CAI系列之三

淺談發展CAI程式的步驟與架構

蔡文榮

臺北市政府教育局

一、前　　言

自六〇年代開始，當各式各樣的電腦在各種教育情境中的衝擊不斷地持續擴大，在資訊科學最發達的美國社會中，坊間愈來愈多的書籍也從一般性的電腦領域逐漸專注於電腦輔助教學（Computer-Assisted Instruction，CAI）這一新的園地之探討，以實際地幫助各級教師來發展CAI的課程，在此，我們將對發展CAI 程式的一般性原則作一初步的探討。

二、發展CAI教學單元的步驟

首先，我們假設我們都已學會了一些常用的電腦程式語言與CAI 專用語言（如：BASIC，PASCAL，SUPERPILOT……等語言），知道如何運用這些程式語言的基本指令來寫軟體程式了。以下這些步驟所建議的，則能使一般初出茅蘆的程式寫作者寫出最正統、最完備的電腦輔助教學之程式。

㈠ 詳述教學的目標（Specification of Objectives）：

除非每一課的教學目標已經清清楚楚地釐定了，要不然，我們將無明確的依據來選擇與編排它的教材內容；並且，我們也將無法確定所編排的單元是否有效？若是眞的有效的話，又是以什麼立場來說的呢？

確定教學目標的另一個目的則在幫助我們於“整體分析”（Front-End Analysis, FEA）時（詳見下段說明），能正確評估電腦輔助教學是不是最有效的方式來達成預期

的教學目標。假如答案是否定的，那麼，我們可能會花費了許多時間和精力，而在某一個教學單元中却比其他的教學方式之成效差。譬如在 1974 年時，薄梭等人（ Berthold and Sachs ）用重覆測量的方式對六個情緒困擾的學童施以三種不同的教學方式（電腦式、傳統式、混合式），來教他們乘法和拼字，結果因爲學生“太注意操作電腦而導致分心”之故，CAI 式的教學成果最差。

㈡ 整體分析（ Front-End Analysis ）：

柏克（ Burke ，1982 ）在他的專書（ C.A.I. Sourcebook ）中描述爲：它（ F.E.A.）是一種關於某一教學單元之重要性與理論的決定，它評估該教學單元以電腦輔助教學方式呈現是否適當？對所涵蓋的教材內容也須事先作一通盤的考慮，就如它本身的英文字義所指出的，這一切的分析與評估都必須在詳細設計教學單元之前做完。譬如說：根據 F.E.A. 的分析，我們可能決定將電腦輔助教學的呈現針對某一特別類型的學生（如：低成就的學生）的需要；或者是限定原來的教學目標；或者是放棄以電腦輔助教學的方式來呈現。因此，在一開始就作好 F.E.A.的工作，將能幫助我們該如何決定該教學單元的範圍、角度、目標、著重點、與彈性等。

㈢ 詳述結果（ Outcome Specification ）：

這是以簡潔的方式來呈現一幅清楚的圖畫：到底學生能學到什麼東西？能學到何種層次？只是簡單的再認能力（ recognition ）？還是也包含了高層次的分析、綜合、演繹的能力？柏克所建議的方式包括四方面：⑴組織並列出主題的清單，把要包括進去的主題內容一一作個簡扼的摘要；⑵組織並列出須執行的工作之清單，把學生在學完這一單元後能做的做一摘要；⑶針對每一工作所要求的學習形態作一工作分析；⑷發展標準架構，以測驗題的方式來測量學生在這一單元中的進步程度。若是以是非題方式的話，要答對百分之多少的題目才算過關？若是以選擇題的方式的話，標準又是如何？另外，回答的速度是否也在要求之列？這些都是須預先發展出來的。

㈣ 教學單元之設計（ Lesson Design ）：

在教學單元設計階段，首先，我們應該先草擬出一個整體的大綱，或其各重要部分的流程圖，標出各重要部分的相互關係及其先後次序。從這一份整體的計劃中，我們可以接著進行更詳細的綱目，或者在每一個重要段落中發展出更細緻的流程圖。例如：在一個加減法運算教學單元之程式中，是要先教加法還是先教減法？是先有簡單介紹與範例展示之後才有測驗呢？還是先有測驗（ pretest ）後有介紹與範例，再有後測（ posttest ）？主程式與各分枝的副程式（ subroutine program）之間的架構關係該如何

