Streamlining ICAT Development Through Off-the-Shelf Hypermedia Systems

Michael Orey
The University of Georgia
Ann Trent, James Young, and Michael Sanders
Army Research Institute, Fort Gordon

This project examined the efficacy of building intelligent computer assisted training using an off-the-shelf hypermedia package. In addition, we compared this package to an architecture that had been developed in a previous contract which was based in the C programming language. One person developed a tutor in LinkWay (an off-the-shelf hypermedia system) and another developed the same tutor using the ALM C-based architecture. Development times, ease of use, learner preferences, learner opinions, and learning effectiveness were compared. In all cases, the off-the-shelf package was shown to be superior to the C-based system.

It seems as though the field of Instructional Systems Development (ISD) has abandoned any desire to work on the design and development of Intelligent Tutoring Systems (ITS, Rosenberg, 1987; Schank, Personal Communication, April 5, 1991). There are many reasons for this purposeful disinterest. One of the primary ones has to do with the amount of time that it takes to build a substantial ITS. This paper reports on the examination of using off-the-shelf development packages in order to streamline the ITS development process.

There have been many people who have discussed the complexity of ITS development (Jonassen & Wang, 1991; Merrill, Li, & Jones, 1991; Orey & Nelson, in press; Towne, 1991). Merrill and his colleagues have suggested that traditional computer-based training takes about 100 hours of design and development for every one hour of instruction. Further, they have estimated that the design and development time for ITS is approximately 500 hours per hour of instruction. At least one of the current authors has bragged about the time it took to design and develop an ITS. One of these tutors, a whole number subtraction tutor, took 800 hours of programming in Lisp to develop. Children who have used this tutor, average about 5 hours to achieve mastery. Therefore, it took about 160 hours of time to develop an hour of ITS instruction. This seems prohibitively long and is at the root of the prohibitiveness of ITS.

An ITS must include an expert component. Jonassen and Wang (1991) presented an expert system for science at the annual meeting of the Association for Educational Communications and Technology. They intimated that the development time using an expert system shell for this expert system (not ITS) was very lengthy. The response by the audience was clear. Why spend so much time for so little program functionality? Anderson (1988) suggests that the ITS must "know" the domain to teach the domain. Therefore, the expert knowledge for the domain must be encoded. Expert system shells streamline this process, but Jonassen and Wang's experience suggests that it is still not practical for instructional development purposes.

Doug Towne (1991) has begun work on an automated ITS development system. Like Merrill, he predicts that by using his system, considerable savings will be achieved. He suggested that an expert in the domain could sit down to his system and spend one hour going through it and walk away with one hour of ITS instruction. This is an honorable goal and will certainly make ITS practical. However, this system has not yet been developed, so in the meantime, we have been working on how to streamline the process without ITS specific tools.

During an Artificial Intelligence in Education class taught at the University of Georgia, we examined how to minimize the development time. As part of this class, one of the authors of this paper created an ITS to teach teachers how to link screens together using hyperCard®. This tutorial was created in hyperCard and used Xrules to program the help button. In this way, the developer could use hyperCard to interactively create the interface, use hyperTalk (a high level English-like programming language) to program the simulation, and use Xrules to represent the expert's knowledge in the domain. It took 8 hours to create the simulation and the tutor and the instruction lasted about 30 minutes. This result would suggest that ITS is not impossible or impractical.

At the root of this development process is the idea of what an ITS is. Orey and Nelson (in press) have suggested that there is not a dichotomy between traditional computer-based tutors and ITS, but that there is a continuum. This continuum is based on the level of interaction that is required of the learner. A passive system would be at one end of the continuum. By passive, it is meant that the learner sits in front of the computer screen and information is presented. At most, the learner presses a key to continue. This kind of instruction can be enhanced with questions. The learner is presented with a couple of screens of information followed by a question that is based on the information on those screens. Finally, the other end of the continuum is anchored by the ITS where the information is presented in the context of solving real problems. The information can be presented at the time of an error (a key learning time) or can be presented when requested in the context of solving a problem (when the learner needs to know the information). This system would "know" the domain, "know" what instructional strategy is best for this particular learner, and "know" what the learner does and does not know (or incorrectly knows) about the domain. The tutoring systems described here are not at this end point, but are towards this end. Error feedback is provided based on what the learner has done or left undone, and the advice that is provided upon request is based on the same criteria as error feedback. These systems do not "know" the domain, learner, or pedagogy in any real sense, but can provide appropriate feedback. We now turn to an analysis of two development environments.

COMPARING DEVELOPMENT ENVIRONMENTS

The content was operations procedures for one of the Army's mobile telephones (the Mobile Subscriber Remote Telephone or MSRT). We examine two different development environments -- an off the shelf hypermedia tool and the C programming language. Two programmers worked on this project. One used IBM's LinkWay® to develop a hypermedia-based ITS. The other programmer used a collection of C routines that had been used in a previous ITS to develop a new ITS in a slightly new domain. The C routines were originally developed for both operations and troubleshooting procedures for one of the switching shelters used in the Army's communications equipment. The idea was that considerable time savings could be achieved if the developer of the new tutor (the MSRT tutor) used the existing C routines (libraries). The original tutor, ALM (see, Orey, et al., 1991), essentially was a hypermedia based ITS. Its primary functions were to provide appropriate advice throughout the problem solving process and to provide corrective feedback. The second programmer chose to use a hypermedia tool and build the tutor from scratch. The constraint for both systems was that the system must run on an MS-DOS machine in EGA mode and use 640 K or less of RAM. LinkWay® is a hypermedia tool that meets those requirements. From here on out, we will refer to the LinkWay version as MSRT-L and the version based on ALM and written in C as MSRT-C.

