DOCUMENT RESUME

ED 366 325 IR 016 531

AUTHOR Hyltin, John P.; And Others
TITLE Digital Video Interactive (DVI) Based Authoring Tools for Unit Level Training. Final Technical Report for Period 27 September 1991 - 27 May 1992.
INSTITUTION Betac Corp., San Antonio, TX.
SPONS AGENCY Armstrong Lab, Brooks AFB, TX. Human Resources Directorate.
REPORT NO AL-TR-1992-0159
PUB DATE Mar 93
CONTRACT C-F33615-90-C-0010; PE-65502F; PR-3005; TA-I2; WU-01
NOTE 34p.
AVAILABLE FROM AL/HRPP, 7909 Lindbergh Drive, Brooks Air Force Base, TX 78235-5352.
PUB TYPE Reports - Research/Technical (143)

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS *Authoring Aids (Programming); *Computer Software Development; Military Training; *Multimedia Instruction; *Programming; Programming Languages; Training Methods
IDENTIFIERS Air Force; *Digital Technology; *Digital Video Interactive; Interactive Systems

ABSTRACT

This report describes DVI (Digital Video Interactive) technology, current authoring languages and tools, and the reasons for developing new tools and applications. The work described was performed by Betac Corporation as part of a Phase II Small Business Innovation Research project. Section I provides background information on DVI. DVI technology is described in more depth in Section II, including ActionMedia hardware, ActionMedia production tools, and ActionMedia software libraries; the next section examines authoring languages and tools, including existing systems and simulation requirements in authoring packages. Section IV focuses on the software development approach, including previous developments, generic tools and objects, application specific tools and objects, development methodology, and the Intelligent Tutoring Strategy approach. A Life Support Officer training course using DVI is described in Section V, including training needs assessment, a life support equipment investigations course, resources for course development, target student population, teaching philosophy, and the course outline. Section VI summarizes lessons learned, and the final section discusses future directions, including object libraries and tools commercialization. (TMK)

AL-TR-1992-0159

DIGITAL VIDEO INTERACTIVE (DVI) BASED AUTHORING TOOLS FOR UNIT LEVEL TRAINING

John P. Hyltin
Kenneth R. Janysek
Robert E. Hieronymus
Richard A. Palm
Ronald A. Martens

Betac Corporation
7323 Highway 90 West, Suite 510
San Antonio, TX 78227

HUMAN RESOURCES DIRECTORATE
TECHNICAL TRAINING RESEARCH DIVISION
7909 Lindbergh Drive
Brooks Air Force Base, TX 78235-5352

March 1993

Final Technical Report for Period 27 September 1991 - 27 May 1992

NOTICES

This technical report is published as received and has not been edited by the technical editing staff of the Armstrong Laboratory.

This research was conducted under the Small Business Innovation Research (SBIR) Program as a Phase II effort.

Publication of this report does not constitute approval or disapproval of the ideas or findings. It is published in the interest of STINFO exchange.

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed and is approved for publication.

TERRESA E. JACKSON
Contract Monitor

J. WESLEY REGIAN, Senior Scientist
Intelligent Training Branch

RODGER D. BALLENTINE, Colonel, USAF
Chief, Technical Training Research Division

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1993	Final - 27 September 1991 – 27 May 1992

4. TITLE AND SUBTITLE

Digital Video Interactive (DVI) Based Authoring Tools for Unit Level Training

5. FUNDING NUMBERS

C - F33615-90-C-0010
PE - 65502F
PR - 3005
TA - I2
WU - 01

6. AUTHOR(S)

John P. Hyltin
Richard A. Palm
Kenneth R. Janysek
Ronald A. Martens
Robert E. Hieronymus

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Betac Corporation
7323 Highway 90 West, Suite 510
San Antonio, TX 78227

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Armstrong Laboratory
Human Resources Directorate
Technical Training Research Division
7909 Lindbergh Drive
Brooks Air Force Base, TX 78235-5352

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

AL-TR-1992-0159

11. SUPPLEMENTARY NOTES

Armstrong Laboratory Technical Monitor: Terresa E. Jackson, (210) 536-2034.
This research was conducted under the Small Business Innovation Research (SBIR) Program as a Phase II effort.

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Distribution authorized to DOD components only; proprietary information; 1 May 1992. Other requests for this document shall be referred to AL/HRPP, 7909 Lindbergh Drive, Brooks Air Force Base, TX 78235-5352.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

DVI technology will have a major impact on the PC market as the speed of operation and costs are improved. Its greatest long-term asset is that it provides the ability for any user to program and develop applications in multimedia format. This will allow educators and instructional developers to create their own applications in such a way that, through sight and sound, multimedia information can be experienced on a PC-based system. This paper describes the DVI technology, the current authoring languages and tools and the reasons for developing new tools and applications.

