SCRIPTER: A PROGRAMMING LANGUAGE FOR COMPUTER-BASED PROGRAMMED INSTRUCTION

Haruhisa Ishida and Shigeo Furukawa *

1. Introduction

A CAI (computer-assisted instruction) system must be provided with a means of incorporating into it instructional materials prepared by teachers. A number of programming languages such as COURSEWRITER and PLANIT have been developed to facilitate the process of implementing so-called programmed-instruction materials. Most of these languages, however, require a large amount of key-typing just for introducing simple "frames" for teaching materials, resulting in the waste of both computer time (mostly of a time-sharing system) and man hours.

Also in most CAI systems, the computer is used merely as a page turner and a statistics taker, and the time-consuming process of text preparation is entirely in the hands of human teachers. Clearly, as W. Uttal** has pointed out, in his "generative CAI", the computer should be used to aid the teacher in the tedious process of text preparation as much as possible.

A simple interpreter program called SCRIPTER (after SCRIPT writER) has been written for our PDP-9 computer (a single-terminal CAI system with a small graphic display, a storage scope display, and a hybrid magnetic tape unit). The objectives were assumed to be such that: (1) almost any programmed-instruction materials can be put into the computer, (2) by a careful analysis of a script, the amount of typing or specification should be reduced as much as possible, while retaining readability, especially for non-computer-oriented teachers. (Use is made of default rules such as "assuming previous values unless otherwise specified"), and (3) the semi-automatic structuring of teaching texts should be provided for.

---

* Nippon Electric Central Research Laboratories

** Private communication

2. Steps in an instruction frame

Text materials were analysed and from this, an instruction frame (referred to by a label attached to it) in programmed instruction was defined as consisting of seven steps: Question-, Answer-, Comment-, Score-, Jump-, Decision-, and Final steps. They are as follows:

Q(question)-step/ One writes sentences here, anticipating students' responses for them. When the program is executed, the sentences are typed as they are.

A(answer)-step/ Possible answers are listed here as keywords (AND, OR, or NOT combinations of them are allowed) or as numerical values or ranges, arranged by classes 1, 2, 3, etc. Thus a student can answer by multiple choice or in sentence form or by a numerical value. The actual answer from the student is matched against this list of possible answers.

C(comment)-step/ A sentence written here for each possible answer is typed as a comment or a chimed-in message for the corresponding answer.

S(score)-step/ Integer arithmetic expressions are written here to keep a score or statistics on each answer.

J(jump)-step/ Here are listed the labels of frames where, depending on the answer, control should proceed.

In other CAI languages, these steps are arranged horizontally for each answer, e.g., in a form equivalent to Answer 1; C/..., S/..., J/..., Answer 2; C/..., S/..., J/... But the significant advantage of our vertical arrangement of steps is that, when any of the A, C, S, and J steps is the same as in the previous frame, they need not be specified anew, because default rules apply. In addition to the above steps, the following two steps are also provided.

D(decision)-step/ Integer arithmetic expressions to be evaluated, initial values of variables, or messages to be printed without anticipated answers are written here. A conditional jump to a specified frame can be written to change the course of instruction based on the past performance of the student.

F(final)-step/ This is the step for printing scores and final messages.

The following is an example from part of a script prepared by C. de Linde of Edingburgh University, written in the format acceptable to SCRIPTER(C-, D-, and S-steps are omitted).

P1, Q/ (Imagine you are a student now returning to London by train. A very pretty girl has come into your compartment. You think how nice it would be to get to know her... Each time the girl speaks to you, you will have a choice of two or three ways of carrying on the conversation. If you choose the best route you will get to know her very quickly.)

Excuse me, can you tell me the time please?

A. Five o'clock has passed.

B. I make it five fifteen.

C. I can tell the time.

A/1. A; 2. B; 3. C

J/1. P16; 2. P11; 3. P19

P11, Q/ (Well done. That was the best way to reply. Listen to her next question.)

Do you know when we get in?

A. I think it is occupied. I will look.

B. I got onto the train at three p.m.

C. We're supposed to arrive at ten to seven.

J/1. P7; 2. P14; 3. P8

P24, Q/ (Very well done! That was the best reply.)

That would be fabulous. By the way what's your name?

A. Nobody in England has heard of it.

B. My name's ___ ___. What's yours?

C. Yes, you would find it very interesting.

J/1. P6; 2. P10; 3. P32

P10, Q/ Elsie, Elsie Beady.

A. We're having a get-together at our Embassy next Saturday. Would you like to come?

B. LCBD? What does that stand for?

J/1. P3; 2. P43

P3, F/ Yes, I'd love to. Thank you very much. Where shall we meet?

(You've reached the end and you've 'made a date'. Congratulations!)

3. Structured frames

As a step toward the semi-automatic generation of script materials, a "type" definition has been introduced. An example follows of a type (TYPE T1) defined for a particular extended frame.

```
. TYPE T1
Q/ #S1                      (sentence S1)
A/ 1. Y;2. N                (response is Y or N)
S/ 1. #F;2. #G              (F and G are arithmetic expressions)
J/ 1. NEXT                  (go to next frame in case 1, otherwise continue)
R/                          (repeat the above Q)
Q/ #S2
J/ 1. NEXTQ                 (go to next Q in case 1, otherwise continue)
R/
J/ 2. NEXT
Q/ #S1
J/ 2. NEXT
R/
J/ 1. 2. NEXT               (go to next frame in both cases)
```

When the TYPE T1 is used and the following definitions are given:

```
. TYPE T1
S1=S11*S12;  S2=S13*S12;  S11='Does your brother'
S12='play the guitar?';  S13='Does he';  F=F+1;  G=G-1,
```

An actual session would look like this with the evaluation, N(no); Y(yes) given to the right.

```
Does your brother play the guitar?      *N
Does your brother play the guitar?      *N
Does he play the guitar?                *N
Does he play the guitar?                *Y
Does your brother play the guitar?      *N
Does your brother play the guitar?      *Y
```

Note here that only "kernel sentences" are provided and the sentences used are generated by the type definition.

In the newly introduced R(repeat)-step, the following format is also allowed.

```
R/ PLEASE ANSWER. /     (Type PLEASE ANSWER and the previous Q)
R/ BOOK//               (Delete BOOK from Q and repeat)
R/ BLUE/BLACK/          (Change BLUE in Q into BLACK and repeat)
```

These facilities are effective when a slightly modified question is presented following a question in an extended frame.

With the implementation of our SCRIPTER, it is now possible to put a variety of programmed-instruction materials on the computer with only a small amount of additional typing effort. (Materials are to be presented to SCRIPTER in the form of punched paper tapes.) The Score-step enables the instructor to find out paths no student goes through, thus providing him a means of improving his text scripts. When frames of fixed structures can be employed, the use of various type definitions will ease the task of text preparation.