福山大学工学部紀要
第8号 1986年4月

建築教育におけるパーソナル・コンピューター利用の可能性について（３）（ＩＣＡＩプログラムの一般的機能条件）

谷口興紀*・横井友幸*

A Note on CAI in the Architectural Education (3)

—General Condition of ICAI Program—

Okinori TANIGUCHI and Tomoyuki YOKOI

ABSTRACT

This note discusses the general conditions which ICAI programs have to satisfy. They are clarified by next processes;

(1) Stenius's thinking model is applied to the architectural thinking processes,

(2) in the instruction of architectural design as a phenomenon of real world, one exercise is analyzed by predicate logic,

(3) a simple CAI program (described by Prolog language) was used for teaching structural mechanics as a sample and its function is considered.

The conditions are summarized in next three points;

(i) ICAI programs have to store sequential inputs of a student and the relation-keys for apprehending them resolved by the computer,

(ii) ICAI programs have to classify the trace of his inputs and the relation-keys by analyzing data obtained by step (i),

(iii) from data (i) and data (ii) ICAI programs have to infer the relation-keys for the apprehension of his inner articulate field.

序

最近，エキスパートシステムの話題が，パソコン雑誌等を賑わせており，その核となるＡＩ技術が，種々の分野で応用されようとしている。我々は，過去３年間，建築教育に即したＩＣＡＩシステムの構築を目指して，[1] ある意味で試行錯誤的に作成したプログラムを，実際の授業に使用して来た。そのような実践的試みと理論的研究との相互の交渉によって，朧げながら，あるべきＩＣＡＩプログラムの一つの輪郭が見えてきた。そこで，本稿では，ＩＣＡＩプログラムの備えるべき条件について考察する。

その輪郭の形式的側面は，「教育する」という観点からは，「建築」教育には限らず，他の分野においても，同様な形式が当はまるであろう。したがって，他分野においても，そのような方向への取り組みが始められ，その成果の建築教育へのフィードバックを期待したい。

*建築学科

1. 思考活動のモデル

「Intelligent」という語を付けるかぎりは，仮説的にしろ「Intelligent」とは何かという規定が明示されねばならない。そのため，ここでは，以下で述べるように思考活動を，規定しておく。すると，そのような思考活動を提するものは，「intelligentである」という印象を与えるだろう。思考活動のモデルとしては，ここで述べるものが唯一絶対というものではない。もし，ここで述べられるものよりすぐれたものが出てくれば，それと取り替えることは，やぶさかではない。

Steniusは，〝考えることは，現実の適切な ― 真または偽な ― 像の形成である〟という。そして，〝文は現実の像である〟というWittgensteinの「論理哲学論考」の中の一つの命題の解釈を示す。ここでは，Steniusの「思考」モデルを簡単に説明しよう。[2]

教育という場面では，言葉（文）を媒介にしてコミュニケーションが行なわれることが最も一般的である。考えることや考えたことも文で表現されてはじめて他人にそのことを明示することができる。文は，述語論理的に論理的主語と論理的述語とに分析されるものと，そうでないものとに分けられる。前者の典型は，数学における文である。後者の典型は，詩である。詩においては，論理的主語や論理的述語にその文を分析してみても，殆んど無意味である。本来的に，そのような意図で語られているわけではない。一方，数学的文は，論理的主語や述語に分けられない文は，不正確であったり，答えが一義的に定まらないということになり，数学的文としての役割を果さない。数学の文だけでなく，数学の体系を，論理的主語・述語とそれらの記号化と論理結合詞によって述べる ― もちろん，公理体系と推論規則を定めてであるが ― ことが，Whitehed-RussellによるPrincipia Matematicaによって周知の如く示されている。

建築教育においては，工学部という学問分野の性質上詩よりは，数学的文の性格をもつ文が使用されると言えよう。とは言っても，たとえば，設計製図において，これから設計しようとするもののイメージについて語るような場合には，数学的文というよりは，詩的文の性格をもつ表現が使用される場合もある。このことは，ここでは，指摘するだけにとどめ，以下では，専ら，数学的文の性格をもつ文にかぎって論を進める。

