特　集　「AIツールのユーザインタフェース」

ユーザインタフェースと認知モデル

User Interface and Cognitive Models

甲　洋介[*1]　安西祐一郎[*2]
Yousuke KINOE　Yuichiro ANZAI

*1 日本アイ・ビー・エム（株）東京基礎研究所
Tokyo Research Laboratory, IBM Japan, Ltd.
*2 北海道大学文学部行動科学科
Dept. of Behavioral Science, Hokkaido University.

1987年2月24日　受理

Keywords: user interface, cognitive model, mental model, human-computer interaction, user behavior.

1. はじめに

人工知能技術には，少なくとも2つの利用法がある．1つはできるかぎり知的な情報処理を行うシステムを構築することであり，もう1つはできるかぎりユーザにとって使いやすいインタフェースを作ることである．もちろんこれらは互いに関係があるが，これまでの人工知能研究は，どちらかといえば前者を志向してきたと考えられる．しかし，実際には，むしろ今後ユーザインタフェース技術への応用が重要になるであろうし，すでにこの方向への模索も，人工知能研究者自身によって始まっているといってよいだろう．

このような状況において今後求められるのは，ユーザインタフェースを，①インタフェース設計のための人工知能技術と②ユーザの認知情報処理，および③システムとユーザの間のコミュニケーションという3つの要因の関係としてとらえ，ハードウェア，ソフトウェアから人間の情報処理に至る総合的な視野のもとで，インタフェースの設計論を展開してゆくことである[1]．

本稿では，その中で特に「インタフェースにおけるユーザはどのような情報処理システムと考えられるか」という問題について，いくつかの議論を行うことにしたい．なお，この問題を考えるにあたっては，実際には人工知能技術と人間の認知情報処理機構についてのある程度の知識が不可欠である．

しかしながら，これらについてはすでにいろいろなところに書かれているので，あえて改めてここで議論することは避け，むしろ，こうした先行分野の研究であまり取り上げられてこなかった問題について述べることにしたい．

この意味で，本稿はこれまでの諸研究のいわゆる解説とは異なるものである．また，本稿では技術的問題にはあまり言及しないが，これは技術的問題が重要でないことを意味しない．むしろ，インタフェースの問題は結局は技術の問題であって，実現手段としての要素技術が発展しなければ，いくらユーザにとって使いやすいインタフェースの「お話」をしてみたところで意味がない．

しかしながら，インタフェース志向の研究が始まったばかりと考えられる現在においては，インタフェースにおけるユーザとはどのようなシステムであるのかを，インタフェースの設計に役立つ方向でもう一度とらえなおし，技術の方向づけを試みることも重要だと考えられる．

本稿はこのような趣旨のもとに書かれたことを，ご了解いただければ幸いである．

なお，インタフェースといっても，計算機からプラント，飛行機その他，きわめて広い範囲にわたる．これらの対象はそれぞれ異なる目的を持つものであり，インタフェースについての考え方も違ってくる．本稿でそれらのすべてを扱うことは不可能であり，ここでは計算機，しかも文書処理のようなごく日常的な目的のためのユーザインタフェースに話を限った．この点についてもご了解いただきたいと思う．

2. ユーザがタスクを遂行する過程

まず，インタフェースにおいてユーザがタスクを遂行する過程がどのようなものかを考えてみよう．

たとえば，ユーザが日常的に発生させるタスクは，報告書を作成したい，データを降順にソートしたい，ある研究分野に関する文献の著者とタイトルの一覧表を手に入れたい，ある商品の売上げ高の推移をグラフで検討したいなど，さまざまなものが考えられるだろう．このとき，まずユーザは，そのタスクに適したシステムを選択することから始めることになる．

ここでは，あるタスクが発生してからユーザがそのタスクの実行結果（途中の状態も含む）を確認するまでを，ユーザが1つのタスクを遂行する過程としてとらえることにする．

ユーザとシステムの間で展開されるインタラクティブな過程のモデルとして，これまでに文献（4），(11)，(13)，(14)，(17) などがある．ここでは，ユーザがタスクを遂行する過程を，システムと直接的にインタラクトを行っている部分とそうでない部分に分けて考え，前者をユーザとシステムのコミュニケーションのレベル，後者をユーザの問題解決のレベルと呼ぶことにする．

2・1　コミュニケーション・レベル

ユーザとシステムのインタラクションを考えると，

① ユーザからシステムへの方向

② システムからユーザへの方向

の2つの方向がある．インタラクションにおいて行われているユーザ，およびシステムの活動 (activity) は，それぞれの方向に対応させて次のように想定することができる．

まず，ユーザからシステムへの方向を考えよう．

ユーザは，タスクの実行をシステムに依頼するために，実行すべきタスクの内容をシステムに対して表現し，説明する必要がある．この説明は，たとえばコマンドやアイコンの選択などによって表現される．またシステムは，この表現から実行すべきタスクの内容を解釈しなければならない．

