ソフトウェア工学　46－19
（１９８６．２．７）

ユーザインタフェイス作成支援システムについて

竹村　治雄　　辻野　嘉宏　　荒木　俊郎　　都倉　信樹
（大阪大学　基礎工学部）

１　まえがき

近年，計算機，とくにパーソナルコンピュータやオフィスコンピュータの急速な発達と普及により，様々な分野での計算機の利用が推進されてきた．そして，これらの分野で利用可能な多種多様のアプリケーションプログラム（ＡＰ）が開発されている．　これに伴い，計算機の利用者層も，従来のごく限られた専門家層から広く一般者層へと移り変わってきた．そのため，ユーザインタフェイス（ＵＩＦ）向上の必要性に対する認識が深まりつつある．従来のオペレーティングシステムの良否が，計算機システムの運用効率を大きく左右したように，ＵＩＦの良否は計算機を操作する人間の作業能率を大きく左右する．良好なＵＩＦは，ユーザの作業能率を高めユーザの能力を最大限に発揮させると同時に，不必要な操作や注意からユーザを解放することにより，ユーザに対して満足感を与える．そのため，従来よりテキストエディタ等のＡＰにおけるユーザインタフェイスに関する提案や研究が行われてきた[1～4]．一方，現在のＡＰ開発環境では，ＡＰのＵＩＦ設計はＡＰの作成者によって行われることが多く，良好なＵＩＦが提供されるか否かは，このＡＰの作成者自身（プログラマ）に委ねられることになる．従って，プログラマがＡＰのＵＩＦの向上に無関心であったり，そうでなかったとしてもＵＩＦ向上のための適切な手法を見い出せなかった場合，ＡＰのＵＩＦの向上は望めない．また，これらの問題点が解決されたとしても，ＵＩＦの向上を図ると通常プログラミングコストが増加することは避けられない．

本稿では，　ＡＰにおける良好なＵＩＦ提供の妨げとなる種々の原因をあげ，それらの問題を解決しＡＰでの良好なＵＩＦ提供を可能にするためのユーザインタフェイス作成支援システムＵＩＳＥ(User Interface Support Environment)を提案する．　ＵＩＳＥは，従来ＡＰごとに設計されていたＵＩＦをひとまとめにして，比較的高度なＵＩＦを従来の手法で記述する場合より少ない手間で作成できるようにするもので，プログラマは，ＡＰのＵＩＦをユーザインタフェイス記述言語ＵＩＤＬを用いて記述する．ＵＩＤＬで記述されたＵＩＦは，ＵＩＤＬコンパイラによって処理され，ＡＰの実行時にユーザインタフェイス管理システムＵＩＭＳによって解釈実行される．

以下，第２節で，良好なＵＩＦ提供の妨げとなる原因について考察し，第３節で，それに基づき試作を行ったＵＩＳＥについて述べ，第４節で，ＵＩＤＬとこれを用いたＵＩＦ記述について報告する．

２　良好なＵＩＦ提供の問題点と解決の一手法

本節では，アプリケーションプログラム（ＡＰ）におけるＵＩＦの改良を妨げる原因について考察し，解決の一手法であるＵＩＳＥの考え方について述べる．

ＵＩＦに関する研究，開発により，最近では良好なＵＩＦを提供するための各種の装置や手法が提案されているが[3～7]，これらの装置や手法は他のものとの比較実験の結果良好であると判断されたものであり，絶対的なＵＩＦの評価尺度は依然存在しないのが現状である．しかし，ＡＰのＵＩＦ改善のために，これらの手法などを用いることが有効であることは明らかである．従来，ＡＰにおけるＵＩＦの改良を妨げる原因として，

①計算機の能力的な制限，
②ＡＰの作成者（プログラマ）のＵＩＦの重要性に関する認識不足，
③プログラマの良好なＵＩＦ提供の手法に関する知識の欠如，
④ＡＰ開発コストに制限があることによるＵＩＦ部分の切り捨て，
⑤ＡＰの利用者（ユーザ）のＵＩＦに関する要求がプログラマに伝わらないこと，

などによるものが考えられる．しかし，①については，最近の急速なハードウェア技術の進歩により，計算機の能力的制限により不充分なＵＩＦの使用を強いられることは，少なくなりつつある．②についても，プログラマのＵＩＦの重要性に関する認識も深まりつつある．一方，③については，すべてのＡＰのプログラマに，ＵＩＦに関する知識や手法を教育することにより解決されるべきであるが，ＵＩＦに関する理論の確立していない現時点では困難である．④については，多種多用のＡＰの開発が要求され，必ずしも総てのＡＰに充分な開発コストが割り当てられない現状が反映されており，ＵＩＦの改善のために極端にプログラム記述量を増加させることは避けるべきである．　また，ＡＰのプログラマの関心がＡＰに要求される本来の機能の実現に偏るためにＵＩＦ部分が切り捨てられることも考えられる．⑤については，ユーザの要求をプログラマに伝えて根本的解決を図るとともに，ユーザによってＡＰのＵＩＦをある程度変更できるようにすることでも，改善が図れると考えられる．