「文を理解する」または「文の意味がわかる」とはどのようなことであろうか。また，このことと，「文が誤っている」とか「文が正しい」とかということはどのようなことであろうか（この後者の場合は，もちろん，文法的観点から正誤を問うているのではない）。これらのことは，Steniusに従えば，次のように説明される。

たとえば，次のような文があるとする。

(1) 月は地球より小さい。

これを記号的に表現すると，

(2) mSe

この(1)と(2)とを，要素に分節化（articulate）する。

(3) {月，地球，より小さい一関係}

(4) {名前〝m〟，名前〝e〟，名前〝S〟}

そして，これらの要素同士を対応させる。

(5) 名前〝m〟↔ 月
名前〝e〟↔ 地球
名前〝S〟↔ より小さい一関係

ところが，三番目の対応は，一方が名前であり，他方が関係であるのでこのような対応は許されないとSteniusは言う。そして，(4)の分節化のかわりに，

(6) {名前〝m〟，名前〝e〟，〝二つの対象の間に，第1のものが，文字〝S〟の左に，そして第2のものが右にあるとき成立する関係，（これを〝S〟一関係とよぶ）}

とし，次のように対応させる。

(7) 名前〝m〟↔ 月
名前〝e〟↔ 地球
〝S〟一関係↔より小さい一関係

これは，対応する要素同士が同じカテゴリー，すなわち要素は要素同士，関係は関係同士であるので，内的構造が一致すると言われる。更に，対応(7)は，関係によって結びつけられる要素同士が対応し，その関係同士も対応しているので，外的構造が一致していると言われる。よって，(1)と(2)とは，キー(7)によって同型対応するといわれる。そして，(2)は，(1)を原型とする像であると言われる。そうすると，

〝文が正しい≡文と同型対応する原型が現実に存在する〟と言えよう。したがって，もし，現実に原型が存在しなければ，その文は間違っている，または偽な文であるといわれる。それでは，偽な文の意味を含めて「文の意味」はどのように説明されるだろうか。そのためには，「事態」という概念の導入が必要となる。たとえば，(2)の代りに，

(8) eSm

という文を作ってみよう。これは，解釈のキー(7)によって，(1)とは内的構造が一致するが，外的構造が一致しない。すなわち，(8)と同型対応する原型が現実に存在しないので，偽な文である。しかし，(8)は全く無意味かというとそうとは言えない。少くとも，その「意味」はわかると言っても，納得する。この間の事情をSteniusは，「文は事態を描写する」と言い，「文の意味」がわかるというのは，この描写されている「事

態」をわかるということなのであるとする。ここで「わかる」ためには，当然わかる人の，その文への積極的関与 ── 分節化（要素と関係への分節化）── を前提にする。日常的には。文らしきものを見聞きすれば，殆んど自動的にその分節化は行なわれ，言われていることはわかるが，本当か嘘か，信じるか信じないか，受け入れて行為するか躊躇するかという点にまで進む。しかし，専門的知識の場合には，言っていることがわからないことが起る。特に数式のような文は，字面は目に見えていても，分節化は，不可能ということがはっきりわかる。したがって，「事態」とは，文によって描写されている物事の有様である。その有様が実在するか否かについては問われない。と言っておいて，実は，当該の文を，その一例として解釈するキーが構成されるので，その有様は実在するから，そのような問いは問うても問わなくても良いのであると言い直そう。（この点が，建築設計における言葉と形の関係をカバーする考え方だと筆者は考えている）したがって，自己以外において，そのような有様の現実的存在について問われないと言い直さなければならない。以上をまとめると，次のようになる。

文は現実の像であり，現実との関係で次の三つに分かれる。

ⅰ）真な像

文Fと現実の分節場Gとがあって，Fの要素の集合と，Gの要素の集合との内的構造が同じであり，外的構造が，また同じであるならば，文Fは，場Gの真な像である。

ⅱ）偽な像

文Fと分節場Gとがあって，Fの要素の集合とGの要素の集合との内的構造が同じであるが，外的構造が同じでないとき，文Fは，場Gの偽な像である。場Gは，偽な像の原型であるが，原型を描写せずに，可能な事態を描写する。可能な事態は，偽な像の外的構造によって示されるように在るとして描写される。