次に，システムからユーザへの方向を考えよう．

システムは，タスクを実行し，その結果をユーザに対して表現し，報告する必要がある．またユーザは，この報告に基づいてタスクの遂行結果を解釈しなければならない．この報告は，たとえば文字，記号，グラフィクスなどの画面表示，音声などによって表現

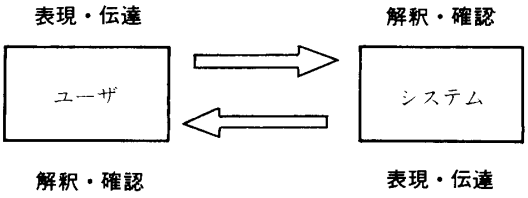


図 1　ユーザとシステムのインタラクション

される．

以上のことからも示されるように，ユーザとシステムの間のインタラクションにおいては，「表現の過程」と「解釈の過程」がユーザとシステムの双方に存在し，一方が表現・伝達した情報を，次の瞬間には他方が解釈・確認するという構図になっていることがわかる（図1）．

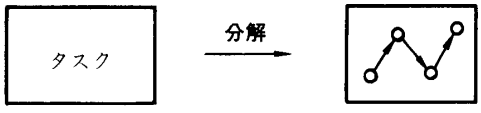
このような意味において，ここで展開されるインタラクションは，コミュニケーションの1つの形態としてとらえることができる．また，それを成立させるためには，タスクやその実行結果を表現し解釈するための体系（言語体系といってもよい）が，インタラクションのそれぞれの方向について必要となる．

2・2　ユーザの問題解決過程

ユーザは，最終的には発生させたタスクをシステムが取り扱うことのできる操作の組合せによって表現しなければならないが，タスクはユーザが使用しているシステムとは独立に生じるものであり，発生したタスクとそれを遂行するための解決方略との間には，本来，ギャップが存在する．したがって，ユーザはシステムに対して表現・伝達をする前に，そのタスクがシステムの用意するどの操作によって実現可能なのかを考え，システムが取り扱い可能な操作の単位にタスクを分解させながら，その構造を明らかにしていくことが必要となる（図2）．

ここではこのようなタスクの変換過程を，ユーザの「問題解決過程」と呼び，先に述べたコミュニケーションのレベルと切り離して考えることにする．

これにより，ユーザがタスクを表現できない状況が発生したとき，それが表現・伝達の段階にその原因がある場合と，それ以前に解決方略を生成させる段階に



「システムの操作体系によって
構成されるタスクの構造」

図 2　ユーザによるタスクの分解

原因がある場合とを区別して取り扱うことが可能となる.

2・3　タスクを遂行する過程(例)

ユーザがタスクを遂行する過程を考えるために，これまでに「タスクを表現・伝達する過程」,「タスクの遂行結果を解釈・確認する過程」,「問題解決過程」の3つの過程を設定した．これらの過程は完全に独立しているというよりは，むしろ相互的に影響し合っていると考えたほうがよい.

さてここで，ユーザがタスクを遂行する簡単な例を示し，上で設定した各過程に基づきながらこの例を考えていくことにしよう.

〔1〕　タスクの発生と認識

まず，ユーザによってあるタスクが発生する.

《例》「昨日作成した報告書で，単語“ZZ”のスペルを全部“XX”に間違えていたので，それらを“ZZ”に訂正した報告書を別に作りたい」

〔2〕　タスクの問題解決過程

発生したタスクを分析し，その構造を明らかにする．タスクを遂行するためのプランニングを行い，その結果として，タスクの遂行に必要な操作の系列(解決方略)を生成する.

《例》「まずエディタに入りファイルAを呼び出す．テキスト中の“XX”を探して，全て“ZZ”に置き換える．ファイル名Bで保存し，エディタから抜ける」

〔3〕　解決方略の表現・伝達過程

問題解決過程で組み立てられた解決方略を，システムが提供する表現体系(たとえば，コマンド，アイコン)によって表現し，入出力デバイス(たとえば，キーボード，マウス，マイクロフォン，タブレット)によって伝達する.

《例》「EDIT“A”/R“XX”,“ZZ”*/S“B”/“Q”をそれぞれキーボードでタイプし，ENTERキーを押す」

〔4〕　状態，結果の解釈・確認過程

システムが提示するタスクの実行結果(たとえば，実行後のテキスト表示)や状態の報告(たとえば，メッセージ，プロンプト)を解釈し，ユーザ自身が要求していた結果との比較，確認をする．一致していれば，それをタスクの成功とし，一致していないのであれば，その結果を目標の状態に一致させる，新たなタスクがここで発生する.

《例》「“error occurred. invalid command”」

以上のように，〔1〕,〔2〕,〔3〕,〔4〕項の各過程を仮定することによって，ユーザのタスク遂行過程が理解しやすくなることがわかるだろう.

