ヒューマンインタフェース 61－7
（1995. 7. 20）

ユーザモデルの利用及び対象システムの視覚化による学習支援

倉橋 利幸, 西村 俊和, 美濃 導彦, 池田 克夫
京都大学 工学部
E-Mail: kurahasi@kuis.kyoto-u.ac.jp

本研究では，特に初心者ユーザを対象として，ユーザが操作対象とするシステムと対話しながらその使用方法を学習するのを支援する学習支援環境の構築を目的とする．その実現に必要な要素として，(1) 対話過程から得られるユーザモデルの利用，(2) 対象システムの視覚化について考察する．ユーザモデルの利用により，対象システムはユーザが必要とする情報を適切な形で教えることができ，対象システムの視覚化により，ユーザは対象システムに関する理解が容易にできる．また，考察結果に基づいて実現したシステムを用いて実験を行い，被験者の感想から，これらの要素がある程度の効果をもつことが確認できた．

Computer-Supported Learning with User Models and Visualization of the System

KURAHASHI Toshiyuki, NISHIMURA Toshikazu,
MINOH Michihiko, IKEDA Katsuo
Faculty of Engineering, Kyoto University
E-Mail: kurahasi@kuis.kyoto-u.ac.jp

We discuss how a computer system can support a user, particularly a novice user, who learns about the target system that he wants to use, and describe two subjects: (1) user models obtained through interaction between the user and the target system, and (2) visualization of the target system. The user model enables the target system to present the user with any information he needs, and the visualization enables the user to understand the target system easily.

1 はじめに

現在のコンピュータシステムの多くは，決して使いやすくない．とりわけ初心者ユーザにとってはそうである．これには次の二つの問題点がある [1]．一つは，コンピュータに指示するためのインストラクション言語が非常に低レベルで，ユーザはそれを学習し記憶しておかなければならないことである．もう一つは，エラーが発生しても，それについての原因説明や対処方法が与えられないことである．

多くの場合，ユーザが自分でマニュアルを調べてこれらの問題に対処してきた．しかし，初心者ユーザにとって，膨大な量のマニュアルの中から，意図するタスクの実行方法やエラーの対処方法など，その時点で必要な情報を探し出すのは困難である．そこで，現在の状況やユーザの操作履歴を調べてユーザに必要な情報を提示し，ユーザが利用方法を学習するのを対話により支援するシステムの構築が必要となっている．

対話によるシステム機能の学習支援に関する研究が過去にいくつかなされている．Hecking の SINIX Consultant(SC) [2] では，ユーザのコマンド系列からユーザの意図するプランを認識し，同じプランに対してもよりよい方法があればそれをユーザに教えている．さらに，Wilensky らの UNIX Consultant(UC) [3]，垣内らの ASSIST [4]，上原らの Neo-ASSIST [5] などでは，ユーザの知識状態を表すユーザモデルを用いて自然言語によるあいまいな質問に対する適切な回答を可能にしている．

しかし，SC のようにシステムが判断してユーザにヘルプを出す場合，特定のコマンド系列を認識するたびにヘルプを出すと，かえってユーザに煩わしい思いをさせる．したがって，ユーザが本当に分からない場合にのみヘルプを出せるようにすることが重要である．また，UC，ASSIST，Neo-ASSIST のように質問方法として自然言語が用いられている場合，特に初心者にとっては，疑問が自分の言葉で表現できない場合には質問することができない．このような場合，画面にあるものを指すという単純な質問方法が好まれると考えられる．

以上のことから本研究では，ユーザに適切な回答をするためだけでなく，システムが判断してヘルプを出す回数を抑制するためにもユーザモデルを用いて対話することにより，システム機能の学習支援を行う．さらに，画面にあるものを指すだけの単純な質問方法を提供する．

以下，2章では学習支援システムの役割と構成について述べ，3章では本システムを構成する上で特に重要なユーザモデルについて述べる．4章では実際に本システムを使用してもらう形で行った実験に関する考察を行う．5章は結論である．