ⅲ）虚な像

文を，論理的主語（個体要素）と論理的述語（関係要素）とに分節化したとき，少くともその内の一つが架空なものに対応すると考える場合，すなわち，自己以外に現実の原型の存在しない像である。すなわち，論理的主語にどんな現実の対象を対応させることなく，とにかくそれ自身以外の対象をあらわすと考えるときや，論理的述語に架空の性質や関係を対応させると解釈するとき，そのような文は，虚な像である。

したがって，ここでは，Steniusの"考えることは，現実の適切な ── 真または偽な ── 像の形成である"ということに加えて，虚な像の形成をも加えた広い観点で，"思考"ということを規定する。

2. 実世界の現象の述語論理的分析

教育場面においてパソコンを導入しようとする場合には，教育場面を論理的に把握することが第一段階である。そのためには，先ず言葉による記述がなされなければならない，そして，その記述された文が，1で述べたように分節化されねばならない。その一つの手段として，文を述語論理式的に表現することが有用である。自然言語的文を述語論理式的に表現することをここでは，述語論理的分析または，述語論理式への変換という。述語論理的分析により得られた述語論理式を眺めることにより，次のようなことがわかる。

ⅰ）文で考えている要素の個数（述語論理的には，個体変項の個数）

ⅱ）一つの文で同時に考えている要素の個数（述語論理的には，結合入れ子量記号の個数）

ⅲ）文では，要素のどんな性質や関係が考えられているか（述語論理的には，述語の種類と個数）

たとえば，設計製図の課題「小公園」において，「どんなものにするか，そのイメージを述べよ」という問に対して，

(10) 噴水などの配置をいろいろな角度から考えてやる。

という文によって答えられたとする。これを，述語論理式に変換しようとすると，「考えて」という語の使用は，Quineの言う，「表示的不透明性の問題」を引き起すので，

(11) 噴水などの配置をいろいろな角度から行なうと言い換える必要がある。この言い換えた結果を，学生は不満とするなら，「考え」の内容を追加して聞くことを行わねばならないが，ここでは，(11)を学生は納得したとして話しを進めよう。そこで，

A①：私は，①をいろいろな角度から行なう，

B①：①は噴水の配置である　として，

(12) $\exists x(Ax \wedge Bx)$

これを，A①②：①は②をいろいろな角度から行なう，

a：私（つまり，固有名詞的に考えて，唯一特定の人の名前と考える）として，

(13) $\exists x(Aax \wedge Bx)$

または，「私」として，特定の教師と特定の学生が対応するとすると，I①：①は私なり　として，

(14) $\exists y \exists x(Iy \wedge Bx \wedge A'yx)$

これを，日常語的に読むと

(15) あるものyが存在し，それに対して，あるものxが存在し，yは，私であり，xは，

噴水の配置であり，そして，yはxをいろいろな角度から行なう。

となる。これらのどれを（11）に対応するものとして採用するかは，（11）をどのように考えるかということによる。学生の個人的設計意図の表明（もちろん，（11）では不充分であるが）とすると，そこでの「私」は，一人しかいないから，（13）が望ましいし，（13）は，（12）へ還元されるから，（12）で良いということになる。もし，教師と学生との対話または，教師の設計指導という場面を考えるならば，一種の追体験をすることになるから，（14）が望ましい。この場合には，この文で考えている要素の個数は2個であり，同時に考えている要素の個数も2個である。また，性質述語は，2個であり，関係述語は1個である。これらの諸種の個数は，考えることの難易度，理解の難易度等を見極めて行くときの一つのデータになろう。たとえば，同時に考える要素の数が6個あるというような文を満足するような形態を考えるというような場合仮にある要素を具体的に決定することによって，同時に考える要素の数を減らせば考え易くなるのではというように操作的な思考の戦略を立てることができよう。

このように，教育場面における様々な文表現を述語論理的に分析することは，考えの筋道を，最小のステップで跡付けることができ，「飛躍による理解不可能」ということを消去することができよう。しかし，逆に，ステップが最小であるが故に，それをたどるのが七面倒であるということも生じよう。

