DOCUMENT NO. CRS004

Guidelines for CBT Courseware Interchange

ORIGINAL RELEASE DATE 31-Oct-95

THIS DOCUMENT IS CONTROLLED BY:

AICC Courseware Technology Subcommittee

ALL REVISIONS TO THE DOCUMENT SHALL BE APPROVED
BY THE ABOVE ORGANIZATION PRIOR TO RELEASE.

POINT OF CONTACT:

Scott Bergstrom
AICC Administrator
University of North Dakota
Box 8216, University Station
Grand Forks, ND 58202-8216

Telephone: (701) 777-4380

PREPARED ON PC FILED UNDER CBT-CW.DOC

Prepared by:

William A. McDonald
CBT Systems Analyst
Boeing Commercial Airplane Group
(206) 662-8485

Date

Approved by:

Mike Medley
Chairman AICC
Douglas Aircraft Company
(xxx) xxx-xxxxx

Date

ABSTRACT

This document describes recommended data import/export features for authoring systems. These recommendations are designed to facilitate the interchange and reuse of CBT courseware. This document contains amplifications and examples to support AGR 007 COURSEWARE INTERCHANGE.

KEY WORDS

Data import & export
Courseware interchange
Export data
Interchange
Platform migration
Re-purpose
Re-use
Standard data formats

REVISION HISTORY

NEW	**31 Oct 1995**
15-Feb-95	The only difference from the final draft (22 Aug 95) is the addition of a description of the relationship of this document to AGR 004 and AGR 008.

Table of Contents

1.0 INTRODUCTION

Purpose

Currently much of the CBT courseware produced in the aviation industry is not easily portable from one authoring system to another. The primary reason for this is that the data (Audio, Graphics, Video, and Logic) are in a format proprietary to the authoring system.

While there are software industry efforts under way to create an interchange format for interactive multimedia (MHEG, HYTIME, ScriptX, etc.), it will probably be years before there is a generally accepted standard.

The purpose of this document is to offer some means of CBT courseware interchange in the interim.

The need for guidelines

There are several reasons for moving courseware content from one system to another:

- Re-purpose: For instance move some training course content to a performance support system.
- Re-use: Use elements of an existing program that were developed in authoring system A to a new program being developed in authoring system B.
- Simplify maintenance: Move courseware into one or two authoring systems so that a staff with expertise in 5 or 6 authoring systems is not required to maintain a company's courseware. This is a typical problem in the airline industry where an airline may get courseware from several different aircraft manufacturers, all developed in different authoring systems.
- Migrate to new platforms: Move coursware from older hardware to new. For example from an 80286 DOS machine to a new DEC Alpha RISK computer running Windows NT, or from an Intel machine running Windows to a RISC machine running Unix.

Scope

This document describes:

- The major data components of CBT courseware
- Standard data formats for these components
- How CBT authoring systems can import and export these components in an automated (or semi-automated) manner

2.0 OVERVIEW

Major Elements

Most CBT courseware can be broken down into the following elements:

Basic	
Text	(ASCII Text)
Audio	(Digital or Analog)
Graphic	(Bitmapped or Vector)
Frame-Based motion	(Analog Video, Digital Video, Cell-based animations)
Complex	
Logic	(Programming)

Basic elements

Basic elements are those portions of a lesson's content that can be stored in a well defined, widely accepted format -- a current defacto standard. Each of these elements is defined in the next section.

Complex elements

Complex elements are those portions of a lesson's content that cannot be defined in an existing industry standard format. These are parts of a lesson that require special (and currently unique) descriptions.

Current status

Both basic and complex elements frequently exist within today's CBT lessons in various combinations "internalized" in file formats proprietary to the authoring system in which they were created. Well-accepted standard formats for interchanging all of the basic data items (Text, Audio, Graphic, and Frame-Based Motion) exist in the software industry today.

Achieving interchange

Given these standard formats, CBT courseware interchange can be achieved (at some level) by interchanging all the basic elements individually (in files) and the complex element descriptions in ASCII text.

Since there are logic (and composite[1]) data standards being developed but there are none currently widely accepted, the ASCII text description of the logic (or composite) could be specific to the authoring system.

[1] Composite: A special logical description that uses text to create a combination of basic elements. For instance, a composite file could describe how to move a bitmap across the screen -- that is, create a path animation.

In order to provide the this kind of interchange, a CBT Authoring system must be able (at some level) to do the following:

1. Export all basic elements to individual files in standard industry formats

2. Import all basic elements from individual files in standard industry formats

3. Export all composites & logic to an ASCII text representation

4. Import all composites & logic from an ASCII text representation

3.0 DEFINITION OF INTERCHANGE ELEMENTS

The following are definitions of the elements of CBT courseware interchange.

Logic

All Logic files are lesson behavior descriptions written in a plain ASCII text. These descriptions could be in the form of a programming language, a scripting language, or even an SGML type of file. Types of lesson behavior described in the logic file include:

- Student navigation
- Answer judging
- Conditional branching
- Combining basic elements
- Sequencing basic elements
- Writing CMI data to a file

Logic files combine lesson behavior descriptions with references to other the other interchange elements (Audio, Graphics, Text, and Frame-Based) to define items such as interactive (or dynamic) graphic displays and path animations. Multiple logic files may be required for a single lesson -- with one logic file referring to another.

The logical "language" used can be in any format as long as it is ASCII text and that it meets the following condition:

> *With the logic file(s) and the other interchange element files, an individual with a good understanding of the (Source) authoring system would be able to reproduce (without any other information) the exported CBT completely.*

Logic file format

Logic elements can be stored in one or more ASCII text files per lesson.

Text

This is text associated with a lesson that is not embedded in a graphic, logic file, or frame-based motion sequence. Typically it includes the audio script and explanitory text that appears on screen in tutorials (but not call-outs and titles that appear in a graphic.)

Text file format

Plain Text (non-logic) data resides in ASCII text files.

Graphic

A graphic is that part of a still image that the student can see on screen but which is not contained in a separate logic or text file.

Each individual graphic referenced in a Logic File must reside in separate graphic files. Separate graphic files may include more than one graphic in each file. The graphic may be either vector or raster and must be in one of the following industry graphic file formats :

Graphic file formats

CGM	Computer Graphics Metafile[2]
TIFF	Tagged Interchange File Format[3]
PICT	Macintosh Picture Format
DIB (or BMP)	Windows Device Independent Bitmap
WMF	Windows Metafile

Frame-based

Frame-Based motion elements are sequences in the lesson in which the student percieves motion that has been produced by a rapidly presented series of pictures. Frame-based motion may be produced from digital video, cel animations, and attached video discs. Some frame-based encoding mechanisms include embedded audio.

