データベース・システム　75－4
（１９９０．１．１８）

オブジェクト指向データベースにおける マルチメディア文書の管理
－ＭＯＲＥに基づいた考察－

堀尾　祥久*、　津田　和幸*、　平川　正人**、　市川　忠男**
広島大学大学院工学研究科*　　広島大学工学部**

文書の管理はオフィス業務の最も重要なものの１つであり、取り扱おうとする文書は文字だけでなく、図表、絵、写真などといった種々の異なった形態のデータを含んでいる。したがって、これらのデータを統一的に管理するとともに、ユーザからの多種多様な操作要求に応えられるような文書管理が要求されている。本稿では、我々が既に提案しているオブジェクト指向データモデルＭＯＲＥに基づく文書管理について述べる。ＭＯＲＥでは、１つのオブジェクトに複数のビューを持たせることが可能であり、操作の局面に応じてビューを変更することができる。この機構を用いることにより、ユーザの思考形態に応じた形式で文書の作成が行える。さらに、文書を作成するためのインタフェースも文書データと同様にデータベースで管理しており、インタフェースをその記述から生成する機構について述べる。

Management of Multimedia Documents in an Object-Oriented Database

Y. HORIO, K. TSUDA, M. HIRAKAWA, and T. ICHIKAWA
Information Systems, Faculty of Engineering, Hiroshima University
Shitami, Saijo-cho, Higashi-Hiroshima 724, Japan

Managing various kinds of documents is one of the most important tasks in office environments. Since the documents to be managed have a variety of data types, such as texts, figures, and images, the database systems must provide the efficient facilities for the management of such data types. And also they must provide the flexible facilities in order to respond the arbitrary user's requests. This paper proposes the management of documents based on an object-oriented data model, MORE. In MORE, each object can take more than one view allowing the user change the view freely at each phase of requests. By using this mechanism, the user can create a document in a form just to fit to his/her own image. In addition, we propose an approach to develop an interface for document creation. The interface is managed by a database as well as document data.

1．はじめに

文書管理はオフィス業務において最も重要なものの1つである[1],[2]。そこで取り扱われる文書は文字だけでなく、図表、絵、写真などといった種々の異なった形態のデータを含んでいる。これらのデータを統一的に管理するとともに、ユーザからの多種多様な操作要求に応えるには、従来のデータベースシステムでは十分でない。そこで最近では、文書管理の新しいフレームワークとして、ハイパーテキスト／メディア[3]やオブジェクト指向データベース[4],[5]が注目されている。

ハイパーテキスト／メディアでは、個別の情報をノードとして蓄え、それらのノード間に意味的つながりを表現するためのリンクが自由に設定できるようになっており、データベースの記述の自由度は高い。しかしながらその反面、データベースの構造を規定するスキーマといった概念がないため、データベース作成にあたってはユーザに膨大な作業を押しつける。また、検索は、基本的にリンクに沿ってノードを次々に移動することに相当し、膨大なサイズのデータベースの中から必要な情報を迅速に引き出すことは困難である。しかも、検索にあたって、ユーザはリンク構造を前もって理解しておくことが必要となる。

一方、オブジェクト指向データベースでは、データの構造が規定できるため、データ（オブジェクト）間のリンクの設定をシステム側でサポートできるだけでなく、柔軟な検索を行なうための検索言語も用意されている。しかしながら、任意のオブジェクト間に自由にリンクを設定することができず、データベース記述の自由度はハイパーテキスト／メディアに一歩譲る。

我々はＭＯＲＥと呼ぶオブジェクト指向データモデルを提案している[6]。ＭＯＲＥでは、従来のオブジェクト指向データモデルの特長に加えて、メッセージによるオブジェクト構造の決定機構、ならびにユーザのクラス定義を支援するクラスタイプというアイデアが導入されている。特にオブジェクトの構造決定機構では、オブジェクトに複数の構造（ビュー）を持たせることが可能となっており、データベース操作の局面に応じて、ビューを次々に変更することができる。なお、ビューの変更は、メソッドの実行依頼と同様に、オブジェクトにメッセージを送ることで行うことができる。