また，述語論理的分析に馴じまない部分 ── 「考えて」のような動詞の出現 ── も明示的になり，そのような部分における表示的不透明性による理解の困難さ等も検知できてくるであろう。

3. 述語論理的分析とProlog言語

このような述語論理的分析とPrologによるプログラムの構成または，その実行との関連について述べる。厳密または精密に述べるには，論理学の歴史的発展にまで話しを広げなければならない。しかし，ここでは，Prolog言語の特性の理解と，ＩＣＡＩへの適用の可能性を開くために必要な範囲で簡単に述べることにとどめる。したがって，Prologプログラムの実行様態から，その論理的解釈へと，いわばProlog言語の発展を逆にたどるという説明の仕方になる。しかも，その源流への途中まで。

Prologプログラムの実行は，次に述べる「導出規則（resolution rule）」の繰り返し適用によって，「空節」といわれる結果を求めることである。導出規則とは，ハードウェアー的に見れば，次の（16）と（17）とから（18）のような式を作り出すことである。

(16)　?－$B_1, B_2, \cdots\cdots, Bm$.

(17)　B_1：－$A_1, A_2, \cdots\cdots, An$.

(18)　?－$A_1, A_2, \cdots\cdots, An, B_2, \cdots\cdots, Bm$.

すなわち，（16）の横棒の右にあるB_1と（17）のコロンの左にあるB_1とが同じものであるから，それを消去して，（18）のような式を作る。Prologで書かれたプログラムは，（17）の形式の文が，必要な数だけ並んでいる（ただし，コロンと横棒とA_1，……，Anが無いというものを含めて）。プログラムを実行するということは，プログラムに加えて，（16）のような形式の文を，新に与えてやる。するとハードウエアは，（18）のような式を作り出すべく，（17）のような形式の式を探しはじめる。そして，該当するB_1が見出されると，（18）の形式の式を作り出す。そして，今度は，この作り出された（18）を（16）の形式の式が与えられたと解して，新たな（17）の形式の式を探し出し，新たな（18）を作り出す。以下，同様に続けて行くと，BやAの式が一つずつ減って行き，最終的に，BやAが無くなってしまうということが生じる。そうすると，そこで実行は終了したことになり，「yes」という表示が出る。一方，（16）が与えられても，それに対応する（17）の形式の式を見出すことができないと，「no」という表示が出て終了する。

このハードウエア的導出規則に，述語論理的意味を与えると次のようになる。（ハードウエア的導出規則は，本来の導出規則の意味を払拭し，形式的に述べている。したがって，その形式に意味を再び与えるとき，本来の意味とは異なっていても，その異なった意味で，実際の現象が矛盾なく説明できるならば，そのような解釈も許されることが，述語論理の完全性定理によって保証されている。たとえば，データベースと検索，書き換え規則，手続き的解釈等。その中で，ここでは，論理的解釈を述べるのである。）ただし，∧：連言（そして），∨：選言（または），～：否定（でない），⊃：条件法（もし……ならば……）とする。すると，（16），（17），（18）は，次のような論理式をあらわしている。

(19)　(16)：～($B_1 \wedge B_2 \wedge \cdots\cdots \wedge Bm$)

(20)　(17)：～$B_1 \supset$～($A_1 \wedge A_2 \wedge \cdots\cdots \wedge An$)

(21)　(18)：～($A_1 \wedge A_2 \wedge \cdots\cdots \wedge An \wedge B_2 \wedge \cdots\cdots \wedge Bm$)

条件法⊃を，否定と選言で表現し，括弧をはずすと，

(22)　～$B_1 \vee$～$B_2 \vee \cdots\cdots \vee$～$Bm$

(23)　$B_1 \vee$～$A_1 \vee$～$A_2 \vee \cdots\cdots \vee$～$An$

(24)　～A1∨～A2∨……∨～An∨～B2∨……∨～Bm

そこで，(22)と(23)が成り立つと，(24)が成り立つかどうかを考えてみる。(22)と(23)とを選言で結合すると，その中には「～B1∨B1」という式が含まれている。これは，恒に真である。したがって，(22)と(23)とを選言で結合したものは，(22)と(23)おのおのの真偽にかかわらず恒に真である。そして，この恒に真な式「～B1∨B1」を除去した残りは，(24)であり，この(24)の真偽について考える。そのため，再び(24)を(22)ととらえ直して，(23)に相当する式を探して同じ操作を繰り返すと，最終的には，「～B∨B」という式を除去すると，何も残らなくなるか，または，(23)に相当する式が無く，(24)のままで終る。前者の何も残らないという場合は，プログラムの中に，(24)を構成する式～A1，……，～An，～B2，……，～Bmのそれぞれに対応する否定式A1，B2，……，Bmが存在するということを示すから，(22)は，プログラムに書かれた式の全体によって反証されることになる。逆に，(22)の否定である