Frame-based file format

Frame-Based video and cel animations can be stored in the following ways for import/export:

1. In one of the standard digital video or animation file formats:

---

[2] CGM: The ISO CGM standard is a large and complex one. The preferred subset of this standard is defined in *ATA Specificaiton 2100, Graphics Exchange.*

[3] TIFF files can hold graphics that have been encoded in several different ways. The preferred TIFF encoding methods are defined in "AGR 004, Courseware Delivery Stations: Software."

AVI	Audio Video Interleaved.[4]
MPEG	Motion Picture Experts Group
QuickTime	Apple format
FLIC/FLI	AutoDesk Animator format
PICS	Apple format

Each sequence must reside in an individual file. Depending on the format, individual frames may be accessed

2. In logical textual descriptions in logic/composites file(s) that describe a location on external media (such as video disc.)

Frame based encoding format

The digital motion encoding mechanism may be in any AICC approved format.

Audio

Audio refers to sound not embedded in a frame-based motion sequence. Audio may include verbal information, music, and sound effects. Audio may be produced from digital files, or attached audio device like a CD player.

Audio file format

Audio can be stored in the following ways:

1. In individual files in one of the standard digital Audio File formats :

WAVE	MS Windows audio
AIFF	Audio Interchange File Format (Mac)
AICC	Aviation Industry CBT Committee standard formats (DOS)

2. In logical textual descriptions in logic/composites file(s) that describe a location on analog audio on external media (such as Audio CD or video disk.)

Audio encoding format

Digital audio should be encoded in

- Mu-law PCM -- 8Khz sample rate and 8 bit sample size. (ISO G711)
- ADPCM -- 8Khz sample rate and 4 bit sample size. (ISO G721)
- Linear PCM -- 11 Khz sample rate and 8 bit sample size.

---

[4] The AVI and QuickTime formats support a variety of digital video compression mechanisms. See "AGR 008, Digital Video" for general compression recommendations.

4.0 RELATION TO OTHER GUIDELINES

ATA and AICC

There are a number of guidelines in both the AICC and ATA[5] that are related to multimedia production. This chapter discusses their relationship to this document.

4.1 Relation to ATA Spec 104

Contents

ATA Specification 104, Guidelines for Aircraft Maintenance Training addresses

- Levels of training,
- Course objectives,
- Development process
- Maintenance Manual content and format and style
- Function and type of learning materials
- Delivery platform for CBT

This document addresses the data elements that make up a CBT lesson. It focuses on the export and import of these elements in industry standard formats. The content and presentation formats described in Spec 104 are independent of the data formats defined in this document.

This document does not conflict with any of the guidelines addressed in Spec 104.

[5] ATA: Air Transport Association

4.2 Relation to ATA Spec 2100

Contents

ATA Specification 2100, Digital Data Standards for Aircraft Support is a comprehensive guideline designed to address all elements of an aircraft maintenance system -- including documentation, performance support, and training.

Spec 2100 includes a comprehensive plan or approach to this system. The document includes detailed guidelines for delivery of electronic documentation as part of an interim solution. It also points to a longer term solution.

SGML

The interim solution describes the delivery of documents required to support airplane maintenance. SGML (Standard Generalized Mark-up Language) is the adopted for this approach. There are detailed descriptions of tags (DTDs or Document Type Definitions) for each type of manual.

Because SGML files are simple text files, non-text elements of publications (namely pictures) are referenced in the SGML files. Spec 2100 also defines data formats for these elements.

Training

Training data guidelines are not currently addressed by Spec 2100. However, it is assumed where data elements are similar (namely pictures), the guidelines will include the same standards. Where CBT data elements are different, obviously new or extended standards will need to be adopted.

This document

This document adopts the Spec 2100 guidelines for all elements that are the same or similar to elements in CBT. The approach to data interchange described here however, is much closer to the long range goals of Spec 2100 than the interim solution currently described for document (not CBT) delivery.

When a Spec 2100 team is formed to address CBT data, it is possible that they will use this document as a baseline for their CBT data delivery guidelines. When Spec 2100 CBT guidelines are completed, and if those guidelines address CBT data element formats, they will supersede this document.

4.3 Relation to AGR 004

Contents of AGR 004

AGR 004 is entitled ”Courseware Delivery Stations: Software." In addition to networking and operating system recommendations, it includes a recommendation for "the exchange of vectorial graphic information" and "raster graphic information."

AGR 004 recommendations are:

Vector graphics	CGM -- ATA defined subset
Raster graphics	TIFF -- 4 specific encoding methods: None - no compression CCITT Grp 4 PAKBITS (RLE) - JPEG

AGR 007 adds formats

AGR 007 includes the formats recommended in AGR 004, but in addition, adds several others. The additional formats are all widely used industry standards. Their addition enables a wider selection of media development tools and authoring systems.

Formats for internal use

Inside a company, these additional formats ease the process of moving media elements from one project to another, or from media development tool to authoring system. The more restrictive recommendations of AGR 004 are more appropriate for moving media elements between companies.

There are a number of off-the-shelf conversion tools that enable changing graphics from one format to another. Using conversions utilities it is possible to maintain AGR 004 compliant formats within a company. The result of such adherence to AGR 004 would be extra steps in some development processes and in some cases, degradation of the images converted.

Inter-company vs. intra-company

For internal development, it is simpler to purchase tools that support the same standard formats; then move elements between development projects using the formats directly supported by these tools. The graphic elements can then be converted to the limited subset recommended by AGR 004 when sending media elements to another company.

Having fewer recommended formats for moving media between companies also has an advantage. A conversion tool can be acquired or built to optimize conversion from the limited number of formats more easily than finding a conversion tool to handle the wider range of formats recommended in this document.

4.4 Relation to AGR 008

Contents of AGR 008

AGR 008 is entitled ”Digital Video." It includes a number of video recommendations, but only addresses two file formats. This document (and AGR 007) refers to 5 formats for motion data.

The two formats recommended in AGR 008 are:

AVI	Microsoft's architecture for video in Windows environments.
QuickTime	Apple's architecture for video in Windows and Macintosh

As described in the section relating this document to AGR 004, the larger number of formats can result in more efficient production of CBT and reuse of media elements within a company. The more restrictive set of formats is appropriate for moving motion data between companies.