2 対象システムを学習するための支援システムの役割と構成

2.1 学習支援システムの位置づけ

学習支援環境の概要図を図 2·1に示す．

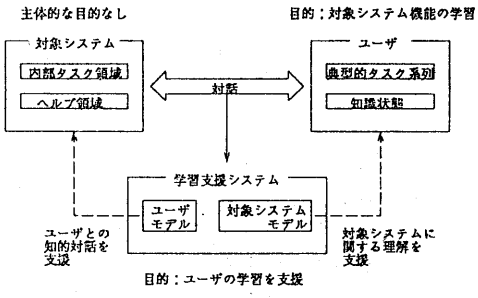


図 2·1 学習支援環境の概要図

ユーザとユーザが操作対象としているシステム（以下，対象システム）とが対話を行う．

ユーザ　対象システム機能を学習するという目的をもった主体的な学習者．対象システムに対してコマンド入力及び質問を行い，その応答により状態を変え，さらなるコマンドや質問を生成する．

対象システム　主体的な目的をもたず，ユーザからの入力に対して状態を変え，ユーザに対してある決められた応答をする．

この二者に対して，学習支援システムの役割は次のようになる．

学習支援システム　ユーザが対象システム機能を学習するのを支援するという目的をもち，対象システムが参照するためのユーザモデル（2.2節）と，ユーザが参照するための対象システムモデル（2.3節）を備える．

2.2 学習支援を目的とした対話機構

2.2.1 知的対話の重要性

対象システムが，
- ユーザの質問に対して，単にマニュアルをみせる，
- ユーザの正しい操作に対して，単に実行結果を示す，
- ユーザの誤りに対して，単に誤りであると知らせる，

だけでは，コミュニケーションをとるための知的な対話をしているとはいえない．

学習支援システムは，ユーザの学習を支援するという立場からすると，

h1) 質問回答支援　ユーザが質問すれば，そのユーザに適した内容の回答を提示する

h2) タスク実行支援　ユーザが正しいことをしても，より良い方法があれば適切なときに教える

h3) エラー回復支援　ユーザの誤りに対しては，対処方法を教える

などによる知的な対話を，対象システムが行えるように支援しなければならない．

2.2.2　知的対話におけるユーザモデルの必要性

前節で考察したように学習支援システムは，対象システムがユーザと知的な対話を行えるように支援する．そのため学習支援システムは，ユーザの状態を推測して得られるユーザモデルを用意し，それを対象システムに参照させる．対象システムはこのユーザモデルを参照することによって，ユーザと知的な対話を行うことができる．

ここで，前節で考察した知的対話の重要性h1)～h3)のそれぞれについてユーザモデルの必要性を詳しく考察する．

h1) 質問回答支援

ユーザの質問に対する回答としてヘルプを提示するとき，マニュアルのようにあらかじめ用意された内容をユーザに見せるだけでは，同じことの繰り返しや説明不足を招く．この問題を解消するために学習支援システムが，その時点でユーザが何を知っていて何を知らないかをユーザモデルとしてもっておき，それを対象システムに参照させることにより，ヘルプの内容を動的に変える必要がある．

h2) タスク実行支援

ユーザが実行したコマンド系列が冗長であった場合に，より効率の良い系列を学習支援システムが能動的に教える．これはユーザの要求なしにヘルプを出すものであるので，ユーザはかえって煩わしく思うこともある．この問題を解消するために学習支援システムは，ユーザが冗長な入力を用いている頻度をユーザモデルとしてもっておき，それを参照してヘルプを出すのを抑制する必要がある．

h3) エラー回復支援

ユーザの操作誤りの内，コマンド入力誤りなどの単純な誤りは対処方法がすぐに分かるが，対象システム状態に関する誤解が原因である誤りは対処方法がすぐには分からない．これに対しては，コマンドの出すエラーメッセージの強化など，対象システムの設計においてある程度対応できると考えられる．