14. SUBJECT TERMS

Digital video interactive
Intelligent tutoring systems
Multimedia
Object oriented authoring
SBIR program
Toolbook

15. NUMBER OF PAGES
36

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

PREFACE

This is the Technical Paper for the DVI Based Authoring Tools for Unit Level Training developed under a Phase II Small Business Innovation Research (SBIR) Effort. The purposes of this effort were to develop a object-oriented authoring toolkit for developing training programs that use the Digital Video Interactive (DVI) technology, and to develop a training system for the Air Force which demonstrates the capabilities of the authoring toolkit.

I. INTRODUCTION

Background

Digital Video Interactive™ (DVI) technology is a combination of hardware and software that enables a Intel 80386 or 80486 based personal computer (PC) or compatible to support multimedia. This technology is capable of seamless integration of digital video, audio, graphics, and text, and allows for real-time decompression of compressed digital imagery. It provides for the delivery of a video medium as well as other multisensory applications on desktop PC-based systems. DVI technology was developed by RCA/GE in the early 1980's. This technology was acquired by Intel Corporation in 1987, and is presently being fully developed by the DVI Group in Princeton, New Jersey. Betac Corporation, SWED, has participated in the development of DVI applications since February 1989.

DVI grew out of the desire to merge the controllability of the PC with the realism of video and audio in television. It was made possible by the development of a VLSI chip set capable of real-time decompression of compressed digital imagery. DVI consists of four unique elements:

- the custom integration VLSI chip set,
- a specification for a runtime software interface,
- audio and video data file formats, and
- digital data compression and decompression algorithms.

Using these four elements, DVI has the capability to compress and store imagery, audio, and textual information on either a hard disk or CD-ROM optical

disk. The data is arranged on the disk so that it can be individually and selectively used in applications programs.

The most exciting aspect of DVI is its inherent user interactivity. This characteristic makes it an ideal system for education and training applications. This interactivity applies to the courseware author, as well as the student, thus providing low cost, high fidelity training packages. Realistic simulations, animation, and special effects flow smoothly together with text and audio. This desktop PC-based system allows for multiple window displays; for example, presenting text in one window, a component closeup in a second window, and an instructor speaking in a third. The student's interaction with the system determines the speed and direction of the presentation. In short, the system exploits hypermedia technology to allow students to learn by using discover techniques under their own control. Where there is already a broad base of acceptance for computerized training, such as the Armed Forces, there will be very high interest in DVI Applications.

In the future, DVI technology will have a major impact on the PC market as the speed of operation and cost are improved. Its greatest long term asset is that it provides the ability for any user to program and develop applications in multi-media format. This will allow educators and instructional developers to create their own applications in such a way that, through sight and sound, multi-media information can be experienced on a PC-based system.

Organization

This paper describes the work performed as part of a Phase II Small Business Innovation Research (SBIR) project. The goal of the effort was to develop DVI based authoring tools for developing training applications, and to

demonstrate the use of the tools by developing a training application. Specifically, these tools are in the form of object oriented C++ programming language classes and executable programs.

The remainder of this report is organized as follows. Section II describes the DVI technology, Section III describes the authoring languages and tools currently available to develop DVI applications, and the reasons for developing additional tools. Section IV describes Betac's software development approach for developing the tools and applications using the tools. Section V describes the demonstration application, the Life Support Officer Training Course, and how it was developed. Section VI documents the lessons learned during this project for the benefit of future developers. The final section of the report presents some of Betac's future directions with the DVI technology.

II. DIGITAL VIDEO INTERACTIVE™ TECHNOLOGY

ActionMedia Hardware

Intel developed the ActionMedia 750 Delivery Boards as the second generation of hardware designed to deliver Digital Video Interactive™ Technology applications. The ActionMedia 750 Delivery Board enables playback of full-motion video, true color still images, and high quality audio in an all digital, interactive format. It is compatible with both IBM PC AT or Micro Channel Architecture personal computers.

ActionMedia Production Tools

ActionMedia Production Tools assist in editing and assembling audio and video data files in interactive applications. They provide capabilities to capture

and compress still images, audio, and full-motion video. They also allow editing and playback of audio and video sequences.

ActionMedia Software Libraries

Intel provides a language of C programming language callable software routines for developing DVI Technology applications. Many of these routines are primitive operations that produce dynamic image effects on bitmapped graphic images in VRAM like copying, blending, scaling, and color transformations. Other routines set and retrieve information about bitmaps and the state of the DVI system. An operating system called RTX is embedded in this library that allows some multitasking.

III. AUTHORING LANGUAGES AND TOOLS

Existing Systems

There are several existing DVI Technology high-level authoring tools currently available. CEIT Systems Inc.'s Authology: Multimedia™ and Network Technology's MEDIAscript™ are the most notable. Authology: Multimedia offers a text-based point and click methodology for developing DVI Technology applications. MEDIAscript provides a scripting language and front end programming environment. MEDIAscript offers more flexibility, but requires a scripting language compared to Authology: Multimedia's point and click interface.

Simulation Requirements in Authoring

The available authoring packages provide a subset of the full capabilities available to develop DVI Technology applications. Many of the features required

in simulation are difficult or impossible to implement using the available authoring systems.

