, 1991.

[CAR,80] Caroll, J.M., Thomas, J.C., Malhotra, A.: Presentation and representation in design problem-solving, British Journal of Psychology, 71, pp143-153, 1980.

[FRI,89] FRIEND21研究センタ: 昭和63年度情報化推進基盤整備未来型分散情報処理環境基盤技術開発研究報告書, pp36-38, 1989.

[KAT,92] 勝山 恒夫 他, MONSTERのヒューマンインタフェース, 情報処理学会研究報告92-HI-40(3), 1992.

[SMI,82] Smith,D.C., Irby,c., Kimball,R., Verplank,B. & Harslem,E.: Designing the Star User Interface, Byte, Vol.7, No.4, pp242-282, 1982.

[TAH,89] 田原, 吉見, 平川, 田中, 市川: 視覚的プログラミング空間に関する一考察, 情報処理学会研究報告98-SE-64(16), pp121-128, 1989.

[WAN,92] Wanger, L.R., et.al.: Perceiving Spatial Relationships in Computer-Generated Images, IEEE Computer Graphics & Applications, May 1992, pp44-55.