2.3　対象システムの視覚化

2.3.1　対象システム視覚化の効果

ユーザは，対象システムを使いながら，その対象システムに関するモデル（対象システムモデル）を自分の中に構築していく．そして，対象システムの状態変化を推測したり，次に入力するコマンドを決めるときに，その対象システムモデルを参照する．したがって，ユーザが誤った対象システムモデルを構築すると，対象システムに対して誤った操作をする機会が増え，対象システムについての理解が進まない．このことから，学習支援システムが対象システムを視覚化し，ユーザが正しい対象システムモデルを得られるようにすれば，対象システムを正しく使用できる機会が増え，対象システムに関する理解が進むと考えられている [1].

ユーザが対象システムに対して操作を行うには，二通りの指示の与え方がある．再認方式では，「これを実行して下さい」という形で指示を与える．「これ」は画面に提示されているものの中から選ぶことができる．再生方式では，「～を実行して下さい」という形で指示を与える．「～」はユーザが自分で記述しなくてはならない．特に初心者にとっては，再生方式よりも再認方式の方が簡単に操作できるので理解しやすい．対象システムを視覚化することにより，再認方式を用いることが可能になる [1].

コマンド再生支援　ユーザのコマンド実行方法に関する学習を支援するため，メニューを用いた再認方式のインタフェースを用いる．しかし，再認方式は可能な操作が限られている場合にのみ有効であり，自由度の高さや効率においては再生方式の方が優れている．したがって，ユーザが再生方式のインタフェースを学習するのを支援するために，コマンドラインとの対応をとりやすくするように，メニュー実行されたコマンドに対応するコマンドラインを逐次画面に提示していく．

単純な質問方法の提供　ユーザに自然言語によるあいまいな質問方法を提供する研究がいくつかなされている [4, 5, 3]．この方法は再生方式であり，質問方法に自由度がある．ところが，ユーザが疑問に思ったことや分からないことを自分の言葉で表現できない場合には質問できない

先に述べたコマンド実行においては，学習の目的としてコマンド再生を考えた．しかし，質問は学習の過程で必要となるものであって，質問を再生できるようになる必要はない．したがって，再認方式を用いて，画面にあるものを指すだけで簡単に質問できるように

する．

2.3.2 対象システムモデル

ユーザと対象システムに関連したタスクは，ユーザが対象システムを使って実行することのできる外部タスクと，ユーザが対象システムを使うために実行しなければならない内部タスクとに分けられる [1]．内部タスクは，ある特定の対象システムにおける操作単位であり，キーストロークやボタン押下の一連の操作のことである．UNIX システムの例では，プログラムの作成やそのための編集，コンパイルなどが外部タスクで，emacs，gcc などの実行が内部タスクである．

学習支援システムは，ユーザに対して対象システムを視覚化するために，あらかじめ次に示す対象システムモデルを用意しておく必要がある．

内部タスク領域の構造 ユーザがタスクを実行して操作する対象（UNIX では，ファイル・ディレクトリ）を一般的に表現するための構造

内部タスク領域の状態 ある一時点における内部タスク領域の状態

タスク構造 ユーザが実行するタスクの階層構造

前二つは対象システムとして何を用いるかに大きく依存するのでここでは特に考察せず，タスク構造の表し方についてのみ考察する．

対象システムにおけるタスク構造は，タスク領域の状態に依存するタスクの場合，「タスク領域の状態が S のとき目標 G を達成するには，M を副目標とする」（以下，[S, G → M]）の形で，タスク領域の状態に依存しないタスクの場合，「目標 G を達成するには，M を副目標とする」（以下，[G → M]）の形でそれぞれ表すことができる．