(25)　B1∧B2∧……∧Bm

は，証明される。一方，(24)のままで終り，それ以上進まない場合は，(24)を構成する式のおのおのすべてに対応する否定式A1，……，An，B2，……，Bmが存在しないことになり，(22)は反証されないことがわかる。この最終的段階を逆にたどると，(24)は，その構成要素の式が反証されないので，全体として反証されないということにとどまる。そして，Prologでは，「no」がCRT上に表示される。

ここで，(16)と(19)とを比較すると，(16)の「？－」が，(19)の「否定～」記号に対応しているとも解される。したがって，ユーザーは，(16)の「？－」の右辺に，B1 そして……そしてBmと書いて肯定的に読み，肯定的に考えているが，その実行の論理は，否定式を作って，それが，プログラムの中に書かれている式によって反証されるかどうかをチェックし，反証されると「yes」という表示があらわれ，ユーザーは，「証明された」ことを知る。したがって，prologの実行とは，プログラムの中に書かれた式を前提とし，新に(16)の形で書かれた式を証明することであるということになる。

4. ICAIプログラムの一般的機能条件

prologの実行が，プログラムの中の式を前提とする証明プロセスであるという解釈のもとで，ICAIプログラムが備えるべき，機能条件を一般的に考察してみる。この条件は，これからプログラムを作成する際のガイドラインとするものであり，そのようにして作成されたプログラムを実際の使用に供したときの結果によって変わってくることは言うまでもない。

再び，(12)を取り上げよう。この(12)は，証明できるであろうか。もし(12)が証明されると，個体変項xに入る噴水の配置の存在が保証される。しかし，設計のはじめの段階では，そのようなものが存在するわけではないから，(12)のprolog言語による表現

(26)　？－a(X)，b(X)

を実行すると，「no」と出る。ただし，prolog言語の規約によって，変項は大文字，述語は小文字化し，存在量記号は表示不用となっている。したがって，(12)が証明されて，噴水の配置を手に入れるには，ユーザーが，それらの形や配置を図形的に構成し，プログラムの中に式として加えてやらねばならない。すなわち，prolog言語によるグラフィックスのはじまりである。たとえば，CRT上に「四角形」を描くという場合，

(27)　g1:－g_box(100, 100, 200, 200, 7, 0).

という式を作ってプログラムとして与え，

(28)　？－g1.

を実行すると，CRT上に対角項点の座標が(100, 100)と(200, 200)の長方形が描かれて，「yes」で終る。したがって，図形を構成していくプロセスは，図形の座標値を与えていくことによって，すなわち，

(29)　g_box(X1, Y1, X2, Y2, 7, 0)

の変項X1，Y1，X2，Y2の値を変えたものを前提として，実行中または，プログラムの中の式として蓄積することにより，「～g1」が反証されて，g1が証明され，CRT上に図形が残る。このようにして得られた一まとまりの図形を図形素とよぶと，図形素を構成している線分の座標点は，他の線分の座標点とおのずからある幾何学的関係にある。それらの関係は，座標値を調べることによって取り出すことができる。これらの諸関係を内的秩序関係とよぶ。この内的秩序関係は，その図形素を，移動や拡大(縮少)しても保存されるものであるとイメージすると，二つ以上の図形素間の関係が考えられる。これは，一つの図形素の移動・拡大等によって保存されないものである。このような関係を外的秩序関係とよぶ。この外的秩序関係には，二種類あって，偶然にそうなっているという場合と，設計者の意図によってそうなっているという場合である。後者は，設計者が，文(式)で表明しないかぎりは明示されない。内的秩序関係の場合はどうであろうか。