また，単一のＡＰに於けるＵＩＦの改善とともに，それらのＡＰが利用される計算機システム全体のＵＩＦについても考察する必要がある．一般に計算機システム上では，複数のＡＰが利用されることが多く，これらＡＰ間でのＵＩＦの統一がとれていないと，利用者がしばしば混乱し，計算機システム全体のＵＩＦが悪化する．従って，少なくとも，同一利用者に使われるＡＰは，ＡＰ間でのＵＩＦの統一が行われるべきである．しかし，現在のＡＰプログラム開発環境でＡＰ間でのＵＩＦの統一を図るためには，ＡＰのＵＩＦ設計の細部に渡っての統一を行う必要があり，ＡＰの開発がＡＰごとに独立して行われる場合には，統一は非常に困難である．

以上の問題点を総合すると，良好なＵＩＦを提供しようとする際には，

①プログラミングの量はできるだけ増やしたくない，
②自然にＡＰ間でのＵＩＦを統一したい，
③ユーザがある程度ＵＩＦを管理することによりＵＩＦの不備を補ったり，個人の好みに対応できるようにしたい，

などの要求が生じる．ＵＩＳＥは，以上の点を考慮して，

①良好なＵＩＦを実現するために必要と思われる諸機能を，ユーザインタフェイス管理システムＵＩＭＳのもとにまとめ，
②ＡＰとユーザの間にＵＩＭＳを置くことによりＡＰ間のＵＩＦの統一をはかり（図２．１），
③ＵＩＭＳの持つ機能をＡＰ側からＵＩＦ記述言語ＵＩＤＬを用いてできるだけ少ない記述で利用できる

ように考えられたユーザインタフェイス作成支援システムである．この考え方では，ＵＩＭＳをユーザインタフェイスを統轄する一種のＯＳ（オペレーティングシステム）と見なしており，ＵＩＦに関する総ての資源を管理する．通常，このような環境を実現する場合，ＵＩＭＳを用いてのＵＩＦの管理を，

①ＡＰ側のみから行う，

②逆にユーザ側のみから行う，

③両方から行う（本システムでの管理方法），

の３通りの実現方法がある．①は，ＵＩＭＳをＡＰから見てＵＩＦを管理するサブルーチンの集まりとして実現するものである．この方法ではＡＰは必要な機能をサブルーチンとして呼び出せばよく，従来のＡＰの構造をそのまま利用できるが，サブルーチン間にまたがるＵＩＦをユーザ側からチューニングできない欠点を持つ．一方，②は，ＡＰプログラムをＵＩＭＳから見てサブルーチンの集まりとして利用できるようすることにより，完全にＵＩＦの記述をＡＰ本体の記述から切り離し，ユーザによるＵＩＦの自由なチューニングを可能にするが，ＡＰの構造を根本的に変える必要があり，プログラマの負担が増加する．③は，①と②の中間に位置するものであり，ＡＰ側とユーザ側それぞれにどの程度のＵＩＦの管理を許すかで実現手法が各種考えられる．　本システムでは，プログラマにとって処理のしやすいデータ入出力をセッションと呼ぶ単位にＵＩＤＬを用いて定義し，ＵＩＭＳが，データの入出力を，セッション単位で一括して行うこととした．このセッション単位の入出力により，ひとつのセッション内ではＵＩＦはＵＩＭＳによって管理され，ＵＩＤＬ記述によるＡＰ側からのコントロールとユーザからのコントロールとを両立させることができ，かつＡＰはセッションに関する記述を除き従来どおりのプログラミングが可能となっている．

本システムを用いてＡＰのＵＩＦを記述するプログラマは，あらかじめＵＩＤＬを用いて，

①一括して入出力するデータの構造，

②画面上での二次元の書式や変換規則を表すテンプレートの指定，

③テンプレート内のフィールドと呼ばれるデータ入出力のための領域とデータの対応，

④入出力データの満たすべき条件とデフォルト値，

⑤入力エラーに対する再試行回数，

⑥エラー，ヘルプなどのメッセージ，

を記述する．この記述がＵＩＤＬコンパイラの処理によりＵＩＤＬデータと呼ばれる中間コードに変換され，これをＡＰ実行時にＵＩＭＳが解釈実行することによりＡＰのＵＩＦを実現する．プログラムの処理の中心部分は，従来のプログラミング言語（現システムではＣ言語）を用いて記述する．ＵＩＭＳはこの他に，日本語入力，同義語定義など幾つかのＵＩＦに関する機能を単独で持つ．

