3次元UIMSの実現（2）ー定義言語論

9H−5

朝日　宣雄、田中　昭二、李　殷碩
三菱電機（株）　情報電子研究所

1．はじめに

複雑な計算機システムを誰もが簡単に扱えるようにするためには、メタファの利用が有益である。本稿では、メタファをより効果的にユーザに提供するための3次元UIMSに関する定義言語論について述べる。

なお、本稿は、筆者らが構想している知的対話環境（Intelligent Interaction Environment with 3D UIMS：図1参照）のうち、3次元仮想操作環境構築部で利用する知識ベース部に相当する。

2．3次元仮想操作環境構築部における定義言語論

2.1　研究背景

現在のウィジットを中心としたグラフィカルユーザインタフェース（以下、GUI）の表現力の限界の問題およびアプリケーション分離の問題を解決するために、筆者らは(1)3次元アニメーションを用いてより現実に近い仮想的な操作環境を構築でき、(2)ユーザの操作に応じて対話的にアニメーションを表示しながら複数のアプリケーションを実行する機能を備えた3次元UIMS−MECOT（Metaphor Environment COnstruction Tools）ーを構築した。

MECOTは、UIMSであるため、プレゼンテーション記述およびダイアログ記述[1]によってインタフェースの表示および記述を定義し、それを実行することにより任意の3次元仮想操作環境を表示／制御する。これらの記述を行う定義言語の設計にあたり、備えるべき性質として、構造性、記述容易性および汎用性を考慮した。MECOTの定義言語は、独自の構築モデル論[2]に基づくことにより、これらの性質を満たすように設計されている。

以下、本定義言語の構成および詳細を論じる。

2.2　基本概念

(1) 構造性

図2にMECOTの定義言語の構成を示す。本構成はMECOTの構築モデルで採用する次の階層的部品化に従って構造化されている。

- コンポーネント：　3次元形状（Polygon Component）、または、従来型GUIのウィジェット（Widget Component）からなり、部品化の最も下位レベルを構成する。図2では、ポリゴンコンポーネント定義、ウィジェットコンポーネント定義がこれに相当する。
- クラスオブジェクト：　コンポーネントを3次元的に配置することにより構成する部品。高々有限個の論理状態をもち、状態遷移によってその動作を定義する。図2では、クラスオブジェクト定義がこれに相当する。
- 操作環境：　クラスオブジェクトからインスタンスオブジェクトを生成し、それを3次元的に配置することにより構成する部品。複数のオブジェクトインスタンスが関与する動作、視点の移動など環境内でのみ定義可能な動作が定義される。図2では、アプリケーション定義、レイアウト定義がこれに相当する。

個々の定義は個別のファイルとして構成されており、参照はファイル名によって行われる。階層毎に独立性が高く、下位階層への参照変更の影響が他に伝搬しないため、仮想操作環境の変更が容易に行える。