It should be noted that the C programmer had many hours of experience with the original ALM code. He had written several alternative implementations of the ALM program prior to developing his MSRT tutor. The LinkWay® programmer had extensive experience in developing ITS and using hypermedia tools such as Apple's HyperCard and Asymetrix's Toolbook, but had no experience with LinkWay®.

Functionality of the Systems

The MSRT is essentially made up of several pieces of equipment (a radio, a phone, a power supply and various connections). Each piece of equipment had push buttons, toggle switches, knobs, and/or light indicators. Part of the ITS development and its functionality was to create a simulation that allowed lights to "light up" and allow the learner to throw switches, push buttons, and turn knobs. It turns out that the most important aspects of the simulation for learning were related to lights and sounds. That is, the lights blink, go on, or go off according to what the learner did to the knobs, switches and buttons. In addition, the radio and phone produce various tones as the result of actions performed by the learner. Much of the development time was spent on getting these equipment reactions to accurately correspond to the actions performed by the learner while allowing the learner the freedom to make errors as they attempt to operate the equipment. This freedom was constrained to a degree. Certain knobs, switches and buttons were not used at all in some procedures. If the learner were to click on these, they would receive an error message. Also, certain sequences of actions were required. If the learner performed an action out of sequence, an appropriate error message was provided.

In addition to the error feedback and the simulation programming, there were two other primary foci of the development process -- the "Coach" button programming and the navigation requirements of the system. First, the Coach button is probably the most "intelligent" part of these systems. When a learner clicks on the Coach button, the program needs to determine the most appropriate action that the leaner needs to take given the goal of the problem. This achieved through a collection of state variables and solution step variables. Essentially, every knob, switch and button has a state variable associated with it. As the item is altered by the learner, the state variable is updated. In addition, each step of the procedure is represented by a variable that indicates whether the step has been completed or not. In this way, the program can know that the learner may have set a switch to off at an appropriate time (the step variable), but later changed it to on (the state variable). This state change might then cause a conflict in the equipment (perhaps an electrical shorting of the equipment). The Coach button therefore, might check that the knob is in the correct location and that the preceding step had been completed before suggesting that the student execute a step that might cause a problem.

The final aspect of the system that requires some time is to establish links between screens. These links can be in the form of transparent buttons over pieces of equipment or buttons on the bottom of the screen that indicate the piece of equipment (say the phone). In this way, the learner can quickly and easily move from one device to the next without much cognitive effort and therefore, little interference with the carrying out of the procedure. These are the common development efforts between developing in C and LinkWay. There were also graphics that needed to be created, but just one person did this and the two systems used them. The C programmer had created these graphics prior to beginning the procedures, so times are for the tasks mentioned above, and do NOT include the time taken to create graphics. The graphics took approximately 80 hours. The images were scanned into the computer, edited, and colorized. There was a total of 10 images and it took about a day to do each one.

Limitations of MSRT-L

1. There is not much control over the placing of buttons. The vertical displacement is approximately 20 pixels and the horizontal is about 10 pixels. This was a problem when trying to place buttons over the drawings that represent the MSRT equipment. LinkWay Live (the new version) allows pixel by pixel movement (this project began with LinkWay 2.01 and finished with LinkWay Live).

2. The Paint program would not allow you to select more than about 20 percent of the screen at a time. So if you wanted to move the image to the right a bit, you could not (or shrink it or any other manipulation).

3. While LinkWay has built in utilities for using sound boards from IBM, it does not have any ability to make differing tones (something quite useful when building a phone tutor). The only sound resource available was to play a simple beep. However, you could write the tone routine in another language and run it from within LinkWay (which is what was done).

4. One of the ways LinkWay handles the limitations of a 640K environment is to limit the objects and scripts associated with those objects. The limit is 10K of memory for each screen. This limit was reached several times. The recommended solution is to use multiple screens to represent the necessary actions for that screen. Therefore, you might have half of the knobs representing the Digital Secure Voice Terminal (DSVT) on one screen and clicking on any of the other half of the knobs would send you to another screen that has those knobs represented. This was deemed a poor alternative. We found that the literals associated with error and status messages could be provided on a separate screen without limiting the simulation fidelity. Therefore, pop up messages were eliminated and messages were provided by taking the learner to another screen to display the text. Clicking on anything would return them to the equipment on which they had been working.

5. Another limitation is that each object is limited to 4K of script. However, other objects can contain procedures that the object can then call. The only problem that this caused was that everything that was added to a script could get lost (in our case the most lost was about ten minutes).

6. The programming utilities were minimal. You could not find words, replace words, select pieces of code, debug code, or move code easily. Copying a piece of code from one place to another could only be done one line

at a time and the manual did not indicate that this could even be done (the LinkWay programmer learned from a friend who has used LinkWay a great deal how to copy code and delete more than one line of code).

7. Apparently, as the page size increases, strange things begin to happen. One of those things is that when you leave an MSRT screen to go another (say Advice) screen, some times a box corrupts the screen. This box is where the mouse cursor happens to be. The only fix is to hide the mouse icon before leaving the screen and show it after displaying the next screen. This is cumbersome and makes a link button useless.

8. Another problem that has arisen manifests itself while simulating a flashing light. Frequently, during the procedures of operating the MSRT a flashing light appears on the equipment. This was simulated by placing a button over the light and alternating a filled circle and open circle icon on top of the light. However, if the learner moves the mouse pointer into the flashing light area, stray lines appear in that area. As long as the mouse is in that area, these lines continue to appear. Again, the fix is to hide the mouse icon before changing the icon of the light and showing it after the light icon is displayed (or use Bitputs instead of icons).