ＵＩＤＬを用いたプログラミングの特徴は，

①プログラムのＵＩＦ部の記述が宣言的に行えるので記述が容易である，

②比較的高度なＵＩＦを従来の手法で記述する場合より少ない手間で作成できる，

③プログラマが意識せずにＡＰ間でＵＩＦを統一できシステム全体のＵＩＦの向上につながる，

④ＡＰ側でのＵＩＦの処理に関するオーバヘッドが低減される，

⑤ＵＩＦの個別設計や変更が容易である，

などがある．

次節ではＵＩＳＥの構成と機能について述べる．

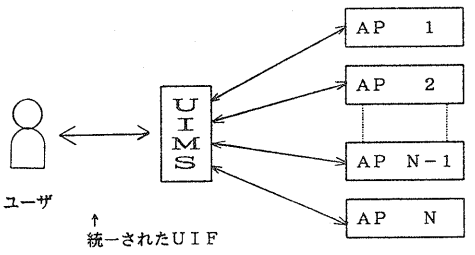

図２．１　ＵＩＭＳを用いたシステムでのＵＩＦ

３　システムの構成

ＵＩＳＥは，大きく，

⑴ユーザインタフェイス管理システムＵＩＭＳと，

⑵ＵＩＤＬ処理系　（ＵＩＤＬコンパイラ，テンプレートエディタ等）

の二つに分けることができる．システムの全体図を図３．１に示す．本節では，ＵＩＳＥの構成と機能について概略を述べる．

３．１　ユーザインタフェイス管理システムＵＩＭＳ

ＵＩＭＳはＡＰとユーザの間に介在し，システムのＵＩＦ全般の管理を行う．ＵＩＭＳの機能としては，①入出力データの一括した入出力，

②ヘルプ機能，

③自動エラー報告機能，

④自動メニュー生成機能，

⑤入出力データ変換機能，

⑥デフォルト値設定機能，

⑦日本語入力，

⑧マルチウィンドウ機能，

⑨同義語定義機能

⑩マクロコマンド機能等がある．

このうち，①～⑥の機能はＵＩＤＬによる記述が，ＵＩＭＳに組込まれたＵＩＤＬデータインタプリタによって実行されることにより実現される機能を含む．ＵＩＤＬデータインタプリタは，ＵＩＤＬコンパイラによって処理されたＵＩＤＬ記述の中間コードを解釈，実行することにより，ＡＰの実際のＵＩＦを実現するものである．ＵＩＤＬデータインタプリタとＡＰとは，別々のタスクとして実行され，ＡＰ中の入出力プリミティブにより同期がとられる．ＵＩＤＬデータインタプリタとＡＰの同期の様子を図３．２に示す．

ＵＩＭＳの残りの機能（⑦～⑩）は，ＵＩＤＬで陽に記述されていないＵＩＦに関する機能を実現するためのものである．これらは，特に新しい概念ではないが，ＵＩＦ向上のために有効であると考えられ，かつＡＰ毎の対応による一貫性の欠如を無くすためＵＩＭＳの機能として採用したものである．これにより，複数のＡＰ間でこれらの機能を共有することができ，これらの機能に関するＵＩＦがＡＰ間で統一できることになる．

以下では①～⑩の機能について簡単に紹介する．

①入出力データの一括した入出力

ＡＰでの処理に都合のよいひとまとまりのデータを一括して入出力する機能であり，ＵＩＤＬで記述されたセッション単位の入出力を行う．従って，ユーザは，セッション内ではデータの入力を任意の順序で行うことができる．またＡＰ側にとっても，内部処理に都合の良いデータ構造に直接データが入出力されるので，入出力のためのデータ構造の用意が不要である．一つのセッションで入出力されるデータの個数には制限はなく，図３．３に示されるような３次元の配列データの入出力も可能である．この例は，一度のセッションで３次元の配列に試験の成績データが入力される場合である．成績表は３次元の配列データで構成され試験回数×科目数×人数の構成になっているとする．ユーザに対して，このままの形でデータを表示することはできないので，ＵＩＭＳではこれをあらかじめ用意されたテンプレートと呼ぶ２次元の書式を用いて表示する．　従って，テンプレートさえ用意されれば，この成績表は，人数×科目の表を試験回数分入力することによっても，試験回数×科目数の表を人数分入力することによっても，試験回数×人数の表を科目数分入力することによっても作成可能である．この様子を図３．４～３．６に示す．いま表示が人数×科目の２次元で行われている場合，試験回数が３次元データの一種の切り口となっており，ＵＩＤＬで宣言することによりテンプレート中の切り口を変えることにより表示される面を変えることができる（図３．７）．これらの機能は，ＵＩＤＬを用いてＡＰで宣言的に記述をしておくことにより，ＵＩＭＳにより実現される．

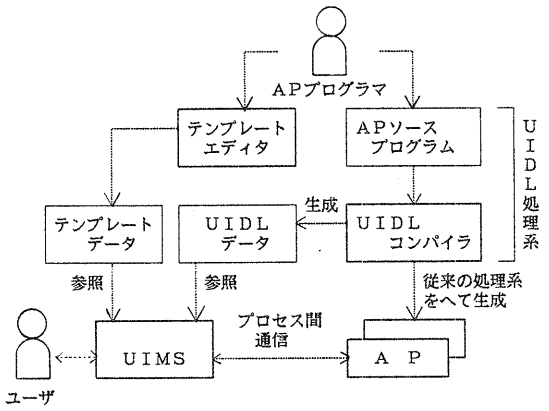

図３．１　システムの全体図

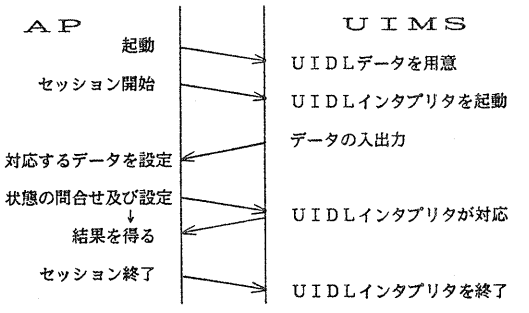

図３．２ＵＩＤＬデータインタプリタとＡＰの同期

②ヘルプ機能

ユーザがヘルプキーを押すことにより，起動される機能である．ＵＩＭＳ自身の操作法などのメッセージ又は，ＡＰによって用意されているＡＰにおけるデータの入力法，コマンドの説明などのメッセージが表示される．ＵＩＭＳ自身の操作法に関するメッセージは，グローバルヘルプキーが押された場合に，ＡＰによって用意されたものは，ローカルヘルプキーが押された場合にそれぞれ表示される．ＡＰによって用意されるメッセージは，あらかじめＵＩＤＬのメッセージ定義文で定義されたもので，セッション中の現在カーソルが位置しているフィールドで入出力されるデータに割り当てられたメッセージ総てが階層的に表示される．ヘルプメッセージ表示の例を図３．８に示す．

