

非 定 形 UIMS の 概 念†

谷 越 浩 一 郎^{††} 荒 井 俊 史^{††} 谷 正 之^{††}
横 山 孝 典^{†††} 谷 藤 真 也^{††}

計算機システムがユーザに提供するユーザインタフェースの技術として従来から、ユーザが画面上で用いるスクロールバーやメニューといったものを部品として用意して、それらを組み合わせてユーザインタフェース画面を作成するという UIMS (ユーザインタフェース管理システム) の考え方があった。しかしこの方法ではあらかじめライブラリに登録されている部品単独、もしくはそれらの組合せでしか画面を作ることができないという問題があった。ここで画面構成を部品によらず行える非定形 UIMS という概念を提案し、その実現方式を検討する。非定形 UIMS は従来の UIMS とは逆に、まずビジュアルエディタで画面を作成し、次にでき上がった画面を構成する図形要素やシンボルにさまざまな対話機能を持たせる。この対話機能には、ユーザ操作の内容をアプリケーションに伝え、アプリケーションはそれに伴い色の変更などの指示を与えるといった単純なものほかに、ユーザ操作に応じて図形の形状を動的に変えていく機能をもたせる。このようにすることで非定形 UIMS では、各応用分野に固有の世界を画面上に表示し、それを用いてアプリケーションと対話を行うことができる。そのため標準部品だけで使って作られる限られた対話処理画面に比べ、ユーザにとって分かりやすいユーザインタフェース画面を実現できる。

1. は じ め に

近年、マウスなどのポインティングデバイスとグラフィックディスプレイを用い、高機能の対話性を持ったユーザインタフェースを実現するアプリケーション・ソフトが増えてきている。このようなアプリケーション・ソフトの作成には大変なコストがかかる。

この問題を解決するため我々は、UIMS (User Interface Management System) の考え方によるユーザインタフェース構築支援システム MU を開発してきた。UIMS とは、ユーザインタフェース部分とアプリケーションプログラム本体とを切り離して開発できるようなシステムである^{1)~3)}。

この方式は CAD や OA のような定形的な処理を行う応用ソフトのユーザインタフェースを開発する場合は、非常に大きな効果を発揮する。そこでこのシステムを定形 UIMS と呼ぶ⁴⁾。

現在、Xウィンドウシステムの widget⁵⁾ に見られるようにユーザインタフェースの表示部品をパッケージ化してアプリケーションプログラムの生産性を上げ

るといった試みがなされているが、定形 UIMS も基本的にはこのような枠組みに入れられる。

一方、故障診断システム等ではそのプラントの全体図を表示し、故障箇所をハイライト表示するといったユーザインタフェースが望まれるが、それを標準部品から作り上げることは難しく、ユーザが自由に部品の形と機能とを作り上げられるような仕組みが必要である。

このような機能を持つものとして Desoi らの ADG がある⁶⁾。これは、まず作り上げる部品の動きを、ゲージであるとかグラフであるとかいうように部品の種類を指定することによって規定する。その後でグラフィックエディタのようなもので部品を作画し、その作画した部品の個々のグラフィック要素に対してその部品における役割を割り振る。しかしこの方法では、システムにあらかじめ決められておいた部品の動きのパターンしかできず、その部品の品揃えによって動きが制限されてしまう。

そこで本論文では、動きを部品の種類によって規定してしまうのではなく、抽象的な動きそのものを自由に描いた絵柄に対して設定するような非定形 UIMS の概念を提案する。

以下第2章では、現在の定形 UIMS の基本概念を我々が開発してきたシステム MU に基づき説明する。第3章では、非定形 UIMS の概念を提案し、第4章ではその問題点と解決策を図形オブジェクトの構成法と直接操作の定義法といった2点から論じる。最後に

† Design Concept of System for Developing Application-Dependent User Interface by KOUICHIROU TANIKOSHI, TOSHIFUMI ARAI, MASAYUKI TANI (The 3rd Department, Hitachi Research Laboratory, Hitachi, Ltd.), TAKANORI YOKOYAMA (ICOT Research Center, Institute for New Generation Computer Technology) and SINYA TANIFUJI (The 3rd Department, Hitachi Research Laboratory, Hitachi, Ltd.).

†† 日立製作所日立研究所第3部

††† (財)新世代コンピュータ技術開発機構

第5章では、まとめと非定形 UIMS の今後の課題について述べる。

2. 定形ユーザインタフェース 構築支援システム

本章では、我々が開発してきた MU を取り上げ⁷⁾、定形 UIMS の基本的な考え方とその限界を説明する。

2.1 MU の基本機能

(1) オブジェクト

定形ユーザインタフェース構築支援システム MU による実際のアプリケーションの例を図1に示す。二次元 CAD システムのメニュー構築に MU が用いられている。

MU では、この図に見られるようなメニューやコマンド入力域などユーザインタフェースの構成部品を標準品として用意している。それらをここでは「オブジェクト」と呼ぶことにする。オブジェクトには属性と呼ばれるものがあり、表示の位置、大きさ、色などがこの属性に値として設定されている。属性の値を変更すると実際の表示も変化する。これらのオブジェクトや属性の考え方は、通常のオブジェクト指向型システムと同様である^{8),9)}。

