グラフィクスとＣＡＤ　25−2
（１９８７・２・２６）

ユーザインタフェース管理システムの方式とその試作

守屋慎次，斉藤剛，中谷吉久，倉澤美恵，菅居徹
東京電機大学工学部電気通信工学科

本論文では、ユーザインタフェース管理システム（ＵＩＭＳ）の情報構造モデルとその基本的な考え方を示し、そのモデルに基づいて試作中のテキストエディタの構成方式について述べる。本論文の方式によって実現される対話型システムの、利用者に対する特長は次の通りである。①より利用者主導になる。②より広範な定義機能を通じてシステムを利用者に適応可能である。　また、本方式によるＵＩＭＳには応用プログラムと入出力装置の両者から独立な部分が存在する。これにより、①本方式に基づいて書かれる応用プログラムは入出力装置と技法から独立となり、②本方式に基づいて開発される入出力装置やモジールは応用プログラムから独立となる。さらに、③ＵＩＭＳの透明度が高まる、従って、④使い易くなることが期待できる、などの特長がある。

An Information Structure Model for User Interface Management System and it's Implementation

Shinji MORIYA, Tsuyoshi SAITOH, Yoshihisa NAKAYA,
Mie KURASAWA, Toru SUGAI

Tokyo Denki University, Department of Communication Engineering
2-2 Kanda-Nishiki-cho,Chiyoda-ku,Tokyo,101 Japan

In this paper, firstly, we show several basic principles in constructing an information structure model for user interface management system(UIMS). Secondly, based on the model, we descrive the method to implement a UIMS which is bound to a text editor.

An interactive system developed by the method has following properties: (1) the system becomes more user_centered, because the UIMS has the initiative in controling the system, and (2) the system becomes more adaptable to the user, because the UIMS can provide to the user a wider range of oppotunities in specifying the input_output structure of the system.

The UIMS includes an independent part from not only input_output devices but also application programs. Therefore, (1) the application program developed by the method is independent from input_output devices and techniques, (2) the input_output device and technique developed by the method is independent from application programs, and (3) the UIMS can be more transparent than ever presented.

1．はじめに

本論文の目的は、①ユーザインタフェース管理システム（以降ＵＩＭＳと略記する）の情報構造モデルとその基本的な考え方を示すこと、②そのモデルに基づいて試作中の対話型システムについて述べることである。

この様な研究には次のような意義が認められよう。①対話型システムの利用者や用途は多種多様であるが、システムの仕様は画一的である。このため利用者がそのシステムに適応しきれない場面が数多くみられる。したがって適応力のあるユーザインタフェース（以降ＵＩと略記する）の実現が強く求められている。②対話型システムにおける入出力部は開発コストの主要因の一つである。従って応用プログラムからの独立度が高い、すなわち応用プログラムへの適応力のあるＵＩＭＳの開発が望まれている。

本論文が目的とするＵＩＭＳには次のような利点があると考えられる。①利用者は自分向きのインタフェースを組立て易くなる、②入出力装置からの、応用プログラムの独立度が増す、③ＵＩにhuman factorを組込み易くなる、④応用プログラムの開発コストが低減する[7]、などである。

研究目的を達成する上で主として次の点を明らかにする必要がある。①応用プログラムとＵＩＭＳのインタフェース。②ＵＩＭＳの構造（又はモデル）。③適応可能にすべき、ＵＩの要素・機能とその方法。

本論文ではまず上記の問題①②③について基本的な考え方を明らかにし、筆者らが構築したＵＩＭＳの情報構造モデル[3]の全体的な性質を述べる。次いで、そのモデルに基づいて試作中のテキストエディタと、実現中のＵＩＭＳについて説明する。最後に本論文における方式の特長を述べる。