③自動エラー報告機能

ユーザが入力したデータが，ＡＰが要求しているデータの型，範囲と異なる場合に，ユーザに対してエラーの種類，入力すべきデータの型や範囲を表示する機能である．この機能により，報告されるエラー種類としては，

⑴入力されたデータの型がＡＰで要求される型と異なる場合（タイプエラー），

⑵入力されたデータの範囲がＡＰで要求されるデータの範囲と異なる場合（レンジエラー），

⑶入力されたデータ（複数のデータの場合を含む）があらかじめ設定された条件を満足しない場合（条件エラー），の三通りがある．それぞれのエラーに対するメッセージは，ＵＩＤＬのメッセージ定義文で定められる．また，データに要求されるタイプとレンジはテンプレートのフィールドに指定されたものであり，複数データ間の条件は，ＵＩＤＬの条件定義文で定められるものである．単独の入力データから判定可

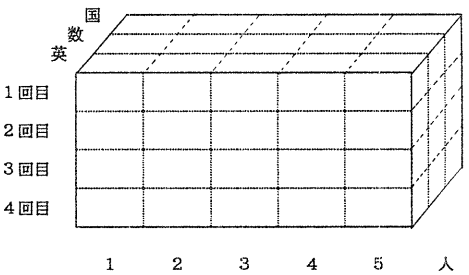

図３．３　３次元の配列データの例

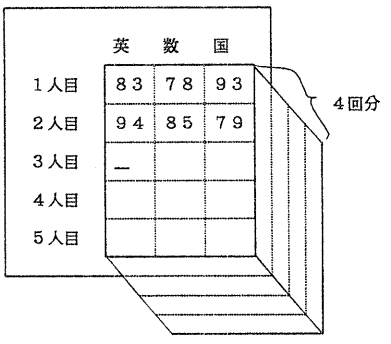

図３．４　人数×科目での入力

能なタイプエラーとレンジエラーはデータが入力された時点で報告され，条件エラーは，条件判定に必要なデータが総て入力された時点で判断され，エラーがあれば報告される．
それぞれのエラー表示の例を図３．９～３．１１に示す．

④自動メニュー生成機能

ユーザがメニューキーを押すことによって，その時点で，ユーザが選択可能な入力をメニューとして表示する機能である．ヘルプキーの場合と同様に，グローバルメニューキーとローカルメニューキーとが存在しそれぞれ，ＵＩＭＳの操作に関するメニューとＡＰでのメニューを表示するために使用する．ローカルメニューは，ＵＩＤＬで記述された要求される入力データが，限られた数以下の入力要素から構成される場合に，ユーザがローカルメニューキーを押すことにより自動的に生成され表示されるものである．ユーザは，表示されるメニューから選択を行うことにより，メニューを用いないで直接入力を行うのと同じに入力ができる．メニュー生成機能の例を図３．１２に示す．この例では処理内容として許される入力がＴＹＰＥ，ＳＯＲＴＡ，ＳＯＲＴＢのいずれかであることが，ＵＩＤＬによって記述されており，メニューを用いない場合は，これ以外の入力はレンジエラーとして処理される．そして，ユーザがローカルメニューキーを押すことにより，図のようなメニューが表示され，ユーザはメニューから適当な項目を番号入力もしくはポインティングデバイスによって選択する．

⑤入出力データに対する変換機能

ＵＩＭＳに対して入力されたデータは，必要ならばＡＰによって指定された型に変換される．　変換する型は，テンプレートのフィールドに割り当てられている．　この機能により入力データには，文字列から整数，実数などへの変換が，また，出力データには，その逆の変換が行われる．

⑥デフォルト値設定機能

プログラマによってあらかじめ用意されたデフォルト値を入力する機能であるが，ユーザはこのデフォルト値を変更できる．また，デフォルト値の設定されていない入力にもユーザにより設定が可能である．