(2) イベント

マウスボタンの押下、マウスカーソルの移動、キーの押下などユーザの操作はイベントという形でオブジェクトに伝えられる。イベントはその種類を表すイベント名とイベントの位置などの付加情報からなっている。イベント名は実際の物理的なイベントに対して複数の名前をつけることができ、その切り換えを行うことにより同じ物理イベントでも異なった目的に用いることができる。これをイベント名切り換え機構と呼ぶ¹⁰⁾。

(3) アクション

オブジェクトは、イベントを受け取ると、エコー表示を指定したり、アプリケーションプログラム側のサブルーチンを呼び出すといった動作を行う。この動作のことをアクションという。

アクションはほかのオブジェクトからのメッセージ送信によっても起動される。このため MU ではアクションは、イベント名と動作記述の対の形、またはメッセージ種別を表すメッセージ名と動作記述の対で定義する。

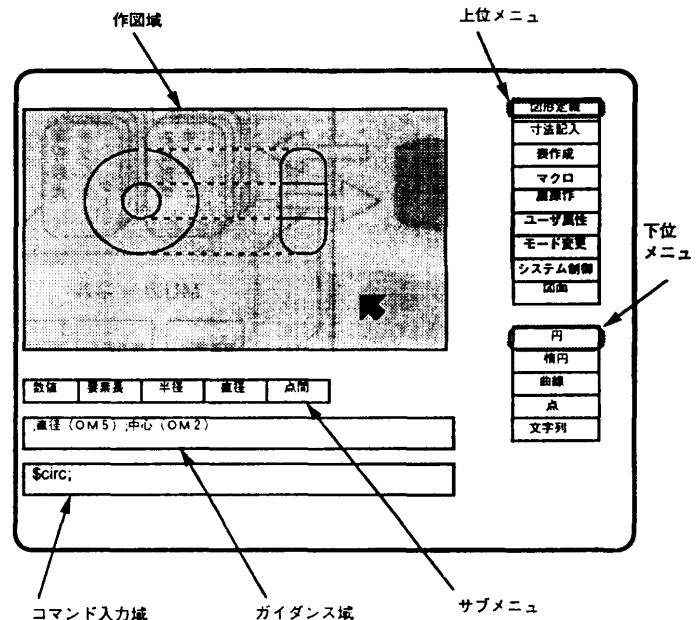


図1 定形ユーザインタフェース構築の例 (作図システム)
Fig. 1 Example of user interface construction for CAD system.

(4) 直接操作

MU ではメニューをマウスで直接所望の位置まで移動するとか、メータの針の表示位置をマウスで指定するといったように、描かれた絵柄を直接操作して入力を行うことが可能になっている。

2.2 システム構成

MU のシステム構成を図2に示す。

一番下にオブジェクトを管理する部分であるカーネル部があり、個々のオブジェクトはその管理下で動く。具体的には、データの入出力やメッセージの受け渡し、イベントの管理などがカーネル部の仕事である。

オブジェクトを編集するためのメニュー等システムのオブジェクトもユーザインタフェースのオブジェクトと同レベルに実現されているため、編集環境自身を簡単に変更することが可能である。

オブジェクトはユーザインタフェース記述言語 UIDL で記述される。そして応用システムの起動時に、実行形式に変換され画面上に表示される。

しかし例えばメニューの場合、各項目の名称や表示位置、色といった仕様データや、メニュー項目選択に対するアクションが UIDL を使って定義されているが、あらかじめ可変部分は固定されており、メニューの入力方法といった直接操作を受け持つアクションは決まったスタイルの入力しかサポートしない。

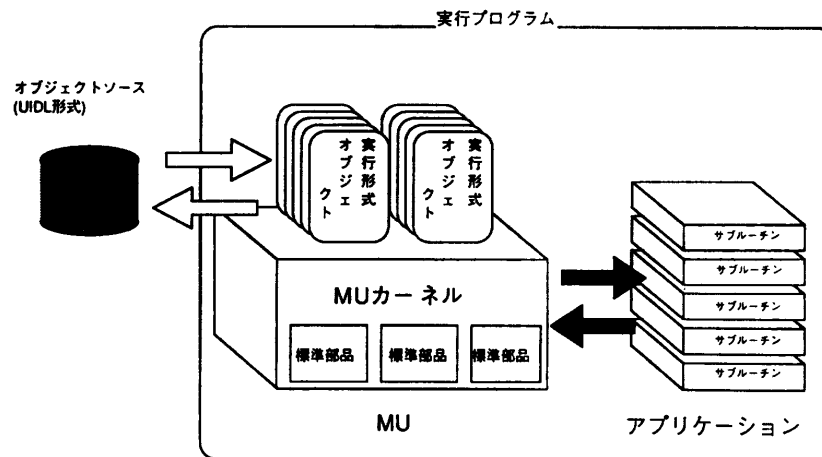


図 2 現在の MU のシステム構成
Fig. 2 A present architecture of MU.