設計？這些都須先詳細設計並畫出流程圖。

㈤　教學單元之發展（ Lesson Development ）：

在這一個步驟中，我們實際地寫出教學單元，但是，這與一般撰寫BASIC或PASCAL的程式時又有些不一樣。在我們動手撰寫之前，我們必須有內容，並確定該單元內容的各種特性，諸如：它的長度、用詞、和先後次序。這在柏克與其他人的書中都有一些建議，然而，根據筆者與許多C.A.I.軟體設計師，尤其是現職中小學教師自己發展C.A.I.軟體的共同經驗，先在草稿紙上（有依螢幕展示之八十格乘二十四行之小格）把教材以螢幕大小的方式草列出來，這是最省時而有效的好辦法。因爲在每張草稿紙上所呈現的，就是在每一顯示幕上所要展示的，所以，當我們把教材都一張一張地在草稿紙上編好，若是以BASIC語言來寫程式，這時就只須加上行號碼，便可以很容易地把這些都順利地轉換到眞正的電腦程式中，發展C.A.I.程式的工作到此才算有了具體的成果。

㈥　教學單元之確認（ Lesson Validation ）：

這個步驟是最常被遺忘、却又是不可少的關鍵步驟。根據美國教育產品資訊交換中心（ Educational Products Information Exchange ）所作的研究顯示：在他們抽樣調查的一百六十三套軟體之中，約有百分之八十的C.A.I.軟體之發展過程竟然忽略了這個重要步驟（ Hassett ，1984 ）。事實上，雖然我們可能認爲我們現在完成的這一套C.A.I.程式十分高明而毫無瑕疵，然而，別人可不一定認爲這樣。所以，教學單元之確認就是以受教的學生團體中抽樣來測試其適用性。在測試的過程中，我們所須要考慮到的問題諸如：這個教學單元設計看起來眞的能達到它預期的教學目標嗎？換句話說，我們所期望的學習發生了嗎？這一個教學單元以電腦輔助教學的方式來呈現是實際的嗎？換言之，在規定的教學時間內能完成嗎？電腦的硬體設備夠嗎？學生的電腦素養（ Computer Literacy) 的背景夠嗎？另外，就C.A.I.程式的本身來看，它在執行時都沒有錯誤嗎？它的文法結構、中英文拼法、用詞遣字、與內容的排列是否也沒有錯誤呢？經過了這一道確認的程序，它的適用性才能夠有一客觀的認定，也才能談得上大力推廣到其他班級或其他地區。

從上面所談到的這六點中，我們當能明白發展一套完美的C.A.I.教學軟體並不是一蹴可及，事先細心的規畫與循序漸進的系統發展實是不可或缺。

三、常用的CAI程式架構（Frames）

所謂架構（Frame），又稱爲畫面頁，是在電腦螢幕上所呈現的一整個畫面，每個架構不限於只有一個畫面，一般而言，可歸納爲七種架構。

㈠ 簡介與指導語（Introduction and Instructions）：

C.A.I的教學單元一般是由標題頁（title page）開始的，這一標題頁一般都會包含了教學單元的名稱、作者的姓名、受教的學生年級層次、版權聲明、與其他簡介的話。通常在標題頁之後，無論多麼簡短，總要有些指導語，這些指導語應該提供關於如何使用這一個教學程式的重要資訊，而不必一開始就談的很細，其實，若眞有需要，可用一附帶的袖珍手册或講義來呈現其內部設計的詳細資訊。另外，因爲這些開頭的指導語對那些已使用過這一個程式的人來說常是重複的，所以，在這部分也應提供使用者能跳略指導語的選擇。圖一就是用BASIC語言所寫的簡單實例。

（在IBM電腦上的BASIC程式）

```
10 CLS:REM CLEAR ALL SCREEN
20 PRINT TAB(14);"Basic Operations":PRINT
30 PRINT TAB(16);"Written By":PRINT
40 PRINT TAB(14);"Mr. Wenrong Tsay":PRINT:PRINT:PRINT
50 PRINT TAB(14);"For Grades 1-5":PRINT:PRINT:PRINT
60 PRINT TAB(7);"Do you need instructions (Y/N)?":
70 PRINT:INPUT A$
80 IF A$ = "Y" THEN 400
90 IF A$ < > "Y" AND A$ < > "N" THEN BEEP: GOTO 10
100 REM LESSON STARTS HERE
200 REM LESSON STARTS HERE
400 REM INSTRUCTIONS START HERE
999 END
```

（螢幕上所呈現的）

```
Basic Operations
  Written By
Mr. Wenrong Tsay

For Grades 1-5

Do you need instructions (Y/N)?
 ? Y
```