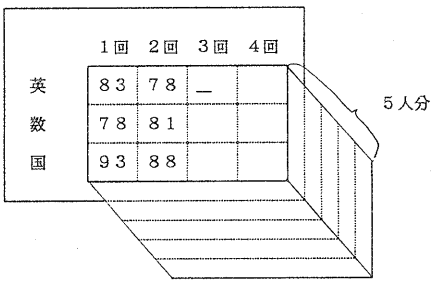

図３．５　試験回数×科目での入力

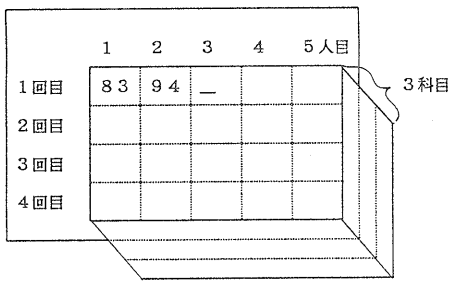

図３．６　試験回数×人数での入力

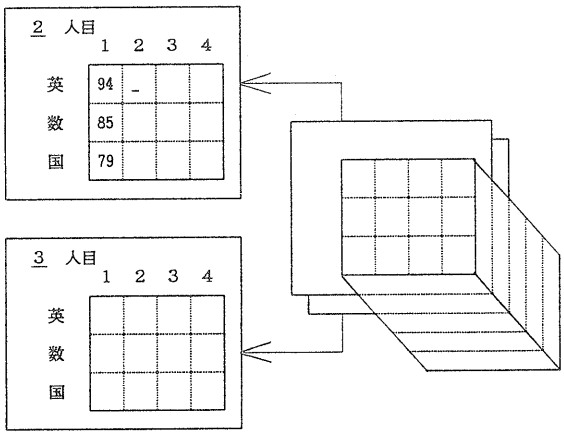

図３．７　切り口を移動する例

成績入力

第　1　回目

番号	名前	国語	数学	英語	合計
1	山田　太郎	83	74	89	246
・					
19	小泉今日子	76	▩		
20	松田　聖子				

↓　ローカルヘルプキーを押す

成績入力

第　1　回目

番号	名前	国語	数学	英語	合計
1	山田　太郎	83	74	89	246
・					
19	小泉今日子	76	▩		
20	松田　聖子				

ヘルプ：
数学のテストの得点を
入力して下さい

▩：カーソル

図３．８　ヘルプ機能の例

成績入力

第　1　回目

番号	名前	国語	数学	英語	合計
1	山田　太郎	83	74	89	246
・					
19	小泉今日子	76	18.3		
20	松田　聖子				

タイプエラー：
整数値で入力して下さい

図３．９　タイプエラーの例

⑦日本語入力機能

従来，日本語入力の方法としては，いろいろなものが提案されてきたが，最近はかな漢字変換によるものが定着してきた．ＡＰにおいて日本語入力を実現する場合の方法としては，

⑴ＡＰレベルで入力方法を実現する，

⑵汎用ライブラリとして入力方法を用意しておきＡＰ間で共有する，

⑶ＯＳレベルで入力方法を用意する，

の３通りが存在するが，ＵＩＭＳでの日本語入力は，⑶のＯＳレベルでの日本語入力に相当するものである．そのため，プログラマの意識なしにＵＩＭＳ上のＡＰでの日本語の取り扱いが可能になり，かつ日本語入出力の方法も統一できる．ただし，一般に，

⑴入力に用いられる辞書はＡＰによって必要な分野が異なることが多い，

⑵またユーザによっても必要な用語に差異がみられる，

⑶入力方法にもそれぞれ一長一短があり，個人の好みによる選択ができるほうが望ましい，

等の点を考慮して，ＡＰ及びユーザの両方から制御可能なものである必要がある．

⑧マルチウィンドウ機能

マルチウィンドウの概念は，物理的に限られた大きさをもつＣＲＴ画面を，仮想的な大きさを持つ画面として取り扱うことにより，画面の大きさを超えた利用を可能にするとともに，仮想的な画面を複数個用意することにより，自然なマルチタスクの概念を導入するものである．ＵＩＭＳでは，ＡＰでのセッションが，それぞれひとつのウィンドウをもち，入出力を行う．従って，ＵＩＭＳ下でのＡＰは，他のＡＰとの関係を考えることなく複数同時実行が可能である．ＵＩＭＳで用意されているウィンドウシステムは，縦方向可変長の仮想スクリーンを持つ．ユーザには，この仮想スクリーンの一部がウィンドウを通してＣＲＴ画面上に写像されて見える．仮想スクリーンに表示されている内容は，ウィンドウを通してローカルエディットを行うことができる．仮想スクリーンに表示された内容は，ファイルとして外部記憶装置上に保存することができる．

⑨同義語定義機能

複数のＡＰの入力のある文脈において，同じ意味を表すのに異なる語が用いられることがある．このような場合にＡＰのユーザは，しばしば混乱を生じ，誤った操作を繰り返すことがある．また，コマンドやパラメタとして要求される語の意味が通常ユーザが考える意味と大きく異なる場合，ユーザは不安感や違和感を覚える．この問題は，ＡＰのプログラマが用いる語の統一を行うことによりある程度改善できるが，ＡＰの用いられる分野やＡＰを用いるユーザによって語の意味が大きく異なることが多く，完全な解決は困難である．そこで，ＵＩＭＳではＡＰのある文脈において，ユーザが語の置換を設定できるようにすることで，この問題の解決を図っている．これがＵＩＭＳにおける同義語定義機能であり，ＵＩＤＬ記述によるＡＰの入出力ではＵＩＭＳによって入出力の文脈が特定できるために実現可能な機能である．

⑩マクロコマンド機能

ある特定のＡＰにおいて，しばしば特定のコマンド及び一連のパラメタが要求されることがある．また，場合によってはこれらの繰り返しが要求される場合がある．このような場合，特定のユーザにとっては，これらのコマンドやパラメタが常に一定であることがある．従って，あらかじめ入力するコマンド，パラメタをマクロコマンドとして定義しておき，マクロコマンド名の入力一回でコマンド，パラメタの入力が代用できる機能があると，便利である．この機能がＵＩＭＳにおけるマクロコマンド機能である．マクロコマンド機能とデフォルト値設定機能の違いはデフォルト値が個別のデータに設定されるのに対して，マクロコマンド機能はＡＰの入力のある文脈に於て一連の入力が定義される点である．この機能もＵＩＤＬ記述を用いたＡＰによる入出力では，ＵＩＭＳによって入出力の文脈の特定が出来るために実現可能な機能である．

３．２　ＵＩＤＬ処理系

ＵＩＤＬ処理系は，ＵＩＤＬによる記述を処理するＵＩＤＬコンパイラとＵＩＤＬによるユーザインタフェイスで用いられるテンプレートと呼ばれる一種の二次元書式記述を作成する為の専用エディタ（テンプレートエディタ）で構成され