2．背景

ＵＩＭＳと適応性の研究は現在、いずれも立上りの初期段階にある[1,2,4,8]。両分野とも、主として日本、北米、欧州を中心にこの１，２年急速に多くの関心を集めている[4,8]。ＵＩＭＳはすでに１１個のシステムが開発されたと報告されている[4]（いずれも欧米にて）。そのうちのいくつかは市販されている。このような状況で筆者らが現状を「立上りの初期段階にある」とする理由は次の通りである。（１）ＵＩＭＳのモデルの検討が十分でないこと。（２）適応性に関する基礎的研究が不十分なこと。（３）ＵＩＭＳと応用プログラムのインタフェースがad hocであること。（４）次のような概念とＵＩＭＳとの関係が未開拓であること。①ユーザモデル、②ＵＩの品質を決定する要因と機構、③user friendliness、④応用プログラムの統合方式。（５）次のようなシステムとの関係が不明確であること。①オペレーティングシステム、②ウィンドウ管理システム[9]、③図形処理国際規格[10]。以上のように、ＵＩＭＳ実現に当っての本質的問題の多くがいまだ殆ど未解決な状態にあるからである。

本論文はこれらの諸問題に完全な解を与えるものではない。しかしながら、解決に向けての一つの方向は示していると考える。その理由は以下の通りである。（１）ＵＩＭＳのモデルに関しては、筆者らが開発した情報構造モデル[3,5,6]が、今までのもの[4,8]に比較して最も先進的かつ精密なものと考えている。ＵＩＭＳは人間・機械系の機械側の中核に位置するものであり、そのモデルの重要性については言を待たない。（２）筆者らの情報構造モデルは、二つの実用的な応用プログラムの実データに基づいて構築された。その結果、ＵＩＭＳの構造や用途はより一般性に富み、同時に応用プログラムとのインタフェースもより汎用性を増しているものと考えている。

3．ＵＩＭＳのモデル構築における基本的な考え方

ＵＩＭＳを実現する際の前提条件は、応用プログラムから入出力部を分離することである。ここでは、分離することの意義・価値と分離する際の基本方針について述べる。

3．1　応用プログラムから入出力部を分離することの意義

標題の「分離の意義」については文献1,2,3で少し触れ、文献4で比較的詳しく述べた。それらに加えて次に述べるような「分離の意義」が存在する。

(1)利用者からの要求

通常、入出力部[3,4]の仕様は応用プログラム設計者によって決定され、多くの場合固定されている。ところが現実には、仕様が利用者や仕事に整合していない場合が見受けられ、利用者がそれを変更したい場合が多々ある。そこで、①従来は応用プログラム中に埋め込まれていた入出力の各種要素・機能を応用プログラムから分離してＵＩＭ

Sへ所属させ、②それら要素・機能群と応用プログラムとを利用者がＵＩＭＳを用いて編成可能にする、という方法が考えられる。この編成することを筆者らは適応と呼んでいる。

(2)応用プログラム作成者からの要求

対話型システムにおける入出力部は開発コストの主要因の一つである[3,4]。理由の１つとして、入出力機能に対する要求仕様とその記述ツール間のセマンティックギャップの大きさが考えられる[4]。従って入出力部を対話型システムから分離し、その構造を分析してモジュール化することにより、有用な共用ツールを作成できる可能性が非常に大きい。

（尚、分離による価値については文献1,2,4,7を参照されたい。）

３．２　応用プログラムから入出力部を分離する際の基本方針

標題の「基本方針」、換言すれば、応用プログラムとＵＩＭＳのインタフェースはＵＩＭＳ研究の最大の問題の１つである。基本方針のうちの２つを次に述べる。

(1)利用者主導 ── 制御の主権をより利用者側に近づける。

通常の対話型システムでは、利用者は応用プログラムからの入力要求に応じて対話を進行する。制御の主権は対話型システムが持っており、この意味で機械主導の方式であるといえる。