本稿では、ＭＯＲＥを用いた文書管理について述べる。オブジェクト構造の決定機構の導入によって、オブジェクト指向データベースの持つ効率性とハイパーテキスト／メディアの持つ柔軟性とを兼ね備えたシステムが構築できる。すなわち、ビューを変更することによって、ハイパーテキスト／メディアの持つ関連情報の引出しに対応する操作を実現するのみならず、文章作成においても、対象とする文章を様々な側面から作成していくことができる。

ＭＯＲＥでは、文書管理のためのスキーマが個々の文書（クラス）ごとに存在するが、そのスキーマに束縛されることなく、個々の文書が独自にそれぞれのデータ構造を持ちうる。このような柔軟な操作機能に加えて、文書の記述支援を行うために、本システムでは、設定されたオブジェクト構造ごとに、エディタ、メニュといったインタフェースを整備できるようにしており、インタフェースに表示されるエディタ、およびメニュなどもシステムで管理するようにしている。これによって、インタフェースに関する情報も通常のデータと同様に取り扱うことができる。

まず、2章で文書表現について述べる。3章では文書作成支援ツールについて述べ、4章で文書管理システムの構成について説明する。さらに5章に実現例を示し、そして最後にまとめを行う。

2．文書表現

文書管理は次の2つの側面から捉えることができる。

1）既に作成されている文書の検索
2）新たな文書の作成

効率的な検索を実現するためには、従来型データベースシステムのように、スキーマ構造を固定することが望ましい。ユーザは文章スキーマに従って特定の文書を探索したり、ある条件を指定してフィルタリング操作を実行することができる。一方、文書の新規作成に関しては、このような固定されたスキーマはユーザに対し様々な制約を与える。この場合、ハイパーテキスト／メディアの様な自由度の高い操作が要求される。

本章では、これらの要求を満たす文章管理システムの実現を目標とした、ＭＯＲＥモデルに基づく文書表現について議論する。

2．1　オブジェクト構造の変更と文書管理機能

ＭＯＲＥでは、クラスで定義されたアグリゲーション階層のサブセットを、ビューとして定義付けることができる。オブジェクトのビューはメッセージを送ることで評価される。この機能を用いることで、文書検索ならびに文書作成の局面において以下の様な操作が実現できる。

例えば、論文を表現するオブジェクトに対して、参考文献として引用されるための構造と完全な論文表現のための構造とが定義されている場合を考える。このとき、それぞれの構造に沿ってオブジェクトを見るためのメッセージを!refと!paperとする。まず最初にある条件を満たす論文を検索し、それらを参考文献の形式で出力することを考える。これは、以下のようにＳＱＬ風の問合せとメッセージ!refを用いて記述できる。

```
!ref (SELECT *
      FROM   Paper
      WHERE  Paper.author[?]="J.Jack")
```

このとき、著者の１人が"J.Jack"である論文を表現するオブジェクトが検索され、それらのオブジェクトにメッセージ!refを送ることによって、それが参考文献の形態で表示される。

次に、これらの論文のうちのいずれかに対してメッセージ!paperを与えることにより、論文の本体を表示させることできる。さらに、その論文中に引用されている参考文献のうちの１つにメッセージ!paperを与えることにより、対応する論文の全体を見ることができる。すなわち画面上ではハイパーテキスト／メディアの様にオブジェクト間に関連づけがなされているかの様に操作できるが、内部的にはこれらのオブジェクトは１つのオブジェクトとして管理されている。

またこの仕組みを用いることにより、ユーザは様々な構造を用いてオブジェクトを生成することができる。例えば、論文の場合、まず章構成を記述した後に各章における文章を記述する、といったことができる。また逆に、各章を記述していけば、章構成を自動的に作成することもできる。