9. LinkWay uses its own graphics formats. This is very IBM. Rather than use one of the many standard formats, IBM chose to use its own. Hence, we needed to use a graphics utility to display the PCX file and then use LinkWay's screen capture utility to save it in LinkWay's format. However, we wanted the system to be in a EGA format and the machine we were using had a VGA monitor. Every time we tried to capture a screen, it saved it in VGA format. The solution was to find and use a machine that had an EGA monitor.

10. The ALM architecture allows the user to either click the mouse button or press a function key (although this is not support for the interaction buttons like a knob for example). There is no apparent way to have a button execute with either a mouse click or a specific function key press. Buttons only execute when they have been activated by a mouse click.

11. The ease of development comes at the expense of loss of control. Many things are automatically handled by LinkWay (the way the mouse cursor looks and behaves, how buttons look and behave, how screens are displayed, how scripts are executed). It may occur that you do not want these things to happen the way that LinkWay handles them. In ALM, these things are modifiable. In LinkWay, it is unclear if they can be changed. For example, the mouse cursor changes from an arrow to a hand when it comes into the area of a button in LinkWay. This may be too subtle for the learners to react to. In ALM, when you encounter a button, a square area is depicted around the "hot" area of the button. It clearly indicates that this is a button.

Limitations of MSRT-C

1. The way that screens are drawn within MSRT-C is that the system calculates what goes on the new screen, then it removes what is currently on the screen and pops up the new screen. This takes about 3 seconds on a Zenith 286 EGA system. What this means is that the learner clicks on a button to go to the next screen and nothing happens for 3 seconds. During that delay, the learner could try to do something else. The result is the first thing that was clicked gets executed at the time the second thing is clicked. This causes some confusion.

2. A second problem in MSRT-C is that buttons (their location, size, and nature) are defined in a textual resource file. This means that to place an MSRT-C button on top of a picture of a switch, you need to guess about the screen coordinates of its location and size, enter this information, run the program, go to the correct screen, see where the button actually showed up, quit, adjust parameters, and do it again. In MSRT-L, you place and size it on the screen with the picture already displayed. This is much easier and faster.

3. When you want to write the program that attaches to the button you just placed, you need to do the following. First, you need to create the case instance to turn control over to the procedure that you need to write for the button. Second, you write the procedure for the button. Third, you recompile the code and relink it. Fourth, you run the program and test the button to see if it works. Last, you revise the procedure and do it again. In MSRT-L, when you create a Script button, you are provided a small editor and you can write the code to be executed. When you click on the close box, you can then double click on the button to see if the code works and revise it accordingly. This is tremendously faster.

4. When you want to add an additional screen to the tutor, MSRT-C uses a similar process to the one for adding a program to a button. The programmer for the MSRT-C tutor estimated that the act of creating a new screen (if the graphic is already created) is about 20 minutes. The process includes copying the screen loop code, writing the case instances for each screen that would call the new screen, write any code that needs to execute when the screen first displays (like current icon settings), and creating definitions for the screen in the resource file. In MSRT-L, it took 26 seconds to do the same thing. Over a large project this can be a big difference.

5. The ALM architecture is not an event driven object oriented architecture like LinkWay. The event loops are separate loops with giant case statements that have to be modified with each addition and deletion of objects in the program. When you delete an object in MSRT-L, you also delete its code. There is no clear way to do this in MSRT-C. You have to delete code in many different parts of the program and resource files.

6. MSRT-C is not an integrated environment. However, there are several tools that can be integrated. For example, you can leave MSRT-C and start PC Paint to create your graphics. Both MSRT-C and MSRT-L have poor icon editors. Multiple Fonts are not possible in MSRT-C unless you paint them on in another package.

7. When you reach a memory limit in LinkWay, you lose the code you were just working on and LinkWay informs you that you have reached a limit. MSRT-C reaches a limit and system crashes occur at non-systematic times. This makes MSRT-C a much less stable environment. There have been cases where a small modification to the program required a week of reprogramming to get the system to recompile and run again.

Summary

Although there were many limitations to the LinkWay environment, most of the limitations could be resolved in some way (which implies that they are not limitations). Our overall impression of this programming environment is positive. It is a powerful hypermedia development tool that meets the needs of the Army. The MSRT-L tutor runs in 640 K, displays on an EGA monitor, and fits on one 3.5 inch diskette. These are all positive aspects. Development time, however, is the overwhelmingly positive aspect.

Time Comparisons

The times that it took to develop each version of the MSRT tutor were quite disparate. It should be noted that a conservative estimate for the time that it would take to achieve mastery of the procedures that were simulated (there were four in total) is about 3 hours of instruction. The programs were essentially the same. They covered the same procedures and had the same screens. Factoring out the graphics production, the MSRT-C tutor took 180 hours to develop using the existing C libraries. Included in this estimate is the time taken editing the icons so that all the knobs, switches and buttons could be represented.