筆者らは、ＵＩＭＳに制御の主権を与えることにしている。基本的な制御の流れにおいては、ＵＩＭＳの要求に応じて応用プログラムが起動され、ＵＩＭＳの要求に応じて利用者が対話を進める。データ入力など、応用プログラムが要求を出すことは可能である。しかし、制御の流れはＵＩＭＳがその主権を持つ。ところが３．１節(1)の①②で述べたように、利用者はＵＩＭＳを用いて入出力の各種要素・機能を自分に適応させることが可能である。すなわちＵＩＭＳが持つ制御の主権を、適応能力の及ぶ範囲内において利用者が支配することができる。この意味で、この方式はより利用者主導の方式（外部制御方式[11]）といえる。

この方式はＵＩＭＳが必要に応じて応用プログラム中の単位機能を呼び出すことを意味している。すなわちこの方式を実現するためには応用プログラムをモジュール群として構成するのが自然である。

(2)機能の整合 ── 対話型システムの機能を利用者に整合させる。

ある種の仕事（例えば文書の入力・編集・出力）を対話型で進める場合について考えてみよう。考えられる全ての仕事について詳細に分析し、どんな仕事も基本となる単位仕事の列として記述できたと仮定する。（筆者らは、この仮定は成立すると考える。）このとき、①必然的に、その種の仕事を対話型で進める際のモデル（データ構造とファイル）操作は、単位仕事に含められることになる。（なぜならば、単位仕事に含められていないモデル操作が存在すれば単位仕事の列として記述できない仕事が存在し、上の仮定に矛盾するからである。）このようなモデル操作を行う単位仕事をここではsemanticsと呼ぶことにする。さらに、②入力と出力を行う操作も単位仕事に含められることになる。ここで②の単位仕事はＵＩＭＳが行うべき仕事に属す。

さて、対話型システムの機能を利用者に整合させるためには、利用者が要求する仕事を単位仕事の列によって記述する機能（組立て機能）が必要となる。この組立て機能は、３．１節(1)の①②の方針から、ＵＩＭＳによる適応能力の一部に加えられることになる。

以上の考察から、分離後の応用プログラムは上記①の単位仕事、すなわちsemanticsの処理を分担する、と考えるのが自然である。

この方針と、３．２節(1)の方針から導かれた結論（応用プログラムはモジュール群から成る）とを統合すると次のように言える。すなわち、応用プログラム内の各モジュールは各単位仕事を実行する。

４．ＵＩＭＳの情報構造モデル

以上に述べた基本方針に基づいて筆者らはＵＩＭＳの情報構造モデルを開発した[3]。それを図1に示す。図1は文献3の図に一部加筆したものである。ここでは文献3には述べられていない、このモデルの全体的な性質を述べる。

図1の箱は機能単位を、矢は情報の流れを、楕円は定義情報や状態を表す。①②間は入出力装置と装置ドライバ（以降本章内ではIOと略記）から成る。①⑤間はユーザインタフェース（より精密な定義は文献４参照）、②～⑤はＵＩＭＳ、⑤⑥間はモジュール群から成る応用プログラム（以降本章内ではAPと略記）である。ＵＩＭＳ内の機能単

位は文献3に説明されている。

ＵＩＭＳのうち②③間だけがIO依存部である。IN_{exec}はAP依存である。本論文では述べていないが$IN_{compile}$はAP独立部と依存部に分割可能である。従って入力側には、③から$IN_{compile}$の中間までに、IOとAPの両者から独立な部分が存在する。本論文では述べていないがOUT_{trans}もAP独立部と依存部に分割可能である。従って出力側にも、③からOUT_{trans}の中間までに、IOとAPの両者から独立な部分が存在する。③を論理入出力言語と呼ぶ所謂である。

④における情報の入力流の「受信者」はAP、出力流の「受信者」はＵＩＭＳ自身と利用者である（IOではない）。④は各受信者と整合した言語水準になっている。

ＵＩＭＳのモデル中にAPとIOの両者から独立な部分が存在することには次のような重要な利点がある。

(1)IO依存部はAPから独立に、AP依存部はIOから独立に、それぞれ作成・変更・定義等が可能である。

(2)このモデルに準拠して作られる応用プログラムはIO独立となる。IO独立とは入出力装置と入出力技法の両者から独立なことを指す。このモデルに準拠して作られるIO装置とモジュールは応用プログラムから独立となる。