2．2　問合せオブジェクト

問合せオブジェクトとは、ＰＯＳＴＧＲＥＳ[7]で実現されているような、問合せをオブジェクトの値として持つオブジェクトのことであり、このオブジェクトを用いて実際のオブジェクトを参照することができる。問合せをオブジェクトの値として持たせることにより、そのオブジェクトはいつでもデータベース内の最新のデータを参照している。問合せオブジェクトの例を図１に示す。図に示すように、この問合せオブジェクトは、クラスPaper内のinfo.kindで参照されるオブジェクトのうちで、値が"OODBMS"である論文を参照する。ここで、データベース内にオブジェクト指向データベースに関する論文オブジェクトが増えたとしても、問合せオブジェクトは更新された後の状態を参照している。

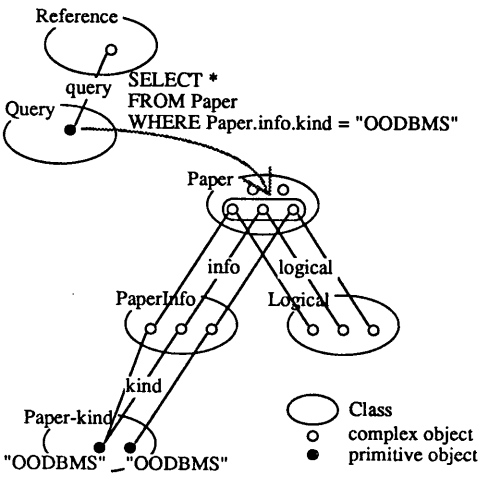
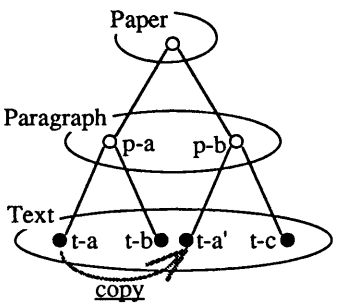
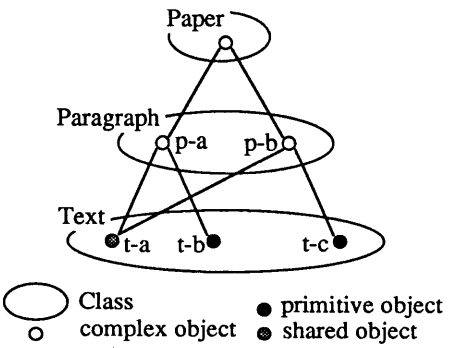


図１　問合せオブジェクトの例



（ａ）全オブジェクトのコピー



（ｂ）共有オブジェクトを持つコピー

図２　オブジェクトのコピー

2．3　コピー

一般に、オブジェクトの同一性を議論する際、オブジェクト識別子の同値とオブジェクトの持つ値の同値の２つを考える必要がある。従って、オブジェクトのコピーにも図2に示す２つを考慮する必要がある。図２（ａ）では、新しくオブジェクトを作成し、それに元のオブジェクトの持つ値と同じ値を持たせる。図２（ｂ）では、コピー先に対応したオブジェクトから、そのオブジェクトへのリンクを設定している。オブジェクトの更新が起こる前は、それら２つは画面上で同様に表示されるが、コピーされたオブジェクトの変更が行われた際には大きな違いが起こる。すなわち、図２（ｂ）ではオブジェクトの共有が行われているため、画面上ではそれに対応する全ての部分が変更される。

3．文書作成支援ツール

3．1 文書作成

２．１で述べたように、ＭＯＲＥでは１つのオブジェクトを様々なビューによって参照することができる。文章作成の過程においてこの特長が、操作の自由度を上げると共に、ユーザの思考形態に適した支援を実現できる。

ところで、オブジェクト指向データモデルでは、オブジェクト固有の操作系列をクラスのメソッドとして記述できる。これを用いることによって、単純（印字可能）オブジェクトの作成法およびその値の入力方法などの記述を与えることができる。しかしながら、論文オブジェクトの様な複合オブジェクトの作成やオブジェクト間への新たなリンク付加といった処理は、この方法では記述量が大きくなり、実現も困難となる。

