

マルチメディア文書処理システムの一実現

三島 善成、三好 力、吉川 耕平、久保 登、坂上 誠司、千葉 徹

シャープ株式会社・技術本部・コンピュータシステム研究所

従来、コンピュータシステムによって仕事の効率を改善する試みが数多くなされ、伝票処理や事務計算等の定形業務については多くの実用的なシステムにより、著しい改善が達成された。一方、オフィスに於ける個々の業務のうち40~60%は非定形的であり、多様な情報を複雑に組み合わせ、それらに関連づけて報告書作成や意志決定業務を進めている。現在、これらの業務の一部は、OA機器の上で処理されるようになってきたが、統合的なOA環境の実現には今一歩という感がある。即ち、コンピュータシステムの開発は処理効率重視の路線を進んできたこともあって、マルチメディアデータに対する統一的な操作や処理のための手段、データ構造を支援しておらず、まだまだ多くの問題点を残しているのが実状である。

本文では、未来のオフィスシステムを具体化する第一歩として、多様な情報を一元的に扱える環境を実現するため、マルチメディアデータから成る文書の表現法と、それに対する編集、通信、回覧、配布等の多様な処理を、効率よく実現する手段について述べる。

“An Editing System for Multi-Media Document” (in Japanese)

Y. Mishima, T. Miyoshi, K. Yoshikawa, N. Kubo, S. Sakaue, and T. Chiba
Computer Systems Laboratories, Engineering Center, SHARP Corporation
2613-1 Ichinomoto, Tenri, Nara, 632 Japan

A document editor for multimedia data and its user interface is described. The editor is implemented on a workstation which is equipped with a bit-map display and a local area network or PBX. This system provides a user with facilities such as an editor, database manager, mailing, etc. Document structure is based on ODA of ISO and actual data structure is enhanced for dealing with an extra media and its associating operations. Since a user interface is one of the most important factor for raising ability of office systems, then in our system, one can realize his own interface environment dynamically using the interface definition facilities. The user interface is described by a frame, and has a nature of extension facility.

1. はじめに

従来、コンピュータシステムによって仕事の効率を改善する試みが数多くなされ、伝票処理や事務計算等の定形業務については多くの実用的なシステムにより、著しい改善が達成された。一方、オフィスに於ける個々の業務のうち40~60%は非定形的であり、多様な情報を複雑に組み合わせ、それらに関連づけて報告書作成や意志決定業務を進めている。現在、これらの業務の一部は、OA機器の上で処理されるようになってきたが、コンピュータシステムによる、統合的なOA環境の実現には今一步という感がある。即ち、コンピュータシステムの開発は処理効率重視の路線を進んできたこともあって、マルチメディアデータに対する統一的な操作や処理のための手段、データ構造を支援していない。これに対する反省から、近年、オフィスを個人と情報処理装置の集合から成る分散環境と捕らえ、その中で非定形業務を統合的に支援する試みが為されているが、まだまだ多くの問題点を残しているのが実状である。

我々は、未来のオフィスシステムを具体化する第一歩として、多様な情報を一元的に扱える環境を実現するため、マルチメディアデータから成る文書の表現法と、それに対する、編集、通信、回覧、配布等の多様な処理を効率よく、実現する手段を以下に述べる。

2. システムの概要

オフィスシステムとして主流となりつつある統合処理環境を提供するシステムでは、マルチメディアデータ(文章、図形、表、イメージ)を含むドキュメントの作成、編集、保管、検索、配布などの機能を、統合化されたユーザインターフェイス(デスクトップインターフェイス)を通して利用することができなければならない[1,2]。しかし、資料を収集して分類をし、考えをまとめて、略図化しながら文書化を行っていく作業は、まだコンピュータシステムから離れたところで行われている。例えば、考えをまとめるという作業自体も、統合処理環境で行われることが望ましい。