(3) 作成されるＵＩＭＳの透明度（ transparency)[1]が増す。従ってＵＩＭＳをより利用し易く設計できることが期待できる。

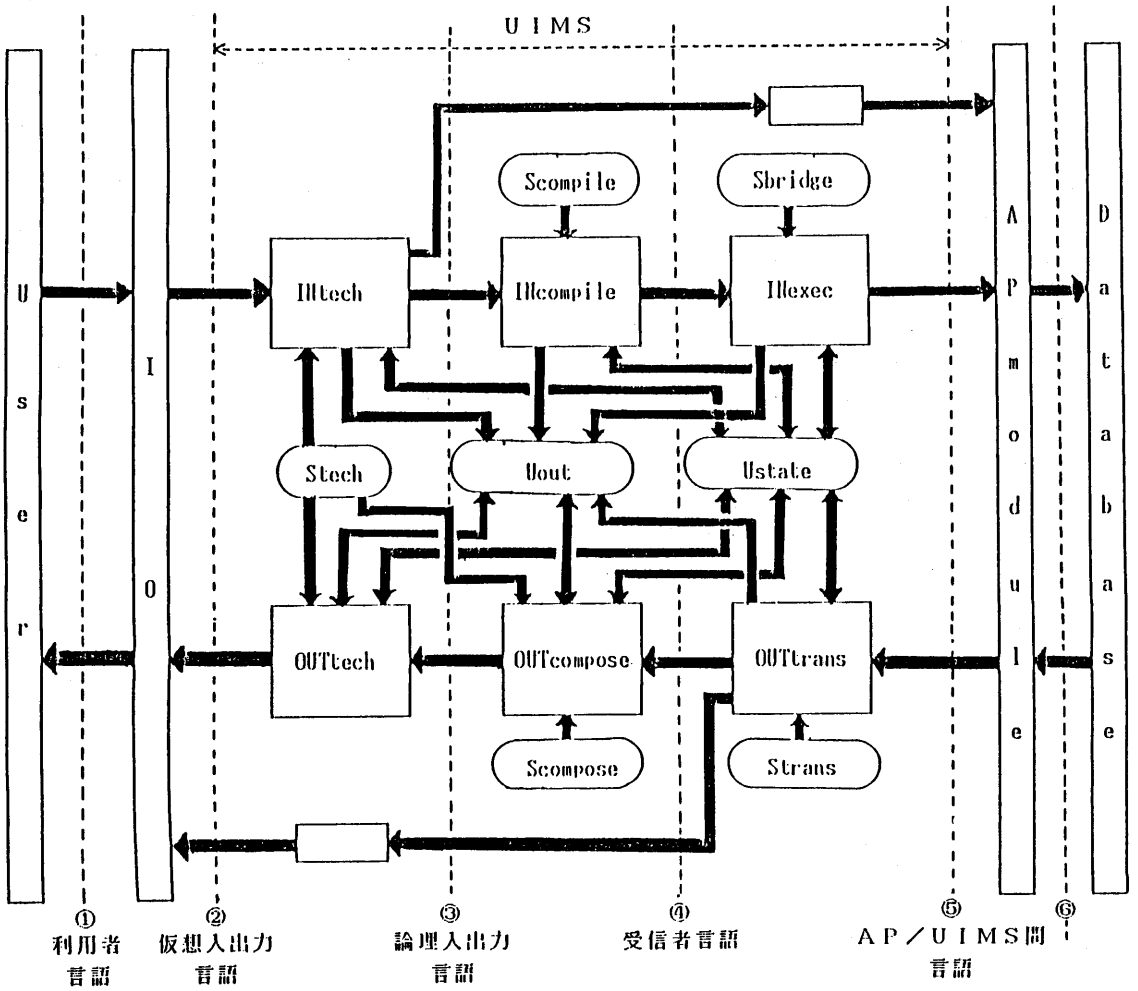

図１．ユーザインタフェース管理システムの情報構造モデル

５．ＵＩＭＳの情報構造モデルに基づいた テキストエディタの実現法

筆者らは、図1の情報構造モデルに基づいたＵＩＭＳ、および、それと接続されるテキストエディタを試作中である。本章ではＵＩＭＳモデルに基づくエディタの実現方法について述べる。