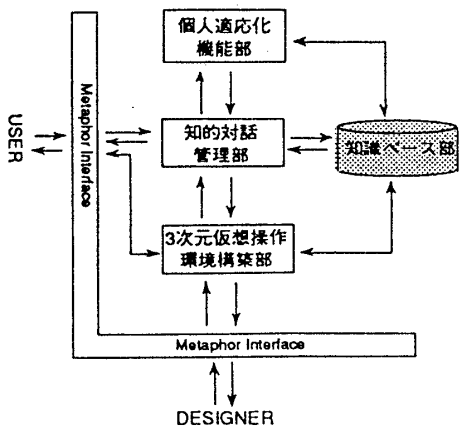

図1　知的対話環境の構成図

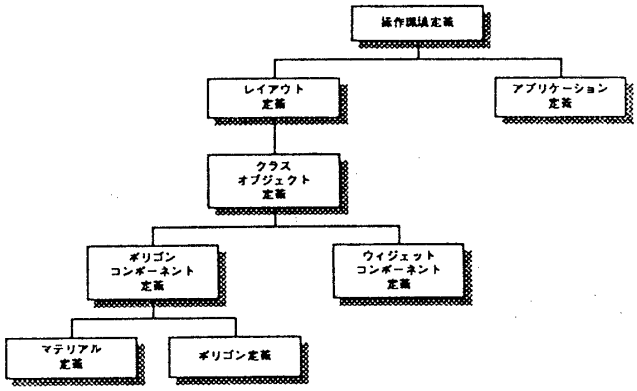

図2　定義言語の構成

(2) 記述容易性

3次元仮想操作環境を構成するために定義すべき項目は、3次元形状／配置、動作記述、アプリケーション記述の3つに大きく分類される。

- 3次元形状／配置：　基本的に3次元CADおよび独自の構築ツールの利用によって、座標をグラフィカルに設定できるシステムとするが、コンポーネント／オブジェクトの配置については、親子関係およびそれに基づいたローカル座標定義をテキスト形式で表現するため、テキストエディタによる編集も可能としている。
- 3次元動作記述：　移動、回転、スケーリングなどのプリミティブの組み合わせで全ての動作を記述することは困難であるため、概念レベルの高い記述による定義を可能としている。クラスオブジェクトでは、状態遷移ルールによる動作記述を採用し、論理状態とグラフィカル状態を対応させることにより容易な記述を実現している。操作環境では、ユーザおよびアプリケーションからのイベントを統一的に扱い、イベント駆動に基づくルール記述、マクロ動作関数などのサポートにより容易な動作記述を実現している。
- アプリケーション記述：　アプリケーションとMECOTとのインタラクションの形態に応じて、接続形態および定義を分離することにより、少ない記述でアプリケーションとの連携を表現可能としている。

(3) 汎用性

3次元形状／配置については、ポリゴン定義に基づく任意

"Implementation of Three Dimensional UIMS - Discussion on Definition Language"
Nobuo ASAHI, Shouji Tanaka, Eun-Seok LEE
Computer & Information Systems Laboratory,
Mitsubishi Electric Corp.

の３次元表示が可能である。３次元動作については、コンポーネントを単位とした移動、回転、スケーリングなどのプリミティブを用意することによりかなり細かい動作まで指定可能である。一方、記述容易性を満たす目的からコンポーネントを単位とした動作にとどめているため、コンポーネント変形などは実現されていない。

アプリケーション記述においては、様々なアプリケーション作成の過程で明かにする必要があるが、現在、電子メール、テキストエディタ、ファイル検索を統合した「３次元オフィス」を試作し、記述の有効性を確認した。

３．３次元仮想操作環境定義言語

図２の構成に基づいて、定義言語の詳細について述べる。紙面の都合上、コンポーネント定義、マテリアル定義については省略する。

3.1 クラスオブジェクト定義

(1) コンポーネント配置

コンポーネントの配置によりクラスオブジェクトの形状を定義する。通常はクラスオブジェクトエディタ[3]を用いる。

```
OBJ_DEF::= component_name (COMPONENT_DEF): O_SPEC
  { O_CHILDREN }
COMPONENT_DEF::= polygon_def, material_def | widget_def
O_SPEC::= POS, ROT, SCALE, SIZE (*1) | POS, SIZE, DISP (*2)
POS::= position ( x, y, z)
ROT::= rotation (x, y, z)
SCALE::= scale (x, y, z)
SIZE::= size(x, y, z)
DISP::= display(yes) | display(no)
O_CHILDREN::= <empty> | O_CHILDREN OBJ_DEF
(*1) for polygon components  (*2) for widget components
```

(2) 属性

属性には、attributeとpropertyがあり、attributeはアプリケーションなどから得られる値を格納する変数、propertyは例えば、「ドラッグで移動可能」などのクラスオブジェクトの操作／挙動を規定する属性である。

```
ATTRIBUTE_DEF::= <empty> | ATTRIBUTE_DEF ATTRIBUTE
ATTRIBUTE::= TYPE attribute_names
TYPE::= integer | float | bool | string

PROPERTY_DEF::= <empty> | PROPERTY_DEF  PROPERTY
PROPERTY::= select | drag | grab | gravity | on | rotate | move |
  hilite | label
```

(3) 状態記述

論理状態およびグラフィカル状態の定義を行う。

```
STATE_DEF::= logical_state: | logical_state(action_name):
  GRAPHICAL_STATE
GRAPHICAL_STATE::=  component_name:  <S_SPEC,>*
  {S_CHILDREN}
S_SPEC::= POS | ROT | SCALE | VISIBLE | DISPLAY |
  WIDGET_ACTIONS
S_CHILDREN::= <empty> | S_CHILDREN GRAPHICAL_STATE
```