このようなコンピュータシステムを作り上げるためには、次のようなことが考察すべき事柄となるであろう。

- a) 文章、図形、表、イメージを統一的に取り扱えるデータ表現構造。
- b) ワークステーションと利用者が対話をしながらマルチメディア文書を実時間で編集できるハードウェア。

c) 分散環境下でのマルチメディアデータ通信のためのデータ交換。

d) 使いやすいユーザインターフェイス。

ワークステーションは、コンピュータ、ビットマップディスプレイ、キーボード、ポインティングデバイス(マウスなど)等から構成される。利用者はワークステーションのビットマップディスプレイの上に構成されたデスクトップインターフェイスを通じて、キーボード、ポインティングデバイスを用いてコンピュータと対話を行う。ワークステーションはLANによって接続され、それぞれのワークステーションが対等に通信を行うことを前提とする、いわゆる水平分散接続が実現されている。通信の機能はメール・サービス、ファイル・サービスなどがある。メール・サービスは個人対個人間通信が基本となり、その機能としては配布、返信、回覧、同報などがサポートされる。ファイル・サービスの機能としては格納、検索、変更、消去などがサポートされる。文書を転送する場合には、依頼者のコメントが添付できる。

思考の道具としてコンピュータシステムを利用するためには、コンピュータと利用者とは結ぶユーザインターフェイスが重要である。ユーザインターフェイスには、

- a) 操作性が統一されていること、
- b) 利用者の好みに合わせた環境が設定できること、

の2つのことが、デスクトップ環境だけでなくアプリケーションプログラムレベルまで拡張されてサポートされる必要がある。これを実現するためにアプリケーションプログラムとユーザインターフェイスは独立して設計できる構造をしている。

利用者はデスクトップ・インターフェイスによりワークステーションと対話を行う。ここで中心となる処理はマルチメディア文書編集操作である。編集プログラムは文章、図形、表、イメージのメディア毎にモジュール化して用意される。従って新しいメディアの追加には、そのメディアに対応する編集プログラムモジュールを組み込むことにより対処できる。マルチメディア文書はODA[3,4]に準拠して、ドキュメント→ページ→エリア→プリミティブの階層構造をしており、エリア単位で各メディアは管理されている。各エリアそれぞれに独立した編集プログラムモジュールが作用することにより、編集作業が行われる。

3. マルチメディアデータの表現

3.1 図形と文字列

図形と文字列とはその持っている性質が異なるために、内部での処理を同一にすることは難し

い。即ち図形の場合には、各図形データを他の図形データや文字データと独立に書いたり消したりの操作を行うことを可能とし、また、多くの図形が複雑に重なっている中から、任意の図形を正しく選択できる事が望ましい。内部では幾つかの領域やプリミティブに分けて管理をしており、ユーザの行う操作と内部データに対する操作との間にギャップがあり、これをどう上手く埋めるかが問題である。

文字列の場合には、一文字に対する操作が全ての文字列に影響する上、図形を文字列中の任意の場所に挿入するために、文字座標をどの様に管理するかが問題となる。文字列は、常に文書として正しく整形されてユーザに見えている事が望ましい。これに関して、我々は以下のような方法を提案する。

3.2 データ構造

異なる性質のデータを出来るだけ統一的に扱うために、図1に示すようにそれぞれのデータをヘッダ部とデータ部に分け、ヘッダ部を固定長として、データの識別を行うのに必要な情報を格納し、データ部は可変長として、各々のデータに依存した情報を格納する。これによって拡張性を高く保ち、かつ、統一的に扱うことが可能となった。