Codecs

Codec refers to Compresion-Decompression algorithms for digital video. There are no specific Codecs recommended in this document or AGR 007. Refer to AGR 008 and the AICC's "Digital Video" technical paper for general codec recommendations.

5.0 COMPLIANCE REQUIREMENTS FOR CBT INTERCHANGE

Minimum Compliance Level-1

At a minimum, an authoring system must be able to do the following :

1. Export, in an automated fashion, all Graphic, Audio, Text, and Frame-based elements (as they are described on 3.0) from a lesson.

2. Export, in an automated fashion, an ASCII text file containing a list of all Graphic, Audio, Text, and Frame-based elements in a lesson including identification information and their filenames.

Basic Compliance Level-2

In addition to Level-1 requirements, an authoring system must be able to do the following for Level-2 Compliance:

- Import all Graphic, Audio, Text, and Frame-based elements (as they are described in section 3.0).

Intermediate Compliance Level-3

In addition to Level-2 requirements, an authoring system must be able to do the following for Level-3 Compliance:

- Export, in an automated fashion, a ASCII text file(s) containing a description of lesson behavior in logic files (as they are described in section 3.0).

Comprehensive Compliance Level-4

In addition to Level-3 requirements, an authoring system must be able to do the following for Level-4 Compliance:

- Import, in an automated fashion, the logic contained in ASCII text files (as described in section 3.0) and all accompanying Graphic, Audio, Text, and Frame-based elements (as they are described in section 3.0).

5.1 Media Element Storage Scenarios

Scenarios

The following are three descriptions of how multimedia elements may be handled by different authoring systems. In each case there is a description of what must be done to meet level 1 compliance.

5.1.1 Completely internalized Media Elements

Scenario 1

The media elements are imported into the authoring system and converted to that authoring system's own proprietary format.

Authoring systems such as WISE, PCD3, Authorware, and VACBI use this method.

Compliance

Level 1 compliance features be must provided as part of the authoring system or its associated tools. These tools would

- Extract each multimedia element embedded in proprieatry formats
- Convert each extracted element to an industry standard format
- Write each standard element to an industry standard file
- Print an ASCII text file with
  - ◊ a list of file names containing the media elements
  - ◊ the format of each media element and file -- TIFF uncompressed, PICT, CGM etc.
  - ◊ identification of location in the lesson -- for instance a frame number or icon name that appears in the lesson code.

5.1.2 Runtime Reference to Media Elements in Files

Scenario 2

In this scenario, ALL of the media elements for a lesson already reside in external, standard format, data files (as Described in section 3.0). The CBT runtime references (or "plays") these files at runtime.

Authoring Systems such as Icon Author and Quest use this method.

Compliance

If ALL the media elements are in such files and the file formats already are as specified in Section 3.0, the authoring system vendor would only need to print an ASCII Text file containing a list of these file names and their indentification within a lesson to meet the requirements for level 1.

If any of the media elements in a lesson reside in the authoring system's proprietary file format, then level 1 compliance requires that these embedded elements be removable with a tool provided as part of the authoring system.

5.1.3 Runtime Reference to Media Elements in a Central Database

Scenario 3

In this scenario, the authoring system uses a separate DBMS to hold media elements that are retrieved as required. The media elements are not stored as separate files, but as binary records in the database file format.

Compliance

In order to meet the export requirements for level 1, the DBMS used must be able to export its media data in standard formats described in level 1. If media elements reside in both the authoring system AND the DBMS, the authoring system must also be able to support the export functionality described in level 1.

5.1.4 "Compiler" Type Authoring

Scenario 4

In this scenario, the authoring system compiles lessons from an ASCII text file defining lesson organization and series of media element files in standard formats. When compiled, the system may internalize the external files, or just refer to them at runtime.

In some cases (such as the Tutor -- PLATO -- language) there may be no separate graphics or animation files. The system may use clear text drawing commands to create on-screen graphics and animations. After the code (including the drawing commands) is compiled, the system draws the graphic/animation at execution time.

Compliance

Since the authoring "source" contains a comprehensive description of the lesson in a form that can be understood by a person with knowledge of the programming language, such a "compiler" type authoring system is level 4 compliant.

6.0 COURSEWARE INTERCHANGE PROCESS

The interchange process would be as follows

Export

1. Export the basic elements to individual files
2. Export a listing of all the basic element files.
3. Export the complex elements to ASCII text representations

Import

1. Import or reference[6] the basic elements from individual files
2. Either manually or by means of a conversion program convert the logic text from the source authoring system's format to the target authoring system's format.

[6] In some authoring systems, basic elements are already in separate files. These files, in industry standard formats, are merely referenced at run-time. They do not necessarily have to be “imported” or internalized in the lesson code.

Appendix A
Examples Of Interchange Format

A.1 AUTHORWARE INTERCHANGE EXAMPLE

The following is an example to illustrate how the "Export" might work:

The "Page-Turner" Lesson.

This is a very simple lesson with foward & backward buttons for lesson navigaton. There are 3 frames or "pages" in the example:

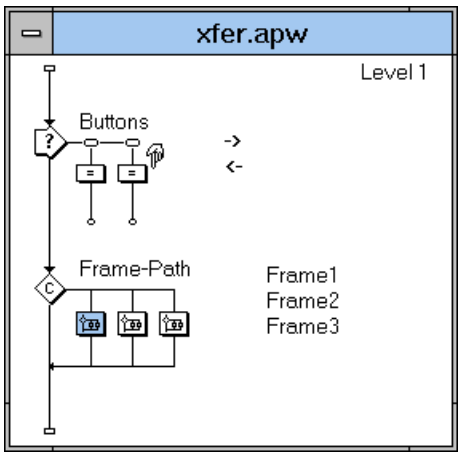
Frame1 (1 Bitmap Graphic,1 Vector Graphic, and 1 Audio String)
Frame2 (1 Vector Graphic and 1 Audio String)
Frame3 (1 Bitmap Graphic, 1 Digital Video Clip, and 1 Audio String)

Note: The following is an example of how the guidelines for CBT interchange *could be* implemented in Authorware Professional (AWP). AWP does not now have this capability, nor has Macromedia made any commitments to support such features in the future.

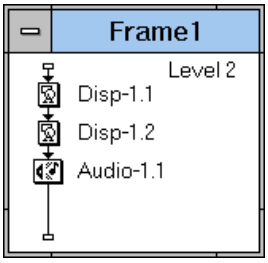
The following example is provided only to help illustrate the concepts described in this guideline.