The MSRT-L programmer was required to do the same tasks. The only difference was that this programmer also had to learn LinkWay. The outcome is that it took him 75 hours to create the same system. It took the C programmer 2.4 times longer than the LinkWay programmer. This in itself is reason to choose to use LinkWay over C to develop these programs. The functionality of these systems was essentially the same. If anything, the MSRT-L version was more flexible. For example, the MSRT-L version would allow the learner to set a knob in an incorrect position. The MSRT-C version would inform the user that an error occurred if they moved the knob beyond where it was supposed to be set. The former is more like the actual equipment and therefore, higher transfer would be expected.

t is estimated that the graphics took about 80 hours to create. Combining this estimate with the development time for the MSRT-L version, the total time to develop the MSRT-L tutor was 155 hours. Given the conservative estimate of 3 hours of instruction, this translates to about 50 hours of development time for every hour of instruction. This is a considerable savings over the 160 hour development time that was involved in POSIT (Orey & Burton , 1989/90). In fact, instructional developers ought to take notice. ITS is a doable technology. It is time to take the ITS development out of the research labs and put it into the development labs.

COMPARING THE RESULTING INTELLIGENT TUTORING SYSTEMS

The above comparisons were conducted by the research and development staff at the Fort Gordon Army Research Institute. In addition to this subjective comparison, we conducted a formal evaluation that compared the two systems. This comparison included subjective measures and objective measures. It should be noted, however, that the two systems were implemented by two different developers and specific design decisions were implemented differently. The most apparent difference in the systems manifests itself in the form of the advice statements provided by the Coach. Anderson (1990) has suggested that when people learn procedures, that they go through a variety of stages and use a variety of processes. The basic idea is that the learning of procedures proceeds from the small verbalizable component parts of the procedure to the automatic (non-conscience) execution of the procedure. During the initial learning of these component pieces of the procedure, processes can be used to intervene or aid in the acquisition of new components. For example, if the learner is trying to execute one of the MSRT operations procedures and they know that they need to connect one piece of equipment to another (but they have not done this before), they may use knowledge about how to connect other pieces of equipment (say a VCR to a TV) to aid the acquisition of the new component. In this way, the learner reasons from a known example to the new example. This process aids both the execution of the procedure as well as the retention of the new component of the procedure.

When building an ITS that provides advice, this theoretical idea is very important. Essentially the Coach function in the MSRT tutors determines the next appropriate component of the procedure that the learner needs to execute. It then needs to provide the learner with this information. Both MSRT tutors do this. However, the nature of the messages provided differ. The MSRT-C developer decided to provide strictly procedural information. For example, at one point in a procedure, the learner is required to remove a piece of equipment from another. There are several component parts of this procedure. The MSRT-C version of the tutor provides this message:

Set KYK-13 Z/ON/OFF CHECK switch at OFF CHECK. (Go KYK-13 screen).

This is the first step to executing this procedure. The MSRT-L version provides this message for the same step:

Now return to the KYK-13 screen and begin to disconnect the cryptovariable cable. This process begins with the setting of the KYK-13 to the OFF position.

The intention of the first developer was to be direct and limit the message to strictly procedural information. The intention of the second developer was to tie the components conceptually together.

A second difference between the two versions was in the ability to navigate through the system. The original approach to navigation was to present a battlefield map that indicated a variety of communication equipment. The learner would then click on the MSRT and be shown typical layout of all of the MSRT components. The learner could then click on the piece of equipment that they wanted to work with and the system would take them to a screen depicting that device (with all of its lights, buttons, switches, and knobs). The learner could then perform any operation on that device that was appropriate at that time for that procedure. When they needed access to another device, they would click on the "backup" button and it would show the main MSRT screen again. The MSRT-C developer decided to include buttons on the individual screens that would allow the learner to go to another device without returning to the main MSRT screen. However, because it is difficult to alter screen layouts in the MSRT-C version, these buttons may be at the top of one screen and on the right side on another screen. Further, not all devices were made available from each device. This is not necessarily good interface design, but it was easiest way to implement this change.

The MSRT-L developer decided that the idea of being more "hyper" about access to equipment was a good idea. However, in about half an hour, the MSRT-L developer created a button for each screen on the base page and separated these buttons by placing them in a color coded area at the bottom of the screen. What this means is that the learner could now get to any device from any other device with one mouse click. Beyond this difference, the two systems were essentially the same. Figure 1 depicts a typical depiction of a device for the two versions. Given these differences, we conducted the following evaluation.

METHOD OF EVALUATION

Participants

he participants for this evaluation were 24 (13 in the MSRT-C group and 11 in the MSRT-L group) Signal Corp soldiers (enlisted) that were receiving training on basic communication equipment. Ages ranged from 18 to 35 years and averaged about 21 years. Education ranged from twelfth grade through completion of 2 years of college and the average was about a half year beyond the high school diploma. Because the collection of data took place over a period of several weeks, participants were at varying points in their communication equipment curriculae. They ranged from the fifth week of training to the ninth week of training and averaged about six weeks. All soldiers were required to know how to operate the MSRT and therefore, the participants had an interest and need in learning from the systems. The abilities of the soldiers in the two regroups were roughly equivalent. The MSRT-L group (m=111.2, sd=4.98) had slightly higher IQ estimate than the MSRT-C group (m=110.2, sd=7.88). This measure is described below. Further, the correlation between this ability measure and the performance measures used was quite small (the largest correlation was -0.19 with the time spent on the first procedure).

Materials

The two tutors collected specific data. They saved what procedures were worked on, how long it took to solve the procedure, how many errors were made, and how many times the participants used the Coach button. The errors, advice, and time on the procedure were the main objective measures of learning. Besides the two tutors, there were three other instruments used in the study. To make sure that groups were of equal ability, the Shipley measure of ability was used. The Shipley has a test-retest reliability of 0.80 and correlates with the WAIS intelligence quotient at 0.90. In addition to this ability measure, two questionnaires were constructed. The first asked eight four-choice questions (bad, poor, good, excellent) related to the tutors. This was followed with four open ended questions about the tutors and a general comments section. The second questionnaire was the same as the first, except that it included a third page that addressed the comparison of the two tutors. It also included two items about how much experience with computers they had (none-monthly-weekly-daily) and the other asked what type of computer they used.