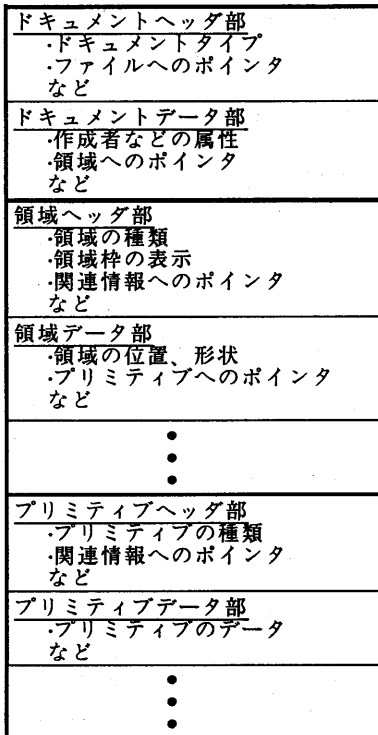


図1 文書データ構造

3.3 データ管理法

文書および文書内のデータは、図2に示されるごとくカテゴリ、ドキュメント、ページ、図形プリミティブ、テンプレート、グループ、文字プリミティブ、行、段落に分かれて管理されている。カテゴリは、作業の色々な側面からの分類や件策が行いやすいように、自由に関係付けする事ができ、ネットワークの様になっている。その下に複数のドキュメントがある。ドキュメントもカテゴリと同じ理由から、一つのドキュメントが複数のカテゴリに属することができる。ドキュメントの下には複数のページが存在し、ページの中では、文字列に関する物として、段落、その下に複数の行、行の中に複数の文字プリミティブがある。文字プリミティブは、今のところ英字、数字それに英記号であるが、漢字についてもサポートの予定である。図形データに関しては、テンプレートとグループは並列で、再帰的にも利用できる。各々の下に複数の図形プリミティブがある。テンプレートは利用者が再利用を目的として、図形プリミティブを使って定義した図形であり、グループは一時的な移動等のために利用するもので、テンプレート化、グループ化と各々の解除には異なる手続きを必要とする。図形プリミティブは基本的な図形(直線、長方形、円、楕円など)を選び、10個用意されている。

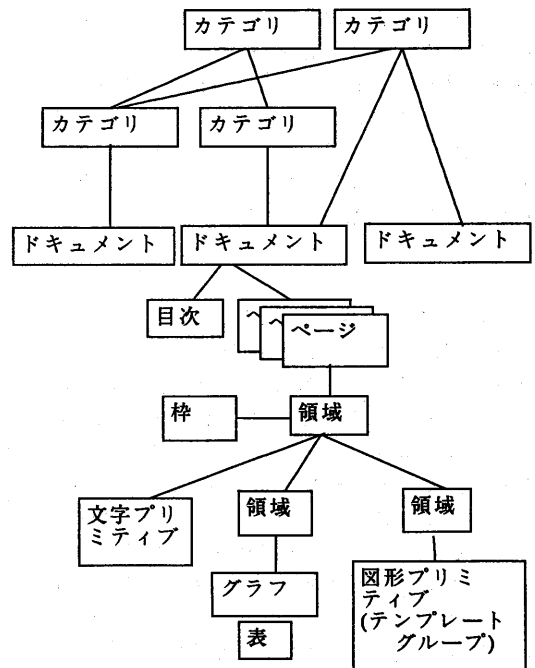


図2. 文書データ管理

4. 編集操作

4.1 編集操作の問題点

図形と文字とが混在した文書の編集を行う場合、多くの既存のシステムでは、図形領域、文字領域の概念があり[5]、現在どの領域のデータ編集を行っているのかを利用者自身がいつも把握していなければならない。さらに、文字や図形の領域を変更する等の操作が必要で、コマンド体系も文字用コマンド、図形用コマンド、或いは領域自身に対するコマンド等、大変複雑で、システムによっては何段階ものモード設定のステップを踏まないと目的のコマンドを選択できないものもある。また、領域の区別があるため、何かの操作でその枠からはみ出した場合、システムが一方向的にキャンセルし、その操作が無駄になることも多い。このように、文書作成のために直接必要でないような操作でわされることが非常に多く、普段机に向かい紙の上に文書を作成する場合とは大きな隔たりがあり、それが文書作成の妨げとなっている。

4.2 モードレス編集

以上の諸点を改善するために、編集操作をモードレスで行うことを考える。ここで、モードレスとは、対象とするデータの種別毎にある編集操作のための条件設定(モード設定)をデータの種別によらずできるだけ統一的にし、利用者から見ると、あたかも条件設定(モード設定)が無く、図形、文字、領域等、対象の種類によらず同じ操作で編集を行えるようにする事である。この、モードレスの概念を取り入れる事により、これから何を編集するのかを気にせずに、図形も文字も同じような操作で編集ができ、文書作成に専念できる。