Authorware Example

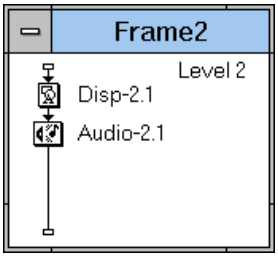
The page-turner would might look like this in Authoware :



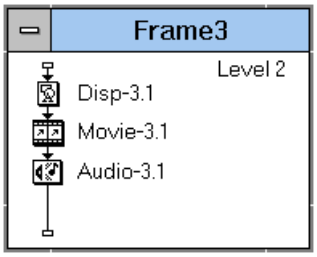
"Page-Turner" Example (Authorware - Top-Level View)



"Page-Turner" Example (Frame 1)



"Page-Turner" Example (Frame 2)



"Page-Turner" Example (Frame 3)

Files Exported

File name	Element Type	Data Format
XFER.TXT	Logic	Clear Text File
0010.BMP	Bitmap Graphic	Windows BMP format
0011.WMF	Vector Graphic	Windows Meta-File WMF format
0012.WAV	Audio	WAV format
0013.BMP	Bitmap Graphic	Windows BMP format
0014.WAV	Audio	WAV format
0015.BMP	Bitmap Graphic	Windows BMP format
0016.AVI	Digital Video Clip	Windows AVI format
0017.WAV	Audio	WAV format

ASCII-Text Logic File (XFER.TXT)

```
VARIABLES
Begin
    Path = 1;
End

Icon ID:0001;Title:"Buttons";Type:Interaction
Begin
    Path1(BUTTON;PERPETUAL;STANDARD;SIZE:40,50;LOC:470,30;ICONID:0002)
    Path2(BUTTON;PERPETUAL;STANDARD;SIZE:40,50;LOC:470,60;ICONID:0003)
End

Icon ID:0004;Title:"->";Type:Calc
Begin
    Path := Path + 1;
    if(Path>Pathcount(0006),Path:=Path-1);
End

Icon ID:0005;Title:"<-";Type:Calc
Begin
    Path := Path - 1;
```

```
    if(Path<1,Path:=1);
End

Icon ID:0006;Title:"Frame-Path";Type:Decision
Begin
    Path1(GROUP;NOERASE;ICONID:0007)
    Path2(GROUP;NOERASE;ICONID:0008)
    Path3(GROUP;NOERASE;ICONID:0009)
End

Icon ID:0007;Title:"Frame1";Type:Group
Begin
    MEMBERS(0010,0011,0012)
End

Icon ID:0008;Title:"Frame2";Type:Group
Begin
    MEMBERS(0013,0014)
End

Icon ID:0009;Title:"Frame3";Type:Group
Begin
    MEMBERS(0015,0016,0017)
End

Icon ID:0010;Title:"Displ-1.1";Type:Display
Begin
    File("0010.BMP")
    Position(10,20)
End

Icon ID:0011;Title:"Displ-1.2";Type:Display
Begin
    File("0011.WMF")
    Position(40,20)
    Effects(None)
End

Icon ID:0012;Title:"Displ-1.2";Type:Audio
Begin
    File("0012.WAV")
    Play(Once,Wait Until Done)
End




Icon ID:0013;Title:"Displ-2.1";Type:Display
Begin
    File("0013.BMP")
```

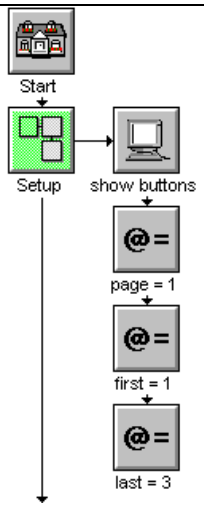
```
    Position(40,20)
    Write("PATH = "^{Path},100,50,Arial,24pt);
    Effects(None)
End

Icon ID:0014;Title:"Audio-2.1";Type:Audio
Begin
    File("0014.WAV")
    Play(Once,Wait Until Done)
End

Icon ID:0015;Title:"Displ-3.1";Type:Display
Begin
    File("0015.BMP")
    Position(40,20)
    Write("PATH = "^{Path},100,50,Arial,24pt);
    Effects(None)
End

Icon ID:0016;Title:"Audio-3.1";Type:Audio
Begin
    File("0016.WAV")
    Play(Once,Wait Until Done)
End

Icon ID:0017;Title:"Movie-3.1";Type:Movie
Begin
    File("0017.AVI")
    Play(Once,Wait Until Done,10 Fps, Speed)
End
```

A.2 ICONAUTHOR INTERCHANGE EXAMPLE

IconAuthor Example

A sample "page flipper" application in IconAuthor could be built as follows:

The first part of the application uses the SmartObject editor to create three buttons and set some variable values that define the current page, first page and last page as depicted in figure 1.



Show a SmartObject page called buttons as shown in figure 2. The back button is initially invisible.

Start on the first page by setting the variable @page equal to 1.

Set the first page to one by setting the variable @page_first equal to 1.

Set the last page to three by setting the variable @page_last to 3.