2.3 現システムの限界

MU のような非定形 UIMS でのユーザインタフェース構成法は、CAD システムのようなスタイルのアプリケーションに対しては効果的である。これは、2.1 節での例のように CAD システムでは使われるオブジェクトがメニューや入力域などパターン化されており、標準部品を再利用する形でユーザインタフェースを作ることができるためである。

しかし広く一般のアプリケーションプログラムに対しては、あらかじめオブジェクトを用意することは難しい。これを図 3 の圧延機の故障診断のユーザインタフェースの例で示す。

このシステムでは例えば故障が起きた時に、スタンドと呼ばれる圧延機の構成単位 (図 3 (a) * の部分) の詳細図を図 3 (b) のように出し、どこがおかしくなったのか詳細に指定させるといったユーザインタフェースが必要である。

さらに、診断を行っている際に他の機器の情報が必要になった場合、図 3 (c) のように入力を求めるシートを出すことも必要である。その際にユーザは値をタイプするのではなく、シート上のスライド入力装置の絵柄のつまみ部分を動かすといった直接操作により値を入力できることが望ましい。

最終的に診断結果は、故障部分が詳細図にハイライト表示されて示される。

以上のようなユーザインタフェースを定形 UIMS

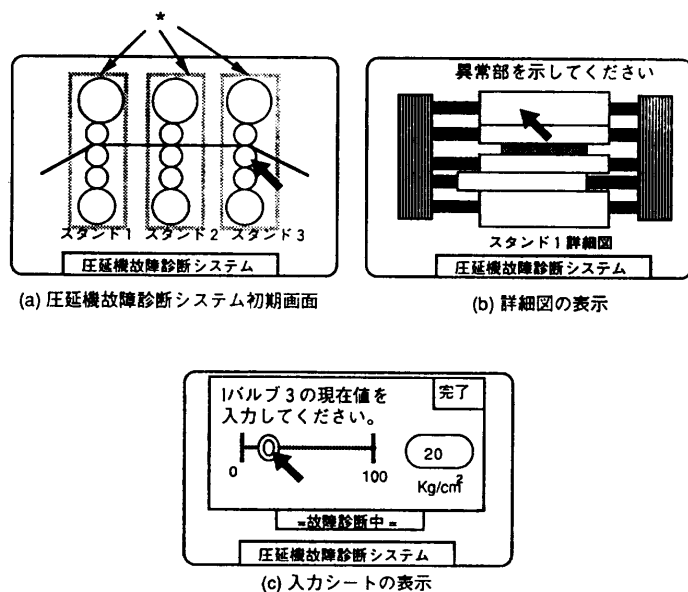


図 3 故障診断システムのユーザインタフェース
Fig. 3 Example of user interface for consulting system for mill.

の考え方で構成するためには、圧延機の絵や、シート上に現れたスライド入力装置などをオブジェクトとして用意する必要がある。しかし診断システムによって圧延機の種類も変わり、診断のスタイルも変わるため圧延機やつまみのオブジェクトを部品として用意しても再利用の可能性は少ない。

そのためせっかく部品を用意しても、絵柄を変更したり、新しい対話機能を割り付けるといった部品プログラムの修正・作り直しが必要であり、結果としてユーザインタフェースの開発効率はあまり上がらない。

3. 非定形ユーザインタフェース構築支援の提案

定形 UIMS の持つ問題点を解決するため、次のような特徴を持った非定形ユーザインタフェース構築支援の考え方を提案する。

1) 圧延機の絵やつまみの絵といったような応用ソフト固有の絵柄で画面が構成できる。この段階では、ユーザインタフェースの機能の細かい部分は気にせず、絵を自由に描く感覚でできることが必要である。

2) ユーザがマウスなどを使って操作を加えると、それに応じて絵柄を変化させたり、色や位置を変化させることができる。すなわち個々の絵柄に直接的な対話機能を組み込むことができる。

このような非定形ユーザインタフェース構築支援を行えるシステムを非定形 UIMS と呼ぶことにする。非定形 UIMS は、対象となるオブジェクトを標準部品から作られた定形的なものから、任意の表示上の形を持ちそれに応じた対話機能を備えたものに変えることにより構築できる。このように MU カーネルを土台として非定形ユーザインタフェース構築支援を実現する時の主な検討課題は次のようになる。

(1) オブジェクトの概念の拡張

MU では、オブジェクトごとにユーザ操作を感知していろいろな動作を行うことによりユーザインタフェースを構成している。非定形 UIMS では、このオブジェクトの外見が、定形 UIMS のように固定しておらず、円や矩形といった図形を使って自由に作成することが必要である。またその際に、ユーザ操作を感知する領域もその外見を反映していなければならない。