開発の目的は、実現し実用して評価することである。評価項目として、①ＵＩＭＳと応用プログラムのインタフェース、②ＵＩＭＳモデル、③適応可能にすべき要素・機能とその方法、④使い心地などがある。

テキストエディタを選択した理由は以下の通りである。すなわち、対話型システムの中ではエディタの利用者が圧倒的に多く、ユーザインタフェースの高級化に対する要求が広範に末長く続くことが明らかだからである。

以下に、設計したエディタの概要を述べ、ＵＩＭＳ全体の構造を示す。

５．１　エディタの機能概要

設計したエディタは、スクリーン英文テキストエディタである。編集操作における特長として「色による操作対象の区別機能」およびリカバリ機能などがある。これらに加えて、次のような利用者への適応機能が用意されている。

（１）利用者コマンドに関する定義機能。

（２）画面の表示属性・レイアウトに関する定義・変更機能。

適応機能については６．３節で詳しく述べる。

５．２　エディタからの入出力部の分離

対話型システムから入出力部を分離する意義・価値、および、その方針を第３章で述べた。その方針に基づいて本エディタを、そのsemanticsを処理する部分（モデル操作部）と入出力を処理する部分（入出力部）に分離した。すなわち、図２のようにエディタを二つの部分に分割した。各部分が果すべき機能を以下に示す。

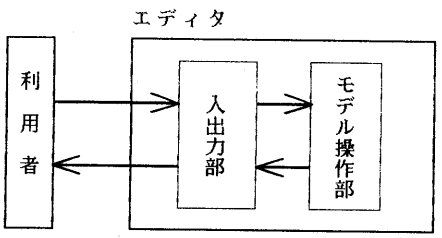

図２．テキストエディタは２つの部分に分離される

（１）　モデル操作部が扱う機能

操作対象となるテキストの管理、テキストの挿入・削除・複写及びそれらの試行・回復、パタンおよびフラグ検索、属性とその意味の処理、ファイル・バッファの管理、など。

（２）入出力部が扱う機能

コマンドの入力および翻訳、画面の編成・表示およびスクロール、カーソルの移動、など。

５．３　エディタにおける単位仕事

５．２節（１）（２）で示した各機能を綿密に分析し、基本的な操作の集まり、すなわち、エディタにおける単位仕事の集まりを設計した。ここでモデル操作部における各機能を実現するための単位仕事を「モデル操作用の単位仕事」、入出力部における各機能を実現するための単位仕事を「入出力操作用の単位仕事」と呼ぶことにする。モデル操作用の単位仕事の例を表１に、入出力操作用の単位仕事の例を表２に示した。表２の単位仕事は表層入出力と深層入出力の二つに分類されている。深層入出力は表示されるデータの論理的性質に係わる入出力操作を、表層入出力は係わらない入出力操作を行う。

５．４　ＵＩＭＳ付きエディタの構造概要

筆者らが設計したエディタは、図２の入出力部における機能を図１中のＵＩＭＳ部分で代行させ、また、図２のモデル操作部を図１中のAP moduleとして構成するものである。