成績入力

第　1　回目

番号	名前	国語	数学	英語	合計
1	山田　太郎	83	74	89	246
・					
19	小泉今日子	76	183		
20	松田　聖子				

レンジエラー：
得点は0－100の間で
入力して下さい

図３．１０　レンジエラーの例

成績データ表示

表示する学年とクラスを指定して下さい

学年　1　年
クラス　5　組

条件エラー：
この学年には
指定したクラスはありません

図３．１１　条件エラーの例

メニュー選択の例

成績処理

処理内容
処理項目
出力形式

↓　ローカルメニューキーを押す

成績処理

処理内容
処理項目
出力形式

次のなかから選択してください
1．TYPE
2．SORTA
3．SORTB

番号入力またはポインティングデバイスで，
メニューから選択する．

図３．１２　メニュー生成機能の例

る．既に述べたように，ＵＩＳＥでは，プログラムのユーザインタフェイスをＵＩＤＬと呼ばれる専用記述言語を用いて記述する．ＵＩＤＬによる記述は，通常，プログラムのユーザインタフェイス以外の部分を記述した従来のプログラミング言語による記述部分に埋め込まれて用いられる．従って，ＵＩＤＬコンパイラは，ＵＩＤＬによる記述の埋め込まれたソースファイルを入力とし，これを処理し，　ＵＩＤＬ記述を含まない従来のコンパイラで処理可能なソースプログラムと，実際にＵＩＤＬによる記述を実現するためにアプリケーションプログラム実行時にＵＩＭＳによって解釈実行される一種の中間コードであるＵＩＤＬデータとを出力する．今回作成したＣ言語用のＵＩＤＬコンパイラは，Ｃ言語の構文のうち，ＵＩＤＬの処理に必要な部分を解析しながら処理を行う．実際にはＣコンパイラの構文解析部を一部そのまま利用した．ＵＩＤＬコンパイラの出力を中間コードとしたのは，ＵＩＭＳの移植性を確保するためである．

テンプレートエディタは，ＵＩＤＬのテンプレート宣言文によって指定されるテンプレートを実際に作成する際に用いるエディタで，テンプレート自身が二次元の書式を記述するという性質から，実際にＣＲＴの画面を見ながらテンプレートがユーザ表示される状況を常に把握しながら作業が出来るように画面エディタとして実現されている．テンプレート内のフィールドが持つ各種の属性はモード切換えによって表示されるフィールド設定用画面で設定できる．テンプレートエディタによるテンプレート編集の例を図３．１３に示す．
図の例では，テンプレートを作成するプログラマは，まず画面上でタイトル，罫線などのテンプレートの固定部分を入力し，更に可変部分であるフィールドの位置と番号を入力して行く，同じ番号のフィールドが用いられた場合これらは一次元で順序付けられる，さらにカーソルがフィールドを指している状態で「書式キー」を押すことにより，そのフィールドに関しての書式を入力できる（図３．１４，３．１５）．この状態でプログラマは，フィールドの大きさ，入力データの変換ルール，表現形式，許される範囲を入力できる．テンプレートエディタによって作成されたデータは，テンプレートファイルとして保存され，ＵＩＤＬデータファイルと共に入出力の実行時にＵＩＭＳによって利用される．

ＵＩＤＬ処理系を用いてアプリケーションプログラムを作成する場合の処理の流れを図３．１６に示す．図の例では，プログラマはＵＩＤＬによる記述を含んだＡＰのソースプログラムを作成する．このソースプログラムがＵＩＤＬコンパイラによって処理され，ＵＩＤＬデータファイルとＵＩＤＬ記述を含まないソースプログラム（例ではＣ言語）に分けられる．さらに，このソースプログラムは従来のコンパイラによって処理され，ついでリンカにより通常のライブラリ及び

成績入力

第　＄６回目

番号	名前	国語	数学	英語	合計
1	＄１	＄２	＄３	＄４	＄５
2	＄１	＄２	＄３	＄４	＄５
3	＄１	＄２	＄３	＄４	＄５
．					
３９	＄１	＄２	＄３	＄４	＄５
４０	＄１	＄２	＄３	＄４	＄５

図３．１３　テンプレート編集（１）

成績入力

第　＄６回目

番号	名前	国語	数学	英語	合計
1	＄１	＄２	＄３	＄４	＄５
2	＄１	＄２	＄３	＄４	＄５
3	＄１	＄２	＄３	＄４	＄５
．					
３９	＄１	＄２	＄３	＄４	＄５
４０	＄１	＄２	＄３	＄４	＄５

フィールド　＄１
フォーマット　＃＃＃＃＃
変換　なし
入出力ルール　左づめ
範囲指定　なし

図３．１４　テンプレート編集（２）

成績入力

第　＄６回目

番号	名前	国語	数学	英語	合計
1	＄１	＄２	＄３	＄４	＄５
2	＄１	＄２	＄３	＄４	＄５
3	＄１	＄２	＄３	＄４	＄５
．					
３９	＄１	＄２	＄３	＄４	＄５
４０	＄１	＄２	＄３	＄４	＄５