Available DVI authoring tools did not allow us to develop training applications with enough dynamic features. They fell short in that they did not allow us to keep dynamic state information tracking a student's progress in the simulation, and those that made attempts at this, turned their tool into a programming language. They also are not able to give us all the visual effects needed to support the presentation of a realistic environment, and a complex user interface to that environment.

This is not intended to be an indictment of the individual authoring tools. It is an example of the reality that there is no way one authoring tool can provide a developer with every functional feature his imagination can dream up. The closer they get to this, the closer they are to traditional programming languages. What is required is a hybrid, point and click authoring environment, with a capability to incorporate and generate higher order language (HOL) source code.

IV. SOFTWARE DEVELOPMENT APPROACH

Previous Developments

Our previous efforts in beta testing the DVI Technology hardware, as well as Phase I efforts for the AFHRL and NASA, and in-house research and development work have produced a library of DVI authoring tools. These object-oriented programming tools have broad applications for functional users of DVI technology. The Tools developed previous to this effort include:

- Relative-Crop-Scale (RCS) is used to crop and scale an image relative to another. This is especially useful for resizing the imagery to simulate a panel of buttons,

indicators, and knobs. Since it is sometimes necessary for that imagery to be captured at different sizes, it will need to be cropped and scaled to fit in the overall control panel. This application allows the user to use the control panel background as a guide for sizing the images.

- CROP is used for many different image manipulation utilities, including changing brightness, contrast, cropping and scaling to fixed numerically entered sizes, etc. It is most useful for applications that have fixed window sizes, when the original imagery can come in any size.

- FINDHOT is useful for finding the hotpoints on an image.

- EMSMAP is used as part of the process for speeding up applications by putting compressed and uncompressed imagery in EMS. This utility allows the user to select what imagery, used in an application, will come from EMS.

- BLANK is used to turn EMS all one color. This, in conjunction with LOOK, is useful for determining what imagery is loaded into VRAM at a given time, how much VRAM was used in an application, and what it was used for. This information is invaluable in the testing and debugging processes.

- LOOK is used to examine VRAM, which allows the programmer to determine what imagery is loaded into VRAM at a given time, how much VRAM was used in an application, and what it was used for. This information is invaluable in the testing and debugging processes.

- EMSLOOK is used to examine EMS. See LOOK.

- TOTIF is used to convert a black and white TIFF file to a 16 bit DVI file. We used this to scan forms, pages from books, and news paper articles.

The Betac Object library included the following classes prior to beginning this effort:

- Cursor_Class

- Clipping_Class

- Controller_Class

- Ems_Mapping_Class
- Event_List_Class
- Hotpoint_Class
- Memory_Class
- Mouse_Class
- Rubber_Band_Box_Class
- Viewpoint_Class

These tools and objects are documented in the Appendix, DVI Based Authoring Tools Program Documentation.

Generic Tools and Objects

During the Life Support Equipment Investigations (LSEI) Course development, Betac programmers identified several tools required to support development of training packages. Betac developed the following tools during this effort:

- **KEYOUT** was developed to allow us to more quickly key out the background of an image. Using this program, we avoid converting the image format to that of a traditional artwork tool, which would introduce color artifacts. Working in the native DVI 9Y image format allows more control over the V and U plane values on the edges of a keyed out image, and prevents confusion of the background color with the colors in the image. With some practice, even the most complex images could have the background keyed out in under 5 minutes. (With more effort, involving the incorporation of fuzzy set logic, the process could be made almost completely automatic.)

- **REPCOLR** is used to change a single color in a 9-bit or 8 bit image to some other color. It is useful for changing the color or a keyed out image, to quickly make up for the short comings of VimCvt with regard to 8 bit images, and to remap images to a different CLUT.

- **Panel Matcher** is useful for matching up the edges of images to create a large visual panorama. The resulting image files can then be used in an application with the panel utilities. Using several images to make a panorama offers several advantages. It allows the application to load only the imagery needed at the time, which saves VRAM. It also allows the imagery to be loaded faster. Additionally, graphics operations are performed on a portion of the imagery at a time, instead of the entire panorama, which speeds up the operations.

- **EXCVT** is used to compress 8 and 9 bit imagery using a lossless compression algorithm. Lossy compression and decompression of 8 bit imagery and 9Y imagery causes some intolerable color artifacts to show up.

- **CBN8-9** is used to put an 8 bit overlay on a 9 bit background for 9Y image files. We found that converting an image to 9 bit when it has elements of photographic imagery and graphics overlays causes the overlaid graphics to "bleed", and this utility is part of the process to correct that problem.

- **9YTO16** is used to convert a 9Y image file to an I16 image file, while preserving the 8 bit graphic overlay parts. VimCvt does not preserve the 8 bit graphic overlay portions of the image.

- **EDITCLUT** edits the colors in a Color Look-Up Table (CLUT). This is useful for correcting some of the difficulties VimCvt has with 8 bit images, and for creating a CLUT that will be used in a displayed image that combines 2 8 bit images with different CLUT tables. It can also be used to support CLUT configuration management processes for developing an application.