2・4　ユーザとシステムがタスクを遂行する過程

システムがタスクを遂行する過程は，システム側に問題解決過程を仮定することによって，ユーザのタスク遂行過程と同様の枠組で考えることが可能である．そこで，ユーザがタスクを遂行する過程とシステムがタスクを遂行する過程を，双方向のインタラクションによって結ぶと，図3に示すような図式が得られる．この図式を，仮に「タスク伝達/変換モデル」と呼ぶことにする.

さて図3におけるタスクのフローに注目することにしよう．ユーザが発生させるタスクとシステムが遂行するタスクは,「表現過程」と「解釈過程」を一組と

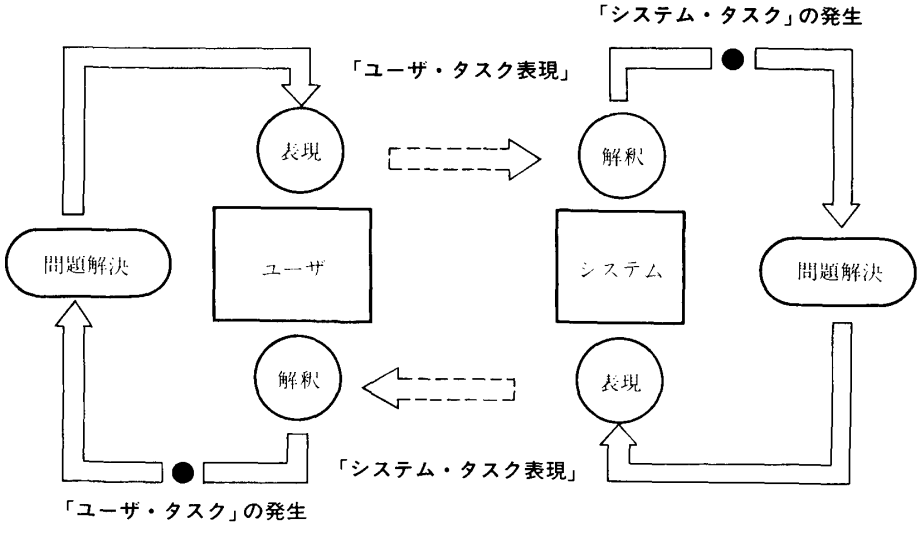


図3　「タスク伝達/変換モデル」によるタスク遂行の過程

するコミュニケーションの過程を経て結ばれている．この2つのタスクを分けて，それぞれ「ユーザ・タスク」，「システム・タスク」と呼ぶことにする．「ユーザ・タスク」はユーザ自身の要求によって発生するが，「システム・タスク」は，システムが「ユーザ・タスク表現」を解釈した時点で初めて発生する．

次に，これらの2つのタスクが，ユーザ・システム間を伝達される状態を考える．ユーザがシステムに対して，「ユーザ・タスク」を伝達するために表現したものを「ユーザ・タスク表現」と呼び，システムがユーザに対して，「システム・タスク」の遂行結果を伝達するために表現したものを「システム・タスク表現」と呼ぶことにする．

それぞれのタスク表現を2・3節に示した例で考えれば，コマンド表現は「ユーザ・タスク表現」に相当し，システムからのエラーメッセージは「システム・タスク表現」に相当する．一般的にいってタスク表現は，何らかの意味ある記号列によって構成されており，文字，図形，グラフィクスなどの視覚情報，音声などの聴覚情報など，さまざまな媒体を想定することができる．

3．ユーザインタフェースによる制約

次に，ユーザがインタフェースから受ける制約について考えてみよう．

ユーザインタフェースは，図3におけるユーザとシステムとの直接的なインタラクションの部分に関与しており，それが提供する表現体系や入出力デバイスによって，「ユーザ・タスク表現」および「システム・タスク表現」の形態を規定していると考えられる．その結果として，ユーザのタスクを遂行する各過程は，さまざまな形で制約を受けることになる．

ここでは，ユーザが受ける制約の内容とそれによって引き起こされる問題を，①ユーザ・タスクの発生からユーザ・タスク表現を生成するまでの流れと，②ユーザにシステム・タスク表現が伝達されてから新しいユーザ・タスクを発生させるまでの2つの流れに沿って検討していくことにする．

3・1　ユーザのタスク遂行における制約(1)

ユーザ・タスクの発生からユーザ・タスク表現が生成されるまでを考えることにする．ここでのユーザの目的は，発生させたユーザ・タスクを忠実にシステムに伝達することである．しかし，そのタスクがどのような形で発生したにせよ，ユーザは，最終的にはそれをユーザインタフェースが規定するユーザ・タスク表現の形態で表現しなければならない．この制約により，ユーザの表現・伝達過程，および問題解決過程において，次のような問題状況が発生する．