フィールド　＄２
フォーマット　＃＃＃
変換　１０進　整数
入出力ルール　右づめ　ゼロサプレス
範囲指定　０　．．　１００

図３．１５　テンプレート編集（３）

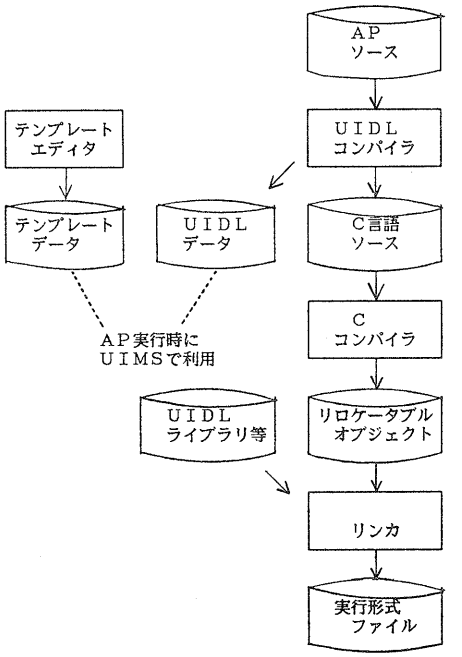

図３．１６　ＵＩＳＥでのプログラム開発の流れ

ＵＩＭＳ用ライブラリと結合され実行形式のプログラムが生成される．ＵＩＭＳ用ライブラリは，ＡＰがＵＩＭＳを通しての入出力を行う際にＵＩＭＳとのプロセス間通信等に必要なプリミティブを集めたものである．

4　ユーザインタフェイス記述言語ＵＩＤＬ

本節では，ＵＩＤＬとＵＩＤＬによる記述の具体的な例について述べる．

今回設計したＵＩＤＬは，第３節で既に述べたようにＣ言語に埋め込んで用いるが，一般にＡＰのＵＩＦを記述する言語の実現方法としては，

①ＵＩＦ以外の部分を記述する言語とは独立に作る，
②ＵＩＦ以外の部分を記述する言語に依存して作る，

の二つが考えられる．

①の方法を用いた場合，利点としては，

(1)ＵＩＤＬ処理系は容易に作成できる，
(2)複数の言語で同じＵＩＤＬを用いることができる，

などがあり，欠点としては，

(1)ＵＩＤＬによる記述と従来の言語による記述の間に違和感が生じる，
(2)ＵＩＤＬ記述部分とそれ以外の部分との橋渡し的な記述が必要である，

などがある．

一方，②の方法を用いた場合は，利点としては，

(1)ＵＩＤＬによる記述とそれ以外の部分との記述に違和感が少ない，
(2)ＵＩＦの記述に必要な記述量を減らすことができる，

などがあり，欠点としては，

(1)処理系を各言語ごとに作る必要がある，
(2)各言語ごとにＵＩＤＬが異なる，

などがある．

しかし，通常ＡＰのプログラマが複数の言語を混在して用いることは少なく，複数言語間のＵＩＤＬの統一よりも，単一言語でのＵＩＦ記述を減少するほうが，プログラマの負担を減少できると考えられる．また，処理系作成のためのコストは一時的なものであり，大きな欠点ではない．これらを総合して，今回設計したＵＩＤＬは②の言語に依存したものとした．

ＵＩＤＬによるＵＩＦの記述は，大きく３つの部分に分かれている．それらは，

①入出力されるデータの実体を定義する部分，
②セッションを定義する部分，
③実際の入出力を記述する部分，

である．以下では，具体例をあげてこれらについて述べる．図４．１は，４０人の英語，数学，国語のテスト各５回分の成績を入力する場合のＵＩＦの記述例である．

まず，①は，ＵＩＭＳがセッション単位でのデータ入出力を行なう場合に，実際に入出力されるデータの実体を定義する部分であり，プログラマが内部処理に都合のよい構造で宣言することにより，そのデータの実体をそのまま入出力用の領域として用いることができ，記述量の増加を防いでいる．図の例では成績表は３次元の配列として宣言されている．

次に，②は，ＵＩＤＬによるＵＩＦ記述の中心をなすもので，この定義部には，

(1)セッション名とそのセッションで入出力されるデータを表す仮の名前と構造，
(2)テンプレートの指定とテンプレート内のフィールドと入出力データの割り当て，
(3)入出力データのデフォルト値，
(4)入出力データ毎のリトライ回数，
(5)入出力データが満足すべき条件を表す式，
(6)各種メッセージ

を記述する．(1)では，セッション名とそのセッションで入出力されるデータを表す仮の名前と構造をＣ言語の関数定義と同様の記述で定義する．図では，SIO という名前を持つセッションが定義されている．(2)は，画面上で二次元の書式や変換規則を表すテンプレートの指定とテンプレート内のフィールドと呼ばれる領域とデータの対応を示すもので，図の例では，tempa という名前のテンプレートが指定され，略記法を用いて簡素にフィールドの割り当てが記述されている．また，変数ｉが３次元データの切り口を表す切り口変数として用いられている．(3)～(6)の項目は，必要な場合のみ記述すればよく，図の例では(3)と(4)は記述されていない．図の例では(5)の条件式が代入に関しては常に真であるという性質を利用して合計得点の計算を行っている．また(3)(4)(6)の各項目は，単一データに対してだけでなく，データ構造の領域を指定して割り当て可能である，図の例では，メッセージの定義の部分で３次元配列shyou の２番目の添字を０に固定して，他の部分を指定しないことで〝国語のテストの得点を入力してください〟というメッセージをshyou の２番目の添字が０である総ての配列要素に割り当てている．

最後に，③は，幾つかのプリミティブを使ってＣプログラム中に記述される．これらのプリミティブは，

(1)セッションの開始(opensession)，
(2)セッションの状態の問い合わせ(getstat)，
(3)セッションの状態の再設定(setstat)，
(4)セッションの終了(closesession)，