このような新しいオブジェクトを定形 UIMS の対話要素であるオブジェクトと区別するために、定形 UIMS のものを部品オブジェクト、非定形のものを図形オブジェクトと呼ぶことにする。

(2) 直接操作の定義の支援

定形 UIMS ではあらかじめ部品オブジェクトの種類が決まっていたため、メータの針をマウスに応じて動かすといったような直接操作の機能は、部品オブジェクトに特有のものがプログラムとして組み込まれていた。

一方、非定形 UIMS では、部品オブジェクトとして決められたものがなく、あらかじめ直接操作の動きを用意することができない。そのため、ユーザインタ

フェース作成時に図形オブジェクトに対して動作の定義を行う必要がある。このような直接操作を図形オブジェクトに定義するための支援環境を用意する必要がある。

次章ではこれらの図形オブジェクト、直接操作の定義の詳細について検討する。

4. 非定形 UIMS の検討

4.1 図形オブジェクトの概念

4.1.1 図形オブジェクトの基本機能

ユーザインタフェースを構成する上で図形オブジェクトに求められる機能は以下のようなものである。

(1) 構成単位

非定形 UIMS においては、描かれた絵に対して自由にユーザインタフェースが定義できるようにする。そのため図形オブジェクトの構成基本単位は円、直線といった細かなレベルで扱う必要がある。

また、図 4(a)は圧延ロールの詳細図であるが、このようにたくさんの直線の図形を組み合わせで作られる圧延ロールを1つの図形として扱いたい時もある。つまり、大きなレベルに図形をまとめて扱うグループ化の機能も必要になる。

(2) グループ化における特殊機構

オブジェクトのグループ化と、絵を描くのに使われる図形エディタで一般に言われるところのグループ化とは少し意味が異なる。図形エディタでは一度描いてグループ化した図形全体に対し色などを設定することはあまり必要とされない。しかしユーザインタフェースに使用される図形はグループ化してもオブジェクトとして扱われ、色を全体的に変えたりといった要求が高い。

例えば図 4(a)のようなグループ化した図形オブジェクトに対し、図 4(b)のように輪郭を太くするよう

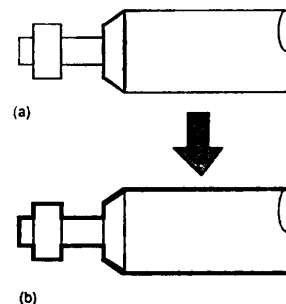


図 4 グループの一部だけの属性変更
Fig. 4 Example of changing some members' attributes in grouped objects.

なハイライト表示を行いたいという場合がある。

この時、線の太さは線属性を変えることで容易に変更できるが、すべてのグループ構成要素の属性を同じに設定するのでは図4(b)は実現できない。

グループ化した時には同時に、グループの一部の属性を変更できるような細かな属性の制御を行う機構も設ける必要がある。

(3) グループに特有のイベント時の動作定義

圧延機の故障診断では、診断の初期にはスタンドの単位で故障診断を行い、診断が進むにつれて圧延ロールの単位の診断、軸受け部の診断というように詳細な診断に移る。つまり、スタンドというグループ図形がイベント入出力の単位になる場合もあるし、圧延ロールを表す円プリミティブが入出力単位になることもある。

このようにイベントが発生した時、ある場面ではグループ単位で動作を行い別の場面では構成要素単位に行うといった動作の切り換え機能が必要である。

4.1.2 図形オブジェクトの実現

(1) 図形プリミティブオブジェクトとグループ化オブジェクト

前項(1)の機能は、図形オブジェクトを図形プリミティブオブジェクトとグループ化オブジェクトから構成することで実現する。図形プリミティブオブジェクトは直線や円、弧などの表示形態を持つ。そして線の色や、内部の塗り潰しのパターンなどの情報を属性として持っている。

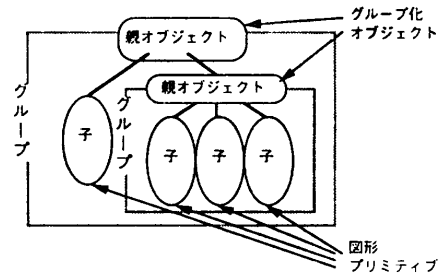
グループ化オブジェクトはグループ構成要素の親オブジェクトとなり管理を行う(図5(a)参照)。この時にグループ構成要素としては、図形プリミティブオブジェクトだけでなく他のグループ化オブジェクトも扱うようにする。これにより多階層のグループを実現できる。

(2) 属性設定禁止と参照先設定

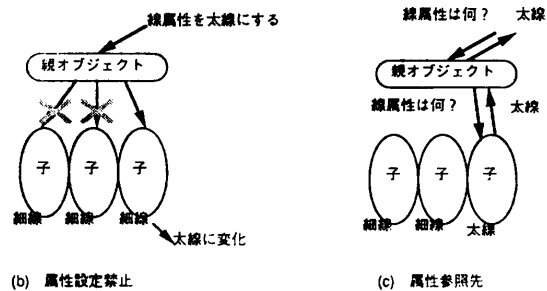
前項(2)の属性の細かな制御は、選択的な属性設定機能と、選択的な属性参照機能によって行う。