Procedure

The idea of the procedure was suggested by Richard Goodis (an employee in the ARI lab). The basic idea is to get people to learn one of the tutors and then have them evaluate both systems. They probably would become attached to the system that they learned on and would likely not evaluate the other system as highly as "their own". However, if the system they learned on was much poorer than the other system, then they might rate the other system higher. Essentially, this is the procedure that was used. First, the participants went through each of the four procedures twice and in sequence (approximately 80 minutes). At this point, we asked them to do the first procedure again. This was the third time they did this procedure and the time to complete, errors, and accesses to coach were the performance measures that we used as objective measures.

After completing the learning phase and on-computer testing, we asked them to fill out a questionnaire that evaluated the system they learned on. Having completed the questionnaire, we then asked them to examine a different tutor (the other tutor) by performing the first procedure again. We also asked them to do this keeping in mind that they were going to be asked to compare the two versions. After completing the procedure, they were again asked to fill out a questionnaire which evaluated this "new" system.

Analysis

There are a variety of comparisons that were conducted on these data. First, t-tests were used to determine if significant differences were to be found between the performance of the two groups on the first procedure and the test procedure. The very first procedure they solved would give an indication of how easy the system was to learn. The test procedure (third time the first procedure was solved) was used to see how well they learned the

procedure from the system. An ANCOVA was considered if the random assignment failed to yield equivalent ability groups, but since they were considered equivalent, a series of t-tests were used.

For subjective comparisons, t-tests were run on each of the 16 multiple choice questions (there were 8 of these questions on each questionnaire). The open ended questions were analyzed by frequencies of categories of responses. In other words, the responses were listed and similar comments were grouped and counted. The most common comments and a selection of others are reported here.

RESULTS AND DISCUSSION

To begin the discussion of results of the evaluation, we will examine the objective measures. There were essentially three performance measures at three points in the evaluation (a total of 9 t-tests. see Table 1). The first point when we collected the performance measures was on the very first procedure that the learners solved. The reason for including this measure is that this would give an indication of the ease of learning the operating mode of the intelligent tutoring system (not necessarily learning from the system). As can be seen from Table 1, all of the performance measures with regard to the first procedure are significant at the 0.05 alpha level. The MSRT-L group took less time, used the coach button less frequently, and made fewer errors than the MSRT-C group. The reason for this quite likely is due to the difference in the structure and content of the text messages. There are 15 steps in the first procedure. The MSRT-L group used the coach button on the average 18 times. Therefore, assuming that the soldiers had no prior knowledge of MSRT (which they did not), there is almost a one-to-one correspondence between the coach button and the number of steps. The MSRT-C group, on the other hand, needed to access the coach button 26 times. One explanation is that the coach messages in the MSRT-C version were more difficult to understand than the MSRT-L version. Another possible explanation is that the navigation buttons in the MSRT-L version made it easier to understand how to execute the message that was provided by the coach button.

The fact that the MSRT-C group also made a large number of errors on the first procedure (14) relative the MSRT-L group (4) does not help decide which explanation is more plausible. The errors may have occurred as the result of misunderstanding the message or may have resulted from misunderstanding how to carryout the action described in the message within the two alternative systems. The differences in time do not shed any more light on these alternative explanations. After all, the MSRT-C group was busy making errors and asking for advice from the coach over and over again. Of course it would take them longer to finish the procedure. The differences in the two designs are related to the two development environments. It is much more straight forward to create navigation buttons that are easily understood and create large elaborate text messages that can be used by the coach function in MSRT-L than in the MSRT-C architecture.

Another way of determining ease of use is to look at the objective measures when the groups switched to the other system. After learning each of the operations procedures from one system (for a period of approximately 80 minutes) the participants were asked to examine another version (the other version). Therefore, the MSRT-L group examined the MSRT-C version and the MSRT-C group examined the MSRT-L version. When looking at the performance data from the "other" version, only one measure was significant at the 0.05 alpha level (see Table 1). That measure was the frequency of accessing the coach button. The MSRT-L group (using MSRT-C, about 18 times) used the coach button more frequently than the MSRT-C group (using MSRT-L, about 12 times). The MSRT-L group returned to a level of coach use that was nearly as high as when they first used the MSRT-L version. They used coach about 18 times on the first procedure and 17 times on the other system, even though this is the fourth time they have performed this procedure. The data from the test procedure (the third time they performed the first procedure) show that they only used the coach button about 12 times on average. The MSRT-C group only slightly increased there use of coach (about 2 extra accesses) from the third time (same system) to the fourth time (different system). This may indicate the ease of use of the MSRT-L version is the explanatory factor rather than the differences in the text messages.

Table 1. Tables of means, standard deviations, t values and probabilities for each of the performance measures at each learning point in time.