〔1〕 表現・伝達過程における問題

この過程において問題となるのは，ユーザが明確な解決方略を生成させることができたとしても，それを表現することができない状況が発生することである．たとえば，コマンドを思い出せない，適切なアイコンを弁別できないなどの，解決方略の表現方法（いわば翻訳方法）に関するケース，間違ったキーを押したりマウスのクリックを誤ったなど，入力デバイスの操作方法に関するケースなどが考えられる．ユーザは，これらの方法のいずれを知らない場合でも「エラー」を引き起こすことになるが，もしこのユーザに対して「あなたは何をしたいのか？」と問うことができるなら，彼の日常使っている自然言語によってタスクの解決方略を説明することができるだろう．

このように，解決方略を明確に持っていても表現することができないという状況は，ユーザ・タスク表現が制約されていることによって発生し，また，ユーザのストレスの原因ともなりうる．

〔2〕 問題解決過程における問題

この過程において問題となるのは，ユーザが仮に明確なユーザ・タスクを持っていたとしても，それを達成するための解決方略を（システムの操作体系に基づいて）生成することができない状況が発生することである．初心者のユーザであれば，操作体系についての知識が乏しいために，このような状況が頻発することは容易に予想できる．

しかし，たとえあるシステムに熟練したユーザであっても，全く別の体系を持つ他のシステムでタスクを遂行する場合を考えれば，やはり同じ状況が発生するのである．

ユーザ・タスクは，もともとユーザが利用しようとしているシステムとは独立に発生するものであることを先に述べた．したがって，与えられた操作体系とは全く整合しないような解決方略を生成することもありうる（図4）．しかし生成された方略のうち，最終的にユーザ・タスク表現として表現可能なのは，システムの操作体系に従って組み立てられた方略（図4：方略A）に限定され，操作体系に整合しない方略（方略B）は棄却される．このことは，問題解決過程においてユーザが利用できるオペレータ群の体系が，システムの提供する操作体系によって厳しく制限されていることを意味する．

このことから，ユーザインタフェースがユーザ・タ

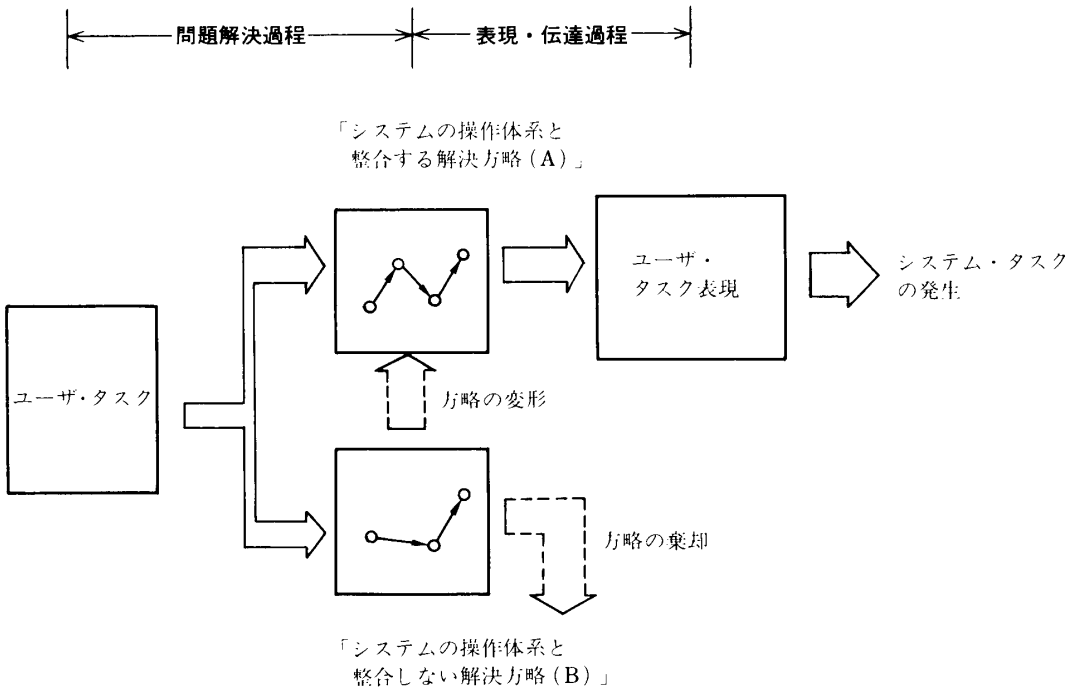


図 4 ユーザによる解決方略の生成とシステムの操作体系との整合性

スク表現の形態を規定することによって，ユーザの問題解決過程までもが制約されていることがわかる．したがって，システム設計者は，システムの操作体系を設計する際に，このことを十分留意しなければならない．

3・2 ユーザのタスク遂行における制約（2）

次に，システム・タスク表現がユーザに伝達されてから，ユーザによって新たなユーザ・タスクが生成されるまでを考えることにする．このとき，まず，システムがシステム・タスクを遂行した結果をシステム・タスク表現としてユーザに伝達する．ユーザは，システム・タスク表現を解釈・確認する過程において，タスクの達成，失敗の判断をすることになる．