圖一 標題頁

㈡ 選項頁（Menu Page）：

選項頁也有人譯爲清單頁，它通常是跟隨在標題頁和指導語的架構後面，它的目的乃在於呈現不同的選擇項目，好讓使用者能控制教學單元的流程。然而，是否要用一個螢幕的選項頁、或者說要如何用它，則須視我們的教學情境而定。譬如說：低年級的學生可能對一個科目還未具有基本的知識、或者還不知道在面對一大堆選項時不知如何做合適的選擇。而對其他學生來說，我們也不難想像他們會常常故意去選擇選項頁中的“

結束單元”（ End Lesson /Quit ），所以，我們在設計選項頁時，要對學生可能選擇的種種反應先做周詳的考慮。圖二是在螢幕上實際呈現的例子：

```
Basic Operations' Menu

1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Practice Quiz
6. Mastery Test
7. End Lesson

Enter the number of your choice here:
```

圖二　選項頁

在這例子中，我們可看出以數目字來取代英文字母的好處，至少學生不會因字母的大小寫的講究而困擾（註：A與a在電腦上各有不同代碼，餘類推，詳見美國ASCII 碼對照表）。

㈢ 教學架構（ Teaching Frames ）：

在這教學架構中，我們就是作實際的教學工作，它可能包括一些敍述、一些問題、一幅圖、或一張表、一個生動的程序或其他材料。事實上，對於那些須要很多文字上解釋的主題來說，C.A.I是一種差勁的媒介，一本教科書或一份完備的講義將會更方便、更便宜；至少，要找一段主題時，電腦搜尋資料所花費的時間常常會久一些。

一般說來，教學架構有許多形態，若要一一將它們歸類詳述則爲本文篇幅所不許，所以，在此我們只介紹二種常用的類型，這是在柏克（ 1982 ）的專書中提到的，第一種是EGRUL式（E.G.就是e.g.，意即範例，R.U.L.就是rule，意即規則），它要求學生能從一些特殊的範例中歸納出一條規則或一般的類別（詳見圖三），第二種是剛好相反的PULEG式，它是先呈現所要教的規則，並加以說明，然後要求學生去辨認範例是否相符（詳如圖四）。

```
A$, B123$, BOY$, and GIRL$ are all
examples of what kind of variable?
```

圖三　FGRUL式

```
String variables must include a letter
as the first character and a $ as the
last character.  Which of the following
are valid string-variable names?

1. M%     2. B$     3. $QR     4. FFGG
5. EGG$   6. ASDF   7. A3$     8. FOOL$

Type the numbers of all your selections
in succession (No spaces or commas).
Answer:
```

圖四 RULEG式

㈣ 測驗架構（Test Frames）：

測驗架構是用以評估學生學習教材內容的成效，所以，這些測驗題必須避免提供學生任何能讓他們猜中正確答案的線索。一般而言，CAI 的測驗題是分爲二種方式，第一種是多選一的選擇題（詳如圖五），主要目的是在評估學生的再認能力；第二種是填充題（詳如圖六），主要目的則在評估學生的記憶力（recall）。但是在CAI 上出填充題並不容易，關鍵乃是在於電腦上只能對一字不差的正確答案計分，所以學生的答案若只是同義詞、或拼漏了一個字母，甚至不小心多按一個空格，電腦的反應却一定是錯誤的答案，而這些缺陷在傳統式的手筆測驗中却不成問題。所以，在設計C.A.I的填充式測驗題時，應先考慮到下列這些重點：⑴確定所要求的答案形式，告訴學生要特別注意拼法；⑵若是必須的話，可用些提示符號來幫助學生界定所要的答案形式，例如：“ The two types of computer memory are RAM and___”. 在這題中，這三個空格的提示能幫助學生縮小回答的方式；⑶可以用陣列的方式將正確的答案之同義字編列進去，例如：A$(1)=“ DISK ”，A$(2)=“ DISKETTE ”，A$(3)=“ DISC ”，

```
Temporary or programmable computer
memory is called _ _ _?

1. ROM
2. CPU
3. RAM
4. CAI
```

圖五 選擇題之測驗架構

```
Temporary or programmable computer
memory is called _ _ _.
```

圖六 填充題之測驗架構

A$(4)＝“ FLOPPY DISK ”，然後一一來比較受試者的答案，如果能與其中一個相符，就可視爲正確答案。

㈤ 回饋架構（ Feedback Frames ）或補救架構（ Remedial Frames ）：

在很多情況下，我們因學生的反應勢必岔開到補救教學的路上來，以使學生能有完全學習，例如，一個學生在圖五的例題中選擇R.O.M爲該題答案，在電腦螢幕上就應馬上可以看到合適的回饋。（詳如圖七）。

```
You are wrong!!!  ROM is the computer's built-in or permanent memory.
ROM stands for Read-Only Memory. It cannot be influenced by your program.

Press any key to try again.
```