[S, G → M] の構造はタスク領域の状態に依存する．ユーザがもつタスク領域の構造を表す対象システムモデルが学習の過程で順次更新されると，タスク領域の状態に依存するタスク構造も動的に変化していくことになる．しかもそれは人によって異なると考えられるので，[S, G → M] の形のタスク構造全体を学習支援システムがあらかじめもっておく必要はない．

このことから，学習支援システムが対象システムモデルとしてもつタスク構造は，[G → M] のみとする．タスク領域の状態に依存するタスク構造については，それに関するヘルプを見せることにより，ユーザの中でまとまりのあるタスク構造として再構築されることになる．

以上，本章で考察した，学習支援システムの役割を表 2·1にまとめる．

3 ユーザモデルの構成と利用

3.1 ユーザの知識状態

3.1.1 ユーザの獲得する知識

ユーザは対象システムとの対話の過程で，対象システムに関する様々な概念的知識を別々に獲得する．ユーザはこれらの知識をもとにして，自分の中でまとまりのある対象システムモデルを構築する．

2.3.2節で考察したように，タスク構造に関する知識については，[G → M] のように構造化された形でユーザに見せる．これは，図 3·1のような階層構造を形成している．個々の知識は，例えば，

[programming → edit, compile_link, execute, process_files],

[edit → emacs]

のように表現される．

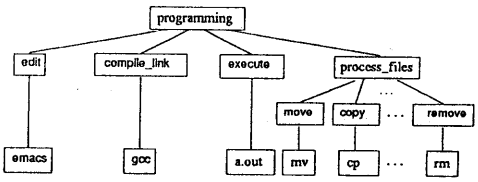


図 3·1 タスク構造に関する知識の階層構造

3.1.2 ユーザモデルとしての知識状態

2.2節で述べたように，対象システムがユーザと知的な対話を行うために，学習支援システムは，ユーザの知識状態を表すユーザモデルを対象システムに参照させて，その時点でのユーザの知識状態に合った適切な内容の応答（ヘルプ文）を出せるようにする．

ASSIST[4] では，個々のヘルプ文に使用する概念がユーザの既知概念であるかどうかをユーザモデルとしてもっている．これに対して本研究では，既知概念の代わりに，個々のヘルプ文を，個々の文章ごと，あるいは内容的にまとまりのある複数の文章ごとに区切り，その区切られたそれぞれ（ヘルプ要素とよぶ）についてユーザが知っているかどうかを調べることにする．これにより，あるヘルプ要素がすでにユーザに提示され，ユーザがそれを知っていると判断される場合には，関連するヘルプ要素は提示しないことが可能になる．

3.2 ユーザの典型的な内部タスク系列

3.2.1 プラン認識

SC[2] では，ユーザのコマンド系列からユーザの意図するプランを認識することを試みている．しかし，

表 2.1 学習支援システムの役割

支援内容	ユーザに必要な情報を適切なときに教える	ユーザの対象システムに関する理解の支援
支援方法	対象システムがユーザと知的対話を行えるようにする	対象システムを視覚化してユーザに見せる
必要なもの	ユーザモデル	対象システムモデル

ユーザのコマンド系列を完全に意味のあるプランに分けるためには，多くの知識を設計者があらかじめ与えておかなければならない．また，あらかじめ多くの知識を与えるとしても，個々のユーザの意図するすべてのプランを認識するのはかなり困難である．

従って本研究では，ユーザの意図するプランすべてを認識することは考えず，h2) タスク実行支援を実現するのに必要なプランに関する知識のみを与えることにする．

学習支援システムは，対象システムに同じ状態変化を生じさせる内部タスク系列の集合（プラン）I_1〜I_nをもっておく．各I_iにはI_{i1}〜I_{im}の要素がある．例えば，「ファイル名を変更する」プランI_1は，

$I_1 = \{I_{11}, I_{12}\}$，
I_{11} = {'cp file1 file2', 'rm file1'}，
I_{12} = {'mv file1 file2'}
と表すことができる．