（１）AP module部の構造　　３．２節で述べたように本エディタのAP moduleは、表１に示したモデル操作用の単位仕事と１対１対応している。

（２）ＵＩＭＳ部による入出力機能の代行　　図１のＵＩＭＳ部は、表２で示した入出力操作用の単位仕事を実行する。

（３）ＵＩＭＳによる流れの制御　３．２節で述べたように、利用者との対話および単位仕事の実行の流れはＵＩＭＳが制御する。

６．ＵＩＭＳによるエディタの入出力機能の代行法

本章では、図１のＵＩＭＳにおける各機能単位の働きや定義情報の内容を、試作中のエディタを実例として詳細に述べていく。

6．1　ＵＩＭＳの構造と機能分担

試作中のＵＩＭＳは、図１の情報構造モデルに基づいて構成されている。図１には６つの機能単位（IN_{tech},$IN_{compile}$,IN_{exec},OUT_{trans},$OUT_{compose}$,OUT_{tech}）と５つの定義情報（S_{tech},$S_{compile}$,S_{bridge},S_{trnas},$S_{compose}$）がある。これら６つが何を行い、５つに何が定義されているかを以下に述べる。

（１）IN_{tech},S_{tech}：本エディタでは３つの入出力技法を用いる。これらの何れもが、利用者コマンド入力に関する技法である。第一は、キーボード上の各キーに利用者コマンドを割り当て、あるキーが押されるとそのキーに対応するコマンドに置き換える技法である。

編集処理	モデル操作用の単位仕事
操作対象の管理	set_charater
	set_beginning_point
	set_end_point
	reset_beginning_point
	reset_end_point
テキストの挿入	insert_string
テキストの削除	delete_region
回復	recover_around
確定	decide_around
データ要求	get_paragraph
	total_paragraph
モデル管理	bof
	eof
	bop
	eop
パタン検索	point_search_forward
	point_search_backward
構造検索	point_bop
	point_eop
	point_paragraph
フラグ検索	point_flag
フラグ管理	set_flag
	reset_flag
属性処理	set_color
	change_color
	draw_underline
	erase_underline
バッファ管理	copy_to_buffer
	yank_from_buffer
ファイル管理	read_file
	save_file
	quit_file
	switch_to_file
初期化	init_editor
終了	exit_editor

表１．テキストエディタのモデル操作を行う単位仕事（基本コマンド）

第二は、文字列の入力のための技法である。これは、コマンドを文字列として入力する場合、および、例えばファイル名や検索する文字パタンなどを直接文字列として入力する場合、などに利用される。

残る一つは、テキストの入力・文字の挿入に利用されるものである。この技法は、特定のキー以外は「エディタの文字挿入コマンド＋入力された文字」に展開する。これらの技法はそれぞれ論理入力装置としてIN_{tech},S_{tech}内でモデル化されて

表層入出力操作	
入出力処理	入出力操作用の単位仕事
カーソル制御	set_cursor
	point_cursor_up
	point_cursor_right
	point_cursor_down
	point_cursor_left
	cursor_more_chars_forward
	cursor_more_chars_backward
	cursor_more_lines_upper
	cursor_more_lines_lower
	cursor_in_window
ウィンドウ制御	window_more_lines_upper
	window_more_lines_lower
	full_window
スクロール制御	scroll_up_one_line
	scroll_down_one_line
表示指令	refresh
深層入出力操作	
入出力処理	入出力操作用の単位仕事
カーソル処理	clear_cursor
	cursor_append_paragraph
	cursor_prepend_paragraph
ウィンドウ処理	clear_window
	window_append_paragraph
	window_prepend_paragraph
	write_beginning_point_mark
	write_end_point_mark
	write_insert_mark
	write_current_para_number
	write_total_para_number
	write_file_name
	erase_beginning_point_mark
	erase_end_point_mark
	erase_insert_mark
属性処理	change_color_mode
初期化	init_UIM
終了	end_UIM

表２．テキストエディタの入出力操作を行う単位仕事（基本コマンド）

いる。

（２）$IN_{compile}$,$S_{compile}$：ここでは、利用者が入力したコマンドを解析し、表１，２で示した単位仕事の列に翻訳する。この解析のための「コマンドの構文」、および、翻訳に必要な「変換規則」が$S_{compile}$に定義される。この翻訳された単位仕事の列は、IN_{exec}で解釈実行される。そこで、$IN_{compile}$から出力される単位仕事を「基本コマンド」と呼ぶことにする。モデル操作用の単位仕事をモデル操作基本コマンド、入出力用の単位仕事に対応するものを入出力基本コマンドと呼ぶ。表１および表２に示した名前が、各々の基本コマンド名である。