選択的な属性設定機能は、図形オブジェクトに属性設定禁止の機構を設けることで実現する。言い換えるとグループに対する属性の更新処理は、禁止されていないすべてのグループ構成要素の属性を更新するという形で行われる(図5(b)参照)。

選択的な属性参照機能は、グループ化オブジェクトに属性参照先設定の機能を設けることにより実現する(図5(c)参照)。すなわちグループ化オブジェクトに



(a) グループの構成



(b) 属性設定禁止

(c) 属性参照先

図5 図形オブジェクトの実現方法

Fig. 5 Treatment for attributes of grouped objects.

対する属性の問合せは、この参照先の図形オブジェクトの属性を問い合わせるという処理になる。

(3) イベント名の切り換えによる受け渡し先の切り換え

前項(3)の場面によるイベントに対する動作対象の切り換え機構は、MUにあるイベント名切り換え機能によって対処する。

例えば図6のように、グループ化オブジェクト「スタンド1」にイベント名「マウスイベントA」を、グループ構成要素である「ロール5」にイベント名「マウスイベントB」を割り当ててそれぞれにアクションを定義する。

診断実行時に診断の進行状況に応じ、ユーザもしくは診断プログラムからイベント名切り換えを指示する。マウスイベントが発生した時、MUカーネルは指示されているイベント名のイベントをスタンド1に送る。スタンド1はイベント名が「マウスイベントA」であれば自分の内部に定義されてアクションを実行し、そうでなければそのイベントを構成要素である「ロール5」へ送る。

このようにして場面によってイベントに対する動作対象を変えることができる。

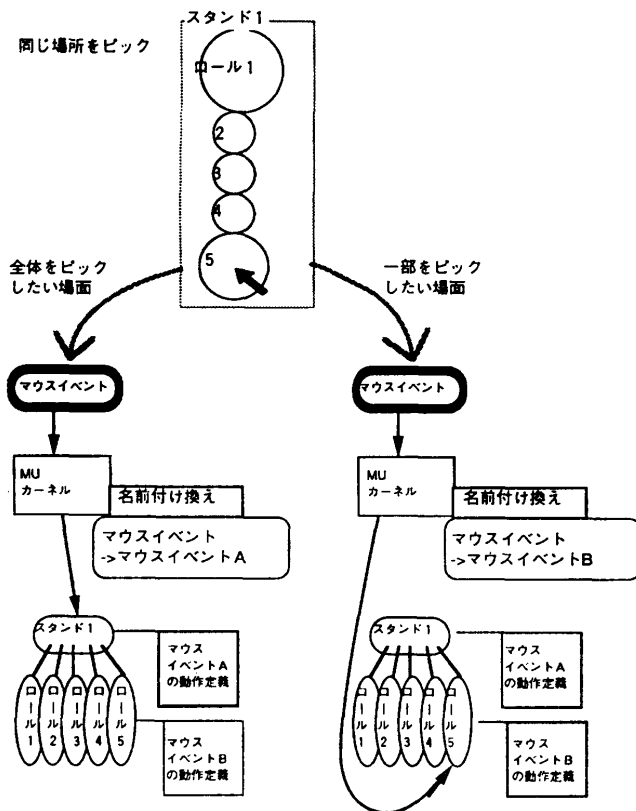


図 6 イベントの場面による切り換え

Fig. 6 Exchanging of event name in a different scene.

4.1.3 図形オブジェクトの作成方法の例

圧延機を例にとり、非定形 UIMS において図形オブジェクトを作る手順を説明する。

(1) まず、非定形 UIMS を立ちあげる。画面は図 7 (a) のようなものになる。

(2) 編集メニューを用いて、通常のグラフィックエディタのように UI 画面を作成する。画面は円、直線といった図形のプリミティブのオブジェクトを用いて描いていく (図 7 (b))。

(3) さらに操作対象となるべき図形オブジェクトをグループ化してまとめる。図 7 (c) では円を数個まとめてスタンド部を作っている。

(4) でき上がった図形オブジェクトに対してイベント時の動作やアクションを定義していく、図 7 (d) では、スタンド部にマウスボタンの押下を感知したら、詳細図を出すように定義をしている。この部分は 4.2 節で詳しく述べる。

4.2 直接操作の定義方法の検討

この節では、非定形 UIMS における直接操作の実現と定義方法について検討する。

4.2.1 対象とする直接操作

図形を用いた直接操作には、大きくわけて 2 種類ある。1 つはピック操作によるものであり、もう 1 つはドラッグ操作によるものである。

(1) ピック操作

ピック操作によるものには、例えば“ボタン”と呼ばれるものがある。ボタンはピック操作が行われることによりその図柄が変化し、あたかも本当のボタンスイッチであるかのようにスイッチが入る。それによって診断プログラムを起動したり、故障状況の設定をしたりする。