- **FONTLOOK** is useful for viewing and picking different text fonts for an application.

- **PANDVR** is a tool used for testing out matched panels. After matchup up edges of images using MATCHDVR, you can use PANDVR to see how they will look in your application. This program prompts you for a data file with records that identify the image filename, dimensions of the images, and the image format. Use the left mouse button to pan through the imagery, moving the mouse cursor to the left side of the screen to pan left, or move the cursor to the right side to pan right. You may also pick the size and

location of the window to use as a panning window, specifying the dimensions numerically, or using a stretch box.

- **TRANSTST** is a tool used for testing out different transition effects. There are several different effects to chose from on the menu. The Mask_Trans effect uses an 8 bit mask to produce the effect. The user can create his own mask or use some default masks provided (mask1-mask6, wave1-wave2). The mask defines the effect for this routine and must be 8 bit. The color ranges present in the mask control the effect. For instance to get a left to right (vice versa) wipe a mask with increasing colors could be created. Each column or set of side-by-side columns in the mask has a specific color in the color range specified.

- **ISETPIX** is a very small art package for 9Y images. It has the advantage of working in this native DVI format. You may draw 1 pixel at a time by clicking the left mouse button, and an "undo" function is available with the right mouse button, which retracts the most recent pixel setting. A menu of choices allows you to draw in all of the bitmap planes, to draw in any one plane (Y, V, or U), to zoom in on the image at the cursor while you draw, to change the draw color, or to save the image.

- **EX9SHOW** is the equivalent of the Intel ActionMedia Production Tool VSHOW, except that this displays images generated by the EX9CVT tool, which compresses 8 and 9 bit images with the lossless compression algorithm.

- **READCOLR** is used to interactively poll the colors in a bitmap. The use must first supply the name of an image file to examine, and it will be displayed on the screen. The user may then use the mouse to move the cursor to any point on the screen, and click on any point, The color in the center of the cursor crosshair will be displayed on the screen.

The following classes were added to the Betac Object library during this effort:

- Animation_Class
- Doubly_Linked_List_Class
- Float_Vector_Class

- Generic_List_Class
- Generic_Stack_Class
- Motion_Class
- System_Class

These tools and objects are documented in the Appendix, DVI Based Authoring Tools Program Documentation. To fully demonstrate the use of these tools, we intend to develop, and include in the final paper, a cross-reference matrix to demonstrate the use of these tools and objects within the LSEI Course.

Application Specific Tools and Objects

A special set of application specific classes were also added to the Betac Object library during this effort. These classes are specific to the needs of the LSEI Course and may be difficult to understand without completely understanding the LSEI application. These objects are:

- The **Action_List_Class** is a lower level class which maintains a list of actions that the presentation will display to the user when appropriate. Each window has an action list associated with it. The branching windows may have numerous action lists per window. This class is responsible for freeing the memory associated with each action when the time is appropriate.
- The **Expert_System_Class** is part of the Intelligent Tutoring System. It implements the expert system used to determine the correctness of the trainee's actions.
- The **Feedback_Class** is part of the Intelligent Tutoring System. It identifies scripts used for feedback to the trainee and tracks when the user has seen that feedback.
- The **Feedback_Groups_Class** is part of the Intelligent Tutoring System. When the trainee makes a mistake in the simulation, the Tutorial Strategy module selects from a group of feedbacks identified for that particular error. This

class is used to represent the association of the feedbacks and the error identifier.

- The **Navigation_Class** is responsible for keeping track of the current position of the user in the presentation program. It also keeps track of all the window titles in the presentation. This class also allows a user to jump or navigate forward and backwards through the program.

- The **Notebook_Class** tracks the entries the student has made into the notebook, and supports the trainees capability to review the notes.

- The **Popup_Window_Class** controls the user interface portion of the branching windows in the presentation. It is responsible for drawing and refreshing user interface control options in the window as well as returning handles to options chosen by the user. It keeps track of the sizes of each of the interface window buttons.

- The **Question_Answer_Class** is used to display textual multiple choice questions to the user, and allow him to choose answers. This object can be adapted to other applications with some modification to the source code. While some effort was expended to make this object more generic, it was necessary to make specific allowances for the LSEI application. In this application, the questions are displayed on a static background consistent with the rest of the application. Since the questions and answers would not all fit in that background, the questions are scrolled. Script identifier numbers are associated with the question and each answer, so that explanations and teaching points could be made with the questions. Finally, each answer is correlated to an area of expertise involved in the LSEI simulation, so each trainee's answer is used to assess the trainee's expertise in that area.

- The **Scenario_Class** controls the simulation scenarios presented to the trainee. Scripts are written to identify key elements of each scenario in the simulation. This class interprets those scripts and performs operations necessary to set up each scenario. Among the activities for setting up a scenario are defining hotpoints, specifying imagery, and telling the expert system about the scenario.