そこで我々はこうしたオブジェクトの作成を支援するツールとして、オブジェクトの階層表現を基に構造化エディタを生成するエディタマネージャと、構造化エディタの起動およびオブジェクトへのメッセージパッシングを実現するメニュを生成するメニュマネージャとを提供する。以下３．２および３．３で、どのように構造化エディタおよびメニュを作成するかについて詳述する。

3．2　エディタマネージャ

エディタマネージャは、エディタ構成情報を基にデータ入力用の構造化エディタを作成する。エディタ構成情報はエディタクラスとエディタオブジェクトを用いて管理する。

エディタクラスのメソッド記述部には、エディタの機能がメソッドとして記述されており、エディタオブジェクトとはエディタクラスに属するオブジェクトのことである。エディタオブジェクトは以下の５つ組で定義される。

｛エディタ名，メニュ数，メニュ名，
構成情報，対象オブジェクト｝

各フィールドはそれぞれ次の意味を持つ。

エディタ名：タイトルバーに表示されるエディタの名前。

メニュ数：ポップアップメニュ内に表示されるメニュの数。

メニュ名：ポップアップメニュ内に表示されるメニュの名前とエディタクラス内に記述されているメソッドの名前との対応情報。

構成情報：作成する構造化エディタの構成情報を保持するオブジェクト。

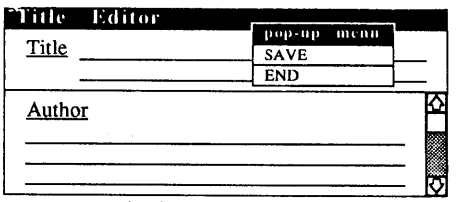
対象オブジェクト：このエディタで取り扱うオブジェクト。

```
CLASS
      NAME:      Title.Editor
CLASSTYPE:       EDITOR;
CONNECTION       SUBSET_of Editor;
CONSTRAINT
      STRUCTURAL-CONSTRAINTS
            name,       String,     1+;
            number,     Integer,    1+;
            menu,       Button,     N+;
            struct,     Struct,     1+;
            target,     Paper,      N;
METHOD
      !init x:      #connect target Self $x.
      !editor:      #disp_editor.
      !save:        target %title; target !createO.
      !end:         #quit_editor.
```

（ａ）クラスTitle.Editorのクラス記述

{Title Editor, 2, {(SAVE, !save) (END, !end)},
oid011, -}

（ｂ）エディタオブジェクトの例



（ｃ）エディタの例

図３　エディタの管理

例として、エディタクラスのサブクラスTitle.Editorのクラス記述を図３（ａ）に示す。また、エディタオブジェクトの例を図３（ｂ）に示す。

エディタの生成は、クラスTitle.Editorのメソッド!editorのアクション記述部 '#disp_editor'を実行することで行われる。#disp_editorの実行によってエディタオブジェクトがエディタマネージャに送られ、そのエディタオブジェクトの記述に沿って構造化エディタが生成される。生成された構造化エディタを図３（ｃ）に示す。このエディタのポップアップメニュの１つを選択すると、それに対応したクラスTitle.Editorのメソッドが起動される。

３．３　メニュマネージャ

メニュマネージャは、作成するメニュの構成情報を基にメニュを作成する。メニュ構成情報はメニュクラスとメニュオブジェクトを用いて管理する。

メニュクラスのメソッド記述部には、データ作成用のエディタの起動、あるいはサブメニュの表示を行うためのメソッドが記述されており、メニュオブジェクトとはメニュクラスに属するオブジェクトのことである。メニュオブジェクトは以下の５つ組で定義される。

｛メニュ名，ボタン数，ボタン名，
　サブメニュ，対象オブジェクト｝

各フィールドはそれぞれ次の意味を持つ。

メニュ名：タイトルバーに表示されるメニュの名前。
ボタン数：メニュ内に表示されるボタンの数。
ボタン名：各ボタンの名前とこのメニュオブジェクトが属するメニュクラスのメソッドの名前との対応情報。
サブメニュ：サブメニュのオブジェクトの集合。
対象オブジェクト：このエディタで取り扱うオブジェクト。