このようなボタンの機能は MU の仕組みに既に入っており、特に問題なく実現できる。

(2) ドラッグ操作

ドラッグ操作によるものには、圧延機の例でみたようなスライド入力装置がある。このドラッグ操作は、ピック操作に比べて複雑なイベント処理を含む。

非定形ユーザインタフェースでは、図 7 で示したように図形オブジェクトを作成したあとでオブジェクトの機能を決めるという手順を取る。すなわちドラッグ操作のような複雑なイベント処理も、後から図形オブジェクトに組み込

むといった手順が必要になる。

以下では、ドラッグ形式の直接操作を詳しく分析し、その定義方法の一般化を行う。

4.2.2 ドラッグ形式の直接操作の分析

圧延機の故障診断において、診断システムがシートを開いてユーザにバルブの圧力値を問い合わせる場面がある (図 3 (c) 参照)。このシートにはスライド入力装置状の絵柄が描かれており、そのつまみの部分はマウスでドラッグすることによって移動することができる。

このつまみは絵柄のある限られた区間のみ移動できる。つまみの右側の入力フィールドには、ボタンの位置に応じた値が現れるようになっていて、つまみが左端では 0 で右端では 100 である。バルブの圧力値を適当に設定したら、ユーザは完了のボタンの絵柄をピックしてシートを閉じる。

一方、バルブの値を設定するのにスライド入力装置を用いず行いたい場合もある。このような時は入力フィールドにキーボードから直接入力して、入れられた値はスライドの位置に反映されるというスタイルが用

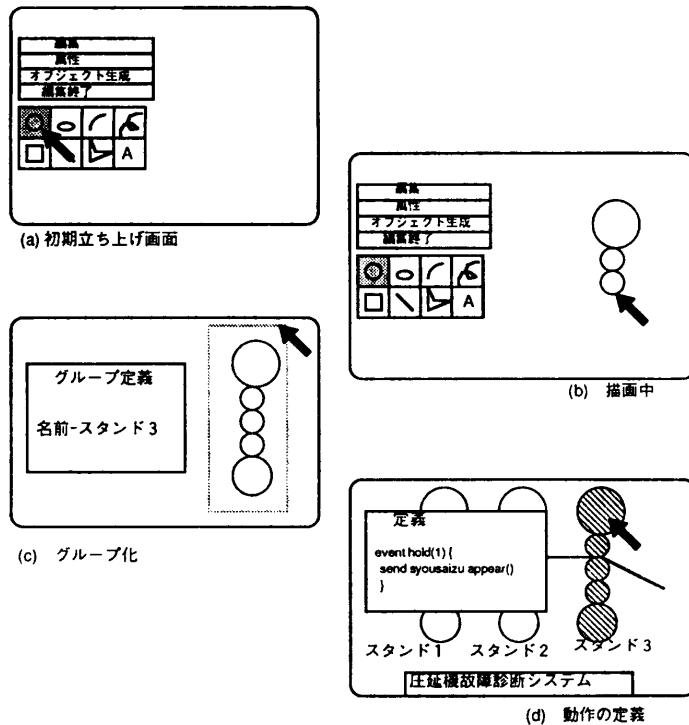


図 7 ユーザインタフェース・エディタによる
非定形ユーザインタフェース構築
Fig. 7 Example using user interface editor.

いられる場合が多い。入れ終わったら同じように完了のボタンをピックしてシートを閉じる。診断システムはこの入力値を使って診断を続ける。

このようにドラッグによる直接操作を行う図形（入出力図形と呼ぶ）を特徴づける要素は、

- (1) どのように操作するかという「操作方法」
 - (2) その操作はどのような領域制限があるかという「操作可能範囲」
 - (3) 操作をした時にどういう値として入力されるかという「値域とそのマッピング」
- であることが分かる。

4.2.3 ドラッグ形式の直接操作の一般化

(1) 操作方法についての一般化

いろいろな応用システムに関して直接操作を比較・検討すると、それらのほとんどは、(a)一次元移動、(b)二次元移動、(c)回転、(d)変形、に分類できることが分かった。

(a) 一次元移動

一次元移動はスライド入力装置に見られるように、図形をドラッグして動かしその図形的位置によって入力値を与える操作方法である。

(b) 二次元移動

二次元移動は、マウスカーソルのように平面上の位置によって入力値を一度に2つ与える操作方法である。

(c) 回転

回転はダイヤルなどに見られるように、図形をドラッグして回転させ、その角度によって入力値を与える操作方法である。

(d) 変形

変形についてはいろいろな種類がある。例えば棒グラフの端を目的の値までドラッグして引き伸ばす入力や、扇形の辺をドラッグして大きさを設定する入力などがある。

変形は、他の操作に比べてバリエーションが多く抽象化が難しい。我々のシステムでは扱う対象を図形プリミティブの変形にとどめ、グループ化図形は対象外とする。

このように操作として移動、回転、変形といった抽象的なものを用意することにより、メータとダイヤルといったように機能の異なったものでも、その操作形態に着目して同じように定義できる。