- The **Simulation_Script_Class** implements scripts for use in the LSEI simulation to present information in a multimedia

format. The scripts are used to inform the trainee on how to use the system, give him information about the current scenario, and to provide feedback on his actions.

- The **Student_Class** tracks personnel information on the trainee, including name, rank, etc. It also tracks information about time spent with the LSEI course and the current position of the student in the presentation program.

- The **Student_Profile_Class** is part of the Intelligent Tutoring System. It is used to repeatedly assess and reassess the student's skill level in different areas. This assessment information is used by the Tutorial Strategy logic and is intended only for the purpose of determining what feedback should be used.

- The **User_State_Class** is part of the LSEI simulation software. This class represents how far the trainee has progressed in the simulation, and the actions he has taken that effect the scenario, including the pictures taken, the evidence collected, and also tracks where flags have been placed.

- The **Video_Windows_Class** controls the user interface portion of the presentation. It is responsible for opening, closing and resizing the branching windows as well as returning handles to options chosen by the user. It keeps track of the sizes of each of the interface windows and buttons. This class makes call throughs to operations on the Popup_Window_Class objects because the presentation program does not have direct access to those objects.

- The **Window_Actions_Class** encapsulates the Action_List_Class objects. This class forms the next layer up from the Action_List_Class. The information presented in the branching windows uses this class to store the actions. The branching windows need this class because they may have multiple presentations to show to the user.

Development Methodology

Betac has developed a three phase approach for developing DVI applications. The methodology consists of an Analysis Phase, Design Phase, and Implementation Phase.

In the Analysis Phase, Analysis Paragraphs are written, a requirements list is prepared, and a function description is developed. The following paragraphs further define the steps within the Analysis Phase:

- *Analysis Paragraphs:* Include paragraphs to define the problem at a high level. It should be brief and to the point.
- *Requirements List:* This is a list of what the user will be able to do with the solution system. The requirements should be checked to ensure the problem defined during analysis will be solved.
- *Functional Description:* This is a description of the user's view of the system. It describes a method of interaction between the user and the system in solving some problem. It will be an important point of reference during the design phase. The designer must understand and meet the requirements expressed in the list of requirements. He should also keep in mind when writing the Functional Description (FD) if the proposed method is reasonable.

In the Design Phase, problem and solution spaced objects are described, interactive control structures and procedures are described, structure charts and object diagrams are developed. The design is validated against the functional description developed in the Analysis Phase. The following paragraphs further define the steps within the Design Phase:

- *Define Problem Space Objects:* These are objects which characterize real world entities such as input devices. These will most likely be identified by reading the FD. Some problem space objects may be implemented directly as a software entity or may be implemented using a set of solution space objects. Use this step to define what the operations are and what information is represented by it.

Note: The next 4 steps are done in parallel. The results of each activity can feed back into the definition of problem space objects, i.e., they imply information about their existence and what they represent.

- *Define Solution Space Objects:* These are objects which do the work of the problem space objects. These may include objects used to communicate between other objects or procedures, or objects used to keep track of local implementation data. These objects will be defined by analyzing the relationships between the problem space objects and the functionality described in the FD.

- *Define Interactive Control Structures/Procedures:* These are the procedures which control actions resulting from user input. The ICS's have the general form of input functionality followed by a loop. The loop contains a case structure to decode user input. Procedures are called to implement each selections.

- *Define Process Control Modules (PCM):* These are higher level procedures that control the flow of execution. They are similar to ICS's, except a control variable determines which operation is executed next. PCM's have the general form of a loop containing a case structure to decode the control variable into an executed procedure. Each executed procedure returns a new value of the control variable, telling the PCM which module is executed next.

- *Analyze Object Relationships:* This analysis step determines if other objects should be defined to simplify the interfaces between already defined objects. There is no real activity that defines this step. It is more of a note to think about how objects are related. The Interactive Control Structures are an important part of this analysis

- *Structure charts and object diagrams:* Charts and diagrams are drawn based on Interactive Control Structures.

- *Validation Step:* Check the functional description and requirements list to make sure all functionality will be implemented with this design. An optional fallout document from this step is a cross reference chart diagramming which objects and control structures are used to address each requirement in the requirements list.

In the Implementation Phase, a schedule is developed for the order of module development, modules are coded and tested, and the software modules

are documented. The following paragraphs define the steps within the Implementation Phase:

- *Implementation Planning Step:* Plan the implementation, by determining which modules should be developed first. The plan should enable the programmer to test many modules in a manner similar to unit testing, which is usually the most thorough. This can be done mentally by looking at the structure chart, or by using a copy of the structure chart to sequentially identify the order of implementation.

- *Implementation Step*: Code and test the application.

- *Documentation Step:* Document software modules that may be reused, and incorporate a programmer's guide to using each into a software library reference document. Further documentation should tell how each module was used in the application. Also document the use of the application and incorporate that documentation into the tool reference document.