	Groups					
	MSRT-L				MSRT-C	
	Mean	s.d.	t	p	Mean	s.d.
First time	939.00	317.17	2.07	*0.05	1295.92	492.30
First coach	18.64	7.37	2.03	*0.05	26.39	10.64
First errors	4.00	3.66	3.58	*0.00	14.39	8.99
Test time	242.27	96.35	-0.47	0.65	225.08	83.80
Test coach	11.91	4.01	-0.70	0.50	10.54	5.40
Test errors	0.64	1.21	2.62	*0.02	2.31	1.80
Other time	740.82	246.25	0.39	0.70	661.62	636.27
Other coach	17.73	6.68	-2.34	*0.03	12.15	4.98
Other errors	7.00	4.47	-1.95	0.06	4.15	2.58

Note: Times are in seconds. Errors and coach accesses are frequency counts. First indicates first procedure. Test indicates the last procedure on the system on which the subject learned. Other indicates the procedure solved on the system opposite from the one on which they learned.
*significant at the 0.05 level

Recall that the first procedure requires 15 different steps to successfully "power up" the MSRT. Both groups had reduced their access to advice by the third time to less than the number of steps. Therefore, they must have remembered the other steps and could rely less on the coach button. When they were introduced to the "other" version, the MSRT-C group was able to perform some of the steps without asking for help from the coach button (they used the coach button 12 times on average). A possible explanation is that the MSRT-L version was easy enough to understand that they could simply carryout their learned procedures in the new environment. Another explanation is that the MSRT-C version is better for transfer. However, combined with the subjective data, this seems like an unlikely explanation. Therefore, the best explanation considered to this point is that the MSRT-L version is easier to operate.

A final comment on the data collected on the "other" system is that the pattern of results is similar to the first procedure. That is, the MSRT-L group took longer, used coach more frequently, and made more errors than the MSRT-C group when they were using the "other" system. While time and errors were not significant at the alpha level, it is interesting to note that the pattern of results based on comparing means is similar when people use the MSRT-C system. Again, this can be attributed to an explanation which is based on ease of use.

The final group of performance data are those related to the "test" phase of the procedure. After having performed each procedure twice, the subjects were asked to perform the first procedure a third time. This was used as a measure how well they had learned this procedure. The only statistically significant result was that the MSRT-L group made fewer errors than the MSRT-C group. A probable explanation for this is that the MSRT-L version displayed errors messages on a bright red screen that had a title at the top stating, "Errors". The MSRT-C version displayed a window with a red background and no clear note that the message was an error message. In addition, the MSRT-C version used the same error message for all errors. The MSRT-L version used error message specific to the error that was committed (i.e., the MSRT-L version was more ITS like than the MSRT-C version). The reason for this is that the error handling is difficult and time consuming in the MSRT-C version. It is much easier to implement in the MSRT-L version. Therefore, MSRT-L may be better suited for ITS development than ALM. The other performance data do not indicate any differences. It took both groups about the same amount of time and used coach with almost equal frequency when performing the start up procedure.

Table 2. The means, standard deviations, t values, and probabilities for each question both for the system the subject learned on and the "other" system.

	Groups					
	MSRT-L				MSRT-C	
	Mean	s.d.	t	p	Mean	s.d.
Learned on						
Interface	3.64	0.67	1.41	0.17	3.23	0.73
Access	3.82	0.41	3.50	*0.00	2.92	0.76
Realistic	3.55	0.52	1.94	0.07	3.08	0.64
Coach	3.64	0.67	1.46	0.16	3.15	0.90
Enjoyment	3.46	0.93	1.37	0.18	2.92	0.95
Errors	3.09	0.83	0.52	0.61	2.92	0.76
Confidence	3.46	0.52	1.59	0.13	3.00	0.82
Overall	3.64	0.67	0.64	0.53	3.46	0.66
Other system						
Interface	2.55	0.93	2.81	*0.01	3.46	0.66
Access	2.00	1.00	5.77	*0.00	3.77	0.44
Realistic	2.91	1.14	0.89	0.38	3.23	0.60
Coach	2.55	1.04	2.22	*0.04	3.31	0.63
Enjoyment	2.18	1.17	3.18	*0.00	3.31	0.48
Errors	2.36	1.29	1.85	0.08	3.08	0.49
Confidence	2.36	1.12	1.79	0.09	3.00	0.58
Overall	2.09	1.05	3.59	*0.00	3.23	0.44

Note: The means are based on an integer scale from 1 to 4 where 1 is bad and 4 is excellent. The two important categories are the evaluation of the system on which the subject learned and evaluation of the other system on which they did NOT learn.

In summary, both the performance data from the first procedure and the "other" procedure indicate that MSRT-L is easier to use than the MSRT-C version. Further, on the learning measures, MSRT-L may be better for learning than the MSRT-C architecture. Therefore, the MSRT-L version is easier to use and students learn better from it. We now turn to the question of preferences based on the questionnaire data.

The results of questionnaire component of this evaluation are presented in Table 2. To begin this discussion, refer to the first group of questions. These are labeled "Learned on". After the subjects completed using the first system that they learned on, they were asked to fill out a questionnaire. This section represents the results of this phase of the procedure. If you examine the column of probabilities, the only significant result is that MSRT-L group rated the MSRT-L version higher when asked about how easy it was to go from screen to screen. As described above, this was easily done in LinkWay. Not only was it easy to program, but it makes it easier for the learners to navigate through the system. The only other observation to be made with regards to the first group of questionnaire responses is that although the other areas do not indicate a significant difference between groups, the means for each question are higher for the MSRT-L group than the MSRT-C group. While this is not statistically significant, it certainly suggests an interesting pattern, especially given the results of the second part.

After the subjects had used one version for about 80 minutes, they were asked to examine the "other" system. They were asked to perform the start up procedure for the fourth time, except that this time it was on the other system. In addition, they were told that they would be evaluating this system using similar (to the first one) questionnaire. The results of this questionnaire are in the second part of Table 2.

The first point to be made about the second part is that the subjects now had a basis for comparison. They could compare this "other" version to the one they had learned on. Prior to using the "other" system, most of the subjects had not used a computer-based ICAT simulation. Having seen both systems, they had a basis for making comparisons.