figure 1 - the first section of the "page flipper" application

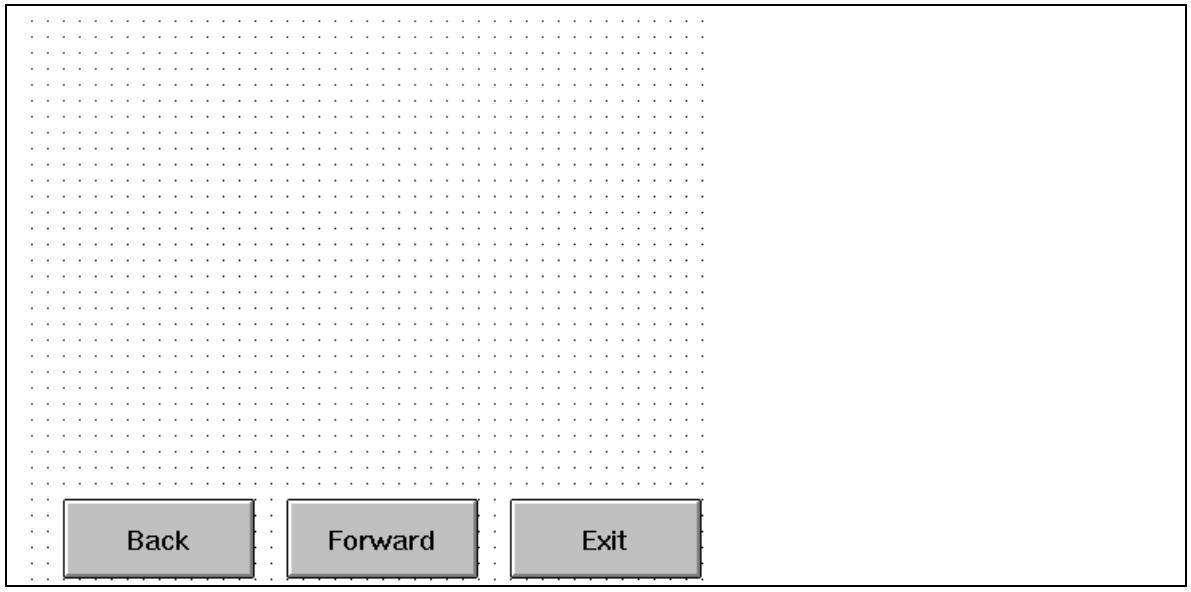


figure 2 - the buttons used to control the "page flipping"

The main section of the application, shown in figure 3, enters into a loop. At the start of the loop the current page is displayed and the application waits for events to occur. If the forward button is selected, then all of the objects of the old page are deleted. The page number is then incremented. A check is made to see if this is the last page (@page equal to @page_last). If they are equal then the forward button is made invisible. The last part of the branch makes the back button visible. Similar processing is done for the back button. The loop is exited if the exit button is pressed.

Figure 4, 5 and 6 show sample pages used by the application but the general nature of the design could incorporate any use of media and data on these pages.