例として、メニュクラスのサブクラスPaper.Menuのクラス記述を図４（ａ）に示す。またメニュオブジェクトの例を図４（ｂ）に示す。

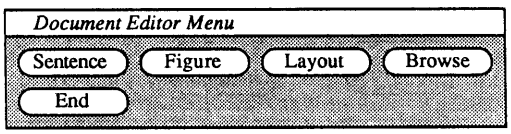
メニュの生成は、クラスPaper.Menuのメソッド!menuのアクション記述部'#disp_menu.'を実行することで行われる。#disp_menuの実行によってメニュオブジェクトがメニュマネージャに送られ、そのメニュオブジェクトの記述に沿ってメニュが生成される。作成されたメニュを図４（ｃ）に示す。このメニュのあるボタンを選択すると、そのボタンに対応したクラスPaper.Menuのメソッドが起動される。

```
CLASS
     NAME:      Paper.Menu
CLASSTYPE:      MENU;
CONNECTION      SUBSET_of Menu;
CONSTRAINT
     STRUCTURAL-CONSTRAINTS
          name,      String,     1+;
          number,    Integer,    1+;
          button,    Button,     N+;
          write,     TE.Menu,    1+;
          draw,      DE.Menu,    1+;
          layout,    LE.Menu,    1+;
          target,    Paper,      N;
METHOD
     %name:     button %name.
     %meth:     button %meth.
     %icon:     button %icon.
     !menu:     #disp_menu.
     !sentence: write !menu.
     !figure:   draw !menu.
     !layout:   layout !menu.
     !browse:   #connect target Self (#exec_Br <"Paper">).
     !end:      #quit_menu.
```

（ａ）クラスPaper.Menuのクラス記述

```
{Multimedia Document Editor Menu, 5,
     {(Sentence, !sentence) (Figure, !figure)
          (Layout, !layout) (Browse, !browse) (End, !end)},
          oid021, oid022, oid 023, -}
```

（ｂ）メニュオブジェクトの例



（ｃ）メニュの例

図４　メニュの管理

４．文書管理システム

実験システムを、我々が提案しているオブジェクト指向データベースシステムＭＯＤＥ[8]のオブジェクトスペース上に構築している。システム構成を図５に示す。

システムは、インタフェースのためのツールであるEditor Manager（ＥＭ）、Menu Manager（ＭＭ）、Event Interpreter（ＥＩ）、ならびにデータベースの操作ツールであるBrowser（Ｂｒ）、Message Interpreter（ＭＩ）、Space Database（ＳＤ）から構成される。

ＳＤは、作成するオブジェクトのクラス、オブジェクト、インタフェースを構成するメニュクラスと

エディタクラス、およびメニュオブジェクトとエディタオブジェクトを管理している。これらのクラスおよびオブジェクトは、ＭＯＤＥのオブジェクトデータベースから取り出される。