〔1〕 解釈・確認過程における問題

この過程で問題となるのは，ユーザがシステム伝達表現に基づいてタスクの遂行結果を解釈できない状況が発生することである．

その原因として，次の2つを考えることができる．まず1つは，システム・タスク表現がユーザに伝達できるのは，実行結果のある限定された一部分についてであり，ユーザはその限られた情報に基づいてタスクの遂行結果を推論しなければならないことである．たとえば，システムからのエラーメッセージは，ユーザに提供すべき情報のごく一部分であり，そこからエラーの原因を究明し，それを解決するための新たなタスクをユーザが生成するためには，何段階にもわたる推論プロセスを必要とする．

もう1つは，システムから伝達されたタスク表現の形態とユーザの要求する形態とが一致しない場合である．たとえば，ユーザがある商品の売上げ高の推移を検討しようとするとき，膨大な数値データのリストが出力されても，ユーザがタスクの達成を解釈することは不可能である．あるいは可能であっても，自分の要求する形態に変換する（たとえば，グラフ化）ために大変な認知的プロセスを必要とする．このように，ユーザに提示されるシステム・タスク表現の形態が，ユーザのタスク遂行結果を解釈・確認する過程において重要な意味をもっていることは明らかである．

4．認知モデル

さてインタフェースの問題を論ずるときには，「認知モデル」という概念がしばしば用いられる[9][10]．ここでは，ユーザが主体的にシステムとインタラクトする状態に注目し，ユーザのタスク遂行過程と，この概念との関係を検討しながら，ユーザインタフェースの問題を考察する．

4・1 ユーザのタスク遂行と認知モデル

いま，あるタスクを遂行しようとしている初心者のユーザが，ディスプレイ，キーボード，マウスなどを前にして呆然としている状況を想像してみよう．このとき，ユーザにとっていったい何が問題となっている

のだろうか．この問いに対する答として，タスクの遂行過程に対応させて少なくとも以下の〔1〕,〔2〕項の2つを考えることができる．

〔1〕 いったい何をすればよいのか？

ユーザは，ユーザインタフェースからどんな制約を受けているのかを把握することができない．したがって，タスクを達成するために何をしたらよいのかわからない．

このような状況にユーザがおかれた場合には，タスクの問題解決，方略の表現・伝達など，タスクの各遂行過程を全く行うことができない事態が起こりうる．たとえば，そのタスクを達成するためにどのような操作群が用意されており，それぞれの操作がどのような状態のときにどのような対象に対して適用できるのか，またその結果，どのような状態変化をもたらすのか等々，操作体系について考えるだけでもさまざまな制約を挙げることができる．

これらの制約が原因となって，ユーザの各過程に多くの問題が発生することはすでに述べた．しかし，初心者のユーザであっても，試行錯誤を繰り返しながら，あるいは提供されたマニュアルを参照しながら，タスクを次第に巧みに達成することができるようになる．

このようにユーザは主体的にシステムとインタラクトしながら，実際にインタフェースから負荷されている制約を「ユーザなりに」とらえていく．このとき，ユーザがとらえている制約の様式をユーザの「認知モデル」と考えることができる．

〔2〕 いったい何が起こったのか？

ユーザがシステムに対して，ユーザ・タスク表現を制約に従って伝達したとしても，それがシステムによってどのように解釈され，遂行されているかがわからない．

たとえば，初心者のユーザが恐る恐る入力したコマンドが，要求したとおりに解釈され，タスクが実際に達成されることを確認すると，ユーザは次第にタスクの遂行方法についての因果的手続きを獲得していくものと考えられる．また，ユーザは，この獲得した因果的手続きに基づいてシステムによるタスクの遂行結果を予測できるようになる．このような過程を経て，また，さまざまなタスクを達成することによって，ユーザは，ユーザインタフェースから受ける制約の上で起こるインタラクションに対して信頼感を持つようになる．たとえば，「システムに対してこのように表現すれば，こうなるに違いない」という確信である．

この段階に入ると，ユーザの形成している認知モデルは，すでに，単なるシステムから受ける制約についての知識ではない．ユーザ・タスクの表現・伝達からシステム・タスク表現が伝達されてくるまでのプロセスを，ユーザのとらえるシステムの機能的な単位に基づいてシミュレートする機能を果たすようになる．たとえば，タスクの遂行中に発生したエラーの原因をユーザが推論する際には，この認知モデルによるシミュレーションが，きわめて重要な役割を果たしているものと考えられる．

システムから受ける制約の様式を認知モデルの静的な側面であるとすれば，上に述べたタスク遂行過程のシミュレーションは，認知モデルの動的な側面であり，モデルの持つ最も基本的な機能であると考えることができる．

なお，上では触れなかったが，認知モデルはユーザが個々に形成するものであり，あるシステムを利用しているユーザ全体に対する平均的な認知モデルというものは，本質的には存在しないことに注意されたい．

4・2 認知モデルの変容

ユーザがタスクを遂行する過程と認知モデルの関係は，上のようなものだと考えられるが，それでは認知モデル自体の変容はどのようになされるのだろうか．今度はこの問題について考えてみよう．