の４つである．

(1)は具体的には，入出力するデータの実体をセッションに対応付け，ユーザに対してテンプレートを通しての入出力を開

```
/*  入出力されるデータの実体を定義（Ｃ言語による）*/
char namelist[40][12];  /*  名前表  */
int  seiseki[5][3][40];   /*  成績表  */
int  goukei[5][40];     /*  合計の表  */

/*  セッションを定義   (SIO という名前のセッション記述) */
session SIO(namelist,seiseki,goukei)
    char namelist[40][20];
    int  seiseki[5][3][40];   -/*  仮入出力引数の構造を定義  */
    int  goukei[5][40];
{                       /*  tempa という名前のテンプレートを指定  */
  template tempa (i=0..4) { /*  テンプレートとの対応を定義する  */
    $1[0..39] = namelist[0..39],output;
    $2[0..39] = seiseki[i][0][0..39],input;   /*  ｉは切り口変数  */
    $3[0..39] = seiseki[i][1][0..39],input;
    $4[0..39] = seiseki[i][2][0..39],input;
    $5[0..39] = goukei[i][0..39],output;
    $6 = i,input;
  }
  cond cnda (i=0..4; j=0..39) {   /*  条件，簡易計算記述  */
    goukei[i][j] = seiseki[i][0][j]+seiseki[i][1][j]+seiseki[i][2][j];
  }                       /*  この例では  成績の合計を計算  */
  msg {                  /*  ヘルプ，エラーメッセージ記述部  */
    seiseki[ ][0][ ] = help: "国語のテストの得点を入力して下さい";
    seiseki[ ][1][ ] = help: "数学のテストの得点を入力して下さい";
    seiseki[ ][2][ ] = help: "英語のテストの得点を入力して下さい";
    seiseki[ ][ ][ ] = type: "整数値で入力して下さい",
                       range:"得点は０－１００の間で入力して下さい";
  }
}
/*  実際の入出力の記述（Ｃ言語による呼出部分）  */
  sid=opensession(SIO,namelist,seiseki,goukei);/* 入出力開始 */
  while(getstat(sid,ALL) != DONE);            /* 入出力終了を待つ */
  closesession(sid);                          /* 入出力終了 */
```

図４．１　ＵＩＤＬによる記述例

始する．⑵は，各入出力データまたはデータ全体に対して，入力が終了したか否か，出力が終了したか否か，現在までの入力エラー回数を得る．　⑶は，逆にこれらの状態を設定する．　⑵と⑶により，同一セッションでの再入力や，入力エラー時のより細い処理がＡＰ側で可能である．⑷は，入出力を終了し，対応するテンプレートのユーザに対する表示を終える．図の例では，セッションを開始したのち総てのデータが入力されるまで待ち，その後セッションを終了している．

5　あとがき

ＵＩＳＥは，広範なＡＰに於て，ＵＩＤＬを用いて比較的少ない記述で，統一のとれた比較的良好なＵＩＦを提供することを可能にする．このため，ＡＰのプログラマにもユーザにも充分利用価値を持つと考えられる．一方では，システムが非常に複雑になりＵＩＳＥ実現にかかるコストは大きくならざるを得ない，しかし実現のためのコストは一時的なものであり，システムによりもたらされる利益を考えれば，このような環境を実現する価値は充分にあると考えられる．

現在，ＵＩＳＥのプロトタイプの試作を行い，既にＵＩＤＬコンパイラなどが実際に試用されている．今後，実際にＵＩＳＥシステムを活用し，その評価を行い，その結果を基にＵＩＤＬの記述能力に関する検討と改良，ＵＩＭＳに必要な機能の追加などを行う予定である．　また，ＵＩＳＥをＡＰのＵＩＦのプロトタイプ作成に利用し，ＡＰのＵＩＦ設計にユーザの意見をフィードバックして，更に良好なＵＩＦの提供をめざす応用についても検討中である．

謝辞

本システムのＵＩＤＬコンパイラの実現に御協力いただいた今中崇雄氏（現シャープ株式会社）に感謝いたします．

参考文献

(1) S.K.Card, T.P.Moran, and A.Newel:"The Key-stroke-level model for user performance time with interative systems", Comm. ACM, 23, 7, pp. 396-410 (July 1980).
(2) M.Hammer, J.S.Kunin and S.Schoichet:"What makes a good user interface?", 1983 Office Automation Conferrence Digest, AFIPS Press, pp. 121-130 (February 1983).
(3) T.L.Roverts and T.P.Moran:"The evaluation of text editors : methodology and empirical results", Comm ACM, 26, 4, pp. 265-283 (April 1983).
(4) T.K.Landauer, K.M.Galloti, and S.Hartwell:"Natural command names and initial learning : a study of text-editing terms", Comm. ACM 26, 7, pp. 495-503 (July 1983).
(5) S.K.Card, W.K.English, and B.J.Burr:"Evaluation of mouse, rate controlled isometric joystic, step keys and text keys for text selection on a CRT", Ergonomics, 21, 8, pp.601-603 (Augst 1978).
(6) K.Snowberry, S.R.Perkinson and N.Sisson:"Computer display menus", Ergonomics, 26, 7, pp. 699-712 (July 1983).
(7) L.J.Bass:"A generalized user interface for applications programs(11)", Comm. ACM 28, 6, pp. 617-627 (June 1985).