Intelligent Tutoring Strategy Approach

Our guidelines for the ITS were based on the traditional model, developing Simulation Interface, Feedback, Student Profile, Tutorial Strategy, and Expert System modules. The simulation provided a series of scenarios with definite responsibilities for the trainee to satisfy. The Student Profile kept a history of user actions and errors, categorizing them into activities of Evidence Recognition, Evidence Handling, Documentation, and Safety. The trainee's actions were assessed against the Expert Knowledge to determine correctness. A Tutor Strategy module used the history and accuracy assessment information, and selected the most appropriate feedback available for that situation. Each feedback was assigned one of 4 characteristic values. A feedback could be a hint at what to do, it could tell the user what to do, it could show the student what

to do, or it could yell at him or make a sarcastic joke to indicate the user should know what to do. All parts of the ITS were developed in C++.

V. LIFE SUPPORT OFFICER TRAINING COURSE USING DVI

Training Needs Assessment

"The Air Force conducts safety investigations to find the causes of mishaps to prevent recurrence." (AFR 127-4, paragraph 3-1)

This statement sets both the requirement and the tone for all USAF Safety Investigation Board (SIB) activities. Life Support Officers will be called to participate in SIBs related to mishaps involving people in aircraft because the Air Force sets a high priority on human life and is sincerely concerned about the well being of its people.

USAF Life Support Officers (LSO) are currently trained for their duties through ATC Course S-V8G-A, a two-week course taught at Randolph AFB, and through OJT. Included within the formal course is a two-day section related to mishap investigation, taught at Kelly AFB.

Mishap investigations are highly complex, team efforts, conducted under the direction of an Air Force Safety Investigation Board, as directed by AFR 127-4. The team can draw on the guidance in AFM 127-1, Volumes 1, 2, and 3, in order to insure that all necessary topics are addressed. However, principles and methodologies of investigations for Life Support Officers are taught only in this two-day mini-course at Kelly AFB. Formal ATC-level documentation for this subset is minimal. Graduate level courses in aircraft mishap investigations are taught at the University of California in Los Angeles, but the average LSO has no access to them.

Life Support Equipment Investigations Course

The Contracting Office Technical Representative, as well as the principal people involved with LSO training, agree that the Air Force needs to be looking for ways to make this training readily available in the field. The two-day mishap investigation training subset was selected as the subject for demonstrating the capabilities of the DVI authoring tools being developed under the SBIR. The DVI course will be identified in the following discussion as the Life Support Equipment Investigations (LSEI) Course. Ease of updating the material is a concern since even the science of mishap investigations is subject to refinement. The LSEI subject matter was chosen for this DVI prototype also because it lends itself to the highly visual nature of evidence collection and evaluation, and because the interactive nature of DVI training is well suited to the busy, demanding environment of the squadron/wing LSO.

The LSEI Course presumes that the LSO using the course and necessary computer equipment will be located in the squadron/wing environment. He can participate in training on an irregular basis because of the press of other military duties. Experience in the field has been that the LSO seldom is called on to apply his mishap investigations skills for an extended period of time, sometimes measured in years, after completing Course S-V8G-A. But when that time comes he must be able to conduct the investigation thoroughly and accurately. Experience has also shown that since no two mishaps are the same there is no way a single set of rules can apply in all cases. The LSO needs mature judgment and a keen mind, backed up by a good understanding of the principles and methodologies that have proven effective in analyzing evidence. A requirement therefore exists for allowing the LSO to maintain skills in the

principles and methods of investigations, to easily receive updated information, and to have a tool for quick reference.

The LSEI Course is designed to permit the LSO to progress through the material at times of his choosing and at his own speed, to review any or all of the previously presented material at will, or to proceed to portions of the material that he finds important at the moment. The computer system is also designed to allow a training supervisor to monitor the amount of time the LSO spends in training.

Resources for Course Development

The system developers were provided access to the following materials in preparation of the course.

- Mr. Michael Grost, USAF subject matter expert and manager of the Kelly AFB mishap investigations laboratory. Mr Grost provided advice and located materials required for the LSEI Course. These materials included 35mm slides of actual mishap investigations, 16mm film and VHS tape clips related to investigations, and selected SIB report extracts. The development team also had ready access to the investigations laboratory at Kelly AFB to study Life Support equipment.
- AFR 127-4 and the AFM 127-1 series.
- Life Sciences Equipment Investigation manual (Draft, undated), authored by Mr. Grost.
- Interview with Capt. Nick Radovich, an active wing LSO, following his work on an SIB, 9 October 1991.
- Access to observe actual SIB in progress, England AFB, 11 October 1991.
- Attendance (audit) at the two-day Life Support Officer Investigations Course at Kelly AFB, 22-23 May 1991.

Target Student Population

The expected student audience for the LSEI course is composed almost exclusively of rated officers who have been designated as Life Support Officers at the squadron and wing levels. A select group of senior NCOs may also be so designated. All the students can be expected to be assigned as Life Support Systems investigators on formal Safety Investigations Boards subsequent to completing their training. This target population can be considered to be highly motivated, highly skilled in their own technical areas, and well able to master any mechanical aspects of the DVI computer system itself. Because most of them are volunteers for this duty, they can be expected to have a keenly developed sense of mission and appreciation for the importance of the SIB process. Furthermore, they bring with them a broad background in flight operations and flight safety.