（３）IN_{exec},S_{bridge}：生成された基本コマンド列の実行を制御する部分である。

モデル操作基本コマンドは、S_{bridge}内の定義に従う仕様でAP moduleに送られる。このコマンドで指定されたAP module内のモジュールが実行される。実行の結果、出力が生じればそれはIN_{exec}に戻されOUT_{trans}へ送られる（ＵＩＭＳ－AP module間通信については６．２節で述べる）。

また、入出力基本コマンドは、その殆どがOUT_{trans}へ転送される。残り（例えばＵＩＭＳ内モードをU_{state}に設定するコマンド）はここで直接実行される。

（４）OUT_{trans},S_{trans}：ここでは、次の２つの処理がなされる。第一は、AP moduleから出力される情報の形式変換である。この変換には二種類あり、一つは整数－文字列変換、他の一つは「パラグラフ」の表現形式の変換（表示属性の表現法変換）である。

第二は、$OUT_{compose}$およびOUT_{tech}への指令の生成である。IN_{exec}からの入出力基本コマンドおよびAP moduleからの出力が、表示に関する指令の列に分解され生成される。

S_{trans}は、これらの変換規則および生成規則の集合である。

（５）$OUT_{compose}$,$S_{compose}$：本エディタにおける表示情報は、次の４つに分けられる。

（ａ）エディタの内部状態を表す情報。例えばフラグおよびマークの設定状況、ファイル名、パラグラフ数、操作対象の区別機能に割り当てられている色、など。

（ｂ）対象とするテキスト。

（ｃ）ＵＩＭＳからのメッセージ。

（ｄ）コマンド入力のプロンプトおよびエコー。

これらの各々を出力する論理的な装置（論理出力装置）がS_{tech}内にモデル化されている。$OUT_{compose}$の主な仕事は、OUT_{trans}から送られてくる指令を論理出力装置に対応付けることである。

（６）OUT_{tech},S_{tech}：論理出力装置は、対応付けられた指令を入力し、表示画面の構成要素を生成する。次いで、この構成要素を各々ウインドゥに対応付け、表示画面を構成する。画面の構成法はS_{tech}に定義される。また、ここで、カーソル制御およびスクロールが行われる。

６．２　ＵＩＭＳとAP moduleのインタフェース

ＵＩＭＳからAP moduleへ渡されるデータは、モデル操作基本コマンドの名前とその引数とから成る。６．１節（３）で述べたように、このコマンドによりAP module内の対応モジュールが実行される。実行されたモジュールからＵＩＭＳへの出力は、入出力基本コマンドの名前と該当モジュールからの出力情報とから成る。その内容は、エディタが操作対象としているパラグラフ、ファイル中におけるパラグラフの位置、カーソル設定位置等の情報である。

６．３　適応機能

ここで、本ＵＩＭＳの適応機能についてまとめる。

本ＵＩＭＳの有する適応機能には、次の２種類がある。

（１）エディタ利用者への適応機能

６．１節（１）で述べたキーとコマンドの対応付けは、利用者がS_{tech}内に定義可能であり、利用者の好みや使用しているエディタに合わせることができる。また、６．１節（６）に示したように、利用者が直接見る画面の構成（表示項目、表示位置およびその大きさ等）が利用者によりS_{tech}内に定義可能である。さらに、６．１節（２）で示した$S_{compile}$には、利用者の入力するコマンドの構文とその翻訳法が定義できる。これにより、マクロ命令の定義をはじめ入力コマンドの名前やその構文を利用者に合わせることが可能である。

（２）応用プログラムへの適応機能

６．１節（３）（４）で示したように、ＵＩＭＳとAP module間で通信される情報の仕様はS_{bridge}により定義されている。また、AP moduleから出力される情報の仕様はS_{trans}に記述される。このような仕様をＵＩＭＳ内に定義できることから、