ＥＭは、ＳＤ内にあるエディタに関する情報を保持するエディタオブジェクトを用いて構造化エディタを生成する。

ＭＭは、ＳＤ内にあるメニュに関する情報を保持するメニュオブジェクトを用いてメニュを生成する。

ＥＩは、エディタとメニュに対するユーザの操作をイベントとして受け取り、そのイベントを解釈して、それに応じた動作を行う。

ＭＩは、インタフェースから送られてきたメソッドを解釈して実行する。

Ｂｒは、ＭＯＤＥのオブジェクトデータベースからデータ作成に必要なクラスおよびオブジェクトをＳＤに取り出す。

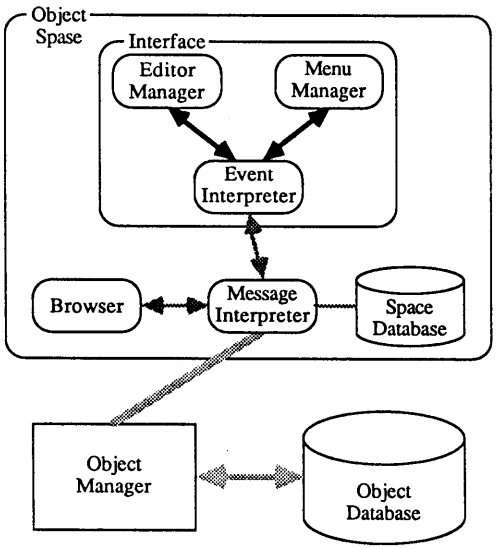


図５　システム構成図

次に、これらのコンポーネント間におけるシステムの動作について述べる。

画面に表示されているメニュあるいはエディタ上でボタンあるいはポップアップメニュを選択すると、そのイベントがそれぞれＭＭあるいはＥＭで受け取られ、ＥＩに送られる。ＥＩはそのイベントを解釈し、選択されたボタンあるいはポップアップメニュに対応するメソッド名がＭＩに送られる。ＭＩはそのメソッドを解釈して実行する。メソッドの種類により次のような４つの異なった動作をする。

（１）メニュ表示メソッドの場合

ＭＩは対応するメニュオブジェクトをＥＩを通してＭＭに送る。メニュオブジェクトを受け取ったＭＭはそのオブジェクトを基にメニュを作成し、表示する。

（２）エディタ表示メソッドの場合

ＭＩは対応するエディタオブジェクトをＥＩを通してＥＭに送る。エディタオブジェクトを受け取ったＥＭはそのオブジェクトを基に構造化エディタを作成し、表示する。

（３）オブジェクト登録メソッドの場合

ＭＩは登録するオブジェクトの構造を決定し、その構造に沿ってオブジェクトを作成し、オブジェクトマネージャに送り登録する。

（４）ブラウザ起動メソッドの場合

ＭＩはＢｒを起動する。問合せオブジェクトを作成する場合には、Ｂｒで作成された問合せがオブジェクトとしてＳＤ内に登録される。

例として、図４（ｃ）のメニュでボタンSentenceが選択された場合について述べる。まず、ユーザがメニュボタンを選択したというイベントがＭＭで受け取られ、ＥＩに送られる。ＥＩではそのイベントを解釈し、ボタンSentenceに対応するメソッド名!sentence（図４（ｂ）参照）がＭＩに送られる。ＭＩはクラスPaper.Menuのメソッド!sentence を起動する。 !sentenceのアクション記述部の記述に従って、ＭＩはwriteで参照するクラスTE.Menuにメッセージ!menuを送る。ＭＩはクラスTE.Menuで受理されたメッセージ!menu に対応したメソッド（メニュ表示メソッド）を選び、そのクラスのメニュオブジェクトをＥＩを通してＭＭに送る。ＭＭはそのメニュオブジェクトを基にメニュを作成し、表示する。

5　実現例

本章では、論文を作成するためのメニュとエディタについて述べ、その後、論文の作成手順について述べる。

5．1　メニュとエディタ

メニュとして図６のような４つのメニュを用意する。文書エディタメニュは最初に表示されるメニュであり、ユーザの操作はこのメニュから始る。テキストエディタメニュはテキストエディタを起動するためのメニュであり、文書エディタメニュ内のボタ

ンSentenceを選択すると表示される。ドローエディタメニューはドローエディタを起動するためのメニューであり、文書エディタメニュー内のボタンFigureをを選択すると表示される。レイアウトエディタメニューはレイアウトエディタを起動するためのメニューであり、文書エディタメニュー内のボタンLayoutを選択すると表示される。