Teaching Philosophy

The material is taught in two primary modules, first a Presentation module in which the student is taught the principles and methodologies of mishap investigations, and secondly, an Application module in which the student makes decisions to implement the strategies taught in the first module. Throughout the course, the imagery used is taken from military archives of real aircraft mishaps and actual scientific tests related to life support systems. One criteria for selection of the imagery, however, was that no identifiable human remains be visible at any time.

The visual presentations in the LSEI course are in full color and accompanied by high fidelity audio. Since the source imagery is from military archives and not originally shot under controlled conditions, it includes a variety

of qualities and formats, including also black and white photos and newspaper pages. Even though the target audience is considered to be highly self-motivated, every effort was made to keep the presentations fluid and interesting. DVI system capabilities were used to provided a mix of motion video clips and still images, graphic overlays, computer generated animations, and text. In addition, many images are presented in windows that can be sized up at the student's command for closer examination. Transitions from one set of images to another are done by randomly selected wipe sequences to ensure visual variety.

Actual Air Force mishaps are used as the backbone of both the training modules, thus providing realistic story lines around which to build the teaching points. Based on these incidents, the story is told as though by an investigator who finds opportunities to discuss related elements of investigations principles and methods as the backbone mishap investigation moves along.

In the first module, the student has numerous opportunities to temporarily leave the story to explore supporting information related to the phase of the investigation at hand. He also has the option of following the main story to its conclusion without taking any of the side roads and later expand the scope of his exploration. A method is provided for the student to go back or forward to any part of the story or the support material at any time, thus satisfying the requirements for learning, review, and reference.

In the second module the student is presented with the requirement to apply the principles learned in the first module, to carry out specific actions, make entries in his (computer) notebook, and answer questions. Feedback is provided through both audio and visual cues under the direction of an intelligent

tutoring system (ITS). Since each event in the story is part of an actual field sequence, the student is not allowed to go forward in the story line until he has completed all the work at each step.

In both modules, the student is permitted to exit the system at any time. A bookmark is automatically set so that, when he signs on the next time, he goes immediately to the place where he left off.

A unit training supervisor will establish the requirement for a student to train on the system. This supervisor will have access to a special file in which he will enter the student identification data, authorize the system to accept the student upon sign-in, and later monitor the fact that the student did accomplish the training. Only persons designated by the training supervisor will be authorized access to the course material. This was considered necessary because the Air Force does not permit the general public to have access to mishap files for obvious legal and humanitarian reasons. No classified material is included in any manner at any time.

Because this LSEI Course is part of a larger project to demonstrate certain computer technologies, this phase of the project is not designed as a complete training course. The story line of the tutorial module mishap, together with the supporting teaching material, is taken only up to the point in the investigation where the evidence has been transported to the field laboratory. The laboratory phase of the mishap investigation would be another major part of the story to be developed during subsequent work. It was necessary to set bounds on the time and resource expenditures available for this project so this is where, with the direction of the contracting office, the line was drawn. Findings related to the mishap in the Presentation module are therefore not presented. In

the Application module, the final conclusions are presented based on the actual SIB findings because they involved laboratory work beyond that normally required of the LSO in the field and could be presented quickly so as to draw the story to a conclusion.

Course Outline

Following are the main sections of the LSEI course.

- *Student Data.* The training supervisor enters the student data into the access authorization file, after which the student can sign on to the system at any time. Data is automatically entered into this administrative file identifying when and for how long the student was on the system.

- *System Introduction.* In this short section, the student can inquire about any specific elements of the training system to learn about its functions, or the computer will run through the entire introduction script explaining what each part of the system does.

- *Presentation.* In this module the student is presented with the backbone mishap through which the principles and methodologies for investigation are presented. Through highly interactive choices, he may explore supporting data related to the phase of the investigation before him, or depart completely from the story line to review previousiy presented material. He can also use this module as a reference tool through use of the NAVIGATE command button and its subsidiary data outline displays.

- *Support Data.* This is the repository for all the audio and visual data that supports the investigation principles and methods presented in the backbone story. This data is presented as branching windows as the story line develops. To satisfy requirements for review and reference, it is also accessible through the NAVIGATE command button on the main screen.

- *Application.* This is the second main module in which the student is tasked to apply the principles and methods taught in the first module. Ten icons provide the tools which the student can use to implement his decisions, and feedback is

given based on his choices. The intelligent tutoring system follows his progress, adjusting the positive or negative feedback responses to his demonstrated skill levels. Four levels of response are included; they are HINT, TELL, FORCE, and DO. The terms generally describe the increasing levels of direction imposed by the system as errors are recognized and corrected by the student or are repeated.

VI. Lessons Learned

During this effort, Microsoft delivered a new version of their personal computer operating system, MS-DOS 5.0. This impacted development in several ways; for example, it provided additional host system memory for our applications, but did not allow us to break the 640 kilobyte DOS "barrier".