圖七　回饋架構實例

㈥ 增強架構（ Reinforcement Frames ）：

我們都知道適當的獎勵對學習過程有增強的功能，在傳統式教學是如此，在C.A.I.上也是一樣。當CAI 程式執行到測驗題部分，程式設計者必須決定學生該在什麼時候並且是如何得到有關他們反應的消息。一般而言，增強架構有下列幾種方式：第一種是逐題式地提供學生反應的資訊，它可能只是簡單地說“ Correct ”（答對了）或“Incorrect ”（答錯了），或是再加上一些鼓勵的話；第二種是在一系列的試題後面才跟著一個總結式的評語，諸如：“你的答對率在95％以上，恭喜！”或“在二十題中你答對了十五題，請再加油！”之類的話；第三種則是綜合式的，不但提供個別題目的資訊，也提供總結式的評語。另外要注意的是不能用特殊的圖案或聲音效果來凸顯學生錯誤的反應，其實只須簡單而有禮地呈現“你的答案不正確”通常就夠了。一般而言，特殊的聲光效果與圖案展示是用來作爲積極的正面增強，然而，目前在美國這些通用的CAI 軟體中，我們可以發現這些特殊的圖案與聲光效果却被氾濫使用，而對那些自動自發、有強烈求知慾的學生來說，這些特殊的增強效果却常被認爲是一種攪擾，因它打斷了學習時間的延續，這時，最好的報償與增強就是快快跳到下一個問題好讓他們能一展身手。

㈦ 圖案架構（ Graphics Frames ）：

一般說來，圖案能增加CAI 程式的多樣性，並使學生更有興趣去學習，然而，我們不可否認的是：圖案架構通常非常費時才能完成。所以，若是我們遇到一個教學單元須要很多或很複雜的圖案設計，我們最好能用一些著作式語言（ authoring language

），諸如 Superpilot 之類，這樣會比使用BASIC的指令更省時。但是，雖然C.A.I的圖案架構在展示一份地圖、表格、與特殊圖案時通常比掛圖的效果好，若是時間不夠，倒不一定要以C.A.I的形式呈現，畢竟並不是每個人都有很多餘暇去慢慢將圖案轉化成電腦程式。

四、結　　語

史寇爾（Schorr, 1983）發現在一般教學情境中，若是老師一味選用現成的C.A.I.軟體，却不事先予以評估其適用性，那麼，在教學過程中將不難發現所用的C.A.I.軟體是反教育的（anti-educational）。譬如說：有的電腦程式在學生反應錯誤之後竟在螢幕上閃示一個大X，甚至發出刺耳的吱聲，這似乎是告訴學生：“你這個大傻瓜！”另外，在筆者本身評鑑美國坊間流行的C.A.I.軟體時，也常常發現有些軟體的設計、用詞、拼字等均很不理想，譬如說：有一套爲升大學之SAT 考試而設計的速讀（Speed Reading）軟體十分暢銷，但是却有一些字漏拼或拼錯了。從上面這些例子中，我們不難發現很多C.A.I.軟體並沒有依照我們所談到的步驟與架構去發展，因此，當我們國內起步發展C.A.I.軟體時，無論是程式設計師或是各級教師，若是都能具備這些基本認識，那麼，國內的C.A.I.前途才可能是一片光明。

五、參考書目

1. Berthold, H.C., and Sachs, R.H., “Education of Minimally Brain Damaged Child by Computer and by Teacher.” Programmed Learning and Educational Technology, No.11, pp. 121-124, 1974.
2. Burke, R.L., “The C.A.I. Sourcebook.” Englewood Cliffs, N.J.: Prentice-Hall, 1982.
3. Hassett, J., “Computers in the Classroom.” Psychology Today, No.18, pp. 22-28, 1984.
4. Judd, D.H., “Teacher Created Programs : Suggestions for Success.” Educational Computer Magzine, No.2, pp. 34-35, 44, 1982.
5. Ross, S.M., “BASIC : Programming for Educators.” Englewood Cliffs, N.J. : Prentice-Hall, 1986.
6. Schorr, B., “Many Schools Buying Computers Find Problems. With Using Them.” Wall Street Journal, April 7, 1983, p. 29.