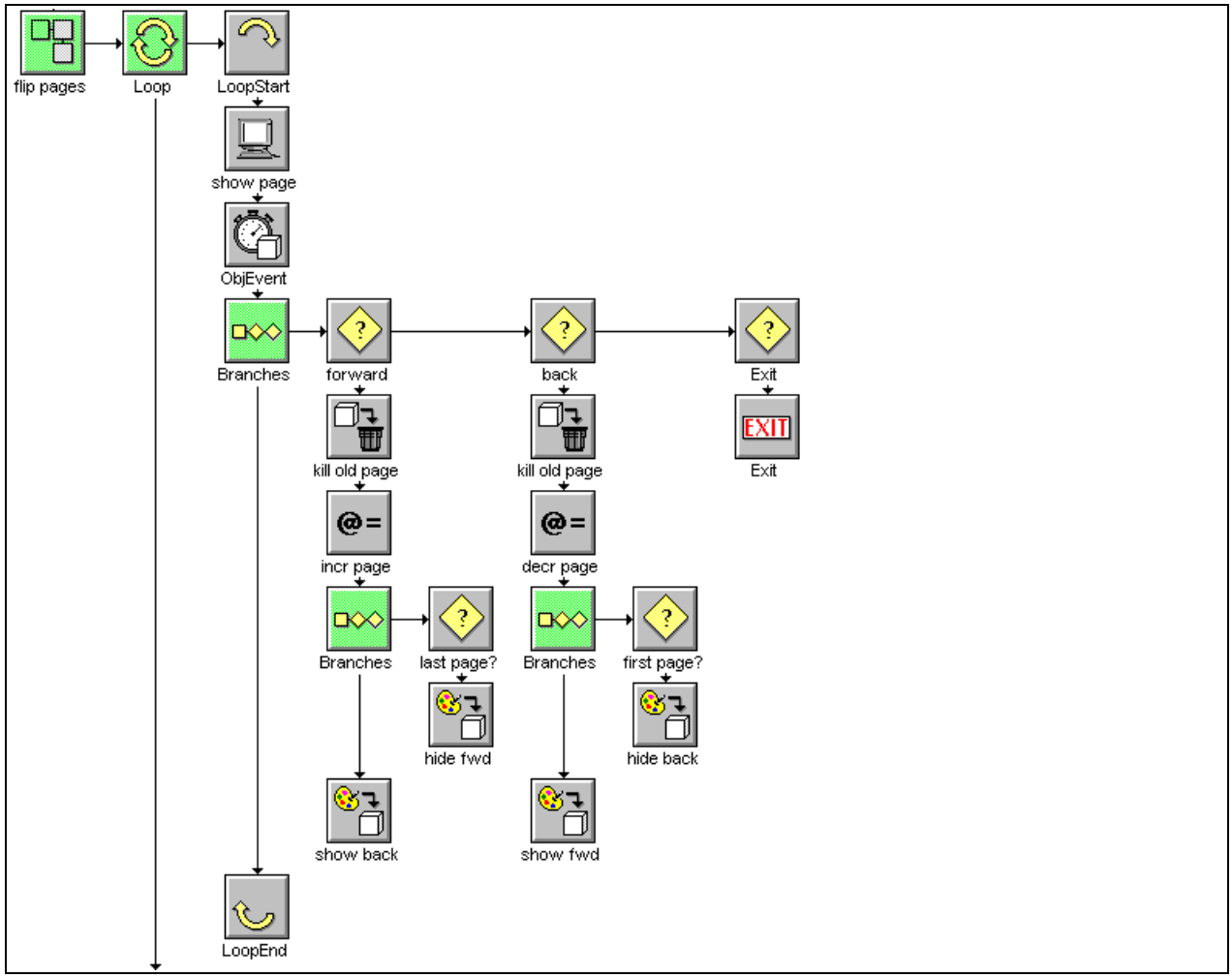


figure 3 - the main section of a "page flipper" application

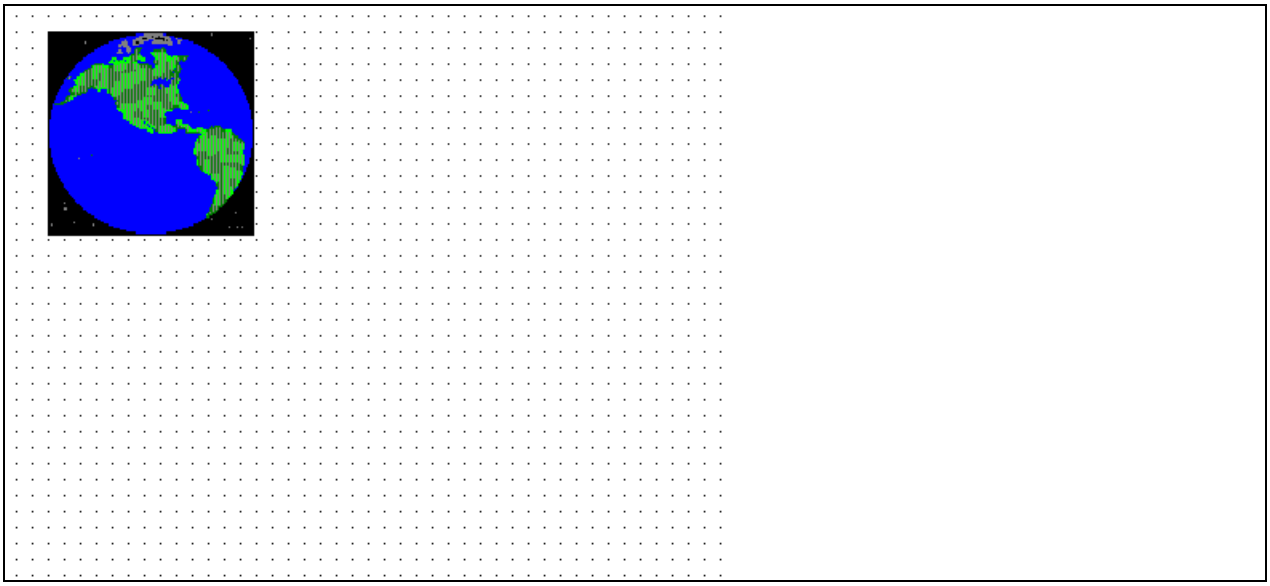


figure 4 - page 1 of this sample application

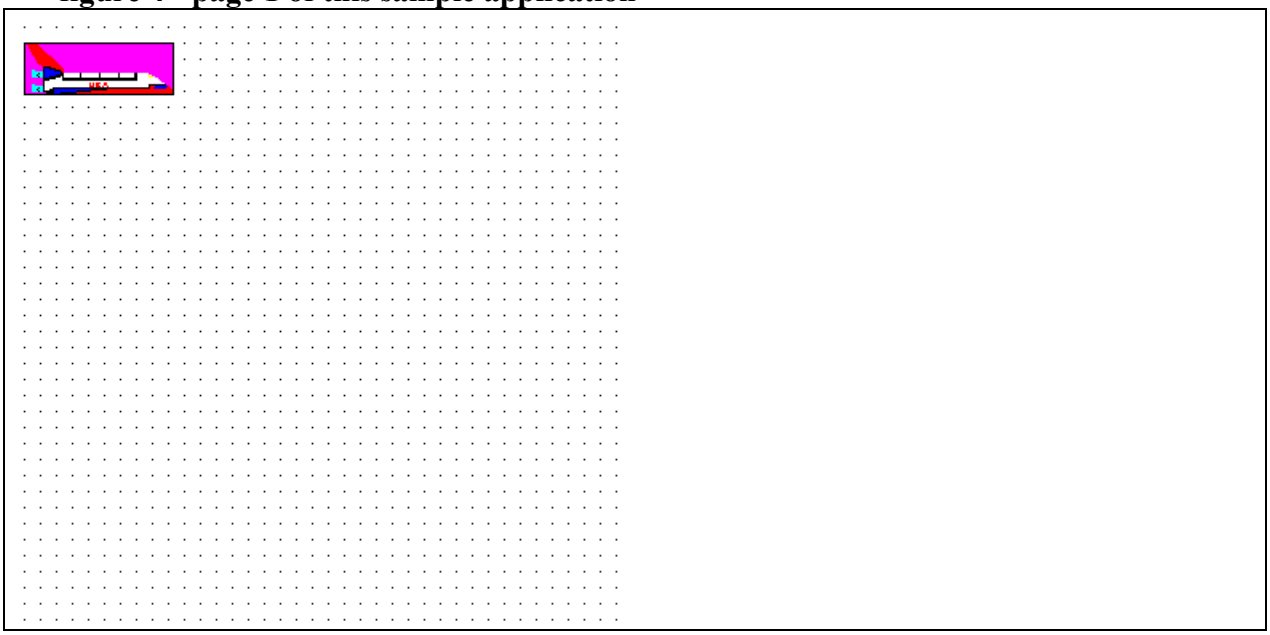


figure 5 - Page 2 of this sample application

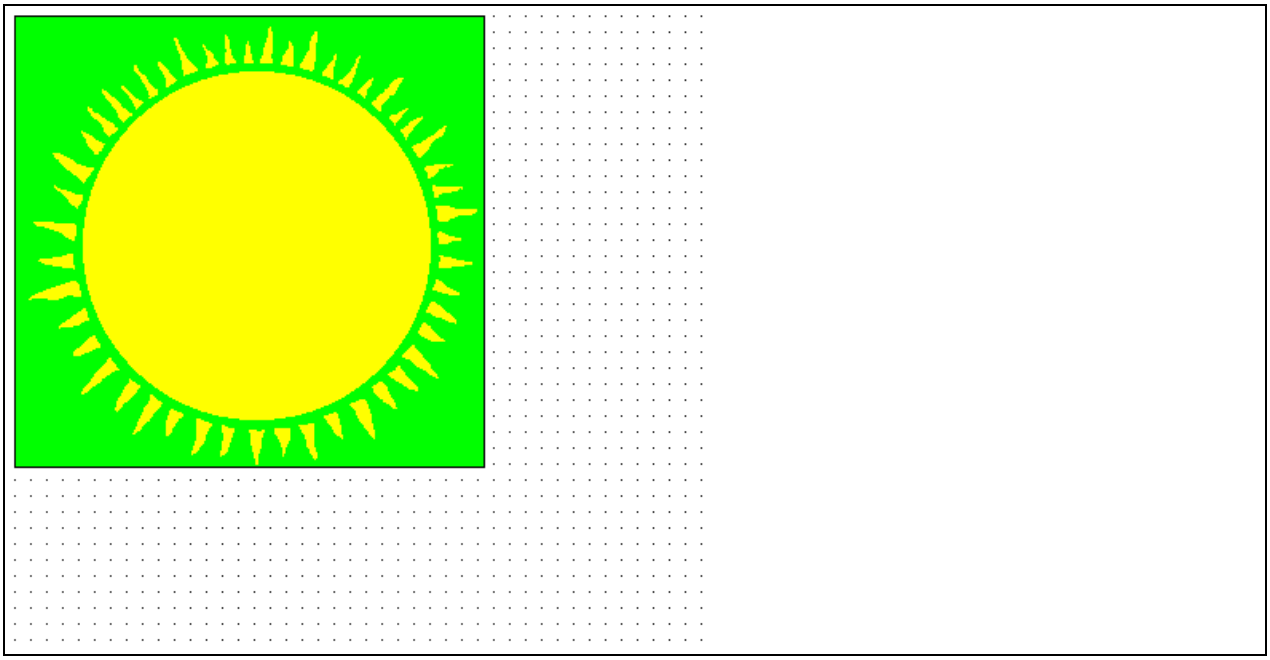


figure 6 - Page 3 of this sample application

The current release of IconAuthor supports common media on all platforms (Windows, Macintosh, OS/2 and Unix). Older releases of IconAuthor required a conversion program to move applications from one platform to another. The conversion program was used to generate a text file that represents the this sample "page flipper" application. IconAuthor does not have to export media files since all media files are accessed in their existing industry standard formats. The rest of the application is represented in three output files that represent the fonts, application logic (the application flowchart), and the SmartObject files. The output files of this sample are shown in figures 7, 8, and 9.

```
@FONT_MAP_SRC
FONTMAP 4.1 (MSDOS)
@LOGFONT_CNT
1,0,24,0,25,49
@LOGFONT[0]
System,7,16,700,0,0,34,0,0,0
```

figure 7 - the font information file

```
80 ICONWARE_PORTABLE 0100 19950301150354 00 0320 0258 00FF 0000001E *******
******************************************************
0 0 4 2 2 2 0 0 0 4 2 0 4 4 4 2
2 2 0 4 4 4 2 2 0 4 4 4 0 2
22 28 5 Start FFFE 2 FFFE FFFF FFFF 0 0
2 8 5 Setup 1 7 FFFE FFFF 3 10000 0
22 B C show buttons FFFF 4 2 FFFF FFFF 10001 4
800 B SmartObject
800 8 AICC.SMT
800 4 0, 0
800 7 buttons
22 2C 8 page = 1 3 5 FFFF FFFF FFFF 20001 2
400 5 @page
800 2 1.
22 2C 9 first = 1 4 6 FFFE FFFF FFFF 30001 2
400 B @page_first
800 2 1.
22 2C 8 last = 1 5 FFFE FFFE FFFF FFFF 40001 2
400 A @page_last
800 2 3.
2 8 7 ObjMenu 2 FFFE FFFE FFFF 8 50000 0
10 18 4 Loop FFFE 1E 7 FFFF 9 50001 0
22 1B 9 LoopStart FFFF A 8 FFFF FFFF 50002 0
22 B 7 Display 9 B FFFE FFFF FFFF 60002 4
800 B SmartObject
800 8 AICC.SMT
800 4 0, 0
400 5 @page
22 4B 8 ObjEvent A C FFFF FFFF FFFF 70002 2
800 4 Wait
800 1 0
2 4 8 Branches B 1D FFFF FFFF D 80002 0
22 12 7 forward FFFF 17 C E FFFF 80003 4
400 D @_OBJECT_NAME
8000 2 EQ
800 7 forward
800 8 alphabet
22 12 4 back FFFF 11 D F FFFF 80005 4
400 D @_OBJECT_NAME
8000 2 EQ
800 4 back
800 8 alphabet
22 12 4 Exit FFFF 10 E FFFE FFFF 80007 4
400 D @_OBJECT_NAME
8000 2 EQ
800 4 exit
800 8 alphabet
22 D 4 Exit F FFFF FFFF FFFF FFFF 90007 2
800 4 loop