Several of the tools we developed are individual artwork tools. We developed these tools since their wass no commercial package which allows the editing of DVI images in their native format. This is still true, as there is no other artwork tool available that works with all native DVI formats.

C++ preprocessors also provided some challenges in this effort. They didn't support all the features of the ANSI 2.1 standard, they had some bugs, and the dependence on Microsoft C made life interesting. We look forward to these difficulties being corrected with the development of AVK which will allow the use of actual compilers, rather than preprocessors, and will run in windows.

Many programmers use some type of Integrated Development Environment (IDE). These environements allow the programmer to compile, link and run your application, all without leaving the text editor. Most also have a debugging capability. This represents a savings because the text editor can take

several seconds to load, and the programmer doesn't have to do anything to load it. In Microsoft C, the IDE is called the Personal Work Bench (PWB).

For all the advantages of using an IDE, we were unable to make them work for our case. DVI applications are sensitive to the restrictions placed on memory, and also to Terminate-and Stay-Resident (TSR) driver programs. This can be worked out for some DVI applications, but not all. We spent time chasing bugs that were caused by the presence of the development environment TSR's, more time than could be saved by them. This section will describe the alternative tools we used in our development environment.

Since our source code editor did not have to include a vendor-specific compile capability, we could choose from the best editors on the market. We chose Brief, from Solution Systems. This editor will allow you to compile, although that does not represent a significant savings. It does many features common to the best text editors available, including windowing, file swapping, programmable macros, and search and replace. Many other source code editors will do the job for DVI applications, but we felt it was worth mentioning since IDE's have this feature.

We used the C++ preprocessor from Intek, or Integrated Technologies. This is important, because this will affect your capability to develop applications using our software libraries. You will have to compile your source code with this preprocessor if you want to link with our library. We chose the Intek preprocessor because it had a few tools with it, such as a tool called HDRPORT that automatically makes C header files compatible with C++ in seconds. Experience tells us that manually porting the DVI header files takes about 4 hours.

We did not find a source level debugger that worked well with preprocessed C++ code. Intek says they are working on it.

We used Microsoft C (MSC) to compile the preprocessed C++ code. We had to use this because the DVI libraries are compiled with this. We used version 6.00 of MSC.

Microsoft C 7.0 has just become available, and it incorporates C++. We have experimented with it some and found it works on our libraries and tools, but not for large applications. Time did not allow us to trace the exact reasons for this, but it may have something to do with the collision of names of global variables between the DVI libraries, and the MSC 7.0 run time libraries. Intel does not intend to do anything about this in version 2.13 of the DVI Libraries, but will investigate it in version 2.20 for the Action Media II boards. Overall though, the results were very promising. The compile time is cut in half or less. Some applications were significantly smaller, but some were larger. There is also a library of source code objects for processing lists and mathematics included with the product. We intend to use this for all future development.

There is one other environment worth mentioning. Digital Video Arts has a library equivalent to the Intel DVI library that is compatible with Borland C/C++. The main difference is that the RTX operating system has been excised from the system. This should allow some access to the Integrated Development Environment, but not total compatibility.

Many DVI applications require numerous images to load into VRAM all at one time. This can cause long delays at some points in the application. Unfortunately, there is no direct method for putting compressed image data into EMS. We designed and developed a program, called EMSMAP, which builds an

image of the data that will go into EMS and saves it as all one image. The resulting map file is loaded by the application with a call to the DVI library routine GrImLoad, along with a file that provides lookup information, equating each image to an integer tag. When the application needs an image from the map file, the application should call the Load_Image operation from the Ems_Mapping_Class, using the integer handle corresponding to that image.

VII. Future Directions

Object Libraries and Tools Commercialization

Betac intends to commercially market the object libraries and tools developed before and during this effort. In April, Intel announced Betac as one of eight Technical Solutions Specialists (TSS). The TSS program provides technical support for DVI developers. This program will provide a channel for Betac to market the products developed under this effort. Betac will also work with DVI hardware distributors and share in sales of DVI products. Betac will also market AVSS version 2.20 software library. Betac iwill also provide consulting services under the TSS program. Betac is currently working with both Intel and IBM on marketing agreements and strategies. Betac is also working with several DVI hardware distributors to develop margin sharing agreements for DVI hardware.

In October 1991, Intel and IBM announced the ActionMedia II™ boards, the successor to the ActionMedia boards. In January 1992, Betac became a beta test site for the ActionMedia II boards as well as the AVSS 2.20 and DVI Multimedia SDK for Windows™ software development environments. Betac will convert, package, and market the object library and tools developed under the

SBIR effort to support the new board and AVSS 2.20. Betac will also convert selected portions of the object libraries and tools to the Windows environment.

EDITING SERVICES
ARMSTRONG LABORATORY/DOKE
2511 KENNEDY CIRCLE
BROOKS AFB TX 78235-5122

OFFICIAL BUSINESS