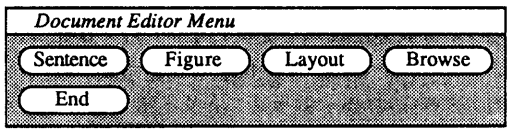
エディタとしては図7に示す3つのテキストエディタと、作図用のエディタ、グラフ・表作成用のエディタ、イメージを取り込むエディタという3つのドローエディタと、そしてレイアウト用のエディタの7つが用意されている。これらのエディタは、メニューにおいて該当する項目のボタンを選択することで起動される。具体的には、表題入力用のエディタは、テキストエディタメニューにおいて、ボタンTitleを選択すると起動される。章構成入力用のエディタは、テキストエディタメニューにおいて、ボタンContentsを選択すると起動される。本文入力用のエディタは、テキストエディタメニューにおいて、ボタンParagraph を選択すると起動される。作図用のエディタは、ドローエディタメニューにおいて、ボタンDrawを選択すると起動される。グラフ・表作成用のエディタは、ドローエディタメニューにおいて、ボタンGraphTableを選択すると起動される。イメージを取り込むエディタは、ドローエディタメニューにおいて、ボタンImageを選択すると起動される。レイアウト用のエディタは、レイアウトエディタメニューにおいてボタンOpenを選択すると起動される。

また、メニューに用意されているボタンBrowseの選択は検索ツールを起動する。

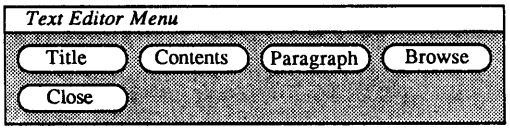
5．2　作成手順

論文を新規作成する場合、まずユーザは文書エディタメニューの中からボタンSentenceを選択する。するとテキストエディタメニューが画面上に表示される。このメニューの中のボタンTitle を選択すると、表題入力用のエディタが画面上に表示される。指示された領域内に論文の表題と著者名を入力する。その後にエディタのポップアップメニューからSAVEを選択するとクラスPaper に属するオブジェクトの構造が決定され、その構造に沿ってオブジェクトが作成され、オブジェクト間にリンクが設定される。オブジェクト構造に含まれる最下位のオブジェクトは具体的な値として論文の表題と著者名を持つ。

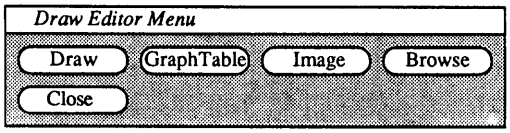
次に、論文の章構成を作成する。テキストエディタメニューにおいて、ボタンContentsを選択する。章



（ａ）文書エディタメニュー



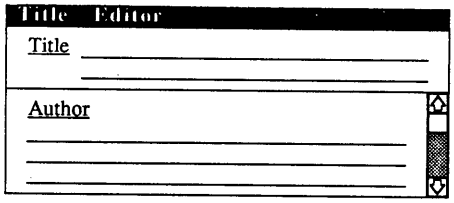
（ｂ）テキストエディタメニュー



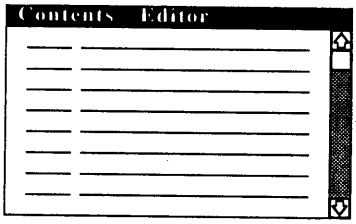
（ｃ）ドローエディタメニュー

（ｄ）レイアウトエディタメニュー

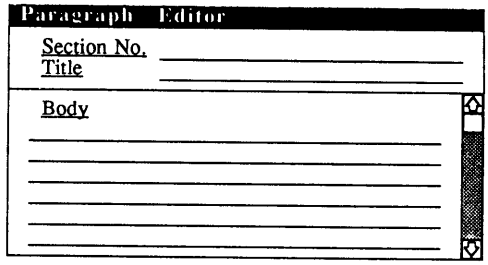
図6　メニュー



（ａ）表題入力用のエディタ



（ｂ）章構成入力用のエディタ



（ｃ）本文入力用のエディタ

図7　エディタ

構成入力用のエディタが画面上に表示されるので論文の章構成を作成し、その後、エディタのポップアップメニュからSAVEを選択する。この操作によって、クラスPaperに属するオブジェクト構造が決定され、その構造に沿ってオブジェクトが作成され、オブジェクト間にリンクが設定される。構造の最下位のオブジェクトは章の数だけ作成され、具体的な値として章のタイトルを持つ。