ユーザは，自分のとらえている制約の様式（認知モデル）に基づいてシステムを操作し，いろいろなタスクの遂行に際してそのモデルを適用する．ところが，ユーザの形成している認知モデルは，ユーザインタフェースが設定した制約全体を網羅しているわけではなく，またモデル内に多くの矛盾を含んでいるものであると考えられる．したがって，そのモデルの適用に際しては，反例（エラー）に遭遇することもあるし，また，逆に新たな適用対象を発見することもある．

認知モデルの変容は，これらを契機にして引き起こされるものと考えられ，また，ユーザが認知モデルを形成する過程は，絶えずその変容を伴って進行すると考えられる．ただしその変容は，すべてのタスクの遂行にとって適切な方向に向かうとは限らない．

また，認知モデルが変容するのは，単にユーザがシステムとインタラクトすることによって誘発的に起こる現象ではない．ユーザの目的は，ユーザ・タスクを遂行することにあり，ユーザは，あくまでもこの目的を達成するために能動的に認知モデルを変容させていくのである．このような認知モデルの変容過程を，ユーザのシステムの制約に対する「適応」としてとらえることが可能であろう．

なお，一般に，ユーザはシステムから負荷される制約に対して驚くばかりの適応を示す．たとえば，人工言語であるコマンドを使いこなすためには，実際には人間の強力な言語獲得能力や記号運用能力が利用される[12]．逆にいえば，ユーザが潜在的に持つこれらの適応能力が非常に優れているために，ユーザインタフェースの問題が長い間放置されてきたのだ，と考えることもできるだろう．

また，ユーザが，ある一つのシステムを経時的に使用することによってモデルを変容させていく場合のほかに，あるシステムから他のシステムに移行する場合にも，ユーザの認知モデルは変容する．ユーザはこのときも，まず新しいシステムの制約を把握することからスタートすることになるのである．

4・3　設計モデルとの整合性の問題

これまではユーザの認知モデルについて述べてきたが，インタフェースを考えるにあたっては，システム設計者についても考慮する必要がある．

システム設計者は，ユーザインタフェースを設計する段階で，ユーザに対して与える制約を何らかの形で認識していると考えられる．これを「設計モデル (design model)」と呼ぶことにする[14]．

これまで，システム設計者の主な関心は，設計者自身が持つ設計モデル内での整合性や無矛盾性に向けられ，ユーザがそのシステムに対してどのような認知モデルを形成するかについては，大きな関心を払ってこなかった．このような状況の根底には，ユーザの起こしたエラーの原因を，インタフェースに設定された制約に対してではなく，ユーザ側の使用方法や学習不足に求める姿勢があったと考えられる．

設計者にとっての理想は，ユーザがこの制約の様式をそのままとらえてくれることであるが，問題は，ユーザがシステムに対して実際に形成するモデルと設計者が持っている概念モデルとは，システム設計者の期待するようには一致しないのがふつうだということである．逆にこの場合，ユーザによって全く別の認知モデルが形成されてしまうことが多いと考えられる．

現在では，このような問題点が指摘され，システム設計者にとっても，その目標の中に，ユーザがタスクを適切に遂行することが含まれるとすれば，ユーザがいかなる認知モデルを形成するかについて関心を向けざるをえない状況になってきている．そして，このような関心を持つエキスパートとしてのユーザインタフェース設計者の存在が必要になりつつあると考えられる．

5．ユーザの受ける制約のメタファーによる緩和

上では，インタフェースがユーザに与える制約としての認知モデルについて述べてきたが，次の問題は，このような制約を緩和し，より使いやすいインタフェースをユーザの面から考えるにはどうしたらよいかということである．ユーザインタフェース研究で以前から問題となっているメタファーは，このような制約緩和の方法の1つだと考えることができる．ここでは，このような観点から，インタフェースにおけるメタファーの役割について考えることにする．

5・1　メタファーの果たす役割

問題解決やある事柄の理解にメタファーやアナロジーを用いることのメリットとして，既存の知識構造との対応関係をつけることによって，理解が促進されたり解決の困難な問題の解決策を導き出すことが可能になったりすることがよく知られており[2][3][6][7]，ユーザインタフェースにおけるメタファーの問題を扱った研究としては文献(5)などがある．この点に関連して，ユーザインタフェースの構成要素，要素間の関係，各要素に対してなされる操作に関するメタファーを考えてみよう．

いま，メタファーのよく知られた適用例として，「ユーザ間の情報の伝達」を「郵便」としてとらえた電子メールのメタファーの例を考える．

このメタファーを与えることにより，構成要素(たとえば電子メールとメールボックスを考える)，要素間の関係，操作についての対応関係がうまく形成され，ユーザが理解すべき領域(「情報伝達」: target domain)に設定されるべき制約は，ユーザが既によく知っている領域(「郵便」: source domain)に設定されている制約を参照することによって，スムーズに決定される．すなわち，「メールボックスは電子メールを保管するものであり(要素間の関係)」，「受け取るという操作は，メールに対して適用され(操作)」，さらに「この操作を適用した結果，メールはメールボックスに保管される(結果の予測)」となる．