この場合は，図形素が，幾何学的に小さく，また単純であることによって，その隅々まで設計者の意図によるといえよう。その意図は，作業仮説的なものであることがしばしばであるが。

このような設計思考活動（微視的には，図形構成活動）を，1で述べた思考モデルによって再述すると，外的秩序構成とは，あるレベルでの図形素を要素とし，図形素間の関係を規定することであり，その関係は，一方では，ＣＲＴ上に，それがあるように描写されるが，他方では，設計者のある意図（または規範）を満足していると言えよう。その場合，設計者のある意図は，文（または式）によって明示される必要があり，それらの文（または式）は，プログラムの中に蓄積される必要がある。その際，当然その解釈キー，すなわち，構成図形と意図の表明文との間の内的構造と外的構造の一致を明示する対応関係も蓄積されねばならない。そして，それらを必要に応じて取り出すこともできねばならない。

一方，内的秩序構成においては，図形素としての性質たとえば，室という図形素であるならば，面積・採光・通風・遮音性能等のような性質が，自動的に計算されるということで良いであろう。というのは，図形素というのは，相対的なものであり，図形素そのものが思考の対象となるならば，それは，更に，要素と関係に分節化され，外的秩序関係を示すものと，設計者によって転化されるから。そして，その場合には，解釈キーの蓄積が行なわれるであろう。

以上は，主として，設計製図の授業を想定したときの一般的機能条件である。次に，構造系の授業の面から，前稿の試作モデルの実際の使用結果を踏まえて考察してみる。

構造系，特に構造力学においては，演習問題に対する答はたとえ解法が異っても誰が解いても同じであり，また同じでなければ困るのである。しかしながらこの唯一の答に至るには幾つかの仮定や前提が在ることも頭に入れておかなければならない（構造計算では近似計算法を用いることもあるのだから）。そして，演習問題を出題するからにはある程度の量の問題（問題ベース）を用意しなければならないし，それらをどのような順に示すかという指導戦略も立てる必要がある。これらは設計製図の課題とは異なる点である。演習問題を基本的なものに限れば，それをそれ程多く作る必要はないが，誤答時等に必要となる派生する問題は誤答の原因によってまちまちであるから自動的に作成され保存されねばならない。また，問題が一つではないのでそれを受け取って提示して学習を実行する部分や入力された解答を分析し正誤判定および誤答原因を探索する部分は問題と独立させる必要に迫られる。こうして前稿で述べたＩＣＡＩの主な要素が浮び上ってくる。

次に誤答原因究明に重点を置いた前稿試作モデルを1の思考モデルにより分析してみよう。

学習者の入力した解答を文Ｆとすれば，用意された答（ある種の論理によれば別の答があるかも知れないという意味でこう記述する）は現実の分節場Ｇであり，正誤の判定や誤答原因の究明はＦの場Ｇに対する内的構造，外的構造の一致，不一致を見る解釈キーの探索に外ならない。解釈キーの探索は，分節場Ｇに対する合理的場合分けにより生じる誤答パターンの解釈キーをある程度用意し，それとのマッチングにより行う。学習はマッチングに失敗したときは解読不能のメッセージの出力により再答を促し，成功したときは各解釈キーに割り当てたコメントを表示して学習者自身に誤りを気づかしめるように再答を促す形式であった。ここでＦと場Ｇとの要素の対応はプログラミングの簡単化のため，解答入力において使用する記号を指定することにより行ったが，本来このような指定はなくして対応がとれるのが望ましい。