(2) 操作可能範囲の一般化

操作可能範囲は操作方法に応じて変る。(1)に則していえば、一次元移動の場合は、直線区間が操作可能範囲である。二次元移動は、その名のとおり平面が操作可能範囲である。回転の場合には、角度範囲である。変形では、例えば棒グラフの変形では直線区間だが、扇形の変形においては角度範囲となる。

(3) 値域とマッピング

一次元移動、回転、変形とも1つの値域と線形なマッピング関数が一般的に用いられる。

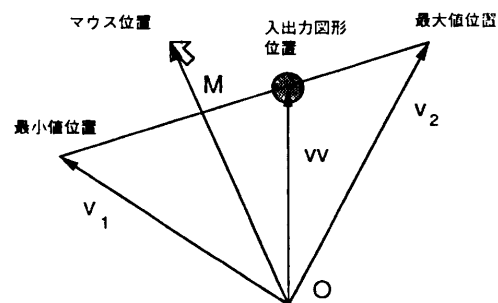


図 8 マウス位置と入出力図形的位置との関係
Fig. 8 Geometric relationship between a mouse cursor and an I/O figure-object.

例えば図8に示すような一次元移動による入出力図形があった時に、入出力図形の位置 vv (位置ベクトル) およびその時の入力値は以下のようにして導き出される。

$$(M-t*(v2-v1)-v1)*(v2-v1)=0$$

を満たす t において

$$vv=t*(v2-v1)$$

ただし $t < 0$ の時 $t=0$, $t > 1$ の時 $t=1$ とする (1)

$$(\text{入力値}) = |vv-v1|/|v2-v1|$$

$$*(\max-\min)+\min \quad (2)$$

二次元移動に関しても値域を2つに拡張することによって対応できる。またこのほかに、操作量の $\log$ や二乗をとって入力値にするようなマッピングも考えられる。

4.2.4 ドラッグ形式の直接操作の実現

および定義方法

前項の一般化を基に、ドラッグ形式の直接操作を図形オブジェクトに組み込む手法として、次の方法を考案した。実際の定義のようすを合わせて図9に示す。

1) 一次元移動, 二次元移動, 回転等, ドラッグ操作の動きを表現するアクションを標準アクション部品として用意する。これにより動きの再利用といったことが可能になる。

2) 図形オブジェクトに直接操作を組み込む場合, どのアクションを使うかを指定する。すなわちその図形オブジェクトのイベントに対する動作定義として, どのアクション部品を実行するかを指定する。実際には, これらが項目として選べるように定義シート上に用意する。

3) そのアクション部品の動作仕様を決める次のパラメタを指定する。

(a) 移動域, 回転域などの操作可能範囲。

(b) その動作域と対応する値域 (最大・最小値)。

例えば, 定義シートを閉じると入出力図形の種類に応じた操作可能範囲の初期状態が示される。種類として一次元移動を指定した場合には, 直線区間が示される。ユーザはこの直線をマウスで直接操作して変形させることによって操作可能範囲を定義する, という具合である。

5. おわりに

あらかじめ決められた種類のアプリケーションプロ

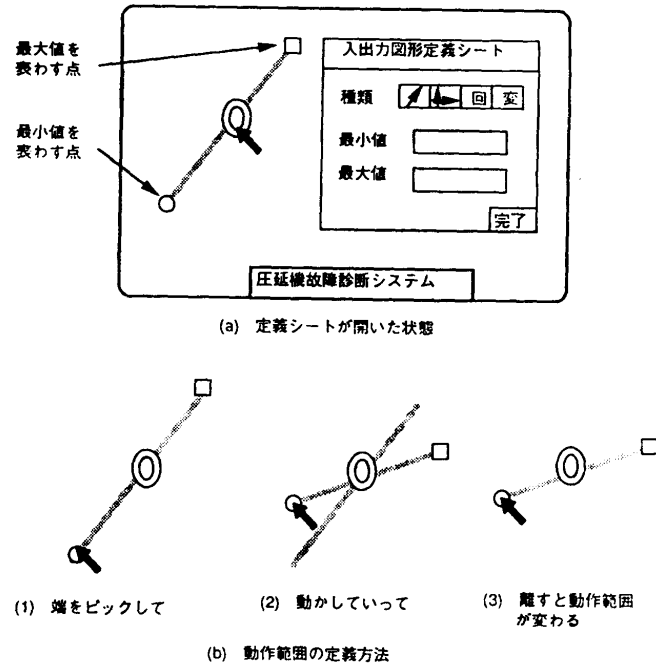


図9 直接操作の定義例 (一次元移動)
Fig. 9 Example definition for I/O figure (1D moving).