ユーザインタフェースを設計する際に，このような優れたメタファーを利用した場合には，ユーザの問題解決のために用意すべき操作体系や，各操作の組合せ方法，表現・伝達方法などに関してインタフェースがユーザに負荷する制約を，ユーザにとって既知である制約とうまく整合させることができる．この場合で

あっても，ユーザには確かに何らかの制約が負荷されているのであるが，それに対してユーザが適応する必要はあまり生じないと考えられる．

5・2　メタファーの限界

メタファーの最大の効用の一つは，対象を理解するための面を明確に切り出してくれることである．しかし一方では，入門的なマニュアル類によくみられる「ワープロとタイプライタ」，「ディスクドライブとテープレコーダ」などの例が示すように，ユーザが対象に対して得ることのできる理解が限定的になりがちであるという弱点を持つ．この結果，単一のメタファーによる対応関係だけでは説明できない不足部分がどうしても残されることになる．

また，メタファーによる理解では，上で指摘した限界のほかに，強引な対応づけ（strong analogy）によってユーザを誤った推論に導く危険性も存在する．このことは，理解すべき対象との対応関係をつけることができても，メタファーを適用すべきでない範囲が存在することを示している．

したがって，ただ単にメタファーを与えればよいといった盲信的なメタファー崇拝は，ときに危険な結果をもたらす可能性があることを認識しておく必要がある[8]．

さて，上に指摘したメタファーの限界を克服する方法としては2つのアプローチを考えることができる．1つは，部分的に最適であるような複数のメタファーを用意し，目的に応じて多重的に使い分ける方法である．もう1つは，単一なメタファーで説明可能な範囲内に，システム側の実行可能な作業を収めてしまう方法である．第2のアプローチで注意しなければならないのは，システムに設定可能な操作体系の範囲がユーザの既知の世界によって制限される可能性があることである．これでは，タスクの遂行をシステムによって実現するメリットを大きく損なう結果となる．

6．ユーザインタフェースのめざす1つの方向

結局，これまで述べたところによれば，ユーザインタフェースの問題は，ユーザとシステムに対してどのような制約を設定するかということに帰着される．この点を中心として，ここでこれまでのことをまとめることにしよう．

まず，ユーザとシステムの間で起こりうる適応には2つの方向：「ユーザの制約⟷システムの制約」があり，これまではユーザがシステムの制約に対して適応することが一方的に期待されてきた．ユーザに対して何らかの制約を負荷することは，タスクの遂行を効率的に行うためにある意味で必要なことであるが，ユーザの人間としての適応能力には限界があり，ユーザに対する制約を強化することによりストレスを誘発する可能性もある．一方，システムには技術的な制約を常に伴うが，ユーザの制約に比べ変更が比較的容易であり，また，技術的な進展による改善を期待することができる．

したがって，ユーザに負荷する制約をなるべく最小に抑えることをユーザインタフェースの目指す目標の1つとして考えることが妥当である．そしてそのためには，まず原点を「ユーザ」に設定し，ユーザ自身の持つ制約を把握することから始めるという方法を考えることができる[15]．このとき，システムの側からユーザの制約に対して適応する方法として次の2つが考えられる．すなわち，システム設計者がユーザの持つ制約を把握し，その知見をユーザに負荷する制約の設定に反映させていく場合と，ユーザ側の持つ制約をシステムがユーザとのインタラクション過程から推定し，ユーザに対して負荷する制約を自律的に変容させていく場合の2つである．前者のアプローチは後者を実現する上でもその基盤をなす研究であると考えることができ，これらの研究は技術開発と平行して進めていく必要がある．

7．おわりに

本稿では，ユーザインタフェースと認知モデルの問題について，いくつかの考え方を述べた．その中でも特に，インタフェースから生まれる制約を認知モデルであるとみなすこと，したがってインタフェースの研究にはユーザ自身の持つ制約を把握することが重要になるという点が，中心的な問題であった．

はじめにも述べたように，ユーザインタフェースの研究は，特にわが国では端緒についたばかりであり，インタフェースをどうとらえるか，ユーザをどうとらえるかということすら確定していないのが現状である．また，使いやすいインタフェースの開発にはもちろん要素技術の進歩が基本的に必要であるが，それだけでなく，ここに述べたようなユーザについての研究，特にユーザとタスクの動的関係についての研究も，並行して進めるべき大切な方向であるように思われる．