ユーザの実行した内部タスク系列からI_{ij}を抽出すると，プランI_iが実行されたことを認識し，このとき，I_{ij}よりも手間の少ない内部タスク系列があれば，それを提示する．

3.2.2 ヘルプ抑制のためのユーザモデル

2.2.2節で述べたように，特定の内部タスク系列に対して常にヘルプを提示すると，ユーザが煩わしく思うこともある．

そこで，ユーザが一定期間t_iの間にI_{ij}を実行した回数n_{ij}が一定回数c_iをこえた場合にのみ，より良い方法I_{ik}を提示することにする．一定期間t_iの間に限定するのは，ユーザはt_i以上過去に実行したことは覚えていないとして考慮に入れないためである．このt_i, c_i, n_{ij}がシステムが判断して出すヘルプを抑制するためのユーザモデルであり，t_i, c_iを動的に変化させることにより，ユーザの好みに合わせる．

- プランI_iに対して，I_{ik}を一度も実行したことがない状態でI_{ij}を実行した場合：$t_i \longleftarrow t_i$

  ユーザはI_{ik}を知らなかったとして，プランI_iに対してI_{ik}が実行可能であることを提示する．ユーザモデルは変更しない．

- プランI_iに対して，I_{ik}を実行したことがある状態でI_{ij}を実行した場合：$t_i \longleftarrow \lfloor t_i \times (1+D) \rfloor$

  より良い方法を知っているのに冗長な方法を使ったとして，ヘルプを出す方向にユーザモデルを修正する．

- プランI_iに対して，I_{ik}を実行した場合：
  $t_i \longleftarrow \lfloor t_i \times (1-D) \rfloor$

  より良い方法を使っているとして，ヘルプを出さない方向にユーザモデルを修正する．

$\lfloor x \rfloor$：xをこえない最大整数，
D：定数，$0 < D < 1$

今回は，t_iはコマンドライン数十行，c_iは数回程度，Dは0.1として，t_iは20より小さくならないようにした．c_iをたびたび修正することはあまり意味がなく，t_iを大きく（小さく）するとc_iを小さく（大きく）したのと同じ効果が得られると考えられるので，c_iは固定したままとした．

以上，本章で考察した，ユーザと対象システムの知的対話を支援するためのユーザモデルは，次のようにまとめられる．

ユーザモデル1

- 構成：ユーザの知識状態を表すヘルプ要素
- 利用：ユーザに対して，その時点でのユーザの知識状態にあった適切な内容のヘルプを提示するために参照される

ユーザモデル2

- 構成：一定期間t_i，その間に冗長な内部タスク系列を実行した回数n_{ij}，及びヘルプを出すと判断するための閾値c_i
- 利用：ユーザが冗長な内部タスク系列を実行したときに，より良い実行方法を提示するかどうかを判断するために参照される

4　実験及び考察

4.1　実験に使用したシステムの構成

次の三つのウィンドウから構成される．

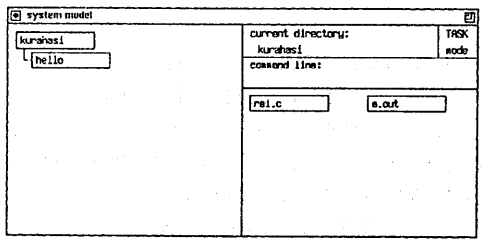
- 対象システムウィンドウ：
  対象システムモデルの内，内部タスク領域及びその状態を表示する（図 4·1）．内部タスク領域に関する質問は，このウィンドウの対応する部分をクリックするとヘルプが得られる．
- タスクメニューウィンドウ：
  対象システムモデルの内，タスク構造を表示する．タスクの実行方法に関する質問は，このウィンドウの対応する部分をクリックするとヘルプが得られる．
- コマンドウィンドウ：
  2.3.1節で述べた，メニュー実行されたコマンドに対応するコマンドラインを提示する．