種々の応用プログラムを本ＵＩＭＳに接続することが可能となる。

第４章で述べたように、これらの定義情報は入出力装置からは独立している。従って、接続は他の方式[10]に比較してより容易であると言えよう。

７．本方式の特長

筆者らのＵＩＭＳモデルおよび本論文で述べた方式の特長を以下に列挙する。

（１）モデル内の機能が、入力流と出力流によって水平に分離され、さらに、言語レベル（図１中の②③④⑤）によって垂直に分割されている。これにより、

・ＵＩＭＳの構造が明確になる。
・ＵＩＭＳの実現が容易になる。

（２）機能単位毎に定義情報を分散させた。これにより、定義情報がより局所化され、従ってＵＩＭＳそのものの透明度が増した。

（３）モデル内に、応用プログラム部からも入出力部からも独立な空間が存在する。これにより、

・入出力に関する仕様定義と応用プログラムに関する仕様定義がそれぞれ独立に行える。
・応用プログラムのモジュールを入出力装置と技法を意識することなく構成でき、また、入出力装置と技法を応用プログラムを意識せずに構成できる。

（４）応用プログラムが基本モデル操作モジュールの集まりとして構成されている。これにより、

・より利用者主導の制御が実現される。
・基本操作コマンドの、ＵＩＭＳによる組立機能により、機能面における利用者への整合性が高まる。

８．おわりに

筆者らが構築したＵＩＭＳの情報構造モデルにおける基本的な考え方と、そのモデルに基づいてＵＩＭＳと実用的なテキストエディタを実現する方式について述べた。本論文の方式には７章で述べたようないくつかの利点が考えられる。実現システムの上でこれらを具体的に評価してゆくことが今後の課題である。

筆者らは現在、本論文で製作しているＵＩＭＳにディシジョンルールベースシステムを接続する計画も進めている。そこでは、複数の応用プログラム間通信、ユーザインタフェースの一貫性等の新たな問題を解決する必要がある。ＵＩＭＳ研究には、２章で述べた諸問題が山積みされている。これらを解明してゆくことも今後の課題である。

謝辞

日頃、御指導、御助言賜る本学　穂坂衛教授、平松啓二教授に深謝の意を表す。ＵＩＭＳ開発に御助力いただいている本学大学院　大塚誠二氏、田村浩三氏に謝意を表す。

参考文献

(1)守屋、「人間・機械インタフェース」、中田・南・平松共編「信号情報処理」第６章 オーム社 (1986.3)
(2)守屋、「人間・機械インタフェースの研究動向」、Computer Graphics Tokyo'86, Special Session, 日本能率協会 (1986.4)
(3)守屋、中谷、大塚、斉藤、「適応性のあるユーザインタフェースの構成方式について」、計測自動制御学会第２回ヒューマンインタフェースシンポジウム (1986.10)
(4)守屋、「ユーザインタフェースおよびその管理システムと標準化」、PIXEL No.51 (1986.12)
(5)守屋、斉藤、中谷、大塚、田村、「ユーザインタフェースの情報構造モデル－全体構造とその機能単位－」情報処理学会第34回全国大会(1987.3)
(6)守屋、中谷、斉藤、菅居、倉澤「ユーザインタフェースの情報構造モデル－機能単位の入出力関係－」情報処理学会第34回全国大会(1987.3)
(7)UIMS reduces amount of code required, IEEE CG & A, p.79 (1986.7)
(8)G.E.Pfaff(ed.), "User Interface Management System", Springer-Verlag (1985)
(9)F.R.A.Hopgood et al., "Methodology of Window Management System", Springer-Verlag (1986)
(10)F.R.A.Hopgood et al., "Introduction to the Graphical Kernel System", Academic Press, (1983);吉川弘之監訳,「コンピュータ・グラフィックス・基本ソフトウェア・ＧＫＳ」, 啓学出版
(11)T.Takala, "Communication Mediator - A Structure for UIMS",in:G.E.Pfaff(ed.), "User Interface Management System", Springer-Verlag (1985)