(4) 動作記述

状態遷移ルールを記述する。

```
O_BEHAVIOR::= logical_state -> logical_state :
                trigger:{ <TRIGGER>* }
                effect:{ <EFFECT>* }
TRRIGER::= EVENT(component_name)
EFFECT::= action_name | ASSIGNMENT_STATEMENT |
  ITERATION_STATEMENT | SIMULTANEOUS_STATEMENT |
  SELECTION_STATEMENT
  (以下略)
```

3.2 レイアウト定義

(1) インスタンスオブジェクト生成

クラスオブジェクトからインスタンスオブジェクトを定義する。通常は操作環境ビルダ[3]を用いる。

```
INST_DEF::= class_name(filename): <instance_name,>*
```

(2) インスタンスオブジェクト配置

インスタンスオブジェクトを配置し、操作環境の初期状態を定義する。通常は操作環境ビルダを用いる。

```
LAYOUT_DEF::= instance_name: PARENT {L_SPEC}
PARENT::= <empty> | instance_name->component_name
L_SPEC::= POS, ROT, SCALE
```

(3) 初期状態／初期属性値

インスタンスオブジェクトの初期状態、初期属性値を定義する。

```
INIT_STATE::= instance_name : state_name
INIT_ATTRIB::= instance_name : {<ATTRIB_SETTING>*}
ATTRIB_SETTING::= attribute_name = value
```

3.3 操作環境定義

(1) 仮想エリア

概念レベルの高い動作記述を可能とするために、移動先の名称として利用可能な仮想エリアを定義する。仮想エリアの大きさ、位置は操作環境ビルダで指定する。仮想エリアは移動先の名称として利用した場合、インスタンスオブジェクトが移動可能な座標値を返す。

```
AREA_DEF::= area_name: {AREA_SPECS}
AREA_SPECS::= ON, POS, ROT, SIZE, CELL_SIZE
ON::= on(componen_name)
CELL_SIZE::= cell(x, y, z)
```

(2) 視点設定

視点移動動作に利用するために操作環境内に複数の視点を予め定義する。通常は操作環境ビルダを用いる。

```
VIEW_DEF::= viewname: {VIEW_SPECS}
VIEW_SPECS::= VIEW_POINT, VIEW_VECTOR, VISION, CLIP-
  PING
VIEW_POINT::= view_point(x, y, z)
VIEW_VECTOR::= view_vector(x, y, z)
VISION::= vision(angle)
CLIPING::= clipping(near, far)
```

(3) 動作記述

複数オブジェクトが関与する動作、アプリケーションからのイベントに基づく動作を定義する。

```
E_BEHAVIOR::= EVENT <PRECONDITION>* {EFFECT}
EVENT::= FROM ->event_message
FROM::= user | application
PRECONDITION::= IS_PARENT_OF | IS | IS_A
EFFECT::= ACTION | ASSIGNMENT_S | ITERATION_S |
  SIMULTANEOUS_S | SELECTION_S
ACTION::= CREATE_INST | SEND_EVENT | FLY | HIDE |
  TRANSLATE | ROTATE | SCALE_CHANGE
  (以下略)
```

3.4 アプリケーション定義

外部プロセスとのプロセス間通信により実行するアプリケーション（プロセス型）とオブジェクトファイルを実行時にリンクし動作記述内で関数として実行するアプリケーション（ファンクション型）を定義する。

```
PROC_APP::=  application_name:{PATH  TO_MESSAGE
  FROM_MESSAGE}
PATH: path::= execution_file
TO_MESSAGE::= to_message:{<MESSAGE>*}
FROM_MESSAGE::= from_message:{<MESSAGE>*}
MESSAGE::= message(TYPE)

FUNC_APP::= TYPE function_name(ARGS): {object_name ,
  library_name}
```

４．まとめ

本稿では、計算機に不慣れなユーザでも容易にシステムイメージを理解できるインタフェースの実現を目標とした３次元ＵＩＭＳについてその定義言語論を述べた。本定義言語は、構造性、記述容易性、汎用性を満たし、様々な仮想操作環境を容易に定義することが可能となる。

なお本論文の一部は通産省の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

[1] Olsen:"User Interface Management Systems: Models and Algorithms",Morgan Kaufmann Publishers, 1992.

[2] 朝日他:３次元ＵＩＭＳの実現（１）ー構築モデル論,第46回情報処理学会全国大会, 1993.

[3] 田中他:３次元ＵＩＭＳの実現（３）ー試作と評価,第46回情報処理学会全国大会, 1993.