質問に対する回答として出すヘルプの内容について，ユーザがすでに知っているとして提示されなかった内容も見たいという要求に対応するため，MORE ボタンを用意しておき，これを押せば以前に見た内容で提示されなかったものも，もう一度見ることができるようにする．

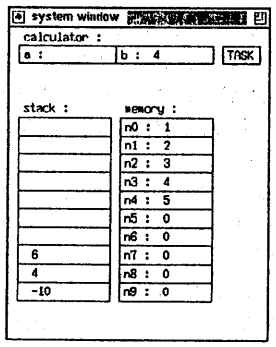
対象システムとして次に示す二つを用いて，二通りの実験を行った．

実験 1　対象システムとして UNIX を用い，タスクメニューには，図 3·1に示すプログラミングに関するタスク構造を用いた (システム 1)．
対象システムウィンドウには，内部タスク領域の構造及び状態として，ホームディレクトリより下のディレクトリ構造，及び，カレントディレクトリ内に存在するファイルが表示される．command line: とある領域にはユーザがキーボードからコマンドを入力できる (図 4·1(a))．
対象システムに関して多少の知識をもつ被験者に，簡単なプログラミング及び実行結果データファイルの整理をしてもらい，本システムを使用した感想を聞いた．

実験 2　対象システムとして新たに設計した算数計算システムを用い，算数計算を行うタスク構造をインプリメントした (システム 2)．
内部タスク領域は三つの領域，memory，calculator，stack から構成される．ユーザは演算数，被演算数を memory にセットし，そのうち必要なものを calculator にロードする．計算は一つの calculator でしか行えないので，計算結果を一時的に保持しておく場所として stack を用いる．システム 1 と異なり，ユーザがコマンドラインを入力してタスクを実行する場合，コマンドウィンドウに直接入力できる (図 4·1(b))．
計算機の使用経験がほとんどない人に本システムを使用してもらい，実験 1 と同じく感想を聞いた．



(a) システム 1



(b) システム 2

図 4·1　対象システムウィンドウ

4.2　ユーザモデルに応じたヘルプ機能に関する考察

4.2.1　ユーザの知識状態に応じたヘルプ内容

ユーザにとって必要のないものを提示しないという点については，実験 1，2 合わせてヘルプが参照されたのが計 83 回の内，必要なことが提示されなくて MORE ボタンが使用されたのが 20 回しかなかったことから，効果はあったことが分かる．特にコマンドの書式を説明するのは一度だけにして，次からは例を示すだけにしたのが分かりやすかったという意見があった．

MORE ボタンを使用した例としては，mv の二種の働きに関するものがある．mv にはファイル移動と名前変更との二種類の働きがあることを最初に示しても，ユーザはすべて見るわけでなく必要な部分しか見ない．そのため，次に見たときに例だけでは不十分だったということである．これについては，現在はヘ

ルプを見ればそのヘルプ要素が認知されたことにしているが，対応するタスクが実行されるまでは認知されないことにするなどの対処が考えられる．

また，今回はヘルプの内容自体については研究対象としなかったので，あまり考察せずに作った．これに対しては，本システムで作成した内容では不十分でもう少し詳しく教えてほしいという意見もあった．ヘルプについてのタスク構造は熟練者のものをベースにすべきだが[6]，熟練者は自分がどのように学習してきたかのかは覚えていないものである．したがって，それぞれのヘルプの内容については，熟練者である設計者がすべて作成するのでなく，実際にヘルプを使用する人の意見を聞いて作成すべきである．

4.2.2 タイミングを考慮した能動的ヘルプ

実験2においてh2)タスク実行支援が行われた様子を図4·2に示す．

特に次の二つのプランに対する能動的ヘルプの効果を考察する．

$I_1 = \{I_{11} = \{$'load a n1', 'load b n2'$\}$,
$I_{12} = \{$'load n1 n2'$\}\}$