The first thing to consider in these results is the number of questions that are within our 0.05 alpha level. Of the 8 questions, 5 are within the level. The overall pattern is the same as the first set of questions -- the participants rate the MSRT-L version more highly than the MSRT-C version. The difference is scale. The extent of the difference is greater. The five specific questions are -- 1) Ability to interact with knobs, switches, etc.; 2) Ability to go from screen to screen; 3) Information provided by coach is sufficient to solve each step of the procedure; 4) As a training device, how enjoyable did you find the program; 5) Overall rating of the program. Two of the other questions that had a big difference, but not a statistical difference were -- 1) How helpful was the text of the "red" error messages?; and, 2) How confident are you that you could start up and use a MSRT stand alone? Clearly, the subjects preferred the MSRT-L version.

As was mentioned when describing the methodology described here, it was thought that people would "bind" with the system that they originally learned from. For MSRT-C group this is somewhat true. If you look at the overall rating of the programs the MSRT-C group rates MSRT-C an average of 3.46 and MSRT-L 3.23. They seem to be equally confident in the two versions rating them both an average of 3.00. However, on every other aspect, the MSRT-C group rates the MSRT-L version higher than "their" version. We use the plural possessive form for the MSRT-C group's relationship to MSRT-C because of the preconceived idea that the subjects would have developed an allegiance to the system they spent all that time on. To some degree, this is true for the MSRT-C group. However, they rate MSRT-L better on every aspect of the system (the first six questions), so this binding is not that strong. If there is a binding that occurs, it is clearly demonstrated in the MSRT-L group's responses.

On virtually every aspect of the questionnaire, the MSRT-L group rated the MSRT-C version lower than "their" version. The range of difference are from a low of 0.64 for how realistic the program was to a high of 1.82 for how easy it is to go from screen to screen. Clearly, the MSRT-L group did not like the MSRT-C version. In the previous discussion, the MSRT-C group had a certain attachment to the MSRT-C version as indicated in their overall rating, but after that they indicated that they preferred all the attributes of the MSRT-L version. Apparently, the magnitude of the difference between the two systems was great enough to overcome any sentimental attachments to the system they had learned on.

In the questionnaire, along with rating the systems, subjects were also asked to comment on what they liked, disliked, changes to the systems, confusing aspects and general comments. (Appendix). The evaluation of the systems "learned on" first were similar, with most subjects enjoying this method of instruction to include the use of the coach and the realistic presentation of the equipment. Furthermore, with the exception of the slowness of the ALM environment, a majority of the subjects reported no need for changes to these environments and found the environments to be an enjoyable means of training.

However, when asked to evaluate the "other system" differences were noted in that most subjects disliked the ALM system; whereas those whose "other system" was Linkway found it to be quick and easy. Additionally, those who used Linkway as the "other system" were more likely to report that they disliked nothing in the program and found no need for changes than those who used the ALM environment as the "other system". Moreover, all the subjects who initially learned on Linkway preferred and found it easier, while only 6 of the 13 who learned on ALM preferred it and 5 found it to be easier.

CONCLUSION

The results of this project all tend to support the idea that off-the-shelf hypermedia packages can be used to streamline the development time for ICAT systems. The functionality comparison indicated some initial problems with re-purposing LinkWay for ICAT development, but most of these obstacles have been overcome. In the end, the MSRT-L version had all of the functionality that the MSRT-C version had, if not more. A second consideration is whether the off-the-shelf version is as effective as building the system within a programming language. The answer is that not only is it as effective, but perhaps more effective. If it is more effective, it may be due to the fact

that it is easier to use. All of our additional performance data suggest that the system is easier to use. Finally, the question of whether there may be some affective reason not to use an off-the-shelf package has been put to rest by the evidence presented here. In summary, the off-the-shelf version was more effective, easier to use, the subjects preferred to use it, and it took less than half the time to build it. We would recommend that people considering the development of an ICAT system ought to consider building it in an off-the-shelf hypermedia system.

Other systems, other content areas, and other needs might not permit the adoption of the decision to use an off-the-shelf system, but there is more to consider. Towards the end of this project, we decided that since we had implemented the system in LinkWay Live, that we might try building a multimedia version of the tutor. In not much time (a couple of weeks), we had a version that would play a movie with narration on a screen that described each procedural step in a text format when the user clicks on the coach button. This multimedia version would not have been considered if the system had been developed in only in C. In addition, we built a gaming environment that required the learner to complete a procedure without error, without using coach, and within a time limit in order to transfer from one army base to another (from Fort Industrial Waste to Fort Fantasy for example). Again, these changes were quite simple to implement and may have taken a total of two weeks to have a bug free version working.

There are two issues that ought to be discussed before closing this paper. First, many people will think that the off-the-shelf system clearly is the way to go and could have told me before I began. Unfortunately, there was no clear evidence that ITS type systems could be developed and could be as effective as systems that are built in more flexible environments like C. Now there is. Second, some people might suggest that the differences that we found between these systems could have been removed if the MSRT-C program was designed exactly like the MSRT-L program. This is true, but that would have added more time to the development of that system and it still would take longer even if the system had similar programming implementations. The implementations that were programmed in the C architecture could be revised and a system could be build that does exactly what LinkWay Live does, but why would anyone want to do that? It has already been done.