グラムに対してではなく, 多種多様なもののユーザインタフェース構築を支援する非定形 UIMS の基本的な考え方を以下のように提案した。

(a) 円, 直線といった図形プリミティブと, それらのグループ化によって作られる図形オブジェクトをユーザインタフェース作成の基本単位とする。

(b) この図形オブジェクトを生成・加工する機能を盛り込んだユーザインタフェース・エディタによってユーザインタフェースを作成する。

(c) ユーザがドラッグ形式の直接操作の定義を図形オブジェクトに対して行うことを支援する環境を提供する。

これより, 自由度の高い非定形ユーザインタフェース構築を行うことができる。

この非定形 UIMS の課題としては, 直接操作の定義をボタンで行う方法の一般性である。特に変形としてまとめた直接操作は, 種々さまざまなものを含んでいる。シミュレーションやサイエンティフィック・ビジュアライゼーションなど非定形 UIMS の広い応用を考える時には, この部分をさらに検討することが必要である。

謝辞 本研究に当たり御助言, 御討論いただいた日立製作所ソフトウェア工場 AI プログラム部の方々に

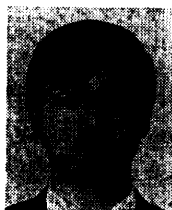
深謝いたします。

参 考 文 献

- 1) Thomas, J. J. et al.: Graphical Input Interaction Technique Workshop Summary, *ACM Comp. Graph.*, Vol. 17, No. 1, pp. 5-30 (1982).
- 2) Pfaff, G. E.: *User Interface Management Systems*, Springer-Verlag (1985).
- 3) Olsen, D. R., Jr. (ed.): ACM SIGGRAPH Workshop on Software Tools for User Interface Management, *ACM Comp. Graph.*, Vol. 21, No. 2, pp. 71-147 (1987).
- 4) 横山, 谷, 荒井: ユーザインタフェース構築支援システム, 第35回情報処理学会全国大会論文集, pp. 929-934 (Sep. 1987).
- 5) Young, D. A.: *X Window Systems Programming and Applications with Xt*, Prentice Hall (1989).
- 6) Desoi, J., Lively, W. and Sheppard, S.: Graphical Specification of User Interfaces with Behavior Abstraction, *ACM CHI '89 Proc.*, pp. 139-144 (May 1989).
- 7) 谷, 荒井, 谷越, 谷藤, 横山: 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, Vol. 30, No. 9, pp. 1200-1210 (1989).
- 8) Barth, P. S.: An Object-Oriented Approach to Graphical Interfaces, *ACM Trans. Gr.*, Vol. 5, No. 2, pp. 142-172 (1986).
- 9) Sibert, J. L. et al.: An Object-Oriented User Interface Management System, *ACM SIGGRAPH '86 Proc.*, Vol. 20, No. 4, pp. 259-268 (1986).
- 10) 荒井, 谷越, 谷: ユーザインタフェース構築支援システム—メタユーザインタフェース構築支援—, 第38回情報処理学会全国大会論文集, pp. 1271-1272 (Mar. 1989).

(平成元年8月21日受付)

(平成2年6月4日採録)



谷越浩一郎 (正会員)

1962年生。1985年東京工業大学理学部情報科学科卒業。1987年同大学院情報科学専攻課程修了。同年(株)日立製作所日立研究所に入社。ユーザインタフェース関連ソフトウェアの研究・開発に従事。マイマシニンタフェース, ヒューマンファクタ, プログラム言語/環境に興味をもつ。



荒井 俊史 (正会員)

1964年生。1986年東京工業大学理学部情報科学科卒業。現在、(株)日立製作所日立研究所にてユーザインタフェース関連ソフトウェアの研究・開発に従事。プログラミング言語, プログラミング環境, OS一般に興味をもつ。



谷 正之 (正会員)

1956年生。1980年東京工業大学工学部電気・電子工学科卒業。1982年同大学院物理情報工学専攻課程修了。同年(株)日立製作所日立研究所に入社。文書処理, 連想記憶, オブジェクト指向プログラミング環境の研究・開発を経て, ユーザインタフェース関連ソフトウェアの研究・開発に従事。現在 MIT メディアラボにて客員研究員としてユーザインタフェースの研究中。



横山 孝典 (正会員)

1959年生。1981年東北大学工学部通信工学科卒業。1983年同大学院工学研究科電気及通信工学専攻修士課程修了。同年(株)日立製作所入社。日立研究所にて図形認識, ユーザインタフェースの研究開発に従事。1987年より(財)新世代コンピュータ技術開発機構に外向。知識表現の研究に従事。電子情報通信学会会員。



谷藤 真也 (正会員)

昭和22年生。昭和48年早稲田大学理工学研究科修了。同年(株)日立製作所日立研究所入社。鉄鋼プロセスの計算機制御, 知識工学応用システムの研究に従事。現在はエンジニアリングワークステーションのソフトウェアおよびユーザインタフェースの研究を担当。計測自動制御学会, 電気学会などの会員。