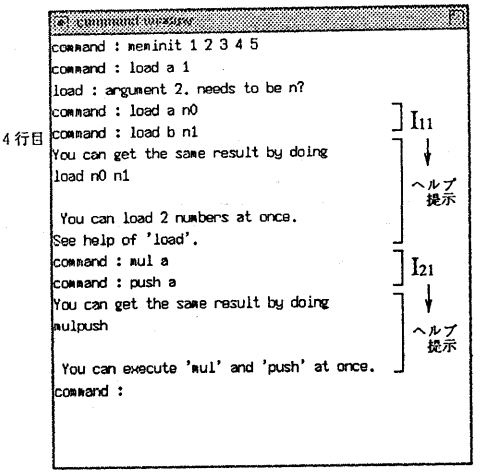
$I_2 = \{I_{21} = \{$'mul a', 'push a'$\}$,
$I_{22} = \{$'mulpush'$\}\}$

I_1は，「n1, n2の内容をそれぞれa, bにもってくる」プラン，I_2は，「a, bの内容を掛けて結果をstackにおく」プランを表す．

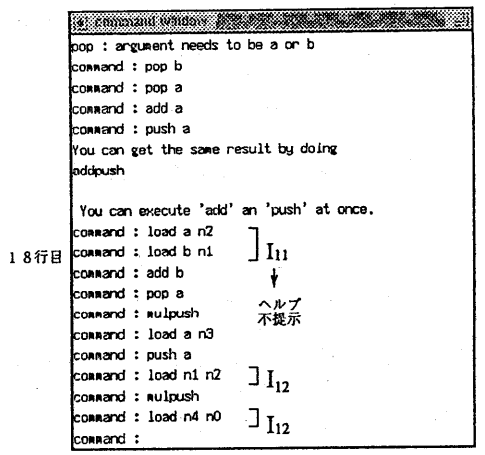
(a)では，I_{11}に対してI_{12}，I_{21}に対してI_{22}がそれぞれより簡単な実行方法として提示されている．(b)は(a)よりコマンドラインで15行程後のものであるが，この時点では，I_{11}を実行してしまったユーザに対してヘルプを出さずにいることが分かる．

また，図4·3に両プランに対する典型的な内部タスク系列の変化を示す．被験者の入力したコマンドラインX行目において，過去t_i行の間にI_{i1}及びI_{i2}が実行された回数Yが示されている．これによると，時間の経過とともに，プランI_iに対してより簡単な内部タスク系列I_{i2}の使われる頻度が高くなっているのが分かる．特にXが50行をこえたあたりからI_{i2}の実行回数が増えている．これは，実験の過程を見ていて，コマンド実行エラーが少なくなり，一つの計算にかかる時間も短くなってきたのとちょうど同じころである．

被験者の感想では，ヘルプに対して煩わしいと感じたことはなく，ヘルプが抑制されたことの効果が認められ，h3)エラー回復支援についても，エラーメッセージが不十分だと感じられたことはなかった．



(a) 結果例1（ヘルプ提示）



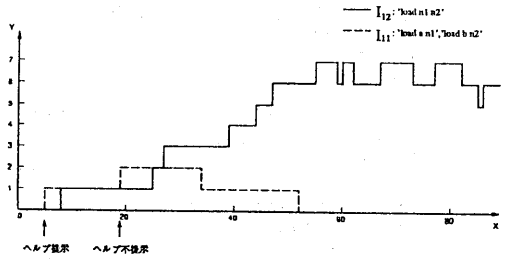
(b) 結果例2（ヘルプ抑制）

図 4·2 実験2の結果例

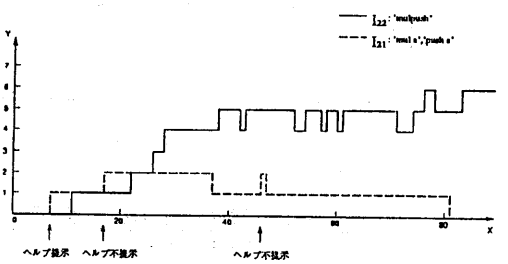
4.3 対象システムの視覚化に関する考察

実験1において被験者を，対象システムウィンドウを提示しメニュー実行を可能にしたグループ1と，対象システムウィンドウを提示せずメニュー実行を不可能にしたグループ2との二つに分けた．