We should add a word of warning about time estimates for this development. The programmer who worked on the MSRT-C version took over the development process when the data collection began. We brought in a Subject Matter Expert (SME) and added two more procedures. This process included rewriting the program so that sounds were correct and lights flashed, when on, and went out as appropriate as the learner interacted with the system. All totaled, they worked on the system an additional 250 hours to make it bug free and subject matter correct and complete. The program now would take about 5 hours for someone to achieve mastery. Combining the 75 hours of initial effort with this 250 hours, the total time was 325 hours or 65 hours of development for each hour of instruction. However, a great deal of time could have been saved if the SME had worked on the project during the design phase. It is much easier to implement a design than it is to make large scale changes to an already implemented program. Nonetheless, 65 hours is still an improvement over the 800 hours some suggest it takes to make good quality ICAT systems.

REFERENCES

Anderson, J.R. (1988). The expert model. In M. Polson & J. Richardson (Eds.), *Foundations of intelligent tutoring systems* (pp. 21-53). Hillsdale, NJ: Erlbaum.

Anderson, J.R. (1989). The analogical origins of errors in problem solving. In D. Klahr & K. Kotovsky (Eds.), *Complex information processing: The impact of Herbert A. Simon* (pp. 343-372). Hillsdale, NJ: Erlbaum.

Merrill, M.D., Li, Z., & Jones, M.K. (1990). Limitations of first generation instructional design. *Educational Technology, 30*(1), 7-11.

Jonassen, D., & Wang, S. (1991). *Building an intelligent tutor from hypertext and expert systems.* Paper presented at the annual meeting of the Association of Educational Communications and Technology. Orlando, FL.

Orey, M.A., & Burton, J.K. (1989/90). POSIT: Process oriented subtraction interface for substraction. *Journal of Artificial Intelligence in Education, 1*(2), 77-104.

Orey, M., & Nelson, W. (in press). Development principles for intelligent tutoring systems: Integrating cognitive theory into the development of computer-based instruction. *Educational Technology Research and Development.*

Orey, M.A., Park, J.S., Chanlin, L.J., Jih, H., Gillis, P.D., Legree, P.J., & Sanders, M.G. (1992). High bandwidth diagnosis within the framework of a microcomputer-based intelligent tutoring system. *Journal of Artificial Intelligence in Education, 3*(1), 63-80.

Rosenburg, R. (1987). A critical analysis of research on intelligent tutoring systems. *Educational Technology, 27*(11), 7-13.

Towne, D. (1991). *Instruction in a simulation environment: Opportunities and issues.* Paper presented at the annual meeting of the Intelligent Computer Aided Training conference. Houston, TX.

APPENDIX

LEARNED ON ALM*	
LIKED	
interesting way of learning	4
directions the coach gave	2
graphics	2
improves one's ability to operate a computer	1
working at own pace	1
easy to understand	1
realistic	1
nothing	1
this training aid works	1
DISLIKED	
wait between screens	5
some of the instructions were confusing	2
nothing	1
F6 action couldn't be displayed on every screen	1
prefer hands on	1
CHANGES	
none	7
make it faster	2
better instructions when first start	2
reword coach instructions	1
voice messages	1
put F-6 ACTION button on every screen	1
CONFUSION	
Go To F-6 ACTIONS	2
coach	2
environment difficult at first	2
"having to jump around on the equipment"	1
COMMENTS	
good training device	2
no explanation of the purpose of this program	1
prefer hands on instruction	1
LEARNED ON LINKWAY*	
LIKED	
easier to learn with computers than with instructors	4
coach	3
equipment realistic	2
fine program	1
easy to understand	1

the sequence of instructions	1
DISLIKED	
nothing	8
red error messages	2
checking all of the lights	1
switching screens	1
preferred hands on	1
had to follow the sequence	1
CHANGES	
none	7
timing between screens	1
red error messages	1
put KYK-13 on same screen as DSVT	1
make the program faster	1
let the trainee know his/her mistakes	1
switch to turbo ball	1
none	10
beginning needs more instructions	1
COMMENTS	
enjoyable	4
ALM AS OTHER SYSTEM*	
LIKED	
nothing	10
similar to first	1
coach	1
liked everything	1
DISLIKED	
wait between screens	7
didn't understand directions	6
outlined boxes confusing	1
everything	1
location of instructions	1
CHANGES	
make it faster	2
throw it away	2
make it user friendly	1
get trained by someone else	1
make it more like the first program	1
getting "fill cap off" was difficult to find	1
better information about directions	1
better graphics	1
CONFUSION	
some directions	4
nothing	3
map	1
coach complicated and indirect	1
buttons hidden	1
screen layout	1

everything	1
COMMENTS	
program was bad	3
prefer Linkway	3
too frustrating	1
both programs were excellent	1
LINKWAY AS OTHER SYSTEM*	
LIKED	
easy	6
quick	4
coach	1
fun	1
DISLIKED	
nothing	6
seemed like there were more steps in this program	1
hot spots not outlined	1
didn't like the way you had to find things	1
there was too much information in coach	1
equipment not precise	1
different from ALM	1
CHANGES	
none	7
make coach more understandable	2
outline hot spots	1
make the knobs on the DSVT more precise	1
CONFUSION	
none	6
coach difficult	2
some places used lights and other places used switches	1
beginning needs more instructions	1
the computer	1
the beginning about the antenna	1
COMMENTS	
good program	1
ideal teaching device	1

* Not all subjects chose to respond; some gave more than one response.

Michael Orey is an assistant professor at the University of Georgia in Instructional Technology. He is interested in emerging technologies for learning.

Ann Trent is a consortium student with Augusta College and the Army Research Institute and is interested in experimental psychology.

James Young is a consortium student with Augusta College and the Army Research Institute and is interested in Computer Science and Intelligent Tutoring Systems.

Michael Sanders is a research psychologist and is the field unit chief at the Army Research Institute at Fort Gordon. He is interesting in training innovation.