800 0
22 4A D kill old page E 12 FFFE FFFF FFFF 90005 2
800 6 Family
400 5 @page
22 2C 9 decr page 11 13 FFFE FFFF FFFF A0005 2
400 5 @page
800 A @PAGE - 1.
2 4 8 Branches 12 16 FFFE FFFF 14 B0005 0
22 12 B first page? FFFF 15 13 FFFE FFFF B0006 4
```

```
400 5 @page
8000 2 EQ
400 B @page_first
800 8 alphabet
22 48 9 hide back 14 FFFF FFFE FFFF FFFF C0006 4
800 6 Object
800 4 back
800 7 visible
800 5 false
22 48 8 show fwd 13 FFFF FFFE FFFF FFFF D0005 4
800 6 Object
800 7 forward
800 7 visible
800 4 true
22 4A D kill old page D 18 FFFF FFFF FFFF 90003 2
800 6 Family
400 5 @page
22 2C 9 incr page 17 19 FFFF FFFF FFFF A0003 2
400 5 @page
800 A @PAGE + 1.
2 4 8 Branches 18 1C FFFF FFFF 1A B0003 0
22 12 A last page? FFFF 1B 19 FFFE FFFF B0004 4
400 5 @page
8000 2 EQ
400 A @page_last
800 8 alphabet
22 48 8 hide fwd 1A FFFF FFFF FFFF FFFF C0004 4
800 6 Object
800 7 forward
800 7 visible
800 5 false
22 48 9 show back 19 FFFF FFFE FFFF FFFF D0003 4
800 6 Object
800 4 back
800 7 visible
800 4 true
22 19 7 LoopEnd C FFFF FFFF 9 FFFF E0002 0
22 4A 9 ObjDelete 8 FFFE FFFF FFFF FFFF F0001 2
800 3 All
800 0
```

figure 8 - the flowchart information file

```
80 SMARTTEXT_PORTABLE 0102 19950301150354 00 7D00 7D00 00FF **************
**************************************************
1 7 buttons
3 0 0 0 7FFF 7FFF 50000000 0
1 1 FFFF FFFF FFFF 1 1 FFFF FFFF FFFF 25 83
8 0 199 4887 FFF AAA 50010000 0
65 1 101000000000000
1 0 0 0 0 0 C0C0 C0C0 C0C0 0
4 back 0   4 Back 0   0   88
8 0 24CC 4887 FFF AAA 50010000 0
65 1 101000000000001
1 0 0 0 0 0 C0C0 C0C0 C0C0 0
4 exit 0   4 Exit 0   0   88
8 0 1333 4887 FFF AAA 50010000 0
65 1 101000000000001
1 0 0 0 0 0 C0C0 C0C0 C0C0 0
7 forward 0   7 Forward 0   0   88
81 1 1 1
3 0 0 0 7FFF 7FFF 50000000 0
1 1 FFFF FFFF FFFF 1 1 FFFF FFFF FFFF 30 83
4 0 333 444 14A3 1B4E 50000000 0
EDDA
65 6A 0 11000000101
0 0   1 1 20 C:\5CIA_50036\5CGRAPHICS\5CEARTHSM.PCX 0
6A FF FF 0 20 C:\5CIA_50036\5CGRAPHICS\5CEARTHSM.PCX 84
81 1 1 2
3 0 0 0 7FFF 7FFF 40000000 0
1 1 FFFF FFFF FFFF 1 1 FFFF FFFF FFFF 6 83
4 0 4CC 666 F0A 70A 40000000 0
EDDA
65 6A 0 11000000101
0 0   1 2 21 C:\5CIA_50036\5CGRAPHICS\5CSHUTTLE1.PCX 0
6A FF FF 0 21 C:\5CIA_50036\5CGRAPHICS\5CSHUTTLE1.PCX 84
81 1 1 3
3 0 0 0 7FFF 7FFF 40000000 0
1 1 FFFF FFFF FFFF 1 1 FFFF FFFF FFFF 6 83
4 0 199 221 2F32 3C5E 40000000 0
EDDA
65 6A 0 11000000101
0 0   1 3 1C C:\5CIA_50036\5CGRAPHICS\5CSUN.BMP 0
6A FF FF 0 1C C:\5CIA_50036\5CGRAPHICS\5CSUN.BMP 84
81 FF
```