4.3 枠

文書内のある場所に図形を入れたい場合、そのための場所とその大きさ(随時、その場所と大きさは変更可能である)を設定するために枠を挿入する事ができる。ここで枠とは、その外にある文字がその領域に重複して置かれることを避けるための物であり、文字領域と図形領域を指定するための物ではない。これにより、あらかじめ、文書内に、どこに図を入れたいか等の構造だけを決めてから文章を入力することが可能になり、例えば定形書類に関してはテンプレートを用意しておくこともできる。更に、挿入した枠内でも枠の外と同様の操作を行う事ができる(図3参照)。

4.4 複数文書

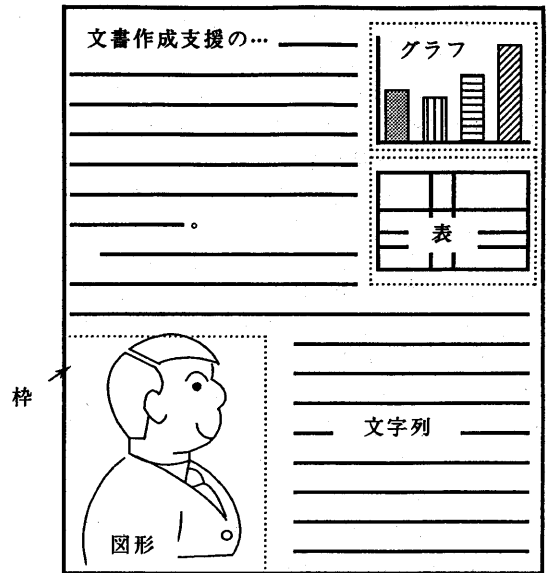


図3 文書の構成例

次に、同時に複数の文書を開いて編集を行う場合について考える。同一文書内での文章の切り貼りは多くのシステムで実現されている。また、異なる文書間の切り貼りも一部のシステムで実現されているが、中間ファイルを使う等、操作的に複雑な物もある。

同一文書内でも異なる文書間でも利用者側からは同じ操作で切り貼りが出来ることが必要であり、特に異なる文書間の場合は、ウィンドウの重なり具合を気にせずに、たとえ対象のウィンドウが他のウィンドウと重なっていても、支障なく操作が出来るようにするべきである。そこで、必要なウィンドウそれぞれにカーソルを設け、マウス等によって選ばれているウィンドウのカーソルをアクティブにする。これによって、ウィンドウをまたがって切り貼り操作を行う場合、ウィンドウの重なりを気にしたり、カーソルを正しい位置に合わせる等の面倒な操作が省略できる。

5. ユーザインタフェイス

現在実現されている多くのユーザインタフェイスについて考察すると、操作の容易性、直感的な分かりやすさ等から、マウス、ビットマップディスプレイを使い、マルチウィンドウ、ポップアップメニュー、アイコン等でユーザと対話する、という形が一般化している[6,7]。以下では、ユーザインタフェイスを使いやすく、また作りやすくするための汎用ユーザインタフェイス構築プログラムについて述べる。

いままでのユーザインターフェイス構築プログラムの問題点として、

- a) ユーザの好みや、使用される様々の目的、環境によってユーザインターフェイスを変更することが難しい、
 - b) アプリケーションプログラムとの役割分担が統一的でない、
 - c) プログラムの記述が容易でない、
 - d) 対話管理の機能が不十分、
- などが挙げられる。

これらの問題を解決するためには、まずインターフェイス構築プログラムをモジュール化し、構築の容易性、アプリケーションプログラムからの独立性を実現すること、次にユーザカスタマイズの枠組みを提供するための高度な対話管理の手法が必要である。本システムでは、図4に示すようなフレーム型データ構造とそのインタプリティブな実行環境を提供することにより、使い易く構築のし易いユーザインターフェイス構築プログラムを実現している。

