Adding Videodisk Control to CAI Authoring Systems: A Software Customizing Methodology

This article briefly describes a project, recently completed at Concordia University in Montreal, in which a program for controlling videodisk players was interfaced with an existing CAI authoring system. The resulting package provides a tool with which professors and students can, with relative ease, turn the University's collection of CAV videodisks into level III interactive video tutorials. Of greater interest than the details of the current project is the process: a methodology for adapting existing commercial software packages to accomplish new and greater tasks. And this, without actually modifying the existing programs. The present discussion involves IBM-PC type computers and assumes some knowledge of the operation of a PC.

Background Details

Before elaborating on the process, it is necessary to provide some background details.

We gained experience in controlling videodisk players through our production, in 1985, of the pilot version of a commercial videodisk training package called *PIVOT,* used to introduce a new business telephone system. While *PIVOT* was well received by the critics, it had an enormous cost in terms of filming, graphics design, and original computer programming. Using similar production techniques in subsequent support of academic courses was simply not economically feasible. The videodisk *driver* program developed for *PIVOT* lay dormant.

Meanwhile, an authoring system called *Scenario* began to be used for in-house production of traditional CAI packages. *Scenario* offers a full colour text screen, support for graphics and music, and a well developed answer-processing capability. The call soon came for some use of *Scenario* in interfacing computers with videodisks players and producing level III interactive video.

The *Scenario* authoring system has its own native module for controlling video. This is available at an appreciable extra cost and requires the use of a third-party video adaptor for the computer which is even more expensive. Even at this, the videodisk operations did not perform at the level desired.

It was known from the earlier *PIVOT* project that no adaptor is required between an IBM-PC and the SONY LDP videodisk player, simply a null-modem serial cable.

A Fundamental Requirement

Could we not use the dormant videodisk driver program from *PIVOT* for control of the videodisk, while using *Scenario* for all other aspects of the desired training packages? It would be necessary for the author to be able to access the videodisk driver from within a *Scenario* program.

The initial, fundamental requirement was that *Scenario* be able to access external programs. Luckily, the authoring system possesses a feature called "borrow" whereby it can load and execute an external DOS program, regaining control when the external program has terminated.

Scenario accomplishes the "borrow" operation by loading a "secondary" copy of DOS's com-

mand processor (COMMAND.COM), in much the same way as BASIC's "shell" instruction, dBASE's "run", or LOTUS's "system." Having a second COMMAND.COM and a second program in memory at the same time as the host program does introduce a certain overhead, in terms of memory. But it can easily be handled by the 640K PCs which have almost become standard.

More serious was the time required to load from floppy disk an additional copy of COMMAND.COM, followed, in turn, by the videodisk driver program. In initial tests, over 12 seconds might pass between *Scenario's* "borrow" step and the videodisk player's response, clearly an unacceptable delay. The problem was solved through the creation of a "RAM disk", and the execution of all programs from the RAM disk. The delay time was reduced to under 1 second. (An additional boon was increased performance from *Scenario* itself.)

Scenario and the Videodisk Driver

Getting *Scenario* to run the videodisk driver program (now renamed VDP) efficiently was only the beginning of the solution. It was necessary for the author to be able to pass information to VDP from within *Scenario*, information on what, exactly, the videodisk player was to do. While *Scenario* offered an intricate system whereby external programs could "peek" at memory locations within *Scenario* to view the contents of variables, it was felt that this sort of intricate interaction should be avoided.

QuickBASIC 4.0 offers a new BASIC function called COMMAND$ which allows a BASIC program to derive information from the DOS command line which started it. Inside *Scenario's* "borrow" step, therefore, we could specify, not only VDP, but also VDP's instructions. The following is an example:

External Program to Borrow? TVDP/INXO/STL1000. In other words, turn index off and still-frame at - 10000.

Problem of Return Communication

Solving the problem of return communication was more difficult. When the SONY LDP videodisk player receives commands from a host computer, it returns feedback to that computer as to what is happening. We wanted this feedback available to the author working within *Scenario.* If, for example, the *Senario*-based tutorial attempted to play the videodisk, but the videodisk player was not turned on, we wanted the *Scenario* program to sense this and provide an appropriate message. If the student pressed the space bar to terminate a video sequence, we wanted to know at what frame the videodisk stopped, so that our tutorial could provide a tailored explanation.

Scenario has a feature called the "Accumulator Step", in which various internal *Scenario* variables can be set by the author, to keep track of such things as the student's score. *Scenario* can be set to branch to various different parts of the tutorial according to the value of its accumulators. A final detail, all of the "steps" that make up a tutorial are saved on the disk in binary files which only *Scenario* can interpret.

Could the VDP program "create" a *Scenario* accumulator step, such that *Scenario* would not know the difference? *Scenario* was used to create dozens of nearly identical "steps" in which accumulator #99 was set to a range of different values. Each of the "steps" was set to branch to different subsequent "steps" inside a fictitious tutorial. All other features remained constant. The binary files were examined in much the same way as an astronomer looks at dozens of pictures of the same section of the sky, hoping to find an object that moves. Anyone who has examined binary knows that they are usually "gibberish" unless one has the key to decode them. And, indeed, the significance of most of *Scenario's* files remains a mystery, but those portions pertinent to setting the accumulator variable and determining where in the tutorial to resume operation were identified and understood.

Thus, upon receiving feedback messages from the videodisk player, VDP "writes" a *Scenario* accumulator step. When control reverts to *Scenario*, accumulator #99 is examined, *Scenario* proceeds accordingly.

We can thus use *Scenario* for creating all aspects of the tutorial and, from within *Scenario*, can issue control messages to the videodisk player and receive and act upon responses from the videodisk player. Without the source code or any other technical support from the designers of the commercial software package, indeed without

modifying it at all, we were able to effect major changes in the way it performed.

Author's Notes:

1. *Scenario,* by Bernard Michaud, is a product of Techbyte Inc. 547 St, Thomas, Longueuil, Quebec, J4H-3A7, CANADA 514-670-9452

2. *VDP,* by Roger Kenner, is available from Concordia University, 1455 de Maisonneuve West, Montreal, Quebec, H3G-1M8, CANADA 514-848-3430. *VDP* was written using *QuickBASIC 4.0.*

3. The *Scenario* and VDP combination was developed in response to a request by Prof. Mariella Tovar and Gilles Doiron of Concordia University's Education Department, who have successfully used it in the creation of a level III interactive videodisk on "Radioactive Decontamination."

4. *PIVOT* was a joint project of Concordia University, SONY of Canada, Bell Canada, and Northern Telecom.

J.E.T.T. Contributor Profile

Roger Kenner is head of the Learning Laboratory Services at Concordia University and teaches computer programming and computer literacy. Interested readers may write to him at the following address: 1455 deMaisonneuve West, Concordia University, Montreal, Quebec, Canada H3G 1M8.