figure 9 - the SmartObject information file

Glossary

ADPCM	Adaptive Delta Pulse Code Modulation. Also known as Adaptive Differential Pulse Code Modulation. A technique for encoding digitized audio. It generally requires about one half the number of bytes required for PCM encoding. As a result, ADPCM is frequently called compressed audio. The resulting audio quality is generally degraded by less than 15% compared to PCM with the same sample rate. With all audio digitizing techniques, the waveform is sampled X times per second -- a typical sample rate would be 8000 times per second (8 KHz). With ADPCM each sample represents a change in the amplitude of the signal. With PCM, each sample represents the entire amplitude rather than the change from the last sample.
AIFF	Audio Interchange File Format. Digital audio file format originally developed for Macintosh audio.
ATA	Air Transport Association. An international organization of airlines headquartered in Washington, D.C., USA.
Audio	In the context of this guideline, audio refers to sound not embedded in a frame-based motion sequence. Audio may include verbal information, music, and sound effects. Audio may be produced from digital files, or attached audio device like a CD player or video disc player.
AVI	Audio Video Interleaved. Microsoft's standard format for digital video and synchronized audio. Also called the Video for Windows format.

basic elements	Basic elements are those portions of a lesson's content that can be stored in a well defined, widely accepted format -- a current defacto standard. The basic elements are text, audio, graphic, and frame-based motion.
BMP	See *DIB*.
CGM	Computer Graphics Metafile. This is an ANSII and ISO standard file format for storing 2 dimensional picture descriptions in a vector format. The descriptions are device-independent -- that is, with the appropriate display software they may be displayed on any monitor at any resolution.
complex elements	Complex elements are those portions of a lesson's content that cannot be defined in an existing industry standard format. These are parts of a lesson that require special (and currently unique) descriptions.
composite	A special logic definition that uses text to create a combination of basic elements. For instance, a composite could describe how to move a bitmap across the screen -- that is, create a path animation.
DIB	Device independent bitmap. These are bitmaps designed for use in the Microsoft Windows environment. They may have three file name extensions: .BMP, .DIB, .RLE. .BMP and .DIB files have the same internal format. .RLE files are run length encoded -- that is compressed. In addition to the extension, the file header indicates whether the file is compressed.
FLIC/FLI	Format for storing cel-based animations established by AutoDesk for their Animator product. This format does not include audio.

frame-based motion	Frame-Based motion elements are sequences in the lesson in which the student percieves motion that has been produced by a rapidly presented series of pictures. Frame-based motion may be produced from digital video, cel animations, and attached video discs.
graphic	A graphic is that part of a still image that the student can see on screen but which is not contained in a separate logic or text file.
logic	All Logic files are lesson behavior descriptions written in a plain ASCII text. These descriptions could be in the form of a programming language, a scripting language, or even an SGML type of file. Types of lesson behavior described in the logic file include: _ Student navigation _ Answer judging _ Conditional branching _ Combining basic elements _ Sequencing basic elements _ Writing CMI data to a file
MPEG	Motion Picture Experts Group. Name of the organization that originally selected the digital video compression algorithm and a file format for encoded video that evolved into the current ANSI and ISO standards. Frequently the standards themselves are referred to as MPEG. Currently there are two ISO compression standards for digital video. They are referred to as MPEG 1 and MPEG 2
PCM	Pulse Code Modulation. A technique for digitizing audio. (Using fixed frequency sampling and linear quantization.) The quality of the sound depends on how frequently the audio waveform is sampled (sample rate) and how many bits are used to record each sample (sample size). CD's use this technique for digitizing audio. In a high fidelity CD recording, about 44,000 samples are taken of the sound each second and 16 bits are used to describe each sample.

PICS	Format for storing cel-animations as a series of PICT bitmaps. Originated in the Apple Macintosh environment.
QuickTime	Apple Computer's format for storing digital video and synchronized audio in a file.
re-purpose	Use some portion of the content of a computer program for a different purpose than that for which it was designed. An example would be using part of a CBT lesson in a performance support system.
re-use	Use some or all of the content of a computer program again, in a different program. The second use is for about the same purpose as the first use. An example would be re-using a section of a lesson on the fundamentals of painting aircraft. This lesson was designed for a 737 course. When the same section is used in a lesson in a 757 course, it is re-use of the section.
RLE	See *DIB*.
Text	This is text associated with a lesson that is not embedded in a graphic or frame-based motion sequence. Typically it includes the audio script and explanitory text that appears on screen in tutorials (but not call-outs and titles that appear in a graphic.)
TIFF	Tagged Image File Format. A format for storing raster or bit mapped graphics. Developed jointly by Aldus and Microsoft, TIFF was designed to be independent of operating systems and processors. It allows a graphic to be encoded in several different ways. The encoding method is identified by a tag in the file. Programs supporting TIFF need to be able to decode many different bit-map encoding techniques.

PICT	Picture file. File format for storing vector and bit-map graphics. This was the first format designed for the Macintosh at its introduction. It has evolved and is currently the most popular format in the Macintosh environment.
WAVE	File format defined by Microsoft for digital audio used in the Windows environment. Different sample rates and sample sizes can be stored in WAVE files.
WMF	Windows MetaFile. Microsoft developed format designed for storing vector graphics in the MS Windows environment.

Index