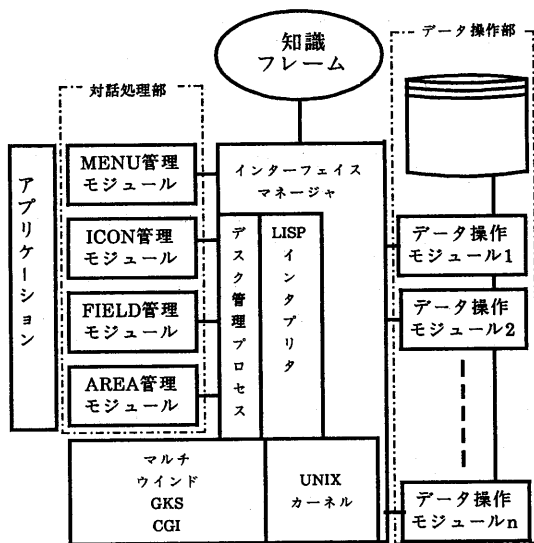


図4.ユーザインターフェイスとシステムの構成

5.1 フレーム型データとそのインタプリタ環境

インターフェイス構築プログラムに必要なとされることは、容易に、宣言的に仕様記述できることである。ユーザとマシンとの対話に使うウインドウやメニューをひとつひとつの部品(プリミティブ)に分解しそれらを組み合わせることによって全体のイ

ンターフェイスを構築するという接近法を取る。ここでプリミティブとはメニュー、アイコン、グラフィックスエリア等と、それらを組み合わせて1つの動きをするものを言う。ユーザインターフェイス構築とは必要なプリミティブを創成し、それらプリミティブにアプリケーションプログラムを結び付け、プリミティブを合成して最終的にひとつのインターフェイスを完成させることである。これは其々のプリミティブをフレーム形式で表現し、それに階層構造を持たせることにあたる。さらに、ダイナミックにユーザインターフェイスを変更するための枠組み、即ち対話管理機能が必要である。

以上のような機能を実現するフレーム型データ構造が一般に持つ特徴として

- 1) 表現が簡潔、
 - 2) インヘリタンスによるデフォルト機能の有効性、
 - 3) 記述のモジュール性が高く、管理変更が容易である、
- などが挙げられる。

またインタプリタ環境に持たせる機能として

- 1) ダイナミックに記述できる、
 - 2) 自分自身を拡張してゆける
- などが挙げられる。

5.2 プリミティブ

使用するプリミティブは、図5に示すように、(1)FIELD、(2)ICON、(3)MENU、(4)AREA、である。

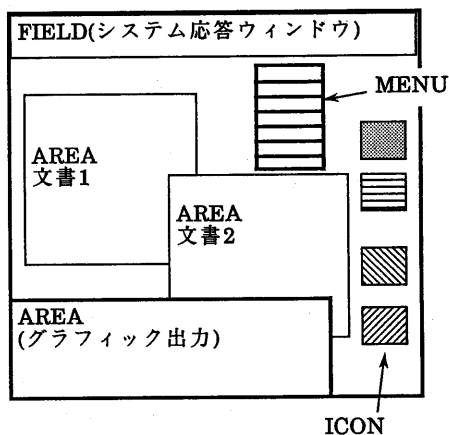


図5 ユーザインターフェイスの概念図

ここでFIELDは、ユーザとの間で文字列入出力を行うためのもので、入出力マスク、出力文字列、入力データの型、入力データの候補、ヘルプメッセージなどの属性を有している。ICONには、その形状を記述する表示データファイルとそれ自身の名

前、起動するアプリケーション、ヘルプメッセージ等の属性、MENUにはそれ自身の名前、表示文字列とアプリケーションへ送るデータのペア、ヘルプメッセージなどの属性を持たせている。最後にAREAはアプリケーションが使用するための領域で、縦横のサイズ、入出力マスク等の属性がある。

5.3 プリミティブの構築法

上記のプリミティブを合成するための手段として、(1)ATTACH、(2)PUT_ON、が用意されている。ATTACHはプリミティブを上下左右にくっつけて新たなプリミティブを作る機能であるが、サイズの異なるプリミティブ同志でも適当なサイズにあわせることが出来る。PUT_ONはプリミティブの上の任意の位置に別のプリミティブをオーバーラップさせる機能を持つ。

以上のようにして合成されたプリミティブの使用法として、(イ)POP_UP、(ロ)WINDOW、の2つが用意されている。POP_UPはプリミティブをポップアップさせる形での使用を宣言するもので、構成するプリミティブと、それらがピックアップされた時にアプリケーションにおくるデータのペア等の属性を持つ。WINDOWとは合成されたプリミティブをひとつのウィンドウとして完成させるもので、ウィンドウ名、ウィンドウを閉じた時のアイコン表示データへのポインタ、ウィンドウのデフォルトサイズ等の属性を持つ。