また、論文の本文については、テキストエディタメニュの中からボタンParagraphを選択し、本文入力用のエディタを表示させた上で、本文の作成を行う。図の作成、グラフ・表の作成は、ドローエディタメニュの中から該当するボタンを選択し、図作成用のエディタあるいはグラフ・表作成用のエディタを起動すればよい。論文のレイアウトは、レイアウトエディタメニュの中からボタンOpenを選択し、レイアウト用のエディタを起動して行う。それぞれのエディタのポップアップメニュの中からSAVEを選択すると、クラスPaperにメッセージが送られ、それに応じてオブジェクトが作成される。

問合せオブジェクトは検索ツールで作成される。例えば、本文中から他の論文の参照を行う場合、テキストエディタメニュの中からボタンBrowseを選択すると、検索ツールが起動される。検索ツールで参照する論文を検索し、そこで得られた検索条件（問合せ）を、オブジェクトの値としてシステムによって登録される。図の作成にあたって、他の論文中で使用されている図を参照する場合、ドローエディタメニュのボタンBrowseを選択すると、検索ツールが起動される。検索ツールで参照する図を検索し、そこで得られた問合せがオブジェクトの値として用いられる。

論文の修正の場合は、メニュを選択して、それぞれのエディタを起動すると、エディタ上に修正する論文のデータが既に表示された状態になっており、そこでデータの修正を行う。それぞれのエディタのポップアップメニュの中からSAVEを選択すると、クラスPaperにメッセージが送られ、それに応じてオブジェクト構造の最下位のオブジェクトの値が変更される。

ここでは、論文の新規作成手順として、章構成を記述した後、各章における文章を記述するといった手順について述べたが、逆に各章から先に記述していくこともできる。この場合章構成は自動的に作成されており、章構成入力用のエディタを起動すると、章構成が表示された状態で現れる。

6．おわりに

本稿では、我々が既に提案しているオブジェクト指向データモデルＭＯＲＥを用いた文書管理について述べた。ＭＯＲＥのメッセージによるオブジェクト構造の決定機構により、決定されたビューに応じて文書を作成することができる。これによって柔軟な文書作成が行える。また、インタフェースにおけるエディタとメニュを通常のデータと同様にデータベース内に管理し、エディタとメニュをそれらの記述からシステムで作成する機構を導入した。これによって、エディタとメニュの概念的な記述を行うだけで、ユーザは目的にあったインタフェースを構築することができる。実験システムを現在、ワークステーションＳｕｎ－３上で構築中である。

今後、多種多様の構造を持った文書データを取り扱うことができるようにするために、文書管理のためのクラス記述、メニュクラスとエディタクラスのクラス記述の作成をサポートするツールの開発、さらに、メニュオブジェクトとエディタオブジェクトの定義をサポートするツールの開発を行わなう予定である。

参考文献

[1] W. Hoark, "Office Document Architecture and Office Document Interchange Formats: Current Status of International Standardization," IEEE COMPUTER, Vol. 18, No. 10, pp. 50-60, 1985.

[2] 田口，坂下，"ＯＡシステムと文書データベース，" 情報処理，Vol. 28, No. 6, 1987.

[3] N. M.Delisle and M. D.Schwartz, "Contexts-A Partitioning Concept for Hypertext," ACM Trans. on Office Information Systems, Vol. 5, No. 2, pp. 168-186, 1987.

[4] D. Woelk, W. Kim, and W. Luther, "An Object-Oriented Approach to Multimedia Databases," Proc. of ACM SIGMOD 1986, pp.311-325, 1986.

[5] W. B. Croft and D. W. Stemple, "Supporting Office Document Architectures with Constrained Types," Proc. of ACM SIGMOD 1987, pp. 504-509, 1987.

[6] 津田，山本，平川，田中，市川，"オブジェクト指向データモデルＭＯＲＥ，" 情報究報，88-DB-(5)，1988.

[7] M. Stonebreaker, "Object Management in POSTGRES Using Procedures," Proc. of International Workshop on Object-Oriented Database Systems, pp. 66-72, 1986.

[8] 津田，山本，平川，田中，市川，"ＭＯＲＥに基づくオブジェクト指向データベースシステム，" 情報究報，88-DB-(6)，1988.