力学の演習問題等のDecode型では，ここに示されたように既にプログラム内に現実の分節場Ｇが正誤のバラエティーを含んで在り，その解釈キーもある程度用意できるが，気をつけなければならないのは入力されたＦに対するコメントの押しつけである。すなわち，通常の講義における教師側からの一方的な情報発信になりかねない。演習問題の種類によっては用意された答とその誤答のバリエーションは全て準備することが可能であるが，果してそれだけで十分なのだろうか。〝これでよし〟とするのでは〝従来のＣＡＩとどこが違うのか〟，〝どこが知的（Intelligent）か〟と問われて答に窮しかねない。ここで前述の正解とせず用意された答と記したことを思い出し，設計製図での機能条件（それはEncodeモデルであることから必然的に学習者の入力した図形素ーすなわち設計者にとっての現実の分節場Ｇーに対するその意図ー文Ｆーおよび解釈キーの何らかの方法によるプログラム内への捉り込み機能）を考えれば，学習者の素朴な内的分節場Ｇ′を無視することはできなくなる。プログラム内に準備された解釈キーとそのコメントの押しつけではなく，入力されたＦに対する学習者自身の場Ｇ′とその解釈キーを類推する機能を備えてこそ知的な処理に一歩足を踏み入れたとしてよかろう。実際に前稿のＣＡＩプログラムを実施した時，次のようなことが見られた。そ

れは，考慮すべき要素の欠落した入力に対する「何か足りませんね」というコメント表示に対し，その誤りに全く気づかず何度も同じ入力を繰返すという光景であった。その状況では，柱頭が移動しないとすれば入力された解答は正しく，その論理に固執する限りパソコンは間違ったことを表示すると受けとられてしまう。後で教師のアドバイスにより違いに気づいたことであった。人間の教師であれば学習者の間違いに臨機応変に対処し，その能力に応じてコメントすることができる。その応変にコメントを弁別する規則を入力Fの軌跡や必要ならばプログラム側から学習者へ質問する等の応答によって生成し，かつ新たに付加させうる機能がなければならない。

この意味で前稿のモデルはまだＣＡＩにすぎないが十分にＩＣＡＩへ発展しうる下地はあると思われる。たとえば，answer－analysis 部に入力F等の軌跡の分析部やそれに対する応答部を追加することにより進展させられる。prologによるプログラミングはこういう時にも非常に有利である。

またもう一つのアプローチとして，プログラム側が常に主導権をにぎるDecode 型に対して主導権を学習者側へ委ねる （すなわちEncode型）指導戦略も考えられる。例えば，解答入力時に質問モードを設け，学習者が問題解答時に疑問点をプログラム側へ質問することができる形式である。この場合質問に対しては，学習者の疑問を完全に解決できないまでもその場で次へ進める程度に，つまり学習の不連続による意欲の喪失を招くことなく，応答できることが絶対条件であり，質問応答部に知識吸収タイプの処理を行うことのできるインタプリタの機能を備えられれば学習は学生主体として進められるようになり，彼の学習意欲をより喚起できるものとなろう。

以上が構造系（問題演習を主として行う場合）のＩＣＡＩプログラムの備えるべき一般的機能条件であり，これらから，ＩＣＡＩプログラムの備えるべき一般的機能条件は，

(i) 学習者の内的分節場をとらえるため，学習者の入力と各入力時のコンピュータによる解釈キーを蓄積する。

(ii) (i)のデータを分析し，学習者の入力の軌跡を抽出する。

(iii) (i)，(ii)のデータから，学習者の内的分節場の解釈キーを類推する。

ということになろう。勿論，未だ仮説の域を出ないが，我々にとっては，ＩＣＡＩプログラム開発のための作業指針である。

その他前稿試作モデルの実施で感じた現実的な問題点を挙げてこの稿を終える。

- プログラムはデータ入力時に如何なるキー入力があっても不都合を生じないように十分な対策を講じておかなければなければならない。
- prologプログラムでは入力を出来るだけ簡単にできるようにし（ read 述語は使用せず），“・”なしで入力できる述語を用意すべきである。
- ＣＡＩ／ＩＣＡＩプログラム開発では，学習の実行部と同時にその学習結果を整理し資料化する部分も合わせて作成すべきである。

参考文献

1. 谷口興紀・横井友幸，建築教育におけるパーソナル・コンピューター利用の可能性について，福山大学工学部紀要第６号，1984
   横井友幸・谷口興紀，建築教育におけるパーソナル・コンピューター利用の可能性について（２）（Prolog 言語と述語論理の可能性について），福山大学工学部紀要第７号，1985
2. E. Stenius, Wittgensteins' Tractatus, Cornell Univ. Pr., 1960
3. 倉田令二郎，数学論序説，ダイヤモンド社，1972， P 176