ユーザインタフェースについて，本稿では触れることのできなかった問題は数多くある．たとえばユーザに関する事柄に限ったとしても，ユーザの分類（熟練

者と初心者，知識，スキル，使用頻度による相違），負荷された制約に対するユーザの適応過程の把握，ユーザの初期状態（適応を起こす前の状態）の把握，ユーザの持つ推論，認識，言語，運動などの能力，およびそれらの能力の限界の把握などは，ユーザインタフェース設計のための基礎研究として重要な問題である．また，これらをインタフェース設計の基本的な問題として位置づけし，解決するための方法論を開発することは，大切な課題として残されている（なお，ユーザインタフェースにおける基本的な問題に言及したものとして文献(16)，主なトピックの概説として文献(18)などがある）．

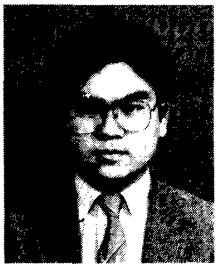
さらに，本稿では大規模な事柄について述べることができなかったが，インタフェースの研究にあたっては，いたずらに抽象論に走ったりごく小規模な実験での議論に終わることなく，その目標として現実的な問題の解決を明確に含めておくことが，研究の進展にとってきわめて重要である．

以上の点については，別の機会に稿を改めて述べることにしたいと考えている．

◇参 考 文 献◇

(1) 安西祐一郎：知識情報処理と認知工学，電子通信学会誌，Vol. 69, No. 11, pp. 1099-1106 (1986).
(2) 安西祐一郎，甲 洋介：類推による問題解決における図の効果，日本認知科学会第1回大会発表論文集，pp. 52-53 (1984).
(3) Carbonell, J. G. : Metaphor : An inescapable phenomenon in natural-language comprehension, In W. G. Lehnert and M. H. Ringle (eds.), Strategies for natural language processing, pp. 415-434, LEA, Hillsdale, NJ. (1982).
(4) Card, S. K., Moran, T. P. and Newell, A. : The Psychology of Human-Computer Interaction, LEA, Hillsdale, NJ. (1983).
(5) Carroll, J. and Thomas, J. : Metaphor and the cognitive representation of computing systems, IEEE Transactions on Systems, Man and Cybernetics, Vol. 12, pp. 107-116 (1982).
(6) Gentner, D. and Gentner, D. R. : Flowing Waters or Teeming Crowds : Mental Models of Electricity, In D. Gentner and A. L. Stevens (eds.), Mental Models, LEA, Hillsdale, NJ. (1983).
(7) Gick, M. L. and Holyoak, K. J. : Analogical problem solving, Cognitive Psychology, Vol. 12, pp. 306-355 (1980).
(8) Halasz, F. and Moran, T. : Analogy concidered harmful, Proc. of the Conference on Human Factors in Computer Systems (1982).
(9) Proc. of the Conference on Human Factors in Computer Systems, Gaithersburg (1982).
(10) Proc. of ACM CHI '83 Conference on Human Factors in Computing Systems, Boston (1983).
(11) Kieras, D. and Polson, P. G. : An approach to the formal analysis of user complexity, Int. J. of Man-Mach. Studies, Vol. 22, pp. 365-394 (1985).
(12) 甲 洋介：コマンドの認知科学的研究，第34回情報処理学会全国大会論文集，pp. 1523-1524 (1987).
(13) Norman, D. A. : Stages and levels in human-machine interaction, Int. J. of Man-Mach. Studies, Vol. 21, pp. 365-375 (1984).
(14) Norman, D. A. : Cognitive Engineering, In D. A. Norman, and S. W. Draper (eds.), User Centered System Design : New Perspectives on Human-Computer Interaction, LEA, Hillsdale, NJ. (1986).
(15) Norman, D. A. and Draper, S. W. (eds.) : User Centered System Design : New Perspectives on Human-Computer Interaction, LEA, Hillsdale, NJ. (1986).
(16) Reisner, P. : Human Computer Interaction : What is it and what research is needed, In J. M. Carroll (ed.) : Interfacing Thought : Cognitive Aspects of Human Computer Interaction, MIT Press (in press).
(17) Shneiderman, B. and Mayer, R. : Syntactic/Semantic interactions in programmer behavior : A model and experimental results, Int. J. of Computer and Information Sciences, Vol. 8, No. 3, pp. 219-239 (1979).
(18) Shneiderman, B. : Designing the User Interface : Strategies for Effective Human-Computer Interaction, Addison-Wesley (1987).

著 者 紹 介



甲 洋介

昭和59年慶応義塾大学工学部管理工学科卒業．昭和61年同大学院工学研究科管理工学専攻修士課程修了．同年日本アイ・ビー・エム(株)入社．現在，同社・東京基礎研究所・認知工学グループ所属．認知心理学，言語学，生体情報処理などに興味を持っている．日本認知科学会，情報処理学会各会員．

安西祐一郎(正会員)

昭和49年慶応義塾大学大学院博士課程修了．工学博士．昭和56～57年カーネギーメロン大学客員助教授．慶応義塾大学講師を経て，昭和60年より北海道大学文学部行動科学科助教授．計算機科学，人工知能，認識の情報処理過程の研究に従事．著書「知識と表象」(産業図書)ほか．情報処理学会，日本ソフトウェア科学会，ACMなどの会員．