このようにインターフェイスの構築を宣言的に、また予め用意されたプリミティブを活用して階層的に組み上げていくという方法は、プログラマにとって分かりやすく、また自然な方法である。

5.4 ユーザカスタマイズ

次に、ユーザの立場でのよりよいインターフェイスの実現のために、以下のようなカスタマイズ機能がある。

- スクリーン上でのウィンドウ、アイコン、メニュー等の並びを、目的や好みに合わせて変更する機能、
 - メニューに表示されるコマンドの並びの変更、及び不必要なコマンドを不可視にする機能、
 - 自分向けの作業手順を作成し、一定の手順ののったオペレーションを定義する機能、
 - 自分だけのコメントを好みの位置にはめる機能
- 以上の機能については現在まだ実現されていないが、フレーム構造の表現の柔軟性と、環境がそれをダイナミックにインタプリットする機能により容易に実現できるものと考えている。

6. 本システムの分散環境に於ける実現

オフィスでの業務全体は、情報処理装置を分散配置し、個別の業務を、個々のデスクで独立して処理できるものとする。その結果として必然的に、作業を調整し業務をスムーズに進めるため、分割したこれら個別の業務間のコミュニケーションが、重要なものとなる。現在、高速にコミュニケーションを行うため、LAN及びPBX-LANによるアプローチを進めている。LANはオフィス内を主な対象とし、PBX-LANではビル全体のより広い範囲を対象としている。LANによる分散環境の構築で問題となる点は、主にLAN上に実現されたプロトコルであり、特に異機種種の装置の接続に関しては困難な要因が多い。当面は、下位層にはTCP/IPをベースとしたプロトコルを採用し、上位層は今後の標準化動向を含めて行く。

7. おわりに

本文では、今後のオフィスで必要とされるコンピュータ支援の統合化された作業環境の内、特に、それらの基礎として位置付けられるマルチメディアデータを有する文書処理に関して報告を行った。

現在、本システムのプロトタイプ開発を行っているが、将来に向けてこれを拡張していくためには、

- 文書構造、文書交換に関して、ISOやCCITTから提案されている規約に基づくこと、
 - マルチメディアデータの編集能力の拡張、例えばメディア間データ変換など、
 - ユーザインターフェイスの高機能化、多機能化を背景とした知的なユーザインターフェイスとインターフェイスのカスタム化技術
 - 分散環境の具体化とその将来像
- 等についての研究が必要であり、今後検討していく予定である。

謝辞

本研究を行うにあたりご指導頂いたコンピュータシステム研究所西岡都夫所長、河田亨副所長ならびに、ご討論いただいたコンピュータシステム研究所第1研究室の諸氏に感謝します。

参考文献

- 1)N. Monden, Y. Yoshino, M. Hirakawa, M. Tanaka, and T. Ichikawa: "A Language Supporting Visual Interaction in Programming," Proc. IEEE Workshop on Languages for Automation, pp.199-205, Nov.1984.
- 2)Y. Yoshino, H. Yoshimura, N. Nishi, M. Tanaka, and T. Ichikawa: "A Hierarchical Model for

Multiple Window Processing," Proc. IEEE Workshop on Languages for Automation, pp.26-31, Nov.1984.

3)規格委員会SC18専門委員会:“文書交換に関する国際標準化動向”情報処理学会 Vol. 26, No. 1, pp.33-41, Jan. 1985.

4)ISO DP 8613/1/2/3/4: “Office Document Architecture and Office Document Interchange Formats.”

5)J.Gutknecht: “Concept of the Text Editor LARA,” Communications of the ACM, Vol. 28, No. 9, pp.942-960, Sep. 1985.

6)杉山広幸,今宮淳美:“新しいユーザ・インターフェイスの設計方法論の提案 — ユーザ・インターフェイス・モデル,”情報処理グラフィックスとCAD, 1984年3月.

7)Mark Green: “The University of Alberta User Interface Management System,” ACM SIGGRAPH, pp.205-213, Jul.,1985