グループ1の方がコマンド実行エラーが少なくなると予想された．しかし，実験1ではコマンド実行でエラーをした割合はほとんど変わらなかった．エラーの原因のほとんどがファイルのパス指定の間違いであった．今回は，タスク領域及びその状態は画面に表示し



(a) プラン I_1



(b) プラン I_2

図 4·3 典型的な内部タスク系列の変化

ただけで，それらを直接操作することはできないようにしていたが，それだけでなく，例えばマウスのドラッグアンドドロップで mv が実行できるようにしたり，パス指定も対話的に行えるようにすれば，いい結果が得られたであろう．対象システムモデルの表示方式を定式化することは難しいと考えられるので，これについては，学習支援システムを設計する段階で十分に考察する必要がある．

また，被験者は対象システムである UNIX について多少の知識があったので，タスク領域の構造（ここでは，ファイル・ディレクトリの構造を表す）を知っていたということも，あまり相違が見られなかった理由の一つであると考えられる．

実験 2 では，被験者は対象システムについて全く知識がなかった．はじめは対象システムを視覚化せずに使ってもらったが，このときは，課題として与えた算数計算が一つも実行できなかった．次に対象システムを視覚化すると，図 4·3のように学習がすすんでいった．(いずれの場合もタスクメニューは与えておいたので，タスクの実行方法に関する質問はできる状況であった．) 視覚化しない場合には，対象システムの内部タスク領域に関する質問をする方法がなかったということがこの理由として考えられ，視覚化する効果があったといえる．

また，画面にあるものをクリックするだけで質問できるという点については，分かりやすくてよかったという感想が得られた．

5 おわりに

本研究では，ユーザが操作対象システムと対話を行いながらその使用方法を学習するのを支援するシステムの実現に必要な要素として，対話過程で得られるユーザモデルの利用，及び対象システムの視覚化について考察した．

まず，ユーザと対象システムとの知的対話の重要性をあげ，それぞれにおいて対象システムがユーザに適切な応答をするために，ユーザの状態を表すユーザモデルを参照することを述べた．これに対しては，対象システムが提示するヘルプの内容や提示するタイミングについて煩わしく感じたことはなかったという被験者の感想が得られた．

また，対象システムを視覚化することによって，ユーザの対象システムに関する理解を支援することも考えた．ユーザに対象システムについての知識がない場合にはある程度の効果が見られた．

今後の課題としては，より長期的な実験を通して二つの要素の効果を調べることや，学習支援に必要な知識の自動抽出を行うことがある．

参考文献

[1] P. ジョンソン 著; 佐藤啓一, 宮井均, 須永剛司, 原田昭 訳: ヒューマンインタフェースの設計方法 (マグロウヒル出版, 1994).

[2] M. Hecking: How to Use Plan Recognition to Improve the Abilities of the Intelligent Help System SINIX Consultant, *INTERACT '87* (1987), 657–662.

[3] R. Wilensky, Y. Arens, and D. Chin: Talking to UNIX in English: An Overview of UC, *Commun. ACM*, **27-6** (1984), 574–593.

[4] 垣内隆志, 槇本英治, 上原邦昭, 豊田順一: ユーザモデルを利用した説明文生成プランニング, 人工知能学会誌, **4-2** (1989), 185–195.

[5] 上原龍也, 上原邦昭, 豊田順一: ユーザモデル上のシミュレーションによる質問意図の推定, 人知学会研究会資料, **SIG-HICG-8902-2** (1989).

[6] M. Neerincx and P. de Greef: How To Aid Non-Experts, *INTERCHI '93* (1993), 165–171.